Algorithmes pour la segmentation et l'amélioration de la qualité des images et des vidéos

Pascal Bertolino

HABILITATION À DIRIGER DES RECHERCHES

préparée au GIPSA-lab

présentée à

l'Université de Grenoble

Spécialité
Informatique

soutenue par

Pascal Bertolino
Maître de Conférences à l'IUT2 de l'Université Pierre Mendès France, Grenoble

le 24 février 2012

Titre

Algorithmes pour la segmentation et l'amélioration de la qualité des images et des vidéos

Jury

Mme Jenny Benois-Pineau	Rapporteur	Professeur à l'Université de Bordeaux I
Mr Fabrice Mériaudeau	Rapporteur	Professeur à l'Université de Bourgogne
Mr Jean-Marc Chassery	Rapporteur	Directeur de Recherche au CNRS
Mr Roger Mohr	Examinateur	Professeur Emérite à Grenoble INP
Mr Andrea Cavallaro	Examinateur	Professeur à la Queen Mary University of London

Table des matières

Chapitre 1

Résumé des activités

Sommaire

1.1 Centres d'intérêt

J'ai rejoint mon équipe de recherche actuelle AGPIG (Architecture, Géométrie, Perception, Images, Gestes) du Département Images et Signal du laboratoire GIPSA-lab (Grenoble Images, Parole, Signal et Automatique) en 1998 peu après ma nomination comme Maître de Conférences. Le laboratoire s'appelait alors LIS (Laboratoire des Images et des Signaux).

Jusque là je m'étais essentiellement intéressé à la segmentation d'images fixes. Depuis mon arrivée au LIS/GIPSA, mes travaux ont continué pour une partie dans cette voie, notamment en segmentation supervisée. Depuis, je m'intéresse également beaucoup au suivi d'objets dans les séquences vidéos que ce soit dans le cas particulier de vidéos à caméra fixe ou dans le cas général de vidéos à caméra mobile.

Dans le but de m'orienter vers le haut niveau et l'interprétation de contenu, j'ai abordé le problème de la structuration du contenu des vidéos et plus particulièrement de l'extraction d'objets clés puis la segmentation automatique de personnes.

Depuis six années, je m'investis aussi dans des problématiques d'amélioration de la qualité d'images sur les écrans plats en collaboration avec ST Microelectronics. Ces dernières années, j'ai entrepris un transfert technologique d'une partie de mes travaux devant déboucher sur la création d'une startup.

Dans ce premier chapitre, je présente un résumé de mes activités. Il indique mon parcours et ma production en tant qu'enseignant-chercheur. Les chapitres suivants seront consacrés aux différents thèmes de recherche dans lesquels je m'investis.

1.2 Curriculum vitae

1.2.1 Etat civil

Pascal Bertolino
né le 18 septembre 1961 à Voiron (Isère)
Marié, père de 5 enfants

Coordonnées personnelles :
42 avenue Hector Berlioz, 38170 Seyssinet Pariset
Tél. : 04 76 09 22 71

Adresse professionnelle 1 :
GIPSA-lab,
ENSIEG, 961 rue de la Houille Blanche, BP 46
38402 St. Martin d'Hères cedex
Tél. : 04 76 57 43 60
Fax : 04 76 82 63 84
Email : pascal.bertolino@gipsa-lab.grenoble-inp.fr
Web : http://www.gipsa-lab.inpg.fr/~pascal.bertolino/

Adresse professionnelle 2 :
Département Informatique,
IUT2, 2 place Doyen Gosse
38031 Grenoble cedex
Tél. : 04 76 28 45 85

Situation actuelle :
Maître de conférences en Informatique au département informatique de l'IUT2 de l'Universtié Pierre Mendès France (Université Grenoble II).
Chercheur dans l'équipe *AGIIG* de GIPSA-lab de Grenoble.
Titulaire d'une prime d'encadrement doctoral et de recherche pour la période 2004-2012.

1.2.2 Parcours

Je présente ci-dessous mon parcours qui débute par plusieurs sociétés du secteur privé. Il est quelque peu atypique ce qui explique sans doute pourquoi je m'intéresse à la valorisation de la recherche dans l'industrie. En gras, le lecteur trouvera mon cursus universitaire imbriqué à mon parcours professionnel, qui explique l'évolution de ma carrière vers le milieu académique, à l'age de 37 ans.

BTS d'informatique	Grenoble	(85)
Programmeur	Kiss France (38)	(85-86)
Analyste programmeur	LOGIC (38)	(86-89)
DPCT de génie informatique	CNAM	(87)
Responsable de projets	Différence Diffusion (38)	(89-90)
DEST de génie informatique	CNAM	(90)
Responsable de projets	JCD Ingénierie (38)	(90-91)
Diplôme d'ingénieur en informatique	CNAM	(92)
Chercheur contractuel[1]	lab. TIMC	(92-94)
Master Image, Vision et Robotique	INP Grenoble	(93)
Ingénieur R & D (tps partiel)	Cloé Technologies (73)	(93-94)
ATER	Université Grenoble II, lab. TIMC	(94-96)
Doctorat en informatique	INP Grenoble, lab. TIMC	(95)
Chercheur contractuel[2]	lab. TIMC	(96-97)
Ingénieur expert	INRIA Rhône-Alpes, équipe MOVI	(97-98)
Maître de conférences en informatique	Université Grenoble II, lab. LIS/GIPSA	(depuis 98)

[1] Projets européens de bio-technologie HOME et IMPACT

[2] Projet européen de télémédecine EUROPATH

1.3 Encadrement de masters

J'ai encadré 13 stages de master. Une majorité concernent le master **SIPT** (Signal, Image, Parole et Télécommunications) et sont des stages de double cursus (master et diplôme d'ingénieur). En voici la liste par ordre chronologique :

- **Stéphane Ribas** (1998)
  Stage de Master in Multimedia Systems and Technology, University of Surrey
  Segmenting digital image sequences using the irregular pyramid
- **Nicolas Enderlé** (1999)
  Stage de Master SIP - 3ème année ENSIEG, Grenoble, co-encadré avec **Denis Pellerin** du LIS
  Segmentation d'images spatio-temporelle à base de pyramide irrégulière et de filtres de Gabor
- **Guillaume Foret** (2000)
  Stage de Master SIPT - 3ème année ENSERG, Grenoble,
  Segmentation spatio-temporelle d'objets vidéo
- **Stéphane Nicolau** (2000)
  Stage de Master Photonique et Traitement d'Images - 3ème année ENSP, Strasbourg
  Détection de contours stochastique dans des images numériques
- **David Cibaud** (2001)
  Stage de Master SIPT - 3ème année ENSERG, Grenoble,
  Continuité temporelle en segmentation de séquences vidéo
- **Kitty Huot** (2002)
  Stage de Master SIPT - 3ème année ENSIEG, Grenoble,
  Réactualisation non supervisée de l'image de référence dans une séquence vidéo
- **Simon Conseil** (2003)
  Stage de Master SIPT - 3ème année ENSIEG, Grenoble,
  Modélisation d'histogramme par mélange de gaussiennes pour la segmentation de séquences vidéo
- **Jérémy Huart** (2003)
  Stage de Master SIPT - 3ème année Polytech, Grenoble,
  Segmentation locale pour l'extraction d'objets dans les images et les séquences vidéo
- **Xavier Pol** (2004)
  Stage de Master Optique Image et Vision - 3ème année ISTASE, St Etienne
  Extraction d'objets en mouvement dans les séquences vidéo

- **Sendiren Manikkam** (2004)
  Stage de Master Compétences Complémentaires en Informatique, Université Joseph Fourier, Grenoble,
  Suivi de zone d'intérêt dans les vidéos aériennes
- **Tien Sy Nguyen** (2006)
  Stage de Master SIPT, Grenoble, co-encadré avec **Cédric Gérot** du LIS
  Modélisation de partition d'images 2D par des patchs de relief
- **Abdul Baseer** (2007)
  Stage de Master SIPT
  Codage et synthèse d'image 2D par représentation en ondelettes d'images segmentées
- **Cyril Migniot** (2008)
  Stage de Master SIPT - 3ème année ENSIEG, Grenoble,
  Gestion du canal alpha pour le rendu des contours d'objets segmentés et des objets semi-transparents

La figure 1.1 présente une vue chronologique de cet encadrement.

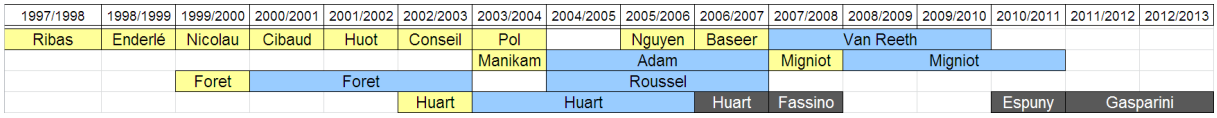


FIGURE 1.1 – Récapitulatif chronologique des encadrements. En jaune (blanc) les masters ; en bleu (gris) les thèses ; en noir les post-doc

1.4 Encadrement de doctorants et post-doctorant

J'ai à mon actif l'encadrement ou co-encadrement de 6 thèses : j'ai encadré seul deux thèses (grâce à des agréments du conseil du collège doctoral) et co-encadré trois autres thèses qui ont été soutenues. Enfin, je co-encadre actuellement une autre thèse qui est en cours. Ces six thèses dépendent toutes de l'école doctorale EEATS (Électronique, Électrotechnique, Automatique, Télécommunications, Signal), commune à l'INP Grenoble et à l'Université Joseph Fourier. La figure 1.1 montre l'imbrication entre mes encadrements de master et de thèses.

1.4.1 Thèses soutenues

- **Guillaume Foret**
  Segmentation spatio-temporelle d'objets vidéo en vue de leur caractérisation
  Co-encadrement (80%) avec Jean-Marc Chassery
  Soutenue le 17 octobre 2003
  Depuis la fin de sa thèse, Guillaume Foret a été ingénieur R&D à SAGEM puis à Streamezzo, leader dans les applications pour la téléphonie mobile.

- **Jérémy Huart**
  Extraction d'objets clés et analyse de leur comportement pour la structuration d'images et de vidéos
  Encadrement (100%, par agrément du conseil du collège doctoral EEATS)
  Soutenue le 14 février 2007
  Jérémy Huart a participé au transfert de technologie que j'ai réalisé pour la création d'une startup (voir section 1.10.3 page 17).

- **Pierre Adam**
  Améliorations d'artefacts sur panneaux LCD

Thèse CIFRE. Co-encadrement (80%) avec Jean-Marc Chassery
Soutenue le 15 juillet 2008
Pierre Adam a ensuite travaillé comme ingénieur pour la société Vesalis qui développe une technique pour réaliser du maquillage virtuel par réalité augmentée.

- **Jérôme Roussel**
  Systèmes de désentrelacement "haute performance" de signaux de télévision
  Encadrement (100%, par agrément du conseil du collège doctoral EEATS)
  Thèse CIFRE avec STMicroelectronics. La thèse s'est déroulée sans problème jusqu'à la fin *. Néanmoins, pour des raisons personnelles, Jérôme Roussel n'a pas finalisé la rédaction de son manuscrit et n'a pas soutenu. Il travaille toujours actuellement comme ingénieur à STMicroelectronics.

- **Eric Van Reeth**
  Système avancé d'interpolation spatiale de signaux de télévision, pour affichage sur écrans haute définition
  Thèse CIFRE. Co-encadrement (80%) avec Jean-Marc Chassery
  Soutenue le 10 mai 2011.
  Eric Van Reeth effectue actuellement un séjour post-doctoral à la Nanyang Technological University de Singapour, sur de l'amélioration d'images IRM.

1.4.2 Thèse en cours

- **Cyrille Migniot**, depuis septembre 2008.
  Détection et suivi de personnes dans les vidéos pour les effets spéciaux.
  Co-encadrement (80%) avec Jean-Marc Chassery
  Cette thèse est financée par une bourse du ministère de l'enseignement supérieur et de la recherche. Elle sera soutenue avant la fin 2011.

1.4.3 Post-doctorants

J'ai dirigé le travail de deux jeunes docteurs, sur un projet de transfert technologique, présenté en section 1.10.3 page 17. Dans le cadre du projet MOOV3D exposé en section 1.8.3 page 12, j'ai la direction de deux post-doctorants. Dans le cadre du projet ReadPlay qui vient juste d'être accepté, j'embaucherai un post-doctorant fin 2011, début 2012 :

- **Jérémy Huart**, projet de transfert technologique, 2007-2008, 15 mois.
- **Sylvain Fassino**, projet de transfert technologique, 2007-2008, 5 mois.
- **Ferran Espuny Pujol**, projet MOOV3D, 2010-2011, 12 mois.
- **Simone Gasparini**, projet MOOV3D, 2011-2013, 24 mois.
- **X**, projet ReadPlay, 2012, 12 mois.

1.5 Liste des publications

Voici les publications auxquelles j'ai participé depuis mon entrée dans le monde de la recherche. Le tableau 1.1 page 22 en fait la synthèse.

∗. 3 publications, 1 brevet

1.5.1 Thèse

- P. Bertolino, Contribution des pyramides irrégulières en segmentation d'images multirésolution, Institut National Polytechnique de Grenoble. Thèse préparée au Laboratoire TIMC-IMAG, Institut Albert Bonniot, La Tronche (Grenoble), 1995
  Composition du Jury :
  – Alain Chehikian, TIRF-INPG, Grenoble (président)
  – Annick Montanvert, ENS, Lyon (directeur de thèse)
  – Jean-Pierre Cocquerez, ENSEA, Cergy-Pontoise (rapporteur)
  – Jean-Michel Jolion, INSA, Lyon (rapporteur)
  – Philippe Bolon, Université de Savoie, Annecy
  – Jean-Marc Chassery, TIMC-IMAG, Grenoble

1.5.2 Revues internationales

1. P. Adam, P. Bertolino, F. Lebowsky, Mathematical modeling of the LCD response time, Journal of the Society for Information Display, 15, 8, p 571-577, 2007
2. E. Bertin, H. Bischof, P. Bertolino. Voronoi pyramid controlled by Hopfield networks. Computer Vision and Image Understanding, vol 63, No 3, pp 462-475, May 1996.
3. C. Sowter, P. Bertolino. Histometry and the home concept : An aid to the grading of intra-cervical neoplasia ? Analytical Cellular Pathology, 9(4) :269-279, December 1995.

1.5.3 Revue nationale

1. G. Foret, P. Bertolino, J-M. Chassery, Suivi d'objets vidéo par propagation d'étiquettes et rétro-projection, Traitement du Signal, Vol. 22(1), pp. 41-57, 2005

1.5.4 Conférences internationales

1. C. Migniot, P. Bertolino, J.-M. Chassery, Automatic people segmentation with a template-driven graph cut, IEEE ICIP, Brussels, Belgium, september 2011
2. E. Van Reeth, P. Bertolino, M. Nicolas, Image interpolation based on a multi-resolution directional map, S&T / SPIE Electronic Imaging, San Jose, USA, 2011
3. C. Migniot, P. Bertolino, J.-M. Chassery, Contour segment analysis for human silhouette pre-segmentation, International Conference on Computer Vision Theory and Applications, Angers, France, 2010
4. E. Van Reeth, P. Bertolino, M. Nicolas, J.-M. Chassery, Adaptive edge orientation analysis, S&T / SPIE Electronic Imaging, San Jose, USA, 2010
5. J. Roussel, P. Bertolino, Graph-based deinterlacing, IEEE ICIP, San Diego, USA, october 2008
6. J. Huart, P. Bertolino, A generic process chain to extract key-objects from video shots, IEEE ICIP, San Antonio, USA, september 2007.
7. P.Adam, P. Bertolino, F. Lebowsky, A simple LCD response time measurement based on a CCD line camera, Asia Display, Shanghai, China, march 2007.
8. T. Habib, M. Gay, J. Chanussot, P. Bertolino, Segmentation of high resolution satellite images SPOT applied to lake detection, International Geoscience and Remote Sensing Society, Denver, Colorado, USA, august 2006.
9. P.Adam, P. Bertolino, J.-M. Chassery, F. Lebowsky, LCD response time estimation, International Display Research Conference, Kent, Ohio, USA, september 2006.

10. J. Roussel, P. Bertolino, M. Nicolas, Improvement of conventional deinterlacing methods with extrema detection and interpolation, Advanced Concepts for Intelligent Vision Systems, Antwerp, Belgium, september 2006.
11. J. Huart, P. Bertolino, Similarity-based and perception-based image segmentation, IEEE ICIP, Genova, Italy, september 2005.
12. J. Huart, G. Foret, P. Bertolino, Moving object extraction with a localized pyramid, 17th International Conference on Pattern Recognition, Cambridge, UK, august 2004.
13. G. Foret, P. Bertolino, Label prediction and local segmentation for accurate video object tracking, Visual Communications and Image Processing, Lugano, 2003.
14. G. Foret, P. Bertolino, D. Cibaud, Partition projection in videos by global and local block-matching, IEEE ICIP, Rochester, USA, september 22-25, 2002.
15. P. Bertolino, G. Foret, D. Pellerin, Detecting people in videos for their immersion in a virtual space, ISPA 2001, Second International Symposium on Image and Signal Processing and Analysis, June, pages 313-318, Pula, Croatia, 2001.
16. P. Bertolino, R. Mohr, C. Schmid, P. Bouthemy, M. Gelgon, F. Spindler, S. Benayoun, H. Bernard, Building and using hypervideos, 4th IEEE workshop on applications of computer vision (WACV), Princeton, N.J., USA, 1998.
17. S. Benayoun, H. Bernard, P. Bertolino, P. Bouthemy, M. Gelgon, R. Mohr, C. Schmid, F. Spindler. Structuring video documents for advanced interfaces. ACM Multimedia Conference, Demo session, Bristol, Royaume-Uni, Septembre 1998.
18. P. Bertolino, S. Ribas, Image sequence segmentation by a single evolutionary graph pyramid, In Graph Based Representations in Pattern Recognition, pages 93-100. Springer-Verlag, 1998.
19. P. Bertolino, S. Ribas, Image sequence segmentation with a single evolutionary graph pyramid, first workshop on Graph Based Representation - GBR'97, IAPR, Lyon, 17 avril 18 avril 1997.
20. P. Bertolino, F. Davoine, J.M. Chassery, Which compression method for still cytological and histological images ?, in 5th conf. of the european society of analytical cellular pathology, Oslo, Norway, May 25 29, 1997.
21. P. Bertolino, A. Montanvert, Multiresolution segmentation using the irregular pyramid, IEEE ICIP, pp 257-260, Lausanne, September 17-19 1996.
22. P. Bertolino, A. Montanvert. Stochastic edge detection based on discrete segments. In 5th Colloquium DGCI, pages 117–126, Clermont-Ferrand, 25-27 Septembre 1995.
23. H. Bischof, E. Bertin, P. Bertolino, Voronoi pyramid and Hopfield networks. In proc of the 12th International Conference on Pattern Recognition, Jerusalem, Israel, 9-13 October 1994.
24. D. Attali, P. Bertolino, A. Montanvert. Using polyballs to approximate shapes and skeletons. In proc of the 12th International Conference on Pattern Recognition, Jerusalem, Israel, 9-13 October 1994.
25. P. Bertolino, A. Montanvert. Edge detection for biomedical image : a self-adaptive and randomized operator. In Proc. of the 14th Annual International Conference of the IEEE Engineering in Medecine and Biology Society, IEEE Comp. Soc. Press, pages 129–133, Paris, November 1992.

1.5.5 Conférences nationales

1. J. Huart, P. Bertolino, Extraction d'objets-clés pour l'analyse de vidéos, GRETSI, Troyes, septembre 2007.
2. J. Roussel, P. Bertolino, M. Nicolas, Détection et reconstruction des éléments hautes fréquences appliquées au désentrelacement , GRETSI, Troyes, septembre 2007.

3. P. Adam, P. Bertolino, Les écrans LCD dans le flou, MajecSTIC 2006, Lorient, France, novembre 2006.
4. P. Bertolino, J. Huart, G. Foret, Segmentation pyramidale localisée dans un ruban fermé, 20ème colloque GRETSI, Louvain-la-Neuve, Belgique, septembre 2005.
5. J. Huart, P. Bertolino, Segmentation pyramidale et groupements perceptuels, 20ème colloque GRETSI, Louvain-la-Neuve, Belgique, septembre 2005.
6. J. Huart, G. Foret, P. Bertolino, Extraction d'objets en mouvement par pyramide locale, 9èmes journées CORESA, Villeneuve d'Ascq, France, mai 2004.
7. P. Bertolino, G. Foret, D. Pellerin, Détection de personnes dans les vidéos pour leur immersion dans un espace virtuel, 18ème colloque GRETSI, Toulouse (France), Septembre 10-13, 2001.
8. S. Benayoun, H. Bernard, P. Bertolino, P. Bouthemy, M. Gelgon, R. Mohr, C. Schmid, F. Spindler. Structuration de vidéos pour des interfaces de consultation avancées. Journées CORESA (France Télécom-CNET), pages 207-214, Lannion, Juin 1998.
9. P. Bertolino, S. Ribas, Poursuite d'objets multirésolution par pyramide irrégulière, 16ème colloque GRETSI, Grenoble, Septembre 15-19 1997.
10. G. Braviano, A. Montanvert, P. Bertolino, Estratégias para a fusão de regiões utilizando conjuntos difusos, SIBGRAPI'97, 14 a 17 de outubro de 1997, Campos do Jordão, SP, Brazil
11. P. Bertolino, A. Montanvert. Coopération régions-contours multirésolution en segmentation d'image. In Actes du 10ème Congrès RFIA, pages 299-307, Rennes, 16-18 Janvier 1996.
12. A. Montanvert, P. Bertolino, Irregular pyramids for parallel image segmentation. In Proc. of the 16th OAGM Meeting, pages 13–34, Vienna, Austria, May 6-8 1992.

1.5.6 Publications diverses

1. P. Bertolino, Détection, description et mise en correspondance de points caractéristiques invariants dans les images, Délivrable 6.5.0, projet MOOV3D, GIPSA-lab 2011.
2. P. Bertolino, R. Mohr, C. Schmid, P. Bouthemy, M. Gelgon, F. Spindler, S. Benayoun, H. Bernard, Structuring Video Documents for Advanced Interfaces, Rapport technique, INRIA 1999.
3. P. Bertolino, J.M. Chassery, Compression with priority to speed, Europath project, deliverable D04-02, 1997
4. P. Bertolino, J.M. Chassery, C. Sowter, F. Davoine, Compression with priority to resolution, Europath project, deliverable D10-03, 1996
5. A. Montanvert, P. Meer, P. Bertolino, Hierarchical Shape Analysis in Grey-level Images, In Shape in Picture - Mathematical Description of Shape in Grey-level Images. Edited by : O. Ying-Lie, A. Toet, D. H. Foster, H. J.A.M. Heijmans, P. Meer. Publisher : NATO ASI Series F, Vol. 126, Springer-Verlag, Berlin, p.511-524, 1994.
6. A. Montanvert, P. Meer, P. Bertolino, Optimal hierarchical shape analysis in gray level images. In NATO advanced workshop "Shape in Picture", volume 126, pages 13–34, Driebergen, The Netherlands, September 7-11 1992. Springer Verlag ed.

1.5.7 Brevets

Deux des trois thèses co-encadrées avec STMicroelectronics ont fait l'objet d'un dépôt de brevet. Le premier, a été soumis, accepté et publié aux USA. Le second a été soumis en France et aux Etats-Unis :

1. J. Roussel, P. Bertolino, M. Nicolas, Image deinterlacing, US Patent Application 20080089614, déposé le 14 septembre 2007, publié le 17 avril 2008
2. E. Van Reeth, P. Bertolino, M. Nicolas, Procédé de détection d'orientation de contours. Déposé en France en janvier 2010 et aux Etats-Unis en janvier 2011.

1.6 Relecture

J'ai réalisé des relectures pour les revues et conférences suivantes :

Revues :

- Computer Vision and Image Understanding (2006, 2007, 2008, 2009)
- Pattern Recognition Letters (2005, 2006, 2007, 2008)
- Signal Processing (2009)
- Signal, Image and Video Processing (2010, 2011)
- IEEE Transactions on Circuits and Systems for Video Technology (2007)
- IEEE Transactions on Image Processing (2003)
- EURASIP Journal on Advances in Signal Processing (2010)
- SPIE Optical Engineering (2011)

Conférences :

- IEEE International Conference on Image Processing (2006, 2007, 2008, 2009, 2010, 2011)
- IEEE International Conference on Information Technology (2009)
- International Conference on Soft Computing and Pattern Recognition (2009)
- International Conference on Pattern Recognition (2002)

1.7 Conférences invitées

A quelques occasions, j'ai eu l'opportunité de présenter mes travaux à des publics variés non spécialisés dans le domaine de l'image numérique. Cet échange s'est matérialisé sous la forme de six conférences invitées, à l'ADIRA, l'ARATEM et l'INRA.

L'ADIRA est l'Association pour le Développement de l'Informatique en Région Rhône-Alpes. Son activité s'articule autour de 4 axes : informer, éditer des publications, organiser des manifestations professionnelles pour informaticiens ou utilisateurs, orienter les adhérents sur les contacts utiles en matière de services et produits du marché. A la demande de cette association, j'ai présenté à Meylan (Isère) en décembre 2005 une conférence intitulée "Le traitement des images et ses nombreuses applications".

L'ARATEM est l'Agence Rhône-Alpes pour la Maîtrise des Technologies de Mesure. Elle a organisé en janvier 2006 à St Etienne (Loire) son colloque annuel intitulé "Destination : innovation", auquel j'ai été invité à présenter une conférence. J'ai abordé le thème suivant : l'extraction d'objets dans les images et les vidéos. J'ai introduit quelques techniques pour extraire l'information pertinente des images et j'ai présenté quelques applications associées. 150 personnes dont une majorité d'industriels ont participé à ce colloque.

L'INRA (Institut National de Recherche en Agronomie) a un nombre important de chercheurs et d'ingénieurs qui travaillent avec des images numériques. Cet institut organise régulièrement une "Ecole Chercheurs" qui vise à (i) permettre aux chercheurs de développer des compétences de bases sur le traitement des images pour qu'ils acquièrent une certaine autonomie et (ii) favoriser les interactions et le fonctionnement en réseau. J'ai été invité à cette école deux fois, en 2006 à Batz sur Mer (Loire-Atlantique) et en 2008 à Guéthary (Pyrénées-Atlantiques) pour y présenter à chaque fois deux conférences respectivement sur la segmentation d'images et la quantification de paramètres dans les images.

1.8 Participation à des projets de recherche

1.8.1 Développement d'applications pour le bio-médical

Pendant mon DEA et ma thèse, j'ai passé trois années au laboratoire TIMC (Techniques de l'Imagerie, de la Modélisation et de la Cognition) à l'Institut Albert Bonniot de Grenoble. Dans cet institut étaient réunis informaticiens, mathématiciens et biologistes. Dans ce contexte, j'ai participé comme cheville ouvrière à trois projets Européens du programme IST (Information Society Technologies) où l'informatique et le traitement d'images étaient au service de la médecine et de la biologie. Dans l'ordre chronologique, voici mes contributions :

- projet HOME (Highly Optimized Microscope Environment) : développement d'un logiciel de morphométrie pour le microscope informatisé AxioHome de Zeiss (1992).
- projet IMPACT (Integrating Microscopy for Pathology Activities and Computer Technology) : développement d'un logiciel pour la gradation de cancers par modélisation de population de cellules par diagramme de Voronoï (1993-1994).
- projet EUROPATH (European Pathology assisted by Telematics for Health) : développement d'un démonstrateur pour la compression adaptative d'images, évaluation de techniques de compression pour la télé-pathologie. Rédaction de deux délivrables (1995-1996).

1.8.2 Projets en traitement et analyse d'images

J'ai participé activement à deux projets orientés segmentation :

- projet national exploratoire du RNRT : OSIAM (Outils de Segmentation d'Images Animées pour MPEG-4/7) : responsable au laboratoire LIS du projet. Développement d'outils de segmentation (1998-2001).
- projet européen Art.Live (ARchitecture and authoring Tools prototype for Living Images and new Video Experiments) : production d'un état de l'art, développement d'outils de segmentation (2000-2002).

1.8.3 Projet d'analyse d'images pour la réalité augmentée

Actuellement, je participe au projet MOOV3D (MObile Original Video 3D) qui a démarré en mai 2010. Ce projet dont le budget est de 4,9 millions d'euros est un des cinq projets Minalogic sélectionnés par le Fonds Unique Interministériel. D'une durée de trois ans (2010-2012), il vise à créer une plate-forme de développement disposant de toutes les fonctions matérielles et logicielles requises pour capturer, traiter et visualiser de la "3D-relief" sur téléphone portable et lunettes à micro-écrans OLED. Outre GIPSA-lab, les partenaires sont STEricsson, MicroOLED, Pointcube, Visioglobe et le CEA-Leti.

J'ai la responsabilité à GIPSA-lab du sous-projet *Outils génériques pour la reconstruction 3D*. Notre contribution est la suivante : à partir d'un flux vidéo constitué de paires stéréo rectifiées provenant de 2 caméras embarquées dans un téléphone portable, il s'agit de mettre en correspondance les paires d'images et d'estimer la pose du système stéréoscopique pour obtenir une reconstruction géométrique tri-dimensionnelle de la scène. Outre les informations vidéo, on utilisera celles fournies par les capteurs embarqués (GPS, accéléromètres et magnétomètre). L'extraction de primitives dans des vidéos prises en milieu urbain ou contrôlé devra ensuite assurer une mise en correspondance entre images réelles et modèles 3D connus, pour réaliser de la réalité augmentée. Ce sous-projet est découpé en trois tâches :

1. Calibration et correction de la distorsion,
2. Extraction et mise en correspondance de caractéristiques,
3. Reconstruction 3D et réalité augmentée.

Je suis le représentant de GIPSA-lab au comité technique du projet (nous avons déjà eu une quinzaine de réunions diverses en 8 mois) et je travaille sur l'aspect *Extraction et mise en correspondance de caractéristiques*. Pour les autres tâches, je suis secondé par deux chercheurs post-doctoraux. Le premier, Ferran Espuny Pujol, qui a réalisé sa thèse sur l'autocalibration vient de l'Université de Barcelone pour une durée de 12 mois. Le second interviendra durant la seconde année du projet sur l'aspect *réalité augmentée*.

1.8.4 Animation de stands

Quand on participe à des gros projets, on doit parfois les présenter à différents publics dans le contexte particulier d'un stand. J'ai eu la chance d'animer les stands suivants :

- Stand Zeiss/projet IMPACT, 1996, Paris, La Défense.
- Stand Zeiss/projet EUROPATH, Conférence du G7 : The Information Society and Development, mai 1996, Midrand (Johannesbourg), Afrique du Sud.
- Stand Zeiss/projet EUROPATH, Telematics for Health and Disabled and Elderly People conference and exhibition, mai 1996, Midrand (Johannesbourg), Afrique du Sud.
- Stand ArtLive, 2001, jardins de Bercy, Paris.

1.9 Enseignement

Statutairement, j'enseigne au département informatique de l'IUT2 de Grenoble. Dans cette section, j'indique rapidement mon implication et mes centres d'intérêt comme enseignant.

1.9.1 Enseignement de l'informatique

Le département informatique de l'IUT2 forme des techniciens de niveau bac + 2 (DUT) et bac + 3 (licence professionnelle). Le DUT peut être fait en 2 ans (cycle initial) ou un an (année spéciale). Les licences sont préparées en alternance. Les promotions représentent environ 120 étudiants de 1ère année, 100 de 2ème année, 20 d'année spéciale et 40 de licence. Pendant les premières années de ma nomination, j'ai enseigné les matières suivantes :

Algorithmique, TD et TP aux étudiants de 1ère année.

Système d'exploitation, TD et TP aux étudiants de 2ème année.

Au fil des ans, j'ai pris la responsabilité d'enseignements. J'enseigne actuellement les matières suivantes :

Architecture des ordinateurs, cours, TD et TP aux étudiants d'année spéciale.

Programmation en langage C, j'ai refondé en totalité cet enseignement lorsque j'en ai pris la responsabilité. Cours et TP aux étudiants de première année et année spéciale, animation de l'équipe pédagogique qui donne les TP avec moi.

Images et vidéo, responsable de l'enseignement que j'ai créé à l'IUT. Cours et TP aux étudiants de licence professionnelle.

Développements en Flash, responsable de l'enseignement que j'ai créé à l'IUT. Cours et TP aux étudiants de licence professionnelle.

1.9.2 Responsabilité des stages

La responsabilité des stages de fin d'études au département informatique de l'IUT est la charge administrative principale qui m'a été déléguée lors de mon recrutement comme maître de conférences : les étudiants de 2ème année, d'année spéciale et de licence doivent effectuer un stage de 10 semaines. Cela représente environ 160 étudiants tous les ans. J'ai assumé cette responsabilité pendant 5 années.

1.9.3 Participation à la création d'une licence professionnelle

Il est motivant et enrichissant de mettre en relation sa thématique de recherche et son enseignement. Cela doit permettre pédagogiquement de mieux captiver l'intérêt des étudiants. Pour l'enseignant-chercheur, c'est un des moyens de prendre du recul sur son travail de recherche. Pour l'enseignant en technologie que je suis, c'est un défi où le but consiste à s'appuyer sur des bases théoriques présentées au niveau conceptuel pour amener à des fonctionnalités et des applications actuelles.

Lors de l'avènement des licences professionnelles, en 2001, notre département a ouvert deux licences professionnelles, dont une s'intitule MIAM (Métiers de l'Internet et des Applications Multimédia). J'ai construit deux enseignements pour cette licence :

- "Images et vidéo", 30 heures qui présentent les bases de l'image numérique, de son traitement et de son analyse. J'illustre les techniques étudiées par leur application à l'édition et la retouche d'images. Nous utilisons les logiciels ImageJ et GIMP.
- "Développements en Flash", 20 heures pendant lesquelles les étudiants apprennent à créer des images vectorielles dynamiques qu'ils manipulent en langage ActionScript pour réaliser des animations et des applications graphiques qui sont placées sur des sites web.

1.9.4 Autres charges d'enseignement

Outre l'enseignement que j'effectue dans mon UFR d'appartenance, j'ai assuré la création et la responsabilité d'enseignements dans d'autres UFR :

- Au master Compétence Complémentaire en Informatique à l'UFR Informatique et Mathématiques Appliqués de l'Université Joseph Fourier, j'ai mis en place le cours et les TP de l'option Imagerie Numérique et j'ai assumé la co-responsabilité de cet enseignement pendant 7 ans.
- Le master Ingénieries pour la Santé et le Médicament (porté par l'UJF, UFR de Médecine-Pharmacie et habilité avec l'INPG) forme des personnels de santé. Il se situe à l'interface de deux disciplines : la physique d'une part et d'autre part la médecine et la pharmacie. Je suis intervenu dans la spécialité Approche spatio-temporelle des systèmes vivants, option Imagerie Morphologique, Fonctionnelle et Métabolique, où j'ai monté deux cours : l'un portant sur l'analyse des images binaires, l'autre sur le mouvement dans les images. J'ai donné ces cours pendant 4 ans.
- Au département Formation Continue de Grenoble INP, deux collègues enseignants-chercheurs de GIPSA-lab et moi-même avons réuni nos compétences pour rajouter au catalogue une formation théorique et pratique de 4 jours. Ce stage s'adresse principalement à des ingénieurs et techniciens possédant des connaissances de base en traitement d'images numériques. Il est découpé en 4 parties : pré-traitement, segmentation, analyse fréquentielle et étude de cas. J'ai monté le cours et les exercices relatifs à l'aspect segmentation. Nous avons donné ce stage a 9 reprises, soit en entreprise (Trixell, Thales, CEMAGREFF, CNRS, CEA), soit en formation inter-entreprises.

1.10 Interactions avec l'industrie

Avant de rejoindre le monde académique, j'ai passé 7 années dans l'industrie dans 5 entreprises différentes, petites et moyennes, à différents niveaux de responsabilité. Ma vision du monde de la recherche s'en trouve imprégnée et je n'envisage pas la recherche sans sa finalité applicative et son transfert technologique. Dans cette section, je relate trois expériences de chercheur que j'ai eues avec l'industrie.

1.10.1 Création de société

Je me suis beaucoup intéressé à l'ergonomie des interfaces graphiques dédiées à l'image et la vidéo ; mon séjour d'un an et demi à l'INRIA (section 3.5 page 67) pour développer entre autre de telles interfaces a motivé encore plus cet intérêt. Pour cette raison, dès 1998, j'ai commencé pendant mes heures de loisir à

développer une application destinée aux utilisateurs d'appareils photos numériques. C'est à cette époque qu'apparaissent sur le marché les premiers appareils numériques dont certains peuvent également produire des séquences vidéo sonores. Après 2 années de travail dans l'ombre sort la première version d'un logiciel pour Windows©. Quelques mois plus tard, nous fondons avec mon épouse la SARL Vizway (page d'accueil du site web en annexe, page 137), pour développer et commercialiser un logiciel de gestion d'images et de vidéos en langue anglaise nommé AfterCam. Ce logiciel a les fonctionnalités principales suivantes :

- Classement et affichage d'images et vidéos aux formats les plus utilisés (une vingtaine).
- Amélioration et transformation d'images (netteté, lumière, contraste, taille, formats, *crops*, rotation, yeux rouge, balance des blancs, divers filtres).
- Player de vidéo évolué plein écran.
- Construction de résumé de vidéos et navigation avec ces résumés.
- Extraction d'images à partir de vidéos.
- Construction de diaporamas et d'albums photos pour pages web.

Hormis le développement du logiciel, le travail a consisté à gérer un serveur de site localisé aux Etats-Unis [†], à développer le site web de la société, à référencer le produit au niveau national et international et à mettre en place des moyens de paiement sécurisés évoluant avec l'offre du marché.

AfterCam (figures en annexe, page 139) était un logiciel en licence *shareware* disponible en version complète et illimitée dans le temps. Il a été acheté par des centaines de particuliers, sociétés et administrations dans plus de 50 pays. Il a été téléchargé des dizaines de milliers de fois. Nous avons dû arrêter la société en décembre 2004 par manque de temps à lui consacrer.

1.10.2 Conception d'un système de vision industrielle

En 2005, la Chambre de Commerce et de l'Industrie de Grenoble me met en contact avec la société Séripress, localisée à Saint Marcellin (Isère), employant une centaine de salariés et spécialiste mondiale de fabrication de transferts sérigraphiques (la sérigraphie est une technique d'imprimerie) pour les vêtements et articles textiles.

Cette PME recherchait une compétence en vision pour un projet. Celui-ci, étalé sur l'année 2005, consistait à réorganiser le système de contrôle qualité jusque là effectué visuellement, en implémentant un système de vision industrielle pour le contrôle automatique des défauts, directement sur les lignes de production existantes. L'entreprise n'avait aucune compétence en interne et avait donné la responsabilité du projet à un technicien avec qui j'ai travaillé.

Le but était de détecter des problèmes de couleurs et d'alignement entre les couleurs de la quadrichromie et différents vernis : dans le processus de fabrication, les feuilles à imprimer sont recouvertes d'une dizaines d'encres et vernis différents puis d'une couche de colle. Le paquet entier de feuilles vierges est tout d'abord imprimé avec une première couleur lors d'un premier passage, à travers un écran (tissu tendu et fixé sur un cadre) qui contient le motif à imprimer. Cette étape est réitérée pour le paquet entier pour chacune des encres, à chaque fois avec un écran différent. Le positionnement précis du cadre est à chaque fois réalisé mécaniquement par le conducteur de la machine, mais des dérives et des défauts d'impression ponctuels peuvent survenir. Jusqu'à présent, le contrôle de qualité était fait visuellement en fin de chaîne (donc trop tard). Le but était de réaliser ce contrôle en temps réel pour améliorer la qualité et la productivité. Dans ce projet, mon rôle a été le suivant :

- Répertorier et classer les problèmes spécifiques du projet.
- Définir les limites de faisabilité du projet.
- Effectuer une recherche bibliographique des solutions existantes sur le marché.
- Définir les mires de contrôle optimales (figure 1.2.a)
- Choisir des technologies évolutives (caméras, capture d'image, langage de programmation et bibliothèques)

†. www.vizway.com

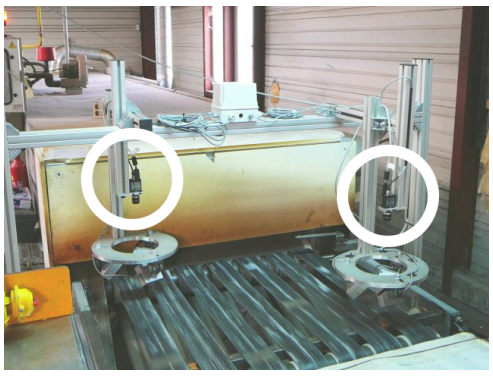
- Fournir les méthodes de traitement d'image appropriées
- Valider les solutions proposées par des tests au laboratoire LIS.
- Aider à l'appropriation des technologies par l'entreprise en formant le technicien.

Un an et demi après le début de notre collaboration, un prototype pleinement fonctionnel était utilisé quotidiennement sur une des 4 machines et était validé par la direction (figures 1.2.b et 1.3). Le déploiement sur les 3 autres machines était prévu pour l'automne 2006. Aucune société extérieure n'est intervenue pour le conseil ni pour le développement ou l'installation de la solution. Le technicien possède les compétences pour faire évoluer la solution existante qui est pleinement satisfaisante par rapport aux buts fixés. La précision atteinte est de 100% : à chaque arrêt de la production par le système de contrôle, le conducteur a reconnu que c'était justifié. En revanche, en 2006, le rappel n'avait pas encore été évalué. En effet, pour cela il fallait contrôler visuellement toutes les feuilles en sortie de chaîne.

(a) en bas, l'ancienne mire pour le contrôle visuel. En haut, la mire pour le contrôle par caméra

(b) Ecran de contrôle qui affiche les 2 mires et les problèmes éventuels

FIGURE 1.2 – Mires de contrôle de calage et de couleurs. La précision obtenue est de 0.1mm



(a) 2 caméras monochromes matricielles (cerclées en blanc) font l'acquisition

(b) le système de contrôle, synchronisé avec le passage de la feuille flashe et analyse la mire en temps réel

FIGURE 1.3 – Système de contrôle de qualité par vision industrielle, société Séripress

1.10.3 Transfert technologique et création de startup

En 2007, contacté par deux jeunes entreprises qui recherchent des outils de détourage pour les vidéos, je décide d'entreprendre un projet de valorisation des travaux de segmentation spatiale et spatio-temporelle que j'ai réalisés et encadrés.

J'obtiens un budget de 65.000 euros auprès de GRAVIT (dispositif de valorisation mutualisé du pôle Grenoblois), ce qui me permet de développer en une année un prototype logiciel avancé. Ensuite, l'incubateur régional GRAIN (GRenoble Alpes Incubation) accueille la petite équipe que j'ai formée (dont mon ex-thésard Jérémy Huart) pour une durée d'incubation d'un an et demi. Pendant cette période, le prototype devient un produit, une étude de marché est réalisée et la future société affine son *business model* et son *business plan*.

L'application logicielle développée (figure 1.4) permet à un opérateur de suivre avec précision des objets d'intérêt dans une vidéo existante sans avoir à les détourer de manière fastidieuse dans chaque image. Les objets ainsi extraits sont stockés indépendamment de la vidéo et des informations ou actions relatives à ces objets peuvent leur être associées et déclenchées pour créer des vidéos enrichies et interactives.

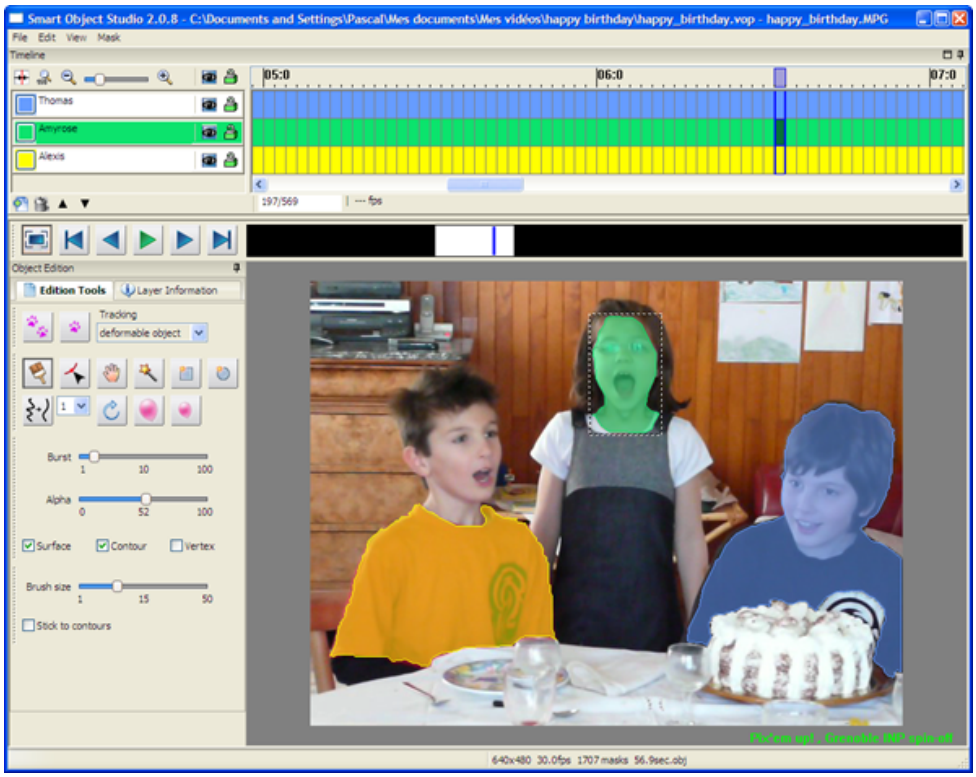


FIGURE 1.4 – Application de détourage dans les vidéos développée lors de la valorisation de travaux en segmentation spatio-temporelle

Partant des objets que peut extraire notre logiciel, de nombreuses applications sont possibles ; parmi elles les vidéos cliquables pour un nouveau type de publicité : dans son navigateur, l'internaute peut cliquer sur les objets d'une vidéo (produit, personnage, ...) et peut par exemple être redirigé sur un site marchand ou une fiche descriptive (figure 1.5). Toutes les informations relatives aux clics sont stockées de façon à ce que le responsable de la campagne publicitaire puisse accéder à des statistiques lui permettant de contrôler et d'améliorer l'impact de sa campagne.

Trois ans et demi après le début de cette aventure, en mai 2010, je décide suite à un désaccord important avec le futur dirigeant de la société, de quitter ce projet. En mars 2011, le projet de création d'entreprise échoue. Grenoble INP récupère la propriété intellectuelle et tous les développements réalisés

FIGURE 1.5 – Exemple de vidéo cliquable réalisée et jouée sur un site web : lorsque le curseur de la souris passe sur le polo, le reste de l'image est grisé et le nom du produit apparaît sous le curseur

dans le cadre du projet, me confie ces derniers et me restitue dans mon rôle de responsable scientifique de ce projet. Dans les mois qui viennent je reprendrai donc la valorisation sous une autre forme pour diffuser ce qui a été développé et poursuivre l'intégration des travaux de recherche dans cette application.

1.11 Applications logicielles développées

Voici quelques applications (figures 1.6 et 1.7) que j'ai développées et qui sont disponibles au téléchargement, soit pour plate-forme Windows, soit Windows ou Linux[‡]. Ces applications correspondent à des outils qui répondent à des besoins ponctuels mais également à des résultats de travaux de recherche.

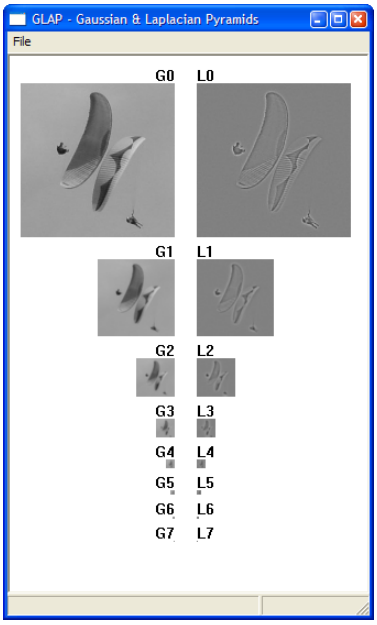
1.11.1 Outils de segmentation

- MCISS : *A Multiresolution Color Image Segmentation Software*
- WAIPS : *Watershed And Irregular Pyramid Segmentation* ; Plugin de segmentation d'image pour le logiciel de retouche d'image GIMP (l'équivalent libre de Photoshop), réalisé avec l'aide de Sébastien Berthier, stagiaire de DUT
- ExtraK'Obs : Segmentation d'images supervisée
- TraFiCS : *Tracking with a Fixed Camera System*
- EDDECS : *EDge DEtection with Canny and Sobel*
- GLAP : *Gaussian and LApplacian Pyramids*
- Smart Object Studio : Application de détourage dans les vidéos

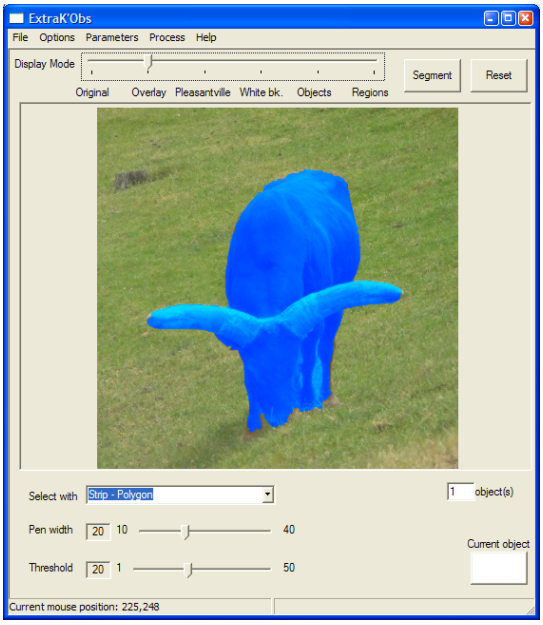
1.11.2 Utilitaires pour l'image et la vidéo

- PSNR : application graphique permettant de visualiser 2 images et de calculer leur PSNR
- mpeg2yuv : application graphique qui convertit des fichiers MPEG au format YUV 4 : 2 : 0
- YUV player : joue tout fichier YUV 4 : 2 : 0
- YUV2any : convertit un fichier YUV 4 : 2 : 0 en un ensemble d'images aux formats jpg, png, ...
- deTizzyer : un *frontend* de FFMPEG. L'interface graphique permet de transcoder des flux vidéos de différents formats en utilisant FFMPEG de manière transparente

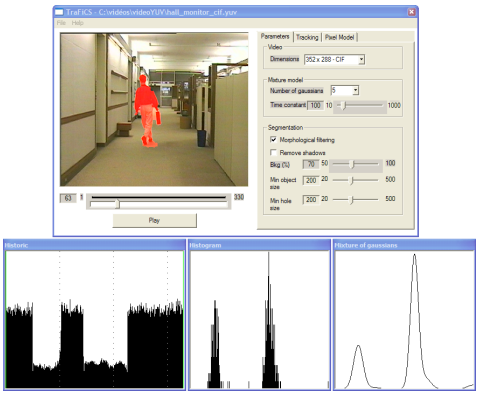
‡. www.gipsa-lab.inpg.fr/p̃ascal.bertolino/software.html



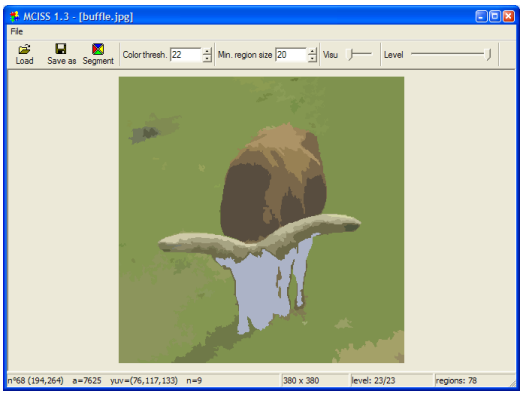
(a) GLAP, illustration des pyramides Gaussiennes et Laplaciennes



(b) ExtraK'Obs, segmentation supervisée



(c) TraFiCS, une implémentation de l'utilisation du mélange de Gaussiennes de [SG99] pour le suivi en temps réel



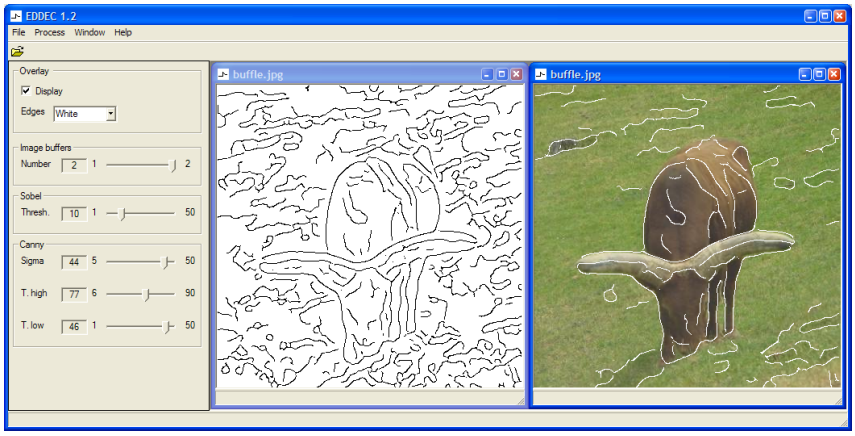
(d) MCISS, Segmentation multirésolution d'images couleurs ave la pyramide irrégulière

FIGURE 1.6 – Applications de segmentation développées lors de mes recherches

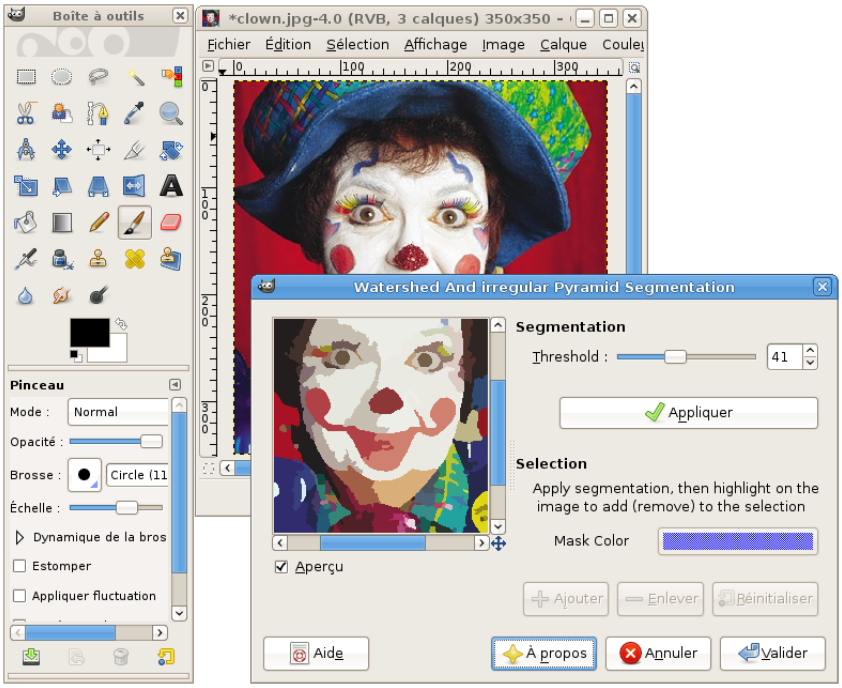
- BertImage : code source qui explique comment charger, afficher, traiter et sauvegarder une image avec la bibliothèque graphique wxWidgets [Sma92] [CHS05]
- VideoPrep : outil de construction de vidéos structurées (figure D.2.a, b et c page 142)
- VideoClic : application de navigation dans les hypervidéos (figure D.2.d page 142)
- AfterCam : trousse à outils pour l'image et la vidéo (figures page 139)

1.12 Structure du document

Ce chapitre qui se termine visait à introduire toutes mes activités qui gravitent autour de la recherche et de l'encadrement de la recherche à proprement parler. Le reste du document est organisé comme suit. Les quatre premiers chapitres ont trait à la segmentation :



(a) EDDECS, pour la comparaison des détecteurs de Sobel et de Cany



(b) WAIPS, un pluggin GIMP pour segmenter des images couleur avec un *watershed* couplé à une pyramide irrégulière

FIGURE 1.7 – Applications de segmentation développées lors de mes recherches

- Le chapitre 2 aborde la segmentation spatiale à l'aide la pyramide irrégulière.
- Le chapitre 3 présente nos travaux en segmentation spatio-temporelle.
- Dans le chapitre 4, nous décrivons notre approche pour extraire des objets clé dans une vidéo.
- Le chapitre 5 traite de segmentation de personnes.

Ensuite, trois chapitres présentent les travaux réalisés lors de trois thèses CIFRE relatives à l'amélioration de la qualité d'affichage sur les écrans plats :

- Le chapitre 6 présente une technique originale de désentrelacement d'images.
- Le chapitre 7 décrit une méthode d'agrandissement d'images.
- Le chapitre 8 propose des algorithmes pour améliorer la qualité de l'image LCD et la perception

des mouvements.

Enfin, le chapitre 9 conclut brièvement ce document et donne des perspectives à ma recherche.

Des annexes contenant des figures sont disponibles à partir de la page 135.

Revues internationales	**3**
Journal of the Society for Information Display	1
Computer Vision and Image Understanding	1
Analytical Cellular Pathology	1
Revue nationale	**1**
Traitement du Signal	1
Conférences internationales	**24**
ICIP (image processing)	6
ICPR (pattern recognition)	3
SPIE (electronic imaging)	2
VISAPP (computer vision)	1
ACM (multimedia)	1
EMBS (engineering in medecine and biology)	1
ESACP (analytical celular pathology)	1
ISPA (image and signal processing)	1
VCIP (visual communications and image processing)	1
WACV (applications of computer vision)	1
DGCI (discrete geometry for computer image)	1
GBR (graph based representations in pattern recognition)	1
ACIVS (advanced concepts for intelligent vision systems)	1
IDRC (international display research conference)	1
Asia Display (display)	1
IGARSS (international geoscience and remote sensing)	1
Conférences nationales	**12**
GRETSI (traitement du signal et de l'image)	6
CORESA (compression et représentation des signaux audiovisuels)	2
OAGM (pattern recognition)	1
RFIA (reconnaissance des formes et intelligence artificielle)	1
SIBGRAPI (computer graphics and image processing)	1
MajecSTIC (MAnifestation des Jeunes Chercheurs STIC)	1
Divers	**6**
Workshop NATO (shape in picture)	2
Delivrables de projet européen	2
Delivrable de projet MOOV3D	1
Rapport INRIA	1
Brevets avec STMicroelectronics	**2**
Image deinterlacing, accepté	1
Procédé de détection d'orientation de contours, déposé	1

TABLE 1.1 – Synthèse des publications

Chapitre 2

Segmentation d'objets dans les images

Sommaire

2.1 Appréhender l'image comme un ensemble d'objets

A l'heure actuelle, le consommateur vit une révolution permanente dans le domaine toujours plus vaste de l'image numérique et de la vision par ordinateur. Cette révolution apporte quotidiennement à un marché d'un milliard de consommateurs des services et des produits encore dans les laboratoires quelques années ou parfois même quelques mois plus tôt. Quoi qu'on veuille faire avec ces images numériques, on sait maintenant que le traitement sera d'autant plus efficace qu'il prendra en compte le contenu de l'image. Bien souvent, on aimerait avoir un traitement d'aussi bonne qualité que celui que nous offre notre système visuel et notre sens artistique. D'autres fois, on voudrait que le traitement offre à nos yeux la meilleure qualité subjective. Pour ces deux raisons, il est devenu inévitable de manipuler une image non plus comme un signal discret, mais comme un ensemble d'objets sémantiquement représentatifs à l'Homme.

Ce chapitre est consacré à la segmentation d'images statiques réalisée avec la pyramide irrégulière. Je commencerai par rappeler les grands principes de cette technique. J'indiquerai comment à l'aide de cette structure hiérarchique il est possible de réaliser de la multirésolution. Ensuite, je proposerai une modélisation particulière des régions pour amorcer une technique de synthèse et de compression d'images. Je présenterai le principe générique de la pyramide irrégulière locale qui est un outil pratique pour localiser précisément les contours d'objets. Enfin, je montrerai comment la ligne de partage des eaux peut efficacement initialiser la pyramide irrégulière et corriger ses principaux défauts.

2.2 Segmentation par pyramide irrégulière

Dans cette section, je présente l'outil générique de segmentation en régions que nous utilisons largement par la suite.

Cette technique a été co-développée par un chercheur du laboratoire GIPSA-lab [MMR91] et j'ai continué ses travaux durant ma thèse [Ber95] ; nous maîtrisons donc bien la technique et son code. Cette technique est intéressante car elle est fondée sur une structure de graphe très souple qui permet une adaptation aisée à tout type de besoin.

La pyramide irrégulière est un empilement d'images dont la résolution décroît de la base vers l'apex *. Sa particularité principale réside dans le fait que la forme des régions extraites n'est pas contrainte géométriquement. La segmentation par approche 'régions' et notamment par structure pyramidale irrégulière utilise une représentation par graphe. Celle-ci est bien adaptée aux relations d'adjacence qui unissent les régions. Elle devient nécessaire pour représenter des relations topologiques entre pixels ou régions : ces relations ne peuvent plus être portées implicitement par une structure régulière. Voici quelques éléments qui caractérisent bien les particularités de la pyramide irrégulière :

- C'est un empilement de niveaux, chacun d'eux étant une partition plus ou moins fine de l'image originale.
- La base de la pyramide est l'image originale où chaque pixel est déjà une région à part entière.
- Le niveau $k+1$ de la pyramide est construit à partir du niveau k.
- Chaque région est modélisée par un sommet de graphe d'adjacence. Par la suite, on utilise indifféremment les termes *région* ou *sommet* en fonction du contexte.
- Tous les niveaux de la pyramide d'images ont la taille de l'image originale. La diminution de la résolution s'obtient par la réduction du nombre de régions qui composent un niveau.
- Les traitements sont locaux : chaque sommet prend des décisions en fonction de ses voisins dans le graphe d'adjacence.
- Une région à un niveau k ($k \neq 0$ et $k \neq$ apex) possède un nombre quelconque (irrégulier) non nul de voisins du même niveau, un parent unique au niveau $k+1$, et un nombre quelconque non nul d'enfants au niveau $k-1$ (figure 2.1).
- Un **champ récepteur** est l'ensemble connexe des pixels de l'image qui correspondent à un sommet de la pyramide, à un niveau donné (figure 2.2). Réciproquement, le sommet récepteur d'un pixel est le sommet auquel est rattaché ce pixel à un niveau donné.

∗. niveau le plus élevé

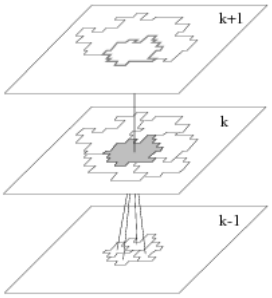
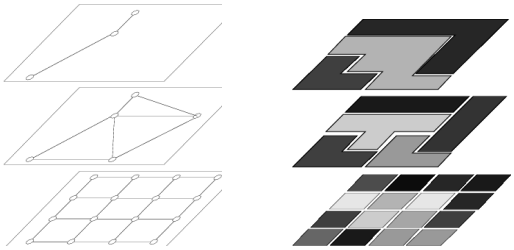


FIGURE 2.1 – Une région de la pyramide irrégulière au niveau k (en gris) et les régions auxquelles elle est reliée (mère, voisines et filles)



(a) Pyramide de graphes d'adjacence des régions

(b) Pyramide de champs récepteurs : 16, 5 puis 3 régions

FIGURE 2.2 – Correspondance entre graphes et champs récepteurs. Les graphes servent à modéliser les adjacences entre régions. Les champs récepteurs indiquent quelle partie de l'image chaque région occupe

2.3 Algorithme de construction de la pyramide irrégulière

La construction d'une pyramide irrégulière est fortement algorithmique. Cette section présente l'algorithme général et résume brièvement le rôle de chaque étape du traitement.

La structure pyramidale est en réalité une pyramide double : une pyramide de graphes qui est une structure de données interne à l'algorithme (figure 2.2.a) ainsi qu'une pyramide (ou empilement) de partitionnements appelés champs récepteurs qui représentent la partie visible au sens propre, des différents niveaux (figure 2.2.b). D'un point de vue implémentation, il est important de noter que pour une image de taille $N \times M$ pixels, un niveau de la pyramide de champs récepteurs est une image de pointeurs de taille $N \times M$ où chaque élément pointe sur le sommet correspondant dans la pyramide de graphes. Cette correspondance permet à chaque pixel d'accéder directement aux informations de sa région, comme lors de l'affichage du résultat par exemple.

La base de la pyramide (niveau 0) est établie à partir des relations de 4 ou 8-voisinage des pixels de l'image. Ce graphe est non orienté : une arête entre 2 sommets s_i et s_j indique que s_i est voisin à s_j et réciproquement. Chaque sommet est une structure de données qui stocke une partie (locale) de la pyramide : liste de ses voisins, de ses enfants, son père et un ensemble d'attributs qui caractérisent la région : couleur moyenne, surface, périmètre, ... L'algorithme de construction (algorithme 1) général est présenté ci-dessous :

- La procédure `CONSTRUIRE GRAPHE D'ADJACENCE` est chargée d'initialiser la structure pyramidale, c'est-à-dire de créer le niveau 0 à partir de l'image à traiter en utilisant la 4 ou 8-connexité.
- L'algorithme est constitué d'une boucle principale. Elle correspond au traitement effectué pour chaque niveau de la pyramide. Lorsqu'un niveau est construit, le nombre de sommets qui le composent est calculé. La construction de la pyramide s'arrête (i.e. l'apex est atteint) lors de la convergence du nombre de sommets.
- La boucle principale de l'algorithme, (i.e. construction d'un niveau), comporte cinq phases successives :

  1. La procédure `CONSTRUIRE GRAPHE DE SIMILARITE` permet de construire le sous-graphe de

Algorithme 1 : Construction de la pyramide

```
niveau ← 0 ;
apex = faux ;
CONSTRUIRE GRAPHE D'ADJACENCE(niveau) ;
tant que NON apex faire
    CONSTRUIRE GRAPHE DE SIMILARITE(niveau) ;
    tant que nombre survivants[niveau] décroît faire
        DECIMER GRAPHE DE SIMILARITE(niveau) ;
    fin
    ALLOUER NON SURVIVANTS(niveau) ;
    METTRE A JOUR SURVIVANTS(niveau) ;
    niveau ← niveau +1 ;
    CONSTRUIRE GRAPHE D'ADJACENCE(niveau) ;
    apex ← nombre régions[niveau-1] = nombre régions[niveau] ;
fin
```

similarité à partir du graphe d'adjacence : chaque sommet détermine les sommets voisins qui lui sont similaires et avec qui il pourra potentiellement fusionner [MMR89].

2. Un processus itératif de décimation des sommets du graphe de similarité (procédure `DECIMER GRAPHE DE SIMILARITE`) permet de sélectionner un sous-ensemble de sommets (représentatifs) qui constitueront le niveau suivant. Il existe des techniques de décimation variées [Lub86, MC89, JM92].

3. Dans la procédure `ALLOUER NON SURVIVANTS`, chaque sommet non retenu pour former le graphe du niveau supérieur est rattaché à un sommet survivant voisin dans le graphe de similarité. Les champs récepteurs associés fusionnent.

4. Les attributs des sommets survivants sont calculés en prenant en compte leurs propres attributs et ceux de chacun des sommets non-survivants associés (procédure `METTRE A JOUR SURVIVANTS`).

5. La dernière phase (procédure `CONSTRUIRE GRAPHE D'ADJACENCE`) reconstruit le nouveau voisinage du niveau $k+1$ à l'aide du graphe d'adjacence du niveau k.

Le processus de construction du niveau $k+1$ est terminé. Une itération identique peut alors commencer pour construire le niveau $k+2$ en partant du niveau $k+1$.

2.4 Principe de fusion

Il est important d'insister sur le principe de fusion qui est une des particularités de la pyramide irrégulière.

Lors de la construction d'un nouveau niveau, le regroupement de sommets [MMR91] se fait indépendamment dans le graphe-étoile centré sur chaque sommet survivant (figure 2.3), selon une phase de décimation (des sommets disparaissent) et une phase de rattachement (les sommets disparus se rattachent au sommet voisin survivant le plus similaire). Ainsi, le nombre de sommets se regroupant localement n'est pas connu *a priori*. Les phases de décimation et de rattachement nécessitent que la similarité soit évaluée entre tous les couples de sommets voisins (cf. algorithme 2). Un seuil de similarité local est calculé pour chaque sommet, en fonction de son voisinage, ce qui explique que s_i peut considérer qu'il est similaire à s_j alors que l'inverse peut ne pas être vrai. Cette relation de similarité est donc modélisée par un graphe orienté.

Algorithme 2 : Construction du graphe de similarité

Entrées :
$G_a = \langle S, A \rangle$, le graphe d'adjacence, (S l'ensemble des sommets, A l'ensemble des arêtes [a])
T, le seuil global de similarité

Sorties :
$G_s = \langle S, E \rangle$, le graphe de similarité, (S l'ensemble des sommets, E l'ensemble des arcs [b])

$E = \emptyset$;
pour chaque *sommet* $s_k \in S$ **faire**
  Calculer le seuil de similarité locale T_k ;
  pour chaque *voisin* s_i *de* s_k *dans* G_a **faire**
    $dif = |NiveauDeGris(s_k) - NiveauDeGris(s_i)|$;
    si $dif < T_k$ *ET* $dif < T$ **alors**
      $E = E \cup arc(s_k, s_i)$; // s_k est similaire à s_i
    fin
  fin
fin

a. G_a est un graphe non orienté, les liens entre sommets sont bi-directionnels
b. G_s est un graphe orienté, les liens entre sommets son uni-directionnels

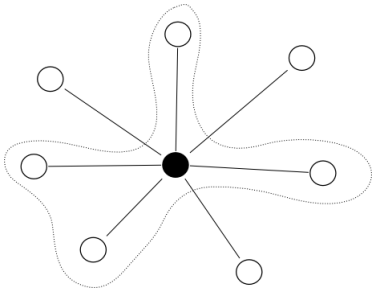


FIGURE 2.3 – Un sommet survivant (ici en noir) forme une étoile avec ses voisins. Le tracé montre un exemple de regroupement possible dans cette étoile qui entrainerait la fusion de cinq régions en une seule

2.5 L'aspect multirésolution

Dans cette section, nous montrons qu'en combinant simplement les partitionnements fournis par la pyramide irrégulière, on obtient une représentation multirésolution (en terme de régions) de l'image traitée.

La pyramide irrégulière n'est pas seulement un empilement de graphes. C'est aussi un arbre dont les racines sont les régions finales à l'apex de la pyramide et les feuilles les pixels de l'image. Alors qu'un niveau donné de la pyramide représente des régions qui forment un découpage arbitraire (par exemple un ciel uniforme découpé en centaines de régions, vers la base de la pyramide), nous nous sommes intéressé à remplacer cet arbitraire par une notion de multirésolution liée à un critère d'échelle. L'idée consiste en partant de la racine, de découper récursivement une région en régions filles lorsqu'un critère n'est pas vérifié [BM96]. Le découpage se fait pour une région en redescendant dans la pyramide d'un étage pour remplacer la région par l'ensemble de ses régions filles.

On se place dans le cas d'images en niveaux de gris. Le critère choisi est l'écart-type σ des pixels d'un sommet. La pyramide est dans un premier temps construite classiquement. Ensuite, en fonction d'un seuil σ_M donné, chaque région r de l'apex est divisée si $\sigma(r) > \sigma_M$, et ce de façon récursive. Autrement dit, alors qu'auparavant, à la fin de la segmentation chaque pixel de l'image était relié à un sommet récepteur

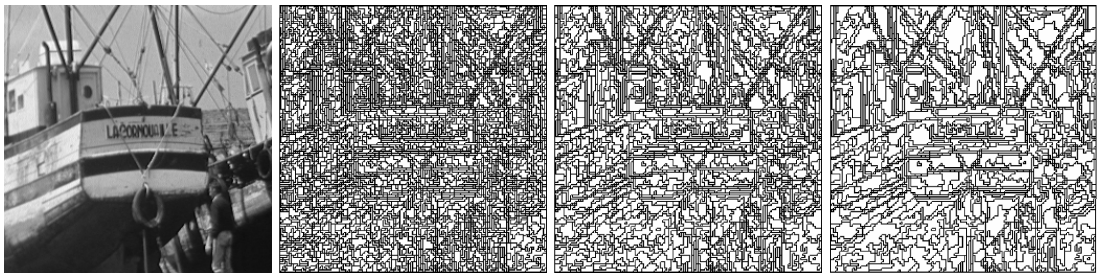
de l'apex, ici chaque pixel est relié à un sommet récepteur final du niveau le plus élevé respectant le critère utilisé. (algorithme 3).

La segmentation à l'échelle σ_M montre alors un partitionnement dont les régions proviennent de niveaux différents dans la pyramide : d'un niveau assez bas dans les zones texturées ou riches en détails, d'un niveau élevé (l'apex par exemple) dans des larges zones homogènes. Lorsqu'on fait varier σ_M, on obtient des segmentations où les détails de l'image apparaissent plus ou moins. De plus, comme la pyramide est déjà construite et que le traitement ne consiste globalement qu'à un parcours d'arbre (associé à un affichage), le résultat pour un σ_M donné est obtenu en temps réel ; un utilisateur peut ainsi à l'aide d'une souris modifier la valeur d'un curseur σ_M plusieurs fois par seconde et voir les détails apparaître ou disparaître, même pour des images de grande taille. Les figures 2.4 et 2.5 comparent les premiers niveaux d'une pyramide avec des résultats de la pyramide multirésolution obtenus pour plusieurs valeurs de σ_M.

L'application graphique qui implémente l'algorithme de la pyramide irrégulière et celui de la pyramide multirésolution peut être téléchargée à l'adresse suivante :
http ://www.gipsa-lab.inpg.fr/∼pascal.bertolino/software/waips.exe

Algorithme 3 : Procédure récursive Diviser(sommet, niveau)

```
Données : un sommet appartenant à un niveau
pour chaque fils f du sommet faire
    si σ(f) > σ_M alors
        Diviser(f, niveau − 1) ;
    sinon
        pour chaque pixel p dont f est le sommet récepteur faire
            sommet récepteur final(p) = f ;
        fin
    fin
fin
```



(a) Image originale (b) Niveau 1 (c) Niveau 2 (d) Niveau 3

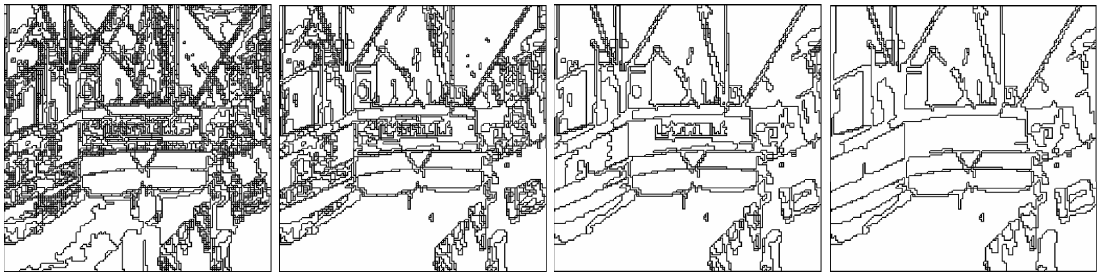
FIGURE 2.4 – Les premiers niveaux d'une pyramide classique

2.6 Synthèse et compression d'images par pyramide de surfaces

Dans cette section, je présente le travail de master de Tien Sy Nguyen qui montre que la pyramide irrégulière est un outil flexible et versatile. Dans cette recherche, nous désirons améliorer la façon dont les régions sont modélisées et également envisager d'autres finalités que la segmentation.

2.6.1 Ajustage de surface et modélisation paramétrique

Dans sa version initiale, la pyramide modélise le contenu de chaque région par une couleur moyenne et un écart-type. Bien que pratique et justifiable, cette modélisation classique, utilisée comme critère de



(a) $\sigma_M = 12$ (b) $\sigma_M = 15$ (c) $\sigma_M = 20$ (d) $\sigma_M = 29$

FIGURE 2.5 – Quelques résultats d'une pyramide multirésolution

fusion possède des limites pour représenter des régions de type dégradé ou texturé. Ici, nous nous sommes intéressés aux régions dégradées en nous inspirant des travaux de Besl et Jain [BJ88] qui segmentent des images de profondeur † en considérant l'image monochrome comme un relief. Le contenu de chaque région est ainsi modélisé par une surface d'ordre 1, 2, 3 ou 4 qui peut représenter un plan, une surface minimale, un sommet, un puits, une crête, une vallée, une selle crête ou une selle vallée. Les auteurs utilisent des polynômes bi-variés représentant une surface à l'aide de 3 à 15 paramètres en fonction de la complexité du relief de la région (équation 2.1).

$$\begin{aligned} f(n,\vec{a};x,y) = a_{00} + a_{10}x + a_{01}y + a_{11}xy + a_{20}x^2 + a_{02}y^2 + a_{21}x^2y + a_{12}xy^2 + a_{30}x^3 \\ + a_{03}y^3 + a_{31}x^3y + a_{22}x^2y^2 + a_{13}xy^3 + a_{40}x^4 + a_{04}y^4 \end{aligned} \tag{2.1}$$

n est l'ordre du modèle (1, 2, 3 ou 4) auquel correspond un vecteur de paramètres $\vec{a}$: 3 paramètres définissent une surface plane, 6 paramètres une surface bi-quadrique, 10 paramètres une surface bi-cubique et 15 paramètres une surface bi-quartique.

Dans leur algorithme, un étiquetage des pixels de l'image est réalisé selon le signe de courbure de la surface locale. Il fournit une partition grossière qui est ensuite érodée pour donner des germes utilisés dans une croissance de région basée sur la mise en correspondance de surfaces d'ordre approprié. Il est à noter que le résultat final n'est pas une segmentation en tant que telle puisqu'il y subsiste généralement un nombre plus ou moins important de pixels non étiquetés (trous).

En s'inspirant de ce modèle, nous désirons d'une part approcher au mieux le relief des régions mais également utiliser le modèle dans la pyramide irrégulière comme critère de fusion. Nous donnons ici les prémisses de ce travail qui nous paraît très prometteur. Notre but est triple :

- utiliser un critère de segmentation de plus haut niveau
- synthétiser l'image originale à partir d'une de ses segmentations
- proposer une nouvelle technique pour la compression d'image

2.6.2 Critère de similarité surfacique

Dans cette partie, nous remplaçons le critère de similarité classique de la pyramide irrégulière (couleur moyenne) par un critère de similarité surfacique. Le relief de chaque région est modélisé par une surface paramétrique définie par n paramètres (de $n = 3$ à $n = 15$ pour des types de surfaces plus ou moins simples). L'algorithme de la pyramide est inchangé ; seuls le calcul de ce nouvel attribut (n paramètres) pour chaque région ainsi que son utilisation comme critère de similarité entre régions sont différents :

- Le champ récepteur de chaque sommet s_i, interprété comme un relief, est approché aux moindres carrés par une surface.

†. range images

- Une région (c.-à-d. une surface) s_1 est considérée similaire à une région s_2 si leur réunion peut *convenablement* être représentée par une surface unique ; c'est-à-dire si l'approximation aux moindres carrés de la surface résultante $s = s_1 \cup s_2$ fournit une variance d'erreur calculée sur l'ensemble des pixels de s inférieure à la fois au seuil global T (fixé par l'utilisateur) et à un seuil local T_1 propre à s_1.
- Le calcul du seuil de similarité locale est réalisé à l'aide de l'ensemble des distances entre s_i et chacun de ses voisins : la distance entre s_i et s_j est égale à l'erreur d'approximation qui serait réalisée si l'union de s_i et de s_j était approchée aux moindres carrés par une surface commune. Pour un sommet donné, l'ensemble de toutes les distances à ses voisins permet de fixer un seuil local de similarité.

Pour plus de précisions, l'algorithme 4 donne la façon de construire le graphe de similarité correspondant. L'algorithme 5 quant à lui indique comment est calculée la similarité entre deux régions voisines.

Algorithme 4 : Construction du graphe de similarité par critère d'ajustage de surface (*surface fitting*)

Entrées :
$G_a = \langle S, A \rangle$, le graphe d'adjacence, (S l'ensemble des sommets, A l'ensemble des arêtes [a])
T, le seuil global d'erreur
n, l'ordre des surfaces utilisé
Sorties :
$G_s = \langle S, E \rangle$, le graphe de similarité, (S l'ensemble des sommets, E l'ensemble des arcs [b])

$E = \emptyset$;
`/* Calcul de la variance de l'erreur pour chaque sommet */`
pour chaque *sommet* $s_k \in S$ **faire**
  Calculer la variance d'erreur entre s_k et une surface d'ordre n ;
fin
`/* Calcul du seuil local pour chaque sommet */`
pour chaque *sommet* $s_k \in S$ **faire**
  pour chaque *voisin* s_i *de* s_k **faire**
    `/* appel de l'algorithme 5 */`
    $d_{ik} = \text{Distance}(s_i, s_k)$;
  fin
  Calculer le seuil local T_k à l'aide des d_{ik} ;
fin
`/* Mise en place des arcs du graphe de similarité */`
pour chaque *sommet* $s_k \in S$ **faire**
  pour chaque *voisin* s_i *de* s_k **faire**
    si $d_{ik} < T$ *ET* $d_{ik} < T_k$ **alors**
      $E = E \cup arc(s_k, s_i)$; // s_k est similaire à s_i
    fin
  fin
fin

a. G_a est un graphe non orienté, les liens entre sommets sont bi-directionnels
b. G_s est un graphe orienté, les liens entre sommets son uni-directionnels

Algorithme 5 : Distance(s_1, s_2) : Calcul de distance entre 2 régions par le critère d'ajustage de surface (*surface fitting*)

Entrées : deux sommets s_1 et s_2
Sorties : distance entre s_1 et s_2

$s = s_1 \cup s_2$;
$m = \text{card}(s)$;
B = matrice $m \times 1$ des niveaux de gris des pixels de s ;
A = matrice $m \times 1$ des coefficients du modèle multipliés par les coordonnées des pixels (cf. équation 2.1) ;
Résoudre l'équation linéaire aux moindres-carrées $AX = B$ pour trouver X ;
$Er = AX - B$;
retourner *variance(Er)* ;

2.6.3 Synthèse d'image et compression

Noous désirons seulement présenter ici des pistes pour une technique de compression réellement orientée objets ou tout au moins régions. Le traitement est réalisé pour des images couleurs par approche marginale : chaque composante R,V et B est traitée indépendamment. Une fois la segmentation terminée, chaque région est représentée par son support spatial et son relief. Le support spatial est simplement exprimé par les coordonnées d'un point de départ suivies d'une chaîne de codes de Freeman qui décrivent le contour. Le relief est donné par les coefficients modélisant la surface pour chacune des trois composantes couleur. La taille d'une image (exprimée en bits) est la somme des tailles des régions :

$$taille(I) = \sum taille(R_i) = \sum 32 + card(contour(R_i)) \times 2 + 3n \times 32 \tag{2.2}$$

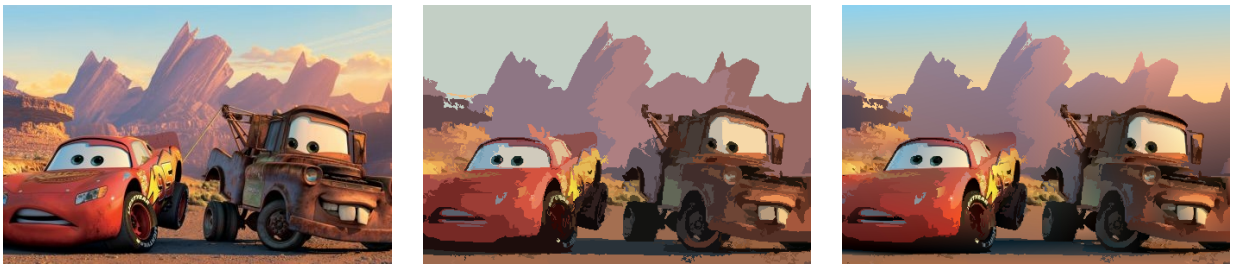
Les coordonnées de départ sont codées sur 32 bits (2 entiers de 16 bits). Chaque direction de Freeman nécessite 2 bits. Chacun des $3n$ coefficients est codé par un réel sur 32 bits.

Bien entendu, ce codage de base peut être sensiblement amélioré : en fonction de la taille des images, les coordonnées de départ nécessiteront moins de bits. Quant aux contours, un codage arithmétique peut faire descendre ce nombre entre 1.1 et 1.3 bits par pixel dans le cas d'un codage sans perte et jusqu'à 0.4 à 0.6 bit par pixel avec perte [PL97]. Enfin, il n'est pas nécessaire d'utiliser un codage en virgule flottante sur 32 bits pour les coefficients (a_{00} peut être assimilé à un entier sur 8 bits).

Il faut noter que dans les images que nous avons traitées, l'approximation avec des plans ($n = 2$) donne le meilleur compromis *temps d'exécution / qualité du résultat*. En effet, la modélisation de régions d'ordres supérieurs ne concerne que des zones de transition dans l'image qui sont peu propices à donner lieu à des régions.

Les figures 2.6.b et 2.6.c montrent des partitions obtenues et synthétisées par des couleurs moyennes et par des plans. On peut voir que le rendu est nettement amélioré par l'utilisation de plans qui s'inclinent en suivant les dégradés de l'image. La figure 2.7 compare deux profils couleurs caractéristiques correspondants. On peut voir dans la figure 2.7.b qu'une grande majorité des régions sont représentées par des dégradés importants.

Du fait de sa représentation hiérarchique / multirésolution, la pyramide irrégulière se prête bien à la scalabilité. En effet, la construction d'une seule pyramide fournit toute une gamme de rapports débit/distorsion (figure 2.8). On remarquera sur ces figures un rapport débit / distorsion assez faible. Effectivement, la modélisation par surface n'est souhaitable que pour des régions qui s'y prêtent (essentiellement les dégradés).

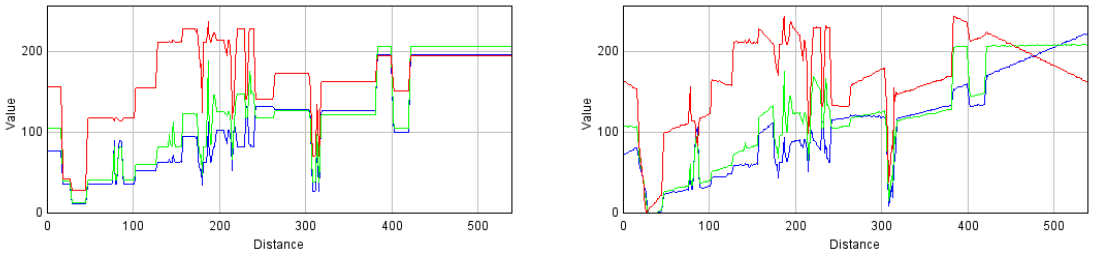


(a) Image originale

(b) Chaque pixel porte la couleur moyenne de sa région. PSNR= 23.47

(c) Chaque pixel appartient à une région modélisée par un plan orienté. PSNR= 26.02. Noter la douceur des dégradés

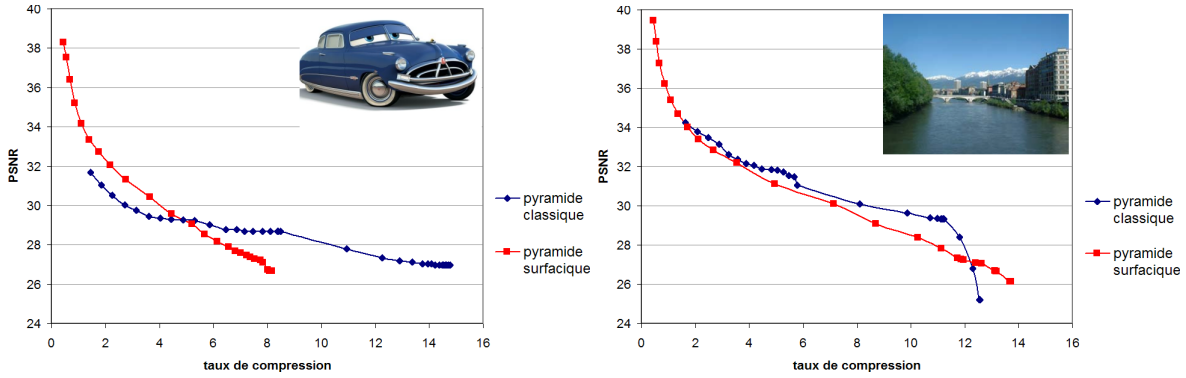
FIGURE 2.6 – Comparaison de partitions représentées en couleurs moyennes et sous forme de plans orientés



(a) Exemple de profil de la figure 2.6.b

(b) Exemple de profil de la figure 2.6.c

FIGURE 2.7 – Comparaison du profil couleur de la diagonale principale pour la pyramide classique et la pyramide surfacique



(a) Image de synthèse du film cars

(b) Photo de Grenoble

FIGURE 2.8 – Chaque courbe débit / distorsion est obtenue à partir d'une seule pyramide. Chaque point représente un niveau

2.7 Pyramide irrégulière locale

Dans cette section, je présente la pyramide locale qui est une pyramide irrégulière permettant de segmenter avec précision les contours d'objets préalablement extraits grossièrement.

2.7.1 Pyramide d'image irrégulière *vs* pyramide locale

Dans la pyramide irrégulière classique [MMR91], le traitement porte sur tous les pixels de l'image. Dans la pyramide locale que nous proposons, seul un nombre réduit de pixels de l'image sont identifiés à des sommets, tandis que les n composantes connexes restantes sont identifiées à n sommets (figure 2.9), et sont des *racines* (régions qui appartiendront au résultat final) tel que le fond par exemple. Pour réaliser cette focalisation sur une zone particulière de l'image, on supposera tout d'abord qu'on dispose d'un moyen pour délimiter grossièrement le contour de la zone à l'aide d'un ruban fermé (figure 2.10.a). Dans la pyramide locale que nous avons imaginée [BHF05], la segmentation n'est donc réalisée que localement.

2.7.2 Propagation des étiquettes

Une fois obtenu, le ruban induit une *trimap*, soit trois étiquettes : l'extérieur, l'intérieur et le ruban lui-même, supposé contenir le vrai contour de l'objet (figure 2.10.b). Les pixels de l'extérieur appartiennent au fond (le premier sommet racine) tandis que ceux de l'intérieur appartiennent à l'objet d'intérêt (un second sommet racine). Tous les pixels formant le ruban représentent une zone indéfinie. Ces derniers seront segmentés afin de fusionner avec l'une ou l'autre des racines.

(a) Exemple d'une initialisation de pyramide locale

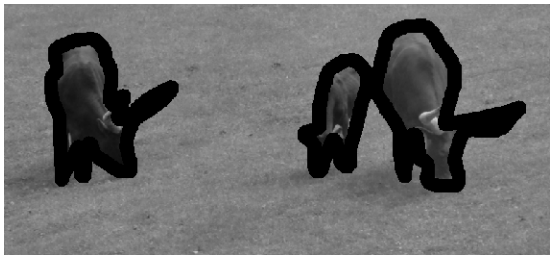
(b) Graphe correspondant à (a)

FIGURE 2.9 – Exemple d'initialisation d'une pyramide locale. Certains sommets sont des racines regroupant un nombre plus ou moins important de pixels

A la base de la pyramide locale, le graphe d'adjacence inclut un sommet pour chaque racine et un sommet pour chaque pixel de la zone indéfinie (figure 2.9.b). A chaque racine est associée une étiquette unique. Les sommets (i.e. pixels) de la zone indéfinie restent non étiquetés. Une couche plus ou moins fine de pixels de part et d'autre du ruban est séparée des racines et constitue autant de sommets portant l'étiquette de la racine correspondante (figure 2.10.d). Ce sont ces deux couches qui vont permettre (en fusionnant avec d'autres pixels) de propager les étiquettes des deux racines dans la zone indéfinie. Durant la construction de la pyramide locale, les fusions s'effectuent toujours selon le critère de similarité, mais respectent également des règles qui assurent la propagation cohérente des étiquettes à travers la zone indéfinie. Rajouter de nouvelles contraintes dans les règles de construction de la pyramide peut simplement être vu comme modifier le critère de similarité. Pour prendre en compte ces nouvelles contraintes, on modifie le critère vu en section 2.4 page 26 et on rajoute des règles concernant la propagation des étiquettes.

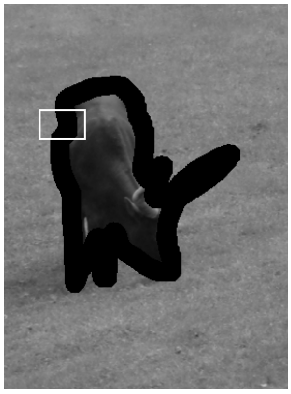
Par la suite, $R(l)$ est une région portant l'étiquette l et R est une région non étiquetée ; $\sim$ est le symbole de similarité.

- **Règle de similarité :** $R_i(l_i) \nsim R_j(l_j)$ si $l_i \neq l_j$ ou si $d(YUV(R_i), YUV(R_j)) \geq T$. Deux régions portant deux étiquettes différentes, ou trop dissimilaires, ne peuvent pas fusionner, ceci pour éviter la fusion de différents objets.
- **Règle de propagation 1 :** Si $R_i \sim R_j(l), R_i \cup R_j(l) = R_k(l)$. Une région non étiquetée fusionnant avec une région étiquetée l donne naissance à une région portant l'étiquette l.
- **Règle de propagation 2 :** Si $R_i(l) \sim R_j(l), R_i(l) \cup R_j(l) = R_k(l)$. La fusion de plusieurs régions d'étiquette l donne naissance à une région portant l'étiquette l.

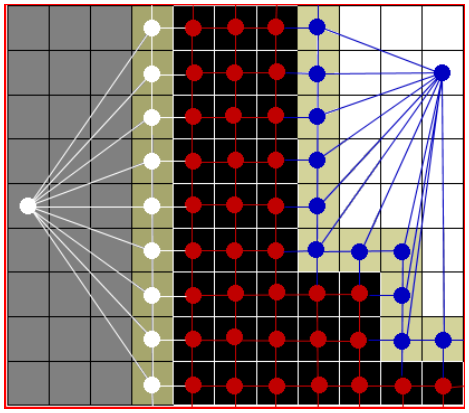


(a) Initialisation de régions d'intérêt

(b) Etiquetage initial des racines induites par (a). Les zones noires n'ont pas d'étiquette



(c) Le rectangle est zoomé dans (d)



(d) Graphe d'adjacence de la zone marquée dans (c)

FIGURE 2.10 – De l'initialisation manuelle d'une zone d'intérêt à la définition d'un graphe d'adjacence où sera effectuée la segmentation

- **Règle de propagation 3 :** Si $R_i \sim R_j, R_i \cup R_j = R_k$. Des régions non étiquetées fusionnant entre elles donnent naissance à une région non étiquetée.

Lorsque plus aucune fusion n'est possible et qu'il reste éventuellement des sommets non étiquetés, la règle de similarité est relâchée : $R_i(l_i) \nsim R_j(l_j)$ ssi $l_i \neq l_j$, $R_i(l_i) \sim R_j(l_j)$ sinon. La partition finale comporte alors autant d'objets que de racines.

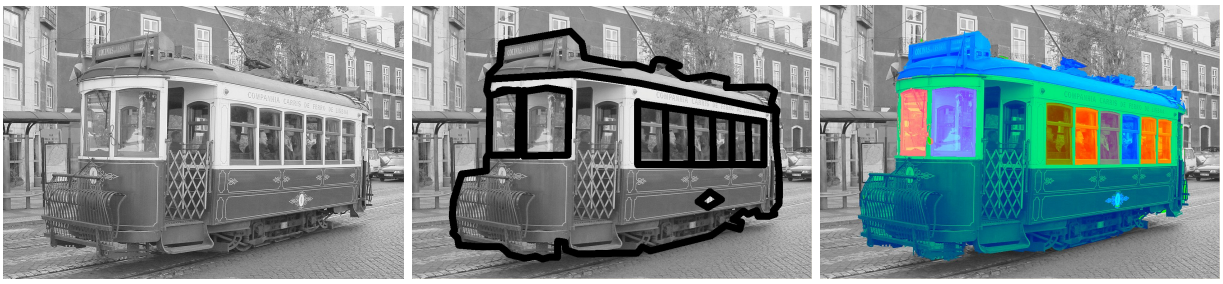
2.7.3 Discussion

Nous proposons une méthode bien adaptée aux images complexes où le contour d'un objet peut successivement prendre des configurations très différentes : changement de signe du gradient le long du contour, contours multiples. La figure 2.11 montre un résultat obtenu avec notre application ExtraK'Obs pour un objet qui se distingue assez difficilement du fond.

La texture du fond et/ou de l'objet est bien entendu un problème important auxquelles doivent faire face la majorité des méthodes de segmentation et notamment celle que nous présentons. La connaissance de présence de textures (objet ou fond) et de contours pourrait servir à favoriser certaines fusions lors de la propagation concurrente d'étiquettes. Il semblerait par exemple judicieux de favoriser la propagation dans les zones peu texturées qui pourraient alors s'étendre jusqu'à être stoppées par des zones plus texturées.

La localisation de la frontière finale n'est pas liée géométriquement à la localisation et à l'épaisseur du ruban. Ces deux paramètres peuvent néanmoins influencer le résultat final dans le sens où une localisation et une largeur mieux adaptées limiteront les erreurs de segmentation (faux contour, fuites, ...). La frontière finale correspond localement à la frontière entre deux régions qui ont crû puis acquis une étiquette. Ainsi la croissance de régions non encore étiquetées est un élément essentiel que nous devons étudier de manière

plus approfondie.



(a) Image originale (b) Initialisation (c) Partition obtenue

FIGURE 2.11 – Exemple de résultat obtenu avec l'application ExtraK'Obs

Le travail que nous avons présenté ici est préliminaire et laisse pressentir une méthode puissante associée à une interaction qui fournit à un utilisateur un outil permettant une extraction facile et précise d'objets. Cette première fonctionnalité peut également être étendue en générant de façon automatique les zones d'incertitude par un pré-traitement des images. La section suivante explique cette automatisation.

2.8 Initialisation de pyramide locale par carte d'homogénéité

Pour initialiser automatiquement la trimap de la pyramide locale, nous utilisons dans [HB05a, HB05b] une méthode présentée dans [JLZZ03] qui réalise une analyse de l'homogénéité dans les images couleur. Contrairement aux auteurs qui calculent l'image d'homogénéité (ou ≪H-image≫) selon les composantes RGB, nous utilisons l'espace couleur HSV ‡ puisqu'il fournit moins de fausses homogénéités. La H-image est une image en niveaux de gris dont les fortes valeurs correspondent à d'éventuelles discontinuités tandis que les faibles valeurs correspondent à des régions homogènes.

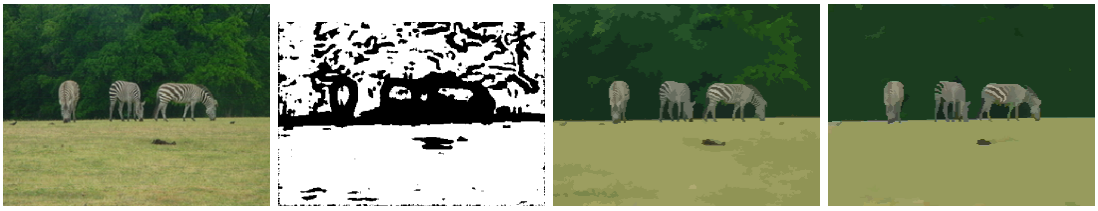
Ensuite, une classification des pixels de la H-image produit une partition binaire comprenant des composantes connexes homogènes et hétérogènes (figure 2.12.b). Les auteurs de [JLZZ03] effectuent une croissance de régions à partir de germes sélectionnés dans les zones homogènes. En ce qui nous concerne, ce masque binaire initialise les racines et les zones indéfinies nécessaires à la segmentation locale (fig. 2.12.c). Les composantes connexes en blanc sont les racines. Les pixels des composantes noires sont amenés à fusionner ensemble et/ou avec une racine voisine selon le critère de similarité.

En comparaison avec le résultat généré par une pyramide classique (fig. 2.12.c), le résultat obtenu avec une segmentation locale initialisée par le masque d'homogénéité (fig. 2.12.d) est plus approprié, puisqu'il comporte moins de régions mais conserve la même précision. Notons que dans les deux cas, un seuil identique a été utilisé. Deux planches de résultats sont visibles en annexe aux figures E.1 et E.2 aux pages 143 et 144.

2.9 Segmentation interactive par pyramide locale

Cette section montre qu'en combinant une pyramide locale et une interface graphique adaptée, il est possible de proposer un outil interactif pour réaliser du détourage dans une application de retouche d'images comme GIMP ou Photoshop.

‡. *Hue, Saturation, Value* ou Teinte, Saturation, Luminance



(a) Image originale (b) Masque d'homogénéité HSV (c) Segmentation par pyramide classique (d) Segmentation locale

FIGURE 2.12 – Comparaison entre segmentation globale et segmentation locale automatique

La segmentation d'images naturelles précise et efficace peut rarement être effectuée automatiquement ; c'est pourquoi le recourt à la segmentation interactive [BJ01, RKB04, BRB+04, ZMM99, MB95] est souvent nécessaire dans de nombreuses applications : son but est d'extraire un ou plusieurs objets d'intérêt en proposant à l'utilisateur un outil simple et pratique à utiliser fournissant des résultats facilement exploitables.

Les contours actifs [KWT88, JZDJ98] sont des méthodes très classiques qui nécessitent une initialisation avec un contour grossier à l'intérieur ou à l'extérieur de l'objet. Malheureusement, les *snakes* sont généralement très sensibles aux minima locaux et assez difficiles à paramétrer. La ligne de partage des eaux [BM92] quant à elle, fournit une sur-segmentation de l'image. Dans [CB01], les auteurs proposent un traitement interactif où les marqueurs sont sélectionnés par des clics avec la souris. Dans [ZMM99], la LPE fournit une segmentation multi-échelle dans laquelle l'utilisateur peut sélectionner des régions d'intérêt. Dans [JGS99], après un seuillage automatique, les contours des régions sont représentés par des courbes paramétriques. Leurs points de contour peuvent être édités par l'utilisateur. Les ciseaux intelligents ont été développés par [MB95] pour la composition. La méthode détecte un contour d'objet potentiel proche du pointeur de souris et trouve le meilleur chemin partant d'un pixel de départ.

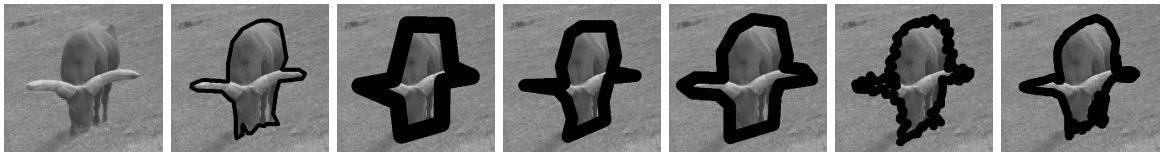
Plus récemment, de nombreuses variantes et améliorations ont été apportées au *graph cut* [BJ01]. L'utilisateur marque de 2 étiquettes différentes des parties colorimétriquement représentatives de l'objet et de l'arrière plan. Le partitionnement final est obtenu en minimisant une énergie qui prend en compte à la fois le coût d'appartenance d'un pixel (ou d'une région) au fond et à l'objet, et le coût d'une frontière entre deux régions, l'une étant dans l'objet, l'autre dans le fond.

Dans notre approche [BHF05], une croissance de régions est appliquée uniquement près du bord de l'objet indiqué grossièrement par l'utilisateur à la souris à l'aide d'un "ruban fermé". Dans ce ruban, une segmentation est réalisée à partir d'une pyramide irrégulière locale. Cette approche est peu contraignante pour l'utilisateur et on remarque qu'une grande variabilité lors de l'initialisation se traduit par une faible variabilité des résultats (figure 2.13). La méthode permet d'extraire en un seul traitement un nombre quelconque d'objets avec une bonne précision.

Il est naturel que la méthode fonctionne bien dans le cas d'objets homogènes sur un fond homogène. En revanche, ce qui est beaucoup plus intéressant c'est de noter que la méthode fonctionne bien dans l'hypothèse où les objets ont une colorimétrie hétérogène sur un fond relativement homogène (ou inversement), même lorsque le gradient s'inverse à la frontière entre le fond et l'objet. Les dégradés sont également bien pris en compte. Cette approche est intéressante notamment lorsque les objets ont une forme complexe.

Les résultats présentés ne sont pas post-traités, et sont obtenus avec le même seuil de similarité T. La figure 2.13.h montre comment il est souvent impossible avec des méthodes simples (ici un seuillage) d'extraire correctement un objet, soit à cause de sa texture, de son ombre portée ou du manque de contraste localement. On peut remarquer que le contour du dos de l'animal (figure 2.13.a) n'est pas du tout contrasté, ni en couleurs, ni en niveaux de gris.

Les résultats (figures 2.13) montrent que notre méthode est peu sensible à la façon dont l'utilisateur



(a) Image originale (b) Largeur = 10 pixels (c) Largeur = 40 pixels (d) Ruban intérieur (e) Ruban extérieur (f) Ruban bruité (g) Ruban irrégulier

(h) Pour référence, seuillage de (a) (i) Résultat de (b) (j) Résultat de (c) (k) Résultat de (d) (l) Résultat de (e) (m) Résultat de (f) (n) Résultat de (g)

FIGURE 2.13 – Une variabilité importante de l'initialisation : épaisseur, positionnement, régularité (première ligne) entraîne une faible variabilité des résultats (deuxième ligne)

initialise le traitement : en effet, une variabilité importante de l'épaisseur du ruban, de son positionnement (plutôt intérieur ou extérieur), de sa forme (plus ou moins régulière), fournit des résultats très similaires. Le ruban peut être tracé soit à l'aide de segments (figures 2.13.b,c,d,e), soit à main levée (figure 2.13.f,g).

Les figures 2.13.b et c présentent deux résultats obtenus avec deux épaisseurs différentes (10 et 40 pixels). Le ruban de la figure 2.13.c est obtenu avec 10 clics de la souris seulement ; en revanche, il fournit des détails précis de la tête et des jambes de l'animal.

Dans la figure 2.13.d, la segmentation est obtenue avec un ruban positionné plutôt dans l'objet alors que dans la figure 2.13.e, il est plutôt dans le fond. Bien que ces résultats ne soient pas exactement identiques, ils diffèrent assez peu.

Dans les figures 2.13.f et g, on a joué sur la régularité de la forme du ruban : l'utilisateur peut effectivement dessiner des "pâtés" qui contiennent de nombreux détails de contour afin de les extraire correctement, comme le montre la jambe arrière dans les figures 2.13.m et n.

Enfin, la méthode présentée peut être utilisée telle quelle, sans modification, avec un nombre quelconque de racines et un ruban fermé peut avoir une topologie quelconque et être connexe à un nombre quelconque de racines, comme le montre l'exemple de la figure 2.14. L'approche proposée garantit un nombre d'objets final égal au nombre de racines. Il est donc important que le ruban soit fermé pour que l'intérieur puisse se distinguer de l'extérieur.

2.10 Des régions vers les objets

Dans cette section, nous proposons une approche qui simplifie un résultat de segmentation en regroupant les régions en objets. Pour ce faire, nous utilisons des critères de plus haut niveau. L'outil utilisé est toujours la pyramide irrégulière.

2.10.1 Introduction

Nous traitons ici les problèmes relatifs à une méthode générique et automatique d'extraction d'**objets** dans les images (voir [MLT99] pour un récapitulatif des méthodes existantes). Afin de proposer une

(a) Image originale (b) Initialisation (c) Partition obtenue

FIGURE 2.14 – Segmentation de plusieurs régions d'intérêt

nouvelle méthode non fondée sur la connaissance *a priori* du contenu sémantique de l'image ou sur un modèle quelconque d'objet, plusieurs méthodes efficaces sont intégrées et interviennent successivement dans la pyramide de graphe irrégulière : (1) Une **analyse locale de l'homogénéité** de l'image est effectuée pour initialiser une segmentation locale et ainsi éviter une sur-segmentation. (2) La pyramide de graphe réalise une **segmentation locale** des zones hétérogènes de l'image. En utilisant un critère de similarité, elle génère un empilement de partitions précises. (3) La pyramide est de nouveau utilisée sur les régions issues de la segmentation pour un traitement de **groupement perceptuel** selon des critères issus de la théorie du Gestalt. Ces critères sont bien adaptés à une méthode n'utilisant pas de modèle puisqu'ils prennent en compte uniquement la pertinence visuelle des régions.

2.10.2 Groupement de régions orienté perception

Principe

Nous appelons groupement perceptuel le fait de fusionner plusieurs régions sur des critères perceptuels. Lors du traitement de groupement perceptuel, deux contraintes doivent être respectées : premièrement, seuls les meilleurs groupements locaux doivent être retenus ; ce qui signifie qu'un maximum de combinaisons de régions doit être étudié (parmi deux, trois, quatre, ..., n voisins). Deuxièmement, le résultat ne doit pas être influencé par l'ordre des groupements.

La pyramide irrégulière a été choisie afin de réaliser l'étape de groupement pour trois raisons principales. Premièrement, sa structure de graphe est bien adaptée à la manipulation en parallèle (i.e. indépendante) de régions. Deuxièmement, les critères de groupement de régions sont facilement interchangeables. Enfin, les itérations du traitement sont simplement obtenues par génération de niveaux supplémentaires résultant des fusions entre régions.

Le graphe final de la pyramide locale constitue le graphe initial de la pyramide de groupement. En effet, la pyramide locale est étendue avec des niveaux supplémentaires induits par le groupement de régions.

Dans [LeG03] les auteurs groupent seulement des paires de régions. Contrairement à leur travail, avec notre méthode, un nombre quelconque de régions peut fusionner simultanément en un seul groupement. Cela fournit plus de choix dans la stratégie de groupement et donc, plus d'adaptativité au contenu de l'image.

Critères de groupement

Les critères choisis pour effectuer le groupement sont dérivés de la théorie du Gestalt [Wer58] qui n'utilise aucun modèle d'objet. La vision humaine effectue des groupements indépendants (appelés Gestalt) fondés sur cinq propriétés principales : la proximité, la similarité, la fermeture, la continuité et la symétrie [ZTB04].

Des énergies sont extraites de ces propriétés et sont calculées pour des régions ou des groupements de régions. Le but est de sélectionner les groupements de plus faibles énergies représentant une forte pertinence visuelle. Le coût d'un groupement est composé de plusieurs fonctions d'énergie proposées par [LeG03].

E_{fusion} est le coût de l'opération de fusion fondé sur la différence des moyennes des composantes Lab, et sur l'étude des jonctions (continuité des contours) des différentes régions du groupement.

E_{region} est le coût de la région résultant d'une fusion. il peut être considéré comme le degré de pertinence du groupement potentiel (plus l'énergie est faible, plus le degré de pertinence est important). Ce coût est fondé sur la compacité, la convexité et l'aire du groupement.

La fonction d'énergie d'une région résultant d'un groupement est donnée par $E = E_{fusion} + E_{region}$. Une énergie faible indique un fort intérêt visuel. Au contraire, une forte valeur indique une région ou un groupement indésirable. Le but étant de réaliser le groupement qui assure la plus faible énergie localement.

2.10.3 Choix des meilleurs groupements

Sélection du meilleur groupement local Soit v_c un sommet, $c \in [\![1, N]\!]$ et n_c le nombre de ses voisins. Tous les groupements incluant v_c et les différentes combinaisons de ses voisins sont considérés. Le nombre de combinaisons est donné par la formule suivante :

$$C(v_c) = \sum_{j=1}^{n_c} C_{n_c}^{j} \tag{2.3}$$

$C_{n_c}^{j}$ étant le nombre de combinaisons de j voisins parmi n_c.
E_{fusion} et E_{region} sont calculées pour chacun de ces groupements.

Soit g_c le groupement incluant v_c ayant la plus faible énergie $E(g_c)$. g_c est un groupement potentiel si : (1) g_c améliore localement l'énergie de la partition, (2) $E(g_c)$ indique une forte pertinence visuelle. Si ce n'est pas le cas, g_c n'est pas retenu.

Notons que dans nos expérimentations, le nombre maximum de voisins par combinaisons est limité à 5 ou 6, ce qui donne respectivement $C(v_c) = 31$ ou $C(v_c) = 63$.

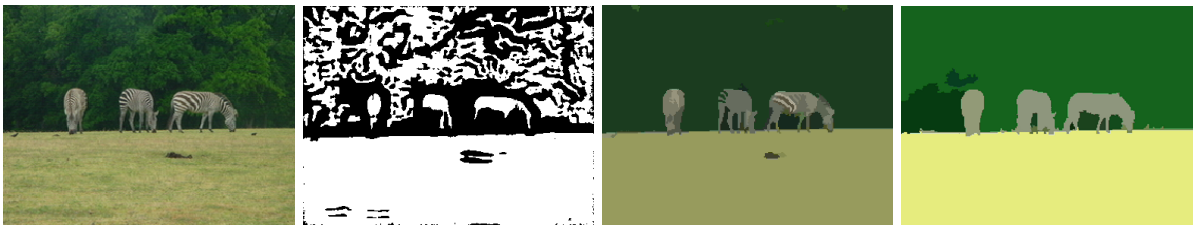
Sélection du meilleur groupement global Un ensemble G de groupements potentiels est à présent défini sur toute l'image. Les groupements effectivement réalisés sont sélectionnés dans G par ordre croissant des énergies. Lorsqu'un groupement g_s est sélectionné, tout groupement de G qui intersecte avec g_s est exclu. Ainsi, la fusion de chaque groupement sélectionné peut être correctement réalisée. Ces fusions engendrent, dans la pyramide, un niveau supplémentaire correspondant à la nouvelle partition.

Cette sélection assure les meilleurs groupements dans l'image entière. Le traitement de groupement est réitéré jusqu'à ce que le nombre de sommets reste stable.

2.10.4 Résultats

Les différents résultats obtenus avec cette méthode sont présentés dans les figures 2.15, 2.16 et 2.17. Pour des images complexes dont la dimension est d'environ 300×300 pixels, la segmentation locale génère habituellement une partition de 100 à 200 régions. De cette partition, l'étape de groupement donne une partition de moins de 20 régions. En général, l'étape orientée similarité converge en moins de 100 niveaux et l'étape orientée perception s'étend seulement sur 10-15 étages supplémentaires. L'aspect hiérarchique de la pyramide constitue un grand avantage car lorsque dans les derniers niveaux de la pyramide des objets sémantiques sont perdus, l'utilisateur peut facilement parcourir la pyramide afin de les récupérer. C'est le cas dans la figure 2.15.d qui représente une partition comportant 13 régions, qui définit avec une bonne précision les animaux.

La figure 2.17 illustre le fait qu'augmenter le nombre de voisins par groupement peut aider à faire de meilleurs choix dans les groupements. Mais cela augmente considérablement le temps de calcul et la partition finale contient approximativement le même nombre de régions.



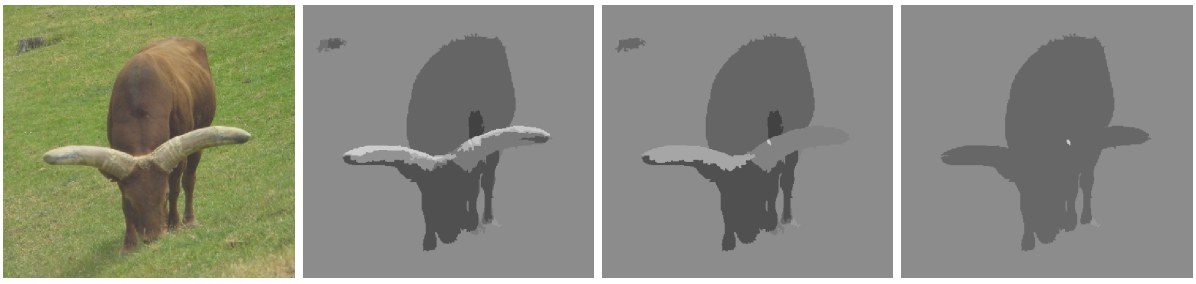
(a) Image originale

(b) Masque d'homogénéité Lab

(c) Segmentation locale

(d) Un résultat intermédiaire de groupements

FIGURE 2.15 – Différents niveaux de segmentation avec plusieurs objets d'intérêt



(a) Image originale

(b) Résultat de la pyramide locale

(c) Groupement intermédiaire

(d) Groupement final

FIGURE 2.16 – Différents niveaux de la segmentation d'une image comportant un objet d'intérêt

(a) Image originale

(b) Utilisation de 2 régions maximum par groupement

(c) Utilisation de 4 régions maximum par groupement

(d) Utilisation de 6 régions maximum par groupement

FIGURE 2.17 – Résultats obtenus avec des groupements locaux pour des nombres de régions maximum différents

2.11 Optimisation par ligne de partage des eaux

Comme on peut s'en douter, la pyramide irrégulière n'est pas la panacée pour la segmentation d'images. Elle induit certains problèmes que j'ai récemment abordés. Avec le recul et compte-tenu des techniques de segmentation qui se sont améliorées ces quinze dernières années, voici une critique des points faibles de la pyramide irrégulière et quelques réponses apportées.

2.11.1 Les faiblesses de la pyramide irrégulière

Les quatre premiers points sont relatifs à l'utilisation de la pyramide et aux résultats qu'elle fournit. Ils peuvent donner lieu à des améliorations. Le dernier point est d'ordre plus général et en soit est une limite intrinsèque à cette technique :

- **Un temps de construction élevé** : les sommets d'un graphe sont accédés en lecture / écriture pour de nombreux traitements ainsi que tous leurs voisins. La construction d'une pyramide qui généralement comporte quelques dizaines à plus d'une centaine de niveaux est loin d'être réalisée en temps réel ni en quelques secondes pour des images de taille classique.
- **Des ressources mémoires nécessaires élevées** : un sommet de pyramide occupe 96 octets pour stocker les attributs de la région (couleur, surface, sommet père, seuils locaux, ...). A la base, chaque sommet possède 8 voisins (8 pointeurs = 32 octets). D'autre part, chaque élément de champ récepteur est un pointeur (4 octets). Pour une image 1000×1000 pixels, il faut au minimum $10^6 \times (96 + 32 + 4)$ octets ($1,32 \times 10^8$ octets) pour stocker le premier niveau de la pyramide, soit environ 126 Méga octets. Dans les premiers niveaux de la pyramide, le facteur de réduction est d'environ 2, ce qui divise approximativement par deux la place mémoire occupée par un nouveau niveau par rapport au précédent.
- **Des niveaux inutiles** : la construction complète de la pyramide génère un nombre de niveaux qui dépend à la fois du contenu de l'image et de ses dimensions. Lorsque l'image est complexe (nombreux détails, textures, objets de formes complexes), la convergence peut être assez lente : une fusion au niveau k peut déclencher localement une fusion au niveau $k+1$, qui jusque là n'était pas possible, et ainsi de suite par réaction en chaîne. En conséquence, les niveaux élevés successifs ont un nombre de régions très proche et sont très similaires. La lenteur de cette convergence est gênante car elle rallonge inutilement des temps de traitement et utilise un surcroit de mémoire important. En outre, lorsque la pyramide est utilisée pour être parcourue interactivement dans sa hauteur, sa partie supérieure est inutile, voire gênante.
- **Des petites régions non significatives** : la décision de fusion qui est prise localement est souvent un handicap aux frontières de régions qui sont contrastées mais qui pour des raisons d'artéfact de numérisation ou de phénomène d'ombre sont en réalité des transitions étalées sur un petit nombre de pixels (un, deux ou trois) comme dans la figure 2.18. Ces régions croissent préférentiellement perpendiculairement au gradient, le long du contour et lorsqu'elles ne peuvent plus s'étendre, il leur est impossible de fusionner d'un côté ou de l'autre du contour car elles sont trop dissemblables de leurs régions voisines. Ces petites régions ne sont pas significatives de la scène mais complexifient le graphe. Divers post-traitements peuvent être envisagés pour les supprimer sans toutefois constituer une solution satisfaisante.
- **Pas de minimisation d'énergie** : ce dernier point est à la fois un point fort et un point faible de la méthode. Les décisions de fusion sont prises localement et indépendamment dans l'image. Cela permet de prendre en compte sur un même plan tout le contenu de l'image, avec ses détails. En revanche, on ne dispose pas d'un critère fondé sur la minimisation d'une énergie globale de la partition, comme c'est le cas dans les approches de type *graph-cut*.

(a) Image originale avec la zone étudiée en (b) encadrée

(b) Zoom sur les petites régions qui n'arrivent pas à fusionner

FIGURE 2.18 – Le problème des petites régions localisées aux frontières d'objets contrastés

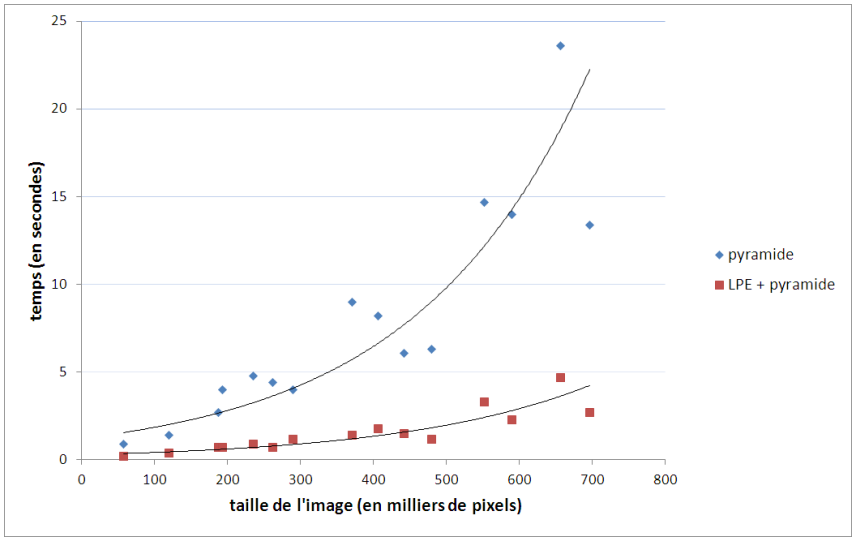
2.11.2 Accélération de la pyramide par ligne de partage des eaux

Pour apporter une réponse simple et efficace à la majorité des problèmes cités ci-dessus, nous proposons un pré-traitement classique qui consiste à effectuer une ligne de partage des eaux (LPE) sur l'image originale et à initialiser le graphe d'adjacence avec les régions obtenues avec la LPE. L'algorithme de ligne de partage des eaux retenu est celui de [ORGLSLV07] qui fournit une partition complète de bassins versants sans ligne de partage des eaux à proprement parler. Ainsi, tout pixel de l'image appartient à une région. Nous en avons fait une implémentation très rapide en C et son temps d'exécution ne fait pas perdre le bénéfice de l'accélération qu'il procure par ailleurs (la segmentation LPE est de l'ordre de $0,6$ seconde pour une image d'un Méga pixels sur un PC doté d'un micro-processeur Intel Core 2 Duo cadencé à 2,4GHz). Les bassins versants obtenus avec la LPE ainsi que leur voisinage sont aisément convertibles pour créer et initialiser un graphe d'adjacence (irrégulier) qui jusque là était obtenu à partir de la grille régulière des pixels de l'image.

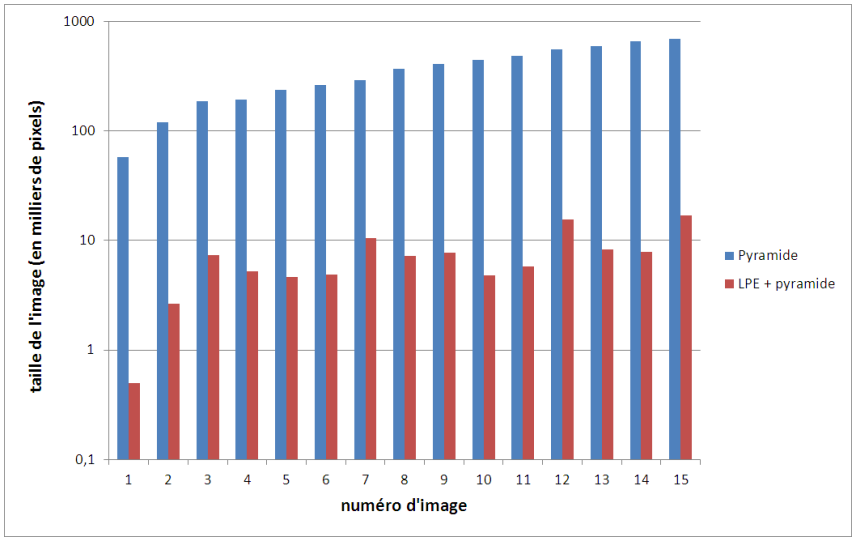
Nous indiquons brièvement ici l'effet bénéfique de l'utilisation des bassins versants de la LPE pour l'initialisation de la pyramide, concernant les quatre premiers points de la section 2.11.1 :

- La pyramide construite à partir de la partition de la LPE sur un ensemble d'une quinzaine d'images de tailles réparties entre 57.000 et 700.000 pixels montre un temps de traitement moyen divisé par 5. La figure 2.19.a montre ces résultats et les régressions exponentielles associées afin d'indiquer les tendances.
- Le nombre moyen de régions à la base de la pyramide est divisé par 58 (figure 2.19.b) pour des résultats (qualité, nombre de régions) comparables (figure 2.20). Pour reprendre l'exemple de l'image 1000×1000 donné en 2.11.1, la mémoire requise correspondante pour la base est donc de $\frac{10^6}{58} \times (96+32) + 10^6 \times 4$ octets $= 5,9$ Moctets (contre 126 Moctets précédemment) soit une amélioration d'un facteur 21 (l'amélioration ne porte pas sur la taille des champs récepteurs qui reste identique).
- Le nombre de niveaux moyen passe de 106 à 67. Cette diminution est due en partie aux régions de départ de taille supérieure à un pixel mais aussi à une moins grande sensibilité aux fusions en cascades. Ceci est sans doute dû à des régions plus compactes, moins bruitées en ce qui concerne leur forme, en relation avec le point suivant.
- Les petites régions non significatives sont beaucoup moins nombreuses car la LPE empêche la création de petites régions allongées le long des contours.

Les traitements sont réalisés sur un PC doté d'un micro-processeur Intel Core 2 Duo cadencé à 2,4GHz



(a) Temps de construction des pyramides



(b) Nombre de régions à la base de la pyramide (échelle logarithmique)

FIGURE 2.19 – Comparaison entre pyramide seule et LPE + pyramide sur une quinzaine d'images de tailles originales comprises entre 57.000 et 700.000 pixels

avec 3,49Go de RAM. Nous avons réalisé deux implémentations de cette technique : un pluggin GIMP sous Linux et une application sous Windows.

2.12 Conclusion

Dans ce chapitre, nous avons étudié les grands principes de la pyramide irrégulière. Diverses utilisations dans les images fixes ont été passées en revue : obtention de partitionnement multirésolutions, segmentation de l'image complète, segmentation locale. Nous avons montré que cette pyramide possède une grande souplesse notamment en ce qui concerne les critères de similarité divers qui peuvent être utilisés.

Néanmoins, certains inconvénients de la méthodes originelle existent et nous les avons listés. Ceux-ci

(a) Segmentation par pyramide seule (428 régions) (b) Segmentation par LPE + pyramide (438 régions)

FIGURE 2.20 – Comparaison entre une segmentation par pyramide et par LPE + pyramide. On peut voir en (b) plus de détails et moins de petites régions parasites dues aux dégradés et aux transitions douces

sont grandement améliorés avec une initialisation par ligne de partage des eaux.

Le prochain chapitre montrera en partie comment cette méthode peut être avantageusement utilisée pour réaliser de la segmentation spatio-temporelle.

Chapitre 3

Segmentation spatio-temporelle

Sommaire

3.1 Introduction : les besoins du marché

La segmentation spatio-temporelle consiste à segmenter l'image au cours du temps, dans un plan séquence de vidéo. C'est non seulement un suivi (*tracking*) d'un ou plusieurs objets, mais aussi et surtout une localisation précise de leurs limites ou contours.

De nombreuses applications ont comme besoin générique de segmenter spatio-temporellement des vidéos. Illustrons avec deux exemples diamétralement opposés : d'une part la vidéo-surveillance qui doit contrôler le comportement des personnes avec des contraintes de temps réel et d'autonomie. D'autre part la publicité interactive où les objets de la scène doivent réagir ou changer d'aspect au passage du curseur de la souris. Cette application nécessite des traitements lourds, supervisés et en différé (*offline*). Cette forme de publicité n'en est qu'à ses balbutiements mais devrait bientôt générer un marché énorme, essentiellement basé sur tous les produits que le spectateur peut voir pendant la diffusion d'un film (habillement, ameublement, tourisme, services, ...). Actuellement, le travail de détourage des objets est bien souvent réalisé manuellement ou semi-automatiquement à l'aide de produits du marché : GrowCut de iPhotoSoft, Smox Editor de Manalee, PatchMaker de Pixmart. Le passage à une grande échelle de ce nouveau marché ne peut être réalisé qu'avec des outils de production efficaces, permettant de réduire l'intervention manuelle.

Fonctionnellement parlant, on peut distinguer le cas d'une caméra fixe et celui d'une caméra au mouvement quelconque. On peut aussi distinguer la segmentation d'un ou plusieurs objets particuliers de celle de l'image entière.

Dans ce chapitre, je présente notre contribution à ces différents aspects du problème. Tout d'abord avec une technique qui a fonctionné en conditions réelles pour une application d'un projet européen. Deuxièmement, avec une proposition de segmentation spatio-temporelle de toute l'image. Ensuite, avec une chaîne complète pour segmenter des objets d'intérêt. Enfin, j'aborderai l'aspect interface utilisateur avec un environnement original que j'ai développé.

3.2 Segmentation supervisée

Dans cette section, je présente une technique d'extraction de personnes à base d'actualisation d'image de référence dans le cas d'une caméra fixe.

Ce travail [BFP01a, BFP01b] s'est inscrit dans le projet Européen (IST 10942) ARTLIVE : ARchitecture and authoring Tools for prototype for Living Images and new Video Experiments, entre 2000 et 2002.

Les partenaires :

- Traitement d'images : UCL (Belgique - Responsable du projet : Benoit Macq), CSELT (Italie), ADERSA (France), UJF-LIS (France), EPFL (Suisse), Fastcom (Suisse)
- Intelligence artificielle : ADETTI (Portugal), UJF-TIMC (France)
- Auteurs multimédia : Casterman (France)

Ce projet avait pour objectif de développer un environnement permettant aux artistes/utilisateurs de créer des espaces narratifs combinant le monde réel et le monde virtuel (par exemple, intégration de personnages réels dans un décor de bande dessinée). Nous avons travaillé sur les aspects extraction de personnages en mouvement, mise à jour d'images de référence, suivi de personnages en mouvement. Pour être plus précis, l'objectif de ce projet est d'incruster en temps réel des personnes filmées "dans la rue", dans des images du domaine de la bande dessinée et de faire interagir ces personnes avec l'environnement de la BD, selon un scénario préconçu. Cette application, qui oriente ses scénarii dans les domaines du jeu et de l'enseignement doit pouvoir fonctionner avec le minimum de contrôle pendant toute une journée. La qualité de l'incrustation dépend en grande partie de l'extraction temps réel et de la qualité des masques des personnes qui passent ou s'arrêtent dans le champ de la caméra qui est fixe. L'environnement peut être soit un stand soit une scène d'extérieur. Dans les deux cas, un grand écran permet de projeter aux passants (à la fois acteurs et spectateurs) leur image réelle dans le monde de la BD. Aucun dispositif spécial (blue screen dans le fond, capteurs ou marques sur les personnages) ne permet de réaliser la segmentation dans des conditions optimales. L'une des principales contraintes imposées par ce projet est le respect du "temps réel" (au minimum 8 images 352x288 pixels par seconde). Les traitements utilisés doivent donc être simples mais efficaces.

Dans le cadre de ce projet, deux démonstrateurs temps-réels ont été mis en œuvre et testés lors de manifestations publiques (exposition "Les Jardins et la bande dessinée" en avril 2001 à Paris et démonstration à Arc-et-Senan en novembre 2001).

La section suivante correspond à la construction du masque représentant la personne. Celui-ci est obtenu par combinaison de deux opérateurs pour être moins sensible à la présence d'ombres dans la scène. La seconde partie présente la gestion de l'image de référence, qui permet d'extraire toute personne mobile ou immobile dans la séquence vidéo. Finalement des résultats sont présentés et commentés.

3.2.1 Construction des masques

La caméra étant fixe, une solution simple consiste à utiliser une image représentant la scène en l'absence de tout individu. L'utilisation de cette image, communément appelée image de référence [DHA88, RE95], rend immédiate la détection de présence d'une personne. Nous considérons dans un premier temps que cette image de référence est disponible, nous présenterons au cours de la partie suivante la manière dont cette image est obtenue.

Combinaison d'un masque région et contour

Une approche commune [Wen83] consiste à calculer la différence D entre l'image courante I et l'image de référence I_{ref} pixel par pixel. Cette image différence D est alors seuillée pour former un masque.

D'autres approches [VMBP96] effectuent le même calcul à partir de l'image gradient I' de l'image courante et de l'image gradient I'_{ref} de l'image de référence. Plus récemment, dans [JDWR00], les auteurs utilisent conjointement l'information couleur et contour.

Afin d'être peu sensibles à la présence d'ombre, nous combinons l'information de niveau de gris et l'information contour : pour chaque image de la séquence, deux masques M_1 et M_2 sont calculés, ils sont alors combinés par un OU logique pour fournir un seul masque M. Pour chaque pixel de coordonnées (x, y) à l'instant t, nous avons :

$D_1(x, y, t) = |I(x, y, t) - I_{ref}(x, y, t)|$
$D_2(x, y, t) = |I'(x, y, t) - I'_{ref}(x, y, t)|$
si $D_1(x, y, t) \geq \lambda_1$ alors $M_1(x, y, t) = 1$ (masque)
sinon $M_1(x, y, t) = 0$ (fond)
si $D_2(x, y, t) \geq \lambda_2$ alors $M_2(x, y, t) = 1$ (masque)
sinon $M_2(x, y, t) = 0$ (fond)

λ_1 et λ_2 sont deux seuils de décision. Leur valeur est comprise dans l'intervalle $[0, 255]$ puisque nous traitons des images en niveaux de gris. Pour calculer les images gradient I' et I'_{ref}, les opérateurs de Prewitt, à la fois rapides et robustes face au bruit, sont utilisés.

Le masque région M_1 est assez sensible à la présence d'ombres dans la séquence. En effet, si on utilise un seuil λ_1 trop faible, l'ombre de la personne apparaît dans le masque (fig. 3.1.b et 3.1.c). Ces masques montrent également que les zones peu contrastées ne sont pas extraites correctement. La construction du masque contour M_2 se montre moins sensible aux ombres car celle-ci n'est pas détectée lors du calcul du gradient. La combinaison du masque M_1 (fig. 3.2.a) avec le masque contour M_2 (fig. 3.2.b) permet de renforcer le contenu du masque de la personne tout en restant insensible à l'ombre (fig. 3.2.c). Les seuils suivants ont été utilisés : $\lambda_1 = 20$, $\lambda_2 = 5$.

(a) image originale

(b) masque obtenu avec un seuil faible

(c) masque obtenu avec un seuil supérieur

FIGURE 3.1 – Problème du seuillage avec le masque région

Pré et post-traitement

Pour limiter l'influence du bruit d'acquisition, toutes les images traitées (I, I', I_{ref} et I'_{ref}) sont pré-traitées avec un filtre moyenneur 3×3.

Comme le montre la figure 3.2.c, le masque obtenu par combinaison possède des trous. C'est pourquoi nous appliquons à ce masque un post-traitement en trois étapes. La première relie les pixels extraits suivant la verticale sous certaines conditions (distance, niveaux gris). Ce remplissage conditionnel permet de compléter le corps des personnes qui est le plus souvent orienté verticalement [KH99]. Les trous de petite taille sont alors supprimés à l'aide d'une fermeture morphologique (fig. 3.2.d). Enfin un étiquetage en composantes connexes permet de supprimer des petites régions parasites.

(a) masque région (b) masque contour (c) combinaison des masques (d) résultat obtenu après post-traitement

FIGURE 3.2 – Combinaison du masque région et du masque contour

3.2.2 Gestion de l'image de référence

La conception de l'image de référence peut se résumer à l'acquisition de la scène sans objet. Cependant la stabilité de l'illumination ne peut être garantie au fil de la séquence, spécialement pour les scènes extérieures. Cette image doit donc être remise à jour régulièrement. Il est également possible que cette image de référence ne soit pas disponible à l'initialisation, il faut alors pouvoir la construire.

Principe de notre approche

De la même manière que [HHD00, Bru97], notre approche est basée sur l'utilisation de deux modes : le premier construit la référence, le second la met à jour. Ces deux modes sont présentés sur le diagramme d'états de la figure 3.3.

A l'initialisation de l'algorithme, nous décidons s'il faut construire ou seulement mettre à jour l'image de référence. En ce qui concerne la construction, elle s'effectue à partir de la première image de la séquence sur un nombre fixé d'images. Ensuite l'application bascule automatiquement vers le mode « mise à jour ». L'application reste dans ce mode tant que l'image de référence ne subit pas de forte dégradation. Si tel est le cas, le superviseur peut forcer l'application à retourner en mode « construction » afin de réinitialiser la référence.

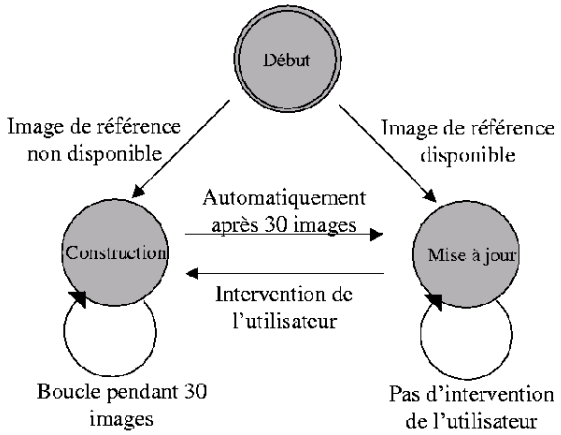


FIGURE 3.3 – Les deux modes de gestion de l'image de référence

Construction de l'image de référence

Pour construire une image de référence, l'application apprend le contenu de l'image au niveau des zones où la valeur des pixels ne varie pas au cours du temps [HW87]. La valeur de ces pixels enrichit l'image de référence grâce à une somme pondérée entre l'image courante et l'image de référence existante, pour chaque pixel (équation 3.1) :

$$I_{ref}(p, t+1) = \alpha_p . I(p, t) + (1 - \alpha_p) . I_{ref}(p, t) \tag{3.1}$$

- $I_{ref}(p, t)$ et $I(p, t)$ sont les valeurs d'intensité du pixel p à l'instant t dans l'image de référence et dans l'image courante.
- α_p est le coefficient d'adaptation. Si p appartient au fond $\alpha_p \in]0, 1]$, sinon $\alpha_p = 0$.

La valeur d'α_p fixe la vitesse d'apprentissage. Pour apprendre progressivement l'image de référence, nous avons choisi une valeur faible : $\alpha_p = 0.1$. Ainsi seuls les éléments stables et durables sont assimilés dans la référence.

Pour réaliser le calcul de l'équation 3.1, nous devons connaître les pixels qui appartiennent au fond. Nous utilisons pour cela une carte de stabilité qui est obtenue en calculant la différence entre trois images successives de la séquence ($I(t-1)$, $I(t)$ et $I(t+1)$). Cette carte de stabilité est une image binaire qui distingue les pixels *mobiles* et les pixels $fixes$ dans ces trois images. Tous les pixels déclarés $fixes$ appartiennent au fond, et l'image de référence est mise à jour pour ces pixels uniquement.

Ce mode d'apprentissage a un inconvénient. En effet, si une personne s'immobilise dans le champ de la caméra, elle est progressivement introduite dans la référence. Nous perdons alors le masque de cette personne. C'est pourquoi nous avons choisi de limiter dans le temps cette phase de construction. Le nombre d'images utilisées pour la construction dépend de la vitesse des entités présentes dans la séquence. Dans le cadre de notre application, nous avons utilisé 30 images successives.

Mise à jour de l'image de référence

Lorsque la phase de construction est terminée, nous considérons que l'image de référence est fiable. Le mode de mise à jour est alors utilisé pour conserver cette qualité. Au cours de cette mise à jour, les changements locaux, qui sont dus aux faibles modifications dans le fond, et les changements globaux, qui affectent l'image dans son ensemble, sont traités différemment.

Changements locaux La technique de mise à jour présentée dans cette partie permet de prendre en compte les faibles variations locales dans le contenu de l'image. La contribution de l'image courante au cours de cette mise à jour est également régulée par l'équation 3.1. La différence avec le mode précédent repose sur le fait que nous n'utilisons pas de carte de stabilité pour déterminer les pixels appartenant au fond. En effet, la qualité de l'image de référence permet d'obtenir un masque correct des entités présentes dans la séquence. L'ensemble de ces entités correspond au premier plan, nous connaissons donc par défaut les pixels du fond.

Changements globaux d'illumination lents ou rapides Les changements globaux d'illumination arrivent fréquemment dans les séquences intérieures et extérieures. Ces changements affectent dans tous les cas le contenu de l'image de référence. Ces changements sont donc détectés afin d'apporter une correction à la référence.

Un changement rapide d'illumination peut être détecté entre deux images successives. Pour cela une différence globale est calculée pour les images t et $t+1$:

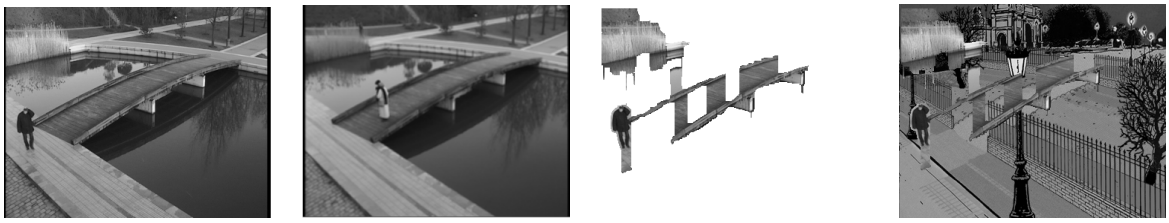
$$\Delta_1 = \frac{\sum_p I(p, t-1) - \sum_p I(p, t)}{N} \tag{3.2}$$

N correspond au nombre de pixels dans l'image. Si $|\Delta_1| > C_1$ ($C1$ étant un seuil) un changement rapide d'illumination est détecté. La différence moyenne Δ_1 est alors ajoutée à chaque pixel de l'image de référence.

Un changement progressif (lent) d'illumination peut être absorbé par l'équation 3.1 tant qu'il n'est pas trop important par rapport au coefficient d'adaptation α_p. Si la valeur d'α_p ne permet pas de compenser assez rapidement cette variation dans la référence, la valeur moyenne d'illumination dans l'image de référence devient de plus en plus différente de celle de l'image courante. Ce type de variation lente ne peut pas être détectée entre deux images successives, mais elle peut devenir détectable entre l'image courante et la référence. Ainsi le même principe de détection et de correction que précédemment est utilisé entre l'image courante et la référence, avec un seuil $C2$:

$$\Delta_2 = \frac{\sum_p I(p,t) - \sum_p I_{ref}(p,t)}{N} \tag{3.3}$$

Si $|\Delta_2| > C_2$ un changement d'illumination lent est détecté. La différence moyenne Δ_2 est alors ajoutée à chaque pixel de la référence.



(a) image originale (b) image de référence (c) masque extrait (d) incrustation (© Casterman-Tardi)

FIGURE 3.4 – Image 1 : Initialisation. L'image de référence doit être reconstruite pour améliorer la qualité des masques.

(a) image originale (b) image de référence (c) masque extrait (d) incrustation (© Casterman-Tardi)

FIGURE 3.5 – Image 25 : Construction de l'image de référence. Plusieurs images sont nécessaires pour apprendre l'image de référence.

3.2.3 Résultats

Cette application a été développée en langage C, elle est utilisable sous environnement Unix, Linux et Windows. Elle peut traiter une séquence vidéo à la fréquence de 8 images (352×288) par seconde sur un Pentium III $800MHz$. Les résultats présentés ont été obtenus à partir d'une séquence du projet Art-live (filmée en extérieur).

(a) image originale (b) image de référence (c) masque extrait (d) incrustation (© Casterman-Tardi)

FIGURE 3.6 – Image 1000 : La variation d'illumination au cours de la journée est prise en compte dans la référence.

La planche de résultats présente à trois instants donnés quatre images : l'image originale de la séquence, l'image de référence, le masque extrait et un exemple d'incrustation.

La figure 3.4 représente l'initialisation du traitement. Pour tester la robustesse de la technique de construction de l'image de référence, nous avons choisi aléatoirement une image appartenant à la séquence pour jouer le rôle de la référence initiale (fig 3.4.b). Cette image contient une personne et son illumination est différente de celle de la première image traitée (fig 3.4.a). C'est pourquoi le masque obtenu est de mauvaise qualité (fig 3.4.c).

La figure 3.5 montre que plusieurs images sont nécessaires pour apprendre la totalité de la référence. Ce but est atteint après environ une vingtaine d'images (fig 3.5.b). La qualité de la référence permet alors d'avoir un bon masque (fig 3.5.c).

La figure 3.6 montre que les variations d'illumination sont prises en compte au cours de la mise à jour de la référence. (Dans l'image fig 3.6.a, le temps s'assombrit)

3.2.4 Conclusion

Nous avons présenté une application qui permet d'extraire en temps réel les personnes présentes dans une séquence vidéo. Cette application utilise une double extraction de masque pour être moins sensible à la présence d'ombres. La technique de remise à jour de l'image de référence qui lui est associée lui permet d'être utilisée sur de longues séquences vidéo. A l'époque de ce travail, les mélanges de gaussiennes pour modéliser statistiquement le comportement de chaque pixel venaient de voir le jour [SG99]. A l'heure actuelle, ces mélanges constituent sans doute la meilleure base pour ce type de problème, d'autant plus qu'on peut maintenant obtenir des cadences de traitement proches du temps réel.

Ce projet s'est poursuivi au laboratoire avec d'autres chercheurs dans le contexte du réseau d'excellence SIMILAR qui s'intéressait aux interfaces homme-machine multimodales [BCNT06].

3.3 Segmentation exhaustive par pyramide évolutive

Cette section présente une méthode de segmentation spatio-temporelle hiérarchique qui prend en compte tout le contenu de l'image et pas seulement une région ou un objet d'intérêt. Cette méthode est fondée sur la pyramide irrégulière présentée à la section 2.2.

La segmentation spatio-temporelle dans des images à caméra mobile peut être vue comme une généralisation du problème avec caméra fixe. On distinguera avec soin les problématiques de suivi (*tracking*) et celles de segmentation spatio-temporelle. Effectivement, au cours des dernières années, ces deux

axes se sont développés en ayant des préoccupations propres. Le but du suivi est de suivre des zones ou des points ayant des caractéristiques particulières, sans notion d'entité, de forme globale ou d'objet. La segmentation spatio-temporelle quant à elle a pour but de réaliser une segmentation spatiale au cours du temps. La notion d'objet (ou de région correspondant à un sous-objet) est capitale, tout comme la nécessité de localiser au mieux le contour de ces objets. En corolaire, l'initialisation des contours des objets est nécessaire et critique, qu'elle soit réalisée manuellement, automatiquement ou de façon semi-automatique.

La pyramide évolutive a été notre première approche pour proposer une technique de segmentation dans les séquences vidéo en niveaux de gris [BR98]. La méthode pourrait également être étendue aux images en couleurs. Convaincus que la pyramide irrégulière était un puissant outil pour la segmentation spatiale, nous avons voulu l'utiliser en spatio-temporel. Plusieurs constats nous ont amené a développer cette méthode particulière :

- une structure pyramidale est coûteuse à construire, tant d'un point de vue ressources mémoire que processeur.
- la complexité de construction d'un niveau décroît exponentiellement lorsque le niveau augmente.
- construire une pyramide par image et essayer de mettre en correspondance leur structure nous paraissait hasardeux.
- dans un même plan vidéo, la redondance temporelle est assez importante.

Une réponse adaptée aux remarques précédentes consiste à ne construire la pyramide qu'une seule fois (sur la toute première image) puis à modifier à la fois sa structure et son contenu pour les adapter au contenu des images successives. La construction sur la première image $I(0)$ est celle qui est utilisée pour la segmentation d'une image statique (voir section 2.2). Elle fournit ainsi un ensemble de partitions $P_i(0)_{i=0...apex}$ et une structure de graphe arborescente qui peut être parcourue grâce à des liens inter-niveaux de type père et fils. L'adaptation à l'image suivante (ou évolution) met en oeuvre un traitement particulier en trois phase : division, intégration et fusion. (figure 3.7).

Cette méthode est un rapprochement entre la structure hiérarchique de la pyramide irrégulière et les approches classiques de division/fusion à base de partitionnements géométriques (carrés, triangles, polygones, ...). Le but de ces approches est à la fois d'avoir une décomposition hiérarchique efficace du contenu, et de réaliser des traitements associés à la résolution la mieux adaptée.

3.3.1 Division

L'analogie peut être faite avec une division de type *quadtree* fondée sur un prédicat d'homogénéité : dans une représentation *quadtree*, tout carré non homogène est divisé en 4 carrés fils. La division effectuée dans la pyramide évolutive suit le même procédé tout en ayant ses propres particularités :

1. Chaque région a une forme qui ne dépend pas d'un critère géométrique.
2. La division récursive n'est pas rigide mais est réalisée en découpant une région r de niveau l en ses propres régions filles de niveau $l-1$.
3. le critère d'homogénéité est évalué sur l'image $I(t)$ avec des partitions obtenues avec $I(t-1)$.

De manière générale, la partition $P(t)$ sert à découper l'image $I(t+1)$, définissant ainsi un ensemble de régions $R(t+1)$. Toute région de $R(t+1)$ n'étant pas homogène selon le prédicat utilisé est récursivement découpée avec $P(t)$.

La division repose sur les critères suivants :

- Variance de la région. Une région dont la variance en niveaux de gris est trop élevée doit être divisée.
- Taille de la région. Une région dont la taille est inférieure à un seuil n'est pas divisée.

Une région homogène n'est pas subdivisée. Le sous-arbre correspondant ainsi que le champ récepteur associé ne sont pas modifiés. Une région qui n'est pas homogène mais qui ne peut pas être divisée, car sa taille est inférieure à une taille seuil, est une région obsolète.

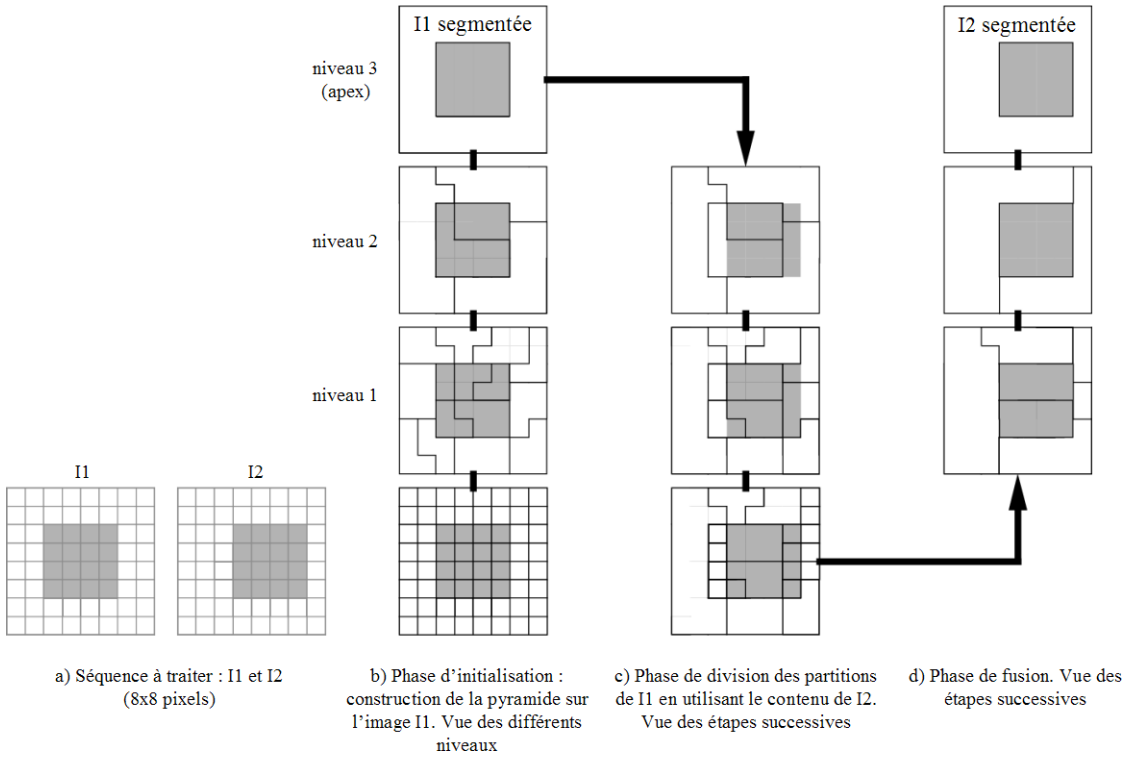


FIGURE 3.7 – Principe de la pyramide évolutive sur un exemple de séquence de 2 images où un carré gris se déplace vers la droite. La flèche noire indique la chronologie des étapes

Lorsqu'une région est non homogène, son sommet est supprimé dans la pyramide de graphes car il n'a plus lieu d'être. Récursivement, on voit donc que lorsque la phase de division est achevée, il ne reste dans la pyramide que des sommets homogènes ou obsolètes, la partie supérieure de la pyramide ayant pu être supprimée plus ou moins localement, en fonction des modifications du contenu des images $I(t)$ et $I(t+1)$.

3.3.2 Intégration

C'est la phase pendant laquelle les sommets de la pyramide de l'image $I(t)$ sont mis à jour avec les données de l'image $I(t+1)$. Chaque région obsolète (quel que soit son niveau dans la pyramide) est mise à jour avec les attributs (niveau de gris moyen et variance de la région) calculés sur son champ récepteur de l'image $I(t+1)$. Dans la pratique, pour réduire la complexité de l'algorithme, la mise à jour d'une région obsolète est réalisée dès lors que la région est connue comme étant obsolète.

3.3.3 Fusion

La phase de division s'est traduite par une destruction partielle de la partie supérieure de la pyramide. La phase de fusion a pour but de "recoller" les morceaux des régions qui ont été arbitrairement découpées (une fois encore, comme dans une fusion classique de *quadtree*). Le traitement n'est réalisé qu'à partir du niveau le plus bas qui contient une ou plusieurs régions obsolètes. Lors de la fusion, seules les régions obsolètes et leurs voisines sont traitées. Les régions obsolètes peuvent fusionner uniquement avec leur voisine ou avec l'ascendant d'une de leur voisine. Hormis ces contraintes, la fusion se déroule de la même façon qu'avec une pyramide construite sur une image statique.

La fusion assure à la fois i) la re-fusion des régions déconnectées lors de la division et ii) la reconstruction de la partie supérieure de la pyramide en tenant compte de contenu de $I(t+1)$. Il faut noter que lors de la fusion, la hauteur de la pyramide peut varier d'une image à l'autre.

3.3.4 Suivi inter-images

Le but premier de la technique présentée est de segmenter chaque image en utilisant les redondances temporelles. En second lieu, on veut également être capable de suivre un maximum de régions entre deux images successives, c'est-à-dire réaliser un *tracking* précis de tout le contenu de l'image. Nous proposons ici une extension simple à la pyramide évolutive pour réaliser ce suivi.

Dans cette partie, nous parlons d'objets pour signifier les régions de l'apex et ainsi illustrer au mieux le suivi, mais il est entendu que cette appellation n'a aucune connotation sémantique.

A la fin de la segmentation de la première image, chacun des n objets de l'apex est numéroté avec une étiquette unique non nulle. Durant la phase de division, chaque sommet fils hérite de l'étiquette de son père. Dans la phase d'intégration, chaque région obsolète est initialisée sans étiquette (0), signifiant ainsi qu'elle n'est reliée à aucun objet connu de l'apex. Durant la fusion, l'étiquette d'un père est déterminée en fonction des étiquettes de ses fils, selon les règles suivantes :

1. Si tous les fils ont l'étiquette i, le père est étiqueté i. La région père est faite des morceaux du même objet. Il s'agit soit d'une partie de l'objet, soit de l'objet lui-même.

2. Si tous les fils ont soit l'étiquette i, soit l'étiquette 0, le père est étiqueté i. La région père est faite de morceaux connus d'un objet, et de morceaux inconnus ayant des caractéristiques similaires. C'est le cas par exemple lorsque l'objet (ou la caméra) a subi une translation.

3. Si des fils ont des étiquettes différentes i et $j, (i, j \neq 0)$, le père est étiqueté 0. Ceci permet éventuellement, grâce à la règle numéro 2 de fusionner ultérieurement cette région à un objet.

4. La propagation d'étiquette de fils en père est itérée jusqu'à l'apex de la pyramide. Parmi les sommets de l'apex, certains ont des étiquettes non nulles (il s'agit des objets connus de l'image précédente qui ont été suivis avec succès). Les objets avec une étiquette nulle se voient attribuer une étiquette inutilisée non nulle.

3.3.5 Résultats

Cette méthode a été testée sur des séquences 128×128 et 256×256. Les temps de traitement obtenus et la possibilité de suivre toute région d'une segmentation aussi complexe qu'elle soit montrent l'intérêt de la méthode. En effet, dans nos expérimentations, la mise à jour pour faire évoluer la pyramide entre deux images est de 3 à 8 fois plus rapide que la construction elle-même. Un exemple de résultat est donné figure 3.8.

3.3.6 Conclusion

La limite principale de la méthode réside essentiellement dans le fait que la propagation temporelle des étiquettes se fait sur un critère de recouvrement entre deux images successives. L'hypothèse de bonne utilisation de la méthode est donc la suivante : toute amplitude de mouvement supérieure à la taille de l'objet ne permet pas le recouvrement des 2 instances de l'objet et ne permet pas le suivi de cet objet. Cette limite pourrait être atténuée en compensant avec une estimation du mouvement dominant.

La méthode gère la disparition ou l'apparition d'objets. Notons que les règles de diffusion des étiquettes lors des phases de division et de fusion permettent le cas échéant d'avoir des objets non connexes portant la même étiquette. Ce qui permet entre autre de pouvoir gérer des occultations partielles d'objet (un objet coupé en deux par un autre objet par exemple).

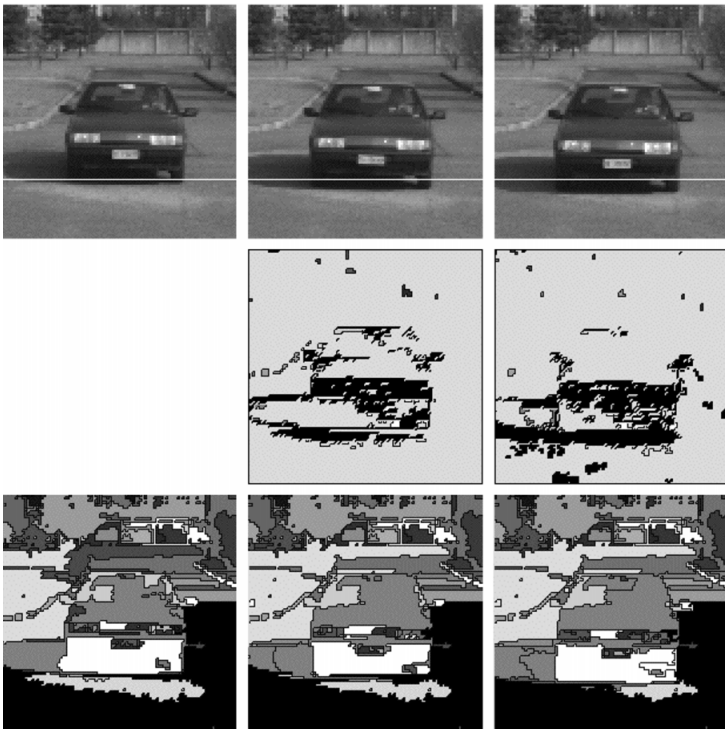


FIGURE 3.8 – Exemple de résultat avec la pyramide évolutive : la première ligne montre 3 images consécutives (la ligne blanche permet de quantifier visuellement le mouvement). La deuxième ligne montre les régions obsolètes. La troisième ligne donne les partitions en fausses couleurs

3.4 Segmentation d'objets d'intérêt par propagation d'étiquettes

Cette section présente le travail de thèse de Guillaume Foret intitulé *segmentation spatio-temporelle d'objets vidéo en vue de leur caractérisation*, que j'ai co-encadré et qui a été soutenu en 2003 [For03]. Le but de cette recherche est de segmenter avec précision, pendant la durée d'un plan, n objets dont les masques initiaux sont connus.

3.4.1 Contexte

Cette recherche [FBC02, FB03, FBC05] s'est déroulée parallèlement au projet exploratoire RNRT OSIAM : Outils de Segmentation d'Images Animées pour MPEG-4/7 (1999-2001) auquel nous avons participé et dont l'objectif était de proposer et d'intégrer sur une plate-forme unique une boîte à outils de segmenteurs pour les images animées. Les partenaires du projet étaient Philips Research France, le CNET, l'IRESTE, l'IRISA, I3S, CREATIS, et l'Université Grenoble I (TIMC / LIS).

La segmentation d'un plan séquence suivant son contenu en objets est une opération délicate et difficile à automatiser. Elle nécessite tout d'abord de définir quels sont les objets d'intérêt par rapport à l'arrière-plan. Dans cette recherche, nous considérons que cette information est déjà disponible pour la première image I(t=0) (figure 3.9.a). Plus précisément, ceci pré-suppose que nous possédons une partition P(0) (figure 3.9.b), dans laquelle une étiquette est attribuée à chaque pixel de I(0) en fonction de l'objet auquel il appartient.

De manière à simplifier la discussion, nous nous limiterons au suivi d'un seul objet. L'extension de la méthode proposée au suivi de plusieurs objets est immédiate (cf. figure F.2 en annexe page 146). De plus, aucune contrainte particulière sur la forme, la texture, ou le mouvement des objets considérés n'est imposée. Ceci offre un système générique utilisable dans une gamme variée d'applications.

(a) I(0) (b) I(1) (c) Partition P(0) associée à I(0) (d) Objet à t=0

FIGURE 3.9 – Exemple d'initialisation de suivi temporel

L'association de l'image I(0) et de la partition P(0) permet de savoir quels sont les éléments de l'image qui constituent l'objet (figure 3.9.b). L'objectif est alors de segmenter automatiquement l'objet dans les images suivantes.

Les changements temporels entre deux images successives étant de faible amplitude, une solution largement adoptée est de construire la partition P(t+1) en fonction du résultat précédent P(t). Cette opération est effectuée en utilisant à la fois des informations spatiales (couleurs) et temporelles (mouvement) mesurées sur I(t) et I(t+1). Les deux principales difficultés rencontrées lors de la segmentation de l'objet dans chaque nouvelle image (à partir de t=1) sont :

- La segmentation des zones faiblement contrastées au niveau du contour de l'objet,
- La gestion automatique des nouveaux éléments apparaissant dans l'image.

La première de ces difficultés peut être résolue en appliquant volontairement une sur-segmentation spatiale. Pour répondre à la seconde difficulté, nous faisons l'hypothèse selon laquelle les caractéristiques colorimétriques des éléments découverts sont en accord avec celles de l'objet auquel ils appartiennent.

A la suite d'une brève introduction sur les techniques de segmentation spatio-temporelle orientées vers le suivi temporel d'objets vidéo, nous introduisons notre propre approche. Les parties suivantes présentent les formalismes retenus en justifiant leur choix. La dernière partie décrit et commente les résultats de suivi d'objets obtenus.

3.4.2 Introduction de la méthode

Grand nombre de méthodes ont été proposées pour déduire la partition courante d'une image à partir d'une autre partition. Le cas le plus couramment considéré est celui de deux images successives I(t) et I(t+1), supposant P(t) connue. Deux catégories de méthodes peuvent être distinguées :

1. Les méthodes caractérisées par une **projection** en avant de la partition : P(t) est compensée en mouvement, puis appliquée directement sur l'image I(t+1). Un **ajustement** de cette partition suivant le contenu de I(t+1) permet d'obtenir P(t+1) [GL98a, PYW00, MM97]. Cet ajustement est réalisé **localement** par un algorithme de segmentation spatiale.

   L'inconvénient de ces méthodes est de privilégier, lors de l'ajustement, le contour le plus contrasté dans les zones remises en cause. Même si ce contour correspond très souvent au contour réel de l'objet, il peut s'avérer néfaste de ne pas considérer les autres possibilités de segmentation dans ces zones. Une légère délocalisation du contour dans I(t+1) peut entraîner des dégénérescences importantes dans les partitions suivantes.

2. Les méthodes, plus récentes, qui proposent d'appliquer une **segmentation** spatiale sur l'ensemble de I(t+1) : la segmentation de l'objet est effectuée en étiquetant les régions segmentées suivant deux classes (objet et arrière-plan) [AOW+98, ML98, GL98b, GPSG99, Pat00]. Cette **classification** est réalisée en fonction de la représentation précédente de l'objet.

L'inconvénient ici est le coût de calcul nécessaire au traitement de chaque image. En effet la segmentation de la totalité de l'image, ainsi que la classification de chaque région obtenue alourdissent le traitement.

Afin de pallier les inconvénients cités précédemment nous avons étudié l'association de ces deux types d'approches. Nous avons abouti à une technique originale utilisant des outils éprouvés, conçue en trois étapes séquentielles :

- Une projection en avant de P(t),
- Une segmentation spatiale appliquée **localement** dans I(t+1),
- Une classification des segments locaux obtenus.

Le schéma de la figure 3.10 présente l'enchaînement de ces trois étapes (modules) permettant de construire la partition P(t+1) à partir de P(t) et des deux images originales I(t) et I(t+1).

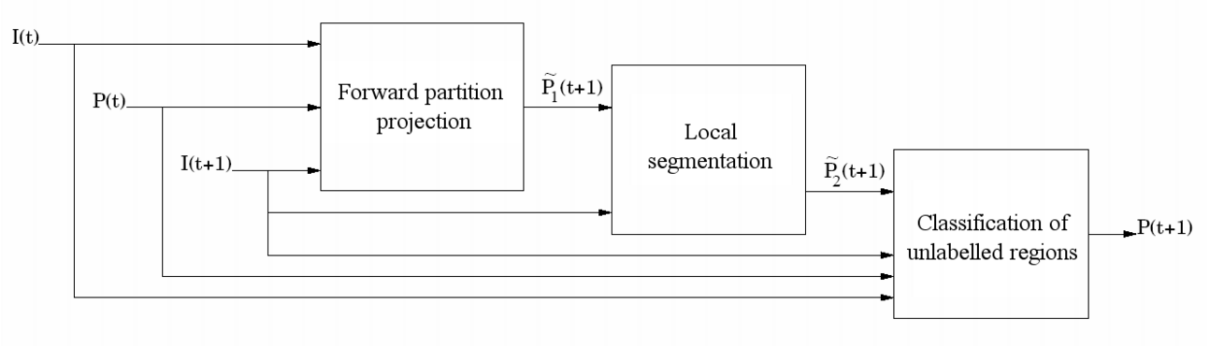


FIGURE 3.10 – Schéma blocs de la méthode proposée ($\tilde{P}_1(t+1)$ et $\tilde{P}_2(t+1)$ correspondent à des partitions intermédiaires)

Le fonctionnement de chacun de ces modules est détaillé dans les parties suivantes.

3.4.3 Projection de partition

Dans cette partie, j'explique comment, à l'aide d'une partition connue au temps t et d'un *block matching* effectué entre les images t et $t+1$, on reconstruit une partition grossière au temps $t+1$.

Etat de l'art

L'objectif est de projeter la partition P(t) sur l'image I(t+1) en fonction des similarités mesurées entre I(t) et I(t+1). Cette démarche permet de réduire le temps nécessaire au traitement de I(t+1) et implique une cohérence entre P(t) et P(t+1) favorable à la stabilité du suivi temporel d'un objet vidéo.

Sa réalisation dépend de la manière dont est modélisée la partition P(t). Lorsque cette dernière est représentée par une approximation polygonale de ses contours [WBPB95, Bon98, MBPB02], la projection est appliquée sur les sommets des polygones en fonction du mouvement estimé sur les régions polygonales associées. On déplace ainsi les contours de la partition.

Plus couramment P(t) représente la segmentation de I(t) en attribuant à chaque pixel l'étiquette d'une région ou d'un objet. Dans ce cas, la projection en avant de P(t) consiste à prédire la répartition des étiquettes dans I(t+1). Elle peut être effectuée en déplaçant les régions [Wan98] ou l'objet [GL98b, PYW00, JBBA01] suivant un vecteur mouvement estimé. Une autre solution est de compenser en mouvement P(t) en considérant non pas des régions mais des blocs de pixels [PS94, MM97]. L'estimation de mouvement est alors réalisée par mise en correspondance de blocs (*block-matching*).

Utilisant par la suite un algorithme de segmentation orienté régions (cf. section 3.4.4), nous avons choisi une représentation par étiquettes de la partition. La compensation en mouvement des régions ou des objets après avoir estimé leur déplacement entre I(t) et I(t+1) est une approche intuitive. Dans notre cas, l'objectif n'est pas de construire une partition définitive de I(t+1), mais une prédiction de cette partition. L'utilisation de l'algorithme de *block-matching* permet d'optimiser ce traitement, tout en conservant une qualité très satisfaisante [ML98].

Mise en correspondance par *block-matching*

Soient I_1 et I_2 deux images d'un même plan séquence ; le vecteur de mouvement d'un bloc de l'image I_1 est obtenu en recherchant dans I_2 le bloc le plus similaire dans une zone limitée de l'image (fenêtre de recherche). Le critère de mise en correspondance de deux blocs est généralement la somme en valeur absolue des différences en niveaux de gris ou en couleurs (*SAD : Sum of Absolute Difference*). La *SAD* de deux blocs X et Y ($X \in I_1$, $Y \in I_2$) de $N \times N$ pixels est définie par :

$$SAD(X,Y) = \sum_{i=1}^{N}\sum_{j=1}^{N} |X(i,j) - Y(i,j)| \tag{3.4}$$

Pour un bloc source X donné, le bloc Y le plus similaire est celui qui minimise la *SAD*. Il est également possible de retenir comme critère la somme des différences au carré (*SSD : Sum of Squared Difference*). Dans la littérature scientifique, de nombreux algorithmes de *block-matching* existent. Un état de l'art récent et synthétique peut être trouvé dans [CHF01]. Nous avons utilisé la méthode intitulée *Block Sum Pyramid Algorithm (BSPA)* [LC97] [LCC98], qui limite le temps de calcul grâce à un algorithme d'éliminations successives *(Successive Elimination Algorithm (SEA))*, et dont l'estimation est d'aussi bonne qualité que la méthode *FSA*. Rappelons brièvement qu'un algorithme de *block-matching* est caractérisé par deux paramètres :

- La taille des blocs manipulés (*taille_bloc*). Ce paramètre dépend de la résolution de l'image et par conséquent du format de la vidéo. Pour une séquence CIF 352×288 pixels (respectivement QCIF 176×144 pixels), nous utilisons des blocs de 16×16 pixels (respectivement 8×8 pixels).
- Le vecteur de déplacement maximum autorisé en pixels (V_{maxBM}). Ce paramètre fixe la taille de la fenêtre de recherche. En général, V_{maxBM} = (8,8) ou (16,16).

Lors de l'estimation de mouvement par *block-matching*, c'est le niveau de gris des pixels qui est utilisé dans le calcul de la *SAD*.

Fiabilité des mises en correspondance

La mise en correspondance de deux blocs est réalisée en minimisant la valeur de la *SAD* des niveaux de gris de leurs pixels. La valeur *SAD* minimum obtenue peut être utilisée pour évaluer la fiabilité d'une mise en correspondance. Elle peut, par conséquent, servir à éviter une erreur d'estimation de mouvement.

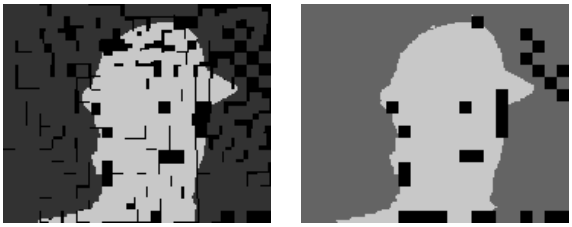
Lorsque cette valeur est faible, le vecteur de déplacement déduit est donc fiable (en particulier dans le cas de blocs hétérogènes). En revanche, lorsque la valeur minimum de la *SAD* est supérieure à un seuil de fiabilité (T_{SAD}), il est préférable de ne pas tenir compte de cette mise en correspondance. Le bloc en question ne pourra pas être utilisé pour l'étape de prédiction.

Application à la prédiction de partition

Afin de limiter les erreurs de prédiction, le seuil de fiabilité doit être assez strict (ex : $T_{SAD} = 5$ pour des blocs de 8×8 pixels). Comme le montre le paragraphe suivant, ce seuil peut varier en fonction du contenu et du niveau de bruit dans la scène traitée.

L'estimation de mouvement entre I(t) et I(t+1) peut être effectuée dans les deux sens temporels $t \rightarrow t+1$ (en mode avant) ou $t+1 \rightarrow t$ (en mode inverse). En mode avant, la division en blocs réguliers

est appliquée sur l'image I(t). Un vecteur de mouvement est estimé pour chaque bloc. Le même découpage par blocs est appliqué sur la partition P(t). Chaque bloc source de P(t) est alors projeté suivant son vecteur mouvement, afin de prédire la partition de I(t+1) (figure 3.11.a). Le résultat obtenu contient des zones non recouvertes, dues soit à la superposition des blocs lors de leur projection, soit au seuil de fiabilité. Aucune prédiction n'est retenue lorsqu'un pixel est associé à plusieurs objets.



(a) projection en mode avant

(b) projection en mode inverse

FIGURE 3.11 – Illustration des résultats obtenus lors de la projection par *block-matching* de la partition P(t=0) (fig. 3.9.c) sur l'image I(1) (fig. 3.9.b)

A cette prédiction de partition, il est préférable d'utiliser celle obtenue avec l'algorithme de *block-matching* en mode inverse (division en blocs réguliers appliquée sur I(t+1)), car les zones non recouvertes sont ainsi limitées à l'ensemble des blocs sans correspondant. Une prédiction de la partition de I(t+1) est alors obtenue en remplaçant chaque bloc source par le bloc correspondant dans P(t), s'il existe (figure 3.11.b).

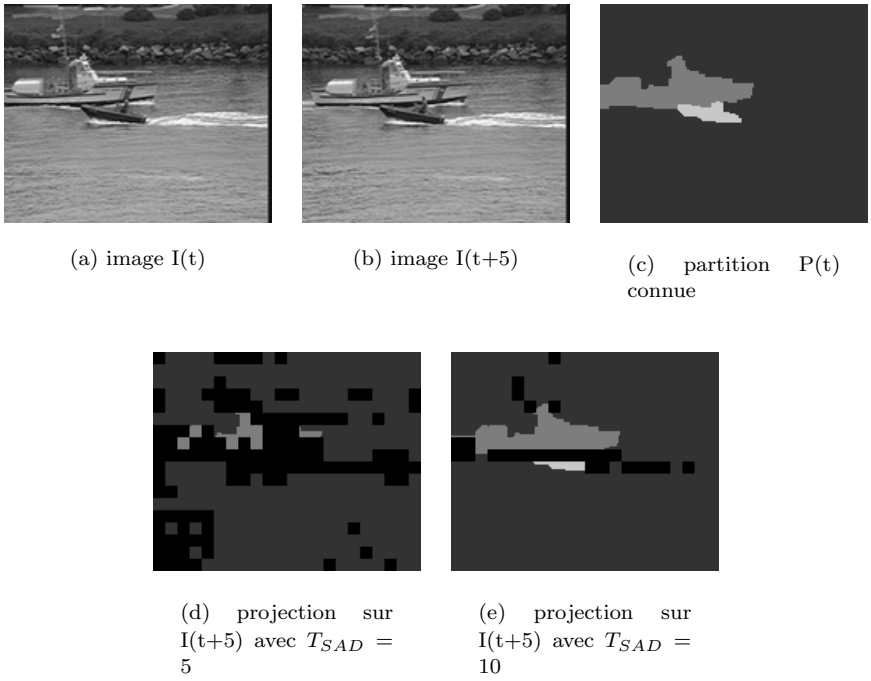
Discussion sur la projection

L'objectif principal de cette étape de projection est de simplifier le traitement des étapes suivantes sans introduire d'erreur. C'est pourquoi par défaut $T_{SAD} = 5$ qui est une valeur relativement stricte. Pour certaines séquences, ce seuil peut être augmenté. Par exemple, dans la séquence Coastguard, la présence d'eau dans la scène rend difficile la mise en correspondance entre blocs. Cette difficulté est accentuée en considérant deux images éloignées de la séquence. La figure 3.12 illustre qu'un seuil plus élevé permet une prédiction plus riche dans ce cas. Cette prédiction facilitera la segmentation dans la nouvelle image. En contrepartie les risques d'erreurs de prédiction sont plus élevés.

Dans la figure 3.11.b, les blocs représentés en noir sont les blocs qui n'ont pu être prédits ; leurs pixels ne possèdent pas d'étiquette. De manière générale, ces zones non prédites peuvent résulter de l'apparition d'un nouvel objet, du découvrement de l'arrière-plan par les objets suivis, ou bien encore de changements importants du contenu de l'image (déformation de l'objet). Ces zones nécessitent une re-segmentation dans la nouvelle image.

En outre la prédiction des étiquettes dans I(t+1) fournit une approximation du contour de l'objet suivi. Les pixels proches de ce contour doivent être re-segmentés pour assurer la qualité de segmentation. C'est pourquoi les étiquettes prédites au niveau du contour sur une largeur de 8 à 10 pixels sont supprimées (figure 3.13.a).

Nous présentons dans la partie suivante la manière dont sont traitées les zones sans prédiction, l'objectif étant de segmenter avec précision l'objet d'intérêt. Notons que les zones sans prédiction seront appelées par la suite zones d'incertitude [PS94].



(a) image I(t)

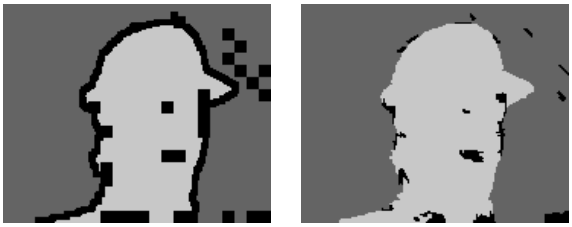
(b) image I(t+5)

(c) partition P(t) connue

(d) projection sur I(t+5) avec $T_{SAD} = 5$

(e) projection sur I(t+5) avec $T_{SAD} = 10$

FIGURE 3.12 – Projection, à partir des images d'origine I(t) et I(t+5), de la partition P(t) sur l'image I(t+5) pour deux valeurs de T_{SAD}



(a) Prédiction finale des étiquettes dans I(1)

(b) Résultat de la propagation d'étiquettes dans (a)

FIGURE 3.13 – Exemple de zones d'incertitude (en noir) avant et après la segmentation

3.4.4 Segmentation spatiale

Cette partie détaille la deuxième étape de notre méthode qui consiste à segmenter localement $I(t+1)$ de manière à traiter les zones d'incertitude résultant de la projection.

Segmentation locale pour le suivi temporel d'objets

Pour ce faire, une segmentation locale par pyramide irrégulière est réalisée sur les zones d'incertitude, en considérant aussi les pixels voisins déjà étiquetés sur une largeur de 2 à 3 pixels. La propagation des étiquettes permet ainsi de segmenter les zones d'incertitude en utilisant la cohérence (continuité) spatiale des objets (figure 3.13.b). Les régions non similaires à l'arrière-plan et à l'objet restent sans étiquette. Par la suite, la classification de ces régions (troisième étape) permettra la localisation complète du contour de l'objet.

Importance de la sur-segmentation

Une sur-segmentation ($T_{seg} = 7$) est préférable afin d'éviter tout risque de fusions abusives dans les zones faiblement contrastées.

L'inconvénient éventuel d'une sur-segmentation est d'obtenir en grand nombre des régions de petite taille. Ceci augmente le temps de traitement nécessaire à la classification. Cependant la segmentation n'étant pas appliquée sur la totalité de l'image, nous pouvons nous permettre de conserver la plupart de ces régions. Seules les plus petites sont éliminées au cours de la segmentation locale par pyramide irrégulière en fonction du paramètre $T_{minRegion}$.

Le choix de ce paramètre est important. La taille minimale des régions manipulées conditionne la précision de la localisation des contours. Expérimentalement, des valeurs $T_{minRegion}$ supérieures à 10 pixels entraînent localement des imprécisions, dues à une propagation forcée d'étiquettes. Ces imprécisions sont souvent accentuées au cours du suivi temporel et rendent inutilisables les résultats.

Certaines régions restent cependant sans étiquette. Leur classification à l'intérieur de l'objet ou de l'arrière-plan est présentée dans la partie suivante.

3.4.5 Classification par rétro-projection

Cette partie présente la dernière étape qui consiste à attribuer une étiquette à chaque région non encore étiquetée, en la projetant sur la partition précédente connue.

Classer une région par rapport à une partition connue est une manière de favoriser la cohérence temporelle dans la description de l'objet suivi.

Les approches présentées dans [ML98, AOW+98] effectuent la classification de régions segmentées dans I(t+1) en fonction de la projection en avant de P(t). Dans notre approche, le résultat de cette projection est utilisé pour guider la segmentation. Il ne peut donc pas servir à la classification des régions sans étiquette.

Une classification par projection en arrière (rétro-projection) des régions sans étiquette est alors réalisée : le mouvement de chaque région est estimé et chacune est projetée sur la partition précédente afin de déterminer si elle appartient ou non à l'objet suivi [GL98a, GPSG99, Pat00].

Estimation du mouvement des régions

La projection des régions sans étiquette requiert l'estimation de leur vecteur de mouvement de I(t+1) vers I(t). A l'instar de [GL98a], un modèle de mouvement translationnel permet d'estimer le mouvement d'une région. Chaque région sans étiquette, dans I(t+1), est alors mise en correspondance (*region-matching*) dans I(t).

Contrairement au *block-matching* utilisé lors de la première étape, l'information couleur (y, u, v) de chaque pixel p est prise en compte pour renforcer l'exactitude de la classification. Soit R une région de I(t+1), son vecteur de mouvement V est obtenu en minimisant la Somme en valeur Absolue, pour tous les pixels $p \in R$, des Différences en couleur (SAD_{color}) :

$$SAD_{color}(R) = \sum_{p \in R} [|(y(t+1,p) - y(t,p+V)| + \gamma.(|u(t+1,p) - u(t,p+V)| + |v(t+1,p) - v(t,p+V)|)] \quad (3.5)$$

γ est un coefficient normalisateur utilisé pour compenser la différence d'échelle remarquable entre la luminance et les deux chrominances. L'estimation du vecteur V est effectuée en considérant un vecteur de déplacement maximal en pixels $V_{maxRegion}$ qui fixe les dimensions de la fenêtre de recherche (ex : $V_{maxRegion} = (16, 16)$).

Soulignons ici que nous nous intéressons principalement à la mise en correspondance, et non pas à la qualité de l'estimation de mouvement obtenue.

Classification des régions sans étiquette

Lors de leur projection en arrière sur P(t) (figure 3.14.a) chaque région se voit attribuer l'étiquette qu'elle recouvre majoritairement. A l'issue de ce traitement, une partition totale de I(t+1) est obtenue (figure 3.14.b).

Un traitement morphologique simple (une fermeture, puis une ouverture appliquées sur le masque de l'objet) est appliqué à la partition finale de manière à lisser les contours et améliorer la qualité visuelle (figure 3.14.c).

(a) projection sur P(0) des régions sans étiquette

(b) classification des régions sans étiquette dans P(1)

(c) partition finale P(1)

FIGURE 3.14 – Classification des régions sans étiquette par rétro-projection

Bilan sur la classification par rétro-projection

Une région sans étiquette correspond dans I(t+1) à une entité ayant ses propres caractéristiques en couleurs. Dans la plupart des cas, cette entité est présente dans I(t). Le fait qu'elle n'ait pas d'étiquette est dû soit à sa petite taille soit à une légère déformation. Au cours de la rétro-projection, ces entités sont mises en correspondance avec leur propre représentation dans l'image précédente et peuvent ainsi récupérer la bonne étiquette. Ceci facilite la stabilité temporelle dans la localisation du contour de l'objet.

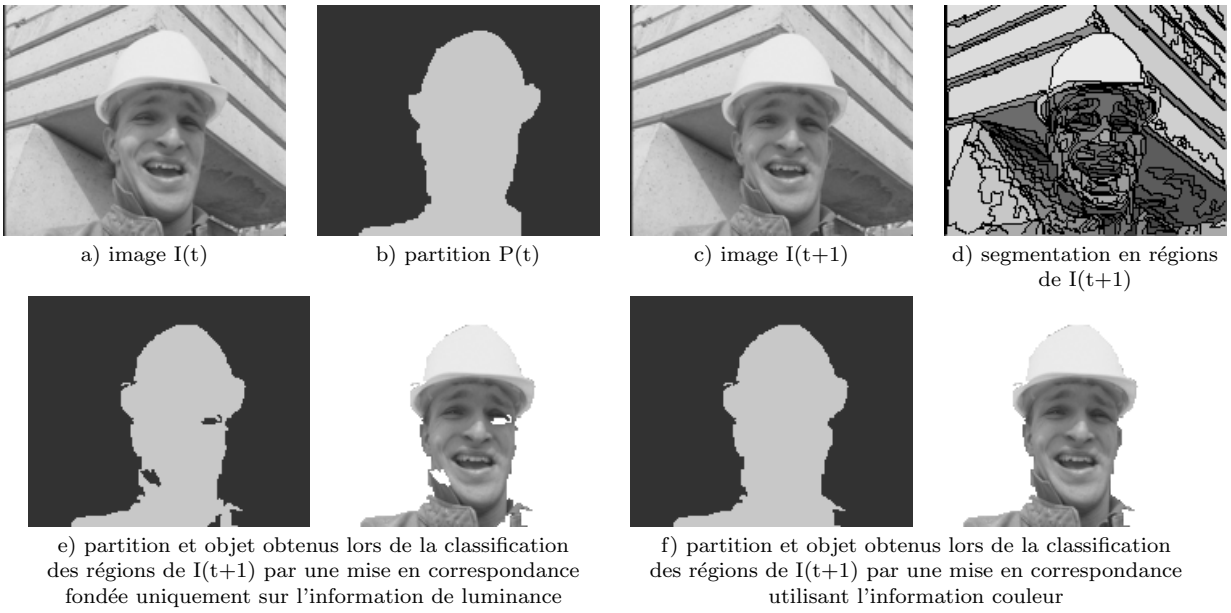
Lorsqu'une entité n'est pas présente dans I(t), elle est projetée par défaut sur une partie qui lui est similaire en couleur. Nous faisons alors l'hypothèse que cette similarité en couleur est suffisante pour lui attribuer une étiquette. Il est à noter que face aux problèmes dus aux découvrements, la classification par rétro-projection est plus pertinente que la propagation d'étiquettes.

L'utilisation des trois composantes couleur lors de la mise en correspondance des régions renforce l'exactitude de la classification. Faute de cartes de vérité terrain pour les vidéos étudiées, une quantification de l'apport de l'information couleur n'est actuellement pas possible. Néanmoins, la qualité visuelle de la segmentation est bien améliorée comme le montre le test présenté à la figure 3.15 . La classification des régions segmentées dans I(t+1) (fig. 3.15.d) échoue à plusieurs reprises lorsque la rétro-projection n'utilise que l'information de luminance (fig. 3.15.e), contrairement au cas où l'information couleur est utilisée (fig. 3.15.f).

3.4.6 Résultats

Nous présentons dans cette dernière partie des résultats de suivi temporel obtenus avec des séquences test du standard MPEG au format QCIF (176 × 144) : Foreman, Coastguard, Mother&Daughter et Carphone. Pour chacune de ces séquences, la partition initiale en objets P(0) (ex. : fig. F.1.a page 145 et fig. F.2.a page 146) a été obtenue à l'aide d'une interface graphique interactive.

Après avoir mis en évidence la qualité du suivi sur des séquences originales, nous nous intéresserons au comportement de la méthode sur des séquences modifiées obtenues par sous-échantillonnage temporel. Nous illustrerons enfin par un exemple une limite d'utilisation de la méthode.



a) image I(t) b) partition P(t) c) image I(t+1) d) segmentation en régions de I(t+1)

e) partition et objet obtenus lors de la classification des régions de I(t+1) par une mise en correspondance fondée uniquement sur l'information de luminance

f) partition et objet obtenus lors de la classification des régions de I(t+1) par une mise en correspondance utilisant l'information couleur

FIGURE 3.15 – Résultat d'un test illustrant l'apport de l'information couleur par rapport à l'information de luminance seule lors de la mise en correspondance de régions. La classification des régions segmentées dans I(t+1) (d) échoue à plusieurs reprises lorsque la rétro-projection n'utilise que l'information de luminance (e), contrairement au cas où l'information couleur est utilisée (f)

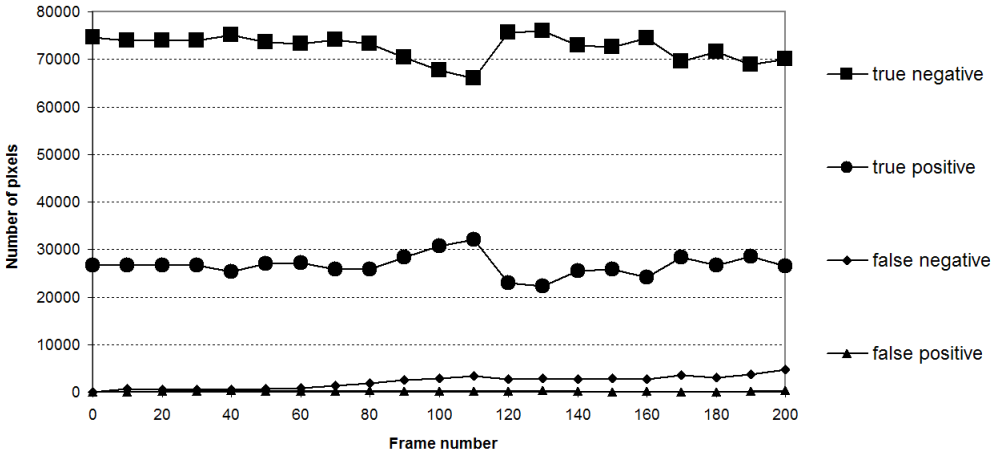


FIGURE 3.16 – Evolution de la qualité de l'étiquetage des pixels : vrais positifs, vrais négatifs, faux positifs, faux négatifs lors de la segmentation de la séquence foreman (figure 3.18)

Suivi dans les séquences originales

Les résultats portent ici sur trois suivis, effectués sur une cinquantaine d'images successives, avec le même jeu de paramètres :

1. Projection de partition par *block-matching* :
   - taille des blocs utilisés : $taille\_bloc = 8$ pixels
   - vecteur de déplacement maximum autorisé en pixels : $V_{maxBM} = (8,8)$
   - seuil de fiabilité de mise en correspondance : $T_{SAD} = 5$
2. Segmentation locale par pyramide irrégulière :
   - seuil global de similarité : $T_{seg} = 7$
   - taille minimale autorisée d'une région : $T_{minRegion} = 5$ pixels

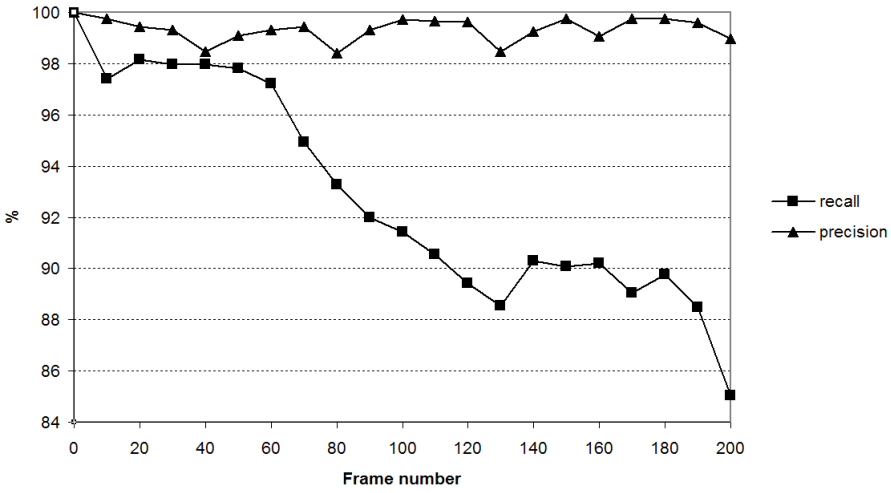


FIGURE 3.17 – Mesures de rappel ($\frac{vrais\,positifs}{vrais\,positifs+faux\,negatifs}$) et précision ($\frac{vrais\,positifs}{vrais\,positifs+faux\,positifs}$) extraites de la figure 3.16

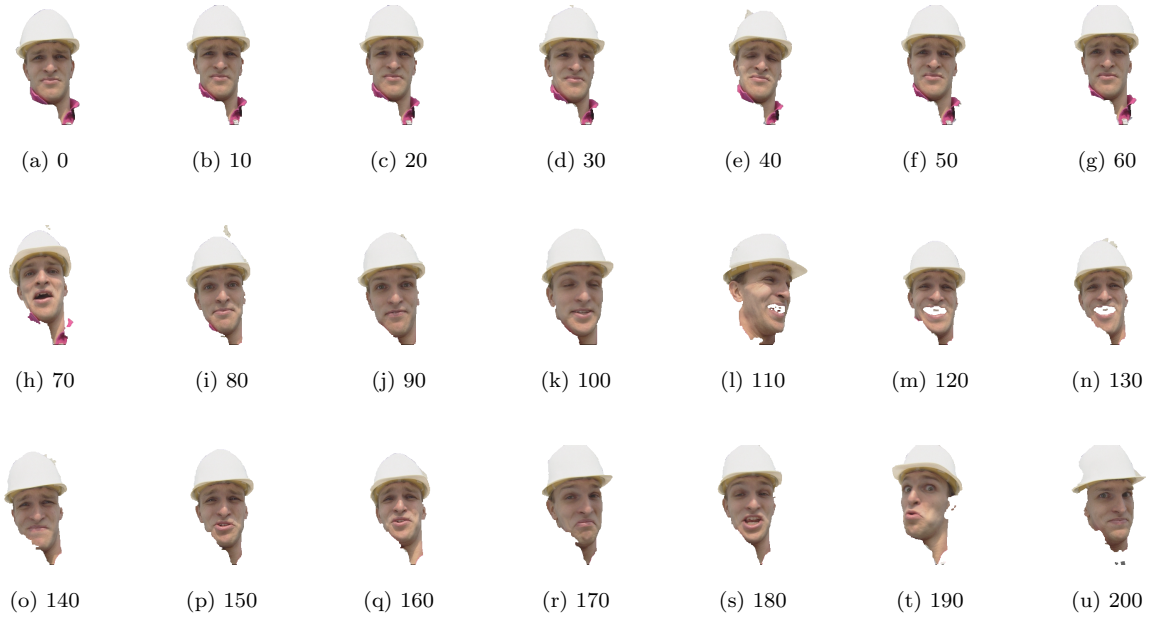


FIGURE 3.18 – Segmentation spatio-temporelle des 200 premières images de la séquence CIF foreman

3. Classification par projection en arrière :
   - vecteur de déplacement maximum autorisé en pixels pour une région : $V_{maxRegion} = (16,16)$

La figure 3.18 montre un suivi de très bonne qualité visuelle pour un objet non rigide. Les déformations de l'objet, ainsi que le mouvement de la caméra, entraînent l'apparition de nouveaux éléments au cours du traitement (exemple : la partie gauche du visage). Ces éléments découverts sont ici attribués correctement à l'objet ou à l'arrière-plan. Leurs caractéristiques spatiales (couleur) ont permis leur mise en correspondance avec les parties déjà visibles de l'objet auquel ils appartiennent.

De manière générale la gestion automatique des éléments découverts est délicate et source d'erreur. C'est pourquoi l'interruption du suivi temporel doit être proposée à l'utilisateur, afin de lui permettre de réinitialiser la partition P à un instant particulier. Le manque d'information sémantique sur les zones découvertes conduit inévitablement à ce constat. On peut noter toutefois que le suivi proposé peut être

réalisé sans interruption sur deux cents images, avec des variations d'aspect non négligeables de l'objet suivi.

La figure F.1 en annexe page 145 fournit le résultat du suivi d'un objet composé de nombreuses régions homogènes de petite taille. Ce résultat illustre la robustesse de la méthode à suivre des contours faiblement contrastés entre l'objet et l'arrière-plan (cf. les contours entre les extrémités du bateau et l'eau).

L'approche présentée peut également s'étendre au suivi de plusieurs objets (figure F.2 page 146). Il suffit de fournir une partition initiale P(0) avec autant d'étiquettes que d'objets. Dans ce cas, la représentation par graphe de la pyramide irrégulière est un atout pour modéliser l'interaction entre les objets qui évoluent dans le plan séquence observé.

Sur un PC Pentium III à 1 GHz, des cadences de traitement de 1 à 3 images par seconde sont obtenues, en fonction de la complexité des objets suivis et de leur nombre.

Suivi dans des séquences sous-échantillonnées temporellement

Afin de valoriser la robustesse de l'algorithme par rapport aux mouvements de forte amplitude et aux déformations brutales, deux nouvelles séquences test ont été construites. La première, nommée Foreman-Bis, correspond à un sous-échantillonnage temporel avec un pas de 5 de la séquence Foreman précédente. La seconde, CarphoneBis, est un sous échantillonnage avec un pas de 10 de la séquence Carphone.

Les figures F.3 page 147 et F.4 page 148 fournissent les résultats de suivi obtenus sur ces deux séquences (les paramètres sont inchangés, excepté $V_{maxBM} = (16,16)$). Nous pouvons y observer l'efficacité de l'étape de projection de partition (deuxième colonne des figures F.3 page et 147 et F.4 page 148), qui permet de localiser le contour de l'objet dans chaque nouvelle image et de détecter les zones temporellement instables.

Il faut noter que dans ces séquences sous-chantillonnées, le déplacement de l'objet découvre de manière plus importante l'arrière-plan. De plus la composition de l'objet varie plus fortement. La gestion des éléments découverts est alors la principale difficulté. Pour ces deux séquences, la méthode proposée parvient à traiter correctement ces zones d'incertitude.

Robustesse aux variations d'initialisation

La technique est également capable de corriger une initialisation imprécise des contours. Les figures 3.19 et 3.20 montrent l'évolution de l'objet lorsqu'il est initialisé avec un masque "vérité terrain" érodé de 3 pixels et dilaté de 6 pixels. La figure 3.21 montre la différence entre les deux résultats obtenus. Quand il n'y a pas de compétition locale entre des contours potentiels, même lorsque le contraste est faible, on peut voir clairement que le résultat ne dépend pas des conditions initiales. Néanmoins, des différences peuvent subsister lorsque des gradients élevés sont proches spatialement (oreille, col par exemple). La vitesse de convergence n'est pas constante et dépend à la fois des contrastes locaux et de leur évolution temporelle.

(a) 0 (b) 1 (c) 5 (d) 10 (e) 15 (f) 20 (g) 25

FIGURE 3.19 – Suivi dans les 25 premières images, initialisé avec un masque de vérité terrain érodé de 3 pixels (a)

(a) 0 (b) 1 (c) 5 (d) 10 (e) 15 (f) 20 (g) 25

FIGURE 3.20 – Suivi dans les 25 premières images, initialisé avec un masque de vérité terrain dilaté de 6 pixels (a)

(a) 0 (b) 1 (c) 5 (d) 10 (e) 15 (f) 20 (g) 25

FIGURE 3.21 – Les différences entre les masques des figures 3.19 et 3.20 diminuent au fil des images

3.4.7 Conclusion

L'originalité de notre approche réside dans la combinaison de trois étapes usuelles en segmentation spatio-temporelle : projection/segmentation spatiale/classification. Cette association est encore peu étudiée. Elle cumule pourtant les avantages de chaque opération pour un suivi rigoureux d'objets vidéo, et entraîne une réduction du temps de traitement de chaque image.

Au cours de notre étude, l'algorithme du *block-matching* s'est avéré être un outil simple et efficace pour réaliser la projection de partition entre deux images.

La pyramide irrégulière, quant à elle, répond bien aux besoins d'une segmentation locale. Elle ajuste les contours prédits des objets, et segmente les zones temporellement instables à l'aide d'une propagation d'étiquettes. La conservation de régions sans étiquette à la fin de la segmentation permet de distinguer certaines entités au voisinage des contours des objets.

Le principe de la classification de régions par projection en arrière a été mis en valeur récemment dans de nombreux travaux. Les résultats que nous avons obtenus permettent de confirmer la pertinence d'une telle classification pour finaliser la segmentation des objets suivis.

L'état actuel de la méthode permet d'envisager des applications qui nécessitent des masques d'objets précis. Certaines améliorations restent toutefois à apporter telle que la détection des dégénérescences de la localisation du contour des objets. De plus une étude approfondie des possibilités pour renforcer la robustesse de la classification des nouvelles entités extraites (petites et grandes), nous paraît d'un intérêt majeur. A titre d'exemple, une meilleure prise en compte de l'information mouvement serait un plus pour cette étape.

L'une des applications visées concerne la représentation de l'information contenue dans une vidéo. A l'heure actuelle, l'indexation du contenu des images et la construction de résumés par détection de rupture de plans et assemblage d'images clé permettent de répondre partiellement à ce défi technologique. Mais le contenu d'une image clé est souvent lui même trop riche, et les objets d'intérêt sont souvent "minoritaires" dans une image (par rapport à l'arrière-plan par exemple). Une suite logique à ce travail consiste à extraire des objets clé caractéristique du contenu d'une vidéo (voir chapitre 4).

3.5 Environnement interactif pour l'hypervidéo

Dans cette section, je présente deux applications que j'ai conçues et développées pour réaliser un démonstrateur des travaux de l'INRIA dans le domaine de l'analyse et la structuration de vidéos.

Après ma thèse, j'ai été embauché comme ingénieur-expert à l'INRIA Rhônes-Alpes dans le projet MOVI dirigé par Roger Mohr, pour développer des applications afin de valoriser les travaux effectués dans le domaine de l'hypervidéo au sein de l'INRIA Rhônes-Alpes et l'équipe Vista de l'IRISA/INRIA à Rennes. Alcatel Alsthom Research à Marcousis était également partenaire du projet ainsi que L'Institut National de l'Audiovisuel. En 1999, le projet est à la une de *Inedit*, la lettre d'information de l'INRIA (figure D.1 page 141).

L'hypervidéo est la notion d'hypertexte associée à des séquences vidéo. Dans une hypervidéo, on peut par exemple cliquer sur des objets qui évoluent dans l'image et obtenir des informations complémentaires ou être amené dans une autre vidéo relative à l'objet cliqué. L'hypervidéo est un concept qui n'est pas encore arrivé à maturité et qui est encore largement du domaine de la recherche. Dans la majorité des cas, force est de constater qu'il n'est pas envisageable dans un futur proche d'avoir des traitements totalement automatisés gérant l'extraction, le suivi et l'indexation d'objets dans les vidéos. Il est donc important du côté conception de développer des interfaces et outils interactifs d'édition (*authoring tools*) permettant d'utiliser et de corriger manuellement les résultats obtenus avec les techniques actuelles.

D'un point de vue utilisateur, il est tout aussi capital de tirer partie au mieux de ces nouveaux supports d'information avec des interfaces adaptées pour la visualisation, la recherche d'information précise ou synthétique et la navigation dans les vidéos.

Dans ce projet, j'ai conçu et développé une architecture logicielle et deux applications graphiques. Les difficultés techniques principales étaient les suivantes :

1. l'inter-opérabilité : exécuter de façon transparente plusieurs traitements existants comportant chacun leur propre paramétrage, entrées/sorties et formats de données.
2. l'interactivité avec l'utilisateur : mettre à disposition des outils de retouche graphique, un *player* vidéo performant, comme il en existe beaucoup maintenant (mais ce projet a commencé en 1997 et date de 14 ans).

Par la suite, sous ma direction, ces applications ont été ré-écrites en Java pour être indépendantes des bibliothèques graphiques Ilog Views que j'avais précédemment utilisées à la demande de l'INRIA.

La première application (Videoprep) est destinée au concepteur de l'hyper-vidéo. Elle utilise des traitements semi-automatiques pour donner une structure à une vidéo. Cette structuration est obtenue en 3 étapes : découpage en plans, extraction de zones d'intérêt et indexation. Chaque étape peut être suivie par une visualisation et édition des résultats en utilisant un lecteur de vidéo et des outils graphiques [BMS+98, BBB+98b, BBB+98a, HM00, Ham02].

3.5.1 Découpage en plans

Après une détection automatique des plans (*cuts* ou transitions), l'interface permet de visualiser chacun d'eux à l'aide d'une imagette ainsi que le découpage de la vidéo obtenu (figure D.2.a page 142). L'usage de la souris permet très simplement de supprimer une rupture de plan ou une transition erronée ou de corriger avec précision (à l'image près) une transition erronée.

3.5.2 Détection d'objets

Une entité qui apparaît pendant un nombre consécutif de n images est appelée objet. Cet objet est constitué de n instances. Après une extraction automatique d'objets, par analyse de mouvement, leurs contours sont vectorisés et affichés en surimpression sur la vidéo afin de vérifier la qualité du résultat. L'outil d'édition permet alors de modifier la forme d'un objet, de créer ou supprimer un objet. Du texte peut être associé à l'objet : un nom et une description (figure D.2.b page 142). Une action particulière peut être attachée à un objet et sera déclenchée lorsque l'objet sera sélectionné par l'utilisateur : son, exécution d'une autre application, connexion à une page web, ... Tout objet statique (i.e. appartenant au mouvement dominant) peut être détouré manuellement ; le mouvement dominant étant connu, toutes les instances de cet objet peuvent ensuite être générées automatiquement tout au long du plan et tant que l'objet reste dans le champ, par compensation de mouvement.

3.5.3 Construction des classes d'objets

Le traitement automatique d'indexation permet ensuite de regrouper les objets en classes. Le but est de mettre dans une même classe des objets qui apparaissent dans des plans différents mais qui partagent des caractéristiques communes (points caractéristiques, couleurs). L'interface affiche une classse par ligne, avec une imagette par objet. Les classes peuvent être modifiées par simple glisser/déposer des imagettes. Comme pour les objets, on peut associer une action quelconque à une classe (figure D.2.c page 142).

3.5.4 L'interface de l'utilisateur final

La seconde application, VideoClic (figure D.2.d page 142) est destinée à l'utilisateur final et a pour but de montrer ce qu'il est possible de faire avec une vidéo qui a été préalablement structurée :

- Des imagettes (une par plan) résument la structure et le contenu de la vidéo et donnent par un simple clic un accès direct à une partie de la vidéo.
- L'ensemble des classes (par exemple les personnages, les voitures, les produits commerciaux) peut être affiché et donne à l'utilisateur une idée du contenu sémantique de la vidéo. L'accès direct dans la vidéo à tout objet d'une classe se fait par simple clic sur son imagette.
- L'utilisateur peut visionner une vidéo à l'aide d'un *player* doté d'une interface de type magnétoscope. Un clic sur un objet pendant que la vidéo se déroule ou est en mode pause affiche la description de l'objet et exécute l'action éventuelle prédéfinie.
- Des boutons particuliers permettent de naviguer dans la vidéo, par exemple jouer tous les plans contenant l'objet sélectionné.

3.5.5 Conclusion

Ce travail mené à l'INRIA a été précurseur dans son domaine. Il a ouvert la voie à plusieurs applications industrielles et a montré quels étaient les éléments essentiels pour obtenir un outil fonctionnel et évolutif :

- la mise en commun des résultats de recherche de différents bords,
- le découpage en deux applications : un *authoring tool* et un *player*,
- l'indépendance entre le flux vidéo et l'information ajoutée,
- l'indépendance entre les traitements et l'interface utilisateur,
- des interfaces graphiques évoluées,
- des outils interactifs pour l'édition des résultats.

Grâce aux retombées de ce travail, on a également compris qu'à cette époque (il y a 14 ans), le marché n'était pas encore prêt pour des vidéos enrichies et donc pour les applications associées. Mais en quelques années, l'ère de la vidéo numérique et de la distribution à la demande est arrivée et il semble certain que

de telles fonctionnalités sont sur le point de rencontrer un très large public, et par là même, faire la part belle aux environnements de vidéos interactives.

Chapitre 4

Extraction d'objets clé dans les vidéos

Sommaire

4.1 Introduction : les prémices des objets vidéos

A l'heure actuelle, il est possible de filmer ou de visualiser des vidéos dans n'importe quelle circonstance et à n'importe quel endroit car les techniques associées sont communément intégrées dans les systèmes portables qui inondent le marché. Ceci est rendu possible grâce à la combinaison de facteurs socio-culturels et technologiques : l'explosion de la production internationale de contenu vidéo numérique, la démocratisation des systèmes numériques d'acquisitions (webcams, camescopes, appareils photos, téléphones), la progression des techniques de compression, des débits des réseaux de télécommunication et enfin la miniaturisation des moyens de stockage.

Les vidéos (films, clips, bandes-annonces, publicités, informations, fêtes de familles, souvenirs de vacances, visiophone 3G), font partie intégrante de notre quotidien. Le consommateur et l'utilisateur se retrouvent face à une masse importante de données difficile à gérer et à manipuler. Il est commun d'entendre "trop d'informations tue l'information". Cet adage éculé nous montre néanmoins qu'il est nécessaire et urgent de trouver des techniques efficaces pour structurer, synthétiser, indexer, archiver, cataloguer, représenter, interroger et parcourir ces vidéos toujours plus nombreuses. Les chercheurs ont tout d'abord représenté le contenu des vidéos avec des images caractéristiques. Désormais, je crois qu'il est indispensable de représenter le contenu d'une vidéo grâce à des objets représentatifs. Par la suite, l'étude du comportement de ces objets pourrait permettre à la fois leur manipulation et une représentation de haut niveau du contenu.

Dans ce chapitre, je présente les travaux de thèse de Jérémy Huart. Leur but consiste à extraire de façon automatique les objets en mouvement dans un plan. L'approche est originale car elle n'est pas fondée sur le suivi des objets mais sur un traitement en différé (i.e. lorsque tout le plan est connu). La présentation de ce travail est précédée par un rapide état de l'art sur l'extraction d'information clé dans les vidéos.

4.2 Extraction d'information clé

Dans [HZ99] et [LZT01], A. Hanjalic *et al.* et Y. Li *et al.* proposent un état de l'art des méthodes d'extraction d'images clés. On distingue un certain nombre d'approches d'extraction d'images clés, fondées sur l'échantillonnage temporel, le découpage en plans, le découpage en segments, la détection de visage, ... La difficulté de ce type d'algorithme est l'évaluation de la pertinence des images choisies.

4.2.1 Les régions clés

Dans [CT04], J. Calic *et al.* proposent une étude temporelle du comportement des ***régions clés*** obtenues par segmentation spatiale à basse résolution. Cette segmentation est obtenue par *clustering* des coefficients de la DCT directement issus du signal de la vidéo compressée. Les images clés sont sélectionnées suivant certaines règles fondées sur les disparitions, les apparitions et les interactions entre régions.

4.2.2 Les objets clés

Certains ont adopté la notion ***d'objet clé*** à la place de celle d'image clé. La méthode décrite dans [OLV03] utilise un suivi de fond par comparaison de signatures, détaillé dans [OHL00]. Grâce à ce suivi, des paires d'images (successives) dont le fond varie très peu sont sélectionnées afin de procéder à une différence entre les 2 images de chaque paire pour obtenir une carte de contour pertinente. Cette dernière permet d'extraire les objets en utilisant une méthode fondée sur une estimation par blocs.

Dans [SF05], Guoliang Fan *et al.* combinent une extraction d'images-clés et une segmentation orientée objet. (i) La segmentation en plans facilite et améliore la segmentation orientée objet en utilisant l'extraction d'images clés pour sélectionner uniquement quelques images pertinentes utilisées pour l'apprentissage d'un modèle statistique d'objets. (ii) La segmentation objet est utilisée pour améliorer la segmentation en plans en utilisant une méthode de raffinement des images clés orientées modèle.

Dans [KH00], Kim et Hwang utilisent la segmentation en objets fondée sur une carte de contour en mouvement (*Mouving Edge*) pour sélectionner les images clés. La première image de chaque plan est automatiquement classée comme image clé. Ensuite, la segmentation en objets appliquée sur chaque image du plan va permettre de comparer continuellement le nombre d'objets contenus dans l'image courante avec le nombre d'objets contenus de la dernière image clé extraite. A ce stade, l'image courante est déclarée comme image clé si :

1. Le nombre d'objets varie.
2. Les régions constituant les objets sont déclarées trop distantes (à nombre d'objets constant).

Cette méthode fonctionne correctement dans le cas où la vidéo ne contient que très peu d'objets telle qu'une vidéo de surveillance mais dans le cas de vidéos plus complexes, son efficacité diminue.

4.2.3 Le mouvement

Les approches orientées mouvement sont utilisées pour contrôler le nombre d'images clés par rapport à la dynamique temporelle dans la scène. Les méthodes les plus utilisées sont les méthodes de différence d'images [LHC+96] ou de flux optique [Wol96].

Dans [Wol96], W. Wolf extrait, pour chaque image, une mesure simple du mouvement grâce au flux optique. Par analyse temporelle de cette mesure, les images correspondant aux minima de mouvement locaux sont sélectionnées pour devenir images clés.

4.2.4 Les mosaïques

Une limitation des techniques vues précédemment est qu'il n'est pas toujours possible d'extraire les images représentant le contenu entier de la vidéo [LZT01]. Par exemple lors d'une séquence panoramique (*panning/tilting*) même si plusieurs images clés sont sélectionnées, les dynamiques sous-jacentes ne seront pas correctement capturées. Dans ce cas, une approche fondée sur la création de mosaïque peut être utilisée pour générer une image panoramique qui représente, de manière indirecte, le contenu entier de la vidéo.

Le procédé s'effectue généralement en deux étapes [VL98] : (1) calcul du modèle du mouvement global entre deux images successives et (2) composition des images en une seule image panoramique par modification des images selon les paramètres estimés de la caméra.

Une fois que la mosaïque est construite, il est possible d'extraire les objets du premier plan. Dans [HS01], les auteurs proposent une extraction progressive des régions du premier plan, en prenant en compte des informations contour. La région clé est ensuite caractérisée par sa forme, sa texture, et sa trajectoire

Bien que les mosaïques apportent plus d'informations que les images clés, elles ont leurs propres limitations. En effet, elles peuvent être calculées uniquement dans les cas particuliers de mouvements panoramiques de la caméra. Cependant les vidéos réelles comportent des mouvements très complexes et changent fréquemment de fond et de premier plan. Une solution à ce problème a été proposée par Taniguchi *et al.* [TAT97] qui consiste en l'utilisation soit d'images clés soit d'une mosaïque dans le cas où un mouvement panoramique est détecté.

4.2.5 Notre approche

Du point de vue de l'utilisateur-spectateur, nous appelons *objet d'intérêt* une entité dans un plan qui offre un intérêt sémantique particulier (personnage, visage, véhicule, objet manufacturé, ...). Cette notion est en partie subjective. Dans la figure 4.1.a, l'objet d'intérêt le plus flagrant est sans doute le bateau qui est suivi pendant plusieurs secondes par la caméra.

Par la suite, on appelle S-VOP (Sub Video Object Plan) chaque instance de composante connexe en mouvement extraite au temps t. Le suffixe "S" de "S-VOP" indique que la plupart du temps, seule une sous-partie de l'objet d'intérêt est détectée.

A partir d'une vidéo brute préalablement découpée en plans, nous proposons d'extraire de façon générique et automatique n occurrences (S-VOP) pour chaque objet d'intérêt ayant un mouvement apparent non nul. L'occurrence la plus représentative est appelée objet-clé. Les $n-1$ autres sont les vues-clé et doivent être représentatives de façon complémentaire à l'objet clé (typiquement l'objet d'intérêt vu sous un autre aspect). Ces différentes vues pourraient permettre par exemple de constituer un modèle 2D multi-vues de l'objet d'intérêt pour prendre en compte sa variabilité et faciliter la recherche par le contenu comme le proposent les auteurs de [ZTB05].

Réaliser ce traitement de façon automatique est complexe. Pour cela, nous proposons la chaîne générique suivante :

1. Extraction des S-VOP en mouvement
2. Rejet des S-VOP pas assez compacts ou dont le masque est de mauvaise qualité
3. Classification couleur des S-VOP, une classe par S-VOP (S-VOP générateur)
4. Suppression dans chaque classe des S-VOP non cohérents avec la trajectoire du S-VOP générateur
5. Fusion des classes pour obtenir une classe par objet d'intérêt

6. Rejet des classes temporellement peu fiables
7. Sélection d'un objet clé et des vues clé associées, pour chaque classe

Chaque étape du traitement peut être vue comme une boîte noire pourvue d'un nombre restreint d'entrées/sorties. De cette façon, il est envisageable que l'une de ces boîtes soit remplacée si besoin par une autre plus efficace ou dédiée à un type d'application particulière.

4.3 Extraction d'objets en mouvement par pyramide locale

Les objets en mouvement sont ceux qui nous intéressent par la suite. Dans cette section, j'explique comment ils sont extraits.

La première phase du traitement est une extraction automatique de toute entité ayant un mouvement apparent dans le champ de la caméra [HFB04a, HFB04b]. A cette fin, la pyramide locale étudiée en section 2.7 est utilisée. Le positionnement du ruban est réalisé à l'aide d'une analyse de mouvement entre deux images qui contiennent éventuellement des objets d'intérêt à extraire. Les objets en mouvement sont supposés avoir un mouvement différent de celui induit par la caméra (mouvement global).

4.3.1 Estimation du mouvement local

L'estimation du mouvement local entre deux images successives est effectuée à l'aide d'un algorithme rapide de *block-matching* : Le *Block Sum Pyramid Algorithm* (BSPA) [LC97]. Ce traitement permet une estimation locale du mouvement entre deux images consécutives $I1$ et $I2$: un vecteur mouvement est classiquement assigné à chaque bloc carré de taille $M \times M$ de l'image. Pour nos expériences, des blocs de 8×8 pixels sont utilisés.

4.3.2 Estimation du mouvement global

Dans cette partie, nous calculons un modèle paramétrique du mouvement global. Ce modèle est obtenu en deux phases [LYKK00] : d'abord sur l'image entière puis plus précisément sur l'image entière sans les objets en mouvement. Un modèle de mouvement rigide à 4 paramètres est calculé entre $I1$ et $I2$ à l'aide de la transformation de Helmert qui inclut une translation (en x et y), une rotation et un facteur de zoom comme suit :

$$\begin{pmatrix} x_i'' \\ y_i'' \end{pmatrix} = \begin{pmatrix} a_1 & -a_2 \\ a_2 & a_1 \end{pmatrix} \cdot \begin{pmatrix} x_i \\ y_i \end{pmatrix} + \begin{pmatrix} a_3 \\ a_4 \end{pmatrix} \tag{4.1}$$

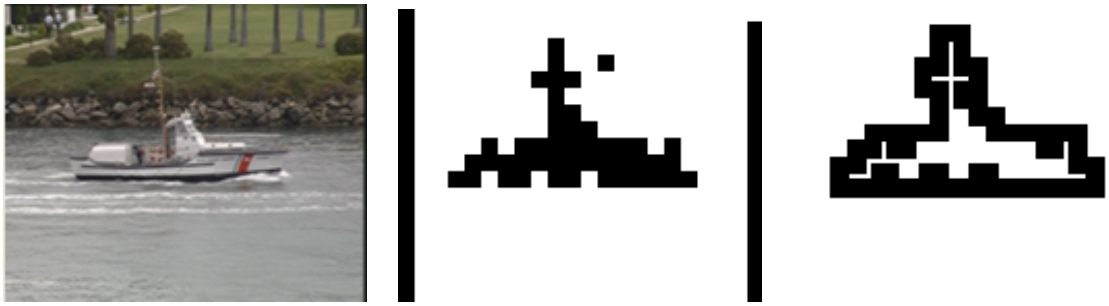
Les couples $(x_i'',\ y_i'')$ et $(x_i,\ y_i)$ représentent respectivement la position centrale du bloc i dans l'image prédite et dans l'image courante. a_1, a_2, a_3 et a_4 sont les valeurs des paramètres à déterminer. Ces derniers doivent minimiser sur l'ensemble des N blocs, l'erreur quadratique Φ entre les positions $(x_i',\ y_i')$ estimées par le *block-matching* et les positions $(a_1x_i - a_2y_i + a_3, a_2x_i + a_1y_i + a_4)$ prédites par le modèle lui-même. Cette fonction de coût est définie par :

$$\Phi = \sum_{i=1}^{N} [(a_1x_i - a_2y_i + a_3 - x_i')^2 + (a_2x_i + a_1y_i + a_4 - y_i')^2] \tag{4.2}$$

La minimisation du critère (4.2) est obtenue par les moindres carrés en utilisant la décomposition en valeurs singulières (SVD). Le code est disponible dans [uP92].

La distance Euclidienne seuillée entre les deux prédictions $(x_i',\ y_i')$ et $(x_i'',\ y_i'')$ nous permet de distinguer les blocs qui ne sont pas animés du mouvement global. L'estimation du mouvement global peut être réitérée (et donc raffinée) en ne prenant pas en compte ces derniers blocs. La distance Euclidienne

est alors calculée une nouvelle fois pour obtenir un masque binaire temporel (figure 4.1.b) qui localise les blocs en mouvement des régions d'intérêt.



(a) Image originale (b) Masque binaire temporel (c) Ruban pour la segmentation locale

FIGURE 4.1 – Le masque obtenu par compensation de mouvement global induit le ruban sensé contenir le contour de l'objet

4.3.3 Extraction automatique des régions d'intérêt

Le ruban est le résultat de la soustraction de la dilatation et de l'érosion des blocs du premier plan (figure 4.1.c). Les blocs du premier plan situés à la périphérie de l'image sont rejetés afin d'éviter les problèmes d'occultation et de désoccultation dus au mouvement de la caméra (figure 4.1.b).

Un seuillage sur une taille minimum ou un filtrage morphologique peuvent être utilisés pour éviter que trop de régions d'intérêt apparaissent dans le cas de vidéos bruitées. Néanmoins, la représentation par graphe permet autant de régions d'intérêt que possible, et ceci quel que soit le nombre d'objets partageant le même arrière plan. Une segmentation par pyramide irrégulière locale (cf. section 2.7) est effectuée dans le ruban. Elle fournit un nombre plus ou moins important de S-VOPs.

4.4 Rejet des S-VOPs non pertinents

Parmi tous les objets extraits dans la première étape, certains ne sont pas pertinents (de mauvaise qualité en quelques sorte) et ne doivent pas être pris en compte par la suite. Cette section explique comment est réalisé ce filtrage.

A partir de l'ensemble des S-VOPs extraits dans un plan, qui peuvent être de qualité et de pertinence diverses, on veut déterminer un représentant de chaque objet d'intérêt. Ce représentant est appelé *objet clé*. Afin d'extraire les objets clés, il est nécessaire de regrouper les S-VOPs en classes, chacune correspondant idéalement à un objet d'intérêt. La classification a un double objectif :

1. La détermination du nombre d'objets d'intérêt qui n'est pas une information connue *a priori*.
2. Le regroupement dans chaque classe, des S-VOPs pertinents et représentatifs d'un objet d'intérêt.

Selon le second point, une classe doit regrouper les S-VOPs les plus représentatifs de l'objet d'intérêt. Ainsi, parmi les S-VOPs extraits, seul un sous-ensemble sera sélectionné et conservé pour la suite du traitement. Pour rendre compte de la validité d'un S-VOP, deux critères sont utilisés : le premier concerne la géométrie des S-VOPs et permet de rejeter des S-VOPs parasites issus d'une mauvaise extraction. Le deuxième exprime la qualité du masque binaire du S-VOP en évaluant la compatibilité des contours du masque binaire avec les gradients de l'image.

4.4.1 Compacité

Un des principaux défauts gênant provenant de toute estimation de mouvement est ce qui peut être appelé de manière imagée les *fuites* de l'objet vers le fond (cf fig. 4.2.b). Une caractéristique souvent discriminante de ce type de régions est une faible compacité qui traduit des régions fines et très allongées. Nous utilisons cette particularité en faisant l'hypothèse que les objets d'intérêts sont assez compacts, afin de discriminer les S-VOPs parasites des S-VOPs pertinents.

La compacité (ou *facteur de forme*) $C1$ d'un S-VOP s est donnée par :

$$C1(s) = \frac{\text{Périmètre}(s)^2}{4\pi \times Aire(s)} \tag{4.3}$$

$D_{C1} = [1, +\infty[$. Une expérimentation menée sur quelques dizaines de plans de vidéos diverses nous a montré que pour chaque plan, on observe un mode prononcé pour un facteur de forme proche de 1. Ce mode correspond à des régions compactes. Afin de ne pas être trop discriminant, un seuil empirique peu restrictif $S_1 = 2,5$ est appliqué afin de filtrer les régions dont le facteur de forme est trop élevé. Ainsi tout S-VOP dépassant cette valeur est considéré comme une région parasite et n'est pas pris en compte dans la suite du traitement.

(a) S-VOP extrait à l'image 55

(b) S-VOP extrait à l'image 56

FIGURE 4.2 – La compacité : un critère discriminant

4.4.2 Qualité du masque

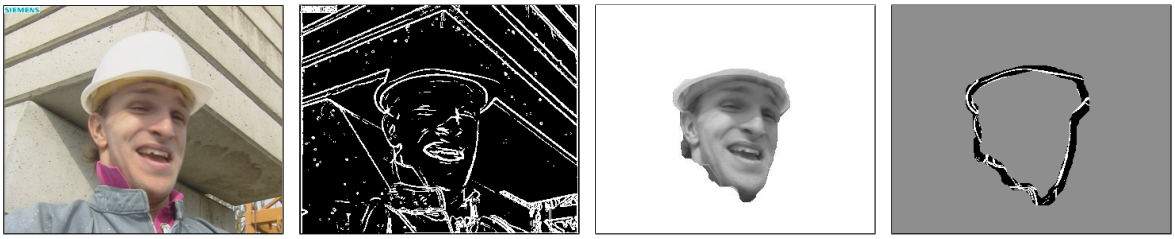
Ici, on mesure la qualité d'un masque de segmentation sous la forme d'une correspondance notée $C2(s)$ entre la périphérie z du S-VOP s et les contours c dans l'image originale. z est obtenue par une dilatation morphologique du contour du masque binaire du S-VOP (cf fig. 4.3) :

$$z(s) = Dilat_\epsilon(s) \setminus Erod_\epsilon(s) \,^* \tag{4.4}$$

$$C2(s) = \frac{Card(c \in z)}{Aire(z)} \tag{4.5}$$

Les points de contour sont extraits grâce à un seuillage adaptatif des normes du gradient de l'image du S-VOP, par un filtre de Sobel. Afin obtenir un seuil adaptatif, l'image de la norme des gradients est modélisée par des pixels de contours auxquels s'ajoute du bruit blanc Gaussien. En observant l'histogramme de cette image, il est clair que le bruit Gaussien est représenté par le premier mode. Ainsi afin d'estimer la distribution du bruit, une hypothèse est faite selon laquelle seulement un faible pourcentage p de l'image est constitué de points de contour (ici $p = 30\%$). La première partie de l'histogramme $(100 - p)$ est alors modélisée par une Gaussienne $\mathcal{N}(\sigma, \mu)$ et représente les faibles gradients dus au bruit. Les points dont le gradient dépasse 3σ sont considérés comme des points de contour.

∗. $A \setminus B$ désigne l'ensemble de tous les éléments de A qui n'appartiennent pas à B. Ici, c'est la "différence" entre le dilaté et l'érodé de s par l'élément structurant ϵ



(a) Image originale

(b) Image des gradients seuillée

(c) S-VOP

(d) Correspondance entre z, la périphérie du S-VOP (en noir) et les contours dans l'image (en blanc)

FIGURE 4.3 – Évaluation de la qualité des masques

L'algorithme supprime pour la suite du processus les S-VOPs dont la valeur de C_2 est inférieure à un seuil S_2, unique pour chaque plan afin de ne privilégier aucun S-VOP. S_2 est adaptatif par rapport à l'ensemble des coefficients C_2 qui est modélisé par une Gaussienne de moyenne μ et d'écart-type σ :

$$S_2 = \mu(C_2) - \sigma(C_2) \tag{4.6}$$

S_2 étant peu restrictif, il permet de conserver les S-VOPs pertinents sans diminuer de manière trop drastique la population des S-VOPs. L'algorithme 6 fait la synthèse de l'utilisation des deux critères vus dans cette section.

Algorithme 6 : Critère de pré-selection des S-VOPs

si $C_1(s) > S_1$ *ou* $C_2(s) < S_2$ **alors**
| s est rejeté ;
fin

L'objectif est maintenant de trouver une modélisation judicieuse des S-VOPs conservés afin de pouvoir les apparier entre eux par rapport aux objets d'intérêt.

4.5 Classification en deux étapes des S-VOPs

A ce stade, de nombreux S-VOPs correspondent au même objet d'intérêt. Cette section montre comment sont construites les n classes représentant les n objets d'intérêt du plan.

4.5.1 Problématique

Afin d'éviter tous les problèmes inhérents au suivi temporel d'objets (décrochement, occultation, disparition, réapparition, déformation), la méthode de classification des S-VOPs est réalisée sur un critère d'*état ponctuel* des S-VOPs, sans apport d'information spatio-temporelle.

Le choix de la classification s'est porté vers une méthode *hors ligne*, en opposition à une classification *en ligne* orientée 'suivi' : d'une part, à cause de la nature sporadique des S-VOPs instables temporellement et d'autre part, à cause du rapport *efficacité/complexité* des méthodes de suivi dans un contexte de vidéos réelles. En effet de nombreuses recherches ont été menées sur les méthodes de suivi mais cependant, le suivi d'objet constitue un véritable défi auquel les méthodes actuelles ne répondent pas totalement, sans rajouter des contraintes plus ou moins fortes.

Le but étant de trouver une représentation de chaque objet d'intérêt du plan, il n'est pas nécessaire de suivre l'objet tout au long du plan mais simplement d'être capable de connaître le nombre d'objets d'intérêt et de savoir quand ils sont extraits de manière correcte. La classification a donc pour but de limiter les fausses détections d'une part, en limitant le nombre de classes et d'autre part, en regroupant *uniquement* les S-VOPs les plus pertinents appartenant au même objet d'intérêt. La méthode de classification choisie privilégie donc la *précision* dans le rapport *rappel/précision*, souvent évoqué dans l'évaluation de tels algorithmes.

4.5.2 Classification 2 temps des S-VOPs

Dans cette partie, je discute et justifie notre choix de méthode de classification

Le choix du critère de classification est capital. Les plus classiques sont la couleur, la forme, la texture, le coefficient de réflexion... Il est possible d'utiliser plus ou moins de critères. Cependant, plus la dimension de l'espace des données est élevée, plus la constitution des classes est délicate. Il est donc nécessaire de n'utiliser que les critères les plus discriminants par rapport à la population à étudier.

La forme ne peut pas être utilisée ici comme critère de classification de par la nature même des S-VOPs qui représentent tout ou partie d'un objet en mouvement et parce que la forme des objets peut changer radicalement.

La couleur reste un des critères les plus représentatifs d'un objet au cours d'un plan, si on fait l'hypothèse qu'un plan est classiquement d'une durée courte. Les variations de couleur des objets d'intérêt y sont relativement faibles. L'étude de R. Hammoud [Ham02] sur les variations intra-plan d'un objet vidéo, montre que ces dernières concernent essentiellement les variations d'éclairement. C'est pourquoi, le choix du critère s'est porté vers celui de la couleur. L'espace RGB n'étant pas invariant aux changements d'éclairement. ce sont les composantes chromatiques a^* et b^* de l'espace $L^*a^*b^*$ qui sont utilisées.

Afin de réaliser une classification dont le nombre de classes est *a priori* inconnu, une classification en *2 temps* a été retenue :

1. Tout d'abord, chaque S-VOP est considéré comme un objet clé potentiel et génère sa propre classe sur un critère couleur. Chaque classe, ainsi obtenue, appelée Classe Couleur ou *2XC*, est également filtrée selon un critère de cohérence spatio-temporelle pour donner un ensemble de Classes Couleur Cohérentes ou *3XC*. On a donc ici un nombre de classe égal au nombre de S-VOP du plan.
2. Ensuite les *3XC* sont fusionnées pour obtenir idéalement une bijection entre objets d'intérêt et classes : chaque classe correspond à un objet d'intérêt et inversement.

Le fait que le nombre d'objets d'intérêt et donc de classes clés soit inconnu motive le choix d'une telle méthode. De plus, l'étude de la cohérence temporelle est simple à mettre en oeuvre puisqu'il est possible de choisir comme référence spatio-temporelle pour chaque classe, le S-VOP qui a généré la *2XC*. La méthode de classification couleur est orientée *un contre tous* ce qui permet d'obtenir une bonne précision en ce qui concerne la constitution des *2XC*.

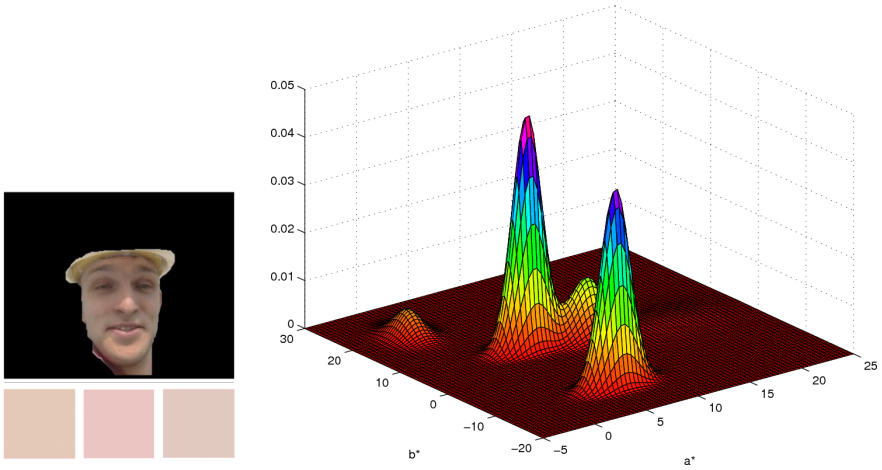
4.5.3 Classification couleur

Cette partie explique comment le critère couleur est utilisé pour caractériser chaque S-VOP et comment on calcule la similarité couleur entre deux S-VOPs, afin de décider s'ils appartiennent à la même classe.

Ici, chaque S-VOP est modélisé par un mélange de Gaussiennes, puis chaque S-VOP générateur de classe est comparé à tous les autres S-VOP à l'aide de cette modélisation. Les S-VOPs possédant des mélanges similaires sont classés dans la même *2XC*. Bien entendu, les classes s'intersectent de façon très importante.

Modèle couleur choisi

Les pixels d'un S-VOP sont utilisés pour constituer un histogramme qui est ensuite modélisé par un mélange de k Gaussiennes à 2 dimensions (cf figure 4.4). Les dimensions correspondent aux deux composantes chromatiques de l'espace couleur $L^*a^*b^*$. Chaque Gaussienne représente un *groupe* de pixels.



(a) Un S-VOP et ses 3 couleurs de poids significatif

(b) Mélange de 5 Gaussiennes dont seulement 3 ont un poids significatif

FIGURE 4.4 – Exemple de modélisation couleur d'un S-VOP

Modélisation par mélange de Gaussiennes

Soit X la variable aléatoire représentant la position du pixel dans le plan a^*b^*. Un mélange de Gaussiennes s'exprime par la fonction de densité de probabilité P suivante :

$$P(X) = \sum_{i=1}^{k} w_i \times G(\mu_i, \Sigma_i, X) \tag{4.7}$$

Où w_i est la proportion de données représentée par la $i^{\text{ème}}$ Gaussienne du mélange telle que $0 < w_i < 1 \forall i \in [\![1, \ldots, k]\!]$ et $\sum_{i=1}^{k} w_i = 1$. μ représente le vecteur des moyennes et Σ la matrice de covariance. $G(\mu_i, \Sigma_i, X)$ est la fonction de densité de probabilité de la $i^{\text{ème}}$ Gaussienne à 2 dimensions donnée par l'expression suivante :

$$G(\mu_i, \Sigma_i, X) = \frac{1}{(2\pi)|\Sigma|^{1/2}} \; exp\left(-\frac{1}{2}\,(x-\mu)^T \Sigma^{-1}\,(x-\mu)\right) \tag{4.8}$$

Les Gaussiennes sont obtenues par l'algorithme itératif des *k-means* dont l'objectif est de découper les données en k groupes appartenant à des distributions de type Gaussien en minimisant la variance intra groupe. L'inconvénient majeur de cet algorithme réside dans le fait qu'il faut fixer au préalable le nombre de groupes. Dans notre cas, le but de la modélisation par mélange de Gaussiennes est de représenter des couleurs dominantes des objets. Expérimentalement, le nombre de Gaussiennes a été fixé à 5.

Comparaison des Gaussiennes

La modélisation des couleurs par mélange de Gaussiennes permet une comparaison pratique et efficace des couleurs deux à deux : pour quantifier le recouvrement (et donc la similarité) de deux Gaussiennes,

on utilise le critère de [Das99] : deux Gaussiennes $\mathcal{N}(\mu_1, \Sigma_1)$ et $\mathcal{N}(\mu_2, \Sigma_2)$ sont *c-séparées* si

$$\|\mu_1 - \mu_2\| \geqslant c\sqrt{2 \cdot max(\lambda_{max}(\Sigma_1), \lambda_{max}(\Sigma_2))} \tag{4.9}$$

Avec $\lambda_{max}(\Sigma_1)$ et $\lambda_{max}(\Sigma_2)$ les plus grandes valeurs propres des matrices de covariance respectives Σ_1 et Σ_2.

Deux Gaussiennes 2-séparées sont considérées comme complètement séparées. Deux gaussiennes 1- ou $^1/_2$-séparées se recouvrent significativement. Ces valeurs permettent d'établir les *2XC* à partir des mélanges de Gaussiennes de chaque S-VOP en quantifiant leur séparation :

- Nous étendons la notion de séparation à la notion de compatibilité en faisant intervenir le poids des Gaussiennes : deux Gaussiennes sont *compatibles* si et seulement si elles sont au plus 1-séparées et sont de poids similaires ($\Delta w \leq 0.1$). La contrainte sur les poids des Gaussiennes permet de ne pas fusionner des S-VOPs d'objet d'intérêt différents ayant des couleurs similaires mais en quantités très différentes.

- La compatibilité permet de définir l'inclusion d'un mélange dans un autre. Soient m_1 et m_2 deux mélanges de Gaussiennes modélisant deux S-VOPs s_1 et s_2. m_1 est inclus dans m_2 si et seulement si chaque Gaussienne de m_1 est compatible avec l'une des Gaussiennes de m_2.

- L'inclusion d'un mélange dans un autre permet de regrouper les S-VOPs correspondants dans la même 2XC (cf. algorithme 7). L'inclusion permet de regrouper les S-VOPs représentant des sous parties d'un même objet d'intérêt.

Algorithme 7 : Critère de fusion des S-VOPs

si $m_1 \subset m_2$ *ou* $m_2 \subset m_1$ **alors**
| s_1 et $s_2 \in$ même 2XC ;
fin

4.5.4 Contrôle de trajectoire dans une classe couleur

Cette partie montre comment dans une classe, on élimine les S-VOPs qui ont une trajectoire non conforme.

Le modèle de classification couleur choisi ne prend pas en compte l'aspect temporel des S-VOPs. En conséquence, des S-VOPs incompatibles spatio-temporellement peuvent coexister au sein d'une même *2XC*. Par exemple, deux visages quasi-immobiles apparaissant éventuellement dans les mêmes images (figure 4.5), deux véhicules similaires qui roulent espacés d'une dizaine de secondes (figure 4.6). Un contrôle antérieur et postérieur de la trajectoire du SVOP générateur de la classe permet de supprimer, par défaut, les S-VOPs non cohérents avec cette trajectoire. Nous montrons ici une version simple de traitement qui améliore de façon certaine les résultats. Les différents paramètres (position, vitesse et taille des SVOPs) étant bruités et les mesures étant incomplètes (SVOPs manquants), il nous paraît indispensable d'améliorer cette première version par un filtre de Kalman.

Image originales

Les S-VOPS extraits appartenant à la même 2XC avant contrôle

FIGURE 4.5 – Le contrôle de trajectoire permet dans une même classe de ne pas mélanger les deux visages dont les caractéristiques couleurs sont très proches

Véhicule 1 Véhicule 2

FIGURE 4.6 – Le contrôle de trajectoire empêche les SVOPs de deux véhicules similaires en couleurs passant à des instants différents d'appartenir à la même classe

Le contrôle consiste, connaissant la position et la vitesse (après compensation du mouvement dominant) du centre de gravité G_{ref} d'un S-VOP de référence S_{ref}, à rechercher itérativement dans les images voisines, les S-VOP correspondants. La toute première référence est le S-VOP générateur S_{gen}. La recherche se fait en deux étapes : postérieurement puis antérieurement à S_{gen}. Elle est itérative à deux titres (algorithme 8) : d'une part pour parcourir les images en s'éloignant de S_{gen} et ainsi contrôler l'ensemble de la trajectoire ; d'autre part pour parcourir tous les S-VOP d'une image pour savoir quels sont ceux qui sont cohérents ou non à la trajectoire du S-VOP de référence courant.

La recherche se fait dans une fenêtre circulaire centrée sur la projection de $G_{ref} = (x, y)$ dont on connaît la vitesse compensée $\vec{V} = (dx, dy)$:

$$proj(G_{ref}) = (x + dx, y + dy) \tag{4.10}$$

Algorithme 8 : Nettoyage d'une classe couleur par contrôle de trajectoire

Données :
$2XC$, une Classe Couleur de S-VOPs
$SVOP_{gen}$, l'élément générateur de $2XC$
t_{gen}, le numéro de l'image contenant $SVOP_{gen}$
t_{fin}, le numéro de l'image contenant le dernier SVOP de $2XC$
r, le rayon de la fenêtre de recherche

Sorties :
$3XC$, la Classe Couleur Cohérente correspondant à $2XC$ sans les SVOPs non cohérents

$3XC = \emptyset$;
$(x, y, dx, dy) = (G, \vec{V})_{SVOP_{gen}}$;
pour chaque *t variant de* $t_{gen} + 1$ *à* t_{fin} **faire**
  $x = x + dx$;
  $y = y + dy$;
  pour chaque $SVOP_i(t) \in 2XC$ **faire**
    si $G_i \subset Cercle(x, y, r)$ **alors**
      $3XC = 3XC \cup SVOP_i$; `/* un SVOP cohérent de plus */`
      $(x, y, dx, dy) = (G, \vec{V})_{SVOP_i}$;
      `/* il devient la nouvelle référence */`
    fin
  fin
  $t = t + 1$;
fin

Le rayon r de la fenêtre de recherche est calculé relativement à l'ensemble de la population de la *2XC* : chaque élément i fournit une valeur r_i égale au plus grand rayon partant de son centre de gravité :

$$r_i = \max_{(x,y)} \|G_i - p(x, y)\| \tag{4.11}$$

Avec G_i : le centre de gravité de l'élément i et $p(x, y)$ un pixel appartenant à l'élément. Soit n le nombre d'éléments de la *2XC*, r est la valeur moyenne des rayons :

$$r = \frac{1}{n} \sum_{i=1}^{n} r_i \tag{4.12}$$

Un élément candidat $S_i(t)$ est cohérent temporellement avec la trajectoire de l'élément de référence S_{ref} si et seulement si :

$$\|G_{S_{ref}} - G_{S_i(t)}\| \leqslant r \tag{4.13}$$

Pour une image donnée $I(t)$, la recherche s'effectue en respectant les règles suivantes :

1. Si aucun (centre de gravité de) SVOP n'est inclus dans la fenêtre de recherche, la recherche recommence dans l'image suivante ou précédente (en fonction d'une recherche postérieure ou antérieure à l'élément générateur), la position de la fenêtre de recherche étant alors incrémentée du vecteur vitesse du SVOP de référence.
2. Si un seul SVOP est inclus dans la fenêtre de recherche, il est conservé dans la classe et c'est lui qui devient la nouvelle référence (position et vitesse).
3. Si plusieurs SVOP sont inclus dans la fenêtre de recherche, il sont conservés dans la classe et c'est leur centre de gravité qui devient la nouvelle référence.
4. Tous les SVOPs dont le centre de gravité est extérieur à la fenêtre de recherche sont exclus de la classe.

Après cette étape, chaque classe ($3XC$) est sensée contenir uniquement des S-VOPs se rapportant à un seul objet d'intérêt. Cependant, à chaque objet d'intérêt est associé plusieurs $3XC$. L'objectif de

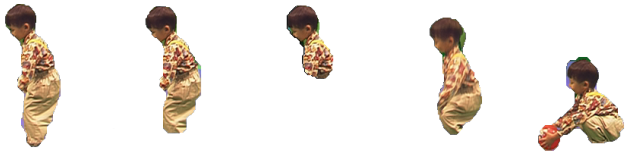


FIGURE 4.7 – Exemple de variation du centre de gravité due à des S-VOPs incomplets et à une déformation de l'objet d'intérêt

l'étape suivante est de fusionner les $3XC$ se rapportant au même objet d'intérêt afin de générer les classes clés.

4.5.5 Fusion hiérarchique des classes couleur

A ce stade, on dispose toujours d'autant de classes que de S-VOPs. Dans cette partie, je montre comment il faut fusionner ces classes afin d'avoir d'une part des classes disjointes et d'autre part une seule classe par objet d'intérêt.

Pour réaliser cette étape, une classification hiérarchique ascendante agglomérative a été choisie. L'indice de dissimilarité et le critère d'agrégation sont présentés dans ce paragraphe. Nous faisons l'hypothèse que les *3XC* concernant le même objet d'intérêt ont un contenu très proche et peuvent être fusionnées sur l'étude de la similarité de leur contenu. Ceci revient à répondre à la question : une classe est-elle globalement incluse dans une autre ? Si oui, les deux classes n'en font plus qu'une.

Indice de dissimilarité

La théorie des ensembles permet de modéliser simplement le problème. En effet, les *3XC* sont des ensembles de S-VOPs et leur similarité peut être évaluée à partir de l'étude de leur intersection.

Soit S_A et S_B deux *3XC*. La dissimilarité [†] entre S_A et S_B qui vérifient $|S_B| \leq |S_A|$ [‡] est donnée par l'expression suivante :

$$d = \frac{|S_B \setminus S_A \cap S_B|}{|S_B|} \text{ §} \tag{4.14}$$

$d = 1$ lorsque l'intersection entre les ensembles est vide et $d = 0$ lorsque $S_B \subset S_A$

Agrégation des classes

A l'aide de cet indice de dissimilarité on agrège itérativement les deux classes les plus similaires jusqu'à ce qu'il n'en reste plus qu'une. A chaque agrégation, il est nécessaire de mettre à jour l'indice entre la classe nouvellement formée et toutes les autres. Il existe plusieurs types de mise à jour de cet indice. Nous avons choisi celle connue sous le nom de saut minimum (*single linkage*) : soit $c_3 = c_1 \cup c_2$ la fusion de plus faible dissimilarité. Les dissimilarités entre la nouvelle classe c_3 et chacune des autres classes c est donnée par :

$$d(c_3, c) = min[d(c_1, c), d(c_2, c)]$$

Cette mise à jour particulière permet d'avantager des fusions centrées sur les classes les plus fédératrices.

†. par dissimilarité, on entend une distance sans l'inégalité triangulaire
‡. $|S_A|$ représente le cardinal de S_A
§. $S_B \setminus S_A \cap S_B$ désigne l'ensemble de tous les éléments de S_B qui n'appartiennent pas à $S_A \cap S_B$, autrement dit, ceux qui n'appartiennent qu'à S_B (figure 4.8)

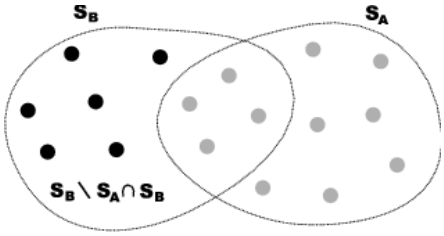


FIGURE 4.8 – Exemple d'intersection de deux classes

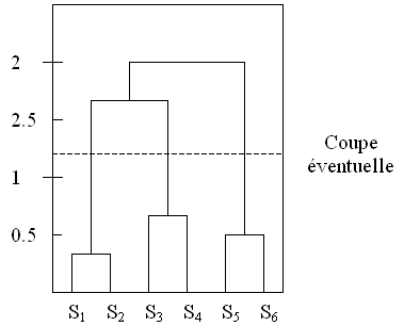


FIGURE 4.9 – Exemple d'un dendrogramme de 6 *3XC*. La coupure induit 3 classes

Représentation sous forme de dendrogramme

Le résultat de la classification ascendante hiérarchique est plus facilement visualisable sous forme de dendrogramme (cf fig. 4.9) qui donne la composition des différentes classes ainsi que l'ordre dans lequel elles ont été formées. L'axe vertical donne la valeur de l'indice d'agrégation pour un groupement donné. Cette représentation permet également de visualiser les sauts de l'indice afin de définir une éventuelle partition.

Détermination de classes clés

Il est généralement pertinent de couper le dendrogramme de classification à l'endroit où est observé un saut dans les valeurs d'agrégation. Il est alors possible d'obtenir une partition de bonne qualité car les individus regroupés en dessous de la coupure (i.e. du seuil) sont proches tandis que les individus situés au dessus sont éloignés. On calcule donc un seuil qui va maximiser l'inertie I_i entre deux sous-ensembles E_i et F_i :

Soit E_i l'ensemble des dissimilarités ≥ 0 et $< i$. Soit F_i l'ensemble des dissimilarités $\geq i$ et < 1 (on exclut les dissimilarités égales à 1 qui indiquent que deux *3XC* sont disjointes). Soit $D = E_i \cup F_i$. À D, E_i, F_i on associe leur moyenne respective m_D, m_{E_i}, m_{F_i}. L'inertie est donnée par :

$$I_i = w_e d(m_{E_i}, m_D)^2 + w_f d(m_{F_i}, m_D)^2 \tag{4.15}$$

Où $w_e = |E_i|, w_f = |F_i|$ et d est la distance Euclidienne.

Le meilleur partitionnement est atteint pour une valeur de i qui maximise l'inertie. Toutefois, si ce seuil est trop faible, le risque est d'obtenir trop de classes. Pour cette raison, on définit empiriquement un seuil minimum m. Le seuil d'agrégation recherché est donc :

$$S_a = min(m, \underset{i}{\text{argmax}}(I_i)) \tag{4.16}$$

Le minimum m est placé à 0.5 afin de fusionner les classes ayant au moins en commun la moitié de leurs éléments. Comme le montre la figure 4.9, le calcul de S_a permet de fixer directement le nombre et la constitution des différentes classes clé.

4.6 Suppression des classes temporellement non significatives

Il est nécessaire de contrôler la validité temporelle des classes obtenues. Dans cette section, on définit deux critères simples permettant de supprimer les classes temporellement non significatives : la *durée* et la *persistance*.

Pour une classe donnée, nous savons qu'elle contient n S-VOPs dont les premières et dernière apparitions chronologiques sont I_{deb} et I_{fin}. Nous en déduisons la durée (en nombre d'images) et la persistance (ou taux d'apparition) calculée sur la durée. Nous faisons l'hypothèse qu'un objet d'intérêt reste à l'image pendant une durée significative d et que pendant cette durée, il est extrait p% du temps. d et p sont fixés expérimentalement ($d = 50$ i.e. 2 secondes et $p = 20$% par exemple) et permettent de valider ou de supprimer chacune des classes (algorithme 9).

Algorithme 9 : Critère temporel

Données :
C une classe clé
I_{deb} le plus petit numéro d'image où apparaît un S-VOP $\in C$
I_{fin} le plus grand numéro d'image où apparaît un S-VOP $\in C$
n le nombre de S-VOP $\in C$

$duree = I_{fin} - I_{deb} + 1$;
$persistance = n/duree$;
si $duree < 50$ *ou* $persistance < 0.2$ **alors**
| C est supprimée ;
fin

4.7 Sélection de l'objet clé et des vues clés

Nous disposons maintenant des classes définitives. Selon nos hypothèses, une classe correspond à un objet d'intérêt. Dans cette section, j'explique comment nous sélectionnons les représentants d'une classe : l'objet clé et plusieurs vues clés.

4.7.1 Objet clé

On peut maintenant sélectionner un unique objet-clé dans chaque classe C. L'équation 4.5 page 76 présente le critère qui a permis d'estimer la qualité d'un masque de S-VOP en terme de segmentation. Ce critère est à nouveau utilisé ici, et c'est le S-VOP qui maximise le critère dans un sous-ensemble $\widehat{C}$ de C qui est l'objet clé de la classe.

Voici comment est composé $\widehat{C}$: comme le critère utilisé s'exprime en pourcentage, les petits S-VOPs sont avantagés au détriment des plus grands. Pour contourner ce problème de maximum local, on estime l'intervalle le plus représentatif des aires de C : C est découpée en 3 sous-ensembles disjoints selon les aires de ses S-VOPs : faibles, moyennes et élevées (figure 4.10). Cette classification est à nouveau réalisée à l'aide de l'algorithme des *k-means*. $\widehat{C}$ est le sous-ensemble dont la qualité moyenne de masque est la plus élevée. C'est lui qui fournit l'objet-clé.

4.7.2 Vue clé

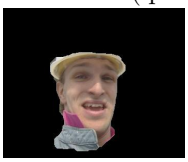
On propose ici d'extraire automatiquement un S-VOP supplémentaire constituant une vue clé représentant un aspect différent de l'objet d'intérêt. La méthode retenue utilise une approche contour. Le principe est de rechercher dans l'ensemble $\widehat{C}$ le S-VOP dont la répartition des contours (cf fig. 4.11.b et 4.11.c) est la plus dissemblable de celle de l'objet clé. Pour cela, chaque S-VOP de $\widehat{C}$ est modélisé par une ellipse qui est ensuite comparée à l'ellipse de l'objet clé (cf fig. 4.11.d).

Pour garantir l'invariance en rotation, il convient de mettre en correspondance au mieux les 2 ellipses à comparer. Les deux axes principaux d'une ellipse la découpent en 4 quartiers comme le montre la figure 4.12.b. Chaque quartier i du S-VOP ainsi délimité fournit une valeur moyenne de la norme du gradient μG_i, calculée sur les pixels qu'il contient. Les μG_i permettent de calculer une différence d (cf eq. 4.17)

(a) Les S-VOPs d'aires élevées (qualité moyenne : 0.42)

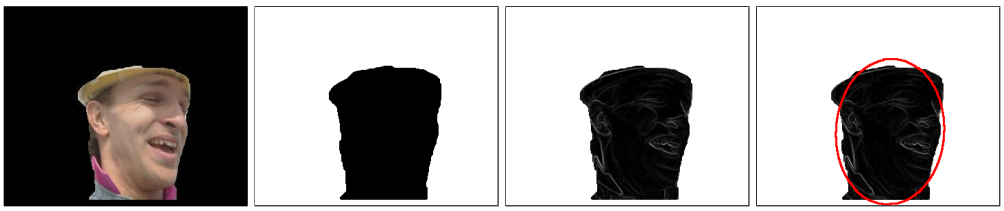
(b) Les S-VOPs d'aires moyennes (qualité moyenne : 0.37)

(c) Les S-VOPs d'aires faibles (qualité moyenne : 0.41)



(c) L'objet clé

FIGURE 4.10 – Découpage d'une classe en 3 sous-ensembles : le sous ensemble (a) qui fournit les masques de meilleure qualité fournit également l'objet-clé



(a) S-VOP extrait de l'image #111 de Foreman)

(b) Masque du S-VOP

(c) Norme du gradient

(d) Modélisation ellipsoïdale

FIGURE 4.11 – Extraction des données contour

entre deux ellipses e_1 et e_2 pour chacune des 4 positions possibles correspondant à une permutation circulaire p des quartiers de e_1 par rapport à e_2.

$$d(e_1, e_2, p) = \sqrt{\sum_{i=1}^{4} (\mu G(e_1)_i - \mu G(e_2)_{(p+i) \bmod 4})^2)} \text{ avec } p \in [\![0, 3]\!] \tag{4.17}$$

La meilleure correspondance entre l'ellipse de l'objet clé et celle du S-VOP candidat est celle qui minimise la différence d et celle qui est utilisée.

Soient e_{oc} l'ellipse de l'objet clé et e_c l'ellipse du S-VOP candidat. La vue clé d'une classe est le S-VOP qui maximise la meilleure correspondance :

$$I_c = \max_c(\min_p(d(e_{oc}, e_c, p))) \tag{4.18}$$

Comme le montre l'équation 4.18, la vue clé est le S-VOP dont l'ellipse correctement comparée (la

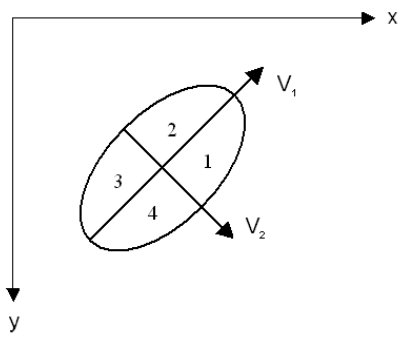


FIGURE 4.12 – Découpage de l'ellipse d'approximation

correspondance la plus probable parmi 4) est la plus dissemblable en terme de contours à l'ellipse de l'objet clé.

Afin de sélectionner N vues clé ($N > 1$), il faut calculer I_c pour chaque paire de S-VOPs de $\widehat{C}$. Le principe est alors de trouver l'ensemble des N S-VOPs maximisant les différences entre eux. Ce travail n'a pas été réalisé par manque de temps mais constitue une fonctionnalité intéressante pour l'utilisateur.

4.8 Résultats

Nous illustrons nos traitements avec deux vidéos. La première, appelée *vélo* dure 3 secondes et comporte 90 images. On y voit un vélo qui rentre dans le champ de la caméra, le traverse et en sort. La caméra est immobile mais est tenue à la main. L'objet cycliste a une surface qui varie d'un facteur 3 environ (figure 4.13.c). Le traitement extrait une classe comportant 14 S-VOPs (figure 4.13.a). Cet exemple montre bien que la méthode peut être utilisée comme suivi à part entière d'objets en mouvement. La figure 4.13.b montre les images originales d'où sont extraits les S-VOPs. On remarque que le fond est complexe et que la cycliste n'est pas bien contrastée avec le fond. Néanmoins, sans être parfaits, les différents S-VOPs sont assez stables et descriptifs par rapport à l'objet d'intérêt.

La seconde vidéo, nommée *Chavant* montre la circulation en ville. La caméra, toujours tenue à la main, se comporte de diverses façons : elle est fixe, puis effectue quelques panoramiques dans le sens des véhicules puis dans le sens contraire. Ces mouvements sont accompagnés de zooms avant et arrière. Douze véhicules traversent le champ de la caméra de droite à gauche. Deux personnages passent au premier plan et se croisent. La vidéo dure 18 secondes c'est-à-dire 540 images. 12 objets-clé sont extraits (figure 4.14.a) au lieu de 14 espérés. Parmi eux, 6 voitures de couleur gris métallisé très similaires et deux piétons. La segmentation est de bonne qualité, les objets-clé ne débordent pas sur le fond et il est assez facile par exemple de reconnaître le modèle de chaque véhicule. Deux voitures (identiques) ne sont pas extraites. Elles se suivent et sont partiellement occultées par des poteaux qui les "découpent" en plusieurs morceaux (figure 4.14.b). On peut supposer qu'elles ont généré d'une part peu de S-VOPs (la caméra ne les suit pas) et d'autre part des S-VOPs trop petits. En conséquence de quoi, les classes correspondantes ont dû être supprimées.

4.9 Conclusion

L'étape de sélection de l'objet clé permet d'obtenir un masque binaire relativement caractéristique de l'objet d'intérêt. Bien qu'il ne recouvre généralement pas totalement l'objet d'intérêt, la correspondance frontière/contour du masque avec l'objet est de qualité suffisante pour envisager une initialisation et/ou

(a) Les 14 S-VOPs extraits

(b) Les images originales correspondantes

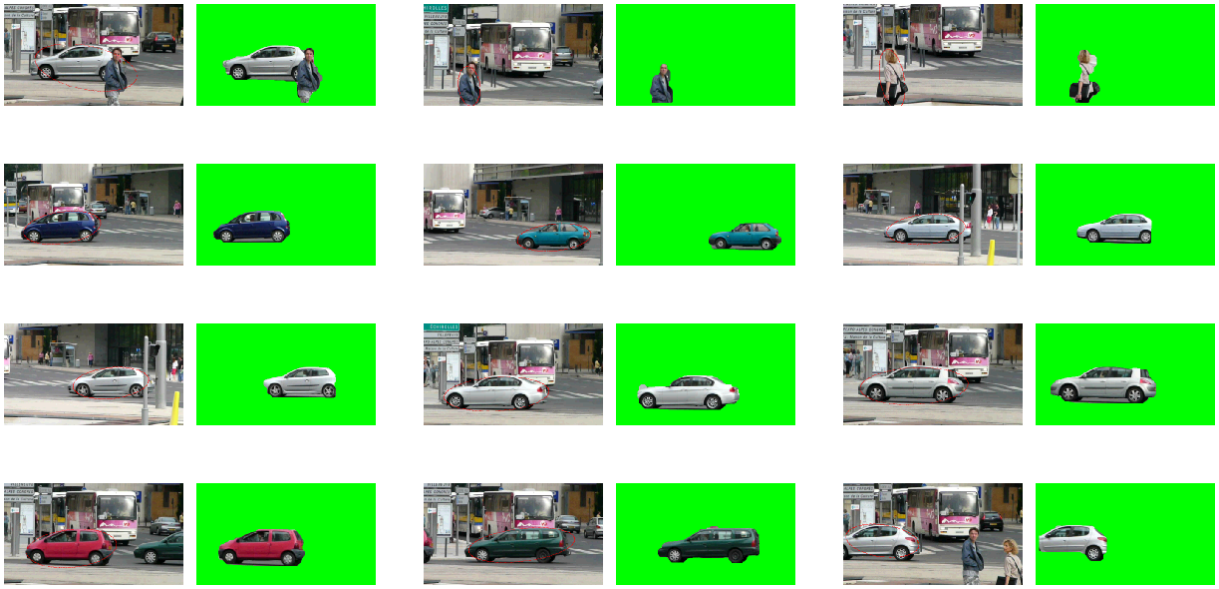
(c) Mixage des images 25, 39 et 63 montrant le léger bouger de la caméra et la variation de taille de l'objet d'intérêt

FIGURE 4.13 – Extraction d'un objet-clé dans la séquence *vélo* (les numéros des images sont indiqués)

un contrôle efficace de suivi tel que celui présenté au chapitre précédent à la section 3.4. Le masque de l'objet clé fournit une initialisation automatique intéressante pour ce type d'application dont le principal défaut est l'initialisation manuelle.

Le suivi peut également être remis en cause lorsque l'image traitée correspond à l'image d'où est extraite une éventuelle vue clé. En effet il est intéressant de pouvoir confronter la vue clé avec le résultat courant obtenu par le suivi. Si les divergences sont trop importantes, il est possible de prendre la décision de réinitialiser le processus de suivi à l'aide de cette vue clé.

La phase d'évaluation comparative de notre approche n'a pas été réalisée. Comme nous avons orienté notre recherche dans une direction peu suivie, il est difficile d'envisager une procédure assez simple et systématique. Toutefois, il nous semble probable que les critères subjectifs sont plus représentatifs que les critères objectifs. Cette démarche ne peut être envisagée sérieusement qu'avec l'aide de spécialistes du domaine psycho-visuel avec lesquels il faudrait travailler.



(a) Ici, on présente un couple d'images par objet-clé : l'image originale où a été extrait l'objet-clé et le masque correspondant

(b) Les 2 voitures non détectées par la technique

FIGURE 4.14 – Extraction de 12 objets-clé dans la séquence *Chavant*

Chapitre 5

Segmentation de personnes

Sommaire

Ce chapitre s'intéresse au cas particulier de la segmentation des personnes dans les images, lorsqu'elles sont debout et en majeure partie visibles. Je présente ici les travaux en cours de la thèse de Cyrille Migniot qui a construit une méthode dont les deux points clé sont une modélisation par gabarits et une utilisation particulière de la technique bien connue du *graph-cut*.

5.1 Introduction

De tous les "objets" présents dans une image ou une vidéo, les personnes sont sans doutes ceux qui intéressent le plus les chercheurs et les applications : détection de visages et de leurs diverses composantes pour la biométrie, la photographie, la robotique, les IHM, la surveillance, la gestion par le contenu, ...). La détection des personnes est parfois insuffisante et une localisation plus précise de leur silhouette est nécessaire pour pouvoir étudier leurs gestes, leur comportement ou pour réaliser de l'édition d'image (extraction / incrustation) souvent prises dans des conditions non optimales.

En raison de la grande variabilité des couleurs et des textures qu'une personne peut porter mais également par les différentes positions qu'elle peut prendre, cette tâche est un vrai défi. Dans ce domaine comme dans de nombreux autres en analyse d'image et en vision par ordinateur, le but est d'éviter une supervision quelconque par un utilisateur.

Pour la détection et la segmentation de personnes, de nombreuses méthodes utilisent des gabarits binaires. Un gabarit est un modèle permettant de caractériser la forme générale des éléments d'une classe. Dans celle des personnes, les postures pouvant apparaître sont assez variées et influencent vraiment la segmentation. Un gabarit représente alors l'allure de la silhouette pour une de ces postures par un masque binaire. Un catalogue de gabarits est réalisé contenant toutes les postures que la personne peut prendre. Ces gabarits sont ensuite comparés un par un aux caractéristiques de l'image (souvent les contours [ZD05]). Si la comparaison est positive, une personne est détectée et le gabarit donne sa posture. La

segmentation est finalement obtenue en adaptant légèrement la silhouette de ce gabarit aux contours de l'image [RS07]. Pour diminuer le temps de calcul, Gavrila et al [GG02] réalisent une répartition hiérarchique des gabarits. Enfin, Lin et al [LDDD07] décomposent le corps en trois parties (le torse, le bassin et les jambes) et cherchent le sous-gabarit correspondant à chacune de ces parties.

5.2 Coupe de graphe

Le *graph-cut* ou coupe de graphe, telle que définie par Boykov et al [BJ01], est une méthode efficace de segmentation. Assez simple, elle possède également l'avantage de permettre une interaction facile avec l'utilisateur. Elle est donc couramment utilisée en segmentation d'images [RKB04]. Le graphe considéré est constitué d'une source et d'un puits correspondant au premier et à l'arrière plan et de noeuds correspondant aux pixels de l'image. Des arêtes de voisinage relient les pixels voisins spatialement et des arêtes de liaisons relient les pixels au puits et à la source. Des pondérations sont associées aux arêtes. La coupe du graphe qui minimise la somme des pondérations des arêtes coupées (ou flot) est alors calculée et sépare les pixels du premier et de l'arrière plan. Les pondérations des arêtes de voisinage sont reliées aux contours de l'image alors que les pondérations des arêtes de liaison sont reliées à la probabilité du pixel d'appartenir au premier ou à l'arrière plan (généralement sur un critère de couleur, grâce aux indications de l'utilisateur).
Un certain nombre de travaux ont déjà été réalisés pour introduire des contraintes de forme dans la coupe de graphe. Un troisième terme a été rajouté par Freedman et al [FZ05] à la fonction d'énergie pour prendre en compte la forme à partir de lignes de niveau. Le terme de région ou de contour (ou les deux) ont été modifiés dans plusieurs travaux [WZ10, SU05, DVZB09]. Malcolm et al [MRT07] ont proposé une solution intéressante mais coûteuse en temps de traitement où une pré-image obtenue par apprentissage avec une ACP à noyau est itérativement réactualisée.

5.3 Approche proposée

Nous introduisons une nouvelle technique [MBC11b, MBC11a] pour adapter la coupe de graphe aux caractéristiques des personnes sans interaction de l'utilisateur. Seule la **segmentation** est prise en compte, la **détection** préliminaire des personnes étant effectuée par la méthode de Dalal et al [DT05] qui fournit des fenêtres normalisées centrées sur la personne (figure 5.5). Notre contribution est dans un premier temps de pondérer le graphe par la silhouette moyenne obtenue sur une base d'apprentissage et appelée gabarit, et ensuite d'adapter cette technique avec un gabarit par parties construit par des coupes de graphe successives appliquées sur chaque partie du corps de la personne.
En section 5.4, nous introduisons un gabarit non binaire à partir d'une base de données d'apprentissage, qui représente la probabilité d'un pixel d'appartenir à la silhouette de la personne. Cette probabilité est utilisée pour initialiser la pondération des arêtes de liaison.
Ensuite, en section 5.5, la segmentation est affinée pour s'adapter aux différentes postures : l'image et donc le corps sont divisés en plusieurs parties. Pour chaque partie, plusieurs sous-gabarits sont testés et le gabarit final (appelé gabarit par parties) est obtenu par concaténation des meilleurs sous-gabarits.
Finalement, en section 5.6, nos deux approches (avec un gabarit unique ou un gabarit par parties) sont évaluées.

5.4 Création du graphe

Le graphe est réalisé à partir d'une fenêtre englobant une personne et issue de la détection. Une source F et un puits B représentent le premier et l'arrière plan. Chaque pixel est relié dans le graphe aux pixels voisins par des arêtes de voisinage et à F et B par des arêtes de liaisons. L'importance relative des pondérations des arêtes de voisinage et de liaison est réglée par deux coefficients (α et β).

5.4.1 Arêtes de voisinage

Les arêtes de voisinage représentent la possibilité que la transition entre deux pixels voisins soit sur la circonférence de la silhouette découpée par la coupe de graphe. Elles sont donc logiquement associées aux contours de l'image. La différence d'intensité est utilisée comme dans la méthode de Boykov [BJ01]. Soit I_p l'intensité du pixel p et I_q l'intensité du pixel q. La pondération associée à l'arête entre p et q est définie par :

$$\omega_{pq} = \beta e^{-\frac{|I_p - I_q|^2}{2\sigma^2}} \tag{5.1}$$

où σ réalise le filtrage opéré par l'exponentielle. Les valeurs faibles représentent les fortes probabilités de contour.

5.4.2 Arêtes de liaison

Les arêtes de liaison rattachent tous les pixels à F et B. Pour un pixel, la coupe passe par une et une seule de ces arêtes, ce qui désigne à quelle région est assigné le pixel. Leur pondération doit donc correspondre à la probabilité du pixel d'appartenir au premier ou à l'arrière plan. Boykov et al [BJ01] relient cette probabilité à la distribution des couleurs. Mais la grande variété des couleurs dans la classe des personnes rend cette caractéristique peu discriminante et nécessiterait l'intervention d'un utilisateur. La forme de la silhouette est une information plus pertinente. On réalise alors un gabarit qui représente la probabilité t_p de chaque pixel p d'appartenir à une silhouette humaine. Ce gabarit est la moyenne d'un ensemble de 200 silhouettes représentatives des différentes postures debout (figure 5.1). Les pondérations attribuées aux arêtes de liaison sont alors définies par :

$$\omega_p^F = -\alpha ln(t_p) \tag{5.2}$$

$$\omega_p^B = -\alpha ln(1 - t_p) \tag{5.3}$$

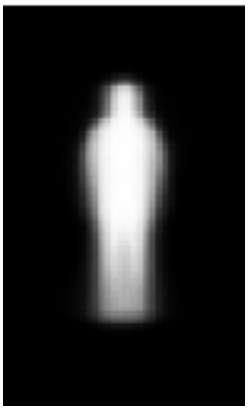


FIGURE 5.1 – Image moyenne d'une base de 200 silhouettes humaines. Notons que la position de la tête et du torse est bien plus stable que celle des bras et surtout que celle des jambes.

5.5 Gabarit par parties

Le gabarit présenté précédemment prend en compte toutes les postures mais défavorise cependant celles d'occurrences les plus faibles (notamment lorsque les bras ou les jambes sont écartés). L'idée est alors de choisir un gabarit qui corresponde à la posture rencontrée. Il serait possible de réaliser une coupe de graphe à partir de gabarits représentant toutes les postures possibles. Mais le nombre de postures étant élevé, on obtiendrait un temps de traitement très important. Nous proposons de partager l'image

en cinq parties : la tête, les parties droite et gauche du torse incluant le bras, la jambe droite et la jambe gauche. Pour chacune de ces parties, un certain nombre de sous-gabarits sont construits pour les différentes postures possibles (figure 5.2). Pour chaque partie, une coupe de graphe est réalisée à partir de chacun des sous-gabarits. Le gabarit par parties obtenu est la concaténation des sous-gabarits ayant

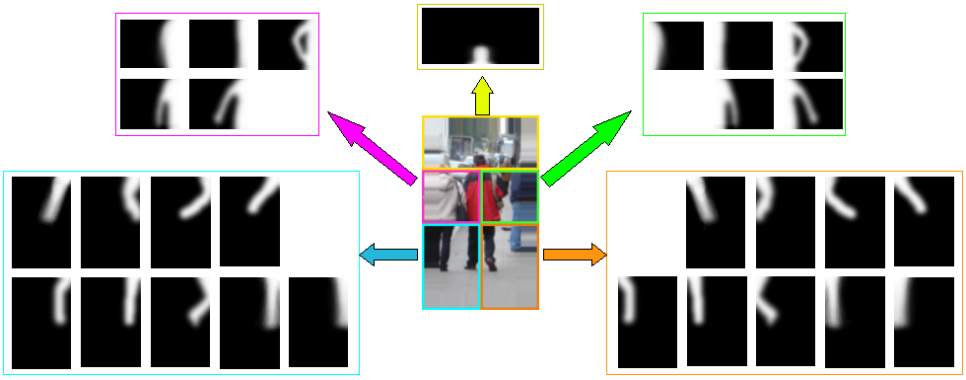


FIGURE 5.2 – L'image est découpée en cinq parties. Des sous-gabarits représentent les postures possibles dans chacune d'entre elles.

donné le flot de coupe le plus faible pour chaque partie (figure 5.3). Enfin, une coupe de graphe sur l'image entière à partir de ce gabarit par parties permet d'assurer la continuité de la silhouette segmentée.

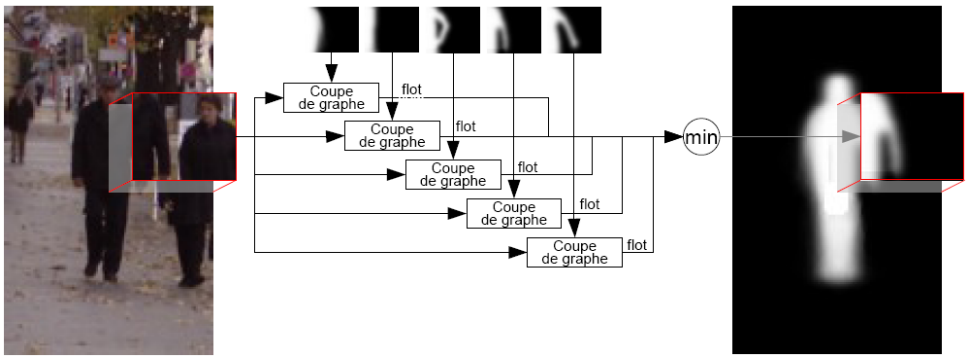


FIGURE 5.3 – Pour chaque partie de l'image, des coupes de graphe sont réalisées à partir de plusieurs sous-gabarits. Celui associé à la coupe donnant le flot minimal est alors ajouté au gabarit par parties.

Un nombre de sous-gabarits restreint est suffisant au traitement. En effet, comme les arêtes de voisinage adaptent la coupe aux contours, les gabarits doivent seulement favoriser un état (bras décollé, jambe pliée, ...) et non parfaitement correspondre à la silhouette de la personne.

5.6 Performances

Pour évaluer notre méthode nous avons réalisé des tests sur un ensemble de 400 images de personnes issues de la base de données statique de l'INRIA. La $F_{measure}$ et la mesure de Yasnoff [PFG08], par comparaison avec une réalité terrain que nous avons réalisée manuellement et qui est disponible dans [MBC10], nous permettent d'évaluer objectivement une segmentation. Les moyennes de ces deux mesures pour les segmentations de l'ensemble des images testées donnent alors une quantification des performances de notre méthode. Les temps de traitement sont observés à partir d'une implémentation C++ non optimisée sur un processeur Pentium D 3GHz.

5.6.1 Réglage optimal du procédé

Pour régler de façon optimale le procédé, nous avons déterminé les valeurs des paramètres α, β et σ qui produisent les meilleures segmentations. Les résultats affichés dans la figure 5.4 sont obtenus avec un gabarit par parties mais ceux obtenus avec un gabarit unique sont très semblables. Obtenir les meilleures segmentations revient à minimiser la mesure de Yasnoff et maximiser la $F_{measure}$. Nous choisissons de garder pour la suite les valeurs : $\sigma = 9$, $\alpha = 12$ et $\beta = 60$.

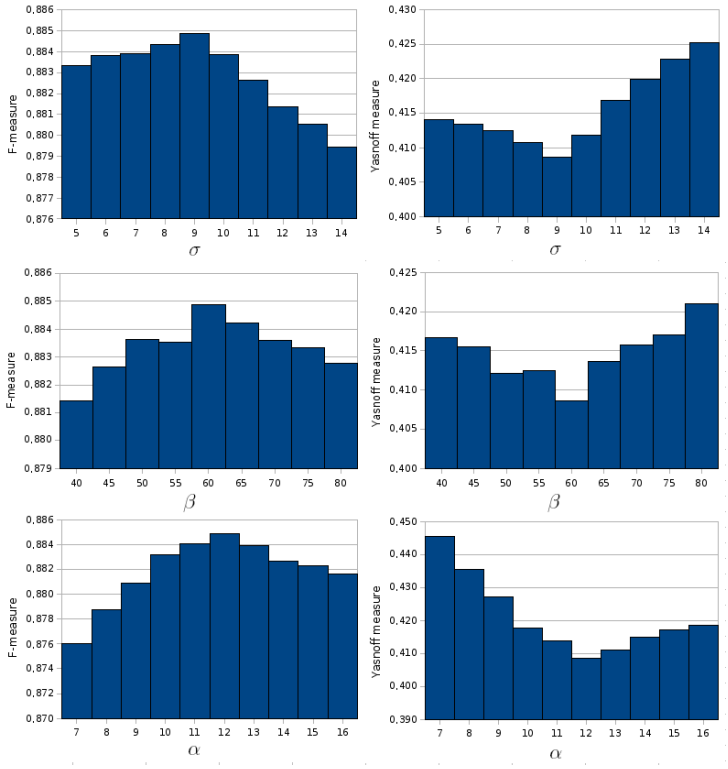


FIGURE 5.4 – Résultats des tests pour optimiser σ (première ligne), β (seconde ligne) et α (troisième ligne). La $F_{measure}$ (première colonne) et la mesure de Yasnoff (seconde colonne) donnent une évaluation de la qualité de la segmentation.

5.6.2 Gabarit unique ou par parties

Puisque nous avons introduit deux procédés différents, il faut comparer leurs performances respectives. Tout d'abord le traitement avec un gabarit unique est logiquement plus rapide avec (pour le traitement d'images 96 × 160 pixels) une moyenne de 12 ms par image contre une moyenne de 70 ms avec le gabarit par parties. Concernant la qualité de la segmentation, en utilisant les valeurs de paramètres fixées précédemment, on obtient les résultats du tableau 5.1 : le gabarit par parties donne en moyenne une meilleure $F_{measure}$ et une meilleure mesure de Yasnoff. Dans la grande majorité des cas, la gabarit par parties réalise une segmentation visuellement mieux adaptée et plus précise (figure 5.5).

Mesure	Gabarit unique	Gabarit par parties
$F_{measure}$	0,8813	0,8849
Yasnoff	0,4178	0,4087

TABLE 5.1 – Sur un ensemble de 400 tests et pour deux mesures objectives, le gabarit par parties donne de meilleurs résultats que le gabarit unique

5.7 Conclusion

Nous avons proposé une nouvelle méthode de segmentation adaptant la coupe de graphe traditionnelle au cas particulier des personnes. Pour cela, nous avons introduit des gabarits non binaires pour évaluer la localisation de la silhouette. Le gabarit est soit général à toute la classe soit adapté à la posture de la personne. Le traitement est efficace et proche du temps réel.
L'étude de séquences vidéo ainsi que la possibilité d'une interaction facile avec l'utilisateur sont parmi les principaux avantages de la coupe de graphe. Des travaux futurs pertinents seraient alors d'adapter notre méthode aux vidéos et de permettre une interaction performante avec la classe des personnes.

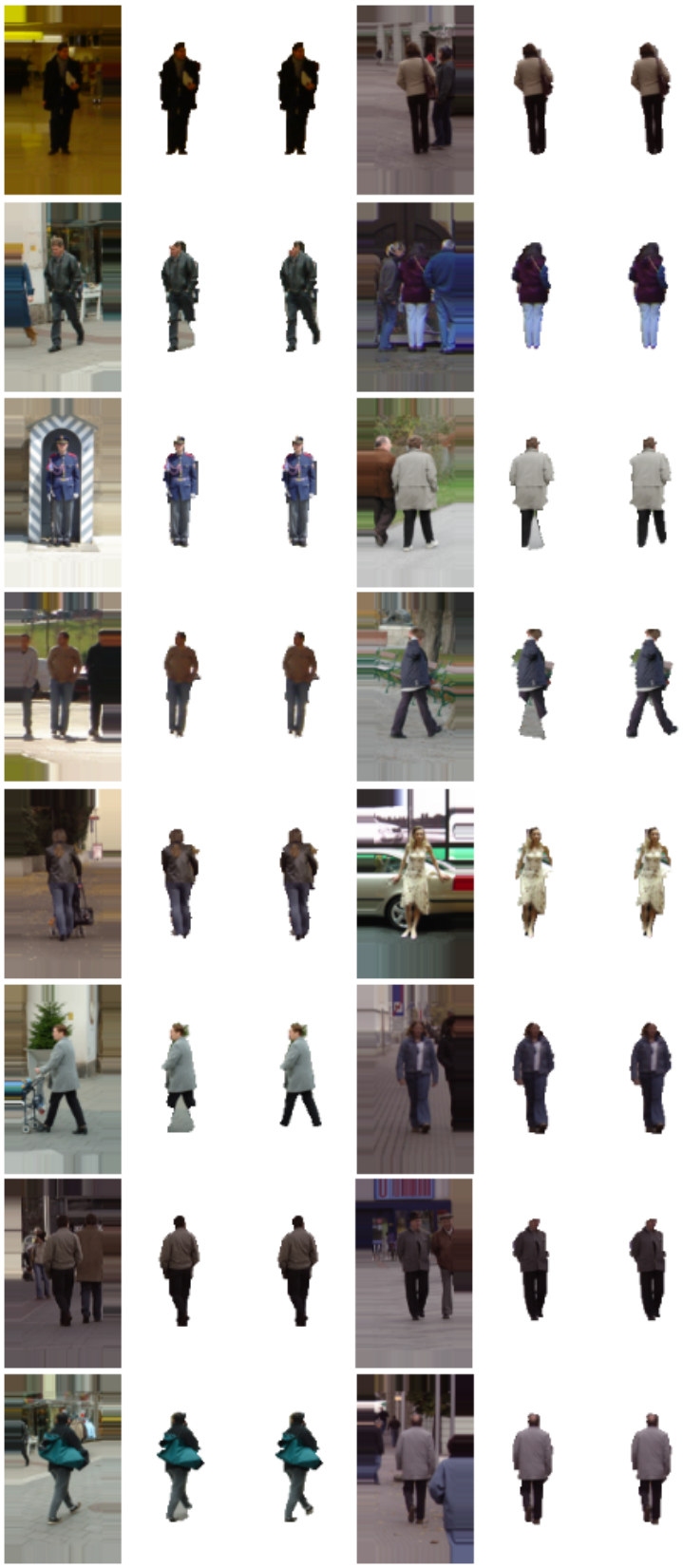


FIGURE 5.5 – De gauche à droite : image initiale, segmentation obtenue avec un gabarit unique puis avec un gabarit par parties. Dans la plupart des cas, la segmentation est plus précise avec le gabarit par parties.

Chapitre 6

Désentrelacement d'images par graphes

Sommaire

6.1 Introduction : l'avènement des écrans plats

Quiconque a poussé les portes d'un magasin d'électro-ménager ces dernières années a remarqué que les télévisions et les moniteurs à tubes cathodiques ont été totalement remplacés par des écrans plats (LCD ou plasma). Intéressons-nous à la technologie LCD : un écran LCD est avant tout constitué d'une dalle qui est une matrice où sont affichés les pixels de l'image. On compte peu de fabricants de dalles : LG/Philips, Samsung, Toshiba, Chi Mei Optoelectronics, AU Optronics, HannStar. A partir de quelques dalles génériques, les fabricants d'écrans fournissent des centaines de modèles différents. Ce qui différentie deux écrans dont les dalles sont identiques est toute l'électronique analogique et numérique et les pré-traitements effectués sur l'image avant qu'elle ne soit affichée. Citons parmi ces derniers : la mise à la bonne résolution de l'image (*scaling*), la correction gamma, le rehaussement de contraste, le désentrelacement.

Ainsi, la différence de qualité entre deux écrans provient-elle en majeure partie des traitements d'amélioration de l'image à afficher. La qualité est bien entendu un argument majeur des constructeurs, et si on prend soin de regarder des images sur des moyens ou grands écrans LCD, on remarque qu'elle est encore largement (mais difficilement) perfectible. La maîtrise de la qualité d'image des TVs à écrans plats devient un facteur stratégique déterminant pour prendre des parts dans un marché mondial qui doit remplacer dans un court terme 1,2 milliard de télévisions).

Il y a quelques années, notre laboratoire a été contacté par la division Home Video de STMicroelectronics dans le but de l'aider à développer des traitements pour améliorer la qualité des images des écrans LCD. STMicroelectronics est un grand constructeur de composants, décodeurs MPEG-2 et set-top box *

∗. "boîtiers de télévisions" qui permet de récupérer, décoder et restituer un signal audio et vidéo

pour la télévision numérique.

Cette recherche concerne notre collaboration avec le projet IQI (Image Quality Improvement) par le biais de la thèse CIFRE de Jérôme Roussel. Le but de ce projet est de développer des algorithmes de désentrelacement qui devront s'intégrer dans une chaîne de traitement d'image vidéo pour écrans plats, en tenant compte des contraintes liées à cet environnement : bande passante, mémoires, latence, etc ... Cette recherche a donné lieu à un brevet américain ([RBN08]).

6.2 Contexte

Le signal vidéo analogique est diffusé dans le monde entier en trames entrelacées. Pour des raisons techniques liées à des contraintes économiques de bande passante, le signal TV a été choisi en fonction de la fréquence du réseau électrique. En outre, la persistance rétinienne permet de recréer le mouvement à partir de 15 à 20 images par seconde. Ainsi la fréquence adéquate serait de 25 images par seconde, cependant le scintillement de larges zones reste très visible. L'entrelacement a permis d'y remédier : l'image est séparée en deux trames qui représentent le champ pair (ensemble des lignes paires) et impair (ensemble des lignes impaires). A l'acquisition vidéo, ces 2 champs sont séparés par 20ms (dans le cas d'un signal pal/secam). Les films (24 images/s) convertis en vidéos (50 ou 60 images/s) utilisent le téléciné (procédé 3 :2 pulldown) qui introduit également l'entrelacement d'images prises à des temps différents.

L'entrelacement provoque aussi du scintillement sur les objets comportant des hautes fréquences horizontales. De plus, aujourd'hui, les écrans plats de type plasma et L.C.D ont un affichage progressif qui nécessite au temps t l'affichage de l'image entière, donc deux champs dont l'origine temporelle est décalée de 20ms. Une vidéo entrelacée visualisée sur un écran plat est difficile à regarder (figure 6.1). Les méthodes permettant de passer de l'entrelacé au progressif sont appelées désentrelacement. Elles doivent prendre en compte les éventuelles modifications locales ou globales dans l'image intervenues entre deux champs espacés de quelques ms. Il est à noter que la plus grande partie de la production vidéo et cinématographique réalisée jusqu'à aujourd'hui est de type entrelacé. En conséquence, tout lecteur de DVD de salon est équipé d'une technique de désentrelacement souvent peu sophistiquée mais qui permet de supprimer l'effet très désagréable de peigne vu en figure 6.1. Les lecteurs sur micro-ordinateur (VLC par exemple) permettent de désactiver le désentrelacement ou de choisir une méthode de désentrelacement parmi plusieurs. La plupart des disques Blu-ray conservent le format original des films (1080p24) et n'ont donc pas besoin de désentrelacement.

Les méthodes de désentrelacement consistent à partir d'un ou plusieurs champs (c-à-d. une ou plusieurs moitiés d'images) d'interpoler une image entière tout en limitant les artéfacts inhérents à cette transformation. Ces méthodes peuvent être classées en deux grandes familles, les méthodes sans compensation et avec compensation de mouvement [SN99]. Nous nous intéresserons ici aux méthodes sans compensation de mouvement. Celles-ci peuvent à leur tour être décomposées en méthodes temporelles, spatiales et adaptatives. La méthode adaptative consiste à privilégier la méthode temporelle lorsqu'il n'y a pas de mouvement, sinon la méthode spatiale qui possède certaines limitations [Koi94]-[LCC03]. La technique proposée [RBN06, RBN07, RB08] est une amélioration significative des méthodes spatiales. Dans un premier temps, les méthodes existantes seront passées en revue puis dans un deuxième temps la méthode proposée sera détaillée. Enfin, des résultats seront commentés.

6.3 Méthodes existantes

Par la suite, f_{in} représente l'image entrelacée et $\tilde{f}$ l'image interpolée. (i, j) sont les coordonnées spatiales où i représente les lignes et j les colonnes. f_{in} n'est définie que pour la moitié des lignes, c'est-à-dire pour i pair ou impair.

(a) Image désentrelacée en *Weave*

(b) détail 1

(c) détail 2

FIGURE 6.1 – Exemple d'une vidéo affichée sur un écran progressif avec un désentrelacement de type *Weave* où deux trames successives sont combinées pour former l'image entière. On aperçoit très bien l'effet de peigne dû au mouvement de la caméra ou des joueurs entre les deux trames acquises à 20ms d'intervalle

Les méthodes et les solutions proposées pour faire de l'interpolation spatiale sont nombreuses [dHB98]. La méthode la plus directe consiste à remplir les lignes inconnues en y recopiant les lignes connues (la méthode est couramment appelée *Bob*). Bien entendu, elle provoque une pixelisation dans la direction y. Une des premières techniques développées consiste à utiliser la moyenne des pixels voisins pour interpoler le pixel manquant, c'est-à-dire :

$$\tilde{f}(i,j) = \frac{(f_{in}(i-1,j) + f_{in}(i+1,j)}{2} \tag{6.1}$$

Cette méthode ne permet pas de reconstruire les hautes fréquences (contours) de manière très nette. Aussi, des contours en escalier, du scintillement ainsi qu'un effet de flou peuvent apparaître. Pour améliorer ces techniques, l'idée a été d'interpoler en considérant la direction des contours avec la méthode E.L.A. (Edge Line Average)[Doy88]. Cette méthode détecte dans une fenêtre centrée sur le pixel à interpoler la meilleure direction Dir possible, puis effectue l'interpolation selon cette direction :

$$\tilde{f}(i,j) = \frac{f_{in}(i-1,j-Dir) + f_{in}(i+1,j+Dir)}{2} \tag{6.2}$$

Malgré une meilleure interpolation des contours, elle présente de nombreux défauts. En effet, la corrélation se fait au niveau local et reste très sensible aux bruits. La direction des contours est donc parfois erronée ce qui donne des artéfacts gênants. De nombreuses variantes de cette méthode [CWY00] permettent de résoudre ce problème sur une majorité des pixels de l'image : la corrélation est effectuée sur des groupes de pixels et non plus pixel à pixel (figure 6.2). Toutefois, ces méthodes sont dépendantes de leur taille de fenêtre qui limite l'angle de reconstruction des contours. En outre, plus la fenêtre est grande, plus le risque de mauvaise interpolation est élevé [YJ02]. Différentes métriques existent pour essayer d'agrandir la fenêtre et rajoutent des poids pour réduire le nombre de fausses directions [PTS98, BPK05]. Mais la complexité augmente pour calculer ces poids de manière efficace. D'autres difficultés subsistent telle la déconnexion des objets horizontaux fins.

La méthode proposée s'affranchit des méthodes actuelles dans la mesure où celle-ci n'est plus basée sur une recherche dans une fenêtre. Elle vient en complément d'une méthode classique de désentrelacement spatial qui pourra elle-même être intégrée à une solution adaptative (figure 6.3). Son terrain d'action est limité aux zones de l'image qui possèdent certaines caractéristiques posant des problèmes aux méthodes existantes.

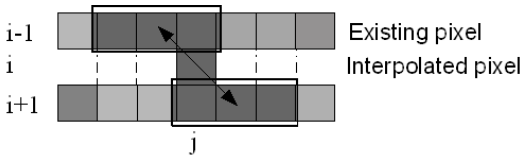


FIGURE 6.2 – Principe de l'interpolation ELA (Edge Line Average)

6.4 Détection des extrema

Dans cette partie, j'explique quel type de configuration locale dans l'image nous allons améliorer avec la solution proposée.

Les méthodes existantes ne sont pas capables de respecter la continuité des structures fines proches de l'horizontale. C'est d'autant plus gênant que les artéfacts visuels dus à ce problème sont souvent très visibles (déconnexion, interpolation erronée), comme le montre la figure 6.4.c.

En comparant le module de la transformée de Fourier d'une trame et d'une image entière, on peut observer dans le cas de la trame, d'une part le phénomène de repliement de spectre constituant l'aliasing, et d'autre part la perte des hautes fréquences horizontales. La difficulté consiste alors à localiser puis reconstituer la continuité de ces structures hautes fréquences qui ont été partiellement détruites et systématiquement déconnectées par le sous-échantillonnage horizontal (figure 6.4.b). Celles-ci correspondent à des minima ou maxima locaux de la fonction intensité, dans la direction verticale. La détection de ces extrema locaux est réalisée sur les lignes connues de l'image en comparant la valeur de chaque pixel $f_{in}(i,j)$ avec les valeurs des lignes inférieures et supérieures voisines $f_{in}(i-2,j)$ et $f_{in}(i+2,j)$. Soit $\mathcal{H}$ l'ensemble des pixels de type maxima et $\mathcal{L}$ l'ensemble des pixels de type minima :

$$\mathcal{H} = \{f_{in}(i,j) \,/ f_{in}(i,j) > \max(f_{in}(i-2,j), f_{in}(i+2,j)) + T\} \tag{6.3}$$

$$\mathcal{L} = \{f_{in}(i,j) \,/ f_{in}(i,j) < \min(f_{in}(i-2,j), f_{in}(i+2,j)) - T\} \tag{6.4}$$

T est une valeur de contraste minimum autorisée ($T = 16$ dans nos expérimentations).

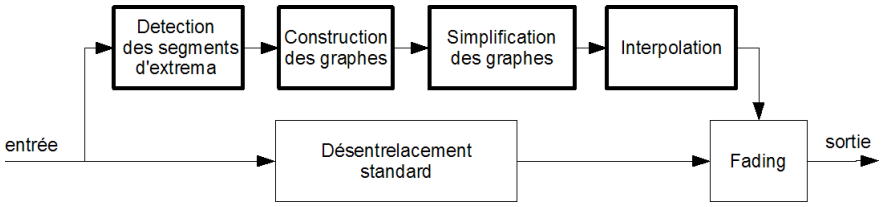


FIGURE 6.3 – Diagramme des traitements. En gras, les étapes de la méthode proposée

6.5 Segments et structure de données associée

Dans cette petite partie, j'introduis la notion de segment et une structure de donnée qui n'est pas la représentation matricielle classiquement utilisée dans le domaine.

Sur une même ligne, les extrema d'un même type peuvent former des composantes connexes (ou segments) au sens de la 2-connexité horizontale. A titre d'exemple, la figure 6.5 montre en noir quelques segments ainsi formés. Comme la suite de la méthode n'est pas fondée sur le parcours et le traitement

(a) Image originale (b) Trame (c) Interpolation ELA

FIGURE 6.4 – Les algorithmes qui utilisent une fenêtre de recherche ne peuvent pas reconstruire fidèlement les structures de type extrema

FIGURE 6.5 – Exemple de segments extraits du même type (en noir). Les lignes grises sont connues, les lignes blanches sont à interpoler

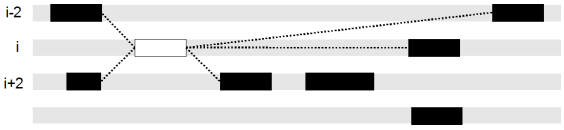


FIGURE 6.6 – Le segment marqué en blanc possède 5 voisins directs

de pixels mais sur le parcours et le traitement de segments, la structure bi-dimensionnelle classique de l'image n'est plus appropriée et est abandonnée au profit d'une structure de plus haut niveau : le segment. Chaque segment est une entité caractérisée par ses coordonnées (ligne, colonne de départ), sa longueur et son type (minimum ou maximum). La structure de données choisie est un compromis entre la taille mémoire requise et la complexité pour parcourir les ensembles $\mathcal{H}$ et $\mathcal{L}$. La solution retenue est un tableau de lignes (une entrée par ligne de trame), chaque ligne étant une liste chaînée des segments extraits sur cette ligne.

6.6 Construction de graphes connexes

Cette partie indique comment nous relions les segments entre eux pour reconstruire la continuité des structures.

Cette étape a deux buts : tout d'abord un but pratique pour remplir la structure présentée ci-dessus avec tous les segments, pour qu'elle puisse être facilement parcourue. Ensuite un but fonctionnel pour inter-connecter entre eux les segments de $\mathcal{H}$ (resp. de $\mathcal{L}$) pour constituer un ou plusieurs graphes connexes de maxima (resp. un ou plusieurs graphes connexes de minima) représentant les structures fines de l'image. Le parcours des éléments de $(\mathcal{H} \cup \mathcal{L})$ dans le tableau de listes chaînées se fait dans le sens du balayage vidéo.

Un segment S sur une ligne i possède au plus 6 voisins directs de même type : 3 du côté ouest et 3 du côté est, 2 sur chaque ligne $i-2$, i et $i+2$ (figure 6.6). La distance entre deux segments voisins de même type (c'est-à-dire soit de $\mathcal{H}$, soit de $\mathcal{L}$) S_1 et S_2 est la distance euclidienne $d(S_1, S_2)$, calculée entre les extrémités les plus proches de S_1 et S_2.

FIGURE 6.7 – Graphe connexe correspondant aux segments de la figure 6.5

FIGURE 6.8 – Les connexions restantes après la simplification du graphe de la figure

Pour un côté donné (ouest ou est), S est connecté à son voisin qui est à la plus faible distance. Si deux voisins sont les plus proches à la même distance, S est connecté aux deux (figure 6.7). Les connexions sont bi-directionnelles. Un seuil adaptatif est utilisé afin d'éviter des connexions peu fiables car trop longues. L_1 et L_2 étant les longueurs respectives de S_1 et S_2, la distance $d(S_1, S_2)$ doit vérifier la condition suivante pour que S_1 et S_2 soient connectés :

$$d(S_1, S_2) < \min(L_1, L_2) + \delta \tag{6.5}$$

Dans nos expérimentations δ est fixé à 2. Les connexions d'un segment avec ses voisins sont stockées dans la structure du segment lui-même (sous forme de pointeurs sur les segments voisins). Il faut noter qu'étant donné la nature même des extrema locaux, un segment ne peut avoir au plus que deux voisins pour un côté (est ou ouest) donné.

6.7 Simplification des graphes

Cette partie indique la manière de simplifier les graphes construits précédemment pour reconstituer le chemin qui guidera l'interpolation.

La connexion entre deux segments indique les deux morceaux de contour à relier dans l'image et donc l'interpolation à réaliser. Néanmoins, l'ensemble des graphes extraits ne peut pas être interpolé tel-quel. Certaines connexions doivent être supprimées (figure 6.8), ce qui provoque la division d'un graphe en plusieurs sous-graphes. Les graphes ne comportant qu'un segment ou qui sont sur une seule ligne sont aussi supprimés. La suppression des connexions doit d'une part privilégier des sous-graphes ayant chacun une direction prédominante et d'autre part supprimer les faux-positifs (connexions effectuées à tort).

Soient les directions NO, O, SO, NE, E, SE correspondant aux 6 connexions possibles pour un segment. Lors du parcours en profondeur du graphe, on appelle direction d'entrée celle par laquelle le segment est accédé. Cette direction est inexistante pour le segment de départ du parcours. Les directions de sortie correspondent à toutes les connexions du segment sauf la direction d'entrée. La simplification respecte les règles suivantes :

1. S'il existe 2 connexions de sortie du même côté, elles sont supprimées. Cette règle permet de ne pas relier à tort des structures potentiellement différentes.
2. Si une connexion de sortie est du même côté que la connexion d'entrée, elle est supprimée. Cette règle permet de ne conserver que des structures étirées dans une direction et non en zig-zag.

Les sous-graphes résultant du parcours et des règles de simplification sont des arbres à une seule branche. De longues structures rectilignes ou courbes peuvent être ainsi reconstituées.

6.8 Interpolation

Dans cette partie, nous voyons comment les pixels correspondant aux connexions des branches sont interpolés grâce à la connaissance de la structure à reconstituer.

L'interpolation est la dernière étape (figure 6.9). Elle est réalisée lors d'un parcours en avant (de l'ouest vers l'est) de chacune des branches. Soient S_1 et S_2 deux segments connectés de longueur L_1 et L_2, ayant comme coordonnées de départ (Y_{S_1}, X_{start_1}) et (Y_{S_2}, X_{start_2}). Les pixels à interpoler avec notre méthode correspondent au segment S_I dont les abcisses extrémités X_{start_I} et X_{end_I} sont interpolées linéairement des abcisses extrémités des segments S_1 et S_2 (figure 6.10).

$$X_{start_I} = X_{start_1} + \left| \frac{X_{start_2} - X_{start_1}}{2} \right| \tag{6.6}$$

$$X_{end_I} = X_{end_1} + \left| \frac{X_{end_2} - X_{end_1}}{2} \right| \tag{6.7}$$

$$\begin{aligned} \tilde{f}(i,j) &= \frac{1}{2} f_{in}\left(Y_{S_1}, X_{start_1} + E\left(\frac{j \times L_1}{L_I}\right)\right) \\ &+ \frac{1}{2} f_{in}\left(Y_{S_2}, X_{start_2} + E\left(\frac{j \times L_2}{L_I}\right)\right) \\ & j \in [X_{start_I}, X_{end_I}] \end{aligned} \tag{6.8}$$

La fonction E renvoie l'entier le plus proche de son argument. L_I est la taille du segment à interpoler $X_{end_I} - X_{start_I} + 1$. Y_{S_1} et Y_{S_2} représentent les ordonnées $i-1$ et $i+1$ (ou $i+1$ et $i-1$).

FIGURE 6.9 – Les segments en gris foncé représentent les pixels de maxima interpolés grâce aux segments connexes noirs

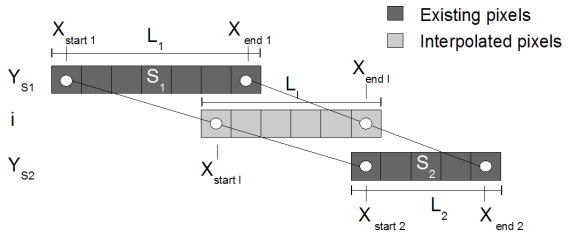


FIGURE 6.10 – Principe d'interpolation

6.9 Résultats

Les tests ont été effectués sur un ensemble assez large de séquences entrelacées d'origine, ou progressives ré-entrelacées. Une interpolation des extrema est réalisée avec notre méthode, comme expliqué dans le paragraphe précédent. La méthode E.L.A [PTS98] est utilisée pour le reste de l'image.

Les premiers résultats ont été obtenus sur des images fixes avec de nombreux détails horizontaux, telle que l'image du phare (figure 6.11) où la méthode reconstruit bien la continuité des structures (figure 6.12). Sur la séquence "tennis", l'amélioration est également très visible (figure 6.13). D'une manière générale, sur les séquences testées, l'interpolation fonctionne bien sur les lignes et les courbes détectées. Les structures quasi-horizontales qui ne sont pas reconstruites par les méthodes classiques sont ici presque identiques à l'original.

Le temps de traitement diffère très peu de celui de la méthode ELA. Nous avons analysé le nombre de segments et le nombre de pixels interpolés que notre méthode traite pour différentes séquences (tableau

Image / séquences	Dimensions	Nombre de pixels	Pixels extrema		Nombre de segments	Pixels interpolés	
			Nombre	% de l'image		Nombre	% de l'image
Phare	768×512	393216	15000	3.8	7800	9500	2.4
Car 2	576×720	414720	10000	2.4	3500	5000	1.2
Calendar	576×720	414720	18500	4.5	13000	13000	3.1
BBC	576×720	414720	9500	2.3	3500	7500	1.8
Tennis	480×720	345600	9000	2.6	5600	4000	1.2
American banner	480×720	345600	4000	1.2	1900	3600	1.0
Moyennes			11000	2.8	5883	7100	1.8

TABLE 6.1 – Tableau récapitulant la quantité de pixels et de segments traités par la méthode pour une image et plusieurs vidéos

6.1). En moyenne le pourcentage de pixels interpolés par notre méthode est de l'ordre de 2% de l'image entière. Du fait de ce faible pourcentage, la mesure du PSNR est peu significative. Le principal critère d'amélioration est donc subjectif : l'œil est très sensible à la continuité des structures rectilignes et à leur scintillement.

6.10 Conclusion

Notre méthode ne s'appuie pas sur le principe de méthodes existantes. Elle pallie ainsi les problèmes entraînés par ces dernières. La méthode s'attache à corriger les artéfacts les plus désagréables pour l'œil en détectant leur origine. Elle est fondée sur la continuité des objets afin de les reconstruire. De ce fait, les structures à fort contraste sont plus stables. Enfin, notre méthode peut se greffer à toutes les méthodes classiques pour améliorer leurs défauts sans un surcoût élevé. Il reste à trouver un réglage de seuil automatique pour la détection des extrema en fonction de la dynamique locale ou globale de l'image.

(a) Image originale

(b) Image désentrelacée

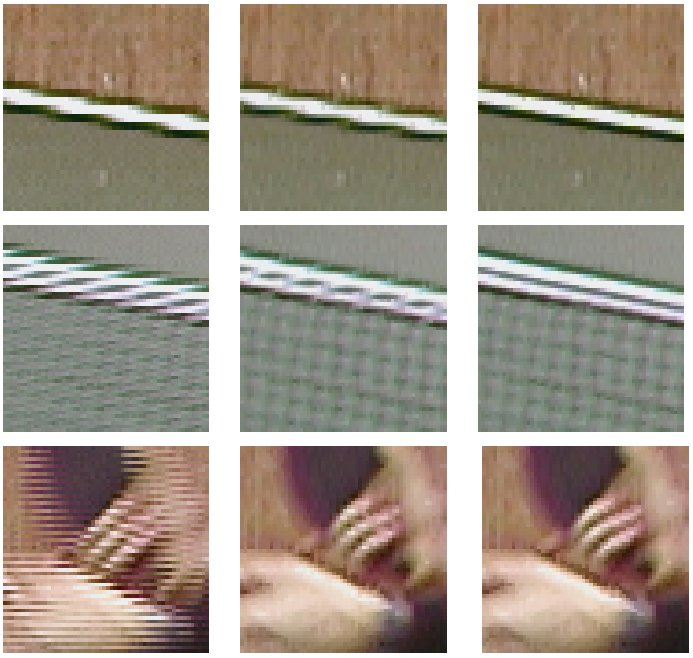
(c) Arbres de minima connectés

(d) Arbres de maxima connectés

FIGURE 6.11 – Arbres d'extrema construits et simplifiés

(g) Original (h) E.L.A. (i) Notre méthode

FIGURE 6.12 – Comparaison des résultats obtenus sur des zones de l'image Phare



(g) Original (h) E.L.A. (i) Notre méthode

FIGURE 6.13 – Comparaison des résultats obtenus sur des zones d'une image de la séquence *tennis table*

Chapitre 7

Agrandissement d'images

Sommaire

Ce travail est également le fruit d'une collaboration avec la société STMicroelectronics, lors de la thèse CIFRE d'Eric Van Reeth qui s'est terminée au printemps 2011. Le but de ce projet est de développer un algorithme d'agrandissement d'images qui surpasse les techniques actuelles. Dans les composants et set top box qu'elle conçoit et produit pour la télévision, STMicroelectronics a besoin de techniques qui permettent de mettre une image source de résolution réduite (SD par exemple) à des résolutions plus élevées (HD par exemple). La technique présentée ici obtient de très bons résultats, meilleurs que l'état de l'art à notre connaissance. Elle est illustrée avec des facteurs d'agrandissement entiers mais peut sans problème être utilisée pour tout facteur réel.

7.1 Introduction

L'agrandissement d'image est largement utilisé depuis une vingtaine d'années : zoom numérique, affichage d'images de définition inférieure à la définition de l'écran, ... Les techniques d'interpolation peuvent être divisées en deux catégories : adaptatives ou non. Les méthodes non adaptatives (bilinéaire, bicubique, spline, ...) sont généralement rapides et faciles à implémenter mais ne parviennent pas à donner des résultats satisfaisants pour tous types de contenu. Des artéfacts bien connus comme le *jagging*, le crènelage et le flou apparaissent alors en particulier au niveau des contours. Les méthodes adaptatives essaient de limiter l'apparition de ces artéfacts en adaptant leur traitement en fonction des propriétés des pixels.

La méthode adaptative présentée vise à ajuster l'interpolation en fonction de la direction du contour sur lequel se trouve le pixel à traiter. La direction des contours est en effet une information très utile puisqu'elle indique la direction le long de laquelle les variations des pixels sont douces (direction parallèle au contour), et à l'inverse la direction le long de laquelle les variations sont franches (direction perpendiculaire au contour). Adapter l'interpolation en fonction de ces caractéristiques permet de conserver au mieux l'aspect des contours de l'image basse-définition dans une grille plus définie. Nous présentons dans un premier temps notre technique de recherche de la direction des contours, puis dans un deuxième temps notre méthode d'interpolation basée sur les directions détectées.

7.2 Calcul de la carte directionnelle

La première étape consiste à isoler localement chaque contour à sa résolution optimale et à déterminer de façon très précise son orientation.

Notre approche s'inspire du travail de Peyré et al [PM05] pour la construction des bases de bandelettes. Dans cette méthode, l'image est partitionnée en blocs de taille variable selon un algorithme de *quad-tree*. L'objectif est d'isoler au plus une direction dans chaque bloc, pour définir l'orientation avec laquelle les bandelettes sont calculées. L'algorithme proposé par Peyré consiste à minimiser un critère de variation de manière itérative pour obtenir la partition optimale. Ce critère est d'abord calculé dans les plus petits blocs qui peuvent fusionner si leur regroupement entraîne la création d'un bloc parent qui fait diminuer la valeur du critère.
A l'inverse, notre méthode estime d'abord les variations dans les grands blocs. Ces derniers sont divisés uniquement dans le cas où ils contiennent plus d'une direction de contour. Cette approche présente trois avantages principaux :

- les variations des blocs de taille inférieure ne sont calculées que si nécessaire,
- le calcul de variation est plus robuste lorsqu'il est effectué dans un grand bloc,
- la précision sur l'angle estimé est meilleure car la résolution angulaire est plus élevée dans les grands blocs.

Les parties suivantes décrivent comment dans chaque bloc, la direction du contour est calculée. La première étape consiste à adapter la résolution du bloc à celle du contour. La deuxième explique comment le bloc est projeté sur des segments de droite 1D afin d'étudier les variations dans les différentes directions afin de déterminer la direction qui minimise la variation.

7.2.1 Adaptation de résolution

Le but de cette étude est d'adapter localement la résolution d'un bloc en fonction des caractéristiques fréquentielles du ou des contours contenus dans le bloc. Pour cela, une transformée en ondelettes isotrope non décimée (IUWT) est utilisée [SFM07]. Les propriétés isotropes de cette transformée sont nécessaires pour ne favoriser aucune direction lors de cette étape préalable à la détection de direction en elle-même. Le fait d'utiliser une transformée non-décimée permet de conserver des tailles de blocs similaires, et donc une résolution angulaire identique à travers les échelles. Notons enfin que seuls trois niveaux de résolutions sont utilisés. Nous considérons en effet que la plupart des contours présents dans les images naturelles peuvent être représentés de manière efficace dans les images de détails à l'une des trois premières échelles. Par la suite, l'image est découpée en blocs réguliers de taille (16×16) pixels et l'échelle la mieux adaptée est choisie pour chaque bloc. L'échelle optimale, J_{opt} est celle qui maximise la moyenne M_j de l'amplitude des coefficients d'ondelettes du bloc. C'est l'échelle pour laquelle la corrélation entre la fréquence de l'ondelette et la fréquence des contours est la plus élevée. Soient M_j^v et M_j^h respectivement les moyennes des amplitudes de chaque colonne et de chaque ligne du bloc Δ_j de taille $N \times N$ à l'échelle j.

$$M_j^v = \frac{\sum_{x=1}^{N} \left(\sup_{y=1\ldots N}(\Delta_j(x,y)) - \inf_{y=1\ldots N}(\Delta_j(x,y))\right)}{N}$$

$$M_j^h = \frac{\sum_{y=1}^{N} \left(\sup_{x=1\ldots N}(\Delta_j(x,y)) - \inf_{x=1\ldots N}(\Delta_j(x,y))\right)}{N}$$

$$M_j = \frac{M_j^v + M_j^h}{2}$$

$$J_{opt} = argmax_{j=1\ldots 3}[M_j]$$

Les figures 7.1(a)-(c) représentent les images de détails de la transformée IUWT d'une portion d'une

image naturelle dont la résolution décroit de gauche à droite. La figure 7.1(d) illustre l'image composée pour chaque bloc d'une des trois résolutions grâce au critère défini plus haut. Remarquons que les zones de contours hautes fréquences (foulard) sont représentées par la résolution la plus fine ($J_{opt} = 1$), alors que les zones de contours basses fréquences (visage) sont représentées par la résolution la plus basse ($J_{opt} = 3$). Notons enfin que si l'image composite présente des frontières de blocs évidentes, ceci n'est pas gênant par la suite car l'étude directionnelle est effectuée indépendamment dans chacun des blocs.

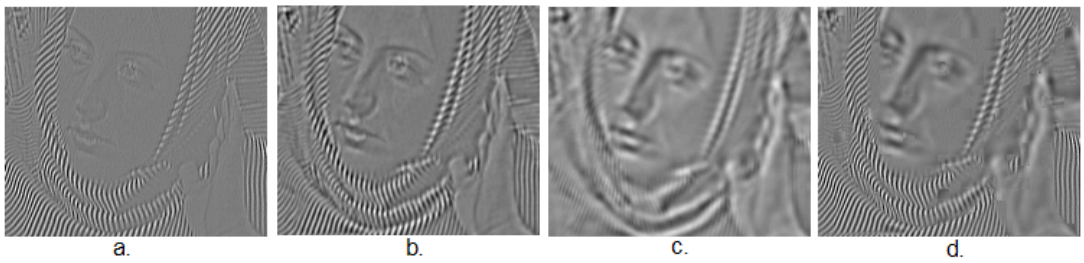


FIGURE 7.1 – (a)-(c) Images de détails de résolution décroissante. (d) Image composite.

7.2.2 Projection du bloc

Dans cette partie, on explique la méthode de projection d'un bloc 2D vers des segments 1D afin d'étudier les variations le long de différentes directions. Dans le domaine discret, une direction est représentée par des droites discrètes dont l'épaisseur (entre autres) est variable. Reveilles propose une description théorique de ces objets dans [Rev91]. Notre algorithme consiste à projeter tous les pixels du bloc le long de droites discrètes 8-connexes (ou naïves), comme illustré dans la figure 7.2. Le choix de cette méthode de projection a été évalué dans [VR11] : notre méthode d'estimation de direction de contours est comparée à deux méthodes existantes, la transformée de Radon et la projection utilisée pour la construction de bases de bandelettes. Elle s'avère plus précise lors de l'estimation de direction de contours dans de petits blocs (à partir de blocs (8×8) pixels et en-dessous), sur des contours bruités (bruit blanc et effet de blocs), et sur des images naturelles en général. Notons que le nombre de directions le long desquelles le bloc est projeté augmente avec la taille du bloc (16 pour un bloc 4×4, 72 pour un bloc 8×8 et 288 pour un bloc 16×16. En effet, plus le bloc est grand plus il est possible de définir des droites discrètes naïves distinctes, et plus la résolution angulaire est bonne. La relation exacte entre la taille du bloc et le nombre de directions que l'on peut créer dans ce bloc est définie mathématiquement par les suites de Farey.

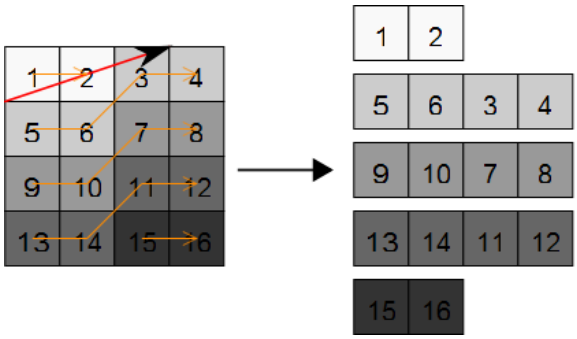


FIGURE 7.2 – Création de cinq segments 1D à partir d'un bloc (4×4) pixels, le long de la direction de paramètres (1,3).

7.2.3 Calcul des variations

Dans le but de trouver la direction prédominante de chaque bloc, les variations des segments projetés sont étudiées. La valeur de variation V_i du i-ème segment projeté s_i est définie par son amplitude :

$$V_i = \sup(s_i) - \inf(s_i)$$

La variation globale V_{tot} de la direction θ est calculée en moyennant la variation de ses segments, avec I le nombre total de segments projetés :

$$V_{tot} = \frac{\sum_{i=1}^{I} V_i}{I}$$

L'angle associé θ_{bloc} est celui qui correspond à la valeur de variation minimale, et que l'on estime parallèle au contour du bloc. La figure 7.3 illustre un cas où le bloc contient une direction de contour.

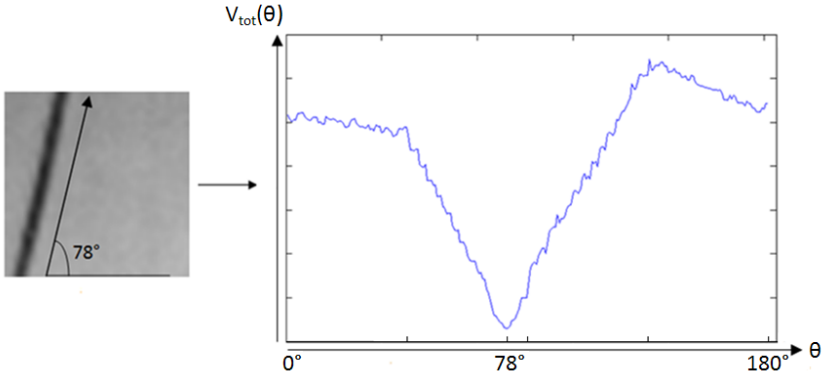
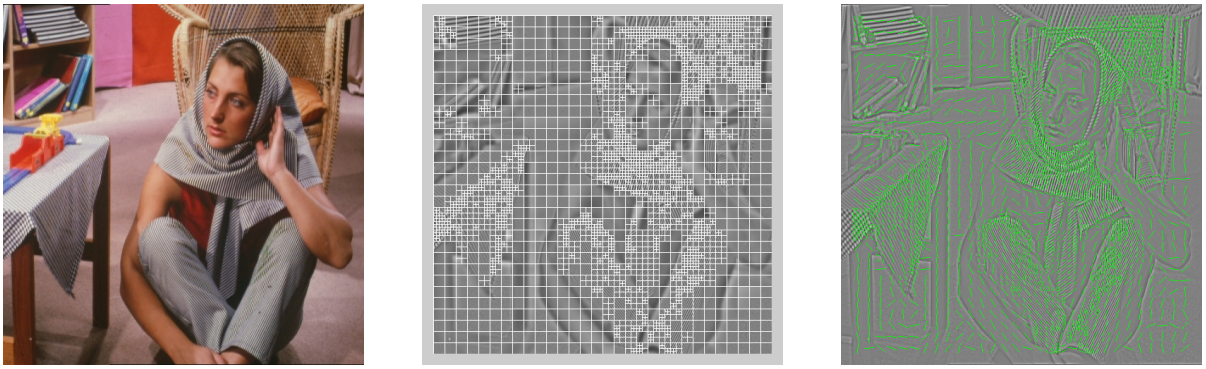


FIGURE 7.3 – Un bloc et sa courbe de variations. La valeur minimale correspond à un angle de 78°.

Dans le cas ou plusieurs directions de contours sont présentes à l'intérieur du même bloc, ce dernier est divisé grâce à l'algorithme de *quad-tree*. Le critère qui détermine la présence d'une ou plusieurs directions dans le bloc est basé sur la comparaison entre les variations des coefficients le long de la direction θ_{bloc}, et la variation totale des coefficients du bloc. Lorsqu'une division du bloc est nécessaire, quatre sous-blocs de tailles égales sont créés. Le calcul de direction prédominante est ensuite effectué à l'intérieur de chaque sous-bloc. Cet algorithme est itéré jusqu'à ce que tous les blocs contiennent au plus une direction de contours ou lorsqu'une taille minimale de bloc est atteinte (4×4 pixels). Un exemple est illustré en figure 7.4.



(a) Image originale (b) Découpage *quadtree* (c) Direction de chaque bloc

FIGURE 7.4 – Exemple de division par *quad-tree* et de calcul des directions

7.3 Interpolation

La seconde étape consiste à corriger une interpolation de base (spline cubique) avec notre interpolation uniquement sur les pixels de contours dans la direction estimée lors de la phase précédente.

Un certain nombre d'interpolations directionnelles [LZT01][JM02][Mur05] présentent des artéfacts rédhibitoires car elles interpolent l'image entière en se basant uniquement sur la direction locale qu'elles ont préalablement détectée. Pour éviter de telles dégradations, notre méthode combine une interpolation isotrope (spline cubique) et une interpolation directionnelle. Cette dernière n'est utilisée que sur les contours dont l'orientation a été détectée lors de la phase précédente. Notons que des approches hybrides ont également été récemment proposées par Wang [WW07], Mallat [MY09] et Mueller [MLD07]. Dans notre approche, la sélection des pixels devant être interpolés directionnellement est obtenue avec des filtres de Gabor. La sortie de ce filtrage est utilisée comme un masque et pondère l'interpolation isotrope spline cubique et l'interpolation directionnelle comme expliqué ci-dessous.

7.3.1 Filtrage de Gabor

Les filtres de Gabor sont des filtres orientés passe-bande. Les paramètres d'un filtre sont l'échelle de l'enveloppe Gaussienne (σ_a, σ_b), l'angle du filtre, la fréquence et la phase de la sinusoïde (ω_0, P_0). La figure 7.5 montre un exemple de filtre dans le domaine spatial. Afin de localiser les pixels qui doivent

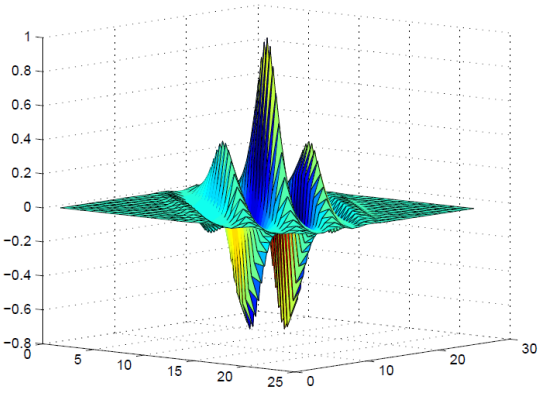


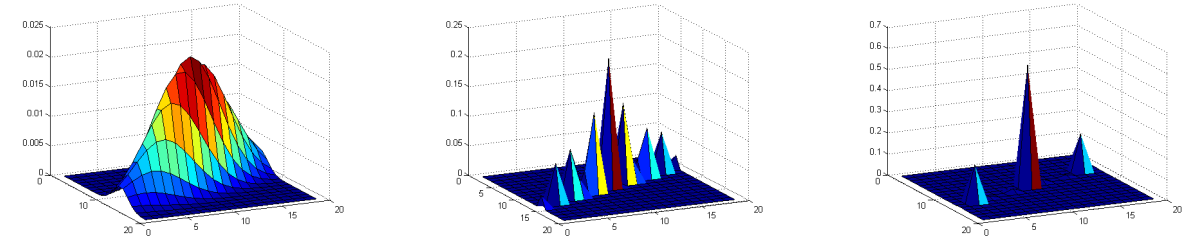
FIGURE 7.5 – Exemple de filtre de Gabor

être interpolés directionnellement, les filtres de Gabor sont appliqués sur l'image d'ondelettes de détails à la résolution la plus fine, pour chaque bloc résultant du *quad-tree*. Les filtres sont orientés dans la direction attribuée au bloc, et les paramètres de l'enveloppe ainsi que la fréquence de la sinusoïde sont fixés empiriquement de sorte que la bande passante du filtre soit adaptée à la fréquence des coefficients d'ondelettes. Le fait d'appliquer ces filtres sur les coefficients d'ondelettes permet de filtrer des images dont le contenu fréquentiel varie peu, et donc de fixer de manière optimale les coefficients des filtres de Gabor sans que ceux-ci ne doivent être modifiés en fonction de l'image. La sortie de ce filtrage est un masque appelé M qui a une valeur élevée pour les pixels ayant la même orientation que le filtre, et faible pour les autres.

7.3.2 Lissage Gaussien directionnel

L'interpolation directionnelle consiste à corriger l'interpolation spline du bloc en la filtrant avec un filtre Gaussien 2D orienté dans la direction θ_{bloc} du contour du bloc, pour donner un bloc interpolé directionnellement B_{dir}. Les paramètres de la Gaussienne (σ_θ et $\sigma_{\theta^\perp}$) sont choisis tels que la Gaussienne

soit allongée selon θ et très fine dans la direction perpendiculaire $\theta^{\perp}$: $\sigma_{\theta} >> \sigma_{\theta^{\perp}}$. Des exemples de filtres sont donnés en figure 7.6 pour plusieurs valeurs de $\sigma_{\theta^{\perp}}$. Ces paramètres permettent de créer des filtres dits *fins*, dont le nombre de coefficients non nuls est faible. Seuls les coefficients alignés dans la direction choisie sont non-nuls et permettent la reconstruction des contours orientés dans la même direction que le filtre, et l'élimination des artéfacts. De plus, la finesse de ces filtres permet de ne pas introduire de flou lorsqu'ils sont appliqués.



(a) (b) (c)

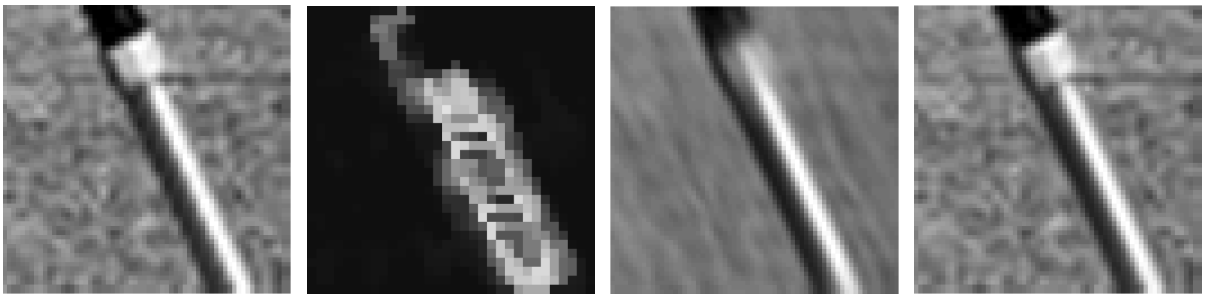
FIGURE 7.6 – Des filtres Gaussiens pour différentes valeurs du paramètre $\sigma_{\theta^{\perp}}$. (a) $\sigma_{\theta^{\perp}} = 1.5$. (b) $\sigma_{\theta^{\perp}} = 0.15$. (c) $\sigma_{\theta^{\perp}} = 0.05$.

7.3.3 Combinaison des interpolations isotrope et directionnelle

Afin de ne pas dégrader les pixels qui ne seraient pas orientés dans la direction θ_{bloc}, seuls les pixels pour lesquels la sortie du filtre de Gabor est élevée sont lissés. Le schéma d'interpolation global est alors une combinaison linéaire du bloc interpolé par un noyau spline B_{spline} et du bloc filtré B_{dir}, pondérée par le masque M interpolé (par un noyau spline) et normalisé entre 0 et 1. Ainsi pour chaque pixel de coordonnées (x, y) :

$$B(x, y) = B_{dir}(x, y) \times M(x, y) + B_{spline}(x, y) \times (1 - M(x, y))$$

La figure 7.7 illustre le fonctionnement global de l'interpolation. Notons qu'afin d'éviter l'apparition d'un effet de bloc, un recouvrement est introduit lors des différents filtrages.



(a) (b) (c) (d)

FIGURE 7.7 – (a) Bloc interpolé par un noyau spline. (b) Masque créé par le filtrage de Gabor. (c) Bloc (a) filtré par le filtre Gaussien. (d) Résultat final de la pondération.

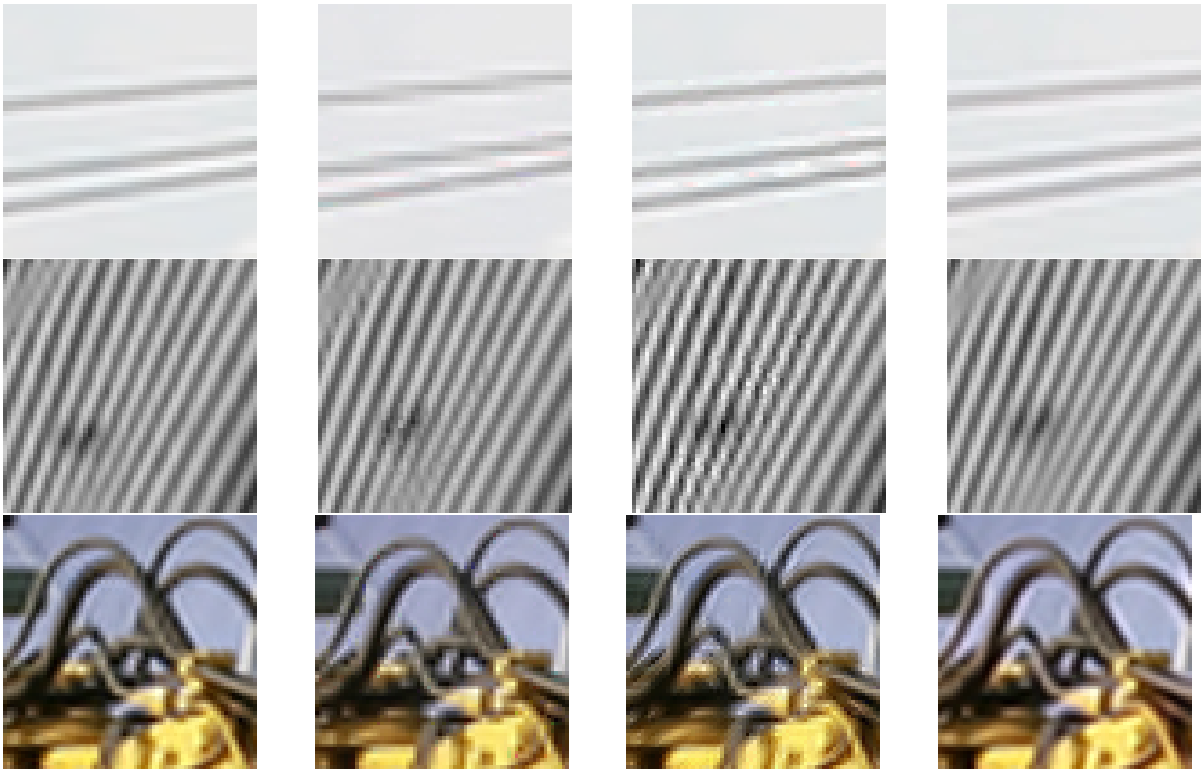
7.4 Résultats et conclusion

La figure 7.8 montre une série d'agrandissements d'un facteur 2×2 avec quatre méthodes d'interpolation directionnelle récentes : la méthode SME de Mallat et Yu [MY09], celle de Wang et al (NOAI) [WW07], l'interpolation de NEDI de Li [LZT01], et la nôtre (GCI, pour *Gaussian Corrected Interpolation*).

Notre méthode parvient à éliminer les artéfacts produits par l'interpolation spline (*jaggy* et *aliasing*) sans en introduire de nouveaux (faux pixels, faux contours). Le comportement est également correct quand de nombreuses directions de contours sont présentes et lors de contours courbes. Des résultats quantitatifs (PSNR) ont été calculés [VR11] et montrent un léger gain de $0.3dB$ en moyenne par rapport à l'interpolation NEDI, sur neuf images traitées. Une évaluation subjective des résultats permettrait de mieux conclure.

Au niveau des temps de calcul, notre méthode est moins rapide que l'interpolation NEDI mais environ dix fois plus rapide que la méthode SME pour une image 256×256, et vingt fois pour une image 512×512 (interpolation d'un facteur 2×2). De plus, plusieurs compromis peuvent être réalisés pour améliorer la vitesse de notre algorithme. Le nombre d'angles détectables peut être réduit, et la définition en prétraitement d'un dictionnaire de filtres de Gabor et Gaussien améliorerait considérablement les temps de calcul sans que la qualité de l'interpolation ne soit altérée de manière significative.

Enfin, notons que les paramètres de nos filtres sont fixés automatiquement pour permettre un comportement optimal de l'algorithme, sans que l'utilisateur n'intervienne. Cela permet une grande souplesse d'utilisation pour tous types d'images.



(a) SME (b) NOAI (c) NEDI (d) GCI

FIGURE 7.8 – Comparaison d'interpolations 2×2 (voir la version électronique pour une meilleure visualisation)

Chapitre 8

Amélioration d'artefacts sur panneaux LCD

Sommaire

Cette recherche concerne notre collaboration avec le projet IQI (Image Quality Improvement) de la division Home Video de ST Microelectronics. Pierre Adam y a effectué sa thèse CIFRE. Le but de son travail est de développer des techniques pour améliorer la qualité de l'image LCD et la perception des mouvements.

8.1 Contexte industriel

Les moniteurs à matrices de cristaux liquides (LCD) remplacent massivement les moniteurs et les télévisions CRT. Les écrans LCD-TFT ont l'avantage du volume, du poids, de la faible consommation et d'une résolution sans cesse croissante. En revanche, ils sont sujet à un phénomène important de traînées (rémanence) lors de mouvement dans l'image. Cet artefact, très visible sous la forme de traînées résulte de la façon dont les écrans sont pilotés (hold type ou maintien, en opposition à l'affichage impultionnel du CRT) et au temps de réponse plus ou moins important des cellules de cristaux liquide. La maîtrise de la qualité d'image des TVs à écrans plats étant un facteur stratégique déterminant pour prendre des parts sur un marché annuel de 1400M$ à 1600M$ (sur une base de 20$ par panneau), les industriels du LCD cherchent, depuis plusieurs années, à réduire le temps de réponse pour remporter la faveur des consommateurs. De nombreuses solutions ont été proposées, comme l'insertion de noir (black insertion) [NSS+01], le blinking backlight [FT+01], le doublement de fréquence (double frame-rate) [IM04], le motion compensated inverse filtering [KJ04] et la technique très largement répandue, introduite en 1992, de l'*overdrive* [O+92].

8.2 L'overdrive

Cette partie explique le principe de l'*overdrive* utilisé pour réduire le temps de réponse des cellules LCD.

La méthode utilisée pour réduire le temps de réponse des écrans LCD actuellement commercialisés est appelée *overdrive*. L'*overdrive* désigne à la fois la technique et le circuit électronique qui la met en œuvre. Ce circuit est installé en amont des drivers qui exécutent la dernière étape de la chaîne : le pilotage de l'affichage effectif de la matrice des cellules LCD. Le principe du circuit est simple et s'applique à chaque pixel de l'image : au lieu de passer du niveau de gris n_1 à t au niveau de gris n_2 à $t+1$, on modifie la forme de la transition grâce au passage de n_1 à $n_2+\epsilon$, avec $\epsilon > 0$ si $n_1 < n_2$ et $\epsilon < 0$ si $n_1 > n_2$. Ainsi, lors du rafraîchissement de l'écran, la valeur affichée sera véritablement n_2, comme le montre la figure 8.1. On remarque que la technique est inopérante pour des valeurs n_2 proches de 255 (resp. de O) car le bus de données 8 bits des drivers ne peut pas porter de valeurs supérieures à 255 (resp. inférieures à 0).

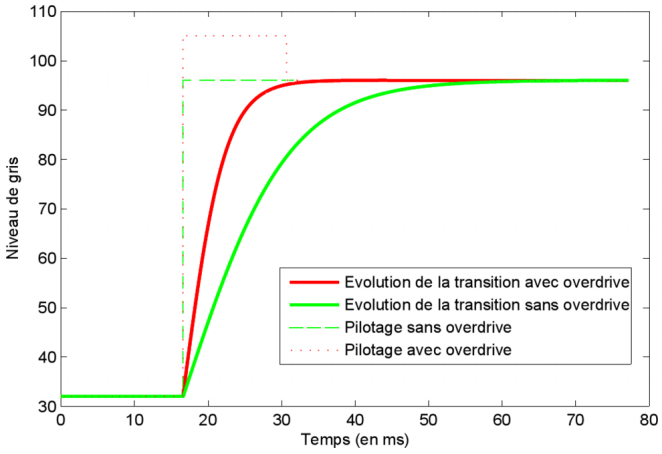


FIGURE 8.1 – Temps de réponse avec et sans *overdrive*

L'implémentation de l'*overdrive* nécessite deux mémoires, une RAM et une ROM. Le rôle de la RAM consiste à stocker l'image précédente pour pouvoir la comparer, pixel à pixel, à l'image courante. La ROM quant-à-elle contient la table de correspondance (LUT) où sont enregistrées dans une matrice bi-dimensionnelle les valeurs de transition $n_2+\epsilon$ pour chaque couple n_1 (valeur précédente) et n_2 (valeur courante). Une seule LUT est implantée et utilisée pour les trois composantes R, G et B.

L'utilisation d'une LUT comporte deux désavantages : le coût de la ROM et son aspect figé qui ne permet pas une correction optimale, chaque exemplaire d'écran LCD ayant une réactivité propre. En revanche, connaître certaines données propres à un écran LCD, comme par exemple la forme et la durée des transitions d'un niveau de gris à un autre permettrait d'envisager pour un panneau LCD donné, une nette amélioration de sa qualité d'affichage.

Mesurer ces transitions par un dispositif électronique automatique est envisageable, mais procéder à toutes les mesures de transitions de niveaux de gris nécessite beaucoup trop de temps et de ressources. En effet, si on suppose que l'on calcule le temps de réponse de toutes les transitions sur 8 bits (256 niveaux de gris), avec un temps moyen de 100 ms par transition (temps comprenant la durée de la plus longue transition et le calcul du temps de réponse), il faudrait $(255^2 - 255) \times 100$ ms, c'est-à-dire environ 1h50 de calcul par panneau, temps beaucoup trop important pour une application industrielle.

nous présentons dans les sections suivantes plusieurs modélisations génériques des transitions des cellules LCD (temps de réponse et forme de la transition) [ABCL06, ABL06, ABL07b]. Pour un écran particulier, les paramètres des modèles sont obtenus à l'aide d'une phase d'étalonnage qui ne nécessite qu'un très petit nombre de mesures, donc potentiellement industrialisable. Grâce au modèle, la LUT (et donc la ROM) est remplacée par un calcul de ϵ effectué au moment de la correction *overdrive*, dans une unité arithmétique.

8.3 Formalisme pour les transitions montantes

Cette partie pose les bases du vocabulaire utilisé par la suite, en se focalisant uniquement sur la définition des ensembles de transitions.

Soit $\mathbb{L}_n = \{L_i | 0 \leqslant i < n, L_i = i\}$ un ensemble fini de n niveaux des gris initiaux et finals des transitions. Sur 8 bits, on a $\mathbb{L}_{256} = [0..255]$.

A partir de $\mathbb{L}_n$, on crée un nouvel ensemble représentant les transitions de niveaux de gris de $\mathbb{L}_n$ vers $\mathbb{L}_n$; il s'agit donc d'un ensemble de couples d'entiers. Ce nouvel ensemble, noté $\mathbb{T}$, est défini par :

$$\begin{aligned} \mathbb{T} &= \{(x,y) | x \in \mathbb{L}_n, y \in \mathbb{L}_n, x \neq y\} \\ Card(\mathbb{T}) &= n^2 - n \end{aligned}$$

Les valeurs x et y correspondent respectivement à la valeur initiale et la valeur finale de la transition à modéliser. De ce fait, l'ordre des éléments du couple est primordial.

On définit deux sous-ensemble de $\mathbb{T}$, notés ${}^{\downarrow}\mathbb{T}$ et ${}^{\uparrow}\mathbb{T}$, respectivement ensemble des transitions descendantes (d'un niveau de gris à un niveau inférieur) et montantes (d'un niveau de gris à un niveau supérieur), par :

$$\begin{aligned} {}^{\downarrow}\mathbb{T} &= \{(x,y) | (x,y) \in \mathbb{T}, x > y\} \\ {}^{\uparrow}\mathbb{T} &= \{(x,y) | (x,y) \in \mathbb{T}, x < y\} \\ {}^{\downarrow}\mathbb{T} \cup {}^{\uparrow}\mathbb{T} &= \mathbb{T} \\ {}^{\downarrow}\mathbb{T} \cap {}^{\uparrow}\mathbb{T} &= \emptyset \\ Card({}^{\downarrow}\mathbb{T}) = Card({}^{\uparrow}\mathbb{T}) &= \frac{n^2 - n}{2} \end{aligned}$$

Actuellement, seules les transitions montantes, c'est-à-dire les éléments de ${}^{\uparrow}\mathbb{T}$, sont étudiées : le comportement des temps de réponse de ${}^{\uparrow}\mathbb{T}$ et de ${}^{\downarrow}\mathbb{T}$ sont très différents comme le montre la figure 8.4. La figure montre également que les temps de transitions de ${}^{\uparrow}\mathbb{T}$ sont plus préoccupants que ceux de ${}^{\downarrow}\mathbb{T}$.

${}^{\uparrow}\mathbb{T}$ est séparé en deux sous-ensembles : le sous-ensemble des transitions montantes dont la valeur initiale est fixée à X, notée ${}^{\uparrow}\mathbb{T}_X$ et le sous-ensemble des transitions montantes dont la valeur finale est fixée à Y, notée ${}^{\uparrow}\mathbb{T}^Y$. On obtient ainsi :

$$\begin{aligned} {}^{\uparrow}\mathbb{T}_X &= \{(x,y) | (x,y) \in {}^{\uparrow}\mathbb{T}, x = X\} \\ {}^{\uparrow}\mathbb{T}^Y &= \{(x,y) | (x,y) \in {}^{\uparrow}\mathbb{T}, y = Y\} \\ Card({}^{\uparrow}\mathbb{T}_X) &= (n-1) - X \\ Card({}^{\uparrow}\mathbb{T}^Y) &= Y \end{aligned}$$

Enfin, on définit la transition de la valeur initiale X à la valeur finale Y par T_X^Y.

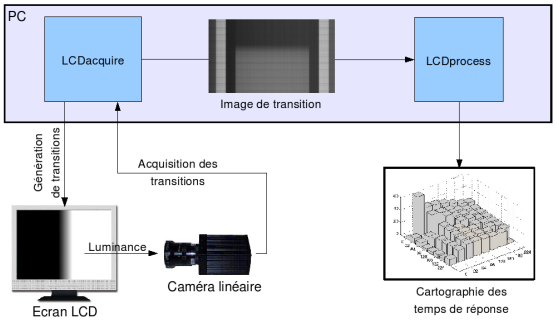


FIGURE 8.2 – Système d'acquisition de transitions et de cartographie des temps de réponse

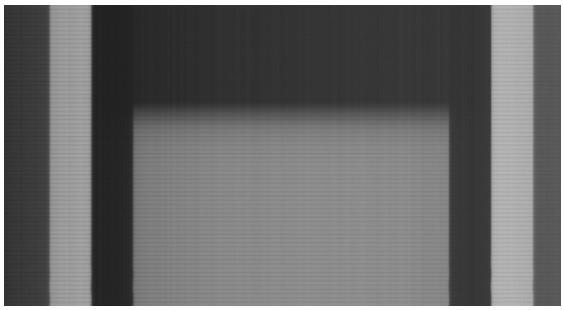


FIGURE 8.3 – Image de la transition T_{50}^{140} résultant de la concaténation de 512 lignes distantes de 0,4ms (2,5KHz) capturées par la caméra linéaire du système de la figure 8.2

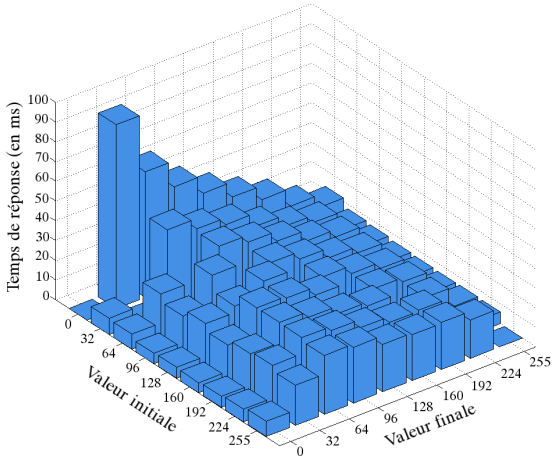


FIGURE 8.4 – Cartographie des temps de réponse mesurés

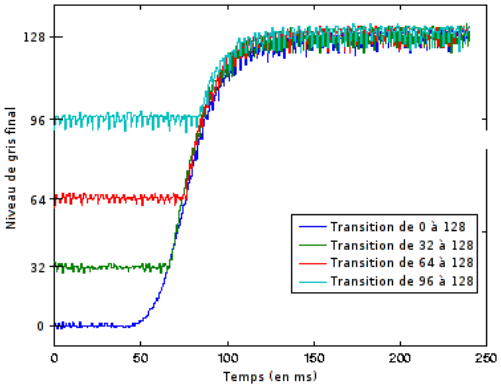


FIGURE 8.5 – Superposition de 4 transitions de ${}^{\uparrow}\mathbb{T}^{128}$

8.4 Généricité de T_0^Y

Cette partie montre comment on peut déduire toutes les transitions de ${}^{\uparrow}\mathbb{T}^{Y}$ à partir de la transition T_0^Y.

Un dispositif matériel et logiciel spécifique (figure 8.2) a été développé [ABL07a] pour la génération, la capture synchronisée de transitions avec une caméra linéaire (figure 8.3) et la mesure des temps de réponse réels sur un écran de type Twisted Nematic (TN) * affichant toute une série de transitions entre différents niveaux de gris. Ces mesures fournissent une cartographie des temps de réponse des transitions de ${}^{\uparrow}\mathbb{T}$. La figure 8.4 donne un échantillon de ces mesures.

Dans la figure 8.5 on a superposé à titre d'exemple quelques transitions de ${}^{\uparrow}\mathbb{T}^{128}$: T_0^{128}, T_{32}^{128}, T_{64}^{128} et T_{96}^{128}. On remarque que les transitions de ${}^{\uparrow}\mathbb{T}^{128}$ peuvent se déduire de T_0^{128}. On observe le même phénomène pour toutes les transitions. Ainsi, les transitions de ${}^{\uparrow}\mathbb{T}^{Y}$ sont liées et peuvent toutes être déduites de la transition T_0^Y.

Les sections suivantes proposent plusieurs modèles de temps de réponse pour les transitions de type ${}^{\uparrow}\mathbb{T}_0$, puis indiquent comment obtenir le temps de réponse et la courbe associée à chaque transition.

∗. il existe principalement trois technologies LCD : TN (la plus répandue), IPS et VA

8.5 Modélisations du temps de réponse

Cette partie expose plusieurs modèles mathématiques simples, qui, initialisés à l'aide de quelques mesures, permettent de calculer un temps de réponse quel que soit le niveau final de la transition.

Pour connaître les temps de réponse de ${}^{\uparrow}\mathbb{T}$, il faut tout d'abord modéliser ceux de ${}^{\uparrow}\mathbb{T}_0$. Pour cela, en observant sur la figure 8.4 l'évolution des temps de réponse de ${}^{\uparrow}\mathbb{T}_0$ en fonction de la valeur finale, on peut affirmer que plus la valeur finale augmente, plus le temps de réponse est faible. Cela montre si nécessaire que la norme ISO 13406-2 n'est pas adaptée à la mesure de réactivité de l'écran : avec cette norme, la mesure des temps de montée et de descente ne se fait pas sur la totalité de l'amplitude du signal, mais sur 80 %. Les 10 % à chaque extrême sont tronqués. Cette mesure simplifie, flatte et fausse la réalité : si certains systèmes mettent plus de temps que d'autres à décoller ou à se stabiliser, la mesure avec cette norme ne le montre pas.

Nous allons définir dans cette partie différents modèles de temps de réponse de ${}^{\uparrow}\mathbb{T}_0$ et les équations associées à chacun de ces modèles ; les fonctions présentées donnent donc les temps de réponse (en millisecondes) en fonction de la valeur finale de la transition (notée L_f, niveau de gris compris entre 1 et 255 pour un écran dont les couleurs sont chacune codée sur 8 bits). La figure 8.6 montre en gris foncé les temps de réponse de ${}^{\uparrow}\mathbb{T}_0$ à modéliser parmi toute la cartographie des temps de réponse.

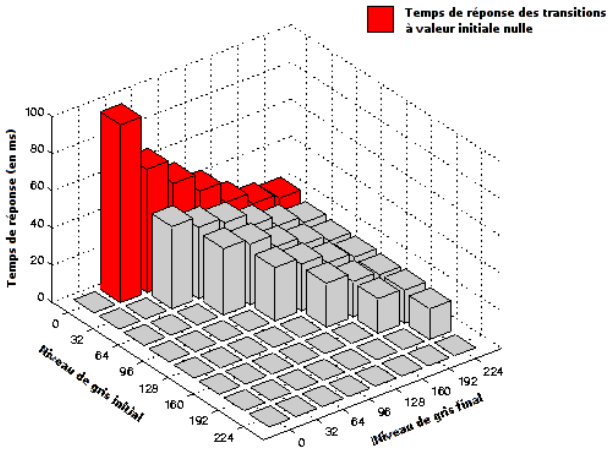


FIGURE 8.6 – Temps de réponse des transitions ${}^{\uparrow}\mathbb{T}_0$ à modéliser (en gris foncé)

8.5.1 Le modèle polynomial

Le modèle présenté est un modèle polynomial, choisi pour sa simplicité et son efficacité. Il ne se base pas sur la forme générale de la courbe ; on ne tente pas ici de retrouver un comportement physique mais juste d'approcher une courbe avec le moins d'erreur possible. Un modèle polynomial de degré d implique la détermination de $d+1$ paramètres à l'aide d'au moins $d+1$ mesures, un nombre de mesures plus important permettant une meilleure approximation finale. Un compromis acceptable consiste à choisir $d = 3$. Le modèle proposé doit fournir pour les transitions T_0^L un temps de réponse τ :

$$\tau = f(L) = a_3.L^3 + a_2.L^2 + a_1.L + a_0 \tag{8.1}$$

On considère que n mesures ont été sélectionnées pour la détermination de la famille des $(a_i)_{0 \leqslant i \leqslant 3}$. Soient $(\tau_j, L_j)_{1 \leqslant j \leqslant n}$, n couples de temps de réponse et de valeurs finales, effectuées expérimentalement.

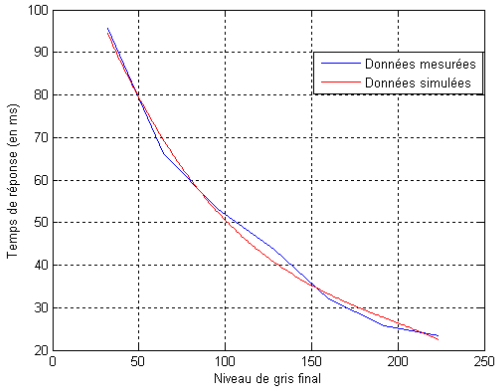


FIGURE 8.7 – Modèle polynomial des temps de réponse

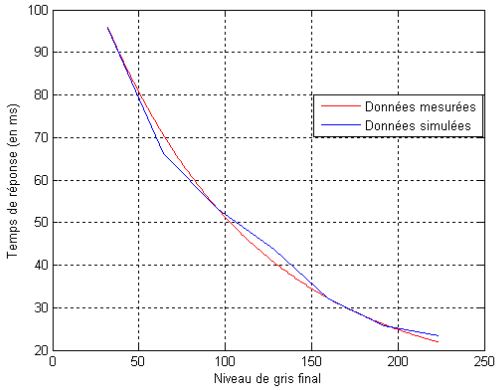


FIGURE 8.8 – Modèle exponentiel des temps de réponse

Ces couples vérifient donc chacun l'équation (8.1) ; on obtient donc le système de n équations polynomiales à 4 inconnues suivant :

$$\begin{cases} \tau_1 = f(L_1) = a_3.{L_1}^3 + a_2.{L_1}^2 + a_1.L_1 + a_0 \\ \tau_2 = f(L_2) = a_3.{L_2}^3 + a_2.{L_2}^2 + a_1.L_2 + a_0 \\ \vdots \qquad \vdots \qquad \vdots \\ \tau_n = f(L_n) = a_3.{L_n}^3 + a_2.{L_n}^2 + a_1.L_n + a_0 \end{cases}$$

Pour résoudre ce système, il faut déterminer la valeur optimale de chacun des $(a_i)_{0 \leqslant i \leqslant 3}$; pour cela, une régression polynomiale de degré 4 est effectuée.

Résultats : La détermination des paramètres pour ce modèle polynomial a l'avantage d'être simple : la résolution d'une équation matricielle permet en effet d'obtenir directement les paramètres optimaux. Le choix et le nombre des mesures nécessaires au bon paramétrage sont des données déterminantes à la bonne minimisation des erreurs ; choisir le plus grand nombre de mesures réparties sur l'ensemble des niveaux de gris donne la meilleure solution.

Par rapport aux mesures effectuées avec la caméra linéaire, les erreurs de simulation se situent autour de 6% ; de plus, cette faible valeur prend en compte le bruit analogique des mesures enregistrées dû à la capture par la caméra linéaire. La figure 8.7 montre le modèle polynomial, paramètré à l'aide de quatre mesures (32, 96, 160 et 224) sur 8 bits.

8.5.2 Les autres modèles

D'après la forme globale de la courbe des temps de réponse de ${}^{\uparrow}\mathbb{T}_0$, on peut émettre l'hypothèse que la décroissance est de type exponentielle. Deux modèles ont été proposés pour les temps de réponse des transitions de ${}^{\uparrow}\mathbb{T}_0$:

$$\tau = \frac{K_1}{(L + K_2)^{K_3}}$$

$$\tau = K_1 + K_2.e^{-K_3.L}$$

avec K_1, K_2 et K_3, trois constantes à déterminer, τ le temps de réponse et L le niveau de gris final. Les modèles ont donc besoin au minimum de 3 mesures pour être correctement paramétrés. Malheureusement,

il n'est pas possible de déterminer de façon formelle les paramètres K_1, K_2 et K_3 pour les deux modèles présentés ci-dessus et seule une résolution numérique est envisageable. Cela nécessite donc un traitement de type calcul numérique dont le coût n'est pas justifiable par rapport au modèle polynomial.

Résultats La figure 8.8 montre les résultats du modèle exponentiel, calculés numériquement, comparés aux données expérimentales.

L'erreur de simulation est ici inférieure à 6%, résultat légèrement meilleur que celui du modèle polynomial. Cependant, plusieurs problèmes se posent comme la résolution numérique, le nombre et le choix des données à utiliser pour obtenir un bon paramètrage. En effet, en choisissant pour ce modèle trois mesures de façon équirépartie sur 8 bits, l'erreur estimée de la simulation passe de 6 à 10%, valeur bien supérieure au modèle précédent. Par conséquent, au vu de ces difficultés de résolution, le choix va se porter par la suite sur le modèle polynomiale en sélectionnant les mesures de paramètrage de façon équirépartie.

8.6 Modélisation des transitions

On se place toujours dans le domaine ${}^{\uparrow}\mathbb{T}_0$, c'est-à-dire dans le domaine où les transitions ont une valeur initiale nulle. Cette partie présente un modèle du comportement des transitions, représentant l'évolution temporelle du niveau de gris.

8.6.1 Modèle en tangente hyperbolique

D'après la démarche méthodologique présentée dans la section 8.4, une fois le modèle des temps de réponse posé, il faut à présent définir la modélisation des transitions de niveaux de gris. Ce modèle, associé à une série d'équations mathématiques, a pour but de se rapprocher le plus possible du comportement des cellules LCD. La figure 8.9 montre un exemple réel d'évolution des niveaux de gris lors de transitions montantes.

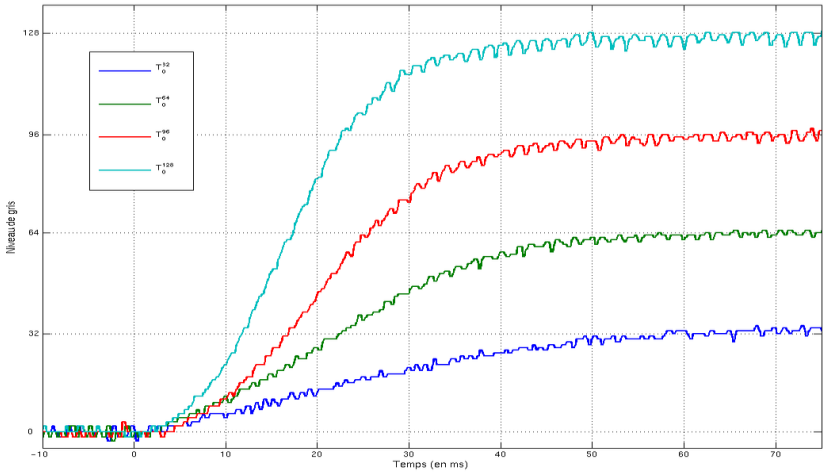


FIGURE 8.9 – Évolution du niveau de gris lors de différentes transitions de ${}^{\uparrow}\mathbb{T}_0$

Au vu de la forme générale des courbes, un modèle en tangente hyperbolique a été proposé. Pour rappel, la fonction tangente hyperbolique est définie par :

$$tanh(x) = \frac{sinh(x)}{cosh(x)} = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

Pour décrire par la suite le comportement des transitions selon sa valeur initiale ou finale, on procède à un découpage en trois étapes de la fonction en tangente hyperbolique :

- Etape dite d'accélération : Le coefficient directeur de la tangente en un point de la courbe est de plus en plus important à mesure que l'on avance dans le temps ; il part de 0 (courbe horizontale) pour arriver à la valeur seuil s.

- Etape dite à vitesse quasi-constante : On peut modéliser cette partie par une droite ; le coefficient directeur s reste quasiment constant.

- Etape dite de décélération : Le coefficient directeur de la tangente diminue de plus en plus au fur et à mesure que le temps s'écoule ; il passe de la valeur s à 0.

La fonction tangente hyperbolique est une fonction continue de $\mathbb{R}$ vers $[-1;1]$; pour la modélisation, il faut ramèner cette fonction sur un intervalle de temps et d'amplitude bien défini. Pour cela, le modèle en tangente hyperbolique a besoin de trois paramètres présentés ci-dessous :

- Le temps de réponse : Chaque transition étant de durée différente (durée en fonction de L), le modèle en tangente hyperbolique doit avoir comme paramètre principal cette valeur. On note cette valeur τ.

- La valeur finale de la transition : Cette valeur correspond au niveau de gris que l'on désire atteindre à la fin de la transition. Elle est notée L

- Le décalage temporel : Bien que non-utilisée dans les calculs qui suivent, cette valeur permet de fixer le début de la transition à $t = 0$ ms. Celle-ci est notée I.

A l'aide de ces trois paramètres, et sachant que la mesure analytique du temps de réponse pour une transition (calcul théorique égal à la durée pour que ce modèle de transition passe de 10% à 90% de sa valeur finale) doit être identique à sa valeur expérimentale (valeur mesurée), on représente la modélisation en tangente hyperbolique de T_0^L par la fonction :

$$f(t) = \left(\tanh\left(\frac{2 \times \tanh^{-1}(0.8) \times (t - I)}{\tau}\right) + 1\right) \times \frac{L}{2}$$

8.6.2 Validation

Afin de tester le modèle en tangente hyperbolique défini ci-dessus, nous avons utilisé des données capturées grâce à la caméra linéaire sur un moniteur PC de type Twisted Nematic (TN). Les données enregistrées étant assez bruitées (présence parasite du Backlight de l'écran et du bruit analogique de la mesure), celles-ci ont été rendues plus lisibles à l'aide d'un banc de filtres médians.

La figure 8.10 montre une transition mesurée (du noir au niveau de gris 96) ainsi que le modèle en tangente hyperbolique associé (le temps de réponse est estimé avec le modèle de la section 8.5).

La figure 8.11 montre l'évolution de l'erreur absolue au cours de la transition de 0 à 96. L'erreur la plus importante est de l'ordre de 4 niveaux de gris, répartie essentiellement sur les étapes d'accélération et de décélération de la transition.

On notera que le maximum de l'erreur absolue n'est pas constant selon la valeur finale choisie ; si il est de l'ordre de 4 niveaux de gris pour T_0^{96}, il descend en dessous de 2 niveaux et demi de gris pour T_0^{32}

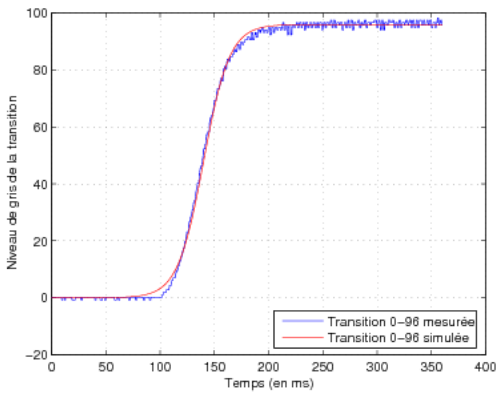


FIGURE 8.10 – Comparaison entre transition mesurée et modèle

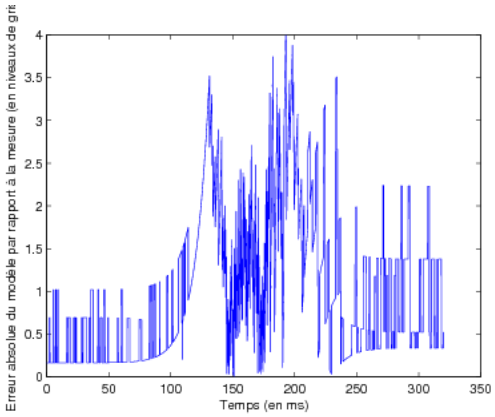


FIGURE 8.11 – Erreur absolue entre le modèle et la transitions de 0 à 96

mais augmente jusqu'à plus de 5 niveaux de gris pour T_0^{128}.

La détermination exacte du début et de la fin d'une transition étant difficile, des valeurs de départ de $\frac{L}{1000}$ (au lieu de 0) et d'arrivée de $\frac{999L}{1000}$ (au lieu de L) ont été choisis. En moyennant sur cette durée totale de la transition, et en posant C_{data} la mesure et C_{modele} la valeur donnée par le modèle, on obtient l'erreur relative suivante :

$$\frac{\Delta Er}{Er} = \frac{|C_{data} - C_{model}|}{|C_{data}|} = 1.5\%$$

Le bruit n'étant pas nul, on peut considérer cette valeur comme très bonne et en déduire donc que le modèle proposé est bien en accord avec les données réelles.

8.6.3 Généralisation aux transitions montantes

Dans cette partie, on généralise le temps de réponse et la forme des transitions de ${}^{\uparrow}\mathbb{T}_0$ aux transitions ayant une valeur initiale non nulle.

Après avoir défini un modèle de temps de réponse et de comportement pour ${}^{\uparrow}\mathbb{T}_0$, il faut maintenant pouvoir obtenir la forme et le temps de réponse de toutes les transitions de ${}^{\uparrow}\mathbb{T} - {}^{\uparrow}\mathbb{T}_0$, c'est-à-dire l'ensemble des transitions montantes dont la valeur initiale est non nulle. On notera ce nouvel ensemble ${}^{\uparrow}\mathbb{T}_*$.

En observant la figure 8.5, on remarque que toutes les transitions de l'ensemble ${}^{\uparrow}\mathbb{T}_*$ peuvent se déduire de ${}^{\uparrow}\mathbb{T}_0$, c'est-à-dire que la transition T_X^Y (avec $X > 0$ et $X < Y$) peut être calculée à partir de la transition T_0^Y.

Pour cela, on extrait premièrement du modèle en tangente hyperbolique la partie de la transition supérieure à la valeur X (partie commençant de t_X et terminant à la fin de la transition). Puis, on remplace la partie commençant de 0 et terminant à t_X par la valeur X. On obtient finalement une nouvelle courbe de la transition T_X^Y sur laquelle il est envisageable de calculer le temps de réponse comme le montrent les exemples figures 8.12 et 8.13 .

Le paragraphe suivant va démontrer comment, à partir du temps de réponse de la transition initiale T_0^Y, il est possible de calculer simplement le temps de réponse de la nouvelle transition T_X^Y avec $X \in]0..Y[$.

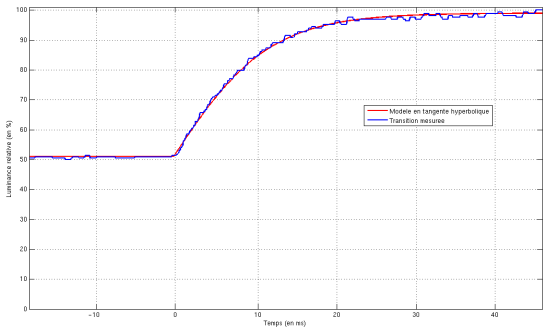


FIGURE 8.12 – Comparaison entre T_x^y mesuré avec $x = y/2$ et son modèle en tangente hyperbolique

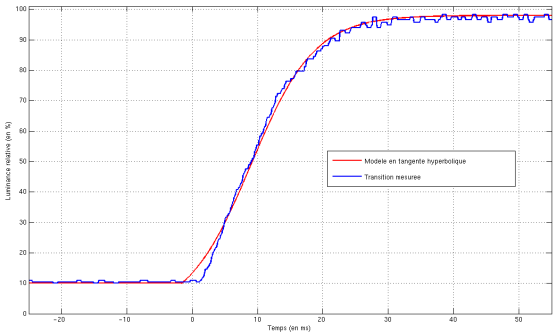


FIGURE 8.13 – Comparaison entre T_x^y mesuré avec $x = y/10$ et son modèle en tangente hyperbolique

Calcul du temps de réponse Soit T_0^{Lf} la transition ayant comme valeur initiale zéro, comme valeur finale L_f et comme temps de réponse τ. On désire maintenant calculer le nouveau temps de réponse τ' de la transition $T_{L_i}^{L_f}$, transition ayant toujours la même valeur finale mais possédant maintenant la valeur initiale L_i non nulle.

Soient t_1 et t_2, définis tels que $\tau' = t_2 - t_1$ avec t_1 et t_2 les instants pendant lesquels la transition est respectivement à 10% et 90% de sa valeur finale [Vid05] :

$$\begin{cases} f(t_1) = L_i + 0.1 \times (L_f - L_i) = \left(\tanh\left(\dfrac{2 \times \tanh^{-1}(0.8) \times (t_1 - I)}{\tau}\right) + 1\right) \times \dfrac{L_f}{2} \\ f(t_2) = L_i + 0.9 \times (L_f - L_i) = \left(\tanh\left(\dfrac{2 \times \tanh^{-1}(0.8) \times (t_2 - I)}{\tau}\right) + 1\right) \times \dfrac{L_f}{2} \end{cases}$$

On pose $p = \dfrac{L_i}{L_f}$ pourcentage de la valeur initiale de la transition par rapport à la valeur finale. Sachant que, pour $x \in]-1, 1[$, $tanh^{-1}(x) = \dfrac{1}{2} ln(\dfrac{1+x}{1-x})$ et en procédant de façon identique à la section 8.6.1, on obtient la dernière équation reliant τ' à τ :

$$\tau' = \frac{\tau}{4.ln(3)}\ ln\left(1 + \frac{16}{1.8\ p + 0.2}\right)$$

On pose $\tau' = C_t.\tau$. On obtient donc un facteur C_t tel que :

$$C_t = \frac{1}{4.ln(3)}\ ln\left(1 + \frac{16}{1.8\ p + 0.2}\right)$$

La valeur C_t permet de calculer le temps de réponse de toutes les transitions de ${}^{\uparrow}\mathbb{T}_*$ à partir des temps de réponse de ${}^{\uparrow}\mathbb{T}_0$.

Le tracé de C_t sur la figure 8.14 montre que cette fonction est une application strictement décroissante de $[0, 1]$ vers $[0.5; 1]$; La valeur du temps de réponse de T_X^Y ne peut donc pas descendre en dessous de 50% du temps de réponse de T_0^Y. Bien évidemment, le temps de réponse de la transition T_X^Y devient égal au temps de réponse de la transition T_0^Y lorsque la valeur p devient nulle.

8.6.4 Conclusion

Le but principal de la modélisation que nous proposons est d'accélérer la génération de la LUT. En général, celles-ci comportent 32×32 valeurs, soit 496 pour les transitions montantes et autant de mesures à réaliser, contre 4 à 7 pour notre modèle. Autrement dit, la méthode permet de réduire de 95% le temps

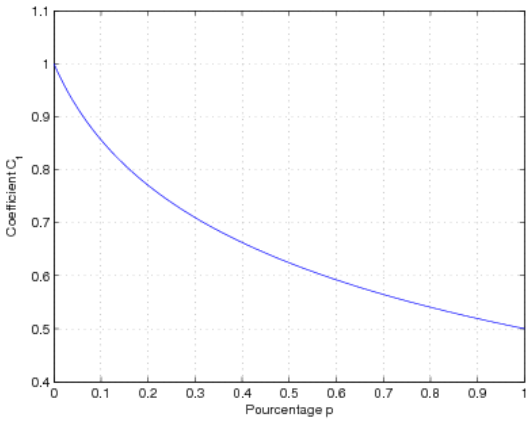


FIGURE 8.14 – Evolution de la fraction C_t en fonction de $p\frac{L_i}{L_f}$

dévolu aux mesures. Cette solution apporte un deuxième avantage non négligeable économiquement : elle permet de supprimer sur le composant final la mémoire ROM stockant la LUT.

8.7 Réduction de traînées par rehaussement du noir

Ici, nous nous intéressons de manière très pragmatique au problème de traînées provoquées par des transitions de type ${}^{\uparrow}\mathbb{T}_0$, c'est-à-dire les transitions partant du noir. L'idée de base est de remplacer le noir par du "presque noir".

8.7.1 Principe

Dans cette étude, nous proposons de réduire la taille des traînées sombres sans utiliser de mémoire d'image comme c'est le cas avec les *overdrives* classiques. Pour cela, nous modifions la valeur des pixels sombres afin d'éviter les temps de réponse trop longs liés aux transitions de ${}^{\uparrow}\mathbb{T}_0$, comme le montre la figure 8.6. Cette modification est basée sur la relation entre $T_{L_i}^{L_f}$ et $T_0^{L_f}$: toutes les transitions ayant comme valeur finale L_f peuvent être déduites de $T_0^{L_f}$. La figure 8.5 montre un exemple de cette relation avec la superposition de T_0^{128}, T_{32}^{128}, T_{64}^{128} et T_{96}^{128}.

Dans cette figure, on note une différence significative de temps de réponse entre deux transitions partant de niveaux sombres : T_0^{128} et T_{32}^{128}. On en déduit la relation :

$$\forall X \in \mathbb{L}_n - \{0\}, \tau(T_0^Y) > \tau(T_X^Y)$$

8.7.2 Une technique sans mémoire d'image

La modification des pixels sombres est appliquée de telle façon à ce que la différence entre leur valeur initiale L_i et leur valeur modifiée $L_i + \delta L_i$ soit invisible. Contrairement aux algorithmes classiques d'*overdrive*, cette solution n'est pas basée pixel mais tient compte de valeurs statistiques (valeur minimum, maximum et moyenne) extraites des images précédentes et courantes. De cette façon, il n'est pas nécessaire de mémoriser l'image précédente entière, mais seulement les valeurs statistiques indiquées, qui ne représentent que quelques octets stockés dans des registres.

FIGURE 8.15 – Rehaussement du noir

FIGURE 8.16 – Rehaussement global

8.7.3 Le contrôle de montée

La comparaison entre les valeurs statistiques des images précédentes et courantes détermine si des images successives ont des caractéristiques colorimétriques identiques ou non. Dans le premier cas, l'image n'est pas corrigée, puisqu'elle l'a déjà été et que des corrections successives provoqueraient une dérive des couleurs. Dans le second cas, on applique la modification aux pixels sombres par rapport aux statistiques de l'image courante (algorithme 8.7.3).

Algorithme 10 : Algorithme de l'*overdrive* sans mémoire d'image

Entrée-sortie : I, une image de n pixels
Entrées :
Min_t, Max_t, Moy_t, $CumulHaut_t$, des statistiques de l'image I
Min_{t-1}, Max_{t-1}, Moy_{t-1}, $CumulHaut_{t-1}$, des statistiques de l'image précédente
Paramètres :
L_{max}, niveau de gris pour le calcul du cumul
N_{max}, nombre de pixels ...
$Seuil_{moy}$, seuil de niveaux de gris moyen
$Seuil_{sombre}$, ...
δ, a, b, paramètres pour la correction

```
si (Min_t ≠ Min_{t-1} OU Max_t ≠ Max_{t-1} OU Moy_t ≠ Moy_{t-1} OU
CumulHaut_t ≠ CumulHaut_{t-1}) ET (Min_t < Seuil_min) alors
    si Moy_t < Seuil_moy alors
        // cas d'une image sombre
        si CumulHaut_t(L_max) < N_max alors
            pour i ← 1 à n faire
                // on applique le rehaussement global
                I(i) = I(i) + δ ;
            fin
        fin
    sinon
        // cas d'une image claire
        pour i ← 1 à n faire
            si I(i) < Seuil_sombre alors
                // on applique le rehaussement du noir
                I(i) = a × I(i) + b ;
            fin
        fin
    fin
fin
```

Pour ne pas créer de nouveaux artefacts visuels, la modification ne sera pas équivalente si l'image est claire ou foncée. La modification des pixels s'opérera donc dans certaines conditions. De ce fait, deux corrections sont utilisés : le rehaussement du noir pour les images claires (figure 8.15) et le rehaussement global pour les images sombres (figure 8.16).

Les images globalement sombres ($Moy_t < Seuil_{moy}$) peuvent être éclaircies globalement : tous les pixels de l'images ont leur niveau de gris corrigé par un offset δ. Comme l'image est sombre, la saturation des pixels clairs éventuels (valeur $> 255 - \delta$) est très limitée.

Les images globalement claires ($Moy_t \geq Seuil_{moy}$) ne peuvent être éclaircies globalement car la saturation des pixels clairs est trop importante. Pour ces images, les pixels clairs ne sont pas modifiés.

8.7.4 Résultats et conclusion

La figure 8.17 montre un exemple de l'amélioration du temps de réponse de T_0^{32} avec la méthode du contrôle de montée. La transition est simulée sur un moniteur 19" TN-LCD avec un temps de réponse de 16ms selon la norme ISO.

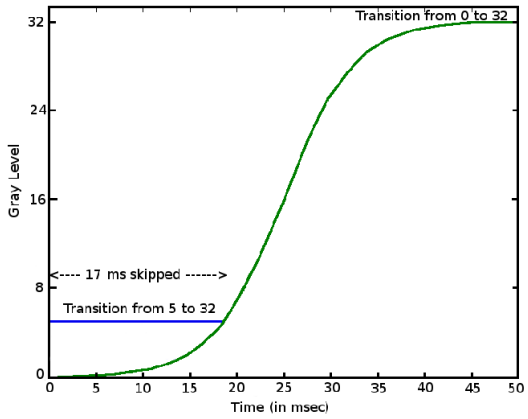


FIGURE 8.17 – La transition T_5^{32} est plus rapide que la transition T_0^{32} mais néanmoins visuellement similaire

Au lieu d'afficher une valeur noire (0), le contrôle de montée la change en valeur presque noire (5). Le temps de réponse de la transition T_0^{32} est d'environ 50ms et passe à 33ms avec le contrôle de montée, soit un gain de 34% . La conséquence immédiate est une réduction visible du flou sombre dû au mouvement.

La figure 8.18 montre la différence visuelle entre la traînée originale (lors d'une transition T_0^{32}) et celle obtenue avec le rehaussement du noir avec la transition T_5^{32}. La différence entre les deux valeurs de noir n'est pas discernable, néanmoins la taille de la traînée est nettement diminuée, ceci sans nécessiter de mémoire d'image contrairement aux systèmes existants.

FIGURE 8.18 – Réduction de la traînée : (a) sans rehaussement du noir, (b) avec rehaussement du noir

Chapitre 9

Conclusion et perspectives de recherche

Sommaire

9.1 Conclusion

DANS la recherche que j'ai effectuée et dirigée, j'ai essayé de conserver une certaine diversité et de faire évoluer ma thématique dans une direction qui permette aux personnes avec qui je travaille et que je forme à la recherche, d'être performantes dans un domaine difficile, concurrentiel et qui évolue rapidement. Ainsi, j'ai pris soin en partant d'images monochromes d'étendre mon intérêt aux images couleurs puis aux vidéos. Partant d'extraction de primitives de bas niveau, j'ai également été attiré progressivement par des problématiques plus proches du haut niveau et passer du signal aux objets. Néanmoins, toujours avec le souci de développer des solutions génériques qui puissent s'adapter à des contraintes, des modèles ou domaines particuliers. Le fil conducteur de cette recherche est sans doute le souci de proposer des briques logicielles et des applications permettant de répondre à des problèmes en traitement et analyse d'images.

Je rappelle ci-dessous de façon synthétique la contribution des travaux que j'ai menés et encadrés :

- Déclinaison et valorisation de la pyramide irrégulière en pyramide d'images couleurs, pyramide locale, pyramide évolutive, pyramide de surfaces, pyramide initialisée par une LPE.
- Segmentation de personnes non supervisée.
- Détection d'objets de premier plan avec caméra fixe.
- Suivi précis d'objets dans un plan séquence.
- Extraction d'objets clés dans les vidéos. Cet outil est générique et constitue une base pour du suivi, de la construction de résumé, de l'indexation et de la modélisation 2D du contenu.
- Proposition d'interopérabilité entre objets clé et suivi : les objets clé peuvent être des références permettant de réinitialiser un suivi et de simplifier les problèmes d'occultation des objets.
- Etude et développement d'applications pour les vidéos structurées.
- Désentrelacement vectorisé avec une approche orientée graphe et non pas matrice.
- Agrandissement d'image pour la TV HD évitant les artéfacts classiques
- Réduction des effets de traînées et des coûts de production par modélisation du comportement des cellules LCD.
- Participation et animation de projets régionaux, nationaux et Européens.

- Valorisation industrielle des travaux de recherche avec l'incubation d'une startup.
- Collaborations variées avec l'industrie.

9.2 Perspectives de recherche

A l'avenir, j'aimerais orienter ma recherche vers la vision par ordinateur et plus précisément développer des méthodes et algorithmes temps réel pour l'analyse, l'interprétation et la reconstruction 3D de la scène. J'ai déjà commencé à amorcer ce virage grâce au projet MOOV3D : j'ai récemment fait une étude comparative des techniques de détection et de mise en correspondance de points d'intérêt pour des vidéos stéréoscopiques. A titre d'exemple, la méthode la plus rapide (FAST) permet d'effectuer ce traitement sur 200 images par seconde sur mon ordinateur portable. Ces méthodes sont largement utilisées, notamment pour faire du suivi d'objets. Elles pourraient également l'être pour contrôler une méthode de segmentation de façon à mieux gérer les occultations et obtenir une meilleure estimation du mouvement. Voici quelques travaux récents qui m'inspirent et m'incitent à penser qu'il y a encore beaucoup de latitude dans le domaine pour proposer des solutions innovantes : dans un article très récent, Xu et al [XLD11] présentent une méthode où sur-segmentation et opérateur SIFT collaborent pour réaliser de la segmentation spatio-temporelle et estimer la trajectoire 3D d'objets en mouvement à une fréquence de 10 à 15 images par seconde. De manière plus générale, des progrès importants ont été réalisés en segmentation spatio-temporelle temps-réel : Tsai et al [TFR10] proposent un algorithme qui réalise la segmentation d'un objet et l'estimation de son mouvement en résolvant un problème global d'étiquetage dans une formulation de champs aléatoires de Markov volumétrique. Bibby et al quant à eux [BR08] mettent en correspondance les images successives pour compenser le mouvement de la caméra, segmentent avec une approche probabiliste les images et gèrent les apparitions d'objets, le tout à une fréquence de 85Hz.

Je vais également m'intéresser à la segmentation de vidéo fondée sur des données multimodales : d'une part dans le projet actuel MOOV3D où le *smartphone* de STEricson propose une caméra stéréoscopique et des capteurs (GPS, accéléromètres, boussole) qui devraient nous permettre de mieux contrôler la segmentation et la reconstruction 3D. D'autre part dans le projet READ-PLAY que nous avons déposé auprès du pôle de compétitivité Imaginove avec deux PME de la région (La Cuisine aux Images Production et GERIP) où j'ai proposé pour augmenter la qualité et la productivité de vidéos enrichies de coupler une caméra classique à une caméra TOF * qui capture la profondeur de la scène. Dans ce projet de deux ans, nous devrons développer les algorithmes et l'IHM *offline* permettant de segmenter les objets d'intérêt avec peu d'intervention humaine. Ce projet de 24 mois qui vient juste d'être accepté vise à réaliser des modules pédagogiques basés sur des séquences vidéos dynamiques pour la formation professionnelle d'adultes en situation d'illettrisme.

Sur un plan applicatif, deux aspects m'intéressent : tout d'abord, j'aimerais proposer nos futurs algorithmes pour les plateformes mobiles (smartphone, tablettes, ...) en prenant en compte les contraintes matérielles (mémoire, CPU, GPGPU). Ce type de dispositif est sans doute amené à être le plus demandeur de cette technologie, si l'on excepte la robotique.
Ensuite, le transfert technologique que j'ai mené pour la création d'une startup a permis de développer une plateforme logicielle qui est revenue au laboratoire et sous ma responsabilité scientifique. Cette plateforme est une base pratique et fonctionnelle pour implémenter et exploiter de nombreux algorithmes pour le traitement et l'analyse de vidéos en post-production. Dans l'année qui vient, je tiens à reprendre ces développements et mettre à la disposition de la communauté et des utilisateurs cette plateforme. Le but est également de valoriser notre savoir-faire dans ce domaine et de favoriser son évolution et sa pérennité.

∗. Time Of Flight

Annexes

Annexe A

Licence Professionnelle

Licence Professionnelle **3ID** - IUT2, Département Informatique, place DoyenGosse - Grenoble
Planning 2005-2006 - Images et Vidéo - RIM-4 (56h) . Responsable : **Pascal Bertolino**

Mots clef : Images et vidéos numériques, traitement, analyse, retouche, compression, normes, synthèse 3D, animation

Cours	Salle banalisée
TP	Salles machines Mac 25 - 27 ( ) et SX ??? ( )

Date	Intervenant	Cours	TP
24 janvier / 27 janvier	P.Bertolino	**images numériques, filtrage** Chaîne du traitement d'images, représentation, Amélioration, rehaussement, Filtrage linéaire, non linéaire (morphologie mathématique)	ImageJ, prise en main, manipulation d'images, LUT, filtrage, premières macros
6 février	P.Bertolino	**Segmentation** Seuillage, composantes connexes, segmentation contours/régions	ImageJ, traitements de bases sur les pixels par programmation
10 février	P.Bertolino	**Segmentation** Seuillage, composantes connexes, segmentationcontours/régions	ImageJ, composantes connexes, morphologie mathématique
20 février	P.Bertolino	**Transformations** Géométriques, photométriques	ImageJ , segmentation, applications
24 février	P.Bertolino	**Analyse** Forme, couleur, texture, distribution	Mini projet reconnaissance de formes avec ImageJ
6 mars	P.Bertolino	**Retouche d'images** Présentation de GIMP	Retouche d'images et infographie avec GIMP
10 mars	P.Bertolino	**Compression d'images fixes** compression GIF, PNG, JPEG : Formats d'images, besoins, mode sans perte, mode avec perte, théorie débit /distorsion. Compression JPEG 2000 : fonctionnalités, ondelettes, codage arithmétique	Mini projet infographie avec GIMP
20 mars / 24 mars	P.Bertolino	**Codage des séquences vidéo** compresion vidéo mpeg1-2, 4. Les applications, les enjeux. Principe de traitement. Mouvement : estimation, prédiction, compensation, normes	Compressions sans pertes : différents formats . Comparaison JPEG / JPEG2000 avec perte. Evaluation objective et subjective des méthodes
3 avril	P.Bertolino	**Les applications**	Compression mpeg. Analyse d'un bitstream mpeg. Production de bitstream avec différents jeux de paramètres.
	P. Barla	**Synthèse d'image** Introduction à Java3d	Personnage
	P. Barla	**Synthèse d'image** Formes et maillages	Maillage rectangulaire
	P. Barla	**Synthèse d'image** Pipeline graphique	Textures
	P. Barla	**Illumination**	Illumination
	P. Barla	**Animation**	Interpolation de positions-clefs, Comportements
	Contrôles	* 3 notes de mini-projets de TP (coefs 7, 6,7) . * 1 note de contrôle écrit : coef30 (**Examen sans documents, une feuille double manuscrite personnelle de notes autorisée**)	

FIGURE A.1 – Programme du cours Images et Vidéo de la licence professionnelle "Systèmes informatiques et logiciels" option "informatique, internet, images et documents" (3ID)

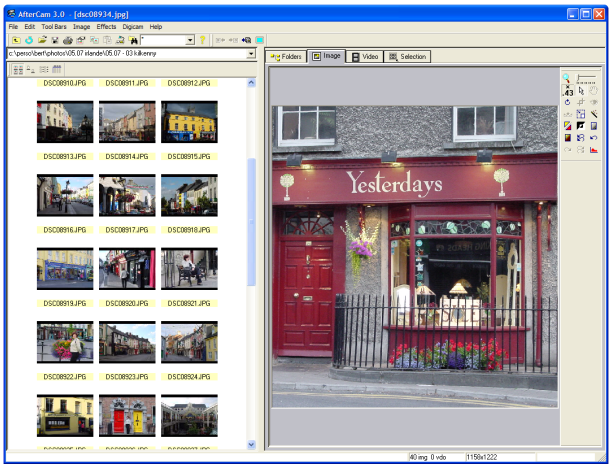
Annexe B

Site web de Vizway

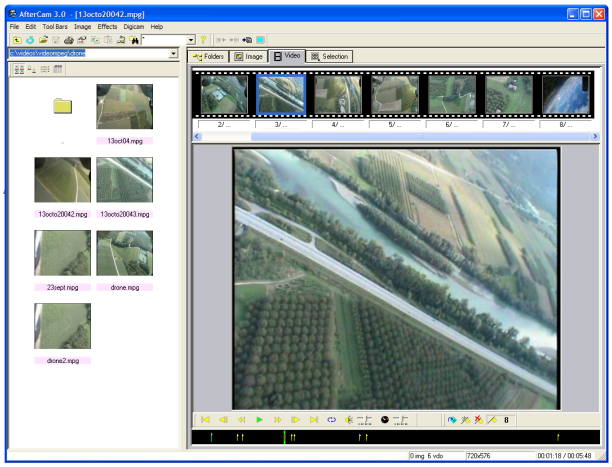
FIGURE B.1 –

Annexe C

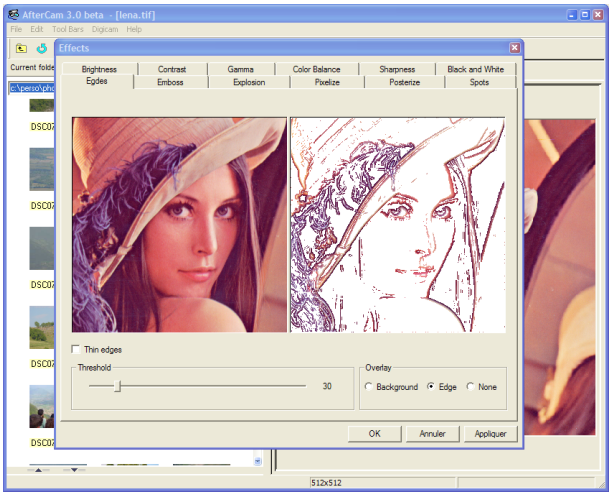
Logiciel AfterCam



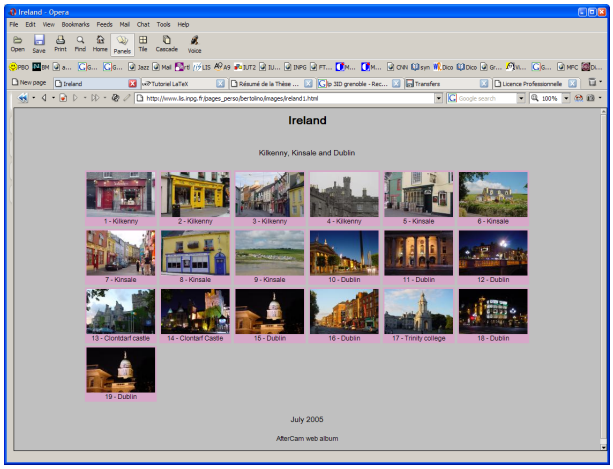
(a) Visualisation d'images



(b) Visualisation d'une vidéo et construction d'un résumé



(c) Exemple d'édition d'images



(d) Album web réalisé avec AfterCam

FIGURE C.1 – Copies d'écran du logiciel AfterCam que j'ai conçu et développé

Annexe D

Vidéos structurées

INTERVIEW

CRÉATION DE DOCUMENTS VIDÉO STRUCTURÉS ET INTERACTIFS

Interview de Patrick Bouthemy, responsable scientifique du projet VISTA, Irisa/INRIA Rennes et Roger Mohr, responsable scientifique du projet MOVI, INRIA Rhône-Alpes

L'INRIA a développé, en partenariat avec Alcatel Corporate Research Center (CRC), un prototype d'environnement de création de documents "hypervidéo" (VideoPrep) et d'exploitation de ces vidéos structurées et interactives (VideoClic). Ce concept de vidéo "cliquable" reste encore peu mis en œuvre dans le monde, les tout premiers produits présents sur le marché relevant d'une approche essentiellement manuelle. L'environnement VideoPrep introduit des fonctionnalités nouvelles et des traitements automatiques, tout en offrant une interface appropriée d'édition des résultats (ajout, élimination, correction).

Inedit : Quels partenaires ont travaillé sur ce prototype ?

Les travaux ont été menés par les équipes de Roger Mohr du projet MOVI de l'INRIA Rhône-Alpes et de Patrick Bouthemy du projet VISTA à l'INRIA Rennes, en collaboration avec l'équipe de Marc Mautref d'Alcatel CRC à Marcoussis. L'objectif était de mettre au point des vidéos susceptibles d'être consultées de manière rapide et adaptée, et pouvant produire des actions sur désignation d'une zone préalablement indexée de l'image.

Inedit : Pouvez-vous préciser les fonctionnalités d'un environnement hypervidéo ?

VideoPrep permet au concepteur d'un document vidéo de le structurer, de créer des liens et de rendre sensibles des objets. Un utilisateur muni de VideoClic peut interagir avec la vidéo : rechercher par exemple la séquence où apparaît un objet donné ou interrompre la vidéo par un clic pour obtenir des précisions sur l'élément qui a retenu son attention.

Inedit : Sur quels logiciels s'appuie l'environnement VideoPrep ?

Il exploite trois logiciels de l'INRIA :

• Le logiciel MD-shots permet de détecter au sein de la vidéo les changements de plans, quels que soient leurs types : coûts, transitions progressives (fondus, volets, etc) selon un principe et une paramétrisation de l'algorithme identiques. Il exploite de façon originale une information intrinsèque à la vidéo (le mouvement dominant estimé entre deux images successives).

• Le logiciel D-Motion exploite à nouveau le mouvement dominant estimé entre deux images. Il le compense, et recherche les zones de mouvement résiduel significatif à l'aide d'un schéma d'étiquetage statistique contextuel.

• Le logiciel d'indexation d'images permet de rechercher des motifs similaires dans un large ensemble images, grâce à l'utilisation de descripteurs locaux.

Inedit : Quels sont les prolongements de ces travaux ?

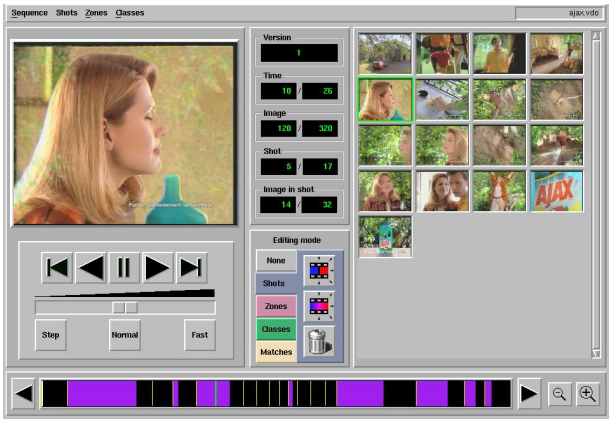
Cet environnement (VideoPrep et VideoClic) constitue un prototype incluant les fonctionnalités décrites ci-dessus. Le passage à un stade industriel devrait inclure l'écriture en Java de VideoClic, un suivi avant et arrière des entités extraites, la détermination de descripteurs pour des formes changeantes, un langage de description de type XML, un langage de requêtes. Ces développements sont en cours.

Plus largement, ces travaux trouveront une continuation dans le cadre du projet Agir qui vient d'être accepté par le RNRT (Réseau National de Recherche en Télécommunications). Ce projet a pour but de traiter les problèmes d'indexation par le contenu de documents multimédias avec un couplage explicite des aspects de texte, images, vidéo et son, ainsi que de recherche d'informations dans de telles bases. Il cherchera aussi à contribuer au développement de la norme MPEG-7. Les partenaires en sont Alcatel-CRC, Arts Video Interactive, l'Afnor, l'INRIA, l'INT Evry, l'Irit, le LIP6. ■

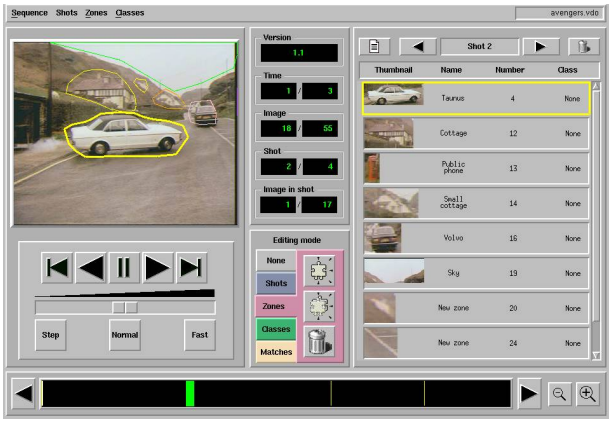
Contacts :
Patrick Bouthemy :
projet VISTA, Irisa/INRIA Rennes
Tél : +33 2 99 84 72 74
Patrick.Bouthemy@inria.fr
Roger Mohr :
projet MOVI, Gravir/INRIA Rhône-Alpes
Tél : +33 4 76 61 52 25
Roger.Mohr@imag.fr

Environnement VideoPrep permettant de structurer la vidéo en plans élémentaires (dans la partie droite, imagettes représentant les plans détectés), d'extraire les objets mobiles (visualisation dans la partie gauche) et d'indexer ces objets dans une base de données.

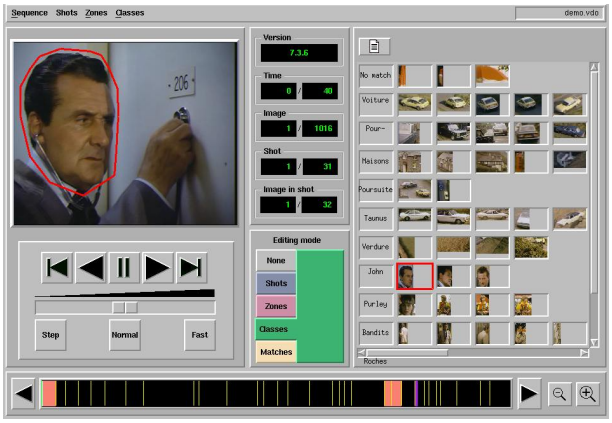
FIGURE D.1 – Extrait de Inedit (lettre d'information de l'INRIA), n° 18, mars 1999



(a) Découpage en plan



(b) Extraction de zones en mouvement



(c) Construction de classes par indexation

(d) L'application d'hypervidéo

FIGURE D.2 – Construction et utilisation d'une vidéo structurée. Les applications VideoPrep (a)(b)(c) et VideoClic ((d)

Annexe E

Pyramide locale initialisée par carte d'homogénéité

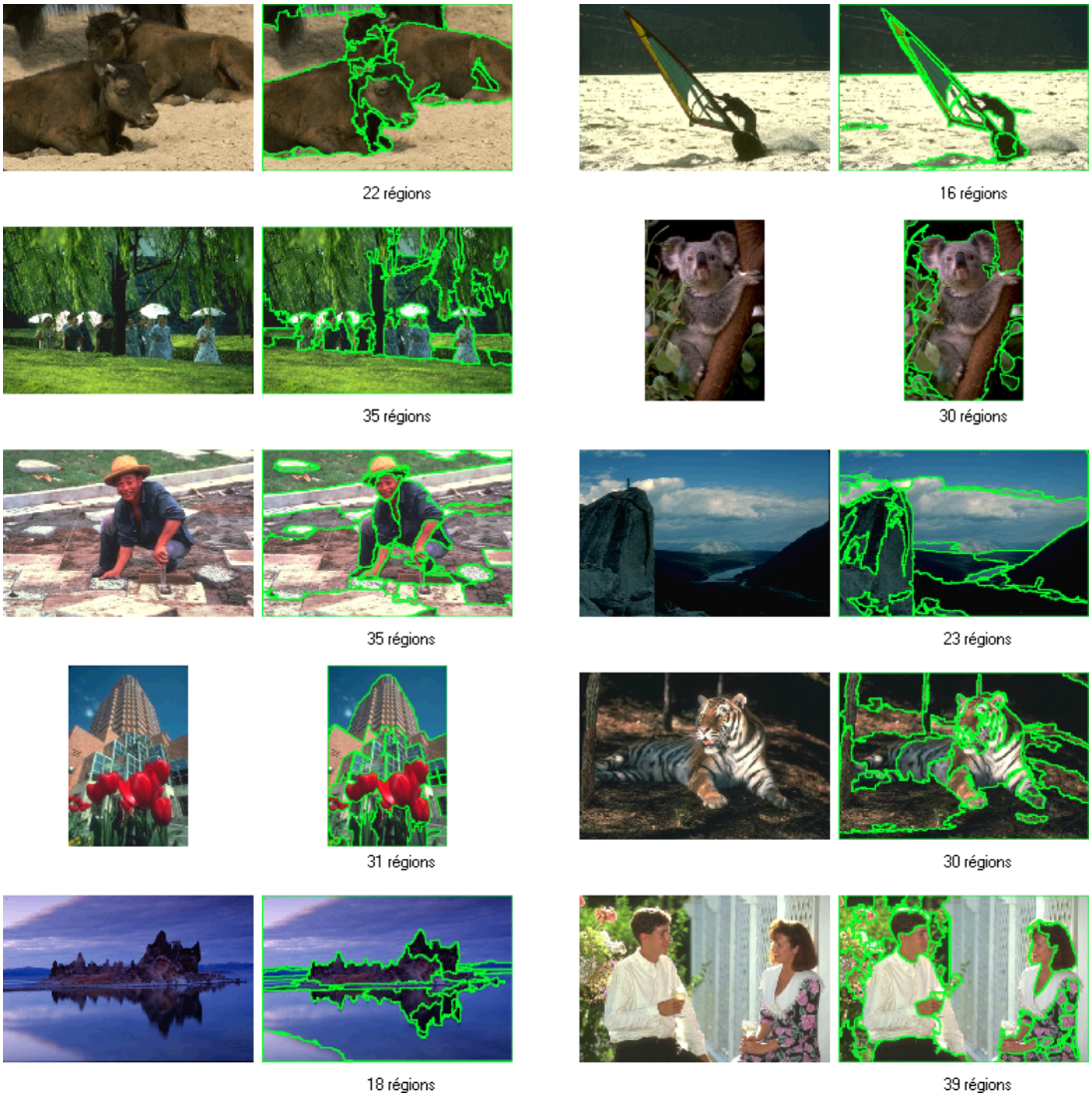


FIGURE E.1 – Planche 1. Des images de la banque Corel et les contours des régions obtenues. Le nombre de régions est indiqué

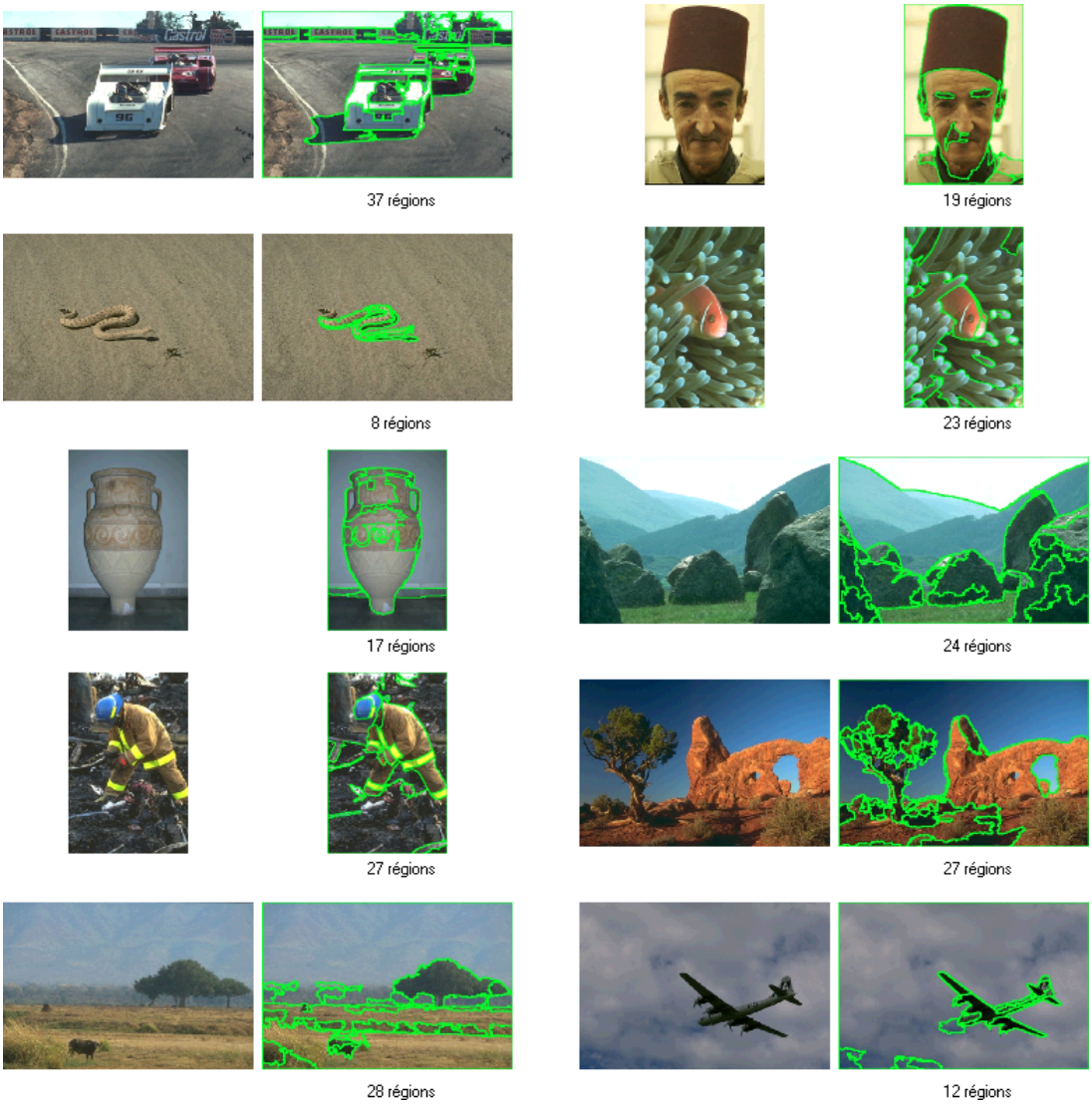


FIGURE E.2 – Planche 2. Des images de la banque Corel et les contours des régions obtenues. Le nombre de régions est indiqué

Annexe F

Suivi d'objets dans une séquence vidéo

a) partition initiale en objets P(0)

b) I(0)

c) VOP(0)

d) I(5)

e) VOP(5)

f) I(20)

g) VOP(20)

h) I(50)

i) VOP(50)

FIGURE F.1 – Suivi d'un objet hétérogène dans la séquence Coastguard (La première colonne présente les images de la séquence à 4 instants différents, la seconde colonne représente l'objet segmenté)

a) partition initiale en objets P(0)

b) I(0) c) I(5) d) I(20) e) I(50)

f) VOP1(0) g) VOP1(5) h) VOP1(20) i) VOP1(50)

j) VOP2(0) k) VOP2(5) l) VOP2(20) m) VOP2(50)

n) VOP3(0) o) VOP3(5) p) VOP3(20) q) VOP3(50)

r) VOP4(0) s) VOP4(5) t) VOP4(20) u) VOP4(50)

v) VOP5(0) w)VOP5(5) x) VOP5(20) y) VOP5(50)

FIGURE F.2 – Suivi de plusieurs objets dans la séquence Mother&Daughter (La deuxième ligne représente les images originales à 4 instants différents, les lignes suivantes fournissent les objets segmentés)

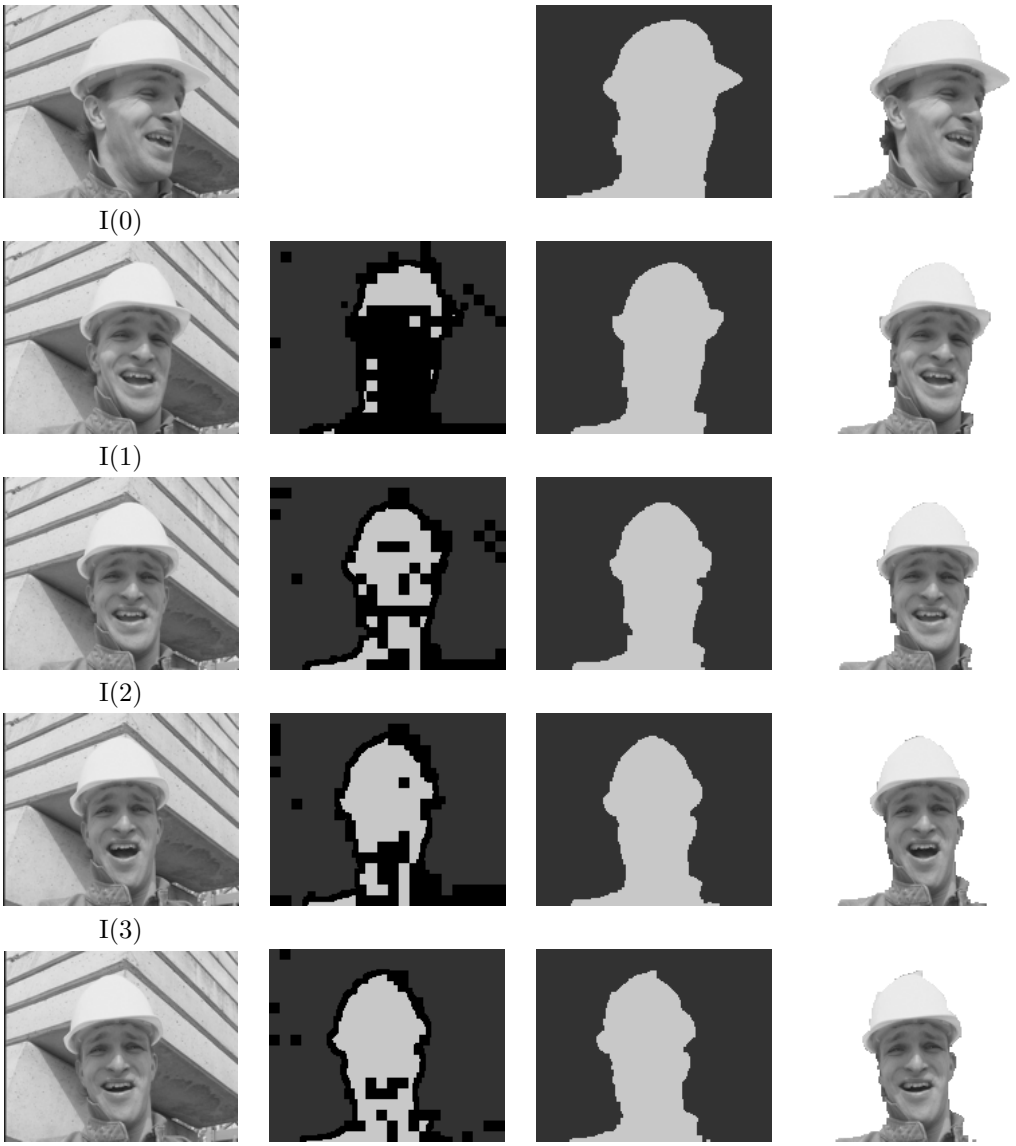


FIGURE F.3 – Suivi d'un objet non rigide dans la séquence sous-échantillonnée ForemanBis (La première colonne représente les images successives de la séquence, la seconde fournit le résultat de la projection de partition, la troisième présente la partition en objets obtenue, la dernière donne l'objet vidéo segmenté)

FIGURE F.4 – Suivi d'un objet non rigide dans la séquence fortement sous-échantillonnée CarphoneBis (La première colonne représente les images successives de la séquence, la seconde fournit le résultat de la projection de partition, la troisième présente la partition en objets obtenue, la dernière donne l'objet vidéo segmenté)

Annexe G

Extraction de vues clés

FIGURE G.1 – La vidéo Clio montre une voiture qui fait le tour d'un rond point. Ce résultat donne dans l'ordre chronologique l'ensemble des vues clé extraites. L'extraction de vues clé peut fournir des bons résultats de suivi

Bibliographie

[ABCL06] P. Adam, P. Bertolino, J.-M. Chassery, and F. Lebowsky. LCD response time estimation. In *International Display Research Conference*, Kent, Ohio, USA, september 2006.

[ABL06] P. Adam, P. Bertolino, and F. Lebowsky. Les écrans LCD dans le flou. In *MajecSTIC*, Lorient, France, novembre 2006.

[ABL07a] Pierre Adam, Pascal Bertolino, and Fritz Lebowsky. A simple LCD response time measurement based on a CCD line camera. In *Asia Display*, page CD, Shangai Chine, 03 2007. Département Images et Signal.

[ABL07b] Pierre Adam, Pascal Bertolino, and Fritz Lebowsky. Mathematical modeling of the LCD response time. *Journal of the Society for Information Display*, 15(8) :571–577, 08 2007.

[AOW+98] A.A. Alatan, L. Onural, M. Wollborn, R. Mech, E. Tuncel, and T. Sikora. Image sequence analysis for emerging interactive multimedia services - the european COST 211 framework. *IEEE Transactions on Circuits and Systems for Video Technology*, 8(7) :802–813, November 1998.

[BBB+98a] S. Benayoun, H. Bernard, P. Bertolino, P. Bouthemy, M. Gelgon, R. Mohr, C. Schmid, and F. Spindler. Structuration de videos pour des interfaces de consultation avancées. In *Journées CORESA*, pages 207–214, Lannion, France, juin 1998.

[BBB+98b] S. Benayoun, H. Bernard, P. Bertolino, P. Bouthemy, M. Gelgon, R. Mohr, C. Schmid, and F. Spindler. Structuring video documents for advanced interfaces. In *ACM Multimedia*, Bristol, UK, september 1998.

[BCNT06] L. Bonnaud, A. Caplier, L. Nigay, and D. Tzoradas. Multimodal driving simulator. In *Workshop eNTERFACE*, Dubrovnick, Croatie, august 2006.

[Ber95] P. Bertolino. *Contribution des pyramides irrégulières en segmentation d'images multirésolution.* PhD thesis, Institut National Polytechnique de Grenoble, 30 novembre 1995.

[BFP01a] P. Bertolino, G. Foret, and D. Pellerin. Detecting people in videos for their immersion in a virtual space. In *ISPA 2001, 2nd IEEE Region 8-EURASIP Symposium on Image and Signal Processing and Analysis June*, pages 313–318, Pula, Croatia, 2001.

[BFP01b] P. Bertolino, G. Foret, and D. Pellerin. Détection de personnes dans les vidéos pour leur immersion dans un espace virtuel. In *colloque GRETSI*, Toulouse, France, september 10-13 2001.

[BHF05] P. Bertolino, J. Huart, and G. Foret. Segmentation pyramidale localisée dans un ruban fermé. In *colloque GRETSI*, Louvain-la-Neuve, Belgique, septembre 2005.

[BJ88] Paul J. Besl and Ramesh Jain. Segmentation through variable-order surface fitting. *IEEE Trans. Pattern Anal. Mach. Intell.*, 10(2) :167–192, 1988.

[BJ01] Y.Y. Boykov and M.P. Jolly. Interactive graph cuts for optimal boundary and region segmentation of objects in n-d images. *International Conference on Computer Vision*, 1 :105–112, July 2001.

[BM92] S. Beucher and F. Meyer. The morphological approach to segmentation : the watersheds transformation. In E.R. Doughterty (Ed.), editor, *Mathematical Morphology in Image Processing*, pages 433–481, New-York : Dekker, 1992.

[BM96] P. Bertolino and A. Montanvert. Multiresolution segmentation using the irregular pyramid. In *IEEE International Conference on Image Processing*, pages 257–260, Lausanne, Switzerland, September 16-19 1996.

[BMS+98] P. Bertolino, R. Mohr, C. Schmid, P. Bouthemy, M. Gelgon, F. Spindler, S. Benayoun, and H. Bernard. Building and using hypervideos. In *Workshop on Applications of Computer Vision*, Princeton, N.J., USA, 1998.

[Bon98] L. Bonnaud. *Schémas de suivi d'objets vidéo dans une séquence animée : application à l'interpolation d'images intermédiaires.* PhD thesis, Université de Renne 1, Octobre 1998.

[BPK05] M. Byun, Min KYU Park, and Moon GI Kang. Edi-based deinterlacing using edge patterns. In *IEEE International Conference on Image Processing*, volume 2, pages 1018–1021, 2005.

[BR98] P. Bertolino and S. Ribas. *Image sequence segmentation by a single evolutionary graph pyramid*, pages 93–100. In J.-M. Jolion and W.G. Kropatsch, editors, Graph based representations in pattern recognition. SpringerWienNewYork, 1998.

[BR08] Charles Bibby and Ian Reid. Robust real-time visual tracking using pixel-wise posteriors. In *European Conference on Computer Vision*, ECCV '08, pages 831–844, Berlin, Heidelberg, 2008. Springer-Verlag.

[BRB+04] A. Blake, C. Rother, M. Brown, P. Pérez, and Ph. Torr. Interactive image segmentation using an adaptive gaussian mixture mrf model. In *Eur. Conf. on Computer Vision*, Prague, Czech Republic, May 2004.

[Bru97] E. Bruno. Détection robuste du mouvement dans des séquences d'images rapides. Master's thesis, laboratoire LSIIT, Université Louis Pasteur, Strasbourg, France,, juillet 1997.

[CB01] J. Cutrona and N. Bonnet. Two methods for semi-automatic segmentation based on fuzzy connectedness and watersheds. In *IASTED International Conference on Visualisation, Imaging and Image Processing VIIP' 2001*, 2001.

[CHF01] Yong-Sheng Chen, Yi-Ping Hung, and Chiou-Shann Fuh. Fast block matching algorithm based on the winner-update strategy. *IEEE Transactions on Image Processing*, 10(8) :1212–1222, August 2001.

[CHS05] S. Csomor, K. Hock, and J. Smart. *Cross-Platform GUI Programming with wxWidgets.* Prentice Hall, 2005.

[CT04] Janko Calic and Barry Thomas. Spatial analysis in key-frame extraction using video segmentation. In *Workshop on Image Analysis for Multimedia Interactive Services*, April 2004.

[CWY00] Tao Chen, Hong Ren Wu, and Zheng Hua Yu. Efficient deinterlacing algorithm using edge-based line average interpolation. *Optical Engineering*, 39 :2101–2105, 2000.

[Das99] S. Dasgupta. Learning mixtures of gaussians. In *FOCS '99 : Proceedings of the 40th Annual Symposium on Foundations of Computer Science*, page 634, Washington, DC, USA, 1999. IEEE Computer Society.

[DHA88] G. W. Donohoe, D. R. Hush, and N. Ahmed. Change detection for target detection and classification in video sequences. In *Proc ICASSP*, pages 1084–1087, New York, USA, 1988.

[dHB98] G. de Haan and E.B. Bellers. Deinterlacing - an overview. In *Proceedings of the IEEE*, volume 86, pages 1839–1857, 1998.

[Doy88] T. Doyle. Interlaced to sequential conversion for edtv applications. In *in Proc. 2nd International Workshop Signal Processing of HDTV*, pages 412–430, 1988.

[DT05] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *Computer Vision and Pattern Recognition*, 1 :886–893, 2005.

[DVZB09] Piali Das, Olga Veksler, Vyacheslav Zavadsky, and Yuri Boykov. Semiautomatic segmentation with compact shape prior. *Image and Vision Computing*, 27 :206–219, 2009.

[FB03] Guillaume Foret and Pascal Bertolino. Label prediction and local segmentation for accurate video object tracking. In *SPIE Visual Communications and Image Processing*, Lugano, Switzerland, 8-11 July 2003.

[FBC02] Guillaume Foret, Pascal Bertolino, and David Cibaud. Partition projection in videos by global and local block-matching. In *IEEE International Conference on Image Processing*, 2002.

[FBC05] G. Foret, P. Bertolino, and J-M. Chassery. Suivi d'objets vidéo par propagation d'étiquettes et rétro-projection. *Traitement du Signal*, 22(1) :41–57, 2005.

[For03] G. Foret. *Segmentation Spatio-Temporelle d'Objets vidéo en vue de leur Caractérisation.* PhD thesis, Institut National Polytechnique de Grenoble, INP, grenoble, 17 Octobre 2003.

[FT+01] N. Fisekovic, T.Nauta, et al. Improved motion-picture quality of AM-LCDs using scanning backlight. *Asia Display /IDW 01*, pages 1637–1640, 2001.

[FZ05] D. Freedman and Tao Zhang. Interactive Graph Cut Based Segmentation with Shape Priors. In *CVPR '05 : Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1*, pages 755–762, Washington, DC, USA, 2005. IEEE Computer Society.

[GG02] D.M. Gavrila and J. Giebel. Shape-based pedestrian detection and tracking. *Intelligent Vehicle Symposium*, 1 :8–14, June 2002.

[GL98a] C. Gu and M.C. Lee. Semantic video object tracking using region-based classification. In *IEEE International Conference on Image Processing*, pages 643–647, Chicago, USA, 1998.

[GL98b] C. Gu and M.C. Lee. Semiautomatic segmentation and tracking of semantic video objects. *IEEE Transactions on Circuits and Systems for Video Technology*, 8(5) :572–584, September 1998.

[GPSG99] D. Gatica-Perez, M.T. Sun, and C. Gu. Semantic video object extraction based on backward tracking of multivalued watershed. In *IEEE International Conference on Image Processing*, Kobe, Japan, 1999.

[Ham02] R. Hammoud. *Construction et présentation des vidéos interactives.* PhD thesis, Institut National Polytechnique de Grenoble, Février 2002.

[HB05a] J. Huart and P. Bertolino. Segmentation pyramidale et groupements perceptuels. In *colloque GRETSI*, Louvain-la-Neuve, Belgique, septembre 2005.

[HB05b] J. Huart and P. Bertolino. Similarity-based and perception-based image segmentation. In *IEEE International Conference on Image Processing*, Genova, Italy, september 2005.

[HFB04a] J. Huart, G. Foret, and P. Bertolino. Extraction d'objets en mouvement par pyrarmide locale. In *Journées CORESA*, Villeneuve d'Ascq, France, 2004.

[HFB04b] J. Huart, G. Foret, and P. Bertolino. Moving object extraction with a localized pyramid. In *International Conference on Pattern Recognition*, Cambridge, UK, august 2004.

[HHD00] I. Haritaoglu, D. Harwood, and L. Davis. W4 : Real-time surveillance of people and their activities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8) :809–830, august 2000.

[HM00] Riad Hammoud and Roger Mohr. Interactive tools for constructing and browsing structures for movie films. In *Proceedings of the 8th ACM International Conference on Multimedia*, Los Angeles, California, USA, November 2000.

[HS01] Javier Ruiz Hidalgo and Philippe Salembier. Robust segmentation and representation of foreground key-regions in video sequences. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2001.

[HW87] N. Hoose and L.G. Willumsen. Automatically extracting traffic data from video-tape using clip4 parallel image processor. *Pattern Recognition Letters*, 6(3) :199–213, August 1987.

[HZ99] Alan Hanjalic and HoongJiang Zhang. An integrated scheme for automated video abstraction based on unsupervised cluster-validity analysis. *IEEE Transactions on Circuits and Systems for Video Technology*, 9(8), 1999.

[IM04] G. Itoh and M. Mishima. Novel frame interpolation method for high image quality LCDs. *Asia Display /IDW 04*, 2004.

[JBBA01] S. Jehan-Besson, M. Barlaud, and G. Aubert. Region-based active contours for video object segmentation with camera compensation. In *IEEE International Conference on Image Processing*, Thessaloniki, Greece, 2001.

[JDWR00] S. Jabri, Z. Duric, H. Wechsler, and A. Rosenfeld. Detection and location of people in video images using adaptive fusion of color and edge information. In *International Conference on Pattern Recognition*, volume 4, pages 627–630, Barcelona, Spain, 2000.

[JGS99] M. Jackowski, A. Goshtasby, and M. Satter. Interactive tools for image segmentation. In *SPIE's International Symposium on Medical Imaging*, San Diego, USA, 1999.

[JLZZ03] F. Jing, M. Li, H. Zhang, and B. Zhang. Unsupervised image segmentation using local homogeneity analysis. In *Proc. IEEE International Symposium on Circuits and Systems*, 2003.

[JM92] J.M. Jolion and A. Montanvert. The adapted pyramid : a framework for 2d image analysis. *Computer Vision Graphics and Image Processing*, 55(3) :339–348, May 1992.

[JM02] H. Jiang and C. Moloney. A new direction adaptive scheme for image interpolation. *IEEE International Conference on Image Processing*, 3 :III–369 – III–372 vol.3, 2002.

[JZDJ98] A.K. Jain, Y. Zhong, and M.P. Dubuisson-Jolly. Deformable template models : A review. *Signal Processing*, 71(2) :109–129, 1998.

[KH99] C. Kim and J. N. Hwang. Fast and robust moving object segmentation in video sequences. In *IEEE International Conference on Image Processing*, volume 2, pages 131–134, october 1999.

[KH00] Changick Kim and Jenq-Neng Hwang. An integrated scheme for object-based video abstraction. In *ACM Multimedia*, pages 303–311, New York, NY, USA, 2000. ACM Press.

[KJ04] M.-A. Klompenhouwer and L.-J. Jeong. Motion blur reduction for liquid crystal displays : Motion compensated inverse filtering. *Proc. SPIE-IS&T Electronic Imaging*, 5308, 2004.

[Koi94] Tero Koivunen. Motion detection of an interlaced video signal. *IEEE Transactions on Consumer Electronics*, 40(3) :753–760, 1994.

[KWT88] M. Kass, A. Witkin, and D. Terzopoulos. Snakes : Active contour models. *Computer Vision Graphics and Image Processing*, pages 321–331, 1988.

[LC97] Chang-Hsing Lee and Ling-Hwei Chen. A fast motion estimation algorithm based on the block sum pyramid. *IEEE Transactions on Image Processing*, 6(11), November 1997.

[LCC98] C.W. Lin, Y.J. Chang, and Y.C. Chen. Hierarchical motion estimation algorithm based on pyramidal successive elimination. In *International Computer Symposium*, 1998.

[LCC03] Shyh-Feng Lin, Yu-Ling Chang, and Liang-Gee Chen. Motion adaptive interpolation with horizontal motion detection for deinterlacing. *IEEE Transactions on Consumer Electronics*, 49(4) :1256–1265, 2003.

[LDDD07] Z. Lin, L.S. Davis, D. Doermann, and D. DeMenthon. Hierarchical part-template matching for human detection and segmentation. *International Conference on Computer Vision*, pages 1–8, October 2007.

[LeG03] Jiebo Luo and Cheng en Guo. Perceptual grouping of segmented regions in color images. *Pattern Recognition*, pages 2781–2792, April 2003.

[LHC+96] R.L. Lagendijk, A. Hanjalic, M.P. Ceccarelli, M. Soletic, and E.H. Persoon. Visual search in a smash system. In *IEEE International Conference on Image Processing*, Lausanne, Switzerland, 1996.

[Lub86] M. Luby. A simple parallel algorithm for the maximal independent set problem. *SIAM Journal of Computing*, 15(4) :1036–1053, November 1986.

[LYKK00] S. Liu, Z. Yan, J. Kim, and C.-C. Jay Kuo. Global/local motion-compensated frame interpolation for low-bit-rate video. *Proceedings of SPIE*, 3974 :223–234, april 2000.

[LZT01] Y. Li, T. Zhang, and D. Tretter. An overview of video abstraction techniques. Technical Report HPL-2001-191, HP Laboratory, July 2001.

[MB95] Eric Mortensen and Wiliam A. Barrett. Intelligent scissors for image composition. In *In Proc. of the ACM SIGGRAPH 95 : Computer Graphics and Interactive Techniques*, pages 191–198, Los Angeles, August 1995.

[MBC10] Cyrille Migniot, Pascal Bertolino, and Jean-Marc Chassery. Contour segment analysis for human silhouette pre-segmentation. In *International Conference on Computer Vision, Theory and Applications*, page CD, Angers France, 05 2010. Département Images et Signal.

[MBC11a] Cyrille Migniot, Pascal Bertolino, and Jean-Marc Chassery. Automatic people segmentation with a template-driven graph cut. In *Proceedings of the 18th IEEE International Conference on Image Processing, ICIP 2011*, page CD, Brussels, Belgique, September 2011. Département Images et Signal.

[MBC11b] Cyrille Migniot, Pascal Bertolino, and Jean-Marc Chassery. Segmentation automatique de personnes par coupe de graphe et gabarits. In *23ème Colloque GRETSI sur le traitement du signal et des images*, page CD, Bordeaux, France, September 2011.

[MBPB02] A. Mahboubi, J. Benois-Pineau, and D. Barba. Tracking of objects in video scenes with time varying content. *EURASIP Journal of Applied Signal Processing, Special issue on Image Analysis for Multimedia Interactive Services*, 2002(6) :582–594, June 2002.

[MC89] P. Meer and S. Connelly. A fast parallel method for synthesis of random patterns. *Pattern Recognition*, 22 :189–204, 1989.

[ML98] F. Marqués and J. Llach. Tracking of generic objects for video object generation. In *IEEE International Conference on Image Processing*, pages 628–632, Chicago, USA, 1998.

[MLD07] N. Mueller, Yue Lu, and Minh N. Do. Image interpolation using multiscale geometric representations. *Computational Imaging*, 6498 :64980A, 2007.

[MLT99] G. Medioni, M.-S. Lee, and C.-K. Tang. *A computational framework for segmentation and grouping*. Elsevier Science, Amsterdam, The Netherlands, 1999.

[MM97] F. Marqués and C. Molina. Object tracking for content-based functionalities. In *SPIE Visual Communications and Image Processing*, volume 3024, pages 190–199, San Jose, USA, 1997.

[MMR89] A. Montanvert, P. Meer, and A. Rosenfeld. Hierarchical image analysis using irregular tessellations. Technical Report CS TR 2322, Computer Vision Laboratory, University of Maryland, September 1989.

[MMR91] A. Montanvert, P. Meer, and A. Rosenfeld. Hierarchical image analysis using irregular tessellations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(4) :307–316, April 1991.

[MRT07] J. Malcolm, Y. Rathi, and A. Tannenbaum. Graph cut segmentation with nonlinear shape priors. In *IEEE International Conference on Image Processing*, october 2007.

[Mur05] D. Muresan. Fast edge directed polynomial interpolation. *IEEE International Conference on Image Processing*, 2 :II– 990–993, 2005.

[MY09] S. Mallat and G. Yu. Super-resolution with sparse mixing estimators. *IEEE Transactions on Image Processing*, 19 :2889 2900, 2009.

[NSS+01] T. Nose, M. Suzuki, D. Saski, M. Imai, and H. Hayama. A black stripe driving scheme for displaying motion pictures on LCDs. *SID 2001 Digest*, 32 :994–997, 2001.

[O+92] H. Okumura et al. A new low-image-lag drive method for large-size lctvs. *Proc. SIS*, pages 601–604, 1992.

[OHL00] J. Oh, K. A. Hua, and N. Liang. A content-based scene change detection and classification technique using background tracking. In *Proceedings of SPIE : Multimedia Computing and Networking 2000*, pages 254–265, January 2000.

[OLV03] J. Oh, J. Lee, and E. Vemuri. An efficient technique for segmentation of key object(s) from video shots. In *ITCC '03 : Proceedings of the International Conference on Information Technology : Computers and Communications*, page 384, Washington, DC, USA, 2003. IEEE Computer Society.

[ORGLSLV07] Victor Osma-Ruiz, Juan Ignacio Godino-Llorente, Nicolás Sáenz-Lechón, and Pedro Gómez Vilda. An improved watershed algorithm based on efficient computation of shortest paths. *Pattern Recognition*, 40(3) :1078–1090, 2007.

[Pat00] S. Pateux. Tracking of video objects using a backward projection technique. In *SPIE Visual Communications and Image Processing*, volume 4067, pages 1107–1114, Perth, Australia, 2000.

[PFG08] Sylvie Philipp-Foliguet and Laurent Guigues. Multi-scale criteria for the evaluation of image segmentation algorithms. *Journal of Multimedia*, 3(5) :42–56, 2008.

[PL97] S. Pateux and C. Labit. Codage efficace de carte de segmentation pour la compression orientee regions de sequences d'images. Technical Report PI 1073, IRISA, 1997.

[PM05] G. Peyre and S. Mallat. Discrete bandelets with geometric orthogonal filters. *IEEE International Conference on Image Processing*, 1 :I – 65–8, sep. 2005.

[PS94] M. Pardàs and P. Salembier. *Joint Region and Motion Estimation with Morphological Tools*, pages 93–100. In J. Serra and P. Soille, editors, Mathematical Morphology and Its Applications to Image Processing. Kluwer Academic Press, 1994.

[PTS98] A. J. Patti, A. Murat Tekalp, and M. Ibrahim Sezan. A new motion-compensated reduced-order model kalman filter for space-varying restoration of progressive and interlaced video. *IEEE Transactions on Image Processing*, 7(4) :543–554, 1998.

[PYW00] D.K. Park, H.S. Yoon, and C.S. Won. Fast object tracking in digital video. *IEEE Transactions on Consumer Electronics*, 46(3) :785–790, August 2000.

[RB08] Jérôme Roussel and Pascal Bertolino. Graph-based deinterlacing. In *IEEE International Conference on Image Processing*, page CD, San Diego, California États-Unis, 10 2008. Département Images et Signal.

[RBN06] J. Roussel, P. Bertolino, and M. Nicolas. Improvement of conventional deinterlacing methods with extrema detection and interpolation. In *Advanced Concepts for Intelligent Vision Systems*, Antwerp, Belgium, september 2006.

[RBN07] Jérôme Roussel, Pascal Bertolino, and Marina Nicolas. Détection et reconstruction des éléments hautes fréquences appliquées au désentrelacement. In *colloque GRETSI*, 11-14 Septembre 2007, Troyes, France, pages –, Troyes France, 09 2007. Département Images et Signal.

[RBN08] J. Roussel, P. Bertolino, and M. Nicolas. Image deinterlacing, u.s.pat. no 20080089614, 2008.

[RE95] P.L. Rosin and T. Ellis. Image difference threshold strategies and shadow detection. In *6th British Machine Vision Conference*, pages 347–356, Birmingham, England, 1995.

[Rev91] Jean-Pierre Reveilles. *Geometrie discrete, Calcul en nombres entiers et algorithmique.* PhD thesis, Universite Louis Pasteur, Strasbourg, France, 1991.

[RKB04] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. "grabcut" : interactive foreground extraction using iterated graph cuts. *ACM Trans. Graph.*, 23(3) :309–314, 2004.

[RS07] M.D. Rodriguez and M. Shah. Detecting and segmenting humans in crowded scenes. In *International Conference on Multimedia*, pages 353–356, 2007.

[SF05] X. Song and G. Fan. Key-frame extraction for object-based video segmentation. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, Philadelphia, PA, March 2005.

[SFM07] J.L. Starck, J. Fadili, and F. Murtagh. The undecimated wavelet decomposition and its reconstruction. *ICIP*, 16(2) :297–309, Feb. 2007.

[SG99] C. Stauffer and W.E.L Grimson. Adaptative background mixture models for real-time tracking. In *Proc. Computer Vision and Pattern Recognition Conf.*, 1999.

[Sma92] J. Smart. wxwidgets, cross-platform gui programming in c++, 1992.

[SN99] Kenji Sugiyama and Hiroya Nakamura. A method of de-interlacing with motion compensated interpolation. *IEEE Transactions on Consumer Electronics*, 45(3) :611–616, 1999.

[SU05] G.G. Slabaugh and G. Unal. Graph Cuts Segmentation Using an Elliptical Shape Prior. In *IEEE International Conference on Image Processing*, volume 2, pages 1222–1225, 2005.

[TAT97] Yukinobu Taniguchi, Akihito Akutsu, and Yoshinobu Tonomura. Panoramaexcerpts : extracting and packing panoramas for video browsing. In *MULTIMEDIA '97 : Proceedings of the fifth ACM international conference on Multimedia*, pages 427–436, New York, NY, USA, 1997. ACM Press.

[TFR10] David Tsai, Matthew Flagg, and James Rehg. Motion coherent tracking with multi-label mrf optimization. In *Proc. BMVC*, pages 56.1–11, 2010. doi :10.5244/C.24.56.

[uP92] Cambridge university Press. Numerical recipes in c : The art of scientific computing. Website, 1992.

[Vid05] Video Electronics Standards Association FPDM Task Group. *Flat Panel Display Measurements Standards v2.0*, May 2005.

[VL98] N. Vasconcelos and A. Lippman. A spatiotemporal motion model for video summarization. In *Computer Vision and Pattern Recognition*, page 361, Washington, DC, USA, 1998. IEEE Computer Society.

[VMBP96] P. Vannoorenberghe, C. Motamed, J-M. Blosseville, and J-G Postaire. Motion detection for non-rigid objects. application to pedestrians monitoring in urban environment. In *IEEE International Conference on IMACS*, Lille, France, July 9-12 1996.

[VR11] E. Van Reeth. *Systeme avancé d'interpolation spatiale de signaux de télévision pour affichage sur écrans haute-définition.* PhD thesis, Université de Grenoble, 2011.

[Wan98] D. Wang. Unsupervised video segmentation based on watersheds and temporal tracking. *IEEE Transactions on Circuits and Systems for Video Technology*, 8(5) :539–546, September 1998.

[WBPB95] L. Wu, J. Benois-Pineau, and D. Barba. Spatio-temporal segmentation of image sequences for object-oriented low bit-rate image coding. In *IEEE International Conference on Image Processing*, volume 2, pages 2406–2409, Washington DC, 1995.

[Wen83] O.S. Wenstop. Motion detection for image information . In 3^{rd} *Scandinavian Conference on Image Analysis*, pages 381–386, Tromso, Norway, July 1983.

[Wer58] M. Wertheimer. Principles of perceptual organization. In *Readings in Perception*, pages 115–135, 1958.

[Wol96] W. Wolf. Key frame selection by motion analysis. In *Proc. IEEE Internat. Conf. on Acoustic, Speech Signal Process. (ICASSP'96)*, pages 1228–1231, 1996.

[WW07] Q. Wang and R.K. Ward. A new orientation-adaptive interpolation method. *IEEE International Conference on Image Processing*, 16(4) :889 –900, april 2007.

[WZ10] Maddy Hui Wang and Hong Zhang. Adaptive shape prior in graph cut segmentation. In *ICIP*, pages 3029–3032, 2010.

[XLD11] F. Xu, K.-M. Lam, and Q. Dai. Video-object segmentation and 3d-trajectory estimation for monocular video sequences. *Image and Vision Computing*, 29 :190–205, 2011.

[YJ02] Hoon Yoo and Jechang Jeong. Direction-oriented interpolation and its application to de-interlacing. *IEEE Transactions on Consumer Electronics*, 48 :954–962, 2002.

[ZD05] L. Zhao and L.S. Davis. Closely coupled object detection and segmentation. In *International Conference on Computer Vision*, volume 1, pages 454–461. IEEE, 2005.

[ZMM99] F. Zanoguera, B. Marcotegui, and F. Meyer. A toolbox for interactive segmentation based on nested partitions. In *IEEE International Conference on Image Processing*, Kobe, Japan, 1999.

[ZTB04] N. Zlatoff, B. Tellez, and A. Bazkurt. Image understanding and scene models : a generic framework integrating domain knowledge and gestalt theory. In *IEEE International Conference on Image Processing*, Singapore, October 24-27 2004.

[ZTB05] N. Zlatoff, B. Tellez, and A. Baskurt. Groupement perceptuel pour la reconnaissance d'objets. In *CORESA*, Rennes, France, novembre 2005.