N° d'ordre : 4096 ANNÉE 2010

UNIVERSITÉ DE RENNES 1

THÈSE / UNIVERSITÉ DE RENNES 1
sous le sceau de l'Université Européenne de Bretagne

pour le grade de

DOCTEUR DE L'UNIVERSITÉ DE RENNES 1

Mention : Traitement du Signal et Télécommunications

Ecole doctorale MATISSE

présentée par

Matthieu Fradet

préparée à l'unité de recherche UMR6074 IRISA
Institut de Recherche en Informatique et Systèmes Aléatoires
Composante Universitaire : SPM

Contributions à la segmentation de séquences d'images au sens du mouvement dans un contexte semi-automatique

Thèse soutenue à Rennes le 22 janvier 2010

devant le jury composé de :

Patrick BOUTHEMY
Directeur de Recherche, INRIA / président

Jenny BENOIS-PINEAU
Professeur des Universités, Bordeaux 1 / rapporteur

Renaud KERIVEN
Professeur, École des Ponts ParisTech / rapporteur

Anil KOKARAM
Professeur, Trinity College Dublin / examinateur

Patrick PÉREZ
Directeur de Recherche, INRIA / directeur de thèse

Philippe ROBERT
Ingénieur de Recherche, Thomson R&D / co-directeur de thèse

Remerciements

Ces travaux ont été réalisés dans un environnement en partie industriel au sein du laboratoire « Signal Acquisition & Processing » de Thomson R&D France, et en partie académique au sein de l'équipe Vista de l'INRIA Rennes-Bretagne Atlantique.

Je tiens tout d'abord à exprimer ma sincère reconnaissance à Patrick Pérez et Philippe Robert, pour la confiance qu'ils m'ont accordée lorsqu'ils ont accepté de diriger ces travaux de recherche. La qualité de leur encadrement m'a permis d'effectuer une thèse enrichissante et motivante. Ce fut réellement une chance et un plaisir de travailler avec eux.

Je remercie également Jenny Benois-Pineau et Renaud Keriven qui ont accepté d'être mes rapporteurs en dépit du travail important que cela représente, ainsi que Patrick Bouthemy et Anil Kokaram qui ont accepté d'examiner les travaux et faire partie du jury de thèse.

Enfin je voudrais simplement saluer collègues, amis et famille qui m'ont entouré et soutenu durant ces trois années.

Table des matières

Notations

Nous présentons ici les notations utilisées tout au long du document.

Fonctions et opérateurs

$\nabla\cdot$	gradient d'une fonction
$\|\cdot\|$	cardinal d'un ensemble
$T(\cdot)$	fonction caractéristique valant 1 si l'argument est vrai, 0 sinon

Notations de la première partie

$\mathcal{P}$	ensemble des pixels à segmenter
$\mathbf{p} = (x, y)^T$	un pixel de $\mathcal{P}$
$\mathbf{dp} = (dx, dy)^T$	vecteur de mouvement associé au pixel $\mathbf{p}$
I_t	image à l'instant t d'une séquence d'images
f_t	carte de segmentation de l'image I_t
f'_t	carte de segmentation prédite de l'image I_t
$f_\mathbf{p} = f_t(\mathbf{p})$	étiquette du pixel $\mathbf{p}$ à l'instant t
$f'_\mathbf{p} = f'_t(\mathbf{p})$	étiquette prédite du pixel $\mathbf{p}$ à l'instant t
$\mathcal{A}_i$	modèle affine (et fonction) de mouvement estimé pour la couche i de l'image I_t vers l'image I_{t+1}
$M_{a,i}$	mosaïque de la couche i associée à l'instant de référence t_a
$\mathcal{A}_{a,i}$	modèle affine (et fonction) de mouvement estimé pour la couche i de l'image I_t vers l'image de référence I_{t_a}
$\mathcal{G}$	graphe
$\mathcal{V}$	ensemble des nœuds d'un graphe
$\mathcal{E}$	ensemble des arcs d'un graphe
$\mathcal{C}$	ensemble des cliques
$\mathcal{N}$	ensemble des paires de pixels voisins

Notations de la deuxième partie

$\mathbf{f}^{(i)}$	trajectoire du i-ème point d'intérêt
$\mathcal{F}_t^{(j)}$	modèle affine (et fonction) du mouvement « avant » du cluster j de l'instant t à l'instant $t+1$
$\mathcal{B}_t^{(j)}$	modèle affine (et fonction) du mouvement « arrière » du cluster j de l'instant $t+1$ à l'instant t
$\mathcal{M}^{(j)}$	modèle de trajectoire affine du cluster j

Introduction générale

L'analyse du mouvement est un problème fondamental du traitement des séquences d'images. Il s'agit en particulier d'un problème central pour des domaines tels que la post-production, la compression vidéo, la vidéo surveillance, la reconnaissance d'actions, l'analyse sémantique de scènes...

L'étude de l'information de mouvement permet de définir des critères suffisamment génériques pour qu'une large variété de situations puisse être appréhendée. L'objectif est souvent double : on souhaite premièrement disposer d'une caractérisation globale du contenu dynamique de la scène, deuxièmement extraire et décrire des régions d'intérêt à partir de critères de mouvement.

L'analyse du mouvement dans une séquence d'images est un vaste sujet qui regroupe plusieurs problématiques. On peut notamment citer l'estimation du champ de mouvement inter-images généralement assimilée à celle du flot optique, la détection et la segmentation de chaque région d'une image ayant un mouvement distinct des autres, le suivi de points ou de régions d'intérêt tout au long de la séquence.

L'idée selon laquelle une scène 3D est constituée d'un ensemble d'objets, situés à différentes profondeurs, est à relier directement à la notion de calques désormais largement répandue dans les logiciels de traitement, de retouche et de manipulation d'images. Une image, objet à 2 dimensions, résulte de la projection des différents objets présents dans une scène 3D. Ces différents objets sont les couches de la scène. Si l'on fait l'hypothèse selon laquelle le mouvement de chaque couche de la scène dans le plan image est bien décrit par un modèle paramétrique, le problème de la segmentation d'une séquence d'images au sens du mouvement consiste alors à estimer les paramètres de mouvement de chaque couche et à extraire les supports des couches.

Les informations de mouvement, lorsque celui-ci est estimé entre deux images consécutives, ne permettent pas toujours la distinction de certains mouvements de faible amplitude. Si l'observation de deux images distantes facilite cette distinction, il faut tout de même proposer une solution au traitement des images intermédiaires, qui pour certaines applications doivent être obligatoirement segmentées elles-aussi. Tenir compte simultanément d'un groupe d'images consécutives permet de travailler avec des informations de mouvement enrichies vraisemblablement plus faciles à distinguer les unes des autres que lorsqu'elles sont issues de l'estimation du mouvement entre deux images uniquement. Parmi les méthodes qui reposent sur un groupe d'images, celles les plus avancées segmentent simultanément toutes les images constituant ce groupe. Contrairement au cas où les images sont segmentées une par une, la cohérence temporelle de la

segmentation peut être encouragée simultanément sur tout le groupe d'images et s'en trouve donc nettement améliorée. Néanmoins ceci a un coût. Et au delà de l'important temps de calcul nécessaire à la résolution du problème de segmentation simultanée d'un groupe d'images, les approches dites simultanées ou par lots sont directement limitées quant au nombre réel d'images pouvant être considérées en même temps, simplement pour des raisons d'occupation d'espace mémoire.

Ces constatations ont constitué le point de départ de nos travaux.

Contexte

Cette thèse s'est déroulée dans un environnement en partie industriel, en partie académique. Elle a débuté dans le cadre d'un projet lié au développement d'une plate-forme de post-production au sein de la société Thomson. Dans le contexte de la post-production cinématographique, l'objectif est de proposer de nouveaux algorithmes qui permettent aux coloristes de réduire le temps passé à des tâches fastidieuses et répétitives et de se consacrer pleinement à la partie artistique de leur travail. Dans le contexte de la post-production d'informations télévisées, l'objectif est de fournir aux journalistes de nouveaux outils qui leur permettent d'accroître la qualité des vidéos éditées tout en respectant des contraintes de temps imposées par leur cadre de travail. Plus généralement, le but est de fournir à un opérateur des outils qui améliorent la qualité et l'efficacité de sa production.

Dans le domaine applicatif de la post-production cinématographique, les exigences de qualité sont si importantes qu'il est bien souvent préférable que l'opérateur puisse intervenir pour fournir des indications précieuses sur lesquelles reposeront les traitements. Ces interventions se font au détriment de l'automatisme du système mais contribuent largement à la qualité des résultats. Idéalement, elles doivent être aussi simples et rapides que possible tout en engendrant une forte valeur ajoutée.

Notre étude aborde le problème de l'analyse spatio-temporelle des séquences d'images, en particulier la segmentation d'une séquence d'images au sens du mouvement. L'objectif principal de la thèse est donc de proposer et développer de nouveaux outils performants pour la segmentation de séquences d'images au sens du mouvement. Ces outils doivent permettre une coopération efficace de l'opérateur et de l'algorithmie en termes de précision des résultats et de temps d'exécution. Le cadre applicatif principalement visé est l'interpolation spatio-temporelle de régions issues du pistage et de la suppression d'objets. Les travaux correspondants sont présentés dans la première partie de ce document.

Le contexte industriel dans lequel nous avons réalisé notre étude a fortement évolué au cours de la thèse. Les derniers travaux ont été réalisés dans un contexte légèrement différent où les aspects interactifs sont beaucoup moins prononcés. Ils sont présentés dans la deuxième partie de ce document.

Première problématique

Étant donné le contexte présenté ci-dessus, la qualité des résultats prime sur la rapidité des traitements. Puisque le temps de calcul n'est pas notre contrainte première, une approche simultanée telle qu'évoquée au début de cette introduction pourrait sembler appropriée même si nous savons d'ores et déjà que toute la séquence ne pourra pas être considérée en une seule fois. Ceci étant dit, dans notre contexte de post-production, un opérateur averti sait qu'il n'existe pas de système infaillible entièrement automatique. Il sait pertinemment qu'il devra au mieux simplement valider les résultats, très souvent les retoucher légèrement, au pire les ignorer totalement et relancer l'intégralité des traitements en modifiant quelques paramètres sans pour autant être assuré d'une amélioration. Ce dernier cas doit être évité autant que possible. Il correspond à une totale improductivité et empêche un coloriste de répondre rapidement et efficacement aux demandes interactives du directeur de la photographie. À ce titre, une approche simultanée ne semble pas souhaitable dans la mesure où l'opérateur n'a accès à aucun résultat intermédiaire avant la fin du traitement de tout le groupe d'images considéré. Un système dont l'utilisation est confortable pour un opérateur doit proposer à celui-ci des solutions pour par exemple :

- surveiller, vérifier la bonne progression des traitements,
- interrompre si nécessaire les traitements afin d'introduire de nouvelles indications qui doivent évidemment être prises en compte pour la suite,
- lancer une deuxième passe de traitements qui profitera de la validation partielle des résultats de la première passe.

Les deux premiers points plaident clairement en faveur d'une approche séquentielle. Nous nous sommes donc efforcés de proposer et développer de nouveaux algorithmes séquentiels pour la segmentation semi-automatique de séquences d'images au sens du mouvement. Comment assurer alors efficacement une cohérence spatio-temporelle de la segmentation sans pouvoir exploiter pleinement les contraintes temporelles proposées dans les approches simultanées ? Comment traduire simplement les connaissances de l'opérateur relatives à la séquence traitée par des interactions rapides, peu nombreuses et intuitives ?

Seconde problématique

Nous venons de justifier notre choix en faveur des approches séquentielles, mais nous n'avons pas pour autant définitivement écarté l'idée des approches simultanées. Si des limites techniques (d'occupation d'espace mémoire notamment) nous empêchent de considérer simultanément tous les pixels de quelques dizaines d'images, plutôt que de se résigner à ne traiter qu'un petit groupe d'images, il est envisageable de considérer toute la longueur de la séquence et de sous-échantillonner spatialement l'ensemble des pixels du volume vidéo. Nous nous sommes donc demandé comment effectuer un tel sous-échantillonnage de manière judicieuse. Les données restantes doivent être compatibles avec une modélisation de contraintes temporelles significatives.

La notion de trajectoire de point répond tout à fait à la question précédente. Le sous-échantillonnage consiste alors à détecter des points d'intérêts dans différentes images et à les suivre au cours de la séquence afin de constituer un ensemble épars de trajectoires. Le problème précédent de segmentation « dense » où tout pixel du volume vidéo devait être assigné à une couche de mouvement est alors reformulé en un problème de partitionnement d'un ensemble de trajectoires de points. Selon les changements de pose et/ou d'apparence des objets qui composent la scène, chaque point est visible sur un ensemble d'images qui lui est propre. Ainsi, chaque trajectoire est définie sur un intervalle temporel qui lui est propre. Si l'on trouve une littérature abondante sur le problème de clustering de trajectoires ayant toutes la même longueur, les quelques méthodes existantes qui s'intéressent au cas des trajectoires incomplètes, c'est-à-dire dont la longueur est différente de celle de la séquence, proposent souvent de compléter ces trajectoires par une étape d'extrapolation précédant le clustering. Cette extrapolation nous semblant souvent risquée, nous nous sommes concentrés sur la problématique suivante : proposer un algorithme pour résoudre efficacement le problème spécifique du clustering d'un ensemble de trajectoires de points de différentes longueurs sans extrapolation des trajectoires les plus courtes.

Organisation du document

Ce document de thèse décrit tour à tour ces différents aspects de nos travaux. Il comporte deux parties dont le problème commun est la distinction de mouvements différents dans une séquence d'images.

La première partie s'intéresse à la segmentation d'une séquence d'images au sens du mouvement. Cette segmentation sera parfois dite « dense » par opposition au clustering d'un ensemble « épars » de trajectoires de points traité dans la deuxième partie.

La première partie du document présente deux nouvelles approches séquentielles que nous avons proposées et développées pour la segmentation d'une séquence d'images au sens du mouvement. Toutes les deux sont semi-automatiques et font appel à des estimateurs de mouvement et au formalisme de coupe minimale/flot maximal dans un graphe pour la minimisation d'une fonctionnelle d'énergie. La première repose sur des triplets d'images consécutives. La seconde est basée sur des instants de référence fixés encadrant différents intervalles temporels. À chaque instant de référence est associé un ensemble de mosaïques de couches de mouvement générées au fil de la segmentation par accumulation des données considérées comme appartenant à une même couche.

La seconde partie du document présente un nouvel algorithme pour le clustering d'un ensemble de trajectoires de points de différentes longueurs. Cet algorithme ne repose sur aucune hypothèse de continuité ou périodicité du mouvement et ne nécessite aucune extrapolation de trajectoire.

La conclusion générale présente une synthèse de notre étude et quelques perspectives liées à ces travaux.

Première partie

Segmentation d'une séquence d'images au sens du mouvement

Introduction de la première partie

La première partie de ce document traite de la segmentation de séquences d'images au sens du mouvement. Elle se décompose en cinq chapitres de la manière suivante.

Le chapitre 1 présente un état de l'art des méthodes de segmentation de séquences d'images au sens du mouvement. Nous différencions plusieurs catégories parmi ces méthodes : celles qui reposent sur le mouvement estimé entre deux images consécutives, celles qui reposent sur le mouvement estimé entre deux images distantes, celles qui travaillent sur un groupe d'images. Ces différentes techniques ont leurs propres avantages et leurs limites. Après avoir présenté la problématique autour de laquelle s'articulent nos travaux, le principe des deux approches semi-automatiques proposées est détaillé. Dans ces deux méthodes, on retrouve les tâches communes d'estimation du mouvement et de segmentation au sens du mouvement.

Le chapitre 2 présente la théorie de coupe minimale/flot maximal dans un graphe. Bon nombre de problèmes de vision par ordinateur (dont celui auquel nous nous intéressons) peuvent être formulés comme des problèmes de minimisation d'une fonctionnelle d'énergie. Le principe est alors de ramener le problème de minimisation d'énergie à un problème de « coupe minimale » dans un graphe, lui-même équivalent à un problème de « maximisation de flot ». Nous distinguons le cas d'une classification binaire du cas d'une classification multi-étiquettes. En fin de chapitre nous étudions l'application de segmentation interactive d'une image fixe.

Le chapitre 3 présente différentes techniques d'estimation locale du mouvement. Nous distinguons les approches fréquentielles, les approches de mises en correspondance de blocs et les approches différentielles. Nous présentons ensuite l'intérêt d'utiliser les modèles paramétriques pour décrire le mouvement des objets vidéo.

Le chapitre 4 détaille les différents critères qui composent la fonctionnelle d'énergie utilisée dans chacun de nos deux schémas. Pour notre premier schéma, nos contributions concernent l'introduction d'une nouvelle contrainte dite « de rigidité » et la restriction de la région à segmenter. Dans notre second schéma, nous avons recours à des mosaïques de couches générées automatiquement alors que le processus de segmentation progresse.

Le chapitre 5 montre les résultats de nos expérimentations et illustre par deux exemples la possible réutilisation des mosaïques de couches de mouvement pour des applications liées à la manipulation d'objet vidéo. Les différents types d'interactions sont listés avant qu'une discussion ne soit lancée sur différents points.

Une rapide conclusion vient clore cette première partie. Elle synthétise notre étude, souligne nos contributions, et donne de premières perspectives propres à cette partie.

Chapitre 1

État de l'art des méthodes de segmentation basée sur le mouvement

Ce premier chapitre a pour but de dresser un état de l'art des différentes techniques de segmentation basée sur le mouvement. La littérature sur le sujet étant très abondante, nous ne cherchons pas à lister toutes les méthodes existantes. L'idée est de donner un aperçu des méthodes les plus couramment utilisées et de distinguer différents types d'approches.

Nous commençons par présenter très brièvement le problème et expliquer en quoi il constitue un sujet d'intérêt. Puis nous définissons les termes de « couche de mouvement » et de « mosaïque de couche » qui seront largement utilisés dans toute la suite du document. Une certaine ambiguïté existant dans certains articles que nous citons, débuter par ces définitions nous paraît nécessaire pour éviter toute confusion par la suite.

Nous présentons ensuite quelques-unes des méthodes existantes. Nous séparons ces méthodes en différentes catégories, selon le nombre d'images utilisées simultanément et selon les distances temporelles séparant ces images. Il s'avère que cette classification est relativement proche d'une organisation chronologique de la littérature. Nous distinguons ainsi les méthodes travaillant sur une paire d'images consécutives, celles travaillant sur une paire d'images éloignées, et celles travaillant sur un groupe d'images consécutives.

Il nous paraît justifié de présenter dans une section indépendante (section. 1.5) les travaux récents de Dupont *et al.* [Dupont 05, Dupont 06b, Dupont 06c]. D'une part, parce que nos premiers travaux se basent sur les leurs. D'autre part, parce que le cheminement de ces travaux met en évidence la problématique que nous présentons après l'état de l'art.

Ce premier chapitre se termine par une brève introduction et description des deux premiers de nos trois nouveaux schémas de segmentation.

1.1 Présentation du problème et de son intérêt

1.1.1 Segmentation basée sur le mouvement

Segmenter une séquence d'images en régions cohérentes au sens du mouvement n'est pas un problème nouveau, mais reste ouvert encore aujourd'hui. De nombreuses équipes de recherche en vision par ordinateur se montrent très actives sur le sujet. L'extraction de couches de mouvement a de nombreuses applications telles que la compression de vidéos, la manipulation d'objets vidéo, la génération de mosaïques, l'analyse sémantique de scènes, etc.

Les couches de mouvement obtenues peuvent être utilisées lors de l'estimation du flot optique, afin d'améliorer la précision des vecteurs de mouvement estimés dans les régions faiblement texturées et dans les régions d'occultations qui posent habituellement problème.

Les couches extraites peuvent également être utilisées pour des tâches avancées d'édition vidéo. Dans le domaine de la post-production, il est fréquent de vouloir effectuer un traitement particulier sur un des objets extraits (suppression, coloration, remplacement de l'arrière-plan).

Ces exemples d'applications montrent l'importance de cette tâche d'analyse bas-niveau de séquences. Par ailleurs, la demande d'une segmentation toujours plus précise, notamment pour les applications liées à la post-production, explique l'évolution des types d'approches qui prennent en compte du plus en plus de données simultanément.

Le problème de la segmentation d'une séquence en régions cohérentes au sens du mouvement peut se résumer de la façon suivante. Sous l'hypothèse standard qu'une scène dynamique est composée d'un ensemble de couches dont le mouvement dans le plan image peut être décrit de manière assez précise par un modèle (paramétrique ou non), le problème de la segmentation basée sur le mouvement consiste à estimer les modèles de mouvement de chacune des couches et à extraire les supports des couches. S'il n'est pas fourni, le nombre de couches à extraire doit également être déterminé.

On est donc face à un cercle vicieux puisqu'il faut réussir à extraire des régions cohérentes au sens du mouvement alors que les modèles de mouvement ne peuvent être estimés de manière précise qu'en connaissant les supports des régions. De nombreuses méthodes font appel à un algorithme itératif comprenant deux étapes alternées : l'estimation des modèles de mouvement pour une partition donnée, et la mise à jour de cette partition connaissant les nouveaux modèles de mouvement. L'algorithme est généralement initialisé par une partition régulière fournissant un grand nombre de couches. Au fil des itérations, le nombre de modèles et donc celui de couches se réduisent en fusionnant deux couches dont les modèles de mouvement sont similaires. Le traitement est itéré jusqu'à ce qu'un nombre de pixels jugé non significatif change encore d'état. La figure 1.3 illustre ce processus itératif.

1.1.2 Définitions

Dans les années 90, la norme MPEG-4 (ISO/IEC 14496) a introduit la notion d'objet vidéo. L'idée est de structurer les données vidéo d'une séquence d'images en un ensemble d'objets possédant un ordre de profondeur relatif. Le codeur MPEG-4 prend en paramètres :

- un masque par image et par objet vidéo,
- une texture par image et par objet vidéo,
- un ordre de profondeur par objet vidéo.

Nous proposons la définition suivante :

Définition 1. *Un « objet vidéo » ou encore une « couche de mouvement », ou simplement une « couche », désigne une séquence d'ensembles (de forme arbitraire) de pixels d'une vidéo, ayant éventuellement une signification sémantique, et partageant des propriétés communes de couleurs, de mouvement et de profondeur relative. Le terme de « couche » sera également utilisé par la suite pour désigner simplement une étiquette de segmentation ou encore un indice de couche relatif à l'ordre de profondeur.*

Dans cette définition, les cohérences de couleurs, de mouvement et de profondeur relative s'entendent à court terme, généralement entre images consécutives. Dans la littérature, il est fréquent que le terme de « couche » soit directement associé à une série temporelle de masques binaires, c'est-à-dire à une séquence de segments d'images, et de modèles de mouvements estimés entre images consécutives.

Néanmoins, dans l'hypothèse où la distribution de couleurs d'un même objet vidéo reste stable pour plusieurs images, et dans l'hypothèse où la cohérence de mouvement peut être définie à plus long terme, il est fort intéressant de proposer une représentation simple et compacte d'une séquence d'images.

À des fins de codage, Wang et Adelson [Wang 94a] proposent une nouvelle représentation en couches qui s'appuie sur la notion d'objet vidéo. De plus, il est supposé que la texture d'un objet sur un groupe d'images peut être représentée sur un support 2D. Dans une image donnée, un objet peut être totalement ou partiellement occulté par un autre objet. La texture d'un objet est reconstituée en assemblant sur un support 2D ses régions visibles dans les images successives.

Nous définissons la notion de « mosaïque de couche de mouvement » comme suit :

Définition 2. *Une « mosaïque de couche de mouvement », ou simplement une « mosaïque », désigne une carte de texture correspondant à une représentation 2D planaire (complète ou non*[1]*) issue du regroupement des différents éléments identifiés au cours du temps comme partageant le mouvement propre de la couche considérée.*

Dans nos travaux, une mosaïque est associée à un instant de référence, ce qui signifie que l'image originale correspondant à cet instant de référence peut être obtenue par

1. On emploie le terme « mosaïque » pour désigner la carte de texture aussi bien dans son état final (complet) que dans ses états intermédiaires (incomplets). Dans notre schéma reposant sur les mosaïques, celles-ci sont construites progressivement en alternant une étape d'enrichissement des mosaïques avec une étape de segmentation de l'image courante (chapitre 4).

superposition directe des différentes mosaïques avec respect de l'ordre de profondeur, puis rognage, et ce, sans nécessiter la moindre déformation.

Toute image de la séquence peut alors être reconstituée à partir des mosaïques de chaque couche et des mouvement associés qui décrivent les transformations vers l'image considérée.

Les figures 1.1 et 1.2 illustrent les deux notions de « couche de mouvement » et de « mosaïque de couche » sur la célèbre séquence *Flowergarden*.

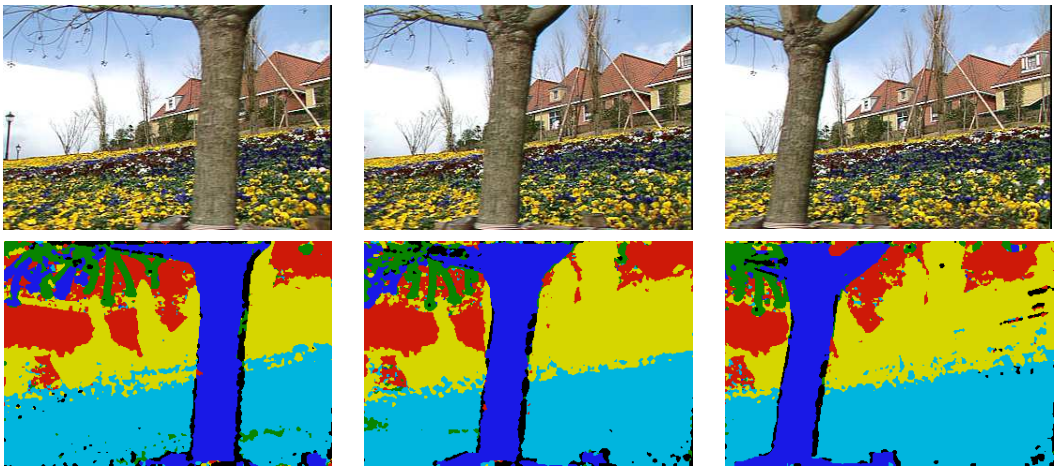


FIGURE 1.1 – Exemple de segmentation en 5 couches de mouvement. Première ligne : images originales correspondant aux instants 0, 14 et 29 de la séquence *Flowergarden*. Seconde ligne : cartes de segmentation correspondantes obtenues par Wang et Adelson. Cette figure provient de [Wang 94b].

Les sections qui suivent donnent un aperçu des diverses approches de segmentation basée sur le mouvement et aident à comprendre le cheminement de la recherche sur le sujet.

1.2 Méthodes séquentielles travaillant sur une paire d'images consécutives

Cet ensemble de méthodes contient notamment des approches qui consistent à segmenter un champ dense de mouvement estimé au préalable [Wang 94a, Ayer 95, Pedersini 96] ; des approches qui, visant à obtenir une estimation plus précise du flot optique, proposent de travailler simultanément à l'estimation et la segmentation d'un champ dense de mouvement [Black 96a, Black 96b, Ju 96, Mémin 02] ; ainsi que des approches qui cherchent à obtenir de manière directe la segmentation sans avoir à estimer explicitement le flot optique [Irani 92, Odobez 98, Cremers 05].

Lorsqu'une de ces méthodes est utilisée pour segmenter toute une séquence d'images, on imagine aisément que si chaque paire d'images consécutives est traitée de manière totalement indépendante, la cohérence temporelle entre les segmentations des différents

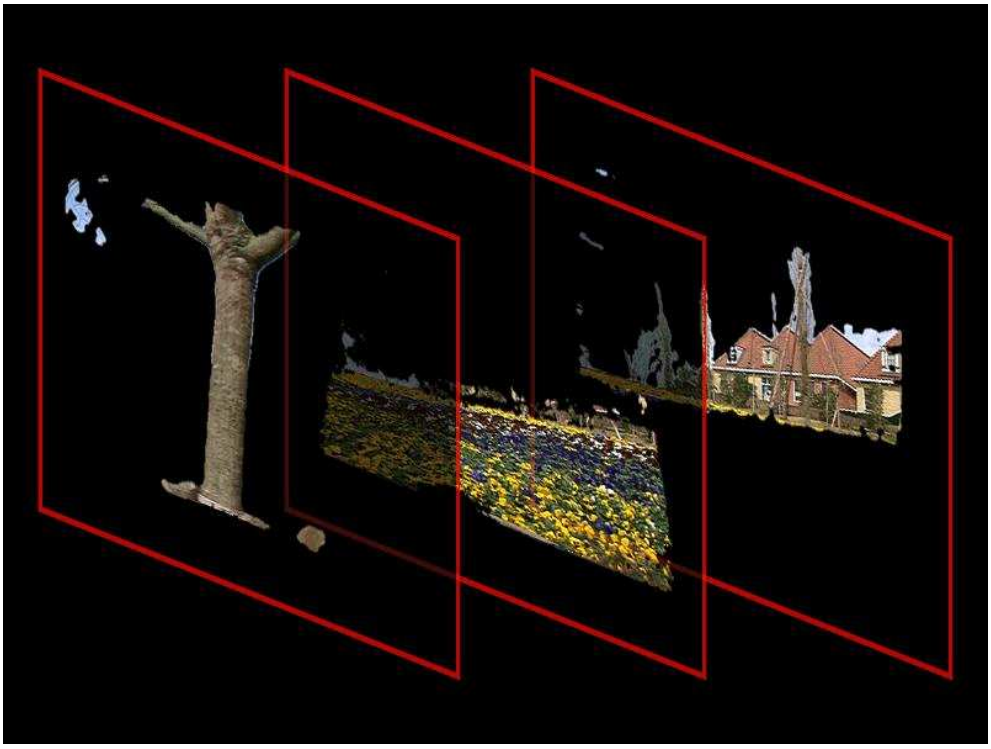


FIGURE 1.2 – Représentation de la séquence *Flowergarden* grâce aux mosaïques des couches de mouvement synthétisées par Wang et Adelson. Seules les trois mosaïques principales correspondant à l'arbre, les fleurs et les maisons sont représentées avec respect de leur ordre de profondeur. Cette figure provient de [Wang 94b].

instants n'est pas garantie. Pour tenter de remédier au problème, les traitements séquentiels de ce type utilisent généralement en guise d'initialisation à chaque nouvel instant t une carte de segmentation prédite f'_t par compensation de la carte de segmentation f_{t-1} qui vient d'être obtenue pour l'instant $t-1$.

1.2.1 Segmentation d'un champ de mouvement dense estimé au préalable

À partir d'un champ de mouvement dense estimé au préalable, Wang et Adelson [Wang 94a] résument le problème de la segmentation basée sur le mouvement à la détermination d'un ensemble de modèles paramétriques décrivant au mieux le mouvement « observé ». À partir d'une partition initiale régulière du champ dense de mouvement, un ensemble de modèles paramétriques de mouvement est estimé par régression linéaire standard. Un modèle lié à une forte erreur résiduelle est considéré comme aberrant puisqu'il ne fournit pas une bonne description du mouvement pour la région sur laquelle il a été estimé. Il est donc éliminé. Des modèles présentant des paramètres de valeurs similaires sont regroupés par l'algorithme de clustering des k-

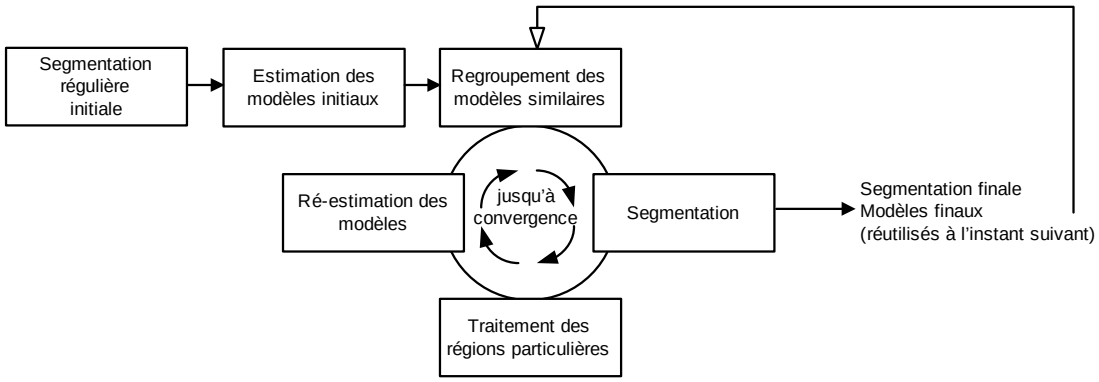


FIGURE 1.3 – Diagramme simplifié de l'algorithme de segmentation basée sur le mouvement [Wang 94a]. La flèche blanche indique le passage au traitement de l'image suivante.

moyennes dans l'espace des paramètres de mouvement, afin de produire un seul modèle à partir du centre du cluster. La segmentation se fait pixel à pixel en testant chacun des modèles-hypothèses, c'est-à-dire en calculant la distance euclidienne entre le vecteur de mouvement issu du champ dense et le vecteur de mouvement synthétisé à partir du modèle paramétrique testé. Chaque pixel est assigné au modèle qui décrit le mieux son mouvement, c'est-à-dire au modèle pour lequel cette distance est minimale. Si la distance minimale reste supérieure à un seuil donné, le pixel considéré n'est pas classé. De manière itérative, à partir de la nouvelle segmentation, les modèles de mouvement sont mis à jour et les ensembles connexes de pixels non-classés produisent de nouvelles hypothèses à condition que leur taille soit suffisante pour permettre l'estimation stable d'un modèle. Le processus d'assignation recommence avec les nouveaux modèles, ainsi de suite jusqu'à atteindre la stabilité lorsque peu de pixels changent encore de classe. La différence inter-images déplacée (« Displaced Frame Difference » (DFD)) est utilisée pour affiner la segmentation dans les régions toujours non-classées. Ces régions correspondent généralement à des régions proches des frontières de mouvement où le flot optique est moins précis. Pour l'image suivante, puisqu'*a priori* le mouvement évolue lentement d'une image à l'autre, les modèles et les régions resteront sensiblement les mêmes. Les modèles courants sont donc utilisés à nouveau pour classer les régions, et très peu d'itérations sont nécessaires pour atteindre la stabilité. La figure 1.3 présente le diagramme simplifié de l'algorithme.

Ayer et Sawhney [Ayer 95] proposent une première formulation probabiliste du problème. Étant donnés les paramètres des modèles de mouvement, les distributions de probabilités des résidus sont modélisées par des mélanges de gaussiennes. Le nombre adéquat de modèles est déterminé en fonction de la complexité de la modélisation selon le principe « Minimum Description Length » (MDL) [Rissanen 83]. L'approche proposée fait appel à l'algorithme itératif d'EM (Espérance-Maximisation). L'étape E correspond à l'estimation des supports de couches par calcul des probabilités d'appartenance d'un pixel à une couche étant donnés le nombre de modèles, les estimées des paramètres de ces modèles et les variances. L'étape M correspond à l'estimation

des nouveaux paramètres de mouvement, des nouvelles proportions des modèles et des nouvelles variances, étant donnés les nouveaux supports de couches.

Pedersini *et al.* [Pedersini 96] se basent principalement sur la méthode de Wang et Adelson [Wang 94a] mais procèdent à la classification en deux temps : d'abord la classification des régions « fiables » (i.e. sur lesquelles le champ de mouvement est jugé fiable), puis la classification des régions « non fiables ». Comme dans [Wang 94a], à partir d'une partition régulière de l'image, un modèle paramétrique est estimé pour chaque région par régression linéaire. Mais contrairement à [Wang 94a], l'estimation ne se fait non pas sur la région entière mais uniquement sur une portion de cette région, portion dont le champ de mouvement est jugé fiable via une procédure de seuil sur la DFD. C'est également la DFD, invoquant cette fois le vecteur de mouvement synthétisé à partir du modèle paramétrique testé, qui est utilisée comme critère de classification des régions « fiables ». Une approche de régularisation basée sur les modèles de champs de Markov aléatoires (« Markov Random Fields » (MRF)) est proposée pour obtenir la segmentation finale (y compris dans les régions « non fiables »). Si deux pixels voisins ont des vecteurs de mouvement différents, un terme de régularisation spatiale basé sur une mesure de régularité du champ de mouvement pénalise le fait de classer ces deux pixels dans une même région.

1.2.2 Estimation et segmentation simultanées du mouvement pour une estimation plus précise du flot optique

Dans les formulations classiques du problème de l'estimation du flot optique, l'hypothèse de conservation de l'intensité lumineuse et la contrainte de continuité spatiale sont violées entre autres dès que l'on considère plusieurs objets se déplaçant indépendamment les uns des autres et générant des occultations. Black et Anandan [Black 96a] proposent une méthode pour estimer de manière robuste plusieurs mouvements paramétriques dans une même image. Appliquée aux formulations du flot optique, cette méthode robuste permet de réduire la sensibilité à ce type de violations.

Black et Jepson [Black 96b] segmentent d'abord une image en régions en s'appuyant sur l'information de luminance, puis ils estiment le mouvement dans chaque région en utilisant des modèles paramétriques de mouvement. Le mouvement peut être estimé de manière précise à condition que la segmentation fournie soit correcte, ce qui n'est pas garanti en utilisant seulement l'information de luminance.

Ju *et al.* proposent le modèle « Skin and Bones » [Ju 96]. Ils fixent une partition régulière de l'image (des patchs de petite taille). Puisque ces patchs fixés peuvent contenir des régions de mouvement différent, ils modélisent dans chaque région plusieurs mouvements affines en utilisant une procédure d'estimation en couches, extension directe des modèles de mélange [Jepson 93]. Ces patchs de mouvements affines constituent les « os » rigides et sont connectés entre eux par une « peau » flexible qui traduit une cohérence spatiale entre les paramètres de patchs voisins.

Mémin et Pérez [Mémin 02] introduisent un modèle qui combine efficacement un lissage local non paramétrique et une représentation paramétrique plus globale par régions. Le système proposé permet de procéder simultanément à l'estimation dense du

mouvement préservant les discontinuités et à la segmentation basée sur le mouvement.

1.2.3 Segmentation paramétrique directe sans estimation de champ de mouvement

Irani *et al.* [Irani 92] proposent une approche hiérarchique descendante fonctionnant de manière séquentielle. Les couches de mouvement sont extraites une par une, par le biais d'estimations successives du mouvement dominant par la méthode de régression standard des moindres carrés. Une fois le mouvement dominant estimé, les régions correspondant à des résidus faibles sont détectées puis exclues de la région d'analyse avant que le processus soit relancé sur les données restantes. Néanmoins les méthodes de ce type ne fonctionnent généralement pas de manière satisfaisante en l'absence d'un mouvement dominant bien défini. D'autre part, une estimation imprécise du premier modèle de mouvement peut empêcher la bonne détection des autres mouvements.

Odobez et Bouthemy [Odobez 98] proposent une méthode de segmentation dite « directe » non seulement dans le sens où elle ne requiert pas l'estimation d'un champ de mouvement, mais aussi dans le sens où un estimateur robuste multi-résolution [Odobez 95] fournit en une seule itération des modèles paramétriques jugés suffisamment précis pour ne pas avoir à alterner les habituelles étapes d'estimation des modèles de mouvement et de classification dont les itérations jusqu'à convergence peuvent s'avérer très coûteuses en temps de calcul.

Cremers et Soatto [Cremers 05] encouragent une frontière de mouvement de longueur minimale. Ils proposent donc deux représentations appropriées de cette frontière de mouvement. La première, assez contraignante, est basée sur les splines et permet par exemple de suivre un objet en mouvement. La seconde, plus avancée, est une formulation implicite basée sur les lignes de niveaux (« level sets »). Elle permet de segmenter plusieurs régions en mouvement. Néanmoins elle reste une approche locale, et ne permet donc pas de traiter convenablement le cas de nouveaux objets entrant dans la scène assez loin de la frontière en cours d'évolution.

1.2.4 Segmentation par regroupement de régions élémentaires fournies par une sur-segmentation spatiale initiale

Morier *et al.* [Morier 97] proposent une méthode de représentation hiérarchique des séquences d'images. Cette méthode repose sur une technique de segmentation d'image basée sur des procédures de détection de contours et de fermeture de contours par croissance de régions [Benois-Pineau 92]. Les régions issues de cette segmentation spatiale initiale correspondent au niveau de base de la représentation hiérarchique. Les niveaux supérieurs sont construits par fusion progressive des régions sur des critères d'homogénéité de mouvement inspirés de ceux introduits dans [Wu 96].

Moscheni *et al.* [Moscheni 98] procèdent d'abord à la sur-segmentation de la première image de la paire en s'appuyant sur les informations de couleur. Ils affinent ensuite cette sur-segmentation en vérifiant l'homogénéité spatio-temporelle propre à chacune des régions obtenues en considérant cette fois les deux images constituant la paire. Si

nécessaire, une région peut être divisée à nouveau en régions plus petites, toujours en utilisant l'information de couleur. Les différentes couches de mouvement sont formées par regroupements successifs de deux régions voisines basés sur une mesure de similarité spatio-temporelle qui met en jeu, d'une part, les informations de mouvement contenues dans la représentation paramétrique et dans la distribution résiduelle des deux régions, et d'autre part, les luminances contenues sur la frontière commune aux deux régions.

1.3 Méthode travaillant sur une paire d'images distantes

Contrairement aux travaux mentionnés précédemment qui s'intéressent au mouvement estimé entre deux images consécutives, Wills *et al.* [Wills 03] ont développé une méthode de segmentation de deux images distantes pour lesquelles l'estimation du flot optique n'est pas efficace car les mouvements présents sont relativement grands. En se basant sur des correspondances de points d'intérêt entre deux images distantes I et I', Wills *et al.* proposent une variante de l'algorithme de RANSAC pour partitionner cet ensemble de correspondances et estimer un modèle homographique de mouvement pour chacun des groupes obtenus. La segmentation dense de l'image I (resp. I') est obtenue par minimisation d'une fonction d'énergie globale composée d'un terme lié aux modèles « avant » (resp. « arrière ») de mouvement et d'un terme de lissage spatial. L'algorithme de graph-cuts est utilisé pour la minimisation de la fonction d'énergie. La cohérence temporelle entre les deux cartes de segmentation est finalement assurée par intersection des segmentations « avant » et « arrière ». La méthode proposée permet donc d'obtenir la segmentation dense de deux images distantes en ignorant totalement les images intermédiaires. La segmentation de ces images intermédiaires n'est pas mentionnée dans l'article. Plusieurs exécutions, indépendantes les unes des autres, lancées sur différentes paires d'images distantes d'une même séquence n'assureront malheureusement pas une consistance temporelle des différentes segmentations.

1.4 Méthodes travaillant sur un groupe d'images

Cet ensemble regroupe des méthodes où les mouvements mis en jeu peuvent être estimés aussi bien entre images consécutives [Shi 98], qu'entre une image de référence fixée et des images pouvant être plus éloignées [Ke 01, Ke 02, Xiao 05b, Liu 05, Schoenemann 08]. Les travaux de Dupont *et al.* [Dupont 06b, Dupont 06c, Dupont 06a] font également partie de cet ensemble de méthodes mais sont décrits dans la section 1.5.

1.4.1 Mouvements estimés entre images consécutives seulement

Shi et Malik [Shi 98] n'utilisent pas le flot optique mais s'appuient tout de même sur une mesure locale, celle de distribution de probabilité de déplacement instantané en chaque pixel calculé entre images consécutives, mesure appelée « profil de mouvement ». La segmentation des images de la séquence se fait en construisant un graphe pondéré, dans lequel chaque nœud correspond à un pixel. Le voisinage spatio-temporel considéré

autorise la création d'arcs entre pixels distants au plus de 3 images dans le temps et 5 pixels dans l'espace [2]. En utilisant une simple corrélation croisée comme mesure de similarité entre les profils de mouvement de deux pixels voisins, le graphe est segmenté par le critère « normalized cut » qui revient à chercher les vecteurs propres d'une matrice associée au graphe. Seule l'information de mouvement est utilisée. L'approche est non-paramétrique et présente l'avantage de ne pas dépendre d'un état d'initialisation. Cependant, elle ignore toute contrainte globale, ce qui a le désavantage de la rendre instable pour des séquences bruitées, même légèrement.

L'approche est simultanée dans le sens où elle permet l'obtention de plusieurs cartes de segmentations à la fois. Toutefois, pour limiter le temps de calcul, les auteurs ont recours à deux stratégies. D'une part, le nombre de nœuds dans le graphe est limité en réduisant la résolution des images d'un facteur 3. D'autre part, le nombre d'images prises en compte simultanément est fixé : le traitement d'une longue séquence se fait finalement de manière séquentielle en utilisant une fenêtre temporelle glissante centrée sur chaque nouvel instant.

1.4.2 Mouvements estimés entre une image de référence fixée et une deuxième image pouvant être éloignée de la référence

Travaux de Ke et Kanade

Ke et Kanade [Ke 01] appliquent l'algorithme de clustering Mean Shift [Comaniciu 02] pour obtenir une sur-segmentation d'une image de référence basée sur la couleur et supposent que tous les pixels d'une même région de couleur appartiennent à la même couche de mouvement. Dans le cas où l'on dispose en entrée de deux images originales (dont la référence), pour chaque région de couleur, on estime un modèle paramétrique de mouvement entre l'image de référence et la seconde image. Pour une région de grande taille présentant suffisamment de variation d'intensité, on estime un modèle affine. Pour une région de faible variation d'intensité, on estime un modèle de translation à partir des frontières de la région. Profitant du fait que ces mouvements forment un sous-espace linéaire de $\mathbb{R}^6$ de dimension 3, le problème de segmentation est alors formulé comme un problème de clustering dans ce sous-espace où les clusters sont plus facilement distinguables les uns des autres. Les modèles de mouvement des régions de couleur sont projetés dans le sous-espace linéaire et c'est à nouveau l'algorithme Mean Shift [Comaniciu 02], appliqué cette fois aux données du sous-espace de dimension 3, qui est utilisé pour déterminer les couches initiales de mouvement. Après ré-estimation d'un modèle affine de mouvement pour chaque couche initiale, les supports de couches sont affinés en réaffectant chaque région de couleur à la couche dont le modèle minimise un résidu de mouvement. Il est bon de noter que cette étape de raffinement ne se fait pas au niveau des pixels mais au niveau des régions de couleur. On ne remet donc jamais en cause la sur-segmentation initiale, ce qui peut s'avérer problématique principalement pour une raison : au cas où la sur-segmentation n'est pas assez fine, elle

2. Certains arcs relient donc des pixels d'images non consécutives mais les mouvements considérés ne sont que les mouvements instantanés estimés entre images consécutives.

risque d'engendrer des régions contenant des pixels voisins de couleur proche mais de mouvement différent qui seront pourtant assignés à une même couche.

Un des inconvénients de la méthode est l'estimation peu robuste des modèles de mouvement sur des régions de taille relativement petite. En effet les modèles estimés conviennent pour les données propres à la région mais peuvent différer largement du modèle global optimal de la couche correspondante. Une sur-segmentation moins fine augmenterait la taille des régions et permettrait d'éviter ce problème, mais le risque qu'une même région contienne plusieurs mouvements serait alors très fort. Afin de résoudre le problème et donc gagner en robustesse, les mêmes auteurs [Ke 02] proposent de découper l'image en patchs initiaux réguliers de taille 16×16, d'estimer un modèle paramétrique de mouvement pour chacun des patchs et d'accroître (ou réduire) progressivement chaque patch en un composant k-connecté (KCC) dont tous les pixels ont un résidu de mouvement inférieur à un seuil donné. Si le KCC final ne recouvre pas le patch initial 16×16 alors le modèle associé est considéré comme aberrant et rejeté définitivement. Ce cas correspond généralement à un patch initial chevauchant une frontière de mouvement. La classification finale se fait toujours au niveau de régions de couleur issues de la sur-segmentation fine Mean Shift. Il est fréquent que la présence de bruit dans une image engendre des régions de couleur de très petite taille, qui sont alors mal classées car le bruit apparaît à des positions aléatoires dans les autres images de la séquence. Pour améliorer la cohérence temporelle des segmentations et décrire ainsi la séquence par un petit nombre de couches compactes, ces régions très petites sont réassignées en considérant que chaque couche correspond à un plan rigide dont la forme reste constante ou bien varie lentement au cours du temps.

Les deux algorithmes précédents visent donc à obtenir la segmentation basée mouvement d'une image de référence. Pour plus de clarté, leurs descriptions ont été faites dans le cas où l'on dispose uniquement d'une paire d'images. Toutefois si plus de deux images originales sont disponibles, ces algorithmes peuvent être directement appliqués en profitant cette fois du fait que les mouvements estimés entre l'image de référence et chacune des N images disponibles autres que la référence forment un sous-espace linéaire de $\mathbb{R}^{6\times N}$ de dimension 3.

Méthode de Xiao et Shah [Xiao 05b]

La méthode de Xiao et Shah [Xiao 05b] se décompose en deux étapes. La première étape de description des couches débute par la détection de points d'intérêt dans une image de référence. Des patchs centrés sur ces points sont suivis dans les images suivantes jusqu'à ce que leur mouvement moyen atteigne un certain seuil indiquant que le mouvement est suffisamment important pour que les différentes couches puissent être distinguées les unes des autres. Une approche par croissance de régions permet ensuite de regrouper les pixels proches des points d'intérêt selon un critère de similarité de mouvement. Enfin, sur des critères de chevauchement et de similarité de mouvement, certaines régions sont fusionnées. À la fin de cette première étape, le nombre de couches de mouvement composant la scène ainsi que les modèles paramétriques de mouvement correspondant à chacune des couches sont connus. Mais à ce stade la représentation en

couches n'est qu'approximative, des ambiguïtés demeurent dans les régions faiblement texturées et les frontières de mouvement demandent à être affinées.

La seconde étape vise à obtenir une segmentation précise en couches de mouvement malgré la présence d'occultations. Le système repose sur l'utilisation simultanée de plusieurs images de la séquence, sur l'identification explicite des pixels occultés (et donc l'utilisation d'une étiquette particulière d'occultation) et sur la définition de contraintes temporelles sur les occultations. En faisant l'hypothèse que les objets en présence ne sont pas trop fins et ne bougent pas trop vite, l'aire des régions d'occultations estimées entre une image de référence I_1 et une image courante I_j augmente avec la distance entre l'instant de référence et l'instant courant. Ainsi, en raisonnant pour un pixel de l'image de référence I_1 associé à la paire d'images (I_1, I_j), si ce pixel est occulté entre les instants 1 et j, ce pixel sera également occulté entre les instants 1 et $j+1$. Autrement dit, le fait d'assigner à un même pixel $\mathbf{p}$ de l'image de référence deux étiquettes différentes $(f_j(\mathbf{p}) \neq f_{j+1}(\mathbf{p}))$ à deux instants consécutifs j et $j+1$ est fortement pénalisé si $f_{j+1}(\mathbf{p})$ n'est pas l'étiquette d'occultation. Cette formulation permet de maintenir la consistance temporelle de la segmentation. La contrainte temporelle sur les occultations est intégrée dans un modèle de graphe multi-images (à 3 dimensions) où de nouveaux liens de voisinage temporel relient les paires de pixels du type $(\mathbf{p}_j, \mathbf{p}_{j+1})$. Le problème de segmentation est formulé comme un problème de minimisation d'une fonction d'énergie qui est résolu par l'algorithme des graph cuts.

En pratique, le nombre d'images utilisées simultanément n'excède pas 5. Un ensemble de 5 images consécutives $1, 2, 3, 4, 5$ est utilisé pour obtenir simultanément les segmentations entre les paires d'images $(1, 2)$, $(1, 3)$, $(1, 4)$ et $(1, 5)$. La segmentation entre $(2, 3)$ est d'abord prédite de manière approximative en utilisant les précédents résultats de segmentation entre $(1, 2)$. Elle est ensuite affinée en utilisant à nouveau l'algorithme des graph cuts faisant cette fois appel à l'ensemble des images $2, 3, 4, 5, 6$. La segmentation entière de la séquence d'images nécessite donc la répétition séquentielle du processus précédent, ce qui paraît sous-optimal et s'avère extrêmement coûteux en temps de calcul.

Par ailleurs, pour certaines applications, cette méthode présente l'inconvénient majeur de ne pas fournir une classification réellement dense de la séquence d'images. En effet, par nature, la méthode classe certains pixels comme appartenant à la couche des occultations et ne les associe donc à aucune des couches de mouvements.

Méthode de Liu *et al.* [Liu 05]

Liu *et al.* [Liu 05] s'appuient sur le clustering d'un ensemble de trajectoires de points d'intérêt pour extraire les différents mouvements présents dans la scène. Plus précisément, des coins de Harris [Harris 88] sont détectés dans la première image de la séquence puis suivis tout au long de la séquence. L'ensemble des trajectoires résultantes est partitionné en un nombre prédéfini de clusters, en utilisant la corrélation normalisée comme mesure de similarité et l'algorithme de clustering spectral [Shi 98]. L'intérêt d'utiliser les trajectoires est qu'elles fournissent une information de mouvement considérablement plus riche que les déplacements instantanés. Pour chaque cluster extrait

(qui correspond à une couche de mouvement), des champs denses de mouvement sont ensuite interpolés entre toutes les paires d'images de la séquence : non seulement entre les paires d'images consécutives mais aussi entre images éloignées. De manière détaillée, pour tout pixel autre que les points suivis et pour une couche considérée, un vecteur de mouvement est interpolé à partir d'un modèle affine estimé localement par régression pondérée sur les points d'intérêt (de la couche considérée) les plus proches du pixel traité. Ces champs de mouvement sont ensuite utilisés dans le terme d'énergie lié au mouvement, aux côtés d'un terme d'apparence basé sur la couleur et d'un terme de lissage spatial. Pour la segmentation d'une image correspondant à l'instant t, le terme lié au mouvement est basé sur la somme de résidus DFD calculés entre l'instant t et les instants $t+1$, ..., $t+10$, $t-1$, ..., $t-10$, rendant le critère plus fiable qu'en se basant uniquement sur un résidu calculé entre t et $t+1$ par exemple.

Pour autant, cette distance de 10 pas de temps, qui a été fixée de manière empirique et qui reste la même quel que soit le pixel traité, n'est pas adaptée aux occultations. Par exemple, pour un pixel visible entre les instants $t-10$ et t mais occulté entre les instants $t+1$ et $t+10$, puisqu'on calcule la simple somme des 20 résidus, le coût correspondant au terme de mouvement risque d'être fort même si le pixel traité appartient bien à la couche testée. Les auteurs proposent de classer comme aberrants les pixels dont le mouvement engendre un fort coût pour toutes les couches. En présence de mouvements rapides, les zones d'occultations étant larges, les pixels classés comme aberrants peuvent correspondre à des éléments importants pour lesquels le système a malheureusement jugé qu'aucun mouvement ne convenait.

La fonction d'énergie globale ne contient pas de contraintes temporelles et les segmentations des différentes images de la séquences sont obtenues indépendamment les unes des autres, ce qui engendre inévitablement des inconsistances temporelles dans les segmentations. Un post-traitement est proposé afin d'améliorer la consistance temporelle des résultats. En utilisant les champs denses de mouvement qui ont été interpolés, à chaque pixel du volume vidéo on associe une trajectoire de mouvement. Les différentes positions traversées par cette trajectoire au cours du temps peuvent s'être vues attribuer des étiquettes de couches différentes. Chacune de ces positions va donner une voix pour l'étiquette qu'elle porte. L'étiquette finale que l'on associe au pixel considéré initialement est l'étiquette (si elle existe) qui a récolté au moins 80% des voix.

Méthode de Schoenemann et Cremers [Schoenemann 08]

La représentation en couches proposée initialement par Wang et Adelson [Wang 94a] n'est réellement obtenue qu'après une seconde étape succédant à l'étape de segmentation basée sur le mouvement. Cette étape de segmentation fournit, pour chaque image de la séquence, les supports non-chevauchant correspondant aux couches extraites et les modèles paramétriques de mouvement associés. Ce n'est que dans la seconde étape de synthèse des couches que les informations appartenant à une même couche mais à des supports d'instants différents sont accumulées, afin de former les « vraies » couches que nous avons préféré appeler « mosaïques de couches de mouvement » pour éviter toute ambiguïté. (Ces mosaïques sont également appelées « sprite d'objet vidéo planaire

(S-VOP) » dans le domaine de la compression vidéo.)

Dans une approche simultanée, Schoenemann et Cremers [Schoenemann 08] intègrent la génération de ces mosaïques dans le processus de segmentation. Dans le terme d'énergie lié au mouvement, un résidu DFD est estimé entre les images originales et les mosaïques alors qu'habituellement ce résidu est estimé entre deux images originales (le plus souvent consécutives). En pratique, pour chaque couche extraite, une mosaïque alignée sur l'instant de référence 0 est générée, ce qui signifie que la première image de la séquence peut être obtenue par superposition directe des mosaïques sans déformation. (Bien entendu, la superposition doit respecter l'ordre de profondeur. Les auteurs supposent que plus le déplacement d'une couche est rapide, plus cette couche est proche. Cette hypothèse est vérifiée entre autres pour les cas d'une caméra en mouvement filmant une scène statique, mais ne convient pas toujours.) La génération d'une mosaïque se fait en alternant successivement estimation du modèle de mouvement, mise à jour de l'apparence et mise à jour du support.

Ces travaux se rapprochent de notre schéma reposant sur les mosaïques [Fradet 08a] dans la mesure où ils sont les seuls à faire apparaître un terme liant les images originales aux mosaïques.

Méthode de Min et Médioni [Min 08]

Min et Médioni [Min 08] développent une approche exploitant des informations de mouvement correspondant à des trajectoires temporelles, et non à de simples mises en correspondance entre deux images. Ils procèdent tout d'abord à la sur-segmentation d'une image de référence basée sur la couleur, et à l'estimation de trajectoires temporelles pour tout pixel de cette image de référence. Les ensembles de trajectoires de points d'une même région issue de la sur-segmentation initiale sont lissés spatialement et temporellement par un procédé de « tensor voting » dans l'espace à 5 dimensions (x, y, t, vx, vy) qui permet à la fois de détecter les trajectoires aberrantes et de les corriger en utilisant les trajectoires voisines appartenant à la même région.

Pour obtenir finalement la segmentation basée sur le mouvement de l'image de référence, des régions peuvent être fusionnées pour former des couches de mouvement complètes. Les vecteurs vitesse moyens sur la frontière commune à deux régions adjacentes sont utilisés pour définir une mesure de similarité entre ces deux régions. En pratique, une dizaine d'images sont traitées simultanément. Les séquences plus longues sont divisées en plusieurs intervalles chevauchant, et chaque intervalle est traité séparément.

1.5 Cheminement de Dupont *et al.*

Dupont *et al.* se sont appuyés sur l'algorithme de Xiao et Shah [Xiao 05b] pour développer leurs travaux [Dupont 05, Dupont 06b, Dupont 06c, Dupont 06a]. Leurs efforts se sont portés notamment sur la définition de contraintes temporelles appropriées entre images consécutives, mais aussi entre images plus distantes. Il est important de

bien comprendre le cheminement de Dupont qui a beaucoup influencé nos propres travaux

Une première approche séquentielle [Dupont 05] segmente l'image courante de l'instant t en faisant appel à un critère de mouvement basé sur un résidu estimé entre les images t et $t+1$, à un critère d'apparence visuelle, à un critère de régularisation spatiale, et à des contraintes temporelles entre t et $t-1$, c'est-à-dire entre les support des couches de l'image courante et les supports des couches connues de l'image précédente qui a déjà été traitée. Un processus itératif d'estimation robuste des modèles de mouvement combine les estimations de modèles « avant » (entre t et $t+1$) et « arrière » (entre $t+1$ et t), ce qui permet de détecter la présence d'occultations et donc d'améliorer l'estimation des paramètres des modèles en diminuant l'influence des zones occultées. L'approche de « coupe minimale-flot maximal » dans un graphe est retenue comme technique de minimisation. L'algorithme d'alpha-expansion permet d'obtenir une bonne approximation de la solution optimale, via la construction d'un graphe dont les nœuds correspondent aux pixels de l'image courante et forment une grille 2D. L'initialisation des supports des couches de l'image suivante I_{t+1} se fait par projection de la segmentation que l'on vient d'obtenir pour l'image courante I_t. Pour la première image, une partition régulière est utilisée en guise d'initialisation.

Les approches séquentielles présentent l'inconvénient de dépendre énormément de la segmentation initiale. Dans leurs travaux suivants [Dupont 06b], les mêmes auteurs proposent une approche simultanée pour segmenter les N premières images de la séquence. En pratique, N vaut entre 3 et 5. Les modèles de mouvement entre les paires d'images successives $(1,2)$, $(2,3)$, ..., $(N-1,N)$ sont d'abord estimés. Puis les supports de couches des N images sont obtenus en même temps en introduisant des contraintes temporelles simultanées entre les couches (encore inconnues) des paires d'images $(1,2)$, $(2,3)$, ..., $(N-1,N)$. Le graphe utilisé dans l'approche séquentielle est étendu en un graphe dont les nœuds correspondent à la grille 3D constituée par les pixels des N images. Des liens de voisinage temporel sont ajoutés entre deux niveaux pour prendre en compte les contraintes temporelles simultanées (ou contraintes de lissage temporel). Le fait d'appliquer cette approche simultanée aux N premières images de la séquence permet de prendre en compte des informations temporelles plus riches que celles utilisées dans l'approche séquentielle. Les segmentations des premières images sont donc plus précises. Pour les images suivantes, on utilise l'approche séquentielle tout en continuant à bénéficier de la précision des premières segmentations puisque l'initialisation des supports des régions de l'image à l'instant t, $t > N$, se fait par projection de la segmentation obtenue précédemment pour l'image à l'instant $t-1$.

En considérant uniquement des mouvements estimés entre paires d'images consécutives, il peut être difficile de distinguer les uns des autres des mouvements de faibles amplitudes. De même qu'en ne considérant que des résidus de mouvement « avant » (entre les images correspondant aux instants t et $t+1$) et des contraintes temporelles « avant », il est difficile de traiter de manière satisfaisante les régions d'occultations. Dans des travaux postérieurs [Dupont 06c], pour rendre le système plus robuste aux occultations, les auteurs font intervenir également le résidu de mouvement « arrière » (entre les images correspondant aux instants t et $t-1$). Les deux résidus sont combinés

en ne retenant que le minimum des deux, ce qui s'avère efficace si l'on considère qu'un pixel (visible dans l'image courante) occulté dans l'image suivante ne l'est généralement pas dans la précédente, et inversement. De plus, pour lever les ambiguïtés dues à des mouvements de faibles amplitudes, le critère de mouvement est étendu en considérant également les résidus de mouvement entre paires d'images distantes $(t, t+2)$, $(t, t+3)$, ..., $(t, t-2)$, $(t, t-3)$, ... De manière similaire, les contraintes de lissage temporel comprennent des contraintes « avant » et des contraintes « arrière » entre les images $(t, t+1)$, $(t, t+2)$, $(t, t+3)$, ..., et $(t, t-1)$, $(t, t-2)$, $(t, t-3)$, ... C'est cette formulation la plus aboutie que Dupont reprend dans son manuscrit de thèse [Dupont 06a]. Il est important de noter que dans [Dupont 06c], le graphe considéré est, comme celui mentionné précédemment, un graphe à trois dimensions, mais sa troisième dimension comporte cette fois autant de niveaux que la séquence comporte d'images. Enfin, cet article traite aussi bien du suivi des parties visibles des couches de mouvement que du suivi des parties cachées. Dans nos travaux, nous ne nous intéressons qu'aux parties visibles.

1.6 Problématique et approches proposées

1.6.1 Problématique

Nos travaux se situent principalement dans le contexte de la post-production cinématographique où l'exigence de qualité visuelle des contenus est la première contrainte. La position des frontières de mouvement à extraire doit donc être particulièrement précise. Nous nous sommes donc tournés assez naturellement vers les approches récentes développées dans le cadre coupe minimale/flot maximal qui sont les approches qui, aujourd'hui, fournissent les meilleurs résultats. Notons également d'ores et déjà que dans ce contexte de post-production, il est généralement préférable que les approches reposent sur des indications fournies par un opérateur, même si cela réduit le degré d'automatisme.

Comme l'a montré l'état de l'art précédent, les méthodes ne traitant que des informations de mouvement estimées entre deux images consécutives rencontrent des difficultés à distinguer certains mouvements de faible amplitude. Cette limite peut être surmontée en travaillant sur des informations plus riches, ce qui est possible en prenant en compte simultanément plus de deux images consécutives. Ceci rend également le système moins sensible au bruit. Le principal désavantage de ces approches simultanées est qu'elles sont particulièrement coûteuses en temps de calcul. Même si ce temps de calcul n'est pas notre contrainte première, nous devons garder en tête qu'il n'existe de toute manière pas de système infaillible et que dans le contexte de la post-production un opérateur a toujours à surveiller, valider, invalider, corriger, voire relancer un traitement. Partant de cette constatation, il faut que les temps de traitement restent compatibles avec une utilisation confortable du système par l'opérateur. Concrètement il ne faut pas que l'opérateur attende la fin d'un traitement interminable pour se rendre compte que les résultats ne lui conviennent pas et qu'à ce stade on ne lui propose rien d'autre que de tout corriger manuellement ou de relancer l'intégralité du traitement en modifiant

un jeu de paramètres.

Nous nous sommes donc intéressés aux problèmes suivants :

1. Comment, dans une approche séquentielle, introduire de nouvelles contraintes spatiales et temporelles pour obtenir des résultats de segmentations dont la cohérence spatio-temporelle serait aussi forte que dans une approche simultanée ?
2. Comment exploiter favorablement et à moindre coût la large quantité d'informations redondantes contenues dans une séquence d'images ?
3. Comment profiter de la présence d'un opérateur pour améliorer les résultats ou simplifier les traitements sans que les interactions soient trop complexes, ou trop fastidieuses ?

Cette première partie du document présente deux nouveaux schémas de segmentation basée sur le mouvement qui ont chacun fait l'objet d'une publication. Le principe général de chacun des algorithmes correspondants est décrit dans la sous-section suivante.

1.6.2 Approches proposées

Pour les deux schémas qui vont être proposés, nous faisons l'hypothèse supplémentaire que le nombre de couches de mouvement à extraire ainsi que leur ordre de profondeur restent constants au cours de la séquence. En pratique, cette hypothèse n'est pas tellement restrictive : dans notre contexte interactif, l'opérateur peut aisément découper la séquence en plusieurs intervalles temporels qui vérifient l'hypothèse.

Les points communs aux deux schémas sont :

– l'utilisation d'un estimateur de champ dense de mouvement,
– l'utilisation de modèles affines à 6 paramètres pour modéliser les mouvements des différentes couches,
– l'utilisation de l'algorithme d'alpha-expansion pour obtenir une solution approchée d'un problème non-binaire,
– le recours à une bande d'incertitude autour des frontières de mouvement prédites.

Les deux premiers points sont présentés plus en détails dans le chapitre 3, les deux derniers dans le chapitre 2.

Tous les termes d'énergie mentionnés ci-après sont détaillés mathématiquement au chapitre 4.

Premier schéma basé sur des triplets d'images consécutives

Le premier schéma séquentiel que nous avons développé fournit la segmentation de l'image courante en considérant simultanément les trois images composant le triplet centré sur l'image courante. Ce triplet permet de confronter les mouvements « avant » estimés entre les instants t et $t+1$ aux mouvements « arrière » estimés entre t et $t-1$. La confrontation rend utilisable le critère de mouvement même dans les régions d'occultations puisque généralement un pixel visible à l'instant t mais occulté à l'instant $t+1$ (resp. $t-1$) est visible à l'instant $t-1$ (resp. $t+1$).

En travaillant de manière séquentielle avec une fenêtre temporelle glissante centrée sur l'instant t et limitée à trois instants, la carte de segmentation précédente f_{t-1} a été déjà obtenue et n'est pas remise en cause, la carte de segmentation courante f_t est celle que l'on cherche à déterminer, tandis que la carte de segmentation suivante f_{t+1} est inconnue pour le moment. Nous ne pouvons pas définir réellement de contraintes temporelles autres que la contrainte standard qui encourage une consistance entre la segmentation courante et la segmentation précédente. Cependant nous proposons deux façons de la renforcer.

1. La contrainte standard n'implique rien pour les régions occultées à l'instant $t-1$ et apparaissant à l'instant t. Pour ces régions, on peut donc se fier à l'un des autres critères qui ont déjà été proposés dans la littérature (critères de mouvement, d'apparence ou de lissage spatial). Néanmoins aucun de ces critères ne peut être exploité efficacement dans le cas particulier de régions apparaissantes, homogènes, et de même intensité que la région occultante. L'ajout d'une nouvelle contrainte de rigidité appliquée aux objets occultants permet d'assurer la consistance temporelle des couches d'avant-plan en décourageant l'insertion des étiquettes correspondantes dans les régions apparaissantes.
2. Après avoir obtenu la carte de segmentation de l'instant courant f_t, cette carte est compensée en utilisant les mouvements « avant » de chacune des couches extraites de manière à générer une carte de segmentation prédite f'_{t+1} pour l'instant suivant. Pour renforcer encore davantage la cohérence temporelle, seuls les pixels situés dans une bande d'incertitude autour des frontières de mouvement prédites peuvent encore changer d'étiquette. Pour tous les autres pixels, leur étiquette finale reste l'étiquette prédite. La considération de cette bande d'incertitude réduit considérablement le nombre de pixels à classer, ce qui a pour effet direct d'accélérer le processus de segmentation.

La figure 1.4 présente un diagramme simplifié de ce premier schéma séquentiel après initialisation.

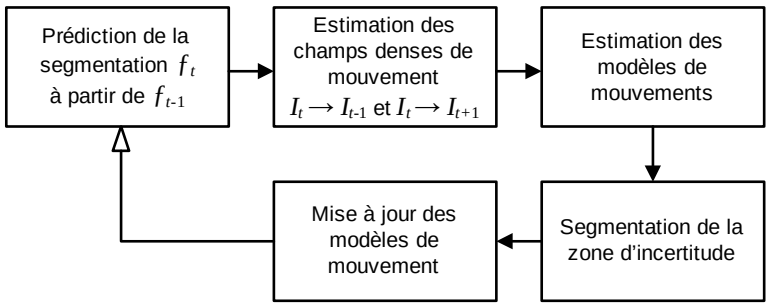


FIGURE 1.4 – Diagramme simplifié de notre premier schéma séquentiel après initialisation. La flèche blanche indique un passage au traitement de l'image suivante.

L'opérateur intervient lors de l'obtention semi-automatique de la première carte de segmentation. En cours de traitement, il peut arrêter momentanément le système

dès qu'un résultat ne correspond pas à ses attentes, corriger ou éditer la carte de segmentation erronée, et relancer le système qui reprend son traitement à l'instant où il avait été stoppé, tout en tenant compte de la carte corrigée.

Ce premier schéma a fait l'objet d'une publication [Fradet 08b] à la conférence internationale ICIP.

Second schéma basé sur des mosaïques de couche associées à des instants de référence distants de l'instant courant

Les approches les plus récentes (du moins au moment où nous avons développé ce second schéma) [Liu 05, Xiao 05b, Dupont 06b, Dupont 06c, Dupont 06a] font appel à de nombreuses mises en correspondance entre paires d'images proches sans exploiter la forte redondance des informations considérées. Partant de cette constatation, nous proposons de faire appel à un nombre réduit de mises en correspondance entre l'image courante et deux ensembles de mosaïques de couche. Un de ces ensembles de mosaïques est associé à un instant de référence t_a antérieur à l'instant courant, l'autre ensemble est associé à un instant de référence t_b postérieur à l'instant courant.

En pratique, la séquence est divisée en plusieurs intervalles temporels $[t_a, t_b]$ ne se recouvrant pas. La division peut aussi bien se faire automatiquement en intervalles de longueur constante, qu'être laissée à l'appréciation de l'opérateur qui profitera de sa connaissance de la séquence pour fixer les références sur des instants particuliers (apparition ou réapparition d'objets par exemple). Pour un intervalle donné $[t_a, t_b]$, les cartes de segmentations f_{t_a} et f_{t_b} sont obtenues indépendamment l'une de l'autre de manière semi-automatique par l'opérateur qui indique à moindre effort le nombre de couches n et la correspondance des couches d'une carte à l'autre. Grâce à ces cartes, on initialise les deux ensembles de mosaïques de couches $(M_{a,i})_{i\in[1,n]}$ et $(M_{b,i})_{i\in[1,n]}$ associés à t_a et t_b. À ce stade initial, la mosaïque $M_{a,l}$ d'une couche l associée à l'instant t_a correspond donc simplement à l'intersection de la région l de f_{t_a} avec l'image originale I_{t_a}. Il en va de même pour les mosaïques associées à l'instant t_b.

L'idée principale est ensuite de progresser de manière séquentielle de t_a à t_b tout en obtenant une succession de segmentations consistantes temporellement. Les cartes de segmentations f_{t_a} et f_{t_b} peuvent alors être perçues comme des conditions aux frontières que l'on cherche à propager au sein de l'intervalle. Alors que le processus séquentiel de segmentation progresse, le critère de mouvement mis en jeu est basé cette fois non pas sur une DFD estimée entre deux images proches, mais sur une DFD estimée entre l'image courante et la mosaïque éloignée de la couche testée. Une fois les couches de mouvement extraites, les supports de couches et les modèles de mouvement reliant l'instant courant aux deux instants de référence permettent la mise à jour des mosaïques. Cette mise à jour consiste à compléter progressivement les mosaïques en y accumulant les régions apparues à l'instant courant.

Les modèles de mouvement estimés entre images distantes sont ici encore des modèles affines à 6 paramètres. L'utilisation de ces modèles n'est pas restrictive dans la mesure où les intervalles traités restent relativement courts (de l'ordre de quelques dizaines d'images) ce qui est compatible avec notre contexte de post-production.

La figure 1.5 présente un diagramme simplifié de ce second schéma séquentiel après initialisation. La figure 1.6 illustre, sur une séquence composée de deux couches de mouvement, la mise en correspondance entre l'image courante à segmenter I_t et les deux ensembles de mosaïques $(M_{a,i})_{i\in[1,2]}$ et $(M_{b,i})_{i\in[1,2]}$.

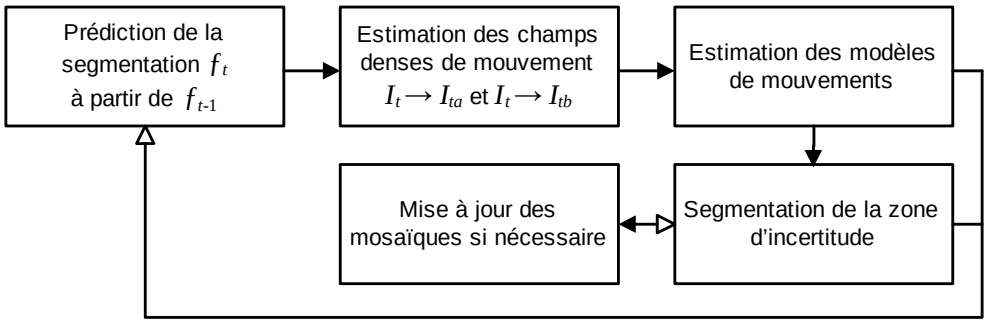


FIGURE 1.5 – Diagramme simplifié de notre second schéma séquentiel après initialisation.

Ce schéma présente plusieurs avantages.

1. Les mouvements mis en jeu sont estimés entre images distantes et il est vraisemblablement plus facile de les différencier d'une région à une autre que quand ils sont estimés entre des images consécutives.
2. Comparé aux approches existantes mentionnées précédemment, nous avons réduit le nombre de paires d'images pour lesquelles une mise en correspondance doit être effectuée.
3. Les ensembles de mosaïques contiennent à eux deux autant, voire plus, d'informations que n'en contiennent une paire ou un petit groupe d'images puisqu'on y accumule tous les éléments qui ont été visibles dans les images déjà segmentées, éléments qui peuvent avoir disparu, mais qui peuvent également réapparaître. Ainsi avec une unique mise en correspondance image-mosaïque on peut faire aussi bien, voire mieux, qu'avec de multiples mises en correspondance image-image.
4. En cas d'erreur de classification, les mosaïques générées peuvent être corrigées par l'opérateur et réutilisées lors d'une deuxième passe de segmentation pendant laquelle les corrections appliquées manuellement sur les mosaïques sont automatiquement répercutées sur toutes les cartes de segmentation de l'intervalle considéré.
5. Enfin, les mosaïques générées peuvent être ensuite réutilisées pour des applications diverses, par exemple pour des tâches de compression ou pour remplir une région d'arrière-plan après suppression d'un objet vidéo.

Ce second schéma a fait l'objet d'une publication [Fradet 08a] à la conférence internationale ECCV.

Nous proposons un résumé « graphique » du contenu de ce premier chapitre dans la figure 1.7.

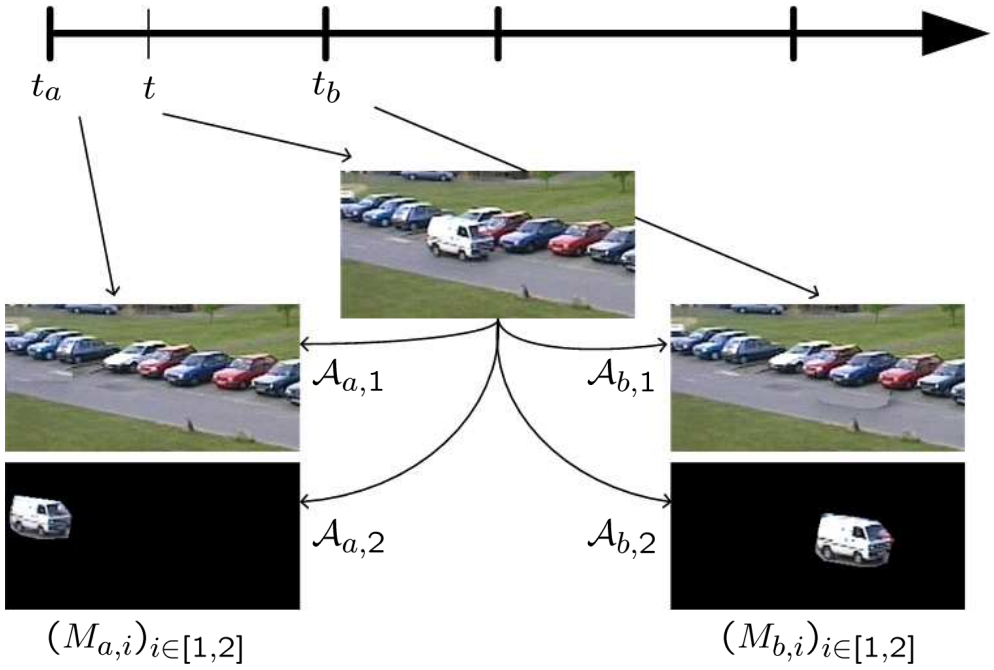
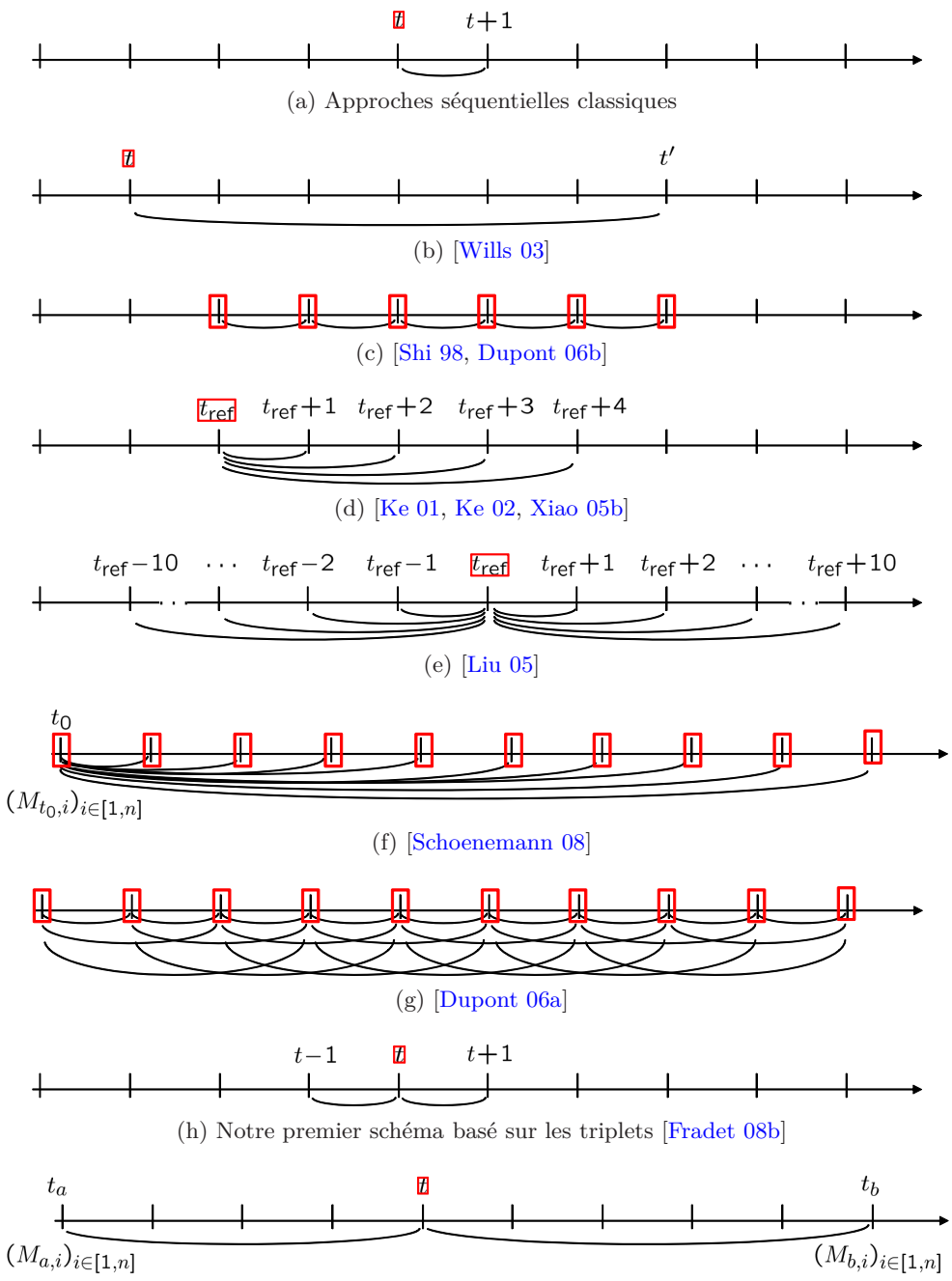


FIGURE 1.6 – Mise en correspondance de l'image courante à segmenter et des deux ensembles de mosaïques de couche associées aux instants de référence t_a et t_b. Les modèles affines de mouvement estimés pour chaque couche de l'image courante I_t vers les images de référence I_{t_a} et I_{t_b} sont notés $\mathcal{A}_{a,i}$ et $\mathcal{A}_{b,i}$. Dans cet exemple, la séquence peut être décomposée en deux couches de mouvement : l'arrière-plan et la camionnette blanche en mouvement. Cette séquence est issue de la base de données de vidéo surveillance du workshop *Performance Evaluation of Tracking and Surveillance (PETS) 2001*, Université de Reading, UK.



(a) Approches séquentielles classiques

(b) [Wills 03]

(c) [Shi 98, Dupont 06b]

(d) [Ke 01, Ke 02, Xiao 05b]

(e) [Liu 05]

(f) [Schoenemann 08]

(g) [Dupont 06a]

(h) Notre premier schéma basé sur les triplets [Fradet 08b]

(i) Notre second schéma basé sur les mosaïques [Fradet 08a]

FIGURE 1.7 – Résumé des différents types d'approches. Le cadre rouge indique l'instant pour lequel la carte de segmentation est obtenue. Pour les lignes (c), (f) et (g), toutes les cartes de segmentation des instants contenus dans l'intervalle considéré sont obtenues simultanément.

Chapitre 2

Minimisation de l'énergie d'un champ de Markov par coupe minimale/flot maximal

Les algorithmes de coupe minimale/flot maximal dans un graphe ont été introduits en vision par ordinateur en 1989 par Greig *et al.* [Greig 89]. Ils ont cependant été oubliés pendant une dizaine d'années avant d'être découverts à nouveau et de connaître un succès réel pour des applications aussi variées par exemple que la restauration d'image, la segmentation avec ou sans *a priori*, la stéréovision ou la synthèse de texture. Ce succès tient au fait que les algorithmes s'avèrent être très efficaces pour trouver, sous certaines conditions mais en un temps polynomial, le minimum global de nombreuses fonctions d'énergie fréquemment utilisées dans les problèmes de vision par ordinateur. L'approche s'appuie sur la théorie des graphes et consiste à calculer un flot au travers des nœuds et des arcs d'un graphe.

2.1 Notations et définitions relatives aux graphes

Les notations introduites par Boykov et Kolmogorov [Boykov 04] étant sans doute les plus répandues pour les problèmes de minimisation d'une fonction d'énergie en vision, nous les reprenons ici.

Définition 3. *Un graphe orienté pondéré* $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$ *se compose d'un ensemble de nœuds ou sommets* $\mathcal{V}$ *et d'un ensemble d'arcs ou arêtes* $\mathcal{E} \subset \mathcal{V}^2$.

On distingue deux nœuds particuliers dits « terminaux » appelés respectivement la « source » s et le « puits » t. (Plusieurs terminaux peuvent être ajoutés mais pour le moment nous ne considérerons que les graphes à deux terminaux correspondant au cas binaire.) Les autres nœuds sont dits « intermédiaires ». On distingue deux types d'arcs :

– les arcs terminaux « t-links », qui relient un nœud terminal et un nœud intermédiaire,

– les arcs de voisinage « n-links », qui relient entre eux deux nœuds intermédiaires. À tout arc orienté (p, q) reliant un nœud p à un nœud q on associe un « poids » positif, ou « coût », ou encore « capacité », $w(p, q) \geq 0$ qui peut être différent du poids $w(q, p)$ de l'arc inverse (q, p) lorsque celui-ci existe.

Définition 4. *Dans un graphe orienté, un « chemin » d'origine p et d'extrémité q est défini par une suite finie d'arcs consécutifs, reliant le nœud p au nœud q, ces nœuds n'étant pas nécessairement voisins.*

Définition 5. *On appelle « coupe » $C \subset \mathcal{E}$ d'un graphe orienté un ensemble d'arcs qui, une fois coupés, séparent la source du puits. C'est-à-dire que pour le graphe $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \backslash C \rangle$ il n'existe pas de chemin orienté de s à t.*

Il peut être plus clair de se représenter une coupe comme une partition $(\mathcal{S}, \mathcal{T})$ de l'ensemble des nœuds,

$$\begin{cases} \mathcal{S} \cup \mathcal{T} = \mathcal{V} \\ \mathcal{S} \cap \mathcal{T} = \emptyset \end{cases} , \tag{2.1}$$

telle que $s \in \mathcal{S}$ et $t \in \mathcal{T}$.

Une coupe C possède un coût $|C|$ défini comme la somme des poids des arcs de la coupe orientés de la source s vers le puits t :

$$|C| = \sum_{\substack{(p,q) \in \mathcal{E} \\ p \in \mathcal{S}, q \in \mathcal{T}}} w(p, q) \ . \tag{2.2}$$

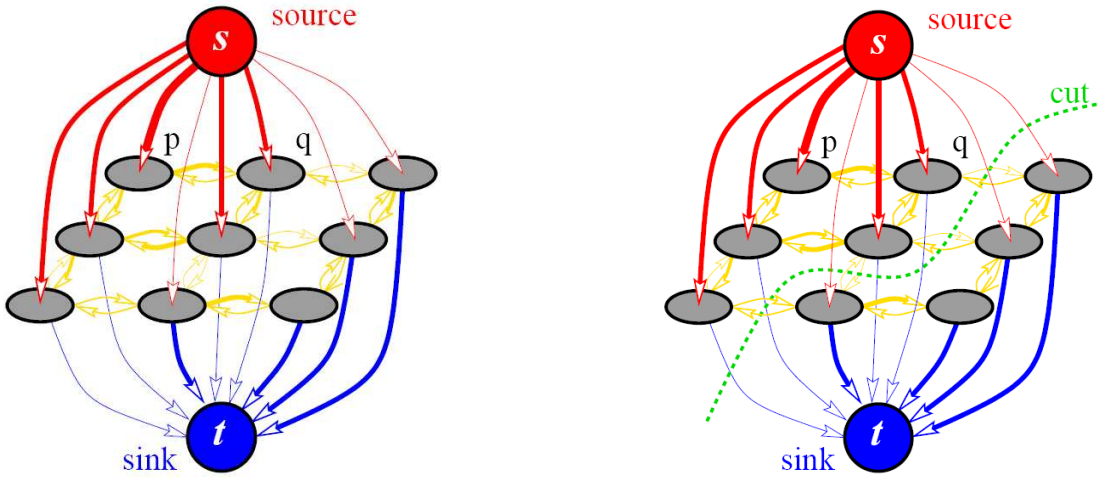
Définition 6. *Une « coupe minimale » d'un graphe est donc définie comme étant une coupe de coût minimal.*

La figure 2.1 illustre ces premières définitions.

Le formalisme de coupe dans un graphe convient aux problèmes de segmentation d'image(s). Les nœuds intermédiaires du graphe représentent généralement les pixels d'une image, mais peuvent aussi correspondre à des entités élémentaires différentes (des « macro-pixels », des régions 2D ou des éléments de volume par exemple), tandis que les n-links peuvent représenter des relations de voisinage entre ces entités élémentaires. Les nœuds terminaux, quant à eux, correspondent à l'ensemble des étiquettes qu'il est possible d'assigner aux pixels.

La partition des nœuds du graphe obtenue par une coupe correspond donc à une segmentation de l'image ou du volume considérés. Une coupe minimale génère une segmentation qui se veut optimale par rapport aux propriétés décrites par le poids des arcs. Assigner un poids faible à un arc c'est donc le rendre attractif pour la constitution d'une coupe minimale.

Dans le cas de la segmentation au niveau du pixel qui nous intéresse, le poids d'un n-link (entre deux pixels voisins) correspond à une pénalité de discontinuité entre pixels. Le poids d'un t-link (entre un pixel et un terminal) correspond à une pénalité engendrée par le fait d'assigner au pixel l'étiquette correspondant à l'autre terminal.



(a) Un graphe orienté pondéré $\mathcal{G}$ à deux terminaux

(b) Une coupe dans $\mathcal{G}$

FIGURE 2.1 – Exemple de graphe et de coupe. En gris, 9 nœuds intermédiaires. En rouge, la source s et les t-links connectés à s. En bleu, le puits t et les t-links connectés à t. En jaune, les n-links entre nœuds intermédiaires. En vert, une coupe. Le poids des arcs est proportionnel à leur épaisseur. Par souci de clarté, certains arcs n'ont pas été représentés. Par ailleurs, dans cet exemple construit sur une grille régulière à deux dimensions, la coupe devrait être représentée par une surface et non par une courbe. Elle coupe effectivement certains t-links afin que pour tout nœud intermédiaire p, un (et un seul) des deux arcs (s, p) et (p, t) soit coupé. Cette figure provient de [Boykov 04].

2.2 Équivalence entre coupe minimale et flot maximal dans un graphe

Comme on le verra, le problème de minimisation de certaines fonctions d'énergie revient à trouver une coupe de coût minimal parmi toutes les coupes possibles dans un graphe pondéré de façon appropriée. Ce problème est dual au problème mieux connu du calcul du flot maximal dans un graphe. En considérant les arcs comme des canalisations, on peut imaginer un flot de liquide qui s'écoule de la source vers le puits via des canalisations ayant chacune leur propre capacité. La quantité de liquide traversant un arc ne peut excéder la capacité de cet arc. En chaque nœud intermédiaire, une loi de conservation similaire à la loi de Kirchoff en électricité doit être vérifiée : la quantité de liquide sortant d'un nœud doit être égale à la quantité entrante.

Pour tout flot φ (fonction positive sur les arcs) respectant les propriétés énoncées précédemment, la quantité totale de liquide sortant de la source s est égale à la quantité totale de liquide entrant dans le puits t. Cette quantité $|\varphi|$ est appelée la « valeur » du flot.

Définition 7. *Un flot maximal est donc défini comme un flot dont la valeur est maxi-*

male.

L'équivalence entre flot maximal et coupe minimale découle du théorème de Folk et Fulkerson (1956) :

Théorème 1. *Le coût d'une coupe minimale est égal à la valeur d'un flot maximal.*

On peut de plus démontrer que tout arc contenu dans une coupe minimale est un arc « saturé », c'est-à-dire que la valeur du flot sur l'arc est égale à la capacité de cet arc. (De même, on dit qu'un chemin de s à t est saturé dès lors que l'un de ses arcs est saturé.)

Pour la démonstration de l'équivalence entre flot maximal et coupe minimale, on pourra se reporter par exemple à [Bugeau 07, chapitre 6].

On voit donc que si l'on sait résoudre le problème du flot maximal dans un graphe, on peut très simplement en déduire une coupe minimale : il suffit de prendre l'ensemble des arcs saturés et d'en retirer de manière itérative les arcs inutiles.

Il existe de nombreux algorithmes pour déterminer le flot maximal d'un graphe. Ces algorithmes peuvent être classés en deux catégories : algorithmes par saturation de chemins et algorithmes par poussage de flot. On ne donne ici que les exemples les plus connus et leur principe général de fonctionnement. [Schrijver 03, chapitre 10] fournit une présentation détaillée des algorithmes de flot maximal.

2.2.1 Algorithmes de flot maximal par saturation de chemins

Ces algorithmes reposent directement sur la notion de saturation. Le principe général est de partir d'un flot initial (souvent nul), puis de manière itérative :

- trouver un chemin non saturé de s à t (s'il existe encore un tel chemin, c'est que le flot n'est pas maximum et qu'il peut donc encore être augmenté),
- ajouter alors autant de flot possible à ce chemin, c'est-à-dire saturer l'arc du chemin présentant la capacité résiduelle la plus faible.

De manière intuitive, la capacité résiduelle d'un arc mesure la quantité supplémentaire de flot que l'on peut encore y faire passer sans que la quantité totale sur cet arc n'excède sa capacité. Pour un arc entre les nœuds p et q sur lequel la valeur du flot est $\varphi(p,q)$, la capacité résiduelle vaut $w(p,q) - \varphi(p,q) \geq 0$.

Au bout d'un certain nombre d'itérations de saturation, il n'existe plus de chemin non saturé. Le flot obtenu est alors un flot maximal.

Ford et Fulkerson [Ford 62] furent les premiers à proposer un algorithme fournissant une solution optimale en un temps polynomial. Les différences entre algorithmes résident dans la façon de rechercher à chaque itération un chemin non saturé.

Boykov et Kolmogorov [Boykov 04] construisent deux arbres de recherche de chemins non saturés dans le graphe $\mathcal{G}$, l'un à partir de la source, l'autre à partir du puits. La rencontre de ces deux arbres permet la détection de chemins non saturés. Après saturation d'un chemin par un flot φ, on construit le graphe résiduel $\mathcal{G}_\varphi$ du graphe $\mathcal{G}$. Ce graphe résiduel $\mathcal{G}_\varphi$ a la même topologie que $\mathcal{G}$, et la capacité de chacun de ses arcs reflète la capacité résiduelle du même arc dans $\mathcal{G}$ étant donné la valeur du flot

déjà présente sur l'arc. Plutôt que de complètement recalculer les arbres de recherche à chaque itération, ceux-ci sont conservés et modifiés dans le graphe résiduel après mise à jour de celui.

Boykov *et al.* [Boykov 01a] ont proposé deux algorithmes itératifs très utilisés pour leur faible temps de calcul observé qui évolue de manière quasiment linéaire par rapport à la taille des données pour des graphes à deux dimensions. Une description plus détaillée de chacun des deux algorithmes est donnée à la section 2.3.2.

2.2.2 Algorithmes de flot maximal par poussage de flot

Les algorithmes de ce type fonctionnent avec des « pré-flots » : des flots qui, volontairement, ne respectent pas la condition de conservation. Le principe général est de faire sortir de la source s autant de flot que possible, ce qui implique naturellement que certains nœuds reçoivent un excès de flot. On cherche donc à envoyer le flot en excès vers d'autres nœuds voisins disponibles.

Le plus célèbre de ces algorithmes est l'algorithme de « Push relabel » [Goldberg 86].

2.2.3 Algorithmes récents de « Dynamic Graph Cuts » et « Active Cuts »

Ces deux algorithmes visent à accélérer les temps de calcul de flot maximal.

L'algorithme de « Dynamic Graph Cuts » est développé par Kohli et Torr [Kohli 05, Kohli 07]. Connaissant le flot maximal dans un graphe donné, l'algorithme dynamique permet de calculer rapidement le flot maximal dans une version modifiée du graphe, où seuls les poids de certaines arêtes changent. Les auteurs ont utilisé cette technique pour résoudre le problème binaire de séparation d'un objet d'avant-plan et du fond dans une vidéo. D'une image à l'autre, les données varient peu, et la solution du problème de segmentation à la première image est utilisée pour accélérer les calculs pour la seconde image.

Plutôt que de réutiliser les flots, Juan et Boykov [Juan 06] réutilisent les coupes. Leur méthode d'« Active Cuts » permet en effet de calculer rapidement une coupe minimale en partant d'une autre coupe. Cette technique se montre particulièrement efficace dans un cadre multi-échelle.

2.3 Minimisation d'une fonction d'énergie

Pour la suite, les nœuds terminaux seront appelés simplement « terminaux » et le terme « nœuds » ne désignera donc plus que les nœuds intermédiaires.

De nombreux problèmes de vision par ordinateur peuvent être vus comme des problèmes d'étiquetage, eux-mêmes pouvant être formulés comme des problèmes de minimisation d'une fonction d'énergie. C'est le cas notamment de problèmes de segmentation, de restauration, et de stéréovision.

Bien souvent, l'étiquetage se fait au niveau du pixel. On se placera donc pour la suite dans le cas où lors de la construction du graphe, on crée un nœud pour chaque pixel

p à étiqueter. Il est toutefois bon de noter que créer un nœud par pixel peut mener, dans certains cas, à des graphes très denses. Dans ces cas précis, il peut être intéressant pour alléger le graphe que l'étiquetage se fasse au niveau d'entités élémentaires à deux ou trois dimensions (« macro-pixels », régions 2D, éléments de volume).

Soit la fonction d'étiquetage $f = (f_{\mathbf{p}})_{\mathbf{p}\in\mathcal{P}}$ qui à tout pixel $\mathbf{p}$ de l'ensemble $\mathcal{P}$ des pixels à étiqueter associe une étiquette $f_{\mathbf{p}} = f(\mathbf{p})$ appartenant à l'ensemble $\mathcal{L}$ des étiquettes possibles. Dans le cas d'un graphe $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$ à deux terminaux s et t, l'ensemble des pixels à étiqueter vaut $\mathcal{P} = \mathcal{V} \backslash \{s, t\}$.

2.3.1 Cas d'une classification binaire

Nous allons nous concentrer pour le moment sur le cas binaire où le graphe ne comporte que deux terminaux, une source s et un puits t, qui par convention sont respectivement associés à 0 et 1, les deux seules étiquettes possibles. Ainsi, $\mathcal{L} = \{0, 1\}$. Nous généraliserons plus tard au cas à étiquettes multiples (sous-section 2.3.2).

La fonction d'énergie considérée est généralement du type :

$$E(f) = E_{\text{données}}(f) + E_{\text{lissage}}(f) \ . \tag{2.3}$$

Elle se compose de deux termes :

– le premier terme est un terme d'attache aux données. Il s'écrit généralement :

$$E_{\text{données}}(f) = \sum_{\mathbf{p}\in\mathcal{P}} D_{\mathbf{p}}(f_{\mathbf{p}}) \ . \tag{2.4}$$

La fonction $D_{\mathbf{p}}(.)$ est une fonction de pénalité qui, pour chaque nœud, indique dans quelle mesure l'étiquetage correspond aux données observées. Plus la valeur de $D_{\mathbf{p}}(f_{\mathbf{p}})$ est élevée, plus il est coûteux d'associer l'étiquette $f_{\mathbf{p}}$ au nœud $\mathbf{p}$, et donc moins il est vraisemblable que l'étiquette $f_{\mathbf{p}}$ soit celle qu'il convient d'assigner à $\mathbf{p}$. Ce terme est dit unaire car il n'implique qu'une seule variable inconnue.

– le second terme est un terme de régularisation. Il s'écrit généralement :

$$E_{\text{lissage}}(f) = \sum_{\{\mathbf{p},\mathbf{q}\}\in\mathcal{C}} V_{\mathbf{p},\mathbf{q}}(f_{\mathbf{p}}, f_{\mathbf{q}}) \ . \tag{2.5}$$

$V_{\mathbf{p},\mathbf{q}}$ est un potentiel d'interaction incitant une paire de nœuds voisins à prendre la même étiquette. $\mathcal{C}$ est l'ensemble des cliques d'ordre 2 : $\mathcal{C} = \{\{\mathbf{p}, \mathbf{q}\} \subset \mathcal{P} : (\mathbf{p}, \mathbf{q}) \in \mathcal{N}\}$, où $\mathcal{N}$ est l'ensemble des n-links, c'est-à-dire $\mathcal{E}$ privé de l'ensemble des t-links. Ce terme de régularisation est dit binaire car il implique une paire de variables inconnues.

Le potentiel d'interaction est généralement de la forme :

$$V_{\mathbf{p},\mathbf{q}}(f_{\mathbf{p}}, f_{\mathbf{q}}) = \lambda A_{\mathbf{p},\mathbf{q}}(f_{\mathbf{p}}, f_{\mathbf{q}}) T(f_{\mathbf{p}} \neq f_{\mathbf{q}}) \ , \tag{2.6}$$

où $A(\cdot)$ est la fonction d'interaction, $T(\cdot)$ vaut 1 si le prédicat en argument est vrai et 0 sinon, et λ est une constante réelle positive qui règle la pondération

relative du terme de lissage par rapport au terme d'attache aux données, et donc la régularité de la solution obtenue. Plus cette constante est élevée, plus il devient coûteux de couper des n-links, ce qui encourage à n'en couper qu'un faible nombre et donc à obtenir des régions de segmentation plus grandes et des frontières plus régulières.

En segmentation, le modèle le plus simple utilisé pour modéliser les interactions locales entre variables à deux états est le célèbre modèle d'Ising :

$$\forall(\mathbf{p},\mathbf{q}) \in \mathcal{N} \qquad (f_\mathbf{p}, f_\mathbf{q}) \in \{0,1\}^2 \qquad V_{\mathbf{p},\mathbf{q}}(f_\mathbf{p}, f_\mathbf{q}) = \lambda T(f_\mathbf{p} \neq f_\mathbf{q}) \ . \tag{2.7}$$

Ce modèle encourage une régularité sur toute la région à classer. Il est pourtant souvent nécessaire de relâcher cette contrainte dans les zones de fort gradient de manière à favoriser la coupe dans ces zones [Boykov 01b, Rother 04] :

$$\forall(\mathbf{p},\mathbf{q}) \in \mathcal{N} \qquad V_{\mathbf{p},\mathbf{q}}(f_\mathbf{p}, f_\mathbf{q}) = \frac{\lambda}{\text{dist}(\mathbf{p},\mathbf{q})} \exp(-\frac{\|I_t(\mathbf{p}) - I_t(\mathbf{q})\|^2}{2\sigma^2}) \cdot T(f_\mathbf{p} \neq f_\mathbf{q}) \ , \tag{2.8}$$

où dist(.) est la distance euclidienne et σ est l'écart type des normes des gradients sur toute l'image courante I_t.

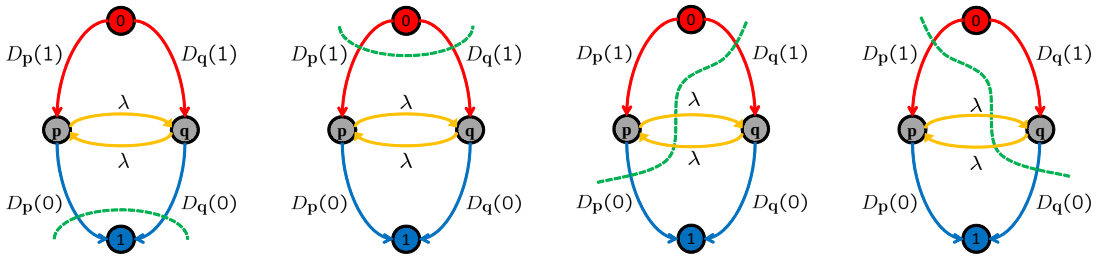
Lors de la construction du graphe, l'assignation d'un poids à chaque arc doit se faire de sorte à impliquer l'égalité entre le coût d'une coupe et l'énergie de l'étiquetage associé (à une constante près). Ceci garantit que l'étiquetage associé au minimum global d'énergie est fourni par la coupe minimale du graphe.

Pour le cas d'une seule variable $\mathbf{p}$, si $\mathbf{p} \in \mathcal{S}$ alors $f_\mathbf{p} = 0$, donc le t-link reliant $\mathbf{p}$ à t est coupé et le poids qui lui est associé est nécessairement $D_\mathbf{p}(0)$. Inversement, si $\mathbf{p} \in \mathcal{T}$ alors $f_\mathbf{p} = 1$, donc le t-link reliant $\mathbf{p}$ à s est coupé et le poids qui lui est associé est nécessairement $D_\mathbf{p}(1)$.

Pour tout le document nous adoptons donc la convention selon laquelle un nœud $\mathbf{p}$ prend l'étiquette 0 (resp. 1) lorsque la coupe le sépare du terminal 1 (resp. 0). Cette convention implique, comme on vient de le préciser, d'affecter le poids $D_\mathbf{p}(0)$ (resp. $D_\mathbf{p}(1)$) à l'arc reliant $\mathbf{p}$ au terminal 1 (resp. 0). Il est toutefois bon de remarquer que l'on trouve parfois la convention inverse dans la littérature : par exemple dans [Boykov 01a], un nœud $\mathbf{p}$ prend l'étiquette 0 (resp. 1) lorsque la coupe le sépare du terminal 0 (resp. 1). Cette convention inverse semble moins naturelle du point de vue de l'étiquetage mais a le bon goût de correspondre à des affectations d'arcs plus instinctives : le poids $D_\mathbf{p}(0)$ (resp. $D_\mathbf{p}(1)$) est affecté directement à l'arc reliant $\mathbf{p}$ au terminal 0 (resp. 1).

Pour le cas de plusieurs variables avec interactions binaires, selon la définition d'une coupe et l'équation (2.2), le coût d'une coupe C est donné par :

$$|C| = \sum_{\substack{(i,j)\in\mathcal{E} \\ i\in\mathcal{S}, j\in\mathcal{T}}} w(i,j) = \sum_{\substack{(i,j)\in\mathcal{E} \\ i=s, j\in\mathcal{T}\setminus\{t\}}} w(i,j) + \sum_{\substack{(i,j)\in\mathcal{E} \\ i\in\mathcal{S}\setminus\{s\}, j=t}} w(i,j) + \sum_{\substack{(i,j)\in\mathcal{E} \\ i\in\mathcal{S}\setminus\{s\}, j\in\mathcal{T}\setminus\{t\}}} w(i,j) \ . \tag{2.9}$$



(a) $f_{\mathbf{p}} = 0$, $f_{\mathbf{q}} = 0$, $|\mathcal{C}| = D_{\mathbf{p}}(0) + D_{\mathbf{q}}(0)$

(b) $f_{\mathbf{p}} = 1$, $f_{\mathbf{q}} = 1$, $|\mathcal{C}| = D_{\mathbf{p}}(1) + D_{\mathbf{q}}(1)$

(c) $f_{\mathbf{p}} = 0$, $f_{\mathbf{q}} = 1$, $|\mathcal{C}| = D_{\mathbf{p}}(0) + \lambda + D_{\mathbf{q}}(1)$

(d) $f_{\mathbf{p}} = 1$, $f_{\mathbf{q}} = 0$, $|\mathcal{C}| = D_{\mathbf{p}}(1) + \lambda + D_{\mathbf{q}}(0)$

FIGURE 2.2 – Graphe, poids, et coupes possibles dans le cas de la classification binaire de deux variables $\mathbf{p}$ et $\mathbf{q}$ en utilisant comme modèle d'interaction le modèle d'Ising.

En passant à des notations relatives aux pixels :

$$|C| = \sum_{\mathbf{p}\in\mathcal{T}\setminus\{t\}} w(s,\mathbf{p}) + \sum_{\mathbf{p}\in\mathcal{S}\setminus\{s\}} w(\mathbf{p},t) + \sum_{\substack{(\mathbf{p},\mathbf{q})\in\mathcal{E}\\ \mathbf{p}\in\mathcal{S}\setminus\{s\},\mathbf{q}\in\mathcal{T}\setminus\{t\}}} w(\mathbf{p},\mathbf{q}) \quad . \tag{2.10}$$

Or on vient de déduire du cas d'une seule variable que pour un pixel $\mathbf{p}$, le poids des t-links reliant la source à $\mathbf{p}$ et $\mathbf{p}$ au puits sont définis respectivement par $w(s,\mathbf{p}) = D_{\mathbf{p}}(1)$ et $w(\mathbf{p},t) = D_{\mathbf{p}}(0)$. En posant alors $w(\mathbf{p},\mathbf{q}) = \lambda A_{\mathbf{p},\mathbf{q}}(f_{\mathbf{p}}, f_{\mathbf{q}})$ et en rappelant que $\mathbf{p} \in \mathcal{T}\setminus\{t\} \Leftrightarrow f_{\mathbf{p}} = 1$ et $\mathbf{p} \in \mathcal{S}\setminus\{s\} \Leftrightarrow f_{\mathbf{p}} = 0$,

$$\begin{aligned}
|C| &= \sum_{\mathbf{p}\in\mathcal{T}\setminus\{t\}} D_{\mathbf{p}}(1) + \sum_{\mathbf{p}\in\mathcal{S}\setminus\{s\}} D_{\mathbf{p}}(0) + \sum_{\substack{(\mathbf{p},\mathbf{q})\in\mathcal{E}\\ \mathbf{p}\in\mathcal{S}\setminus\{s\},\mathbf{q}\in\mathcal{T}\setminus\{t\}}} \lambda A_{\mathbf{p},\mathbf{q}}(f_{\mathbf{p}}, f_{\mathbf{q}}) \\
&= \sum_{\mathbf{p}\in\mathcal{T}\setminus\{t\}\cup S\setminus\{s\}} D_{\mathbf{p}}(f_{\mathbf{p}}) + \sum_{\substack{(\mathbf{p},\mathbf{q})\in\mathcal{N}\\ \mathbf{p}\in\mathcal{S}\setminus\{s\},\mathbf{q}\in\mathcal{T}\setminus\{t\}}} \lambda A_{\mathbf{p},\mathbf{q}}(f_{\mathbf{p}}, f_{\mathbf{q}}) + \sum_{\substack{(\mathbf{p},\mathbf{q})\in\mathcal{N}\\ \mathbf{p}\notin\mathcal{S}\setminus\{s\} \text{ ou } \mathbf{q}\notin\mathcal{T}\setminus\{t\}}} 0 \\
&= \sum_{\mathbf{p}\in\mathcal{P}} D_{\mathbf{p}}(f_{\mathbf{p}}) + \sum_{(\mathbf{p},\mathbf{q})\in\mathcal{N}} \lambda A_{\mathbf{p},\mathbf{q}}(f_{\mathbf{p}}, f_{\mathbf{q}}) T(f_{\mathbf{p}} = 0, f_{\mathbf{q}} = 1) \\
&= \sum_{\mathbf{p}\in\mathcal{P}} D_{\mathbf{p}}(f_{\mathbf{p}}) + \sum_{\{\mathbf{p},\mathbf{q}\}\in\mathcal{C}} \lambda A_{\mathbf{p},\mathbf{q}}(f_{\mathbf{p}}, f_{\mathbf{q}}) T(f_{\mathbf{p}} \neq f_{\mathbf{q}}) \\
&= E_{\text{données}}(f) + E_{\text{lissage}}(f) = E(f)
\end{aligned} \tag{2.11}$$

Dans le cas de deux variables $\mathbf{p}$ et $\mathbf{q}$ avec interaction modélisée par le modèle d'Ising, la figure 2.2 indique quel poids assigner à chaque arc d'un graphe afin que la valeur de la coupe minimale du graphe soit égale au minimum global de la fonction d'énergie considérée (en l'occurrence, $w(\mathbf{p},\mathbf{q}) = w(\mathbf{q},\mathbf{p}) = \lambda$).

Cependant on ne peut pas représenter toutes les fonctions d'énergie du type de celle de l'équation (2.3)-(2.5) par un graphe. Pour tout arc du graphe, le poids associé doit

être positif, or Kolmogorov et Zabih [Kolmogorov 04] ont montré que ceci impliquait une condition particulière sur le terme de lissage.

Théorème 2. *Soit E une fonction d'énergie du type :*

$$E(f) = \sum_{\mathbf{p}\in\mathcal{P}} D_{\mathbf{p}}(f_{\mathbf{p}}) + \sum_{(\mathbf{p},\mathbf{q})\in\mathcal{N}} V_{\mathbf{p},\mathbf{q}}(f_{\mathbf{p}}, f_{\mathbf{q}}) \ , \tag{2.12}$$

alors cette fonction peut être représentée par un graphe si et seulement si chaque terme $V_{\mathbf{p},\mathbf{q}}$ vérifie l'inégalité :

$$V_{\mathbf{p},\mathbf{q}}(0,0) + V_{\mathbf{p},\mathbf{q}}(1,1) \leq V_{\mathbf{p},\mathbf{q}}(0,1) + V_{\mathbf{p},\mathbf{q}}(1,0) \ . \tag{2.13}$$

Si c'est le cas, trouver une coupe minimale dans le graphe permet d'obtenir un minimum global de la fonction d'énergie.

Définition 8. *Une fonction $V_{\mathbf{p},\mathbf{q}}(.)$ vérifiant l'inégalité (2.13) est dite « régulière » ou « sous-modulaire ».*

On vérifie facilement que les fonctions définies aux équations (2.7) et (2.8) sont régulières. Toute fonction de la forme $A_{\mathbf{p},\mathbf{q}}(f_{\mathbf{p}}, f_{\mathbf{q}})T(f_{\mathbf{p}} \neq f_{\mathbf{q}})$, avec $A_{\mathbf{p},\mathbf{q}}(f_{\mathbf{p}}, f_{\mathbf{q}}) \geq 0$ l'est.

2.3.2 Cas d'une classification multi-étiquettes

Nous considérons maintenant la minimisation d'une fonction d'énergie dans le cas d'une classification multi-étiquettes. Désormais l'ensemble $\mathcal{L}$ des étiquettes possibles s'écrit $\mathcal{L} = \{1, \cdots, n\}$, et pour tout $\mathbf{p} \in \mathcal{P}$ on a $f_{\mathbf{p}} \in \{1, \cdots, n\}$.

Dans ces problèmes multi-étiquettes, l'étiquette peut par exemple représenter une disparité dans le cas de la stéréovision ou un indice de couche dans le cas de la segmentation en couches de mouvement.

Pour la stéréovision, la carte de disparité est retrouvée comme une hypersurface. Roy et Cox [Roy 98] considèrent un graphe $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$ formant un maillage à trois dimensions. Pour l'ensemble des nœuds $\mathcal{V} = \mathcal{V}^* \cup \{s, t\}$, $\mathcal{V}^*$ est défini comme :

$$\mathcal{V}^* = \{(x, y, d) : x \in [0 \dots x_{\max}], y \in [0 \dots y_{\max}], d \in [0 \dots d_{\max}]\} \ , \tag{2.14}$$

pour une image de dimension $x_{\max} + 1 \times y_{\max} + 1$, et où $d_{\max} + 1 (= n)$ est donc la résolution en profondeur. La notion d'ordonnancement des étiquettes est traduite par une 6-connexité du maillage. Par ailleurs la source s est connectée à l'avant-plan et le puits t est connecté à l'arrière-plan. Ainsi l'ensemble des arcs $\mathcal{E}$ est défini par :

$$\begin{aligned}\mathcal{E} = \{(u, v) \in \mathcal{V}^* \times \mathcal{V}^* : \|u - v\| = 1\} \cup \{(s, (x, y, 0)) : x \in [0 \dots x_{\max}], y \in [0 \dots y_{\max}]\} \\ \cup \{((x, y, d_{\max}), t) : x \in [0 \dots x_{\max}], y \in [0 \dots y_{\max}]\} \ .\end{aligned} \tag{2.15}$$

Ishikawa [Ishikawa 03] construit le graphe de la même façon (voir la figure 2.3 pour un exemple simple monodimensionnel avec 4 pixels et trois étiquettes possibles) et

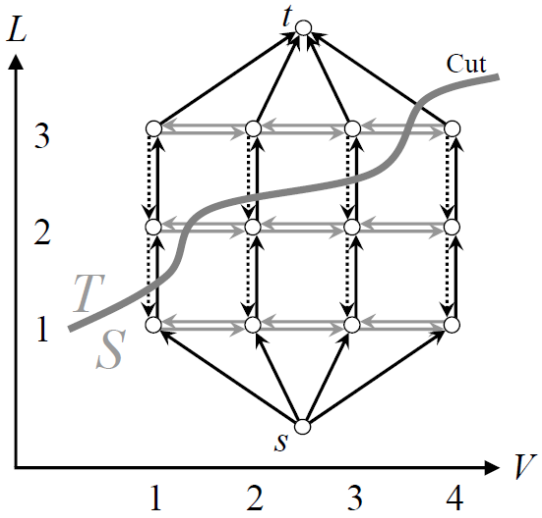


FIGURE 2.3 – Graphe considéré par Ishikawa pour la classification multi-étiquettes de 4 pixels. La coupe correspond aux assignations suivantes : $f(\mathbf{p}_1) = 1, f(\mathbf{p}_2) = 2, f(\mathbf{p}_3) = 2, f(\mathbf{p}_4) = 3$. Cette figure provient de [Ishikawa 03].

montre que la positivité des arêtes dans une telle construction impose que la fonction de régularisation soit convexe.

Dès que les termes d'interaction utilisés visent à préserver les discontinuités, la fonction de régularisation n'étant alors plus convexe, cette forme de graphe ne peut plus être utilisée.

Pour la segmentation, on s'intéresse à nouveau à des fonctions d'énergie où les contraintes de régularité pénalisent les régions de petites tailles tout en préservant les discontinuités. Le modèle de Potts [Potts 52] (2.16) généralise dans le cas non binaire le modèle d'Ising (2.7) évoqué précédemment :

$$\forall(\mathbf{p},\mathbf{q}) \in \mathcal{N} \qquad (f_\mathbf{p}, f_\mathbf{q}) \in \{1,\cdots,n\}^2 \qquad V_{\mathbf{p},\mathbf{q}}(f_\mathbf{p}, f_\mathbf{q}) = \lambda T(f_\mathbf{p} \neq f_\mathbf{q}) \ , \qquad (2.16)$$

où $T(.)$ vaut 1 si le prédicat en argument est vrai et 0 sinon.

L'expression de la fonction d'énergie globale donnée par l'équation (2.12) est toujours valable dans le cas multi-étiquettes. Cependant la minimisation de ce type de fonctions d'énergie est désormais un problème NP-complet pour lequel il n'existe donc pas d'algorithme permettant de trouver une solution optimale en un temps polynomial. Il existe néanmoins des algorithmes [Boykov 01a] capables de trouver rapidement une bonne approximation (minimum local) de la solution optimale, moyennant de nouvelles contraintes sur la fonction de régularisation.

L'idée principale des deux algorithmes itératifs décrits ci-après est de se ramener à une succession de problèmes binaires. Chaque algorithme repose sur un type de « mouvement » permettant à chaque itération de passer d'un ancien étiquetage f à un nouvel

étiquetage f'. À chaque itération, ces « grands » mouvements peuvent modifier simultanément les étiquettes d'un ensemble de nœuds de taille arbitraire, contrairement à d'autres algorithmes de minimisation dont les mouvements plus standard permettent de modifier l'étiquette d'un seul nœud à la fois (« simulated annealing » (recuit simulé) [Kirkpatrick 83] ou « Iterated Conditional Modes » (ICM) [Besag 86] par exemple). Les deux algorithmes convergent rapidement, les premières itérations étant celles qui correspondent aux plus fortes décroissances d'énergie.

Les fonctions de régularisation pouvant être utilisées par ces algorithmes appartiennent à deux classes assez générales : la classe des métriques et la classe des semi-métriques.

Définition 9. *Une fonction V est une métrique sur l'espace des étiquettes $\mathcal{L}$ si pour toutes étiquettes α, β, γ de $\mathcal{L}$, elle vérifie :*

$$V(\alpha,\beta) = 0 \quad \Leftrightarrow \quad \alpha = \beta \ , \tag{2.17}$$

$$V(\alpha,\beta) = V(\beta,\alpha) \geq 0 \ , \tag{2.18}$$

$$V(\alpha,\beta) \leq V(\alpha,\gamma) + V(\gamma,\beta) \ . \tag{2.19}$$

Définition 10. *Une fonction V vérifiant seulement (2.17) et (2.18) est une semi-métrique.*

L'alpha-expansion

Cet algorithme suppose que la fonction de régularisation est une métrique et qu'elle vérifie donc l'inégalité triangulaire (2.19). L'algorithme d'alpha-expansion repose sur le mouvement d'extension. À chaque itération, étant donnée une étiquette particulière $\alpha \in \mathcal{L}$, seuls les nœuds $\mathbf{p}$ d'ancienne étiquette $f_\mathbf{p} \neq \alpha$ sont autorisés à changer d'étiquette pour la nouvelle étiquette $f'_\mathbf{p} = \alpha$ si ce changement entraîne une diminution de l'énergie. Tous les pixels déjà étiquetés α le restent ($(f_\mathbf{p} = \alpha) \Rightarrow (f'_\mathbf{p} = \alpha)$). Le nombre de nœuds étiquetés α ne peut donc qu'augmenter au cours de cette itération, ce qui justifie le nom donné au mouvement : α-expansion. À l'itération suivante, on choisira une autre étiquette particulière α. L'ordre des étiquettes importe peu, il peut être fixé ou aléatoire. La figure 2.4 illustre le fonctionnement de cet algorithme d'alpha-expansion.

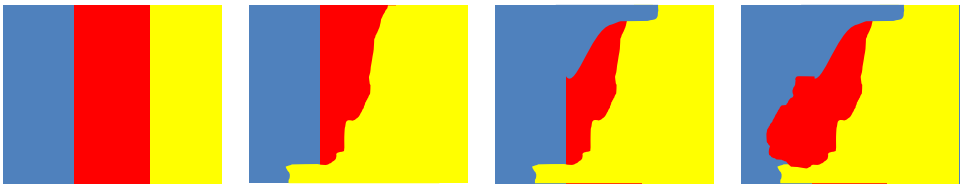


FIGURE 2.4 – Mouvement d'α-expansion permettant à un grand nombre de pixels d'abandonner simultanément leur ancienne étiquette au profit de la nouvelle étiquette α. De gauche à droite : étiquetage initial ; première itération, jaune-extension ; deuxième itération, bleu-extension ; troisième itération, rouge-extension.

Algorithme 1 Algorithme d'alpha-expansion

1: Commencer avec un étiquetage f arbitraire
2: baisse $\leftarrow 0$
3: **pour** chaque étiquette $\alpha \in \mathcal{L}$ **faire**
4: Trouver $\hat{f} = \text{argmin} E(f')$ parmi tous les étiquetages f' pouvant être obtenus par α-expansion de f
5: **si** $E(\hat{f}) < E(f)$ **alors**
6: $f \leftarrow \hat{f}$ et baisse $\leftarrow 1$
7: **fin si**
8: **fin pour**
9: **si** baisse $= 1$ **alors**
10: aller à 2
11: **fin si**
12: **return** f

Dans l'algorithme d'alpha-expansion (algo. 1), il est bon de noter qu'on différencie une « itération » d'un « cycle » [1].

Définition 11. *Une itération est une exécution des étapes 4 à 7.*

Définition 12. *Un cycle est une exécution des étapes 3 à 11.*

Un cycle de l'algorithme d'alpha-expansion comprend donc $|\mathcal{L}|$ itérations. L'algorithme s'arrête si aucune itération d'un cycle complet n'a fourni de meilleur étiquetage (au sens d'une diminution de l'énergie).

La structure du graphe est déterminée de manière dynamique en fonction de l'étiquetage f courant et de l'étiquette α. À chaque itération, on construit un graphe à deux terminaux : l'un pour α associé par convention à l'étiquette 0 et l'autre pour $\bar{\alpha}$ associé à l'étiquette 1. Un nœud est créé pour chaque élément à classer quelle que soit son étiquette courante. On se ramène ainsi à une succession de problèmes binaires : est-il préférable qu'un nœud $\mathbf{p}$ garde son ancienne étiquette $f_{\mathbf{p}}$ ou bien qu'il prenne la nouvelle étiquette α ?

Pour résoudre de manière optimale chaque problème binaire, la fonction de régularisation doit vérifier la contrainte de régularité (2.13) :

$$\begin{aligned} & \widetilde{V}_{\mathbf{p},\mathbf{q}}(0,0) + \widetilde{V}_{\mathbf{p},\mathbf{q}}(1,1) \leq \widetilde{V}_{\mathbf{p},\mathbf{q}}(0,1) + \widetilde{V}_{\mathbf{p},\mathbf{q}}(1,0) \\ \Rightarrow\ & V_{\mathbf{p},\mathbf{q}}(\alpha,\alpha) + V_{\mathbf{p},\mathbf{q}}(f_{\mathbf{p}},f_{\mathbf{q}}) \leq V_{\mathbf{p},\mathbf{q}}(\alpha,f_{\mathbf{q}}) + V_{\mathbf{p},\mathbf{q}}(f_{\mathbf{p}},\alpha) \\ \Rightarrow\ \forall(\alpha,\beta,\gamma) \in \mathcal{L}^3 \qquad & V_{\mathbf{p},\mathbf{q}}(\alpha,\alpha) + V_{\mathbf{p},\mathbf{q}}(\beta,\gamma) \leq V_{\mathbf{p},\mathbf{q}}(\alpha,\gamma) + V_{\mathbf{p},\mathbf{q}}(\beta,\alpha) \ , \end{aligned} \tag{2.20}$$

ce qui est vérifié car $V_{\mathbf{p},\mathbf{q}}(.)$ est une métrique.

Le minimum local obtenu par l'algorithme d'alpha-expansion est borné par un multiple connu du minimum global.

1. Les définitions d'« itération » et de « cycle » restent valable pour l'algorithme d'alpha-beta-swap (algo. 2).

Théorème 3. *Soit $\hat{f}$ l'étiquetage correspondant à un minimum local de la fonction d'énergie obtenu par l'algorithme d'alpha-expansion et f^* l'étiquetage optimal, alors :*

$$E(\hat{f}) \leq 2 \cdot c \cdot E(f^*) \ , \tag{2.21}$$

où c est donné par :

$$c = \max_{(\mathbf{p},\mathbf{q})\in\mathcal{N}} \left(\frac{\max_{\substack{(\alpha,\beta)\in\mathcal{L}^2 \\ \alpha\neq\beta}} V_{\mathbf{p},\mathbf{q}}(\alpha,\beta)}{\min_{\substack{(\alpha,\beta)\in\mathcal{L}^2 \\ \alpha\neq\beta}} V_{\mathbf{p},\mathbf{q}}(\alpha,\beta)} \right) \ . \tag{2.22}$$

Dans le cas d'une modélisation des interactions par le modèle de Potts, on a $c = 1$ donc on est assuré d'obtenir un minimum local de la fonction d'énergie au pire égal à deux fois le minimum global. Il est toutefois bon de noter que ce résultat porte uniquement sur l'énergie, et ne garantit donc pas une proximité de l'étiquetage obtenu par rapport à l'étiquetage optimal.

L'alpha-beta-swap

Cet algorithme suppose que la fonction de régularisation est une semi-métrique. L'algorithme d'alpha-beta-swap repose sur le mouvement d'échange. À chaque itération, étant donnée une paire d'étiquettes $(\alpha, \beta) \in \mathcal{L}^2$, seuls deux types de nœuds peuvent changer d'étiquettes. Les nœuds $\mathbf{p}$ d'ancienne étiquette $f_\mathbf{p} = \alpha$ sont autorisés à changer d'étiquette pour la nouvelle étiquette $f'_\mathbf{p} = \beta$ si ce changement entraîne une diminution de l'énergie. Il en est de même pour les nœuds d'ancienne étiquette β qui peuvent prendre la nouvelle étiquette α. Tous les nœuds portant une ancienne étiquette $f_\mathbf{p} \notin \{\alpha, \beta\}$ sont exclus du graphe courant et conservent donc leur étiquette au cours de cette itération. À l'itération suivante, on choisira une autre paire d'étiquettes. Un cycle de l'algorithme d'alpha-beta-swap comprend donc $\frac{|\mathcal{L}|\times(|\mathcal{L}|-1)}{2}$ itérations. La figure 2.5 illustre le fonctionnement de cet algorithme d'alpha-beta-swap.

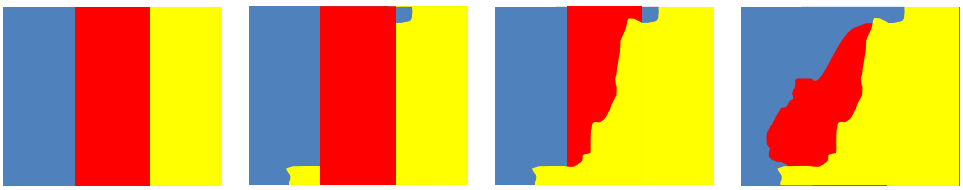


FIGURE 2.5 – Mouvement d'α-β-swap. De gauche à droite : étiquetage initial ; première itération, jaune-bleu-swap ; deuxième itération, jaune-rouge-swap ; troisième itération, rouge-bleu-swap.

On se ramène ainsi à une succession de problèmes binaires. La structure du graphe est déterminée de manière dynamique en fonction de l'étiquetage f courant et des étiquettes α et β. À chaque itération, on construit un graphe à deux terminaux : l'un pour α associé par convention à l'étiquette 0 et l'autre pour β associé à l'étiquette 1.

On créé un nœud pour chaque élément dont l'étiquette courante est soit α, soit β. Et on résout de manière optimale le problème binaire : à quelle étiquette α ou β est-il préférable d'associer un nœud $\mathbf{p}$?

Pour résoudre de manière optimale chaque problème binaire, la fonction de régularisation doit vérifier la contrainte de régularité (2.13) :

$$\begin{aligned} &\widetilde{V}_{\mathbf{p},\mathbf{q}}(0,0) + \widetilde{V}_{\mathbf{p},\mathbf{q}}(1,1) \leq \widetilde{V}_{\mathbf{p},\mathbf{q}}(0,1) + \widetilde{V}_{\mathbf{p},\mathbf{q}}(1,0) \\ \Rightarrow \forall (\alpha,\beta) \in \mathcal{L}^2 \qquad &V_{\mathbf{p},\mathbf{q}}(\alpha,\alpha) + V_{\mathbf{p},\mathbf{q}}(\beta,\beta) \leq V_{\mathbf{p},\mathbf{q}}(\alpha,\beta) + V_{\mathbf{p},\mathbf{q}}(\beta,\alpha) \ , \end{aligned} \tag{2.23}$$

ce qui est vérifié car $V_{\mathbf{p},\mathbf{q}}(.)$ est une semi-métrique.

Algorithme 2 Algorithme d'alpha-beta-swap

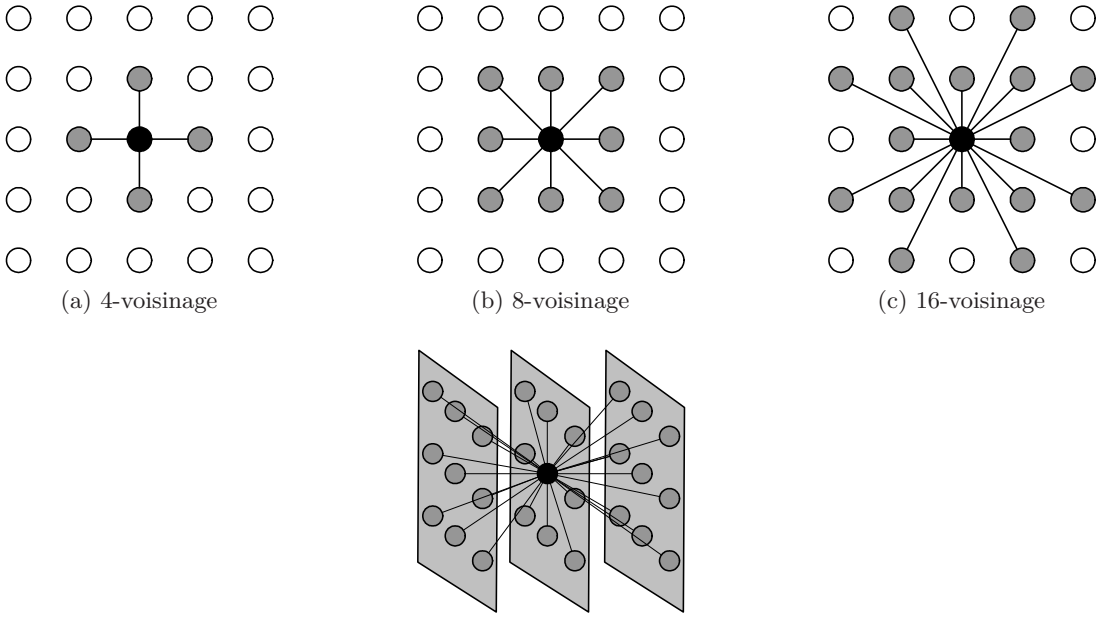
1: Commencer avec un étiquetage f arbitraire
2: baisse $\leftarrow 0$
3: **pour** chaque paire d'étiquettes $(\alpha,\beta) \in \mathcal{L}^2$ **faire**
4: Trouver $\hat{f} = \text{argmin} E(f')$ parmi tous les étiquetages f' pouvant être obtenus par α-β-swap de f
5: **si** $E(\hat{f}) < E(f)$ **alors**
6: $f \leftarrow \hat{f}$ et baisse $\leftarrow 1$
7: **fin si**
8: **fin pour**
9: **si** baisse $= 1$ **alors**
10: aller à 2
11: **fin si**
12: **return** f

Comparé à l'algorithme d'alpha-expansion, l'algorithme d'alpha-beta-swap met plus de temps à converger puisqu'un cycle d'alpha-beta-swap contient plus d'itérations qu'un cycle d'alpha-expansion. Cependant, l'algorithme d'alpha-beta-swap peut être appliqué pour des fonctions d'énergie plus variées puisqu'il requiert que la fonction d'interaction soit seulement une semi-métrique, et non une métrique comme c'est le cas pour l'alpha-expansion. (La contrainte de régularité (2.23) est moins restrictive que la contrainte (2.20).)

2.4 Relations de voisinage

Concernant le type de voisinage retenu pour déterminer $\mathcal{N}$, le 8-voisinage (fig. 2.6b) est généralement préféré au 4-voisinage (fig. 2.6a), qui engendre parfois des résultats dans lesquels l'effet d'escaliers peut être gênant, et aux voisinages supérieurs plus complexes (fig. 2.6c) pour lesquels l'obtention de la solution peut devenir très coûteuse en temps de calcul. Bien évidemment ces considérations correspondent au cas le plus fréquent où l'ensemble des nœuds à étiqueter correspond aux pixels d'une seule et même image. Pour des constructions de graphes plus complexes (par exemple grille 3D régulière (fig. 2.6d) ou graphes dont les nœuds correspondent à des « macro-pixels », des

régions 2D ou des éléments de volume), ce choix du 8-voisinage n'est plus valable et doit donc être revu.



(a) 4-voisinage (b) 8-voisinage (c) 16-voisinage

(d) 26-voisinage

FIGURE 2.6 – Relations de voisinage. Première ligne de gauche à droite : 4, 8 et 16 voisins dans une grille régulière à deux dimensions. Deuxième ligne : 26 voisins dans une grille régulière à trois dimensions.

2.5 Ajout de contraintes fortes pour l'application à la segmentation interactive

Un des avantages du formalisme de coupe dans un graphe est qu'il permet d'introduire facilement des contraintes fortes liées à des indications fournies par un opérateur. En fournissant ces indications, l'opérateur s'attend naturellement à ce que le système les prenne en compte et converge vers une solution les respectant.

Dans l'exemple de la segmentation binaire interactive d'une image, on cherche bien souvent à séparer un objet du fond. L'idée générale est de laisser l'opérateur définir manuellement des régions « graines » qui appartiennent à l'objet et d'autres qui appartiennent au fond. La minimisation de la fonction d'énergie doit ensuite se faire sans que les nœuds du graphe correspondant aux graines ne puissent changer d'étiquette.

Pour éviter à coup sûr des changements d'étiquettes non désirés, on peut facilement construire un graphe qui va garantir que les liens entre les nœuds concernés et les terminaux seront coupés ou non conformément à ce que l'on souhaite.

Supposons que l'on travaille au niveau du pixel. Soit les deux terminaux s et t,

t-link	poids	pour
entre s et $\mathbf{p}$	$D_{\mathbf{p}}(1)$	$\mathbf{p}$ n'appartient à aucune graine
	K	$\mathbf{p}$ appartient à une graine de l'objet
	0	$\mathbf{p}$ appartient à une graine du fond
entre $\mathbf{p}$ et t	$D_{\mathbf{p}}(0)$	$\mathbf{p}$ n'appartient à aucune graine
	0	$\mathbf{p}$ appartient à une graine de l'objet
	K	$\mathbf{p}$ appartient à une graine du fond

TABLE 2.1 – Poids des t-links pour la segmentation interactive avec contraintes fortes [Boykov 01b].

la source s étant associée à l'étiquette 0 et correspondant à l'objet et le puits t étant associé à l'étiquette 1 et correspondant au fond. Si un pixel $\mathbf{p}$ appartient à une graine de l'objet, alors on veut garantir que $f_{\mathbf{p}} = 0$. Pour cela il suffit de s'assurer que le t-link reliant $\mathbf{p}$ et s et de poids $w(s, \mathbf{p})$ n'appartient pas à la coupe (et donc que c'est le t-link opposé reliant $\mathbf{p}$ et t et de poids $w(\mathbf{p}, t)$ qui est coupé puisque la coupe doit séparer $\mathbf{p}$ d'un des deux terminaux). On revient pour cela à la notion de saturation d'un arc (cf. section 2.2). On sait que tout arc appartenant à une coupe minimale d'un graphe est un arc saturé. On va donc fixer $w(s, \mathbf{p}) = K$ et $w(\mathbf{p}, t) = 0$ avec une valeur K suffisamment élevée pour que le lien entre $\mathbf{p}$ et s ne puisse pas être saturé.

Boykov et Jolly [Boykov 01b] définissent K comme suit :

$$K = 1 + \lambda \cdot \max_{\mathbf{p} \in \mathcal{P}} \Big(\sum_{\mathbf{q}:(\mathbf{p},\mathbf{q}) \in \mathcal{N}} \mathcal{V}_{\mathbf{p},\mathbf{q}} \Big) \ , \tag{2.24}$$

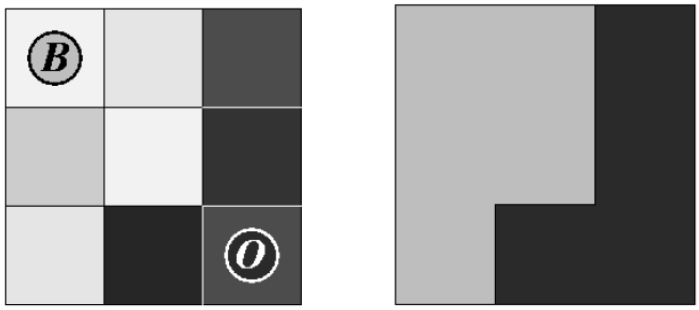
dans le cas où les potentiels binaires sont de la forme $V_{\mathbf{p},\mathbf{q}}(f_{\mathbf{p}}, f_{\mathbf{q}}) = \mathcal{V}_{\mathbf{p},\mathbf{q}} \cdot T(f_{\mathbf{p}} \neq f_{\mathbf{q}})$, généralisant le modèle d'Ising/Potts.

La constante K est donc plus élevée que la somme des coûts de tous les n-links liés à n'importe quel pixel $\mathbf{p}$. Si un pixel $\mathbf{p}$ appartient à une graine de l'objet, il est plus coûteux de couper le lien de poids K entre $\mathbf{p}$ et s que de couper tous les n-links liés à $\mathbf{p}$ (coût cumulé inférieur à K) plus le lien de poids nul entre $\mathbf{p}$ et t. En d'autres termes, une partition où $\mathbf{p}$ porte seul l'étiquette 0 parmi tous ses voisins d'étiquette 1 est moins coûteuse qu'une partition où $\mathbf{p}$ et tous ses voisins porteraient la même étiquette 1.

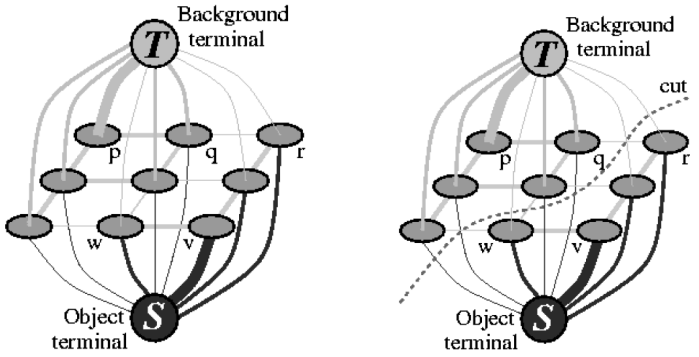
Le tableau 2.1 résume la pondération des t-links selon les différents cas. La figure 2.7 donne un exemple simple de segmentation interactive avec contraintes fortes pour une image 3×3.

Les régions graines introduites par l'opérateur permettent également de faire l'apprentissage des modèles utilisés ensuite dans le terme d'attache aux données de la fonction d'énergie. Ainsi, Boykov et Jolly [Boykov 01b] utilisent les intensités des pixels des régions graines pour estimer deux histogrammes, l'un pour l'objet et l'autre pour le fond. Pour un pixel $\mathbf{p}$ n'appartenant à aucune graine, le terme d'attache aux données $D_{\mathbf{p}}(f_{\mathbf{p}})$ est alors un terme de pénalité défini comme la log-vraisemblance négative de la distribution d'intensité de la couche $f_{\mathbf{p}}$.

Lorsque de nouvelles graines sont introduites par l'opérateur, on peut envisager deux possibilités : soit les histogrammes sont mis à jour, auquel cas les poids de tous



(a) Image 3×3 et les graines (b) Partition obtenue



(c) Graphe associé (d) Coupe minimale

FIGURE 2.7 – Segmentation interactive d'une image 3×3 avec contraintes fortes. Les graines correspondent au pixel **p** pour le fond (marqué $\mathcal{B}$) et au pixel **v** pour l'objet (marqué $\mathcal{O}$). Le poids des arcs est proportionnel à leur épaisseur. Cette figure provient de [Boykov 01b].

les t-links doivent être mis à jour également ; soit on conserve les histogrammes initiaux, auquel cas seuls les poids des t-links liés à des pixels appartenant aux nouvelles graines sont à mettre à jour. La mise à jour de la partition est donc encore plus rapide dans le second cas que dans le premier. Dans les deux cas, on conserve le même squelette du graphe initial (nœuds et arcs) et les mêmes poids pour les n-links, ce qui est un avantage certain pour des applications semi-automatiques puisque le système se montre très réactif à l'introduction de nouvelles indications.

Dans l'approche itérative « GrabCut », Rother *et al.* [Rother 04] proposent de remplacer les histogrammes de luminance par deux mélanges de gaussiennes (GMMs) de couleurs. Et ils réduisent l'intervention de l'opérateur à la simple définition d'un rectangle englobant l'objet. Ainsi dès l'initialisation, on sait assurément que tous les pixels extérieurs au rectangle appartiennent au fond. L'étiquetage des pixels intérieurs au rectangle est encore incertain. Pour la première itération, on associe temporairement tous les pixels intérieurs au rectangle à l'objet et on estime les deux mélanges de gaussiennes à partir de cet étiquetage temporaire. Puis on utilise les modèles estimés dans

FIGURE 2.8 – Deux exemples d'extraction d'objet d'avant-plan par la méthode « GrabCut ». Sans le moindre effort, l'opérateur fournit un rectangle qui a pour seule contrainte de contenir l'objet. La segmentation est ensuite automatique. Ces figures proviennent de [Rother 04].

le terme d'attache aux données de la fonction d'énergie qu'on minimise (étape M). Seuls les pixels intérieurs au rectangle peuvent changer d'étiquette. De manière itérative, la nouvelle partition est utilisée pour mettre à jour les paramètres des deux modèles de gaussiennes (étape E) avant une nouvelle minimisation de la fonction d'énergie. Les itérations se succèdent jusqu'à ce que la segmentation converge. La figure 2.8 illustre la méthode « GrabCut » par deux exemples.

On vient de citer deux exemples de méthodes de segmentation interactive. Dans ces deux exemples, il s'agit de segmentation binaire d'une image. Bien entendu, le même type de contraintes fortes peut être utilisé sans condition sur le nombre d'étiquettes ou sur les dimensions du graphe.

2.6 Alternatives importantes aux algorithmes de coupe minimale/flot maximal

Nous citons ici des alternatives importantes aux algorithmes de coupe minimale/flot maximal pour minimiser l'énergie d'un champ de Markov.

2.6.1 Le recuit simulé (« simulated annealing »)

En métallurgie, un processus physique alterne des cycles de refroidissement lent et de réchauffage (recuit) qui tendent à minimiser l'énergie du matériau. Par analogie avec ce processus physique, le recuit simulé [Kirkpatrick 83, Geman 84] fait appel à un paramètre de température qui dirige la minimisation. Par rapport à la méthode du gradient, le principe général est d'introduire des perturbations aléatoires permettant de sortir des bassins d'attractions des minima locaux. Ces perturbations s'atténuent au cours du temps, de sorte à espérer finalement aboutir à un minimum global. On recherche donc les configurations les plus probables qui correspondent à des états d'énergie minimale. À chaque itération de l'algorithme, on part d'une configuration donnée à laquelle on applique une légère modification qui entraîne une variation de l'énergie. La nouvelle configuration peut soit améliorer le critère que l'on cherche à optimiser, soit le dégrader. En acceptant une configuration améliorant le critère, on tend ainsi

à chercher le minimum dans le voisinage de la configuration de départ. L'acceptation d'une « mauvaise » configuration permet quant à elle d'explorer une plus grande partie de l'espace de solutions et évite de s'enfermer trop rapidement dans la recherche d'un minimum local.

S'il présente l'avantage d'accepter des remontées en énergie, cet algorithme peut se montrer extrêmement coûteux en temps de calcul. La température doit en effet baisser très lentement pour générer un grand nombre de configurations au fur et à mesure des itérations, et donc pour espérer des résultats satisfaisants.

2.6.2 Les modes conditionnels itérés ou « Iterated Conditional Modes » (ICM)

L'algorithme des modes conditionnels itérés (ICM) [Besag 86] a été proposé pour accélérer la convergence mais il est sous-optimal : il n'existe pas de preuve de convergence vers un minimum global. Comme pour le recuit simulé, le principe est itératif, mais cette fois-ci les modifications de configurations se font de manière déterministe. Pour une configuration donnée, l'étiquette d'un site est mise à jour à partir des étiquettes probables des sites voisins en choisissant la valeur qui maximise la probabilité conditionnelle locale. On recherche donc ici une approximation du maximum *a posteriori*, résultat de la convergence des itérations. Les résultats dépendent alors fortement de l'initialisation et l'on constate souvent une convergence vers un minimum local.

2.6.3 La propagation bouclée de croyances ou « Loopy Belief Propagation » (LBP)

L'algorithme de propagation de croyances ou « Belief Propagation » (BP) calcule la solution optimale d'un graphe sans cycle, c'est-à-dire d'un arbre, en passant des messages entre les nœuds *pères* et les nœuds *fils*. L'algorithme de propagation bouclée de croyances (LBP) [Felzenszwalb 06] calcule une approximation de la solution optimale. Il repose sur la propagation de croyances mais peut s'appliquer aux graphes généraux. Dans le cas où le graphe contient des cycles, les messages sont passés soit en parallèle, soit dans un ordre aléatoire sur les nœuds. La LBP peut être appliquée à une catégorie d'énergies plus large que les énergies pouvant être minimisées par les algorithmes de coupe minimale/flot maximal. Cependant, dans certains cas (par exemple pour un problème simple de stéréovision), l'énergie finale obtenue est bien plus forte que celle obtenue par les algorithmes de coupe minimal/flot maximal [Kolmogorov 06a].

2.6.4 L'algorithme « Tree-ReWeighted Message Passing » (TRW)

L'algorithme « Tree-ReWeighted Message Passing » (TRW) a été initialement proposé par Wainwright *et al.* [Wainwright 05], puis ensuite amélioré par Kolmogorov [Kolmogorov 06b]. Cet algorithme se base également sur la BP. En pratique, pour des graphes peu connectés (en 4-connexité par exemple), les résultats du TRW sont aussi bons voire meilleurs que ceux obtenus par les algorithmes de coupe minimale/flot maxi-

mal. En revanche, pour des graphes hautement connectés, les résultats du TRW sont bien moins bons que ceux obtenus par coupes de graphe [Kolmogorov 06a].

2.6.5 L'algorithme « Fast Primal-Dual » (Fast-PD)

L'algorithme « Fast Primal-Dual » (Fast-PD) généralise l'algorithme d'alpha-expansion présenté à la sous-section 2.3.2. Il a été proposé récemment par Komodakis *et al.* [Komodakis 08]. Il peut être de 3 à 9 fois plus rapide que l'algorithme d'alpha-expansion. Son efficacité est liée au fait qu'il exploite des informations provenant non seulement du champ de Markov primal, mais aussi du problème dual. Par ailleurs, le Fast-PD n'implique pas que la fonction de régularisation soit une métrique. Il garantit donc une solution presque optimale pour une classe de problèmes beaucoup plus large que celle abordable par l'alpha-expansion.

2.7 Conclusion

Dans ce chapitre, nous avons présenté l'intérêt des algorithmes de coupe minimale/flot maximal dans un graphe pour la résolution de certains problèmes rencontrés en vision par ordinateur. Dans notre cas, nous sommes face à un problème de segmentation multi-étiquettes. Dans le chapitre suivant, nous présentons des méthodes de segmentation de séquences d'images au sens du mouvement qui s'appuient sur les algorithmes de coupe minimale/flot maximal. Comme [Rother 04], nous utilisons des mélanges de gaussiennes de couleurs mais notre segmentation ne peut pas reposer sur ce seul critère, le mouvement devant être le principal critère de distinction entre deux classes. Nos méthodes reposent également sur les notions de graines et de contraintes fortes présentées à la section 2.5. On ne peut pas imaginer, cependant, que l'opérateur introduise manuellement les graines sur chaque image de la séquence. Nous utilisons donc l'information de mouvement pour placer automatiquement ces graines.

Chapitre 3

Estimation du mouvement

L'estimation de mouvement est un problème essentiel pour l'analyse d'une séquence d'images. L'information de mouvement est en effet fondamentale pour caractériser le contenu dynamique de la scène. Elle occupe en particulier une place centrale dans la segmentation d'une séquence d'images au sens du mouvement.

La première partie de ce chapitre présente les techniques d'estimation locale du mouvement. La seconde partie aborde l'emploi de modèles paramétriques pour décrire les mouvements présents dans une scène.

3.1 Estimation de champs denses de mouvement

L'estimation de champs denses de mouvement (également appelée estimation du flot optique) vise à déterminer un vecteur de mouvement en chaque point de l'image. Le problème n'est pas récent puisque les travaux fondateurs datent de 1981 [Horn 81, Lucas 81]. Mais c'est un problème difficile et mal posé (car sous-contraint), ce qui explique que de nombreuses méthodes ont été développées depuis les premiers travaux, et continuent de l'être, pour améliorer la fiabilité des estimations et réduire les temps de calcul.

Cette section présente brièvement les hypothèses et contraintes les plus répandues dans la littérature, ainsi que quelques contributions récentes présentées sur le site d'évaluation de l'équipe Vision de Middlebury College [Baker 07].

3.1.1 Contraintes locales de conservation de l'intensité et de conservation du gradient spatial

Les méthodes d'estimation du flot optique s'appuient sur l'hypothèse de conservation de l'intensité d'un point de l'image au cours du temps. L'intensité d'un point est donc supposée rester constante le long de sa trajectoire, ce qui correspond à la grande majorité des situations rencontrées.

Soit $I_t(\mathbf{p})$ l'intensité lumineuse d'un pixel $\mathbf{p} = (x, y)^T$ à l'instant t et $\mathbf{dp} = (dx, dy)^T$ le vecteur de mouvement associé à $\mathbf{p}$. L'hypothèse de conservation de l'intensité appliquée au pixel $\mathbf{p}$ entre les instants t et $t + 1$ s'écrit :

$$I_{t+1}(\mathbf{p}+\mathbf{dp}) - I_t(\mathbf{p}) = 0 \ . \tag{3.1}$$

On remarque aisément que cette hypothèse n'est pas vérifiée dans les cas suivants :
- scène composée de plusieurs objets se déplaçant indépendamment les uns des autres entraînant des zones d'occultation,
- mouvements en dehors du plan image,
- phénomènes de transparence,
- variations des conditions d'illumination.

Par ailleurs, on voit que l'on est face à un problème sous-contraint puisque l'on cherche à estimer les deux composantes dx et dy du vecteur de mouvement à partir d'une seule équation.

En supposant que le déplacement est de faible amplitude, la linéarisation de l'hypothèse de conservation de l'intensité conduit à l'équation de contrainte du mouvement apparent (ECMA) :

$$\nabla I(\mathbf{p}) \cdot \mathbf{dp} + \frac{\partial I}{\partial t}(\mathbf{p}) = 0 \ , \tag{3.2}$$

où $\nabla = (\partial_x, \partial_y)^T$ désigne le gradient spatial.

Pour rendre la méthode plus robuste aux variations des conditions d'illumination, on peut s'appuyer également sur une hypothèse de conservation du gradient spatial de l'image au cours du temps.

$$\nabla I_{t+1}(\mathbf{p}+\mathbf{dp}) - \nabla I_t(\mathbf{p}) = \mathbf{0} \ . \tag{3.3}$$

Les deux hypothèses, qui correspondent à des contraintes locales, peuvent être combinées. En formulant le problème comme un problème de minimisation d'une fonction d'énergie par rapport aux vecteurs de mouvement, on peut directement déduire de ces contraintes un terme d'attache aux données qui mesure les écarts par rapport à ces hypothèses. Dans ce terme, il est bon d'atténuer l'effet des vecteurs aberrants, c'est-à-dire de limiter l'influence des forts résidus. Si les deux hypothèses sont combinées, on peut utiliser une même fonction robuste pour la combinaison des deux hypothèses [Brox 04], ou utiliser deux fonctions robustes pour pénaliser séparément chacune des deux hypothèses [Bruhn 05a].

Néanmoins, employées seules, les contraintes locales se montrent inefficaces pour résoudre le problème. D'une part, l'information de mouvement n'est que partiellement observable, ce qu'on appelle communément le « problème de l'ouverture ». Seule la composante de déplacement normale aux contours peut être directement déterminée. Dans les régions uniformes, le mouvement n'est pas observable localement dans l'image. D'autre part, dans les régions d'occultations, certains points se sont déplacés à des positions où ils sont devenus invisibles.

En conséquence, l'estimation du flot optique requiert la formulation d'hypothèses supplémentaires pour caractériser le mouvement de manière plus précise. On peut distinguer trois catégories de méthodes d'estimation du flot optique : les méthodes fréquentielles, les méthodes de mise en correspondance et les méthodes différentielles.

3.1.2 Méthodes fréquentielles

Les méthodes fréquentielles [Heeger 88, Fleet 90, Clifford 95, Weber 95] sont dites aussi méthodes de filtrage spatio-temporel car elles se basent sur des filtres spatio-temporels orientés (souvent des filtres de Gabor 3D) pour extraire l'information de mouvement. Ces méthodes reposent sur la transformée de Fourier en mouvement des images. Elles supposent que le flot optique est constant sur le support spatio-temporel des filtres, ce qui a pour conséquence un résultat lissé à la fois en espace et en temps, éventuellement problématique dans le cas de séquences d'images où les mouvements sont saccadés. Par ailleurs, ces méthodes ont généralement un coût de calcul élevé lié aux opérations de filtrage spatio-temporel.

3.1.3 Méthodes de mise en correspondance de blocs

Les méthodes de mise en correspondance de blocs ou de « block matching » recherchent, pour une fenêtre spatiale ω (ou bloc) donnée dans une première image I_1, la fenêtre de l'image I_2 qui présente la meilleure correspondance. La mesure de cette correspondance se fait généralement par calcul de l'erreur absolue moyenne, de la corrélation ou de l'erreur quadratique moyenne. L'erreur absolue moyenne est le critère le plus répandu car il fournit de bons résultats tout en demandant peu d'opérations.

Pour un bloc ω de taille $M \times N$, on cherche donc le vecteur de mouvement $(u, v)^T$ qui minimise l'erreur absolue moyenne :

$$E(u,v) = \frac{1}{M \times N} \sum_{(x,y)\in\omega} |I_1(x,y) - I_2(x+u, y+v)| \ . \tag{3.4}$$

Ces méthodes permettent de considérer des déplacements de grande amplitude. En général on ne cherche pas un bloc similaire dans n'importe quelle partie de l'image mais on se restreint à une région d'intérêt autour de la position du bloc courant considéré. Le rayon de cette région dépend de la dynamique de déplacement autorisée. L'algorithme de recherche le plus simple et le meilleur en terme de correspondance est celui qui procède à une recherche exhaustive du meilleur correspondant dans la région d'intérêt. Il peut cependant mener, selon le rayon de cette région, à un coût de calcul très élevé.

De nombreux algorithmes de recherche, plus rapides mais sous-optimaux en terme de correspondance, ont été proposés. L'idée générale est de ne parcourir qu'un petit nombre de positions de la région d'intérêt avant de retenir la position (qui est le centre du bloc candidat) offrant la meilleure correspondance avec le bloc considéré. Certains de ces algorithmes utilisent des motifs de recherche fixes (croix [Jain 81, Ghanbari 90], diamant [Zhu 00]) de rayon initial égal à la moitié du mouvement maximal autorisé. Les différents points constituant un motif indiquent les centres des blocs candidats à tester. À l'itération suivante, le motif est déplacé sur le centre du meilleur bloc candidat et sa taille est divisée par deux pour affiner le déplacement estimé. La figure 3.1 illustre le principe de l'algorithme de recherche en croix de Ghanbari.

L'utilisation d'une fonction d'interpolation (bilinéaire par exemple) permet de considérer des déplacements de précision inférieure au pixel.

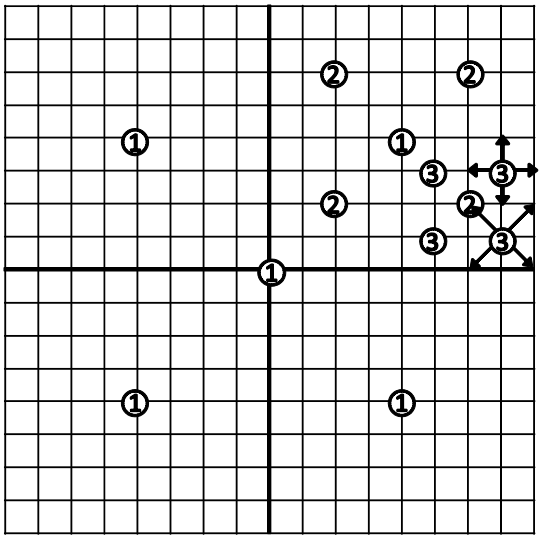


FIGURE 3.1 – Exemple de recherche en croix [Ghanbari 90]. Les centres des blocs candidats testés aux itérations 1, 2 et 3 sont indiqués par les cercles. Les flèches illustrent les différents motifs de recherche utilisés à l'itération supplémentaire 4 selon la position retenue à l'itération 3 : une croix grecque ('+') autour des coins en haut à droite et en bas à gauche ou une croix en sautoir ('×') autour des coins en bas à droite et en haut à gauche.

Les approches hiérarchiques permettent également de réduire le temps de calcul des algorithmes de recherche. Par exemple, pour une approche pyramidale, des pyramides des images originales sont construites par filtrages et sous-échantillonnages successifs. Alors en descendant les pyramides, un vecteur de mouvement estimé grossièrement à un niveau supérieur $k+1$ est transmis au niveau inférieur k, et en tant que prédicteur il guide l'étape de raffinement. D'autre part, pour éviter qu'une mauvaise estimation à un niveau $k+1$ ne soit transmise au niveau k et donc réduire les chances de tomber dans un minimum local, plusieurs vecteurs peuvent être proposés comme prédicteurs au niveau k. On procède alors généralement soit à une recherche plus précise autour de chacun des prédicteurs [Nam 95], soit à une sélection *a posteriori* du meilleur prédicteur par rapport aux images du niveau k des pyramides, soit à la détermination d'un unique prédicteur par combinaison linéaire de ces vecteurs. En plus du prédicteur naturel, les prédicteurs supplémentaires peuvent par exemple être les deuxième et troisième meilleurs vecteurs candidats pour le bloc considéré au niveau $k+1$. Mais on peut également choisir comme prédicteurs supplémentaires, les vecteurs estimés au niveau $k+1$ pour les blocs voisins du bloc considéré, ou encore les vecteurs estimés au niveau k pour les blocs voisins du bloc considéré et déjà traités [Lim 97]. Cette dernière remarque s'appuie sur l'hypothèse d'un champ de mouvement spatialement cohérent.

3.1.4 Méthodes différentielles

Les méthodes différentielles reposent directement sur l'ECMA (équation (3.2)). Du fait de leur nature différentielle (contrairement aux méthodes de mise en correspondance), elles permettent naturellement une estimation subpixellique du mouvement. Pour surmonter le problème de l'ouverture, en plus de l'hypothèse de conservation de l'intensité, on suppose que les vecteurs de mouvement sont cohérents spatialement (et éventuellement temporellement). Cela mène à définir une contrainte de régularité du flot optique.

Le modèle local [Lucas 81] suppose que pour les pixels d'un même voisinage, le mouvement est constant. Ce modèle peut être également appliqué au suivi de points caractéristiques d'une image [Tomasi 91]. Pour l'estimation de champ dense, une solution plus régulière est le modèle global [Horn 81] qui suppose une régularité globale du mouvement dans la séquence d'images étudiée. On ajoute donc dans la fonction d'énergie un terme de lissage ou de régularisation. Notons que modèle local et modèle global peuvent être combinés de manière efficace [Bruhn 05b].

Soit une fenêtre spatiale ω sur laquelle le flot optique est supposé constant, alors, selon Lucas et Kanade, l'estimation de mouvement en tout point de la fenêtre considérée consiste à déterminer le vecteur $(u, v)^T$ qui minimise l'erreur quadratique définie sur le support ω :

$$E_{LK}(u,v) = \sum_{(x,y)\in\omega} \left(\frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \frac{\partial I}{\partial t}\right)^2 . \tag{3.5}$$

Cependant l'approche de Lucas et Kanade a pour défaut d'être trop locale. L'introduction d'une contrainte sur la régularité du flot optique et l'expression du problème d'estimation comme un problème de minimisation d'une fonctionnelle de champs de vecteurs sont des avancées majeures proposées par Horn et Schunck.

Soit la séquence d'images $I(x, y, t)$ définie sur un domaine image Ω et un intervalle temporel $[0, T]$, l'estimation du flot optique selon Horn et Schunck consiste à déterminer les fonctions inconnues $u(x, y, t)$ et $v(x, y, t)$ qui minimisent la fonction d'énergie globale :

$$E_{HS}(u,v) = \sum_{(x,y)\in\Omega} \left(\left(\frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \frac{\partial I}{\partial t}\right)^2 + \alpha\left(|\nabla(u)|^2 + |\nabla(v)|^2\right)\right) . \tag{3.6}$$

Ici, (u, v) n'est plus une constante mais un champ de vecteurs.

Cependant le modèle global correspond à une régularisation homogène quadratique qui provoque un lissage excessif du champ et qui ne préserve donc pas les discontinuités de mouvement aux frontières des objets présents dans la scène. Cette régularisation quadratique est relativement simple à analyser et minimiser mais est sensible aux données aberrantes. On trouve dans la littérature des descriptions de régularisations non-quadratiques mieux adaptées qui visent à respecter les discontinuités de mouvement.

La régularisation « flow-driven » isotrope [Cohen 93, Brox 04, Bruhn 05a] prend en compte les propriétés du flot estimé pour relâcher la contrainte de lissage sur les régions correspondant à des forts gradients du flot.

La régularisation « flow-driven » anisotrope [Weickert 01] encourage le lissage du flot le long des contours du flot estimé mais relâche la contrainte à travers ces contours.

Les régularisations « image-driven » [Alvarez 99, Nagel 86] prennent en compte les gradients des images de la séquence. Elles sont moins performantes puisque les frontières de mouvement et les contours de l'image ne se correspondent pas nécessairement.

Ces régularisations non-quadratiques font en général appel à des fonctions positives croissantes mais à croissance moins rapide que la fonction quadratique. Leur minimisation requiert beaucoup plus de temps de calcul. Cependant les approches multi-résolution permettent non seulement de diminuer fortement le coût calculatoire en travaillant sur une décomposition pyramidale des images, mais également et surtout de vérifier les conditions de validité de l'ECMA (équation (3.2)).

Plus récemment, des chercheurs ont commencé à transférer les connaissances issues des travaux sur l'estimation de cartes de disparité aux méthodes différentielles d'estimation du flot optique. Ainsi, une sur-segmentation des images originales en régions de couleurs similaires permet, en stéréovision, d'obtenir des estimées de disparité de très bonne qualité même dans les zones peu texturées en ajustant un plan de disparité sur chacune de ces régions de couleurs [Klaus 06, Wang 08]. De la même manière, à partir d'un flot optique initial, en estimant un modèle affine de mouvement pour chacune de ces mêmes régions de couleurs, puis en relâchant cette contrainte de paramétrisation dans les régions correspondant à des objets non-rigides, la précision des estimations est améliorée dans les zones peu texturées [Xu 08].

Par ailleurs, la préservation de discontinuités fortes au niveau des frontières de mouvement nécessite l'utilisation de fonctions d'énergie non-convexes basées sur des fonctions statistiques robustes. Ces fonctions d'énergie sont difficiles à minimiser, mais de nouvelles approches ont néanmoins été proposées pour rendre possible l'estimation précise du flot optique en temps réel [Zach 07].

3.1.5 Approches multi-résolution

Pour les approches différentielles, la linéarisation utilisée pour obtenir l'ECMA (3.2) et nécessaire à la minimisation de la fonction d'énergie n'est pas valable pour des déplacements de fortes amplitudes. Pour des mouvements d'amplitude supérieure à 1 pixel entre 2 images consécutives, un algorithme de minimisation peut facilement rester dans un minimum local. En vue de trouver le minimum global, on fait généralement appel à des approches multi-résolution. Elles consistent à construire des pyramides d'images par filtrages et sous-échantillonnages successifs des images originales. Ces pyramides permettent de considérer différentes échelles de déplacement en fonction du niveau de résolution. Concrètement, ces approches procèdent à une première approximation du mouvement à une échelle réduite de l'image, puis, si besoin, affinent de manière incrémentale cette approximation à une échelle plus fine, ainsi de suite jusqu'au niveau le

plus fin qui correspond à l'échelle originale 1 :1 de l'image.

3.1.6 Filtre bilatéral

Le filtrage bilatéral est une solution relativement simple pour préserver les discontinuités tout en lissant les zones homogènes. Le filtre bilatéral [Aurich 95, Tomasi 98] a initialement été introduit pour lisser une image en niveaux de gris ou en couleurs. L'idée est alors de combiner des informations d'intensité en se basant à la fois sur leur proximité spatiale et sur leur similarité photométrique. En donnant plus d'influence aux valeurs proches qu'aux valeurs distantes, aussi bien dans le domaine spatial que dans la dynamique des intensités, il permet un filtrage local sélectif qui propage l'information d'intensité sans traverser les discontinuités.

Son adaptation au lissage du flot optique [Xiao 06] combine des informations de mouvement en se basant sur leur proximité spatiale, sur leur similarité en terme de mouvement mais également sur la similarité photométrique des intensités des pixels considérés de l'image associée au flot estimé. En tenant également compte des régions d'occultation, il permet de propager l'information de mouvement en ne traversant ni les discontinuités du flot, ni les discontinuités d'intensité de l'image, et ce seulement à partir des régions non occultées dont les estimées de mouvement sont jugées fiables.

Plus précisément, le filtre bilatéral proposé par Xiao *et al.* se compose de trois noyaux de convolution gaussiens et d'une fonction d'occultation. Les trois noyaux gaussiens sont associés respectivement au domaine spatial pour réduire l'influence des pixels spatialement éloignés du pixel central, à la dynamique des intensités pour réduire l'influence des pixels dont la luminance est éloignée de celle du pixel central, et à la dynamique de mouvement pour réduire l'influence des pixels dont le vecteur de mouvement est éloigné de celui du pixel central. La fonction d'occultation basée sur la DFD permet de détecter les pixels occultés et de leur associer une influence nulle sur le filtrage. La taille du noyau spatial est adaptée selon la distance de son centre à une région non-occultée pour garantir l'existence de pixels d'influence non nulle lors du filtrage.

Ainsi pour Xiao *et al.*, le filtre bilatéral adaptatif est défini par :

$$\mathbf{dp}^{+} = \frac{1}{k(\mathbf{p})} \sum_{\mathbf{q}\in\omega(\mathbf{p})} g_s(\mathbf{q}-\mathbf{p}) \cdot g_I(I(\mathbf{q})-I(\mathbf{p})) \cdot g_m(\mathbf{dq}-\mathbf{dp}) \cdot \rho(\mathbf{q}) \cdot \mathbf{dq} \ , \qquad (3.7)$$

où $\mathbf{dp}^{+}$ est la sortie du filtre pour le pixel $\mathbf{p}$, $\omega(\mathbf{p})$ est la fenêtre spatiale centrée sur $\mathbf{p}$ qui sert au filtrage, $g_s(\cdot)$, $g_I(\cdot)$ et $g_m(\cdot)$ sont trois fonctions gaussiennes définies respectivement pour les domaines spatial, d'intensité et de mouvement, $\rho(\cdot)$ est la fonction d'occultation et $k(\mathbf{p})$ est le terme de normalisation :

$$k(\mathbf{p}) = \sum_{\mathbf{q}\in\omega(\mathbf{p})} g_s(\mathbf{q}-\mathbf{p}) \cdot g_I(I(\mathbf{q})-I(\mathbf{p})) \cdot g_m(\mathbf{dq}-\mathbf{dp}) \cdot \rho(\mathbf{q}) \ . \qquad (3.8)$$

La figure 3.2 illustre le principe du filtre bilatéral.

Xiao *et al.* ont inclus ce filtre dans l'estimation du flot optique. Si post-traiter est *a priori* moins séduisant qu'inclure directement la régularisation dans l'estimation, nous

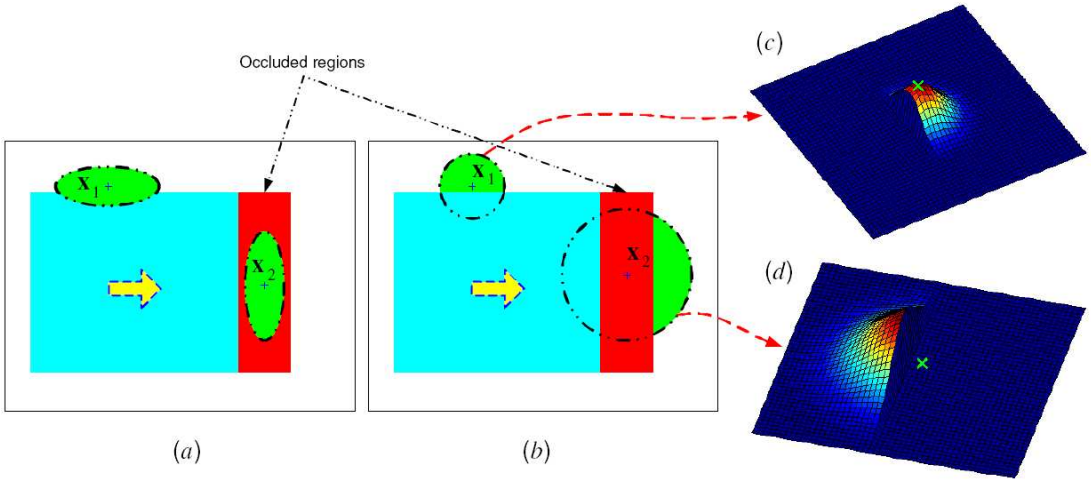


FIGURE 3.2 – Principe du filtre bilatéral. Le fond blanc est fixe. Le rectangle cyan se déplace de la gauche vers la droite et recouvre la région en rouge. (Seule la première image est montrée.) Les pixels $\mathbf{x_1}$ et $\mathbf{x_2}$ sont situés respectivement près d'une frontière de mouvement et dans une région d'occultation. (a) Dans le cas de la régularisation anisotrope, le noyau gaussien est orienté le long de la discontinuité. (b) En utilisant le filtre bilatéral, la forme et la taille du noyau gaussien sont adaptées afin de favoriser les informations similaires en termes d'intensité et de mouvement. Seules les régions vertes sont utilisées pour obtenir la valeur filtrée du flot au centre du noyau marqué par la croix. (c) et (d) Représentations 3D du noyau du filtre bilatéral. La croix verte indique le centre du noyau. Cette figure provient de [Xiao 06].

avons néanmoins implémenté ce filtre pour pouvoir l'appliquer en post-traitement à tout champ de mouvement, indépendamment de la méthode utilisée pour l'estimation.

Si les fonctions de lissage propres à nos estimateurs présentés dans la sous-section suivante ne nous donnent pas satisfaction, nous proposons d'utiliser en post-traitement un filtre bilatéral similaire à celui employé dans [Xiao 06] mais dont nous modifions la fonction d'occultation en la basant sur un critère de cohérence entre champ « avant » et champ « arrière » estimés sur une même paire d'images.

Soit un pixel $\mathbf{p}$ de l'image I_{t_1} dont on a estimé un vecteur de mouvement $\mathbf{dp}$ vers l'image I_{t_2}. Partant de ce pixel $\mathbf{p}$ dans I_{t_1}, l'idée est de déterminer $\mathbf{p}'$ le pixel le plus proche de $\mathbf{p}+\mathbf{dp}$ dans I_{t_2} et de revenir à une position $\mathbf{p}''$ (éventuellement non entière) dans I_{t_1} en utilisant le vecteur de mouvement « arrière » $\mathbf{dp}'$ estimé au pixel $\mathbf{p}'$. On s'intéresse alors à la distance euclidienne entre $\mathbf{p}''$ et $\mathbf{p}$.

$$\|\mathbf{p}'' - \mathbf{p}\| \approx \|\mathbf{dp} + \mathbf{dp}'\| \quad . \tag{3.9}$$

Idéalement, pour un point visible dans les deux images I_{t_1} et I_{t_2}, cette distance est nulle. Mais si le point associé à $\mathbf{p}$ est occulté dans l'image I_{t_2}, dans l'absolu le vecteur de mouvement $\mathbf{dp}$ ne devrait pas exister. Il est donc probable que son estimée soit

incohérente avec celle du vecteur $\mathbf{dp}'$, ce qui conduit à une distance plus forte. Notre fonction d'occultation $\rho(\cdot)$ est alors définie de la manière suivante :

$$\rho(\mathbf{p}) = \begin{cases} 1 & \text{si } \|\mathbf{dp} + \mathbf{dp}'\|^2 < \tau \\ 0 & \text{sinon} \end{cases} . \tag{3.10}$$

La valeur du seuil τ est fixée de manière empirique. Pour nos manipulations nous fixons $\tau = 1.5$. La fonction $\|\cdot\|$ est la distance euclidienne dans l'espace de déplacement.

Cette fonction d'occultation permet, lors du filtrage, d'annuler l'influence des vecteurs estimés dans les zones d'occultation. Par ailleurs, pour calculer un nouveau vecteur à un pixel $\mathbf{p}$ situé dans une zone d'occultation ($\rho(\mathbf{p}) = 0$), nous n'encourageons plus les vecteurs similaires à l'ancien vecteur $\mathbf{dp}$. Pour cela, nous gelons l'effet de la fonction gaussienne liée au mouvement en imposant $g_m(\mathbf{dq} - \mathbf{dp}) = 1$ si $\rho(\mathbf{p}) = 0$.

Cependant, l'opération de filtrage bilatéral s'avère très coûteuse en temps de calcul. En effet, le filtre étant adaptatif, ses coefficients doivent être recalculés à chaque pixel. Paris et Durand [Paris 06] ont proposé une approximation rapide du filtre bilatéral à deux noyaux utilisant un sous-échantillonnage en espace et en intensité. Leur méthode peut être facilement étendue à un filtre comprenant plus de noyaux, mais l'ajout de la fonction d'occultation n'est pas aussi direct.

Pour ces raisons, le filtrage bilatéral n'a pas été utilisé pour l'obtention des résultats expérimentaux présentés au chapitre 5. Il a plutôt été développé à titre exploratoire. Son emploi serait pleinement justifié pour des applications nécessitant l'estimation de champs denses extrêmement précis. Le calcul des coefficients de filtrage étant une tâche hautement parallélisable, nous tirerions un grand profit d'une implémentation sur processeur graphique.

3.1.7 Estimateurs utilisés

Nous utilisons d'une part une approche de mise en correspondance de blocs, et d'autre part une approche différentielle. Ces deux estimateurs issus des travaux de Lemonnier [Lemonnier 93] ont avant tout été choisis pour des raisons de disponibilité, et se sont montrés efficaces sur les séquences traitées lors des expérimentations.

Notre estimateur par mise en correspondance de blocs est un estimateur hiérarchique à 4 niveaux, avec des blocs de taille 4×4. Nous utilisons une fonction d'interpolation bilinéaire nous permettant une précision des vecteurs estimés pouvant aller jusqu'au quart de pixel. Le critère de comparaison retenu est l'erreur absolue moyenne. Au niveau le plus fin, on procède aux lissages indépendants des composantes horizontale et verticale du champ de vecteurs, en fonction de mesures de confiance associées à chaque vecteur dans les directions horizontale et verticale, et basées sur la DFD.

Notre estimateur différentiel est un estimateur multi-résolution qui comprend 5 niveaux. À chaque niveau, on procède à une régularisation spatiale de chacune des composantes du champ de vecteurs. Cette régularisation se fait sur la base d'une mesure de confiance associée à chaque composante de chaque vecteur. Pour chaque pixel du niveau considéré, la confiance accordée à la composante horizontale (resp. verticale) du

vecteur de mouvement estimé dépend du gradient spatial horizontal (resp. vertical) et de l'erreur quadratique moyenne sur une fenêtre de taille 3×3 autour du pixel considéré.

Les mesures de confiance des deux estimateurs sont inspirées des confiances proposées dans [Heel 91].

Pour notre premier schéma de segmentation de séquence au sens du mouvement, l'approche faisant appel à des mouvements estimés entre images consécutives, nous utilisons l'estimateur différentiel qui se montre plus précis et plus rapide pour traiter des mouvements de faibles amplitudes.

Pour notre second schéma, l'approche faisant appel cette fois à des mouvements estimés entre images distantes, nous utilisons l'estimateur par mise en correspondance de blocs qui est le seul réellement capable de traiter les mouvements de fortes amplitudes.

Les figures 3.3 et 3.5 présente des résultats d'estimation de flot optique avec nos deux estimateurs entre différentes images de la séquence *Flowergarden*. Les régions présentant les principales difficultés sont bien montrées par les cartes binaires de cohérence « avant »/« arrière » : les estimations « avant » et « arrière » avant filtrage sont incohérentes dans les zones homogènes correspondant au ciel, et dans les zones d'occultation (bord gauche de l'image et occultations liées au déplacement de l'arbre). Le filtre bilatéral permet de supprimer du bruit et de propager des informations fiables dans les zones d'occultation tout en préservant les discontinuités de mouvement et en les ajustant aux contours détectés dans l'image originale.

3.2 Modèles de mouvement

Les méthodes d'estimation de champs denses de mouvement présentées précédemment sont des méthodes locales qui requièrent l'estimation d'un vecteur de mouvement pour chaque pixel ou bloc de pixels. Dans le cas des méthodes du type Lucas et Kanade, la modélisation est localement translationnelle. Dans le cas des méthodes du type Horn et Schunck, c'est simplement la régularité spatiale du flot qui est favorisée. Les deux types de méthodes conduisent donc à estimer un nombre important de paramètres. Dans la mesure où la segmentation au sens du mouvement vise à extraire des caractéristiques pertinentes de la scène en terme de mouvement, il est bon de limiter ce nombre de paramètres, d'une part pour contribuer à la stabilité de l'estimation, et d'autre part pour travailler avec une information de plus haut niveau que le flot optique.

Selon l'hypothèse standard la plus répandue pour la segmentation d'une séquence d'images au sens du mouvement, le mouvement 2D d'un objet 3D projeté dans l'image suit généralement un modèle paramétrique de faible complexité. L'hypothèse est notamment valide si l'objet 3D est planaire. Les modèles paramétriques généralement retenus sont les modèles affine et projectif. Les éventuelles erreurs locales de modélisations sont tolérées et leurs effets sont masqués via la définition de contraintes spatiales et temporelles qui rendent possible la robustesse du processus de segmentation.

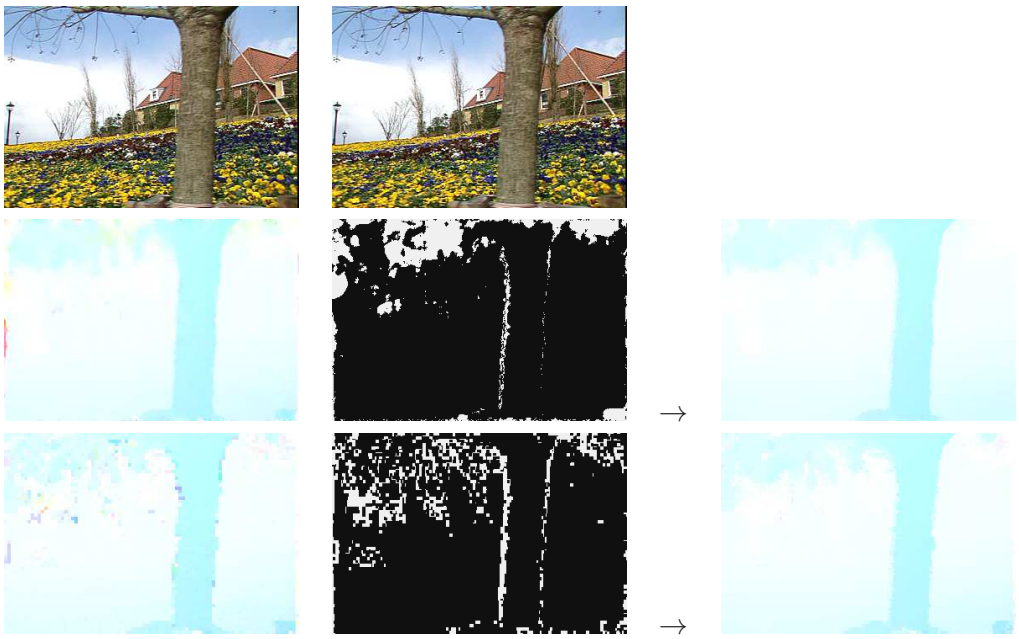


FIGURE 3.3 – Exemples de flots optiques obtenus par nos deux estimateurs et utilisation du filtre bilatéral. Première ligne : Images originales 0 et 1 de la séquence *Flowergarden*. Deuxième ligne : Estimation du flot optique « avant » de l'image 0 vers l'image 1 avec notre estimateur différentiel ; masque binaire indiquant (en blanc) les régions où la cohérence « avant »/« arrière » avant filtrage bilatéral n'est pas respectée ; flot filtré après une itération. La troisième ligne est similaire à la deuxième avec notre estimateur par mise en correspondance de blocs. Le codage couleur est donné par la figure 3.4.

3.2.1 Le modèle affine

Dans les méthodes de segmentation basée sur le mouvement, de nombreux auteurs optent pour le modèle affine complet à 6 paramètres pour décrire le mouvement d'une couche. Bouthemy et François [Bouthemy 93] justifient ce choix en argumentant que le modèle affine fournit un bon compromis entre sa faible complexité et la précision de ses estimations. Sa pertinence en tant que descripteur de mouvement rigide ne permet toutefois pas de considérer des mouvements plus complexes comme des déformations d'objets non-rigides ou bien des mouvements en dehors du plan image.

Le modèle affine de mouvement consiste en 6 degrés de liberté et peut décrire les mouvements rigides rencontrés communément dans les séquences d'images : translation, rotation, changement d'échelle, cisaillement et toute combinaison linéaire de ces mouvements.

Soit $\mathcal{A}$ la fonction de déplacement associée au vecteur de paramètres $(a_{x0}, a_{xx}, a_{xy}, a_{y0}, a_{yx}, a_{yy})^T$, et $\mathbf{p}$ un pixel de coordonnées x et y. Le mouvement affine

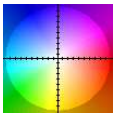


FIGURE 3.4 – Codage couleur du flot optique. La couleur indique l'orientation des vecteurs tandis que la saturation indique leur amplitude. Ce codage est celui qui est utilisé sur le site d'évaluation de l'équipe Vision de Middlebury College [Baker 07]. Dans tout ce document, lorsque ce codage est utilisé, on normalise l'amplitude des vecteurs par l'amplitude du déplacement maximum autorisé.

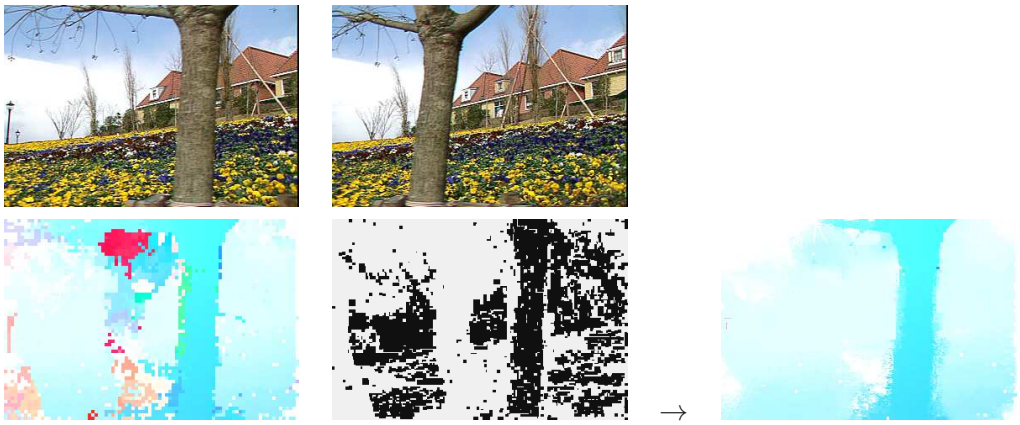


FIGURE 3.5 – Exemple de flot optique obtenu par notre estimateur de mise en correspondance de blocs et utilisation du filtre bilatéral. Première ligne : Images originales 0 et 20 de la séquence *Flowergarden*. Deuxième ligne : Estimation du flot optique « avant » de l'image 0 vers l'image 20 avec notre estimateur par mise en correspondance de blocs ; masque binaire indiquant (en blanc) les régions où la cohérence « avant »/« arrière » avant filtrage bilatéral n'est pas respectée ; flot filtré après une itération. Le codage couleur est donné par la figure 3.4.

est défini par l'équation :

$$\mathbf{dp}(x,y) = \mathcal{A}(x,y) = \begin{pmatrix} a_{x0} + a_{xx} \cdot x + a_{xy} \cdot y \\ a_{y0} + a_{yx} \cdot x + a_{yy} \cdot y \end{pmatrix} , \tag{3.11}$$

où $\mathbf{dp}(x,y)$ est le vecteur de mouvement au pixel $\mathbf{p}$ relatif au modèle affine.

Si ce mouvement est défini de $\mathbf{p} = (x,y)^T$ à $\mathbf{p'} = (x',y')^T$, on a donc :

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} a_{x0} + a_{xx} \cdot x + a_{xy} \cdot y \\ a_{y0} + a_{yx} \cdot x + a_{yy} \cdot y \end{pmatrix} . \tag{3.12}$$

3.2.2 Le modèle projectif

Le modèle affine est une simplification d'un modèle plus complexe à 8 paramètres, le modèle projectif, qui permet de représenter le mouvement du projeté d'un plan 3D

de l'espace dans le plan image. Ce mouvement est aussi appelé « homographie ».

Soit $(h_1, h_2, \cdots, h_8)^T$ le vecteur de paramètres associés au mouvement homographique permettant de passer de $\mathbf{p}$ à $\mathbf{p}'$, on a alors :

$$\begin{cases} x' &= \dfrac{h_1 \cdot x + h_2 \cdot y + h_3}{h_7 \cdot x + h_8 \cdot y + 1} \\ \\ y' &= \dfrac{h_4 \cdot x + h_5 \cdot y + h_6}{h_7 \cdot x + h_8 \cdot y + 1} \end{cases} . \tag{3.13}$$

3.2.3 Modèles non paramétriques

D'autres modélisations non paramétriques de mouvement ont été proposées.

Weiss [Weiss 97] introduit une modélisation qui inclut un *a priori* de régularité, non pas sur toute l'image, mais au sein de chaque couche de mouvement. Cette modélisation requiert l'estimation de mélanges non paramétriques via une variante de l'algorithme EM et permet de segmenter une séquence d'images en décrivant les données spatio-temporelles au moyen de plusieurs champs de mouvement lisses.

Fablet [Fablet 01] utilise des modélisations statistiques non paramétriques de l'information de mouvement dans des séquences d'images pour des problèmes de reconnaissance de mouvement, d'extraction de régions d'intérêt ou d'indexation.

3.2.4 Modèle retenu

Pour nos travaux, nous retenons le modèle affine complet à 6 paramètres pour sa faible complexité et la précision de ses estimations. Son utilisation reste pertinente dans la mesure où nous estimons des mouvements entre images faiblement éloignées : soit des images consécutives, soit des images appartenant à un même intervalle de courte durée (quelques dizaines d'images au maximum selon la complexité des mouvements en présence). La prise en compte de tels intervalles courts est compatible avec notre contexte de post-production. Dans notre cas il est donc raisonnable de supposer que les mouvements peuvent être approchés par des modèles affines, ce qui serait plus rarement vrai si l'on considérait des intervalles nettement plus longs.

3.2.5 Estimation des paramètres du modèle affine

Nous choisissons d'estimer les paramètres d'un modèle affine par régression linéaire sur les vecteurs de mouvement du flot optique contenus sur un support donné. En complément des méthodes existantes des moindres carrés (MC), de la moindre médiane des carrés (LMedS) et des moindres carrés pondérés itérés (MCPI), nous proposons une régression linéaire par méthode des moindres carrés pondérés (MCP) où le poids $w(\mathbf{dp})$ de chaque vecteur $\mathbf{dp}$ estimé en un pixel $\mathbf{p}$ de l'image I_{t_1} et par rapport à l'image I_{t_2} est donné par une mesure de confiance basée sur la cohérence « avant »/« arrière » :

$$w(\mathbf{dp}) = \max\Big(0, 1 - \frac{\|\mathbf{dp} + \mathbf{dp}'\|^2}{\tau}\Big) , \tag{3.14}$$

où $\|\cdot\|$, $\mathbf{dp}'$ et τ restent inchangés par rapport à la fonction d'occultation (3.10).

Notons $\mathcal{R}$ la région de validité d'un modèle affine de mouvement défini comme étant le vecteur de paramètres $\mathbf{a} = (a_{x0}, a_{xx}, a_{xy}, a_{y0}, a_{yx}, a_{yy})^T$, avec $\mathbf{a_x} = (a_{x0}, a_{xx}, a_{xy})^T$ et $\mathbf{a_y} = (a_{y0}, a_{yx}, a_{yy})^T$. Alors pour un pixel $(x, y)^T \in \mathcal{R}$ pour lequel un vecteur de mouvement $(dx, dy)^T$ a été estimé, on a les relations :

$$dx = \mathbf{\Phi}^T \mathbf{a_x} \qquad \text{et} \qquad dy = \mathbf{\Phi}^T \mathbf{a_y} \ , \tag{3.15}$$

où $\mathbf{\Phi} = (1, x, y)^T$ est le vecteur de régression au pixel $(x, y)^T$.

Dans la méthode des moindres carrés, on cherche à estimer $\mathbf{a}$ de sorte à minimiser l'erreur quadratique suivante :

$$\sum_{(x,y)^T \in \mathcal{R}} \Big\| (\mathbf{\Phi}^T \mathbf{a_x}, \mathbf{\Phi}^T \mathbf{a_y}) - (dx, dy) \Big\|^2 \ . \tag{3.16}$$

La régression est appliquée séparément sur chacune des composantes horizontale et verticale car $\mathbf{a_x}$ (resp. $\mathbf{a_y}$) ne dépend que de dx (resp. dy). La solution de ce problème des moindres carrés est :

$$(\mathbf{a_x}, \mathbf{a_y}) = \Big(\sum_{(x,y)^T \in \mathcal{R}} \mathbf{\Phi}\mathbf{\Phi}^T \Big)^{-1} \Big(\sum_{(x,y)^T \in \mathcal{R}} \mathbf{\Phi}(dx, dy) \Big) \ . \tag{3.17}$$

Cependant, dans notre cas les vecteurs de mouvement $(dx, dy)^T$ estimés ne sont pas tous aussi fiables les uns que les autres. En utilisant la méthode des moindres carrés pondérés, notre idée est de donner moins d'importance, dans la détermination du modèle, aux vecteurs affligés d'une faible confiance telle que définie à l'équation (3.14).

Ainsi, nous ne cherchons plus à minimiser l'erreur précédente (3.16) mais la quantité suivante :

$$\sum_{(x,y)^T \in \mathcal{R}} \Big(w(x, y) \Big\| (\mathbf{\Phi}^T \mathbf{a_x}, \mathbf{\Phi}^T \mathbf{a_y}) - (dx, dy) \Big\|^2 \Big) \ , \tag{3.18}$$

où $w(x, y) = w(\mathbf{dp})$ représente la confiance associée au vecteur de mouvement estimé pour le pixel $\mathbf{p} = (x, y)^T$.

La solution de ce problème des moindres carrés pondérés est :

$$(\mathbf{a_x}, \mathbf{a_y}) = \Big(\sum_{(x,y)^T \in \mathcal{R}} (w(x, y) \mathbf{\Phi}\mathbf{\Phi}^T) \Big)^{-1} \Big(\sum_{(x,y)^T \in \mathcal{R}} (w(x, y) \mathbf{\Phi}(dx, dy)) \Big) \ . \tag{3.19}$$

Nous comparons notre méthode aux méthodes existantes MC, LMeds et MCPI. La figure 3.6 donne une idée de la précision des différentes méthodes dans le cas où l'on dispose d'une carte de segmentation connue de l'image correspondant à l'instant t_1 et des champs denses « avant » et « arrière » estimés entre t_1 et t_2. Pour montrer la précision et la robustesse des méthodes, on choisit volontairement un cas non favorable où $t_1 = 1$ et $t_2 = 20$, avec des champs de mouvement obtenus par mise en

correspondance de blocs, sans filtrage bilatéral. Pour chacune des méthodes on estime les paramètres des modèles affines correspondant aux différentes régions. Plutôt que de montrer le champ dense paramétré synthétisé, on utilise ces modèles affines pour projeter la carte de segmentation à l'instant t_2, et nous montrons la superposition des frontières de mouvement projetées sur l'image originale.

La projection se fait en respectant l'ordre de profondeur des couches : de la couche la plus proche à la couche la plus éloignée sans écrasement de données autorisé. L'étiquette $f_{\mathbf{p}}$ portée par un pixel $\mathbf{p}$ à l'instant t_1 est copiée aux 4 pixels les plus proches de $\mathbf{p} + \mathcal{A}_{f_{\mathbf{p}}}(\mathbf{p})$ à l'instant t_2, où $\mathcal{A}_{f_{\mathbf{p}}}$ est la fonction de déplacement affine estimée pour la région d'étiquette $f_{\mathbf{p}}$ de t_1 vers t_2. Si $\mathbf{p}$ est situé sur une frontière de mouvement alors son étiquette n'est copiée qu'au pixel le plus proche. Les régions qui apparaissent restent sans étiquette.

Ce processus de projection est utilisé pour établir des prédictions de cartes de segmentation d'un instant à l'autre dans nos schémas séquentiels.

L'estimation des paramètres pourrait également se faire de manière directe sans avoir à estimer de champ dense de mouvement au préalable. Néanmoins, disposer des champs denses de mouvement pour une utilisation dans un critère de mouvement basé sur la cohérence entre flot optique estimé et flot optique paramétré synthétisé est une possibilité séduisante. Donc quitte à ce que ces champs soient estimés, nous les utilisons également pour estimer les paramètres des modèles.

Nous venons de présenter les méthodes que nous utilisons pour l'estimation du flot optique et pour l'estimation de modèles affines de mouvement. Ces informations de mouvement vont pouvoir être exploitées en vue de la segmentation au sens du mouvement. Néanmoins nos estimateurs et la modélisation affine retenue présentent des limites. La segmentation d'une séquence d'images au sens du mouvement ne peut pas reposer sur un critère unique de mouvement. Elle doit s'appuyer également sur d'autres critères tels que l'apparence couleur et la cohérence spatiale et temporelle.

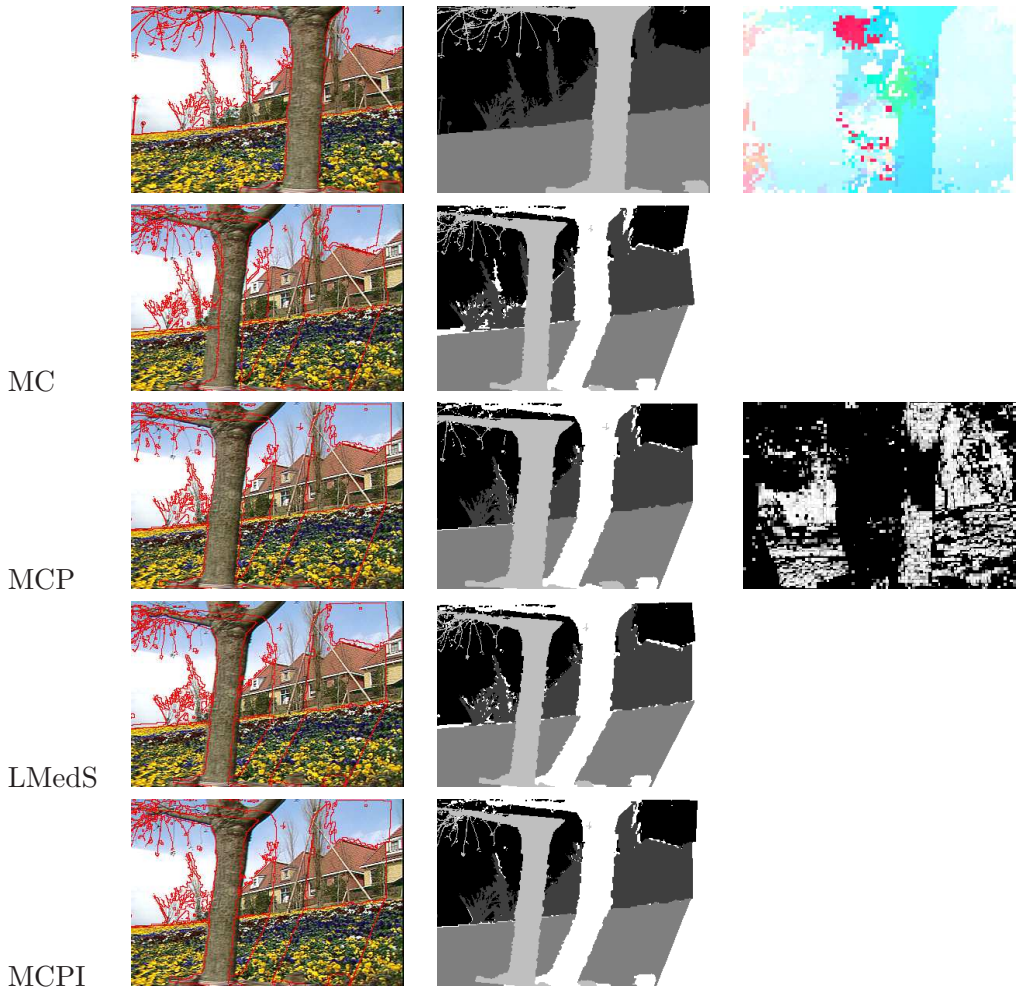


FIGURE 3.6 – Vérification de la précision des modèles affines estimés par différentes méthodes. Première ligne : image 1 avec frontières de mouvement en superposition, carte de segmentation donnée, champ dense estimé de l'image 1 vers l'image 20 à partir duquel sont estimés les paramètres des modèles affines. Lignes suivantes : frontières projetées sur l'image 20 grâce aux modèles affines des différentes régions, carte de segmentation projetée (en blanc, les zones visibles sur l'image 20 mais sans étiquette car invisibles sur l'image 1). Troisième ligne, colonne de droite, carte de confiance utilisée pour la pondération des vecteurs dans notre méthode (valeurs allant de 0 en noir à 1 en blanc). Au vu de la localisation des frontières projetées (première colonne), la méthode MC est clairement la moins précise, notamment au niveau du tronc. Les estimations obtenues en une seule itération avec notre méthode MCP semblent aussi précises que celles obtenues par les méthodes LMedS et MCPI qui requièrent plus d'itérations. Celles-ci sont plus précises au niveau de la branche horizontale mais moins précises sur le tronc lui-même. Les branches fines posent systématiquement problème à cause d'une légère rotation en dehors du plan image. Ce type d'estimation entre images éloignées (ici, images 1 et 20) est le pire des cas que nous rencontrons dans nos expérimentations (estimation entre les images les plus éloignées d'un même intervalle temporel dans notre second schéma, voir sous-section 4.1.3).

Chapitre 4

Segmentation dense d'une séquence d'images au sens du mouvement

Dans ce chapitre on s'intéresse à proprement parler à la segmentation dense de séquences d'images au sens du mouvement. Dans nos travaux, cette segmentation se doit d'être doublement précise, premièrement au niveau des supports des couches extraites, deuxièmement au niveau des estimations de mouvement.

Prenons un exemple pour illustrer notre propos. Supposons que l'on souhaite générer les mosaïques de chacune des couches de mouvement. D'une part, des supports de couches précis sont nécessaires sans quoi la mosaïque d'une couche 1 risque de contenir également des fragments appartenant en réalité à une couche 2. D'autre part, une modélisation précise du mouvement est nécessaire afin que le regroupement des supports successifs de la couche 1 se fasse sans problème d'alignement.

Par ailleurs, la segmentation doit être robuste aux occultations, cohérente temporellement tout au long de la séquence, et enfin cohérente avec les éventuelles indications fournies par un opérateur.

Le principe de nos deux premiers algorithmes a été présenté au début du document (sous-section 1.6.2). Dans nos travaux, le nombre n de couches à extraire est supposé constant tout au long de la séquence. Si ce n'est pas le cas, on se ramène à une série d'intervalles temporels pour lesquels ceci est vérifié, et on traite chaque intervalle indépendamment [1].

On définit le problème de segmentation comme un problème de minimisation d'une fonction énergie $E(f)$. Cette fonction d'énergie dépend de la fonction d'étiquetage f qui à tout pixel $\mathbf{p}$ associe une étiquette $f_{\mathbf{p}}$. On a donc $f = (f_{\mathbf{p}})_{\mathbf{p}\in\mathcal{P}}$ avec $f_{\mathbf{p}} \in [1, n]$ et

1. En réalité notre premier schéma séquentiel supporte la disparition partielle ou totale d'une couche mais ne supporte ni l'apparition d'une nouvelle couche, ni la réapparition d'une couche ayant totalement disparu. Dans les deux derniers cas, les cartes de segmentation prédites ne comportent pas de régions correspondant à ces couches, ce qui dans notre schéma empêche l'estimation des modèles de mouvement correspondants. Ces régions doivent être ajoutées manuellement, soit avant le lancement du traitement séquentiel, soit en temps voulu en interrompant momentanément le processus.

$\mathcal{P}$ l'ensemble des pixels à segmenter.

Comme suggéré au chapitre précédent, même si le critère de mouvement est naturellement celui sur lequel on se base principalement pour procéder à la segmentation au sens du mouvement, cette segmentation ne peut pas reposer sur ce critère unique. Ce chapitre présente donc les autres critères que nous utilisons, ainsi que les différents termes associés utilisés dans les fonctions d'énergie de nos deux schémas séquentiels. Ces fonctions d'énergie seront la somme de plusieurs termes unaires ou binaires.

4.1 Critère lié au mouvement

L'idée est d'attribuer un coût au fait d'associer à un pixel $\mathbf{p}$ le modèle de mouvement correspondant à la couche d'indice $f_{\mathbf{p}}$. Il existe pour cela plusieurs variantes du critère qui induisent chacune la définition d'un terme de pénalité basé sur le mouvement.

4.1.1 Terme de mouvement basé sur la DFD

Soit $D_{\mathbf{p}}(f_{\mathbf{p}})$ le terme d'attache aux données qui indique combien coûte le fait d'associer à un pixel $\mathbf{p}$ le modèle de mouvement correspondant à la couche d'indice $f_{\mathbf{p}}$. Alors, selon cette première variante du critère de mouvement, le coût $D_{\mathbf{p}}(f_{\mathbf{p}})$ est défini comme suit :

$$D_{\mathbf{p}}(f_{\mathbf{p}}) = \arctan(\|I_t(\mathbf{p}) - I_{t+1}(\mathbf{p}')\|^2 - \tau_1) + \frac{\pi}{2} \ , \tag{4.1}$$

où $\mathbf{p}'$ est le correspondant dans I_{t+1} de $\mathbf{p}$ dans I_t selon le modèle affine de mouvement « avant » $\mathcal{A}_{f_{\mathbf{p}}}$ estimé de l'instant t vers l'instant $t+1$ et associé à la couche d'indice $f_{\mathbf{p}}$ ($\mathbf{p}' = \mathbf{p} + \mathcal{A}_{f_{\mathbf{p}}}(\mathbf{p})$). Le paramètre de seuil τ_1 dépend du bruit présent dans la séquence. La version lissée (par la fonction arctangente) de la fonction échelon permet de distinguer les faibles pénalités (pixels bien classés) des fortes pénalités (pixels mal classés ou pixels occultés) tout en constituant un bon compromis entre la fonction échelon classique (ou fonction de Heaviside) et une fonction quadratique tronquée. La version lissée de la fonction échelon est représentée à la figure 4.1.

Ce terme basé sur la DFD ne peut cependant pas être utilisé de manière convenable dans les régions d'occultations où les pixels n'ont pas de correspondant dans l'image I_{t+1}.

4.1.2 Extension à un triplet d'images du terme de mouvement basé sur la DFD

Nous avons adapté le terme précédent à notre approche séquentielle basée sur les triplets d'images consécutives. Pour segmenter l'image courante I_t, on dispose alors des modèles de mouvement « avant » et « arrière » que l'on confronte de manière à rendre utilisable le terme de mouvement dans les régions d'occultations. On suppose effectivement qu'un pixel visible à l'instant t mais occulté à l'instant $t+1$ (resp. $t-1$)

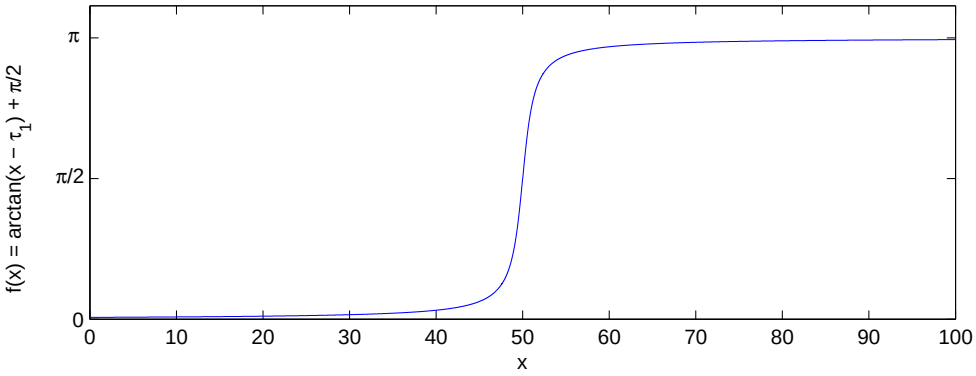


FIGURE 4.1 – Version lissée de la fonction échelon, ici avec $\tau_1 = 50$.

est visible à l'instant $t-1$ (resp. $t+1$), ce qui est vérifié dans la majorité des cas. Ainsi, l'équation (4.1) devient :

$$D_{1\mathbf{p}}(f_{\mathbf{p}}) = \min_{\zeta \in \{-1,+1\}} (\arctan(\left\| I_t(\mathbf{p}) - I_{t+\zeta}(\mathbf{p'}_{\zeta}) \right\|^2 - \tau_1) + \frac{\pi}{2}) \ , \tag{4.2}$$

où $\mathbf{p'}_{\zeta}$ est le correspondant dans $I_{t+\zeta}$ de $\mathbf{p}$ dans I_t selon le modèle affine de mouvement (« avant » ou « arrière ») associé à la couche d'indice $f_{\mathbf{p}}$.

Pour toutes nos expérimentations, nous fixons empiriquement $\tau_1 = 50$. Pour des séquences plus bruitées, la valeur de τ_1 pourra être légèrement augmentée pour compenser le bruit (cf. [Xiao 05b, Dupont 06a]).

En se focalisant uniquement sur le triplet I_{t-1}, I_t, I_{t+1} la fenêtre temporelle utilisée par ce terme n'est pas assez large pour pouvoir distinguer certaines couches les unes des autres si l'on est en présence de mouvements de faibles amplitudes. Pourtant il peut suffire de s'éloigner de l'instant t pour faciliter l'analyse de la scène et pouvoir clairement distinguer les différentes couches. On peut donc penser à une nouvelle extension temporelle du terme de mouvement à un k-uplet, ce qui implique alors d'établir $k-1$ comparaisons entre images originales et donc de disposer d'autant de modèles de mouvement pour chacune des couches. Ce type d'extension temporelle peut donc s'avérer coûteux.

4.1.3 Terme de mouvement d'image à mosaïque

Par souci de cohérence avec les sous-sections précédentes, cette sous-section présente uniquement le terme d'énergie basé sur des mises en correspondance entre images et mosaïques. La génération des mosaïques de couche de mouvement fait partie intégrante de notre second schéma séquentiel de segmentation. Son processus est décrit à la section 4.7.

Dans notre approche séquentielle basée sur les mosaïques de couches de mouvement, la séquence est partitionnée en plusieurs intervalles temporels $[t_a, t_b]$ traités séparément. Sur chaque intervalle, l'approche reste séquentielle mais les mouvements mis eu jeu

sont estimés entre l'instant courant t et les instants de référence t_a et t_b respectivement antérieur et postérieur à t. Une nouvelle adaptation du terme (4.2) permet le calcul du résidu non pas entre 2, 3, ..., k images originales de la séquence, mais entre l'image originale I_t en cours de traitement et les deux mosaïques de référence correspondant à la couche considérée.

Pour une illustration de ces mises en correspondance entre l'image courante à segmenter et les deux ensembles de mosaïques de couche, on peut se reporter à la figure 1.6 présentée au début du document.

Les mosaïques, notées $M_{a,f_\mathbf{p}}$ et $M_{b,f_\mathbf{p}}$, contiennent respectivement plus d'informations relatives à la couche $f_\mathbf{p}$ que n'en contiennent les images originales I_{t_a} et I_{t_b} elles-mêmes. Ceci est possible puisque chacune de ces mosaïques est une accumulation de tous les éléments de la couche $f_\mathbf{p}$ qui ont été visibles dans les images précédemment segmentées, éléments qui peuvent depuis avoir disparu mais qui sont également susceptibles de réapparaître.

Nous entendons ainsi remplacer $k-1$ comparaisons entre images originales par seulement deux comparaisons entre une image originale et une mosaïque. Le coût $D_{2\mathbf{p}}(f_\mathbf{p})$ est alors défini comme suit :

$$D_{2\mathbf{p}}(f_\mathbf{p}) = \min_{\zeta\in\{a,b\}} \left(r_\zeta(\mathbf{p}, f_\mathbf{p})\right) \ , \tag{4.3}$$

$$r_\zeta(\mathbf{p}, f_\mathbf{p}) = \begin{cases} \arctan\left(\left\| I_t(\mathbf{p}) - M_{\zeta,f_\mathbf{p}}(\mathbf{p}'_\zeta)\right\|^2 - \tau_2\right) + \frac{\pi}{2} \\ \qquad \text{si } \mathbf{p}'_\zeta \text{ appartient au support de la mosaïque} \\ +\beta \quad \text{sinon} \end{cases} , \tag{4.4}$$

où $\mathbf{p}'_\zeta$ est la position associée dans la mosaïque $M_{\zeta,f_\mathbf{p}}$ à $\mathbf{p}$ dans I_t selon le modèle affine de mouvement $\mathcal{A}_{\zeta,f_\mathbf{p}}$ estimé pour la couche $f_\mathbf{p}$ de I_t vers I_{t_ζ} ($\mathbf{p}'_\zeta = \mathbf{p} + \mathcal{A}_{\zeta,f_\mathbf{p}}(\mathbf{p})$).

Pour toutes nos expérimentations, nous fixons $\tau_2 = 50$ et $\beta = 10$ de manière empirique.

4.1.4 Variantes des termes précédents faisant appel à la corrélation croisée normalisée

Dans les termes (4.1), (4.2) et (4.4), la différence d'intensité au niveau du pixel est calculée en utilisant une fonction d'interpolation bilinéaire pour estimer l'intensité aux positions non entières.

La simple différence d'intensité de pixel à pixel peut être remplacée par une corrélation croisée normalisée sur un voisinage prédéfini pour lisser les erreurs résiduelles tout en améliorant la robustesse aux éventuels changements de conditions d'illumination. Dans (4.1) (et de la même façon dans (4.2) et (4.4)), on remplace $\|I_t(\mathbf{p}) - I_{t+1}(\mathbf{p}')\|$ par le résidu $r(\mathbf{p}, f_\mathbf{p})$ suivant :

$$r(\mathbf{p}, f_\mathbf{p}) = 50 \times (1 - \mathrm{CCN}(I_t(\mathbf{p}), I_{t+1}(\mathbf{p}'))) \ , \tag{4.5}$$

où la fonction CCN est la corrélation croisée normalisée entre l'imagette extraite de I_t centrée sur $\mathbf{p}$ et l'imagette correspondante (obtenue à nouveau par interpolation

bilinéaire) dans I_{t+1} centrée sur $\mathbf{p}'$ le correspondant de $\mathbf{p}$ selon le mouvement de la couche $f_{\mathbf{p}}$. La taille de la fenêtre de voisinage reste à définir. [Dupont 06a] obtient de bons résultats avec une fenêtre 3×3 qui fournit un bon compromis entre robustesse, localisation et complexité de calculs.

4.1.5 Terme de mouvement basé sur un résidu entre vecteur issu d'un flot optique et vecteur synthétisé à partir d'un modèle paramétrique

Les termes précédents reposant sur des différences d'intensité, ils ne sont pas discriminants dans les régions homogènes. En effet, dans ces régions, il est fréquent que plusieurs modèles de mouvement donnent un faible résidu, signe d'une ambiguïté de classification.

Il semble donc intéressant de baser le critère de mouvement sur un résidu calculé entre modèle affine de mouvement et vecteur de mouvement issu du flot optique. Le résidu $r(\mathbf{p}, f_{\mathbf{p}})$ est alors défini comme suit :

$$r(\mathbf{p}, f_{\mathbf{p}}) = \left\| \mathbf{dp} - \mathcal{A}_{f_{\mathbf{p}}}(\mathbf{p}) \right\| \quad , \tag{4.6}$$

où $\mathbf{dp}$ est le vecteur de mouvement au pixel $\mathbf{p}$ issu du flot optique et $\mathcal{A}_{f_{\mathbf{p}}}(\mathbf{p})$ est le vecteur de mouvement au pixel $\mathbf{p}$ relatif au modèle affine $\mathcal{A}_{f_{\mathbf{p}}}$, le flot optique et le modèle affine étant estimés pour une même paire d'images.

Néanmoins si ce terme de mouvement semble intéressant, il est totalement dépendant de la précision locale du flot optique. On ne peut donc pas garantir qu'il soit toujours plus performant que les termes précédents basés sur les différences d'intensité.

4.1.6 Comparaison des différents termes

La figure 4.2 permet une comparaison des termes de mouvement (4.2), (4.5) et (4.6).

Le terme (4.6) basé sur le flot optique est théoriquement le meilleur à condition de disposer d'une estimation quasi parfaite du champ de mouvement dense. Une possibilité pour améliorer la précision de ce champ de mouvement dense est d'utiliser le filtre bilatéral (voir sous-section 3.1.6). Cependant notre implémentation de ce filtre s'avère très coûteuse en temps de calculs. Si l'apport de ce post-traitement est évident sur la précision de certains des vecteurs de mouvement estimés, l'apport sur la précision des modèles affines de mouvement est très faible et ne justifie pas un tel coût supplémentaire. Le terme (4.6) est donc présenté comme une alternative possible aux autres termes basés sur les différences d'intensités mais n'a pas été utilisé pour l'obtention des résultats expérimentaux présentés dans le chapitre suivant.

Le terme basé sur la corrélation croisée normalisée (4.5) avec des fenêtres 3×3 fournit une bonne précision de la localisation des frontières et améliore la cohérence spatiale de la segmentation par rapport au terme (4.2)

La figure 4.3 montre bien l'intérêt d'utiliser des informations provenant aussi bien du passé que du futur. La segmentation des zones d'occultation s'en trouve nettement améliorée.

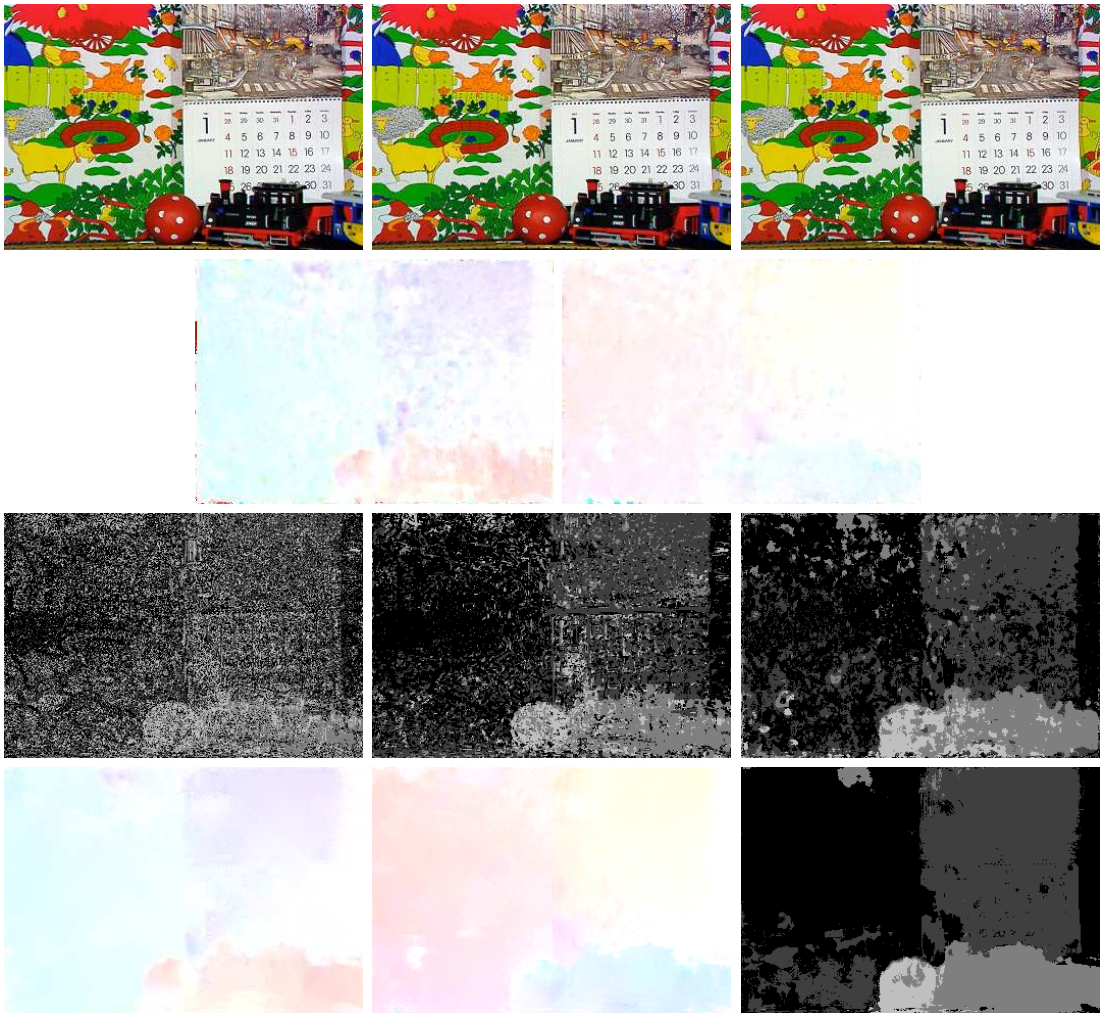


FIGURE 4.2 – Comparaison des termes de mouvement. Segmentation de l'image 2 de la séquence *Mobile & Calendar* en utilisant uniquement le critère de mouvement. On se place dans le cas où l'on dispose du triplet d'images 1, 2, 3 (première ligne de gauche à droite), des champs denses de mouvement estimés de 2 vers 1 et de 2 vers 3 (deuxième ligne) et des modèles affines correspondant pour les différentes couches. Tous les pixels de l'image 2 sont à classer et la segmentation est obtenue en sélectionnant pour chaque pixel l'étiquette qui minimise le terme unaire au pixel considéré. Troisième ligne : à gauche, terme (4.2) ; au milieu, sa version modifiée où la corrélation croisée normalisée (4.5) sur des fenêtres 3×3 vient remplacer la simple différence d'intensités ; à droite, terme (4.6). Par rapport à la simple différence d'intensité pixel à pixel, la prise en compte de la fenêtre 3×3 pour la corrélation croisée normalisée a pour effet de lisser la segmentation. Pour le terme basé sur le flot optique, la conséquence bien connue d'un lissage pas toujours approprié du champ de vecteurs est la non préservation des frontières de mouvement. Bien qu'une meilleure cohérence spatiale soit visible sur les résultats de segmentation, les frontières de régions ne sont pas précisément alignées avec les frontières réelles de mouvement. Quatrième ligne, le filtre bilatéral est appliqué sur le flot optique précédent. Ce post-traitement coûteux en temps de calculs améliore nettement la cohérence spatiale de la segmentation lorsque le terme (4.6) est utilisé mais les frontières de régions ne sont toujours pas parfaitement alignées avec les frontières réelles de mouvement (contour inférieur de la balle par exemple).

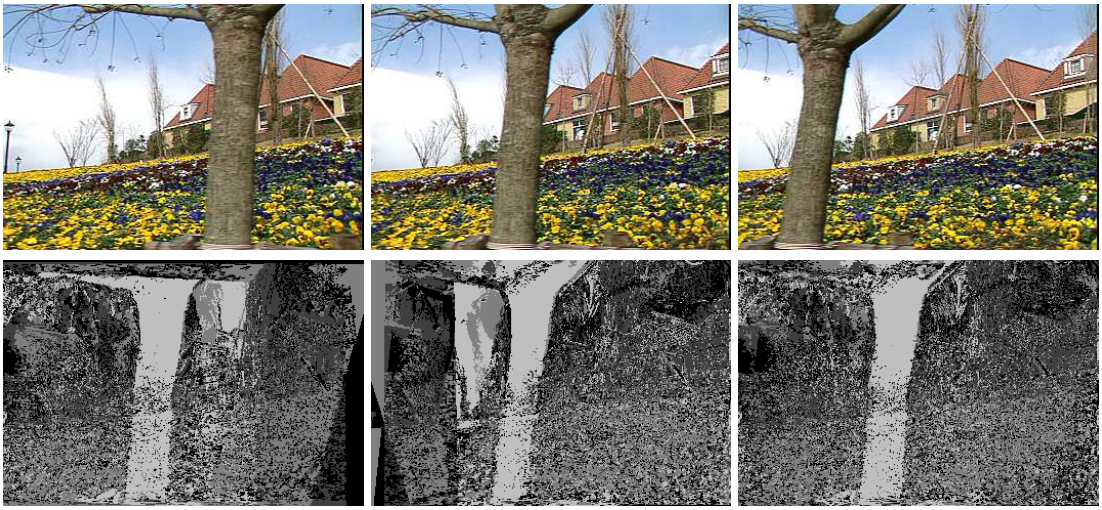


FIGURE 4.3 – Critère de mouvement et occultations. Segmentation de l'image 15 de la séquence *Flowergarden* en utilisant uniquement le critère de mouvement. On se place dans le cas où l'on dispose des images 1, 15, 30 (première ligne de gauche à droite) et des modèles affines estimés de 15 vers 1 et de 15 vers 30 pour les différentes couches. Tous les pixels de l'image 15 sont à classer et la segmentation dense est obtenue en sélectionnant pour chaque pixel l'étiquette qui minimise le terme unaire au pixel considéré. Deuxième ligne : à gauche, terme (4.1) entre les images 15 et 1 ; au milieu, terme (4.1) entre les images 15 et 30 ; à droite, utilisation des mouvements estimés dans les deux sens (terme (4.2)). Les images ont été choisies volontairement éloignées les unes des autres de manière à rendre le phénomène plus visible, mais on procède de même quel que soit le triplet d'images. En prenant en compte à la fois des informations passées et futures, la segmentation de l'image courante est améliorée dans les zones du fond occultées par l'arbre dans l'une des deux autres images.

En conclusion, nous retenons donc les variantes du terme (4.2) (pour notre premier schéma basé sur les triplets d'images consécutives) et du terme (4.3) (pour notre second schéma basé sur les mosaïques de référence) où la corrélation croisée normalisée sur des fenêtres 3×3 remplace la simple différence d'intensité.

On voit néanmoins qu'en vue de la segmentation d'une séquence d'images au sens du mouvement, aucun de ces termes de mouvement ne peut se suffire à lui-même. Dans les régions homogènes notamment, le mouvement, employé seul, ne permet pas de distinguer les couches de mouvement les unes des autres. Ces ambiguïtés peuvent être levées en ajoutant d'autres critères.

4.2 Critère d'apparence couleur

Un critère d'apparence couleur, communément utilisé pour la segmentation d'une image couleur, est ajouté dans nos schémas pour lever les ambiguïtés liées au mouve-

ment, spécialement dans les zones uniformes. L'idée est ici de modéliser la distribution de couleurs de chacune des couches et de faire l'hypothèse que les distributions des différentes couches à extraire sont distinctes les unes des autres. Grâce à un nouveau terme de pénalité, on attribue un coût $C_{\mathbf{p}}(f_{\mathbf{p}})$ au fait d'associer à un pixel $\mathbf{p}$ le modèle correspondant à la couche d'indice $f_{\mathbf{p}}$.

Pour une couche $f_{\mathbf{p}}$ donnée, nous retenons comme terme de pénalité basé sur les données couleurs la log-vraisemblance négative de la distribution de couleurs de la couche $f_{\mathbf{p}}$ [Dupont 05, Rother 04]. Cette distribution est modélisée par un mélange de gaussiennes (GMM) calculé sur les régions de la couche $f_{\mathbf{p}}$ initialement données par l'opérateur. Dans notre premier schéma, ces régions correspondent aux germes introduits sur la première image de la séquence ou sur une image suivante à partir de laquelle on réinitialise le système. (La sous-section 5.4.1 détaille l'étape de segmentation interactive d'une image.) Dans notre second schéma, ces régions correspondent directement aux mosaïques $M_{a,f_{\mathbf{p}}}$ et $M_{b,f_{\mathbf{p}}}$ avant que le processus de segmentation ne commence.

Considérons pour la couche $f_{\mathbf{p}}$ un mélange de $m_{f_{\mathbf{p}}}$ gaussiennes de dimension 3. À la gaussienne $G_{f_{\mathbf{p}},k}$ ($k \in \left[1, m_{f_{\mathbf{p}}}\right]$), on associe une moyenne $\boldsymbol{\mu}_{\{f_{\mathbf{p}},k\}}$ représentant sa couleur moyenne dans l'espace RVB, une matrice de covariance $\boldsymbol{\Sigma}_{\{f_{\mathbf{p}},k\}}$ représentant l'étendue de cette couleur, ainsi qu'un coefficient de pondération $\pi_{\{f_{\mathbf{p}},k\}}$ indiquant sa proportion dans le mélange (avec $\pi_{\{f_{\mathbf{p}},k\}} \in [0,1]$ et $\sum_{k=1}^{m_{f_{\mathbf{p}}}} \pi_{\{f_{\mathbf{p}},k\}} = 1$). Le coût $C_{\mathbf{p}}(f_{\mathbf{p}})$ est alors défini comme suit :

$$C_{\mathbf{p}}(f_{\mathbf{p}}) = -\log\left(\sum_{k=1}^{m_{f_{\mathbf{p}}}} \pi_{\{f_{\mathbf{p}},k\}} p_{\{f_{\mathbf{p}},k\}}(\mathbf{I}_t(\mathbf{p})|\boldsymbol{\mu}_{\{f_{\mathbf{p}},k\}}, \boldsymbol{\Sigma}_{\{f_{\mathbf{p}},k\}})\right) , \tag{4.7}$$

où $p_{\{f_{\mathbf{p}},k\}}(.)$ est la probabilité conditionnelle que le pixel $\mathbf{p}$ soit de la couleur définie par la gaussienne $G_{f_{\mathbf{p}},k}$.

4.2.1 Nombre de gaussiennes que comporte le mélange associé à une couche

Dans la littérature, le nombre de gaussiennes constituant le mélange associé à une couche est parfois déterminé de manière automatique, par exemple par le principe de longueur de description minimale MDL [Rissanen 83]. Le principe MDL, basé sur la théorie de l'information, permet d'identifier le modèle le plus simple à partir d'un ensemble de données, c'est-à-dire le modèle qui fournit le meilleur compromis entre la codification compacte des données et la minimisation de l'erreur de modélisation. Il se trouve cependant en pratique que le modèle retenu (optimal au sens du principe MDL) n'est parfois pas satisfaisant, contenant par exemple, dans notre cas, un nombre bien trop élevé de gaussiennes.

Nous abandonnons donc le principe MDL et préférons fixer $m_{f_{\mathbf{p}}} = 5$ par défaut pour toutes les couches. Nous laissons à l'opérateur la possibilité de modifier le nombre de gaussiennes constituant chaque mélange, les mélanges de deux couches différentes ne contenant donc pas nécessairement le même nombre de gaussiennes.

4.2.2 Estimation d'un mélange de gaussiennes

L'estimation d'un mélange de m gaussiennes à partir d'un ensemble de données (des points de l'espace RVB dans notre cas) dépend avant tout d'un algorithme qui sépare ces données en m clusters. Nous allons effectivement associer chaque point RVB du support donné à une unique gaussienne du mélange [2].

Nous utilisons l'algorithme de quantification des couleurs d'une image [Orchard 91] qui fonctionne par divisions successives de clusters. (On retrouve l'utilisation de cet algorithme pour la séparation du fond et d'un objet d'avant-plan dans une image fixe dans [Talbot 04].) Cet algorithme est détaillé ci-dessous. À l'étape 4, on sélectionne le cluster dont la gaussienne présente la plus forte variance. Ce cluster est divisé à l'étape 5.

Algorithme 3 Algorithme d'estimation d'un mélange de m gaussiennes

1: Grouper toutes les données du support dans un seul cluster C_1
2: Calculer $\boldsymbol{\mu}_1$, $\boldsymbol{\Sigma}_1$ et π_1
3: **pour** i de 2 à m **faire**
4: Trouver le cluster C_n ($n \in [1, i-1]$), dont la matrice de covariance a la plus grande valeur propre, i.e. le cluster dont la gaussienne associée est la plus étendue. (On notera $\mathbf{e}_n$ le vecteur propre associé.)
5: Diviser C_n en deux clusters : $C_i = \{\mathbf{p} \in C_n : {}^t\mathbf{e}_n \cdot \mathbf{I}(\mathbf{p}) \leq {}^t\mathbf{e}_n \cdot \boldsymbol{\mu}_n\}$ et $C_n = C_n \backslash C_i$
6: Calculer $\boldsymbol{\mu}_n$, $\boldsymbol{\mu}_i$, $\boldsymbol{\Sigma}_n$, $\boldsymbol{\Sigma}_i$, π_n et π_i
7: **fin pour**

Aux étapes 2 et 6, pour calculer la moyenne, la matrice de covariance et la proportion de chaque gaussienne, on utilise les formules suivantes :

$$\boldsymbol{\mu}_i = \frac{\sum_{\mathbf{p} \in C_i} \mathbf{I}(\mathbf{p})}{|C_i|} \ , \tag{4.8}$$

$$\boldsymbol{\Sigma}_i = (s_{jk})_{1 \leq j,k \leq 3} \text{ avec } s_{jk} = \frac{\sum_{p \in C_i} (I^j(\mathbf{p}) - \mu_i^j)(I^k(\mathbf{p}) - \mu_i^k)}{|C_i|} \ , \tag{4.9}$$

$$\pi_i = \frac{|C_i|}{\sum_j |C_j|} \ , \tag{4.10}$$

où dans (4.9), les notations en exposant correspondent aux composantes RVB. Des exemples de modélisation de distribution de couleurs par mélange de gaussiennes sont donnés aux figures 4.4 et 4.5.

La figure 4.6 présente deux résultats de segmentation utilisant seulement le terme d'apparence couleur (4.7). Pour la première image de la séquence *Mobile & Calendar*,

2. Il peut sembler préférable de déterminer pour chaque pixel une probabilité d'appartenance à chaque gaussienne ce qui permettrait l'utilisation de l'algorithme EM [Dempster 77]. Cependant d'après [Rother 04], ceci serait bien plus coûteux en temps de calcul pour un gain finalement négligeable en pratique.

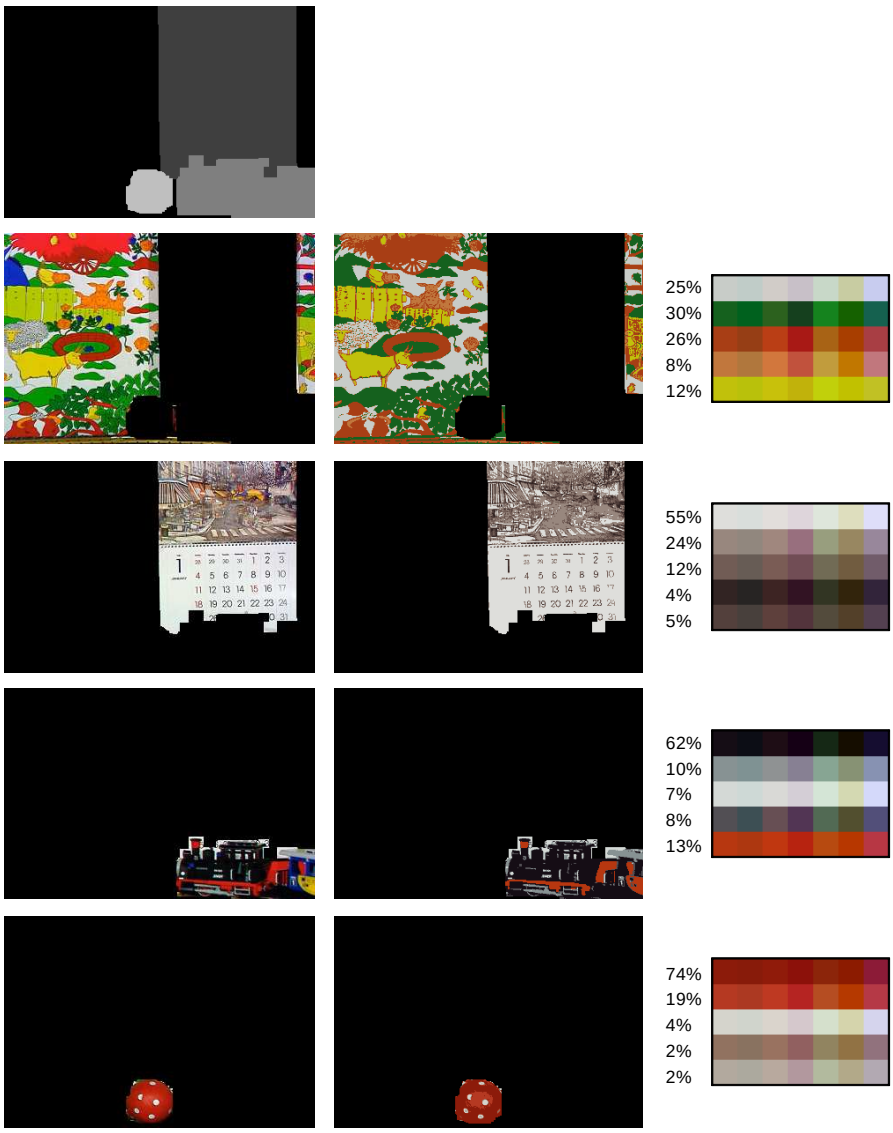


FIGURE 4.4 – Mélanges de 5 gaussiennes de couleurs pour la première image de la séquence *Mobile & Calendar*. Première ligne, carte de segmentation de référence obtenue manuellement et à partir de laquelle un mélange de 5 gaussiennes est estimé pour chaque couche. Lignes suivantes, pour chaque couche de la plus éloignée à la plus proche : à gauche, valeurs RVB originales ; au milieu, les pixels correspondants ont pris la couleur moyenne de la gaussienne à laquelle ils ont été affectés ; à droite, les palettes de couleurs où chaque gaussienne est représentée sur une ligne, le pourcentage indique la proportion de la gaussienne dans le mélange, la couleur du premier carré est la couleur moyenne, et la variance dans l'espace RVB est indiquée par les couleurs des 6 carrés suivants.

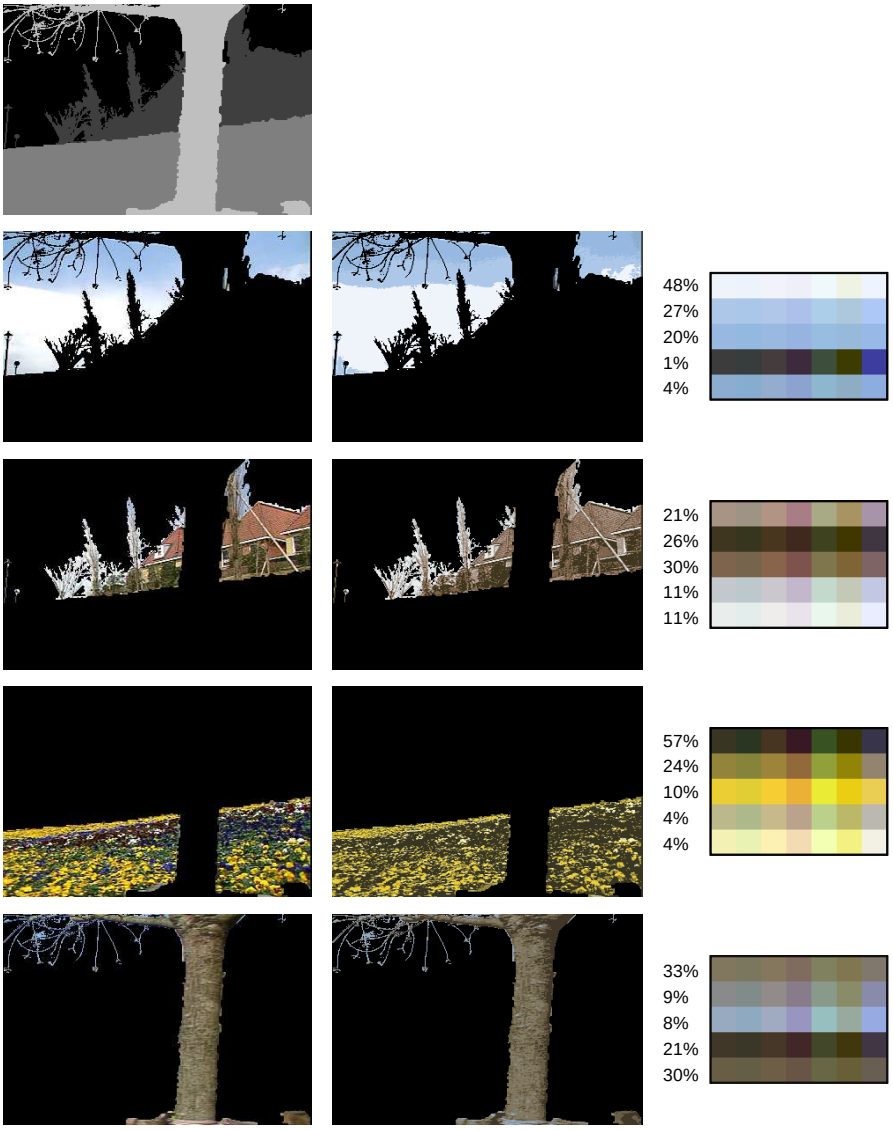


FIGURE 4.5 – Mélanges de 5 gaussiennes de couleurs pour la première image de la séquence *Flowergarden*. Première ligne, carte de segmentation de référence obtenue de manière semi-automatique et à partir de laquelle un mélange de 5 gaussiennes est estimé pour chaque couche. Lignes suivantes, pour chaque couche de la plus éloignée à la plus proche : à gauche, valeurs RVB originales ; au milieu, les pixels correspondants ont pris la couleur moyenne de la gaussienne à laquelle ils ont été affectés ; à droite, les palettes de couleurs où chaque gaussienne est représentée sur une ligne, le pourcentage indique la proportion de la gaussienne dans le mélange, la couleur du premier carré est la couleur moyenne, et la variance dans l'espace RVB est indiquée par les couleurs des 6 carrés suivants.

FIGURE 4.6 – Segmentation de la première image des séquences *Mobile & Calendar* et *Flowergarden* en utilisant seulement le critère de couleur. Tous les pixels de l'image sont à classer. La segmentation est obtenue en sélectionnant pour chaque pixel l'étiquette associée au maximum de la log-vraisemblance. Si l'on considère comme vérité de terrain les cartes de segmentation fournies pour l'estimation des mélange de gaussiennes (figures 4.4 et 4.5), ces résultats correspondent respectivement à des taux d'erreur de classification de 29% et 22%.

les ambiguïtés en termes de couleurs se situent principalement dans les régions rouges entre la balle, le train et le fond, et dans les régions blanches entre le calendrier, la balle et le fond. Pour la première image de la séquence *Flowergarden*, les ambiguïtés en termes de couleurs se situent principalement entre le tronc et les buissons devant les maisons.

4.3 Contrainte temporelle

Comme dans [Dupont 05, Xiao 04, Dupont 06b], nous introduisons un terme supplémentaire $\Psi_{\mathbf{p}}(f_{\mathbf{p}})$ pour assurer la cohérence temporelle de la segmentation entre deux images successives :

$$\Psi_{\mathbf{p}}(f_{\mathbf{p}}) = \begin{cases} 1 & \text{si } f_{\mathbf{p}} \neq f'_{\mathbf{p}} \text{ et } f'_{\mathbf{p}} \neq \emptyset \\ 0 & \text{sinon} \end{cases} , \tag{4.11}$$

où $f'_{\mathbf{p}}$ est l'étiquette prédite qui a été attribuée au pixel $\mathbf{p}$ à l'instant t lors de la prédiction [3] provenant de la segmentation établie pour l'instant $t-1$, et $\emptyset$ est l'étiquette vierge qui correspond aux régions sans étiquette prédite (théoriquement les régions apparaissant). Pour les régions où l'on dispose d'une étiquette prédite, on fait donc

3. Cette procédure de prédiction est décrite à la sous-section 4.6.1.

confiance à la prédiction en pénalisant au niveau de chaque pixel les étiquettes autres que l'étiquette prédite.

Remarquons que cette contrainte ne favorise aucune décision dans les régions apparaissant. Il est pourtant possible dans certains cas d'encourager ces régions à prendre une étiquette plutôt qu'une autre.

4.4 Contrainte de rigidité

Dans le cas de notre premier schéma basé sur les triplets d'images consécutives, pour les régions apparaissant, la contrainte temporelle ne favorise aucune étiquette. Pour ces régions, on peut s'appuyer sur le critère de mouvement, en sachant d'ores et déjà que le mouvement « arrière » vers l'image précédente ne nous apportera rien. Reste donc le mouvement « avant » vers l'image suivante, mais on a déjà précisé que pour des mouvements de faibles amplitudes ou dans des régions homogènes, le critère de mouvement n'est pas discriminant. On peut alors compter sur le critère de couleur ou sur le terme de régularisation spatiale à condition qu'au niveau de la frontière d'occultation, objet apparaissant et objet occultant soient de couleurs différentes. Malheureusement ce n'est pas toujours le cas. Dans la séquence *Carmap*, lors de la réapparition de la voiture, les premières régions à apparaître à droite du bord noir du panneau sont l'ombre et les roues noires de la voiture. À la fin de la séquence *Mobile & Calendar*, la balle rouge occulte les vaches rouges du papier peint.

Dans le premier schéma nous ne disposons pas d'informations utiles issues d'images plus éloignées temporellement. Sous certaines hypothèses il est tout de même possible de renforcer encore la cohérence temporelle de la segmentation. Nous introduisons une nouvelle contrainte de rigidité qui vient compléter la contrainte temporelle.

Considérons un objet rigide dont tout le support à un instant t était déjà visible à l'instant $t-1$, et ce, tout au long de la séquence. C'est le cas par exemple d'un objet rigide d'avant-plan entièrement visible au moins sur la première image de la séquence, qui reste non-occulté par la suite, éventuellement sortant (partiellement voire totalement) du champ de vision sans réapparaître ensuite. Ce cas est relativement fréquent pour les séquences vérifiant nos deux hypothèses : l'hypothèse classique selon laquelle la scène est composée d'un ensemble de couches dont les mouvements dans le plan image peuvent être bien décrits par des modèles affines, et l'hypothèse selon laquelle l'ordre de profondeur des couches reste constant tout au long de la séquence.

Pour un tel objet, son support à un instant t ne doit pas être différent de son support prédit [4] depuis l'instant $t-1$. Cette contrainte est plus forte que la contrainte temporelle puisqu'elle indique qu'il faut décourager l'insertion de l'étiquette de l'objet considéré dans les régions apparaissant. (C'est justement dans ces régions apparaissant que la contrainte temporelle ne permet aucune discrimination.)

Nous introduisons donc le terme supplémentaire $\Phi_{\mathbf{p}}(f_{\mathbf{p}})$:

4. La procédure de prédiction est décrite à la sous-section 4.6.1.

$$\Phi_{\mathbf{p}}(f_{\mathbf{p}}) = \begin{cases} 1 & \text{si } f_{\mathbf{p}} \in \mathcal{S} \text{ et } f'_{\mathbf{p}} = \emptyset \\ 0 & \text{sinon } , \end{cases} \tag{4.12}$$

où $\mathcal{S}$ est l'ensemble des étiquettes correspondant aux couches sur lesquelles l'opérateur souhaite appliquer la contrainte, $f'_{\mathbf{p}}$ est l'étiquette prédite au pixel $\mathbf{p}$ et $\emptyset$ désigne l'étiquette vierge correspondant aux régions sans étiquette prédite.

L'opérateur, ayant visualisé la séquence au préalable, précisera cet ensemble $\mathcal{S}$ et aura tout intérêt également à choisir si la séquence doit être traitée de la première image à la dernière, ou bien l'inverse.

4.5 Contrainte de lissage ou de régularisation spatiale

Pour assurer la consistance spatiale de la segmentation, on incite une paire de pixels voisins à prendre la même étiquette de manière à constituer des régions d'étiquetage uniforme. Pour deux pixels voisins $\mathbf{p}$ et $\mathbf{q}$, le terme binaire $V_{\mathbf{p},\mathbf{q}}(f_{\mathbf{p}}, f_{\mathbf{q}})$ que l'on utilise est un terme standard de régularisation spatiale dépendant du gradient spatial d'intensité :

$$V_{\mathbf{p},\mathbf{q}}(f_{\mathbf{p}}, f_{\mathbf{q}}) = \frac{1}{\text{dist}(\mathbf{p},\mathbf{q})} \exp(-\frac{\|I_t(\mathbf{p}) - I_t(\mathbf{q})\|^2}{2\sigma^2}) \cdot T(f_{\mathbf{p}} \neq f_{\mathbf{q}}) \ , \tag{4.13}$$

où dist() est la distance euclidienne, σ est l'écart type de la norme du gradient spatial d'intensité et $T(.)$ vaut 1 si l'argument est vrai et 0 si l'argument est faux.

La contrainte est relâchée dans les régions de forts gradients de manière à préserver les discontinuités. On fait ici implicitement l'hypothèse que les zones de forts gradients signalent la présence d'un contour d'un objet et qu'il est possible que ce contour corresponde à une frontière de mouvement.

4.6 Prédiction de la segmentation et restriction de la région à segmenter

Dans nos deux schémas séquentiels, nous proposons de prédire l'étiquetage, et donc les frontières de mouvement, d'un instant à l'autre, à partir de l'instant précédent pour lequel nous disposons de la carte de segmentation qui ne sera plus remise en cause. Pour l'instant courant, nous considérons alors, autour de ces frontières prédites, une bande d'incertitude dans laquelle l'étiquetage peut être remis en question. Notons que les régions apparaissant n'ont pas d'étiquettes prédites, leur étiquetage est donc également incertain. Dans les autres régions, nous supposons que les étiquettes prédites sont correctes et les pixels correspondants sont ignorés par le graphe. Une telle hypothèse réduit de manière significative la taille du graphe.

4.6.1 Prédiction de la segmentation

Une fois l'image courante I_t segmentée, nous disposons de la carte de segmentation f_t à partir de laquelle nous construisons la prédiction de la carte de segmentation suivante f'_{t+1}. Pour cela nous utilisons les modèles affines de mouvement estimés entre l'instant t et l'instant $t+1$ pour chaque couche. Grâce à l'hypothèse selon laquelle l'ordre de profondeur des couches reste constant tout au long de la séquence, nous procédons à la projection de chaque support de couche en respectant l'ordre de profondeur. Le support de la couche la plus éloignée est donc projeté le premier à l'instant $t+1$, ainsi de suite jusqu'au support de la couche la plus proche, en autorisant les chevauchements. Étant donné un pixel $\mathbf{p}$ appartenant à la carte de segmentation f_t, le modèle affine de mouvement $\mathcal{A}_{f_\mathbf{p}}$ estimé pour la couche $f_\mathbf{p}$ de l'instant t vers l'instant $t+1$, et la position $\mathbf{p}'$ non entière correspondant à $\mathbf{p}$ à l'instant $t+1$ ($\mathbf{p}' = \mathbf{p} + \mathcal{A}_{f_\mathbf{p}}(\mathbf{p})$), nous distinguons deux types de projection selon la position du pixel $\mathbf{p}$ dans la carte de segmentation f_t.

1. Si le pixel $\mathbf{p}$ est en contact avec une frontière de région dans f_t (c'est-à-dire qu'au moins un des voisins de $\mathbf{p}$ (au sens de la 8-connexité) porte une étiquette différente de $f_\mathbf{p}$), il transmet son étiquette $f_\mathbf{p}$ au pixel de f'_{t+1} le plus proche de la position non entière $\mathbf{p}'$.
2. Si le pixel $\mathbf{p}$ est strictement situé à l'intérieur d'une région dans f_t (ses 8 plus proches voisins portent tous l'étiquette $f_\mathbf{p}$), il transmet son étiquette $f_\mathbf{p}$ aux 4 pixels de f'_{t+1} les plus proches de la position non entière $\mathbf{p}'$.

La projection sur les 4 plus proches voisins a pour objectif de maintenir le caractère connexe de chaque région en remplissant les trous de la carte de prédiction qui seraient dus à la discrétisation des supports de couche.

Pour les pixels en contact avec les frontières, la projection de l'étiquette se fait uniquement sur le pixel le plus proche pour préserver tant que possible la localisation de ces frontières.

Les trous restants dans la carte de prédiction f'_{t+1} correspondent alors aux zones occultées à l'instant t et qui apparaissent à l'instant $t+1$.

4.6.2 Restriction de la région à segmenter

Connaissant la carte de segmentation prédite, nous définissons une bande d'incertitude autour des frontières prédites. Seuls les pixels de cette bande d'incertitude peuvent alors encore changer d'étiquette. Les autres pixels garderont leur étiquette de prédiction.

La figure 4.7 illustre, dans un cas binaire, le fait de réduire la taille de la région à segmenter.

Nous laissons à l'opérateur la possibilité de modifier la largeur w de la bande d'incertitude. Dans nos expérimentations, nous avons fixé par défaut $w = 12$. Cela inclut 5 pixels de chaque côté des frontières prédites, et un pixel supplémentaire de chaque côté pour disposer de conditions aux bords. Les pixels appartenant à ces bandes de largeur 1 sont considérés comme des germes et leur étiquetage prédit, non-remis en cause, fournit des contraintes fortes qui influencent la segmentation finale comme expliqué

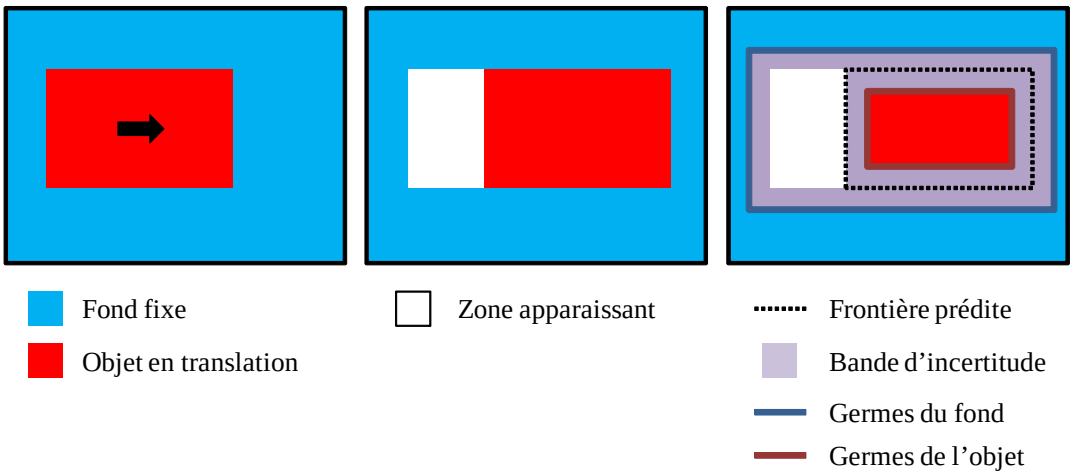


FIGURE 4.7 – Restriction de la région à segmenter, et donc restriction du graphe. Cas synthétique binaire d'un objet rectangulaire d'avant-plan en translation de gauche à droite sur un fond fixe. À gauche, la segmentation obtenue à l'instant précédent. Au milieu, la prédiction à l'instant courant. À droite, la restriction de la région à segmenter. Lorsque la contrainte de rigidité est appliquée à l'objet, on décourage l'insertion de l'étiquette « objet » dans la zone blanche qui vient d'apparaître.

dans la section 2.5. Notons bien que dans ce cas, ces germes ne sont pas introduits manuellement à chaque instant mais sont réellement déterminés automatiquement par projection d'une image à l'autre.

Le fait de réduire la région à segmenter à une bande d'incertitude étroite explique qu'au terme de « segmentation » nous préférons parfois le terme de « raffinement des frontières de mouvement ». Dans nos expérimentations, nous avons noté des gains entre 75% et 90% en terme de nombre de pixels à classifier comparé au cas où la segmentation de toute l'image est remise en cause à chaque instant. Ce gain varie en fonction de la longueur des frontières de mouvement et en fonction de l'amplitude des déplacements des différentes couches. La figure 4.8 montre la bande d'incertitude considérée lors du raffinement des frontières de l'image 2 de la séquence réelle *Carmap*.

En plus de réduire la taille du graphe et donc réduire le temps de calcul nécessaire à la minimisation de l'énergie, la restriction de la région à segmenter présente l'avantage de contraindre la segmentation aussi bien spatialement que temporellement, ce qui a pour effet de renforcer la cohérence. La segmentation est contrainte spatialement car on utilise des conditions aux bords, et temporellement car en dehors de la bande d'incertitude on se fie à la prédiction temporelle.

Notons toutefois que cet avantage peut se transformer en inconvénient. Une bonne segmentation ne peut être obtenue que si les frontières réelles de mouvement se situent dans la bande d'incertitude. En cas de mauvaise estimation des modèles affines de mouvement ou si tout bonnement le mouvement ne peut pas être bien décrit par un modèle affine, on peut se douter que frontières de mouvement prédites et frontières réelles se-

ront décalées. La bande d'incertitude étant définie autour des frontières prédites, elle ne contiendra pas nécessairement les frontières réelles. Or à l'extérieur de la bande d'incertitude, l'étiquetage est forcé (sur l'hypothèse forte de cohérence temporelle). Le raffinement des frontières prédites ne permet donc pas, dans ce cas, d'extraire les frontières réelles. La valeur du paramètre w dépend donc de la complexité des mouvements présents dans la scène. Si ces derniers ne peuvent pas être décrits correctement par les modèles affines, la valeur du paramètre w devra être augmentée.

4.7 Génération automatique des mosaïques de couches

La génération des mosaïques de couche de mouvement fait partie intégrante de notre second schéma séquentiel de segmentation. Dès qu'une image est segmentée, les supports de chaque couche sont ajustés et accumulés de manière automatique dans la mosaïque correspondante.

L'extraction de couches de mouvement consiste à extraire les supports de couches mais également à estimer les paramètres des modèles de mouvement de chaque couche. Ainsi, en utilisant les modèles de mouvement, les couches sont projetées dans les mosaïques correspondantes que nous souhaitons construire. Cela signifie que les mosaïques de référence évoluent pendant que le processus de segmentation progresse. Le support initial d'une mosaïque résulte de la segmentation semi-automatique de l'image de référence associée à cette mosaïque.

La figure 4.9 montre les mosaïques obtenues automatiquement en utilisant les 20 premières images de la séquence *Flowergarden*.

Les régions qui apparaissent pour la première fois sont copiées dans les mosaïques. Pour les régions déjà visibles précédemment, nous gardons simplement les valeurs de la première apparition sans aucune mise à jour de manière à préserver les données de référence en cas d'erreurs de classification. On peut donc rencontrer deux cas :

- soit le pixel $\mathbf{p}$ de la couche $f_{\mathbf{p}}$ dans l'image I_t a déjà été visible auparavant et a donc déjà figuré dans au moins une des images précédemment segmentées, auquel cas, l'information est déjà disponible dans les mosaïques de référence $M_{a,f_{\mathbf{p}}}$ et $M_{b,f_{\mathbf{p}}}$,
- soit le pixel $\mathbf{p}$ de la couche $f_{\mathbf{p}}$ dans l'image I_t apparaît pour la première fois, auquel cas l'information est manquante dans les mosaïques et doit être copiée :

$$\forall \zeta \in \{a, b\} \;\; M_{\zeta,f_{\mathbf{p}}}(\mathbf{p} + \mathcal{A}_{\zeta,f_{\mathbf{p}}}(\mathbf{p})) = I_t(\mathbf{p}) \;\; . \tag{4.14}$$

En cas d'erreurs de segmentation, les mosaïques sont directement affectées. En d'autres termes, les zones mal classées peuvent être copiées dans une mauvaise mosaïque. De telles erreurs ne sont pas corrigées durant la génération des mosaïques. L'opérateur aura tout de même la possibilité de les supprimer manuellement à la fin d'une première passe de segmentation. La réutilisation de ces mosaïques corrigées comme entrées d'une seconde passe de segmentation peut améliorer de manière significative la précision des frontières de mouvement.

4.8 Fonction d'énergie globale

Nous concluons ce chapitre en présentant les fonctions d'énergie globale retenues pour chacun de nos deux schémas de segmentation.

4.8.1 Premier schéma basé sur les triplets d'images consécutives

Pour notre premier schéma séquentiel basé sur les triplets d'images consécutives, la fonction d'énergie globale $E_1(f)$ à minimiser à chaque instant est :

$$E_1(f) = \lambda_1 \underbrace{\sum_{\mathbf{p}\in\mathcal{P}} D_{1\mathbf{p}}(f_\mathbf{p})}_{E_{\text{mouvement}}(f)} + \lambda_2 \underbrace{\sum_{\mathbf{p}\in\mathcal{P}} C_\mathbf{p}(f_\mathbf{p})}_{E_{\text{couleur}}(f)} + \lambda_3 \underbrace{\sum_{\mathbf{p}\in\mathcal{P}} \Psi_\mathbf{p}(f_\mathbf{p})}_{E_{\text{temporel}}(f)} + \lambda_4 \underbrace{\sum_{\mathbf{p}\in\mathcal{P}} \Phi_\mathbf{p}(f_\mathbf{p})}_{E_{\text{rigidité}}(f)} + \lambda_5 \underbrace{\sum_{(\mathbf{p},\mathbf{q})\in\mathcal{N}} V_{\mathbf{p},\mathbf{q}}(f_\mathbf{p}, f_\mathbf{q})}_{E_{\text{lissage}}(f)} , \quad (4.15)$$

où $\mathcal{P}$ est l'ensemble des pixels à segmenter, $\mathcal{N}$ est l'ensemble des paires de pixels voisins selon la 8-connexité ($\mathcal{N} \subset \mathcal{P}^2$), et $(\lambda_i)_{i\in[1,5]}$ sont des paramètres positifs qui pondèrent l'influence de chacun des termes. Ils sont fixés par l'opérateur et dépendent de la fiabilité de l'estimation de mouvement et des caractéristiques propres à la séquence.

4.8.2 Second schéma basé sur les mosaïques

Pour notre second schéma séquentiel basé sur les mosaïques de couches, la fonction d'énergie globale $E_2(f)$ à minimiser à chaque instant est :

$$E_2(f) = \lambda_1 \underbrace{\sum_{\mathbf{p}\in\mathcal{P}} D_{2\mathbf{p}}(f_\mathbf{p})}_{E_{\text{mosaïque}}(f)} + \lambda_2 \underbrace{\sum_{\mathbf{p}\in\mathcal{P}} C_\mathbf{p}(f_\mathbf{p})}_{E_{\text{couleur}}(f)} + \lambda_3 \underbrace{\sum_{\mathbf{p}\in\mathcal{P}} \Psi_\mathbf{p}(f_\mathbf{p})}_{E_{\text{temporel}}(f)} + \lambda_4 \underbrace{\sum_{(\mathbf{p},\mathbf{q})\in\mathcal{N}} V_{\mathbf{p},\mathbf{q}}(f_\mathbf{p}, f_\mathbf{q})}_{E_{\text{lissage}}(f)} , \quad (4.16)$$

où $\mathcal{P}$, $\mathcal{N}$ et $(\lambda_i)_{i\in[1,4]}$ sont définis comme pour l'équation (4.15).

Ces fonctions d'énergie globale ne sont pas simples à minimiser. La qualité des résultats dépend de la méthode de minimisation choisie. Le formalisme de coupe dans un graphe et l'algorithme d'alpha-expansion, tous deux présentés au chapitre 2, ont déjà été utilisés avec succès pour résoudre ce type de problème de minimisation, et font partie des méthodes les plus efficaces. Nous les retenons à notre tour pour la résolution de notre problème. Le chapitre suivant (chapitre 5) présente les résultats obtenus avec nos deux schémas séquentiels.

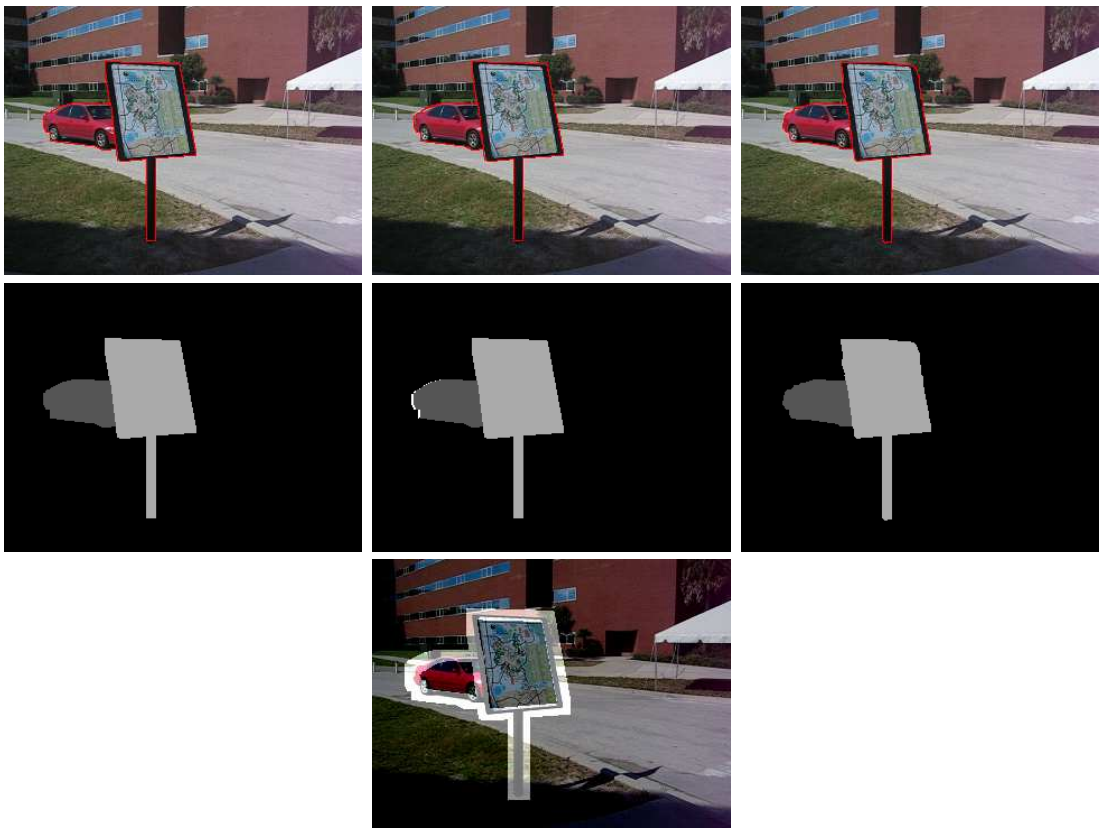


FIGURE 4.8 – Bande d'incertitude pour l'image 2 de la séquence *Carmap*. Colonne de gauche, frontières extraites et carte de segmentation pour l'image 1. Colonne du milieu, frontières prédites et carte de prédiction pour l'image 2. Dans cette carte de prédiction, les zones blanches (juste derrière la voiture) correspondent à des trous, c'est-à-dire des zones sans étiquette prédite car elles apparaissent tout juste. En bas, la région éclaircie autour des frontières prédites correspond à la bande d'incertitude. La région assombrie est ignorée lors de la construction du graphe, elle contient tous les pixels dont on ne remet pas en question l'étiquetage prédit. Colonne de droite, frontières affinées et carte de segmentation finale pour l'image 2.

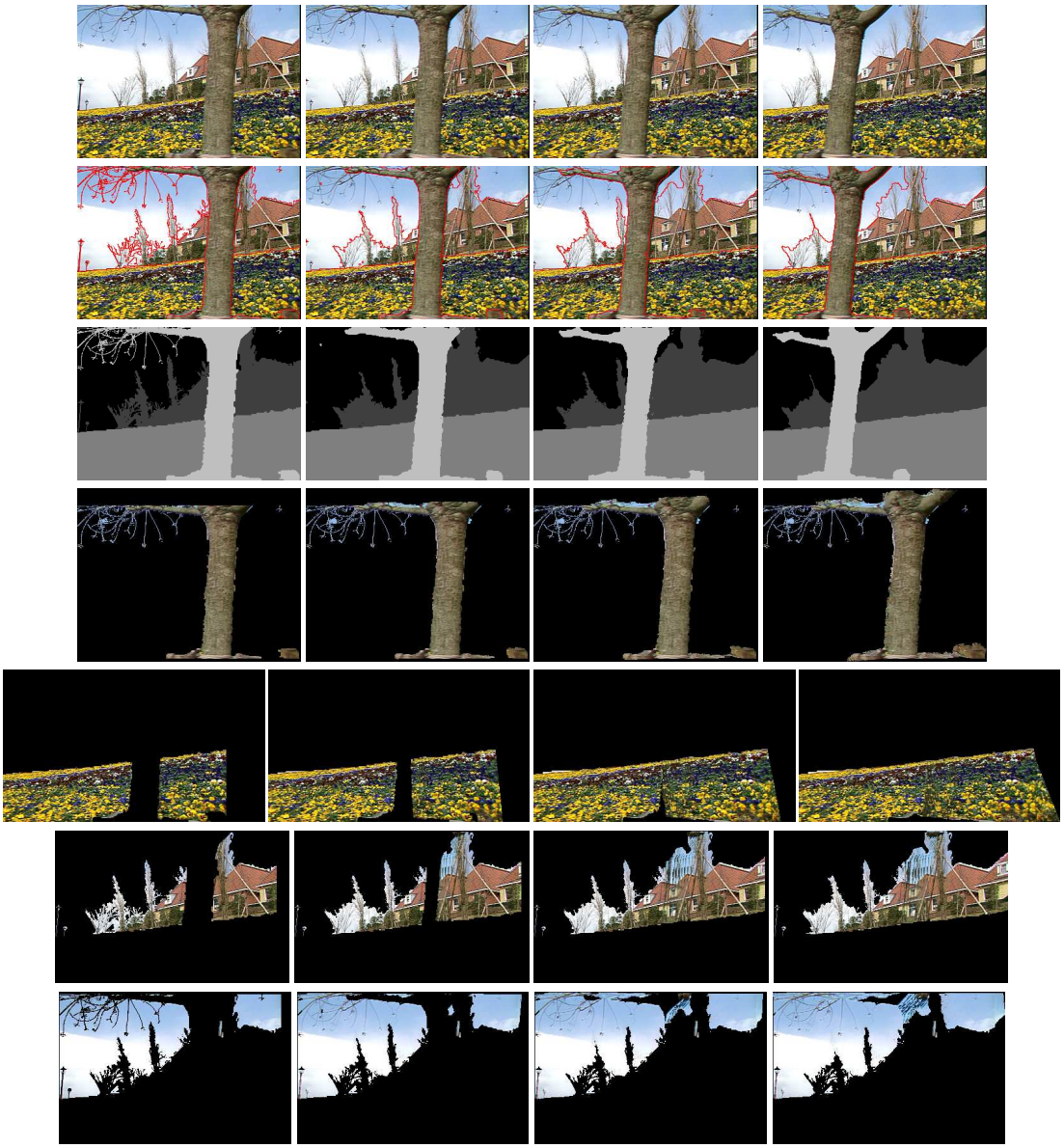


FIGURE 4.9 – Mosaïques de couches générées automatiquement à partir des 20 premières images de la séquence *Flowergarden*. Première ligne, images originales 1, 6, 12 et 20. Deuxième ligne, frontières extraites. Troisième ligne, cartes de segmentation correspondantes. Lignes suivantes, mosaïques $M_{1,k}$ (avec $k = 1, \cdots, 4$) d'instant de référence 1 générées au fil du processus de segmentation. La croissance, au fil du temps, du support des mosaïques est particulièrement visible pour les couches des « fleurs » et des « maisons ». Dans l'ensemble, les mosaïques générées progressivement présentent de bons alignements. Une légère imprécision sur le bord droit du tronc de l'arbre altère les mosaïques des couches « maisons » et « ciel ». On constate que l'extraction des fines branches de l'arbre pose problème, malgré la bonne segmentation de référence fournie par l'opérateur pour l'image 1. À un instant t, un même pixel $\mathbf{p}$ ne peut être copié que dans les mosaïques associées à la couche $f_{\mathbf{p}}$. Mais si à l'instant suivant, son correspondant $\mathbf{p}'$ porte une étiquette différente de $f_{\mathbf{p}}$, alors on retrouvera l'information couleur portée par ces pixels dans les mosaïques de deux couches différentes. C'est ce qui explique que l'on retrouve les branches dans la mosaïque du « ciel ».

Chapitre 5

Résultats expérimentaux

Dans ce chapitre nous présentons d'abord les résultats expérimentaux obtenus par nos deux schémas séquentiels de segmentation de séquences d'images au sens du mouvement.

Pour le premier schéma (section 5.1), nous illustrons particulièrement le rôle de la contrainte de rigidité.

Pour le second schéma (section 5.2), en plus des résultats de segmentation, nous montrons les mosaïques de couches générées automatiquement. Comme avec n'importe quelle méthode, il est possible qu'à la fin du processus de segmentation les frontières obtenues ne soient pas exactement celles attendues. Bien souvent, l'opérateur n'a pas d'autre solution que de modifier manuellement toutes les cartes de segmentation une par une ou de tout reprendre à zéro en relançant le processus avec des paramètres différents. Nous proposons une manière simple pour l'opérateur de corriger manuellement les erreurs possibles sur les mosaïques elles-mêmes. Ces corrections sont ensuite propagées à toutes les images de la séquence grâce à une seconde passe de segmentation.

Nous citons quelques applications possibles en section 5.3 et montrons des résultats pour le remplissage de régions nécessaire lors de la suppression d'un objet vidéo.

Les différents modes d'interaction de l'utilisateur avec les outils de segmentation proposés sont évoqués en section 5.4.

Enfin une discussion est engagée à la section 5.5 pour évoquer les limitations de nos algorithmes et donner des solutions envisageables.

5.1 Résultats de segmentation du premier schéma séquentiel basé sur les triplets d'images consécutives

L'algorithme proposé a été testé sur les trois mêmes séquences que celles utilisées dans [Dupont 05, Xiao 05b, Dupont 06b]. Notre algorithme fournit des cartes de segmentation dense sans étiquette supplémentaire pour le bruit, les occultations ou des indéterminations (contrairement à [Xiao 05b, Dupont 06b]).

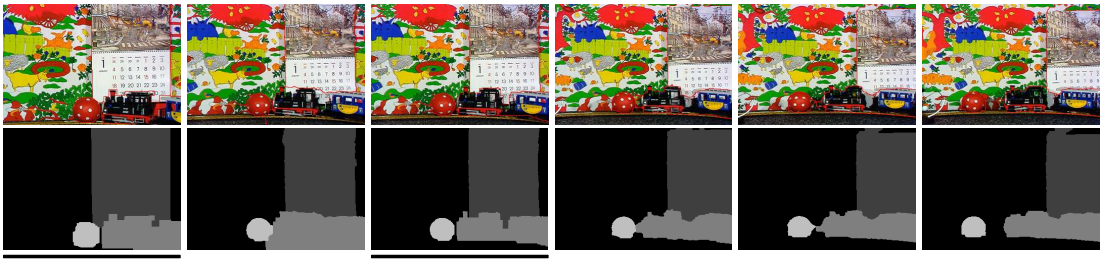


FIGURE 5.1 – Résultats de segmentation sur la séquence *Mobile & Calendar* traitée de l'image 1 vers l'image 99. Seules les images 1, 39, 40, 70, 84, 99 sont montrées. Première ligne, superposition des frontières de mouvement sur les images originales. Deuxième ligne, cartes de segmentation correspondantes. Les cartes de segmentation soulignées (instants 1 et 40) sont celles qui sont fournies au système par l'opérateur. Elles ont été obtenues de manière semi-automatique, les autres sont obtenues automatiquement. $\lambda_1 = 15$, $\lambda_2 = 1$, $\lambda_3 = 10$, $\lambda_4 = 17$ et $\lambda_5 = 35$.

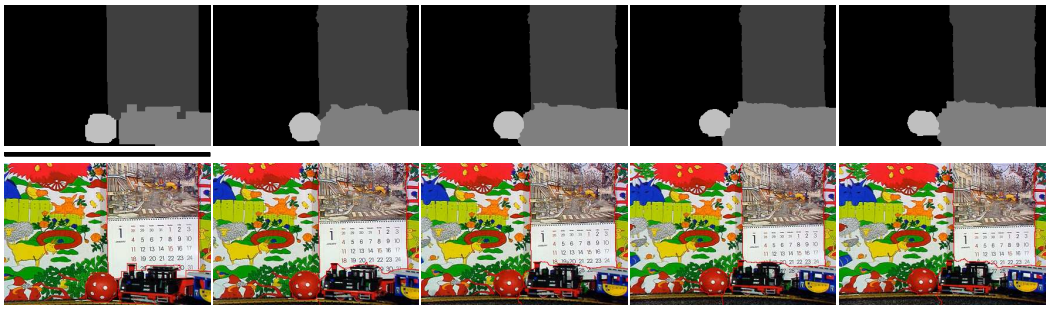
5.1.1 Résultats de segmentation sur la séquence *Mobile & Calendar*

La figure 5.1 montre nos résultats sur la séquence *Mobile & Calendar* traitée de l'image 1 vers l'image 99. Les segmentations obtenues sont correctes même à la fin de la séquence, non traitée habituellement, où la balle et le train ne sont plus en contact. Grâce à notre contrainte de rigidité appliquée à la balle, le support de celle-ci ne s'étend jamais sur les régions du fond qui sont pourtant de couleurs similaires (balle rouge et régions rouges du papier-peint correspondant à une vache). Après l'initialisation, une seule intervention supplémentaire de l'opérateur a été introduite sur l'image 40. Les mélanges de gaussiennes de couleurs sont alors remis à jour. La moquette sombre, invisible à l'instant 1, et qui apparaît ensuite progressivement en bas des images, n'a pas été prise en compte lors de l'estimation des mélanges de gaussiennes initiaux. Ceci explique la mauvaise classification de cette moquette dans la classe « train » jusqu'à l'instant 40 (voir également figure 5.2a). Le choix des instants de référence est laissé à l'appréciation de l'opérateur, ce choix peut avoir une incidence sur la cohérence temporelle de la segmentation (saut brusque entre les segmentations 39 et 40 dans notre cas). Si la réinitialisation est faite trop tôt elle risque de ne pas influencer efficacement les segmentations suivantes.

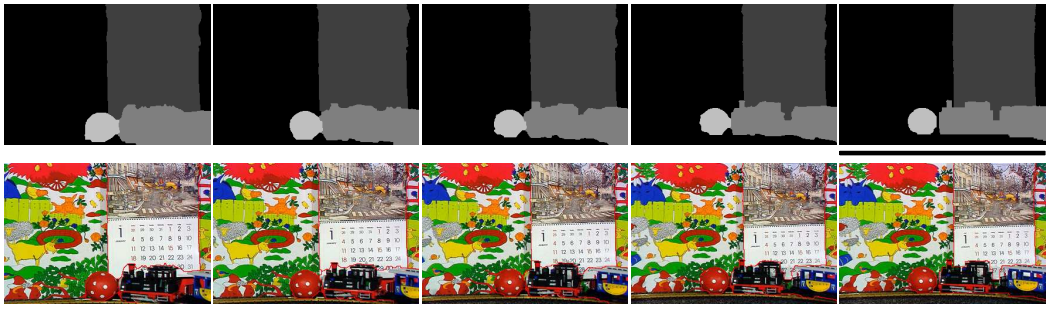
Dans notre cas, une solution efficace pour masquer l'incohérence temporelle, et résoudre en même temps le problème de classification de la moquette avant l'image 40, est de traiter la séquence de façon bidirectionnelle à partir de la seule carte d'initialisation de l'instant 40. On procède alors à la segmentation en deux temps : de l'image 40 vers l'image 1 (sens « arrière », voir figure 5.2b) et de l'image 40 vers l'image 99 (sens « avant »).

Les résultats du traitement à rebours (figure 5.2b) montrent une classification correcte de la moquette sombre en bas des images, contrairement aux résultats du traitement en « avant » (figure 5.2a).

Nous aurions également pu envisager de traiter la séquence dans le sens « avant »

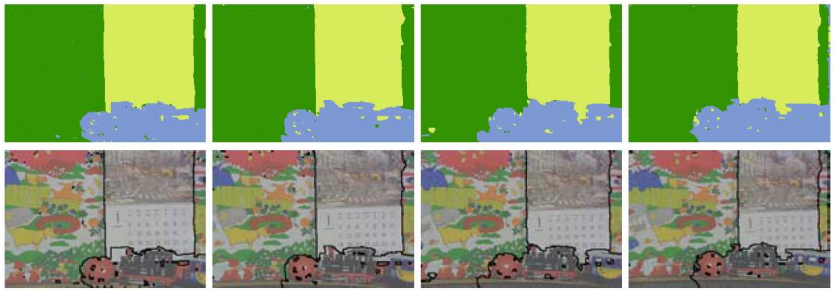


(a) Nos résultats par traitement de l'image 1 vers l'image 40. $\lambda_1 = 15$, $\lambda_2 = 1$, $\lambda_3 = 10$, $\lambda_4 = 17$ et $\lambda_5 = 35$.



(b) Nos résultats par traitement de l'image 40 vers l'image 1. $\lambda_1 = 15$, $\lambda_2 = 1$, $\lambda_3 = 10$, $\lambda_4 = 17$ et $\lambda_5 = 35$.

(c) Résultats de Xiao et Shah copiés de [Xiao 05b]. Les régions d'occultation sont en rouge.



(d) Résultats de Dupont *et al.* copiés de [Dupont 05] (3 couches seulement).

FIGURE 5.2 – Comparaison des résultats de segmentation sur le début de la séquence *Mobile & Calendar*. Pour nos résultats 5.2a et 5.2b, les cartes de segmentation soulignées sont celles qui sont fournies au système par l'opérateur et seules les images 1, 12, 24, 36 et 40 sont montrées. Pour les résultats 5.2c, les images montrées peuvent ne pas correspondre exactement aux mêmes instants. Pour les résultats 5.2d, seules les images 1, 12, 24 et 36 sont montrées.

à partir de l'image 1, d'intervenir sur l'image 40, et de confronter les cartes de segmentation obtenues avant l'instant 40 aux projections en arrière de la nouvelle carte de segmentation de l'image 40. Seules les zones où l'étiquetage du traitement « avant » diffère de l'étiquetage des projections « arrières » auraient alors été remises en question.

Sur la figure 5.2c dupliquée depuis [Xiao 05b], les résultats de Xiao et Shah nous semblent globalement similaires aux nôtres (figure 5.2b). Ils sont peut-être plus précis au niveau des contours supérieurs de la locomotive mais moins précis au niveau des points de contact entre la balle et la locomotive.

Sur la figure 5.2d dupliquée depuis [Dupont 05], le train et la balle n'ont pas été séparés, ce qui n'est pas correct en terme de mouvement. De plus, la moquette sombre est étiquetée comme région appartenant à la classe « balle+train », et non à la classe d'arrière-plan.

5.1.2 Résultats de segmentation sur la séquence *Flowergarden*

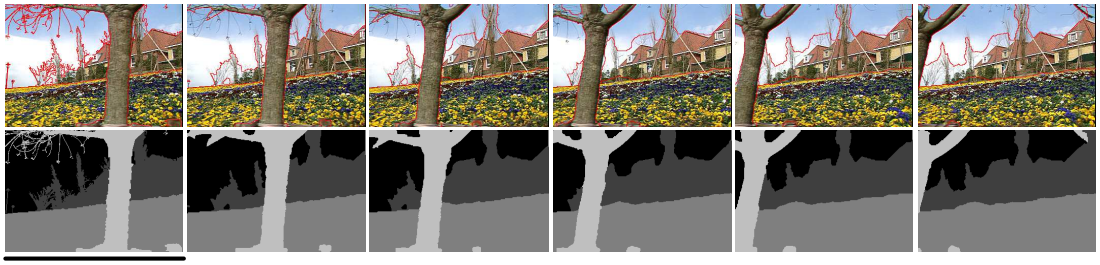
Pour la séquence *Flowergarden*, le nombre de couches à extraire doit être déterminé en fonction de l'application visée. S'il paraît clair que l'arbre doit constituer une couche à lui seul, une couche unique pour l'arrière-plan (ciel + maisons + fleurs) ne convient pas si, après la segmentation, on souhaite apporter un traitement particulier seulement sur les maisons). Dans la suite, nous nous proposons d'extraire 4 couches : le ciel, les maisons, le parterre de fleurs et l'arbre au premier plan.

Comme nous l'avons déjà évoqué, l'obtention de contours précis pour les branches et les buissons nécessiterait un traitement particulier tel une méthode de matting [Xiao 05a]. Les structures trop fines sont en effet supprimées par le terme de régularisation spatiale. Même si l'opérateur fournit une carte très précise de segmentation pour la première image, il ne semble pas possible de maintenir une telle précision tout au long de la séquence. Par ailleurs, dans le plan image, le mouvement des branches fines de l'arbre n'est pas similaire au mouvement du tronc, mais se rapproche plutôt du mouvement du parterre de fleurs. Ceci est dû au fait que les branches et le tronc n'appartiennent pas au même plan 3D.

La figure 5.3a montre les résultats que nous avons obtenus sur la séquence *Flowergarden*. L'opérateur a uniquement fourni la carte de segmentation de la première image.

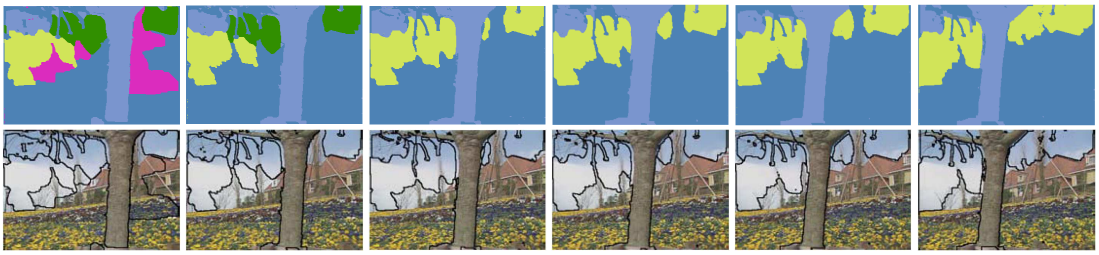
Les résultats de Xiao et Shah (figure 5.3b dupliquée depuis [Xiao 05b]) font apparaître seulement 3 couches. Comparés à nos résultats, les couches « ciel » et « maisons » sont regroupées.

Les résultats de Dupont *et al.* (figure 5.3c dupliquée depuis [Dupont 05]) montrent qu'une initialisation automatique peut mener à un nombre de couches inattendu. Après initialisation et stabilisation du nombre de couches, les trois couches « ciel », « maisons + fleurs » et « tronc » sont extraites avec précision. Quelques branches fines de l'arbre sont même distinguées du ciel, mais classées comme appartenant à la couche « maisons + fleurs » ce qui est intéressant au regard du mouvement, mais incorrect au regard de la sémantique.



(a) Nos résultats. Seules les images 1, 10, 20, 30, 40, 49 sont montrées. Seule la première carte de segmentation (soulignée) est fournie au système par l'opérateur. Elle a été obtenue de manière semi-automatique et l'opérateur a indiqué qu'il souhaitait extraire 4 couches. Les autres cartes de segmentation sont obtenues automatiquement. $\lambda_1 = 15$, $\lambda_2 = 1$, $\lambda_3 = 10$, $\lambda_4 = 0$ et $\lambda_5 = 20$.

(b) Résultats de Xiao et Shah copiés de [Xiao 05b]. Les régions d'occultation sont en rouge. Trois couches ont été détectées.



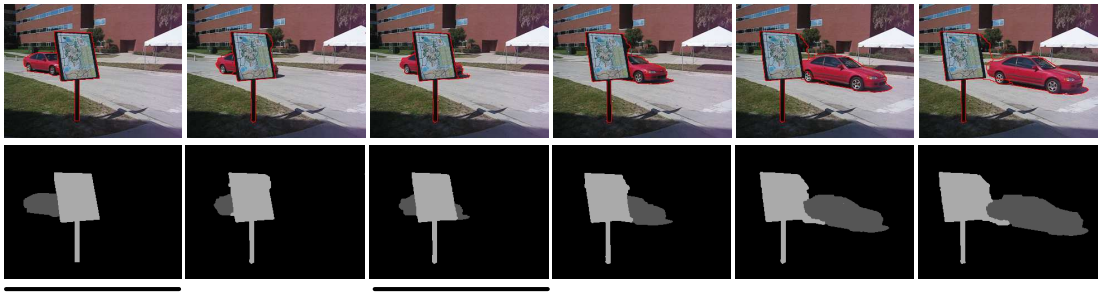
(c) Résultats de Dupont *et al.* copiés de [Dupont 05]. L'initialisation automatique produit une sur-segmentation (5 couches) avant que le nombre de couches ne finisse par se stabiliser à 3. Seules les images 1, 4, 7, 10, 13, 16 sont montrées.

FIGURE 5.3 – Comparaison des résultats de segmentation sur la séquence *Flowergarden*. Les images d'une même colonne ne correspondent pas aux mêmes instants.

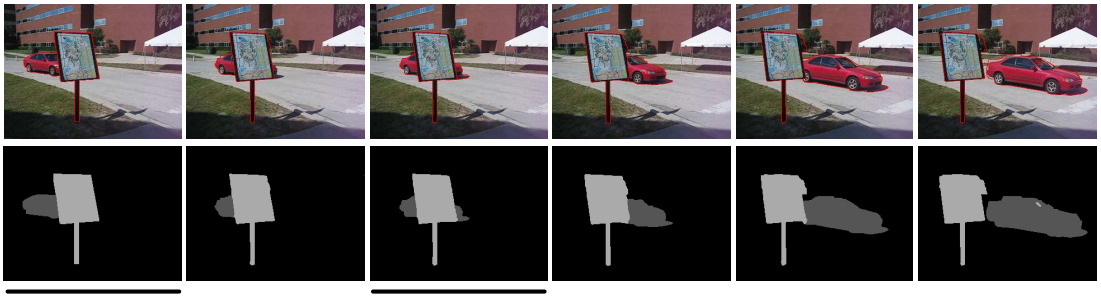
5.1.3 Résultats de segmentation sur la séquence *Carmap*

La séquence *Carmap* représente une voiture rouge circulant dans une rue et passant derrière un panneau de localisation. Trois couches sont généralement extraites. Elles correspondent respectivement au panneau, à la voiture et à l'arrière-plan. La principale difficulté à surmonter est l'importante occultation de la voiture derrière le panneau. Le capot n'est pas visible en début de séquence. Par ailleurs, les roues et l'ombre de la voiture sont aussi sombres que les bords du panneau. De plus, les mouvements de l'arrière-plan et du panneau sont très proches, et même identiques au niveau du pied du panneau.

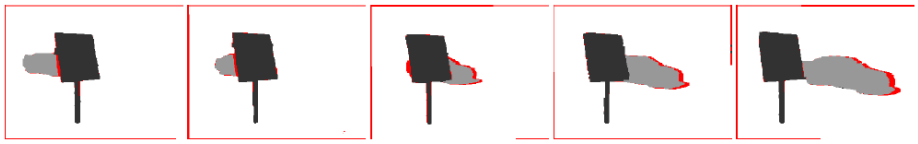
Les figures 5.4a et 5.4b montrent les résultats obtenus avec notre algorithme lorsque l'opérateur a fourni les cartes de segmentation des images 1 et 11, avec et sans contrainte



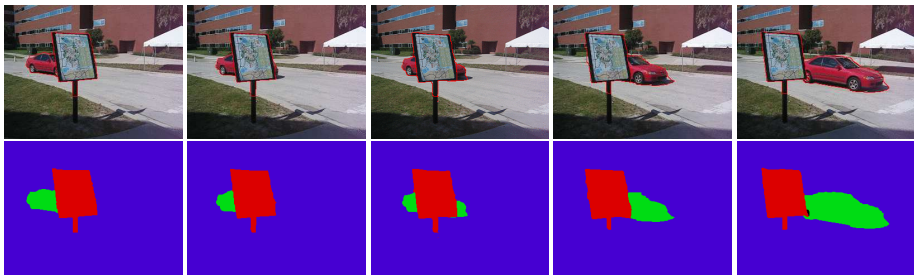
(a) Nos résultats sans contrainte de rigidité sur le panneau. $\lambda_1 = 17$, $\lambda_2 = 1$, $\lambda_3 = 12$, $\lambda_4 = 0$ et $\lambda_5 = 35$.



(b) Nos résultats avec contrainte de rigidité sur le panneau. $\lambda_1 = 17$, $\lambda_2 = 1$, $\lambda_3 = 12$, $\lambda_4 = 17$ et $\lambda_5 = 35$.



(c) Résultats de Xiao et Shah copiés de [Xiao 05b]. Les régions d'occultation sont en rouge.



(d) Résultats de Dupont *et al.* copiés de [Dupont 06a]. Les pixels en noir (roue arrière sur l'image de droite) sont classés comme aberrants ou sujets au bruit.

FIGURE 5.4 – Résultats de segmentation sur la séquence *Carmap*. Pour nos résultats 5.4a et 5.4b, les cartes de segmentation soulignées sont obtenues de manière semi-automatique par intervention de l'opérateur et seules les images 1, 10, 11, 20, 30, 34 sont montrées. Pour les résultats 5.4c, les images montrées peuvent ne pas correspondre exactement aux mêmes instants. Pour les résultats 5.4d, seules les images 4, 10, 11, 20, 30 sont montrées.

séquence	taille des images	taille moyenne du graphe	n	CPU/image
Mobile & Calendar	352×240	11647 pixels	4	~ 6 s.
Flowergarden	352×240	22886 pixels	4	~ 6 s.
Carmap	320×240	9192 pixels	3	~ 3 s.

TABLE 5.1 – Temps de calcul moyen, tous traitements inclus (n est le nombre de couches).

de rigidité appliquée à la couche du panneau. Les autres algorithmes (figures 5.4c et 5.4d) semblent rencontrer les mêmes difficultés à classer dans la classe « voiture » l'avant de la voiture dès qu'il apparaît à droite du panneau.

Dans notre expérimentation, l'opérateur est donc intervenu à l'instant 11 pour indiquer la réapparition de la voiture, sans plus attendre. L'apport de la contrainte de rigidité est évident : cette contrainte évite les erreurs de classification dans les zones uniformes sombres (roues, ombre de la voiture et bords du panneau) où ni le critère d'apparence, ni le critère de mouvement ne sont suffisamment discriminants. Notons que le pied du panneau est correctement segmenté, ce qui n'est pas le cas dans les résultats de Dupont *et al.*

5.1.4 Temps de calcul

Toutes nos expérimentations ont été menées sur un Pentium IV à 3,6GHz, 3GB RAM. Tous traitements inclus (estimation de mouvement, calcul des mélanges de gaussiennes, construction du graphe, segmentation, prédiction ...), le temps de calcul nécessaire à la segmentation d'une image dépend principalement du nombre de couches qui a lui-même une influence sur la longueur des frontières et donc sur la taille du graphe (voir tableau 5.1).

Pour [Xiao 05b], le temps moyen nécessaire à la segmentation d'une image est légèrement inférieur à 30 secondes sur un Pentium IV à 2GHz.

Pour [Dupont 06a], « une dizaine de minutes sont nécessaires pour segmenter une trentaine d'images » sur un Pentium IV à 2GHz, soit environ 20 secondes par image pour la méthode la plus avancée des travaux du Dupont *et al.* qui permet à la fois l'extraction des couches visibles et des couches cachées. Mais la méthode nécessite de considérer toute la séquence simultanément. Le stockage en mémoire de la totalité du graphe construit à partir du volume vidéo non restreint peut rapidement poser problème.

5.2 Résultats de segmentation du second schéma séquentiel basé sur les mosaïques

5.2.1 Validation de notre critère de mouvement basé sur une différence d'intensités entre image et mosaïque

Nous avons dans un premier temps testé notre second algorithme sur une séquence de vidéo surveillance issue de la base de données *PETS 2001* de l'Université de Reading. La séquence est relativement simple et nous a servi à valider notre critère de mouvement basé sur une différence d'intensités entre image et mosaïque. La caméra est fixe. La scène représente un parking avec un seul véhicule en mouvement : une camionnette blanche. Nous n'avons traité qu'un intervalle de 60 images car le mouvement de la camionnette est plus complexe qu'un mouvement planaire dans le reste de la séquence.

La camionnette est entièrement visible sur chacune des images de l'intervalle considéré, et la région du fond cachée par la camionnette dans la première image est totalement visible dans la dernière image (et inversement). Les mosaïques ont donc été facilement générées avant que nous lancions le processus de segmentation et n'ont plus été modifiées ensuite. Ce cas favorable nous a permis de nous concentrer sur l'extraction des couches de mouvement et non sur la génération des mosaïques. Nous avons ainsi validé notre critère de mouvement faisant appel aux mosaïques en imposant des coefficients de pondération nuls aux autres termes unaires (termes d'apparence et de contrainte temporelle). Les mosaïques de référence utilisées sont présentées dans la figure 5.5.

FIGURE 5.5 – Mosaïques de référence pour la séquence *PETS 2001*. Ici, les mosaïques sont obtenues manuellement afin de valider le principe de segmentation basé sur le terme de mouvement d'image à mosaïque de référence (équation (4.4)). Première ligne, mosaïques du fond obtenues facilement à partir des deux images de référence, sachant que le mouvement est nul. Deuxième ligne, mosaïques de la camionnette obtenues directement à partir des cartes de segmentation fournies par l'opérateur pour les deux images de référence 701 et 760.

L'algorithme a permis la bonne extraction des deux couches (figure 5.6). Les seules ambiguïtés sont rencontrées lorsque la camionnette blanche passe devant la voiture blanche stationnée. Notre critère de mouvement faisant appel aux mosaïques se montre efficace sauf dans ces régions « blanc sur blanc ».

Notons qu'à travers le pare-brise, les pixels sont considérés comme appartenant à l'arrière-plan, ce qui peut être intéressant pour certaines applications. En revanche, si l'on cherche à supprimer la camionnette blanche, il est plus simple que ces mêmes pixels soient classés avec l'objet d'avant-plan. Ceci peut être obtenu en ajoutant la contrainte temporelle (figure 5.7).

FIGURE 5.6 – Résultats de segmentation sur la séquence *PETS 2001* sans contrainte temporelle. Les frontières de mouvement sont superposées sur les images originales. Seules les images 702, 710, 725, 743, 751 et 758 sont montrées. La transparence du pare-brise de la camionnette explique les inconsistances temporelles de la segmentation. $\lambda_1 = 17$, $\lambda_2 = 0$, $\lambda_3 = 0$ et $\lambda_4 = 30$.

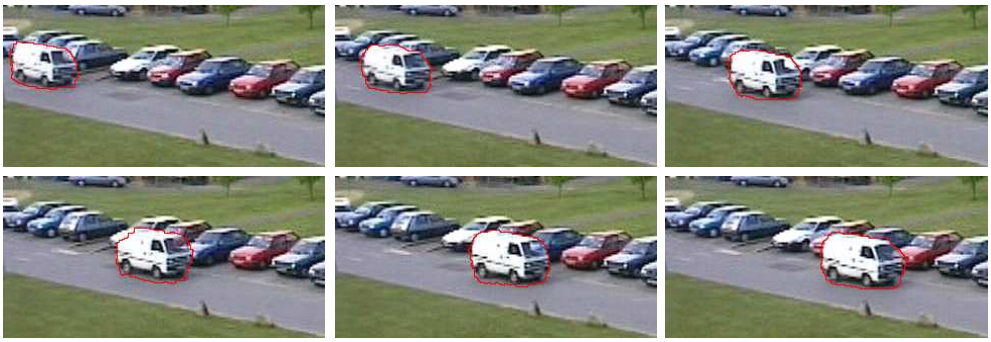


FIGURE 5.7 – Résultats de segmentation sur la séquence *PETS 2001* avec contrainte temporelle. Les frontières de mouvement sont superposées sur les images originales. Seules les images 702, 710, 725, 743, 751 et 758 sont montrées. Le pare-brise de la camionnette est classé comme appartenant à l'objet d'avant-plan tout au long de l'intervalle considéré. $\lambda_1 = 17$, $\lambda_2 = 0$, $\lambda_3 = 15$ et $\lambda_4 = 30$.

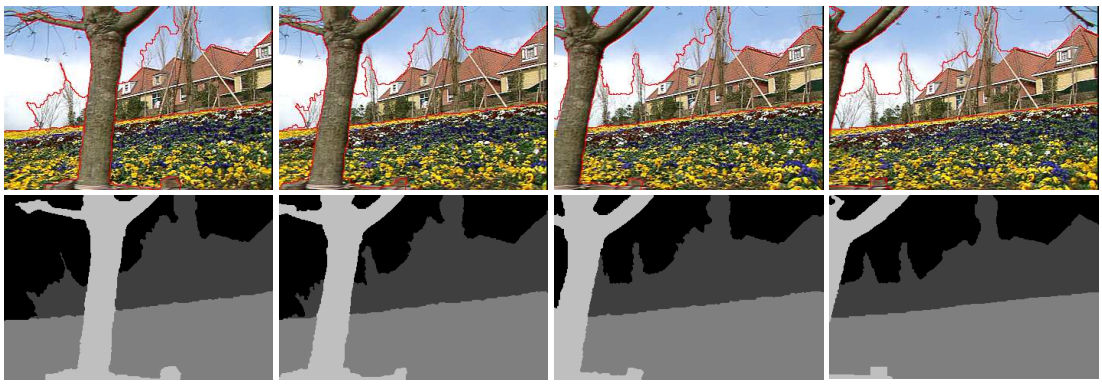


FIGURE 5.8 – Résultats de segmentation sur l'intervalle $[20, 50]$ de la séquence *Flowergarden* avec génération automatique des mosaïques de couche. $\lambda_1 = 17$, $\lambda_2 = 1$, $\lambda_3 = 3$ et $\lambda_4 = 11$. Seules les images 20, 30, 40 et 50 sont montrées.

5.2.2 Résultats de segmentation sur la séquence *Flowergarden*

Nous avons testé notre second algorithme sur la séquence *Flowergarden*. La séquence a été divisée en deux intervalles : images 1 à 20 et images 20 à 50. Les résultats de segmentation du premier intervalle ainsi que les mosaïques générées automatiquement à partir des 20 images et pour l'instant de référence 1 ont été montrés dans le chapitre précédent à la figure 4.9. Les résultats de segmentation du second intervalle sont montrés à la figure 5.8. Pour l'ensemble de la séquence, nos résultats sont aussi bons que les meilleurs résultats de la littérature. L'extraction précise des fines branches de l'arbre nécessiterait le recours à une méthode de matting.

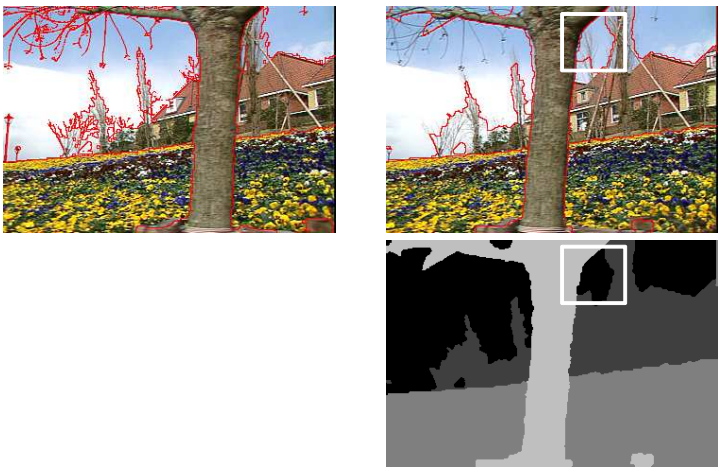
La figure 5.9 permet une comparaison de la précision des frontières de mouvement obtenues par le premier et le second schémas sur l'image 10. La carte de segmentation fournie pour l'instant 1 est la même pour les deux schémas. Les différences résident donc dans l'expression du terme d'énergie lié au mouvement et dans la prise en compte d'une deuxième carte de segmentation de référence dans le second schéma.

5.2.3 Résultats de segmentation sur la séquence *Carmap*

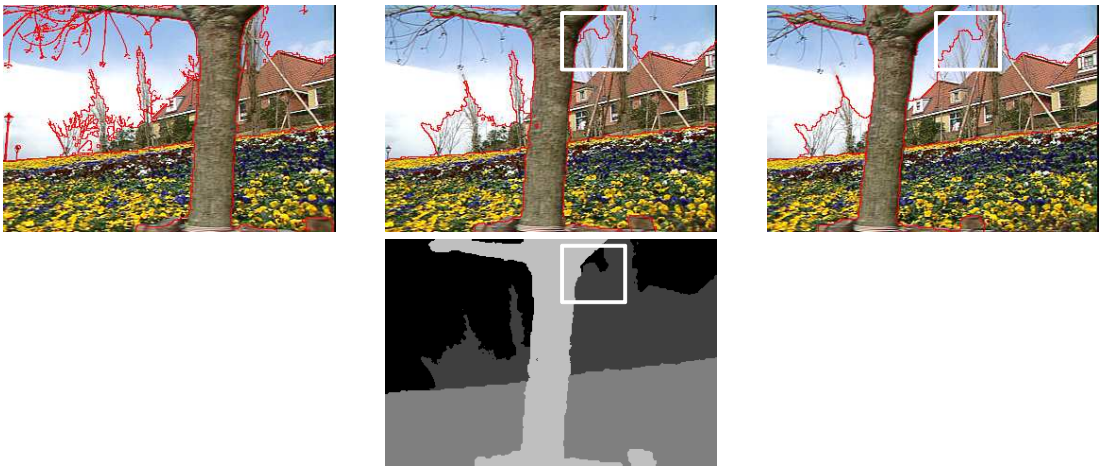
Pour la séquence *Carmap*, la légère rotation 3D en dehors du plan image rend difficile le traitement de la séquence en un seul intervalle. Nous avons donc divisé la séquence en trois intervalles (images 1 à 9, images 9 à 17 et images 17 à 35) de manière à rendre le problème plus simple. L'instant 17 a été choisi comme instant de référence car il correspond au milieu de la séquence. L'instant 9 a quant à lui été choisi après observation de la séquence. Il correspond à l'instant où l'avant de la voiture apparaît à droite du panneau.

La figure 5.10 montre les résultats de segmentation pour les deux premiers intervalles de la séquence *Carmap*.

Pour le troisième intervalle (le plus long des trois), l'approximation du mouvement



(a) Premier schéma basé sur les triplets d'images consécutives : (à gauche) segmentation semi-automatique de l'image 1 fournie par l'opérateur, et (au milieu) segmentation automatique de l'image 10.



(b) Second schéma basé sur les mosaïques de couches et les estimations de mouvement entre images éloignées : (à gauche) segmentation semi-automatique de l'image 1 fournie par l'opérateur (exactement la même que pour le premier schéma), (au milieu) segmentation automatique de l'image 10 cohérente avec la segmentation semi-automatique de l'image 20 fournie par l'opérateur (à droite).

FIGURE 5.9 – Comparaison des résultats de nos deux schémas sur la segmentation de l'image 10 de la séquence *Flowergarden*. En termes de méthodes, et en dehors de la différence d'expression du terme d'énergie lié au mouvement, la différence entre les deux schémas se situe dans l'utilisation d'une référence supplémentaire dans le second schéma, éloignée de l'instant courant et postérieure à cet instant courant. En termes de résultats, les différences majeures de segmentation sont contenues dans le carré blanc : cette région de l'image 10 est occultée dans l'image 1 mais est visible dans l'image 20. Pour le second schéma, la prise en compte d'informations de référence provenant du futur conduit à la cohérence entre la segmentation à l'image 10 et celle indiquée par l'opérateur pour l'image 20. L'arbrisseau visible au dessus du toit appartient à la couche « maisons ».

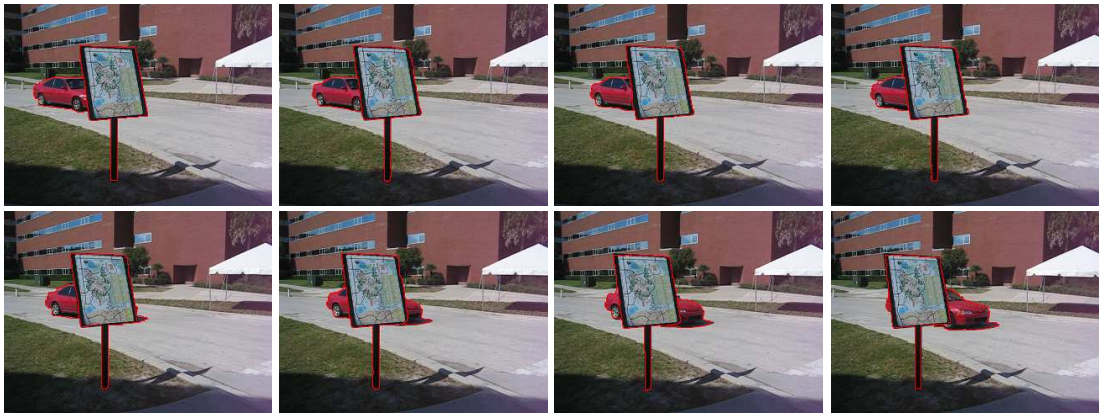


FIGURE 5.10 – Résultats de segmentation sur les deux premiers intervalles de la séquence *Carmap*. Seules les images 1, 3, 5, 7, 9, 12, 14 et 17 sont montrées. $\lambda_1 = 14$, $\lambda_2 = 1$, $\lambda_3 = 10$ et $\lambda_4 = 30$.

de la voiture par un modèle affine n'est pas satisfaisante, menant à des erreurs de classification entre les couches « voiture » et « arrière-plan » et à de mauvais alignements au sein des mosaïques de la couche « voiture » (figure 5.11). Ceci était prévisible à cause de la rotation 3D en dehors du plan, de l'importance de la distance temporelle entre les images mises en correspondance, relativement à l'amplitude du mouvement apparent dans la scène. Plutôt que de diviser à nouveau cet intervalle en deux intervalles plus courts et relancer totalement le processus de segmentation, nous avons légèrement corrigé manuellement les mosaïques générées pour l'arrière-plan avant de lancer une seconde passe de segmentation prenant en entrée ces mosaïques corrigées.

La figure 5.12 montre comment en utilisant un outil standard d'édition d'image, en l'occurrence la « gomme », l'opérateur a supprimé les régions mal classées d'une mosaïque de la couche « arrière-plan ». Une seconde passe de segmentation a été lancée avec en entrée les mosaïques obtenues à la fin de la première passe et cette mosaïque corrigée. L'intérêt de travailler avec les mosaïques de couche est ici pleinement démontré. Les corrections appliquées par l'opérateur dans une seule mosaïque se comportent comme des germes qui sont propagés à toutes les images de l'intervalle durant la seconde passe. Les résultats sont présentés à la figure 5.13. Après une intervention modeste de l'opérateur, les résultats obtenus sont aussi bons, voire meilleurs, que ceux des méthodes simultanées [Xiao 05b, Dupont 06c], plus coûteuses.

5.3 Applications

Nous présentons ici deux exemples de manipulation d'objet vidéo montrant l'utilité des mosaïques de couches de mouvement.

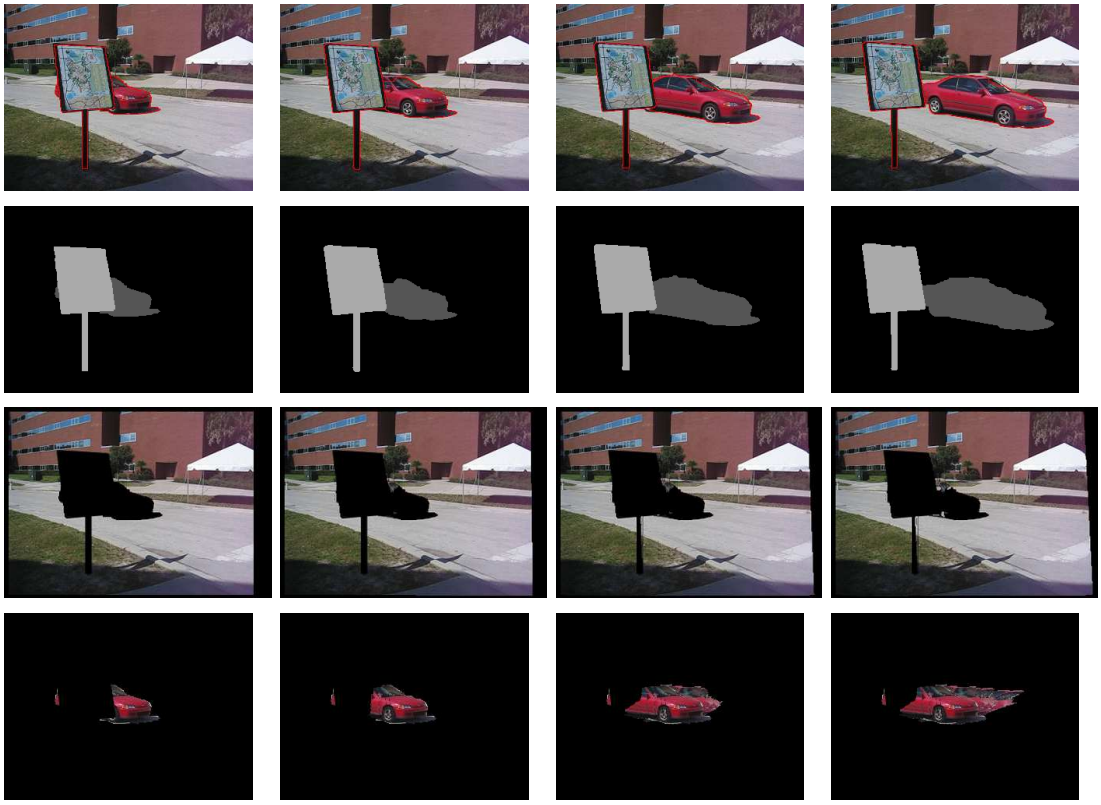


FIGURE 5.11 – Résultats de segmentation sur l'intervalle $[17, 35]$ de la séquence *Carmap* à la fin de la première passe. Seules les images 17, 22, 28 et 32 sont montrées. Notons les erreurs de classification entre les classes « voiture » et « arrière-plan » notamment au niveau du pare-brise. Deux dernières lignes, états intermédiaires des mosaïques d'instant de référence 17 pour l'arrière-plan et la voiture, automatiquement générées au fil du processus de segmentation. L'approximation du mouvement de la voiture par un modèle affine entre images temporellement éloignées n'est pas satisfaisante. $\lambda_1 = 14$, $\lambda_2 = 1$, $\lambda_3 = 10$ et $\lambda_4 = 30$.

FIGURE 5.12 – Mosaïque de la couche « arrière-plan » pour l'instant de référence 17. De gauche à droite : mosaïque obtenue automatiquement à la fin de la première passe, détail du carré blanc qui inclut par erreur certaines zones de la voiture, suppression de ces erreurs par une modeste intervention de l'opérateur. En noir, les trous restants dans la mosaïque, c'est-à-dire les régions qui selon notre système n'ont jamais été visibles durant l'intervalle considéré.

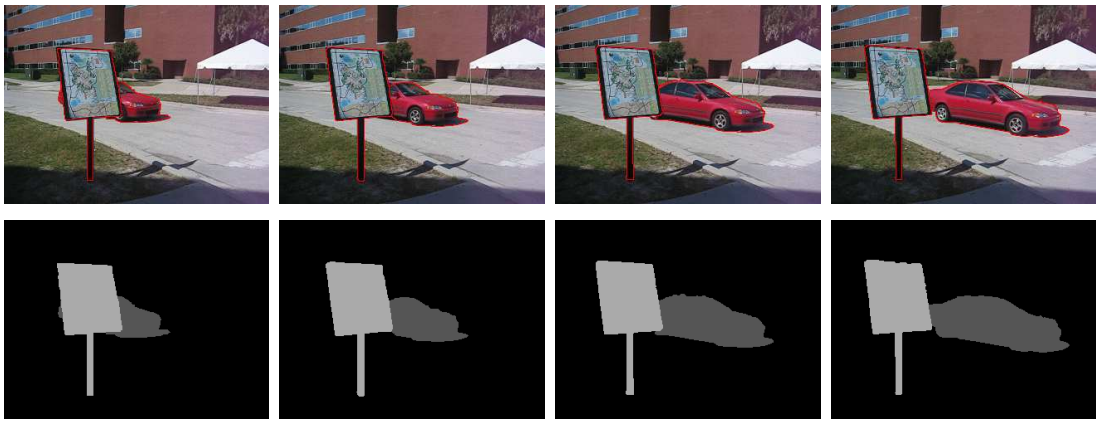


FIGURE 5.13 – Résultats de segmentation sur l'intervalle $[17, 35]$ de la séquence *Carmap* après une légère correction de la mosaïque par l'opérateur et après une seconde passe de segmentation. Seules les images 17, 22, 28 et 32 sont montrées. Les erreurs précédentes de classification ont disparu. $\lambda_1 = 14$, $\lambda_2 = 1$, $\lambda_3 = 10$ et $\lambda_4 = 30$.

(a) Mosaïque de l'arrière-plan rognée aux dimensions de l'image de référence 1. À gauche, le trou blanc contient les régions qui restent invisibles tout au long de la séquence. À droite, remplissage du trou par une méthode interactive basée sur la copie de blocs visibles.

(b) Suppression de la voiture. Première ligne, séquence originale. Deuxième ligne, séquence modifiée.

FIGURE 5.14 – Suppression de la voiture dans la séquence *Carmap*. Après segmentation de la séquence au sens du mouvement, la voiture est supprimée et les régions manquantes des couches plus éloignées (ici l'arrière-plan uniquement) sont reconstruites en utilisant les mosaïques et les modèles de mouvement estimés pendant le processus de segmentation. Seules les images 1, 15, 25 et 35 sont montrées.

5.3.1 Suppression d'un objet vidéo

Pour procéder à la suppression d'un objet vidéo, nous appliquons la méthode proposée par Zhang *et al.* [Zhang 05]. La méthode consiste à segmenter la séquence au sens du mouvement, générer les mosaïques des couches situées derrière l'objet à supprimer, et remplir les régions correspondant à l'objet supprimé par projection de ces mosaïques et respect de l'ordre de profondeur. La génération de la mosaïque d'une couche se fait ici en deux temps : d'abord un remplissage temporel qui permet le regroupement de tous les éléments de la couche considérée visibles dans la séquence (ce remplissage est fait automatiquement par notre second schéma de segmentation), puis un remplissage spatial qui permet le remplissage des régions de la couche qui restent toujours invisibles. Pour ce remplissage spatial, nous avons développé une méthode semi-automatique inspirée de la méthode de remplissage de régions par copie de blocs visibles [Criminisi 04]. Cette

méthode semi-automatique que nous ne détaillerons pas dans ce document s'apparente à [Sun 05]. Un exemple de suppression d'objet vidéo est donné à la figure 5.14.

5.3.2 Édition d'une mosaïque et propagation à toute la séquence

Nous proposons d'éditer une mosaïque pour, par exemple, modifier l'apparence de régions déjà existantes ou rajouter de nouveaux éléments devant partager le même mouvement et le même ordre de profondeur que la couche considérée. L'intérêt d'éditer directement la mosaïque est qu'elle fournit un aperçu complet de la couche contrairement à une image de référence. Les modifications peuvent donc porter sur la couche complète, ce qui est impossible en éditant une image de référence. Les modifications apportées sur la mosaïque peuvent alors être propagées sur toute la séquence avec respect de l'ordre de profondeur des couches en utilisant les modèles de mouvement estimés pendant le processus de segmentation.

Les figures 5.15 et 5.16 montrent deux exemples simples : l'ajout d'un drapeau tricolore sur la camionnette blanche de la séquence *PETS 2001* et l'insertion d'un ours en peluche sur la séquence *Rotation*.

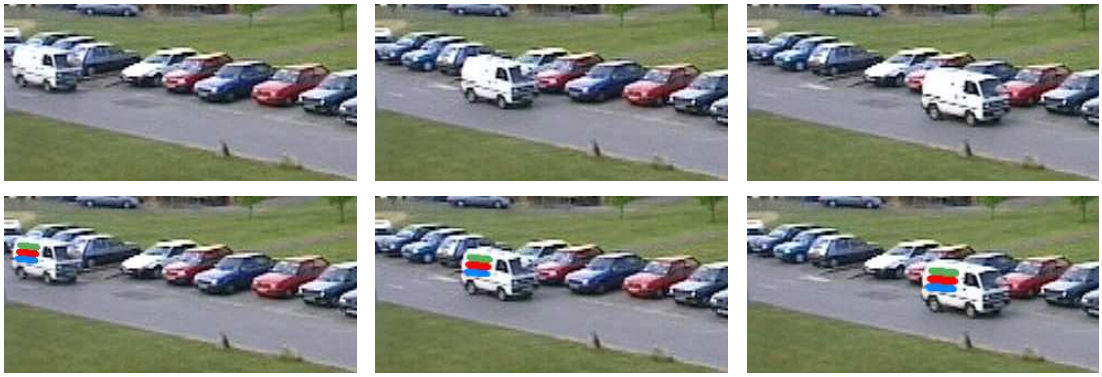


FIGURE 5.15 – Marquage de la camionnette blanche. Première ligne, séquence originale *PETS 2001*. Deuxième ligne, séquence modifiée : propagation automatique à toute la séquence du marquage sur la camionnette blanche. Seules les images 701, 730 et 760 sont montrées.

Rav-Acha *et al.* [Rav-Acha 08] ont récemment proposé une nouvelle représentation « unwrap mosaic » d'un objet 3D déformable. Cette représentation combine une carte de texture de l'objet, une séquence de masques binaires modélisant les occultations et des modèles de projection 2D-2D de la carte de texture vers les images de la séquence. De nombreuses opérations d'édition peuvent être effectuées sur la carte de texture et propagées automatiquement à toute la séquence.

Les problèmes liés aux variations d'apparence de la couche au fil du temps, en particulier le problème du changement des conditions d'illumination, restent à résoudre. La proposition de solutions à ces problèmes peut être l'objet de travaux futurs.

FIGURE 5.16 – Insertion d'un ours en peluche sur le tourniquet. Première ligne, séquence originale *Rotation*. Deuxième ligne, ours à insérer aux côtés de la fillette et séquence modifiée automatiquement après insertion de l'ours sur la mosaïque du tourniquet. Seules les images 459, 464 et 474 sont montrées.

5.4 Aspects semi-automatiques

Dans cette section nous commençons par détailler la tâche de segmentation interactive d'une image, puis nous listons les différents types d'interactions possibles déjà évoqués de manière à les regrouper au sein d'un même paragraphe.

5.4.1 Segmentation interactive d'une image

Pour notre premier schéma, la tâche de segmentation interactive d'une image est requise pour la première image de la séquence à l'initialisation du système. On peut y faire appel à nouveau pour une image ultérieure si une réinitialisation est nécessaire. Pour notre second schéma, cette tâche est requise pour chaque image correspondant à un instant de référence. C'est une étape déterminante puisque c'est durant celle-ci que l'opérateur indique le nombre de couches à extraire, ainsi que leur ordre relatif de profondeur.

Nous préférons nous baser sur les indications fiables de l'opérateur plutôt que sur une méthode automatique car dans certains cas le nombre de couches à extraire est clairement subjectif. Le nombre de couches dépend en fait de l'application finale visée et peut varier facilement. Doit-on considérer par exemple que des roues suivent le même mouvement que la voiture à laquelle elles appartiennent ? Puisque dans nos travaux nous laissons la possibilité à l'opérateur d'intervenir, il nous paraît judicieux de lui laisser le soin d'indiquer le nombre de couches à extraire car lui seul sait à quelles fins est menée la segmentation et quel type de résultats est attendu.

Dans les deux schémas, le type d'interaction est le même. Pour l'image considérée, l'opérateur fournit de larges graines pour marquer grossièrement chacune des couches. Ces graines sont généralement des régions polygonales mais dans le cas de structures fines à extraire, elles peuvent être réduites à de simples coups de crayons, voire de simples points. Dès cette étape, l'opérateur différencie les graines des différentes couches

en les associant à un indice correspondant à leur ordre relatif de profondeur. On choisit la convention selon laquelle l'indice le plus faible correspond à la couche la plus éloignée et l'indice le plus fort correspond à la couche la plus proche. Cet indice varie donc de 1 à n où n est le nombre de couches différentes, fixé par l'opérateur.

Dans le cas d'une réinitialisation du premier schéma et dans le cas du second schéma, les correspondances de couche à couche entre deux instants différents sont établies tout simplement en conservant les mêmes indices, c'est-à-dire que la cohérence temporelle des indices doit être maintenue. Si une couche différente de celle du premier plan disparaît totalement, le premier plan conserve l'indice n, même si seulement $n-1$ couches sont présentes à cet instant dans l'image considérée.

Les graines sont utilisées comme supports initiaux des couches afin de calculer les mélanges de gaussiennes de couleurs à partir de l'image considérée. Si l'on dispose de l'image précédente, les graines sont également utilisées pour estimer un modèle affine de mouvement « arrière » pour chaque couche. Il en va de même pour l'estimation de modèles « avant » si l'on dispose de l'image suivante.

Notre méthode de segmentation interactive d'une image s'apparente alors à une variante des méthodes semi-automatiques de segmentation d'une image [Boykov 01b, Rother 04]. Dans notre cas, nous sommes face à un problème multi-étiquettes. Par ailleurs nous disposons non pas d'une image seule mais d'une séquence d'images. Nous pouvons donc profiter du terme d'énergie lié au mouvement, que nous intégrons dans la fonction à minimiser pour la segmentation interactive. Ce terme permet de résoudre certaines ambiguïtés dues à des similarités de couleurs entre objets, sans que l'opérateur n'ait à introduire de graines supplémentaires.

La fonction d'énergie à minimiser lors de cette segmentation interactive est l'équivalent de la fonction d'énergie globale (4.15) du premier schéma privée de la contrainte de rigidité et de la contrainte temporelle puisqu'ici nous ne disposons pas de la carte de segmentation de l'image précédente. On a alors, en gardant les notations du chapitre 4 :

$$E(f) = \lambda_1 \underbrace{\sum_{\mathbf{p}\in\mathcal{P}} D_{1\mathbf{p}}(f_\mathbf{p})}_{E_{\text{mouvement}}(f)} + \lambda_2 \underbrace{\sum_{\mathbf{p}\in\mathcal{P}} C_\mathbf{p}(f_\mathbf{p})}_{E_{\text{couleur}}(f)} + \lambda_3 \underbrace{\sum_{(\mathbf{p},\mathbf{q})\in\mathcal{N}} V_{\mathbf{p},\mathbf{q}}(f_\mathbf{p}, f_\mathbf{q})}_{E_{\text{lissage}}(f)} \quad . \tag{5.1}$$

Des exemples de graines fournies pour la segmentation interactive d'une image sont donnés aux figures 5.17 et 5.18.

FIGURE 5.17 – Segmentation interactive d'une image seule. Cas classique de segmentation binaire d'une image seule. $\lambda_1 = 0$, $\lambda_2 = 1$ et $\lambda_3 = 10$. De gauche à droite : graines fournies par l'opérateur (les régions non marquées restent incertaines, à classer) ; carte de segmentation finale après classification automatique des régions incertaines ; superposition des frontières sur l'image originale.

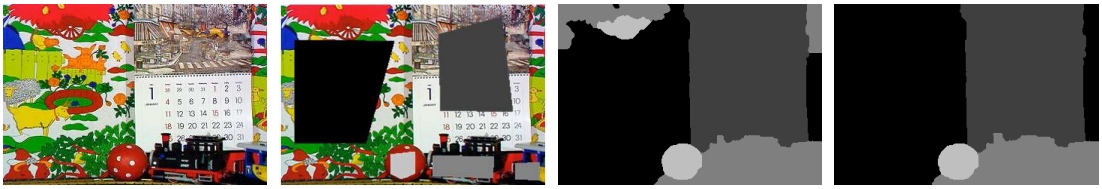


FIGURE 5.18 – Segmentation en 4 couches d'une image issue d'une séquence. De gauche à droite : image originale ; graines fournies par l'opérateur (les régions non marquées restent incertaines, à classer) ; segmentation avec le critère d'apparence et la contrainte de régularisation spatiale mais sans le critère de mouvement $\lambda_1 = 0$, $\lambda_2 = 1$ et $\lambda_3 = 25$; segmentation avec le critère de mouvement, le critère d'apparence et la contrainte de régularisation spatiale $\lambda_1 = 50$, $\lambda_2 = 1$ et $\lambda_3 = 25$. Le critère de mouvement permet de lever les ambiguïtés liées à la couleur et évite à l'opérateur d'avoir à introduire des graines supplémentaires associées à l'arrière-plan dans les deux coins supérieurs de l'image.

5.4.2 Résumé des différents types d'interactions

Nous résumons ci-dessous les étapes de nos schémas qui requièrent l'intervention de l'opérateur.

Pour le schéma 1 :

- Sélection des instants d'initialisation/réinitialisation,
- Segmentation interactive des images correspondant à ces instants,
- Choix d'appliquer ou non de la contrainte de rigidité sur les différentes couches (déjà expliquée à la section 4.4) .

Pour le schéma 2 :

- Sélection des instants de référence correspondant aux extrémités des intervalles temporels à traiter (discutée plus loin en sous-section 5.5.4),

– Segmentation interactive des images de référence,
– Correction manuelle d'erreurs de classification sur les mosaïques de couches si nécessaire.

Pour aller plus loin, nous pourrions imaginer des types d'interaction plus évolués. Par exemple, pour la correction des mosaïques, plutôt que de simplement « gommer » certaines régions mal classées, l'opérateur pourrait indiquer la couche réelle de ces régions et demander leur transfert automatique vers les bonnes mosaïques. Une solution pour cela serait de mémoriser les images originales de provenance de ces régions et de réutiliser les modèles de mouvement appropriés pour procéder au transfert. De plus, la connaissance des étiquettes réelles et des images originales de provenance permettrait de forcer directement l'étiquetage des pixels concernés avant la seconde passe de segmentation.

Comme autre type d'interaction, on peut ajouter pour les deux schémas le choix binaire (soit « avant », soit « arrière ») du sens du traitement séquentiel de la séquence ou de l'intervalle. Ce choix est guidé encore une fois par la connaissance de la séquence : il est généralement plus facile de suivre un objet dans le sens de sa disparition que dans le sens de son apparition. Avec notre premier schéma et la contrainte de rigidité applicable à des couches d'avant-plan, si une couche d'avant-plan n'est que partiellement visible au début de la séquence et apparaît totalement ensuite, il est généralement plus facile de traiter la séquence à rebours, c'est-à-dire qu'il est plus facile de partir d'une segmentation de référence où la couche d'avant-plan est visible dans son intégralité que de partir d'une segmentation où la couche d'avant-plan n'est visible que partiellement.

Un traitement séquentiel « avant PUIS arrière » éviterait à l'opérateur d'avoir à choisir entre les deux sens, mais le problème de la confrontation des résultats « avant » aux résultats « arrière » et donc de la convergence de ces résultats reste ouvert. Ceci peut constituer une piste de recherches futures.

5.5 Discussion

5.5.1 Sur le réglage des paramètres

Nos algorithmes étant basés sur la minimisation d'une fonction d'énergie qui est la somme de différents termes, la principale limitation de nos méthodes est le réglage des coefficients de pondération de chacun des termes d'énergie. L'influence de chaque terme est réglée par l'opérateur, ce qui requiert un certain entraînement d'utilisation du système. Un ensemble de paramètres adaptés pourrait être automatiquement estimé à partir des segmentations de référence fournies par l'opérateur. Les paramètres seraient alors considérés comme adaptés si les résultats de segmentation automatique coïncident avec les segmentations « vérité de terrain » obtenues de manière semi-automatique.

5.5.2 Sur les aspects semi-automatiques

Nous ne disposons pas de « vérité de terrain » ni pour les cartes de segmentation ni pour les modèles de mouvement. La qualité des résultats présentés doit donc malheu-

reusement être évaluée de manière subjective. Nous pensons avoir obtenu des résultats de qualité au moins similaire à ceux des approches simultanées.

Il faut cependant garder à l'esprit que l'intervention de l'opérateur fait partie intégrante de nos deux approches séquentielles. La comparaison aux approches plus automatiques doit être faite avec prudence. Nos contributions sont à notre sens bien adaptées à des applications de post-production. Dans un tel contexte, l'intervention d'un opérateur est pertinente. Pour les séquences résultantes, la contrainte de parfaite qualité visuelle est si forte qu'il est souvent préférable que le système se fie à des indications de l'opérateur, quitte à réduire le degré d'automatisme de la méthode, plutôt que de se baser sur une approche entièrement automatique dans laquelle l'opérateur ne peut que modifier un ensemble de paramètres ou retoucher manuellement les images une par une si les résultats ne sont pas ceux espérés.

Prenons l'exemple de l'étape d'initialisation qui est cruciale pour bon nombre de méthodes d'extraction de couches de mouvement, que celles-ci soient automatiques ou semi-automatiques. L'initialisation automatique proposée dans [Xiao 05b] utilise les premières images de la séquence et est basée sur des étapes particulièrement coûteuses en temps de calcul telles que la croissance de régions germes et la fusion de régions. Premièrement, un opérateur pourrait rapidement obtenir une initialisation similaire en fournissant très peu d'efforts. C'est ce que nous avons choisi de faire. Deuxièmement, il arrive que deux couches aient des mouvements similaires en début de séquence, mais que par la suite elles se déplacent de manière indépendante. Une initialisation automatique telle que celle décrite précédemment sera vraisemblablement incapable de distinguer les deux couches et le système risque de ne pas pouvoir les séparer efficacement ensuite. Un opérateur, quant à lui, s'il visionne la séquence au préalable ou simplement grâce à sa connaissance sémantique et ses *a priori* sur les mouvements possibles des objets composant la séquence, distinguera les deux couches lors de son initialisation semi-automatique.

5.5.3 Sur l'apparition d'une nouvelle couche

Les deux schémas s'appuient sur l'hypothèse d'un nombre de couches constant.

Le second schéma est incompatible avec l'apparition d'une nouvelle couche car cela signifierait soit que le nombre de couches n'est pas le même dans les deux instants de référence de l'intervalle considéré, soit que cette nouvelle couche disparaît totalement avant la fin de l'intervalle, ce qui rend impossible la génération de mosaïques de référence pour cette couche.

Pour le premier schéma on peut procéder de la façon suivante. Pour détecter automatiquement l'apparition d'une nouvelle couche, et donc créer une classe supplémentaire et les modèles de mouvement et d'apparence associés, on peut chercher à détecter les pixels pour lesquels l'erreur résiduelle (soit la DFD utilisant le modèle affine correspondant à l'étiquette du pixel considéré, soit le résidu (4.6)) est supérieure à un premier seuil donné [1]. Une large région connexe de tels pixels est alors considérée comme le

1. Cette détection se fait donc après obtention automatique de la carte de segmentation de l'instant courant.

support d'une nouvelle couche. La taille minimale d'une telle région est donnée par un second seuil fixé de manière empirique.

Hormis l'inconvénient lié à l'introduction de deux nouveaux paramètres, cette méthode permet bien la détection automatique de nouvelles couches, mais souvent avec un temps de retard sur leur instant réel d'apparition. Ce décalage est dû au second seuil qu'on ne peut fixer trop bas sans quoi on risque de considérer du simple bruit comme une nouvelle couche. Or, pour un objet qui apparaît progressivement et dont le mouvement est de faible amplitude, il se peut que la taille de son support soit inférieure au second seuil pour les instants qui suivent directement l'apparition. Cette nouvelle couche est donc ignorée jusqu'à ce que la taille de son support soit suffisamment grande.

On rencontre le même phénomène lors de la réapparition de la voiture à droite du panneau dans la séquence *Carmap* (figure 5.4). La classification de cette région dans la classe « voiture » est retardée de quelques images, le temps que le support soit suffisamment grand pour ne pas être filtré par la contrainte de régularisation spatiale.

Dans nos expérimentations, seule l'introduction manuelle de graines sur ces régions s'est montrée réellement efficace pour l'obtention d'une segmentation correcte dès le premier instant d'apparition.

5.5.4 Sur la durée des séquences traitées

Les séquences traitées au cours des expérimentations sont relativement courtes. Elles contiennent quelques dizaines d'images, ce qui correspond à des plans de quelques secondes seulement.

Pour ce qui est du premier schéma basé sur les triplets d'images consécutives, la longueur de la séquence n'a pas d'impact direct sur le processus de segmentation. L'opérateur conserve la possibilité d'interrompre momentanément le traitement, dès que bon lui semble, pour corriger une carte de segmentation qui sera prise en compte une fois le traitement relancé. Pour traiter une séquence plus longue, une mise à jour régulière des mélanges de gaussiennes de couleurs devrait permettre de tenir compte des nouvelles régions qui sont apparues et d'adapter les modèles d'apparence aux variations des conditions d'illumination que l'on peut rencontrer.

Pour ce qui est du second schéma, la définition des instants de référence est cruciale puisqu'on en déduit la partition de la séquence en intervalles temporels à traiter. Pour chaque intervalle, on fait l'hypothèse que les mouvements des différentes couches sont bien décrits par des modèles affines, et ce entre toute paire d'images de l'intervalle, y compris pour les paires d'images les plus éloignées. Les images correspondant aux extrémités d'un intervalle temporel doivent donc contenir, pour toutes les couches, des régions correspondantes permettant l'estimation d'un modèle affine. Prenons l'exemple d'un long travelling latéral dont l'objectif est de suivre un objet d'avant-plan en mouvement. Si l'intervalle considéré est trop long, on ne peut pas estimer le mouvement de l'arrière-plan car celui-ci est totalement différent entre les instants t_a et t_b. Aucune mise en correspondance n'est alors possible.

Les instants de référence doivent être choisis de manière judicieuse. Ils doivent être aussi peu nombreux que possible de manière à réduire au maximum les interventions

de l'opérateur, tout en respectant la contrainte évoquée ci-dessus afin que les intervalles engendrés permettent l'estimation de mouvement affine entre images éloignées. La détermination automatique de tels instants de référence est une tâche ardue qui peut faire l'objet de perspectives futures. L'analyse d'un ensemble de trajectoires de points (telles que celles évoquées dans la seconde partie de ce document) estimées au préalable tout au long de la séquence, et ayant toutes leurs propres instants de début et de fin, pourrait sans doute contribuer à la résolution de ce problème. À notre sens, un instant de référence correspond généralement à un instant auquel un certain nombre de trajectoires correspondant à un même objet naissent ou meurent.

Conclusion de la première partie

Dans cette première partie nous nous sommes concentrés sur le problème de segmentation d'une séquence d'images au sens du mouvement.

Nous avons présenté deux nouveaux schémas semi-automatiques destinés de préférence à des applications de post-production. Dans les deux cas, le problème de segmentation a été formulé comme un problème de minimisation d'une fonctionnelle d'énergie définie comme la somme de différents termes liés au mouvement, à l'apparence couleur, à des contraintes temporelles et à un *a priori* spatial. Une approche de coupe minimale dans un graphe a été retenue pour résoudre les problèmes de minimisation.

Pour les deux schémas, nous avons fait l'hypothèse que la scène était constituée d'un ensemble de couches dont les mouvements pouvaient être décrits par des modèles affines. Nous avons de plus supposé que le nombre de couches était constant tout au long de la séquence, et que l'ordre de profondeur de ces couches restait inchangé.

Concernant le premier schéma, pour une ou plusieurs images clés, les cartes de segmentation sont obtenues par l'opérateur en utilisant une méthode semi-automatique. Pour tout instant suivant, la première étape de l'algorithme proposé est la prédiction de la carte de segmentation d'une image à la suivante, en utilisant les modèles de mouvement estimés pour chaque couche. La seconde étape est le raffinement des frontières de mouvement prédites. Le critère lié au mouvement fait intervenir les mouvements estimés de l'image courante vers l'image précédente et l'image suivante. Seules les zones apparaissant et une bande d'incertitude définie autour des frontières prédites voient leur étiquetage remis en question. Partout ailleurs, sur une hypothèse de cohérence temporelle forte, nous nous fions à l'étiquetage prédit qui est conservé. Une nouvelle contrainte de rigidité appliquée à des objets rigides d'avant-plan vient renforcer la cohérence temporelle de la segmentation.

Nos expérimentations ont montré que notre approche séquentielle était aussi efficace que des approches simultanées plus complexes tout en étant moins coûteuse en temps de calcul. Du point de vue de l'opérateur, l'intérêt d'une approche séquentielle réside notamment dans la possibilité d'interrompre momentanément le processus dès qu'une carte de segmentation n'est pas satisfaisante. Ainsi l'opérateur peut intervenir à tout moment pour ajouter des indications fiables qui seront prises en compte pour la suite du processus, sans avoir à attendre que toute la séquence (ou le groupe d'images considéré) soit traitée. Il s'avère cependant que de telles interventions sont rarement nécessaires.

Étant donné les contraintes de cohérence temporelle et l'absence de remise en question de l'étiquetage prédit en dehors de la bande d'incertitude, les cartes de segmenta-

tion des images clés et les modèles affines de mouvement se doivent d'être suffisamment précis pour permettre des prédictions fiables, sans quoi notre algorithme n'assurera pas de convergence rapide vers les segmentations espérées.

Le point faible de cette première approche est que les mouvements estimés entre images consécutives ne permettent pas toujours la distinction des différents mouvements, spécialement lorsque les amplitudes de ceux-ci sont faibles. Nous avons donc conclu que si le système prenait en compte des informations de mouvement estimé entre images temporellement plus éloignées, la segmentation pourrait se faire beaucoup plus facilement. En effet, il est plus vraisemblable que des mouvements estimés entre images distantes soient différents d'une région à l'autre que lorsqu'ils sont estimés entre images consécutives.

En développant le second schéma, nous avons voulu proposer un système qui se base sur des informations de mouvement estimées entre images distantes, tout en conservant l'aspect séquentiel du traitement de la séquence. Nous avons de surcroît cherché à considérer plus de deux ou trois images simultanément à la manière des méthodes « par lots » sans pour autant que notre méthode n'impose de nombreuses et coûteuses comparaisons d'image à image. Pour cela, nous avons fait appel à la représentation compacte d'une séquence d'images en un nombre restreint de mosaïques de couches.

L'approche consiste concrètement à partitionner la séquence en un certain nombre d'intervalles temporels. Les instants correspondant aux extrémités des intervalles servent d'instants de référence pour lesquels les cartes de segmentation sont obtenues par l'opérateur de manière semi-automatique. À ces instants de référence, nous associons des mosaïques de couches de mouvement. Dans une mosaïque d'une couche donnée, nous accumulons au fur et à mesure de l'avancée du processus de segmentation les informations considérées comme appartenant à la couche. Le processus de segmentation est lancé de manière séquentielle sur chaque intervalle. Pour chaque image de l'intervalle, les mouvements utilisés pour la segmentation sont alors ceux estimés entre l'instant courant et chacun des deux instants de référence bornant l'intervalle. Un nouveau terme d'énergie lié au mouvement et basé sur la différence d'intensités entre l'image courante et une mosaïque remplace le terme usuel basé sur la différence d'intensité inter-images. Dès que la segmentation de l'image courante est effectuée, chaque mosaïque de couche est complétée par projection des régions de l'image courante associées à la couche considérée. Les mosaïques sont réutilisées pour la segmentation de l'image suivante, ainsi de suite.

La représentation en couches a finalement été exploitée de trois manières.

Premièrement, nous avons réduit le nombre de paires d'images à considérer. Le mouvement est estimé entre chaque image et deux des instants de référence seulement, et non entre de nombreuses paires d'images comme c'est le cas avec les approches simultanées. Quelques comparaisons d'image à mosaïque remplacent alors de nombreuses comparaisons d'image à image, une mosaïque d'une couche donnée contenant l'accumulation de toutes les informations déjà traitées avant l'image courante et qui ont été associées à cette couche.

Deuxièmement, puisque le point précédent implique de travailler avec des paires

d'images distantes, les mouvements estimés entre ces images distantes vont faciliter la segmentation, pour les raisons mentionnées ci-dessus.

Troisièmement, si après une première passe de segmentation les résultats ne sont pas satisfaisants, l'opérateur peut intervenir à moindre coût directement sur les mosaïques générées pour supprimer les erreurs de classification. Il lance ensuite une deuxième passe de segmentation durant laquelle les corrections apportées sont propagées à l'ensemble des images de l'intervalle temporel considéré.

Les perspectives les plus ambitieuses pour nos deux schémas concernent l'évolution des modélisations.

Pour modéliser le mouvement, le modèle paramétrique affine a été utilisé parce qu'il est simple et rapide à estimer car de faible complexité, et qu'il fournit malgré tout des estimations d'une bonne précision. Néanmoins il ne permet pas de considérer des mouvements plus complexes, et se montre trop limité quand il s'agit par exemple de traiter des cas d'objets articulés, déformables ou non planaires. Le principal objectif d'une nouvelle modélisation du mouvement serait d'étendre la portée des algorithmes. Le mouvement d'un même objet pourrait par exemple être décrit non pas par un modèle affine global mais par plusieurs modèles affines locaux (modélisation affine par morceaux) ou par un modèle non paramétrique de mélange de gaussiennes dans l'espace (dx, dy, x, y). La combinaison d'un modèle rigide affine et d'un modèle non paramétrique du type flot optique permettrait de prendre en compte des déformations locales. Quelle que soit la modélisation retenue (y compris pour une modélisation menant éventuellement à une sur-segmentation au sens du mouvement), il nous paraît intéressant d'assurer un lien vers une segmentation qui correspond à ce que perçoit l'opérateur, c'est-à-dire une segmentation sémantique des objets vidéo. Par exemple, dans le cas d'un corps articulé, les mouvements des « segments » des membres peuvent être différents. Si l'opérateur souhaite extraire ce corps, lors de l'initialisation il appréciera certainement de pouvoir fournir des indications sans faire de distinction entre ses différents « segments » et d'obtenir ensuite des résultats de segmentation lui signifiant cette même entité du corps. Il serait donc intéressant que le passage d'un niveau à l'autre (« corps articulé » à « segment », et inversement) soit automatique.

L'éventuelle transparence des différents objets n'a pas été modélisée. Dans nos travaux, un pixel appartient totalement ou n'appartient pas du tout à une couche de mouvement. En aucun cas il ne peut appartenir à plusieurs couches. Les expérimentations relatées dans cette première partie nous ont pourtant montré les ambiguïtés auxquelles pouvait mener une telle représentation binaire. Comment classer par exemple les pixels d'une région de l'arrière-plan visible au travers du pare-brise d'une voiture ?

Une modélisation des variations d'illumination pourrait également être ajoutée, notamment dans le cadre du second schéma où les estimations de mouvement se font entre images distantes et où les mosaïques de couches sont susceptibles d'être réutilisées par exemple pour le remplissage de régions après la suppression d'un objet vidéo.

Deuxième partie

Clustering d'un ensemble de trajectoires de points

Introduction de la deuxième partie

La première partie de ce document a porté sur les approches de segmentation de séquences d'images au sens du mouvement. Nous avons même parlé de segmentation « dense » dans la mesure où notre objectif était d'associer tous les pixels du volume vidéo à une classe de mouvement.

Le critère lié au mouvement et la contrainte temporelle mentionnés en première partie, aussi bien dans l'état de l'art que dans nos propres travaux, s'appuient sur des comparaisons d'informations provenant d'instants différents. Mais des contraintes d'occupation mémoire, ou de traitement séquentiel conduisent souvent à n'utiliser qu'une faible partie des informations utiles disponibles dans la séquence. Les comparaisons sont alors faites entre informations provenant de paires d'images consécutives, de triplets d'images consécutives, de paires d'images distantes, de triplets d'images distantes, ou de groupes (d'un nombre limité) d'images. Et cet échantillonnage temporel de la séquence est fixé pour tous les pixels d'une même image, voire du volume vidéo entier. Pourtant chaque point de la séquence a des instants de visibilité qui lui sont propres. Il semble alors injustifié de procéder à des comparaisons d'informations provenant des mêmes instants pour tous les pixels d'une image par exemple.

S'il est difficile voire impossible à cause de limitations techniques de prendre en compte simultanément plusieurs dizaines d'images d'une séquence, il est relativement aisé de procéder à un sous-échantillonnage non pas temporel, mais spatial de la séquence. Ce sous-échantillonnage spatial peut consister par exemple en la détection de points d'intérêt, chacun de ces points étant ensuite suivi aussi longtemps que possible sans contrainte sur la longueur de l'intervalle temporel de suivi.

Dans cette seconde partie, nous n'allons plus raisonner au niveau de pixels mais au niveau de trajectoires de points. Nous allons effectivement nous concentrer sur le clustering d'un ensemble épars de trajectoires de points d'intérêt. Toutes les informations de déplacement d'un point donné au cours de la séquence sont intégrées dans la trajectoire de ce point. Toutes les positions traversées par une trajectoire de point sont donc liées entre elles et vont être associées à une même classe de mouvement. L'avantage d'une trajectoire de point, comparée à un pixel isolé, est que l'intervalle temporel sur lequel elle est définie est par nature adapté aux instants de visibilité du point considéré. L'intégration adaptée de ces informations temporelles va permettre de procéder au clustering de l'ensemble des trajectoires dans les meilleures conditions sur

l'unique critère de mouvement.

Le chapitre 6 présente un état de l'art des méthodes de clustering d'un ensemble de trajectoires de points. Plusieurs de ces méthodes sont limitées au traitement de trajectoires qui sont toutes définies sur un même intervalle temporel, donc toutes de même longueur. Or, à cause des changements de pose et/ou d'apparence des objets auxquels ils appartiennent, les différents points d'intérêt détectés sont associés à des trajectoires ayant chacune leur propre instant de début, leur propre instant de fin, et donc leur propre longueur.

Le chapitre 7 décrit notre approche pour résoudre le problème spécifique du clustering d'un ensemble de trajectoires de points de différentes longueurs sans avoir recours à l'extrapolation des données manquantes. Des expérimentations ont été menées sur une base de données standard reconnue et sur d'autres séquences.

Cette seconde partie se termine sur des éléments de synthèse et sur une première piste de recherches ultérieures possibles.

Chapitre 6

État de l'art des méthodes de clustering d'un ensemble de trajectoires de points

Dans la suite du document, nous utiliserons parfois le terme anglais de « clustering » pour désigner le partitionnement d'un ensemble d'observations. Le terme « cluster » désignera alors un groupe d'observations résultant de ce partitionnement.

Le problème de segmentation d'une séquence d'images au sens du mouvement peut être formulé comme un problème de clustering d'un ensemble épars de trajectoires de points. De telles méthodes de clustering conviennent pour des applications qui ne nécessitent pas l'extraction précise des frontières d'un objet. On les retrouve donc pour des tâches de détection d'objets en mouvement, de reconnaissance d'action, de suivi, ou encore de surveillance. En général, les trajectoires sont d'abord estimées par suivi de points d'intérêt tout au long de la séquence, puis l'algorithme de clustering est appliqué à l'ensemble de trajectoires obtenues.

Ce chapitre a pour but de dresser un rapide état de l'art des principales méthodes de clustering utilisées pour partitionner un ensemble de trajectoires de points. Cet état de l'art ne se veut pas exhaustif et se limite aux algorithmes les plus connus et les plus répandus. Nous choisissons de regrouper ces algorithmes en quatre catégories : les approches hiérarchiques, les approches de factorisation, les approches de clustering spectral, et les approches basées sur l'algorithme de RANSAC. Pour chaque catégorie d'approches, nous présentons le principe général de clustering d'un ensemble de données quelconques, puis nous citons des travaux liés spécifiquement au clustering d'un ensemble de trajectoires de points. Les travaux de Pundlik et Birchfield sont présentés dans une section indépendante (section 6.5) car il s'agit de la seule approche qui vise l'estimation et la segmentation de trajectoires en temps réel.

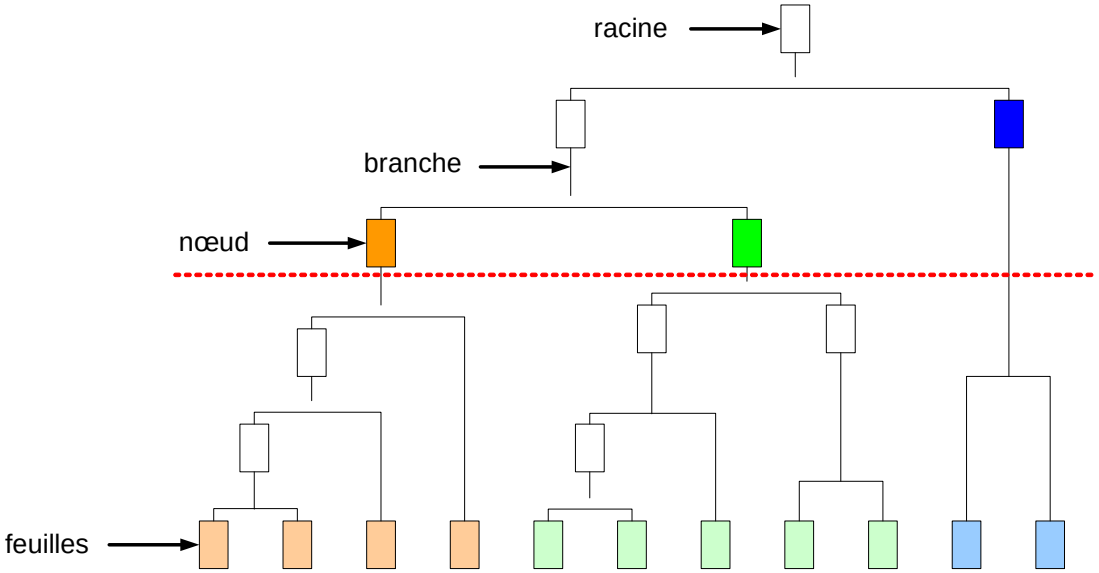


FIGURE 6.1 – Exemple de dendrogramme. La droite de coupure, rouge en pointillé, fournit une partition en 3 clusters. La hauteur des parties horizontales des branches correspond à la distance entre les clusters fusionnés.

6.1 Approches hiérarchiques

6.1.1 Hiérarchie, dendrogramme et algorithme d'agrégation

Étant donné un ensemble d'observations, une hiérarchie sur cet ensemble est une collection de groupes d'observations (clusters) tels que :

- chacune des observations est un cluster (singleton),
- l'ensemble complet des observations est un cluster,
- étant donnés deux clusters de la hiérarchie, soit ils n'ont aucune observation en commun, soit l'un des clusters est inclus dans l'autre (en aucun cas il n'y a de chevauchement partiel).

Le principe des approches de clustering hiérarchique est de construire un arbre appelé dendrogramme. Ce dendrogramme représente la hiérarchie (figure 6.1). Au niveau le plus bas de cet arbre, on trouve les « feuilles » qui correspondent aux clusters singletons contenant chacun une observation. Au niveau le plus haut, le cluster contenant toutes les observations est appelé « racine ». Chaque cluster de l'arbre est appelé un « nœud ». Chaque cluster (excepté les singletons) a plusieurs « enfants », habituellement deux pour des raisons pratiques de construction. Une « branche » désigne une ligne joignant un nœud à l'un de ses enfants. Le clustering final est obtenu en coupant l'arbre horizontalement à la hauteur désirée. Sur la figure 6.1, la droite rouge en pointillé indique la coupure, elle intersecte 3 branches. L'ensemble des nœuds directement supérieurs à la coupure fournit le clustering correspondant : 3 clusters orange, vert et bleu composés des feuilles pâles de même couleur.

La construction de la hiérarchie peut se faire de deux façons. Les approches hiérarchiques ascendantes, ou par « agrégation », sélectionnent de manière itérative les deux clusters dont la « distance » est la plus faible pour les fusionner en un cluster parent. À l'inverse, les approches descendantes, ou par « division », s'attachent à scinder un cluster donné en deux enfants, la distance entre les deux enfants devant être la plus grande possible pour créer deux clusters bien séparés. Dans les deux cas, la mesure de distance entre deux clusters doit être définie *a priori*.

Dans le cas d'une approche hiérarchique descendante, si l'on entend respecter strictement le critère de maximisation de la distance entre les deux enfants, toutes les partitions possibles du cluster en deux enfants doivent être considérées. Les approches hiérarchiques descendantes sont généralement longues et coûteuses, c'est pourquoi les approches ascendantes sont plus répandues.

Nous décrivons maintenant l'algorithme d'agrégation. En pratique, on commence par construire autant de clusters singletons (feuilles) que d'observations. On calcule ensuite les distances entre chaque paire de clusters. Ces distances sont stockées dans un tableau. Les deux clusters les plus proches sont regroupés en un même cluster parent et resteront unis jusqu'à la fin du processus. Le tableau de distances est alors mis à jour en considérant ce nouveau cluster.

On distingue habituellement trois approches que nous présentons de la plus simple à la plus complexe en temps de calcul. Dans l'approche « single link », la distance entre deux clusters est définie comme la plus petite distance entre deux éléments de chacun des deux clusters. Cette approche locale présente l'inconvénient d'avoir tendance à former de longues chaînes qui ne correspondent pas à la notion intuitive de clusters comme étant des objets compacts et sphériques. Dans l'approche « complete link », la distance entre deux clusters est définie comme la plus grande distance entre deux éléments de chacun des deux clusters. Cette approche non locale présente l'inconvénient d'être particulièrement sensible aux observations aberrantes. L'approche « mean link » est un compromis entre les deux approches précédentes. La distance entre deux clusters est définie comme la moyenne de la distance entre deux éléments de chacun des deux clusters.

Les regroupements de clusters sont itérés jusqu'à ce qu'un critère d'arrêt défini *a priori* soit atteint :

- si le nombre de clusters désirés est connu, les regroupements cessent lorsqu'il est atteint,
- si le nombre de clusters désirés est inconnu, on fixe une valeur de distance comme seuil d'arrêt au delà duquel les regroupements sont interdits.

L'inconvénient majeur d'un tel algorithme de clustering est le calcul coûteux d'une distance entre chaque paire d'observations. L'utilisation d'une méthode de ce type pour de grands ensembles de données est donc à éviter. Par ailleurs, l'approche étant ascendante, considérer les observations deux par deux n'autorise que l'estimation de modèles de très faible complexité. Il serait pourtant intéressant de pouvoir utiliser dans la définition de la distance un modèle plus complexe mais son estimation nécessiterait une approche plus globale.

6.1.2 Distances entre séries temporelles

Concernant le clustering d'un ensemble de séries temporelles (des trajectoires de points dans notre cas), les trois distances les plus fréquemment utilisées dans les approches hiérarchiques ascendantes sont la distance euclidienne, la distance « Dynamic Time Warping » (DTW) et la distance basée sur la longueur de la plus longue sous-séquence commune (« Longest Common SubSequence » (LCSS)). Ces trois distances sont définies ci-après, pour des données en dimension 1. Les définitions s'étendent facilement pour des données de dimension supérieure (dimension 2 pour nos trajectoires de points).

Distance euclidienne

Considérons deux séries temporelles $\mathbf{u} = (u_1, \cdots, u_M)$ et $\mathbf{v} = (v_1, \cdots, v_M)$ de même longueur M. La distance euclidienne entre $\mathbf{u}$ et $\mathbf{v}$ est définie par :

$$d_{\text{Eucl}}(\mathbf{u}, \mathbf{v}) = \sqrt{\sum_{t=1}^{M} (u_t - v_t)^2} \ . \tag{6.1}$$

La distance géométrique moyenne entre les deux séries temporelles $\mathbf{u}$ et $\mathbf{v}$ est définie par :

$$d_{\text{gm}}(\mathbf{u}, \mathbf{v}) = \frac{d_{\text{Eucl}}(\mathbf{u}, \mathbf{v})}{M} \ . \tag{6.2}$$

La distance euclidienne compare les éléments temporels des séries « point à point ». Cette définition n'est donc valable que pour deux séries temporelles de même longueur.

À titre d'exemple, une approche hiérarchique ascendante et la distance géométrique moyenne ont été utilisées pour classer en 7 catégories les 365 jours d'une année en analysant des séries temporelles journalières (sur 24h) représentant la consommation électrique d'une entreprise en fonction de l'heure de la journée ou encore le nombre d'employés présents en fonction de l'heure de la journée [Van Wijk 99].

L'approche bayésienne proposée dans [Brostow 06] s'appuie d'abord sur le clustering hiérarchique d'un ensemble de trajectoires de points de longueurs variables. Le but de l'algorithme n'est pas le suivi d'entités individuelles mais leur détection. L'étape de clustering est itérée indépendamment pour chaque image I_t. Pour procéder au clustering à l'instant t, on s'intéresse uniquement aux trajectoires présentes à l'instant t, mais on considère leurs données sur l'intervalle temporel $[t-30, t+30]$. Les trajectoires incomplètes sur cet intervalle sont extrapolées linéairement à partir des dernières positions connues, uniquement à des fins de détection et non de suivi. Le clustering hiérarchique est basé sur un *a priori* spatial en utilisant la distance suivante entre deux clusters C_i et C_j :

$$d(C_i, C_j) = \max_{(\mathbf{u} \in C_i, \mathbf{v} \in C_j)} d_{\text{Eucl}}(\mathbf{u}, \mathbf{v}) \ . \tag{6.3}$$

Ce clustering initial est mené pour disposer d'une sur-segmentation de l'ensemble des trajectoires sur le critère spatial. On conserve donc un nombre volontairement

élevé de clusters avant de procéder au clustering final qui comprend uniquement des fusions de clusters selon des similarités de mouvement. L'hypothèse alors utilisée est celle selon laquelle deux trajectoires suivent vraisemblablement le même objet rigide si leur variance en distance est faible au cours de l'intervalle temporel Δt commun aux deux trajectoires considérées. Cette variance est définie par :

$$\mathrm{var}(\mathbf{u},\mathbf{v}) = \mathrm{var}_{t\in\Delta t}\,|u_t - v_t| \quad . \tag{6.4}$$

La probabilité que les trajectoires $\mathbf{u}$ et $\mathbf{v}$ appartiennent au même objet rigide est alors définie par :

$$Q(\mathbf{u},\mathbf{v}) = \frac{1}{1+\mathrm{var}(\mathbf{u},\mathbf{v})} \quad . \tag{6.5}$$

Notons tout de même que l'hypothèse précédente n'est vraie que pour des mouvements parallèles au plan image. Par ailleurs, comme le clustering est effectué de manière indépendante pour chaque image, à des fins de détection et non de suivi, les incohérences temporelles restent nombreuses.

Distance « Dynamic Time Warping » (DTW)

La distance DTW doit son nom à l'utilisation de la programmation dynamique pour résoudre le problème d'optimisation induit. Considérons deux séries temporelles $\mathbf{u} = (u_1, \cdots, u_M)$ et $\mathbf{v} = (v_1, \cdots, v_N)$ de longueurs M et N. La distance DTW entre $\mathbf{u}$ et $\mathbf{v}$ est définie de façon récursive par [Berndt 94] :

$$DTW(\mathbf{u},\mathbf{v}) = |u_M - v_N| + \min\Big(DTW\big(H(\mathbf{u}),H(\mathbf{v})\big), DTW\big(H(\mathbf{u}),\mathbf{v}\big), DTW\big(\mathbf{u},H(\mathbf{v})\big)\Big) \quad , \tag{6.6}$$

où la fonction $H(\cdot)$ est définie par $H(\mathbf{u}) = (u_1, \cdots, u_{M-1})$.

Cette distance permet de comparer des séries temporelles de longueurs différentes. Les comparaisons des éléments temporels se font en tolérant des étirements et des contractions temporels, ce qui rend la distance DTW plus souple que la distance euclidienne. Néanmoins, comme pour la distance euclidienne, chaque élément d'une série (y compris un élément aberrant) doit être apparié à un élément de l'autre série, ce qui rend la distance peu robuste au bruit. Par ailleurs la distance DTW est plus coûteuse en temps de calcul que la distance euclidienne.

Distance basée sur la longueur de la plus longue sous-séquence commune (« Longest Common SubSequence » (LCSS))

Pour comparer deux séries temporelles de manière efficace, même en présence de bruit, le principe de la LCSS est de les comparer en les autorisant à s'étirer temporellement, tout en tolérant que certains éléments d'une série ne soient pas appariés à un élément de l'autre série.

Avant de définir la distance basée sur la LCSS, nous définissons la longueur de la LCSS elle-même. Considérons deux séries temporelles $\mathbf{u} = (u_1, \cdots, u_M)$ et $\mathbf{v} =$

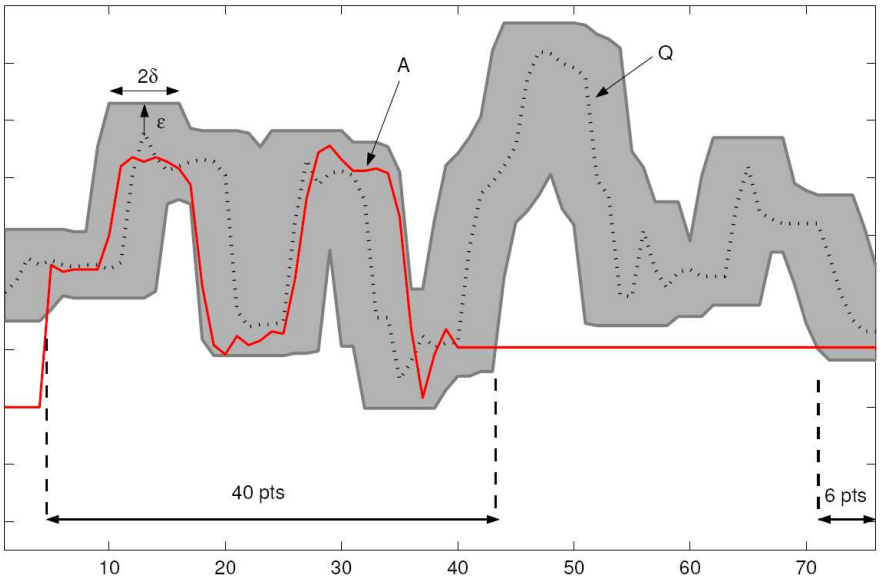


FIGURE 6.2 – Illustration du principe de la LCSS pour des séries temporelles de dimension 1. Un élément de la série temporelle rouge (**A**) ne peut être apparié à un élément de la série temporelle en pointillé (**Q**) que s'il se situe dans l'enveloppe grise définie autour de cette série selon les paramètres de contraintes en temps et espace δ et ϵ. Cette figure provient de [Vlachos 03].

$(v_1, \cdots, v_N)$ de longueurs M et N, et la fonction $H(\cdot)$ définie par $H(\mathbf{u}) = (u_1, \cdots, u_{M-1})$. Alors la plus longue sous-séquence commune à $\mathbf{u}$ et $\mathbf{v}$ est de longueur :

$$L_{\delta,\epsilon}(\mathbf{u},\mathbf{v}) = \begin{cases} 0 & \text{si } \mathbf{u} \text{ ou } \mathbf{v} \text{ est de longueur 0} \\ 1 + L_{\delta,\epsilon}(H(\mathbf{u}),H(\mathbf{v})) & \text{si } |u_M - v_N| < \epsilon \text{ et } |M-N| \leq \delta \\ \max\Big(L_{\delta,\epsilon}(H(\mathbf{u}),\mathbf{v}), L_{\delta,\epsilon}(\mathbf{u},H(\mathbf{v}))\Big) & \text{sinon} \end{cases} , \tag{6.7}$$

où le paramètre δ est un entier positif qui détermine la distance temporelle maximum pouvant séparer deux éléments appariés, et le paramètre ϵ est une constante positive qui détermine la distance spatiale maximum pouvant séparer deux éléments appariés. Ainsi pour apparier deux éléments, ceux-ci doivent être proches à la fois spatialement et temporellement. La figure 6.2 illustre le principe de la LCSS.

La distance basée sur la LCSS est alors définie par :

$$D_{\delta,\epsilon}(\mathbf{u},\mathbf{v}) = 1 - \frac{L_{\delta,\epsilon}(\mathbf{u},\mathbf{v})}{\min(M,N)} \ . \tag{6.8}$$

La fonction $L(\cdot)$ prenant pour valeurs des entiers entre 0 et $\min(M,N)$, la fonction $D(\cdot)$ prend ses valeurs entre 0 et 1.

Dans le cas du clustering de trajectoires de points, considérons deux trajectoires ayant même instant de début, même longueur et étant identiques à une translation

spatiale près (c'est le cas par exemple de trajectoires suivant deux points d'un même objet rigide en translation dans le plan image). On s'attend à ce que ces trajectoires soient regroupées dans un même cluster, c'est-à-dire à ce que leur distance soit faible. D'après la définition précédente de la distance basée sur la LCSS, il n'en est rien si le vecteur de translation qui permet d'aligner spatialement les deux trajectoires est d'amplitude supérieure au paramètre ϵ. On est alors incapable de détecter une similarité entre des mouvements parallèles dans l'espace.

Pour remédier à cela, Vlachos *et al.* [Vlachos 02a, Vlachos 02b] définissent une nouvelle distance toujours basée sur la LCSS mais également basée sur une famille de translations. L'idée est de trouver la translation qui, lorsqu'appliquée à la trajectoire $\mathbf{v}$, maximise la longueur de la plus longue sous-séquence commune entre $\mathbf{u}$ et la translatée de $\mathbf{v}$.

Soit $\mathcal{F}$ la famille de fonctions de translation, $f_a \in \mathcal{F}$ la fonction de translation de paramètre a (ici pour des données de dimension 1), et $\mathbf{u}$ et $\mathbf{v}$ deux séries temporelles de longueurs M et N. La translatée de $\mathbf{v}$ par f_a est définie par :

$$f_a(\mathbf{v}) = (v_1 + a, \cdots, v_N + a) \ . \tag{6.9}$$

La longueur de la plus longue sous-séquence commune à $\mathbf{u}$ et $\mathbf{v}$, et la distance entre $\mathbf{u}$ et $\mathbf{v}$ sont désormais définies par :

$$L2_{\delta,\epsilon}(\mathbf{u}, \mathbf{v}) = \max_{f_a \in \mathcal{F}} \Big(L_{\delta,\epsilon}(\mathbf{u}, f_a(\mathbf{v})) \Big) \ , \tag{6.10}$$

et

$$D2_{\delta,\epsilon}(\mathbf{u}, \mathbf{v}) = 1 - \frac{L2_{\delta,\epsilon}(\mathbf{u}, \mathbf{v})}{\min(M, N)} \ . \tag{6.11}$$

Les travaux basés sur une approche hiérarchique ascendante, et dont la problématique est la plus proche de la nôtre, à savoir le regroupement, selon des similarités de mouvement, de trajectoires de points de longueurs variables, sont à notre connaissance ceux de Buzan *et al.* [Buzan 04], largement inspirés des travaux de Vlachos *et al.* Néanmoins les mouvements qui peuvent être traités efficacement par cette approche sont principalement translationnels.

Si Vlachos *et al.* ont montré que la distance basée sur la LCSS permet une meilleure comparaison de séries temporelles que la distance DTW, le choix des paramètres δ et ϵ constitue un inconvénient majeur. Un autre inconvénient de la distance basée sur la LCSS est le nombre fini de valeurs possibles puisque pour deux trajectoires $\mathbf{u}$ et $\mathbf{v}$ de longueurs M et N, seules $\min(M, N) + 1$ valeurs sont possibles pour $L_{\delta,\epsilon}(\mathbf{u}, \mathbf{v})$, les entiers entre 0 et $\min(M, N)$. Cela constitue en particulier un problème lorsque l'on travaille avec des trajectoires relativement courtes. La distance s'avère alors moins discriminante qu'une distance dont le nombre de valeurs possibles est infini.

6.2 Approches de factorisation

6.2.1 Idée générale

Les approches de factorisation appartiennent aux méthodes de partitionnement de données qui déterminent tous les clusters en même temps. Elles reposent sur le principe de réduction de la dimension des espaces de données. Un prétraitement des données originales brutes est nécessaire pour les remplacer par leurs projections orthogonales dans un sous-espace de faible dimension.

Considérons P trajectoires 2D, toutes de même longueur F, associées à un ensemble de P points 3D appartenant à un unique objet rigide. Sous l'hypothèse du modèle affine de caméra, ces trajectoires résident dans un sous-espace de $\mathbb{R}^{2F}$ de dimension 2, 3, ou 4. Pour démontrer ceci, on représente la matrice $\mathbf{W}$ des trajectoires de points de dimension $2F \times P$, pour laquelle chaque colonne représente une trajectoire, comme le produit matriciel d'une matrice de mouvement de dimension $2F \times 4$ et de la transposée d'une matrice de structure de dimension $P \times 4$. Le rang de la matrice $\mathbf{W}$ est au plus 4 [Costeira 95].

Considérons maintenant P trajectoires 2D, toutes de même longueur F, associées à un ensemble de P points 3D appartenant à n objets rigides. Le problème de segmentation au sens du mouvement revient à partitionner l'ensemble de trajectoires selon les n objets en mouvement. Puisque les trajectoires associées à chaque objet i appartiennent à un sous-espace linéaire de $\mathbb{R}^{2F}$ de dimension d_i, le problème de segmentation revient également à partitionner un ensemble de points en n sous-espaces de $\mathbb{R}^{2F}$ de dimensions inconnues $d_i \in \{2, 3, 4\}$ avec $i = 1, \cdots, n$.

De nombreux algorithmes de segmentation de mouvement sont basés sur cette idée [Boult 91, Costeira 95, Ichimura 99, Kanatani 01]. Parmi les algorithmes plus récents, on peut citer [Sugaya 04, Yan 06, Vidal 04b, Rao 08].

Tron et Vidal [Tron 07] ont généré un ensemble de données pour l'évaluation et la comparaison de tels algorithmes. Il s'agit de la base de données *Hopkins155*.

6.2.2 Projection des trajectoires

Comme mentionné ci-dessus, un prétraitement des données originales brutes est nécessaire pour les remplacer par leurs projections orthogonales dans un sous-espace de faible dimension.

La projection des données doit se faire sur un sous-espace propre dominant de la matrice $\mathbf{W}$ des trajectoires afin de perdre aussi peu d'informations que possible. La décomposition en valeurs singulières (« Singular Value Decomposition » (SVD)) de la matrice $\mathbf{W}$ permet de déterminer ce sous-espace dont on notera D la dimension.

Pour [Vidal 04b], puisque la dimension maximale de chaque sous-espace de mouvement est 4, la projection sur un sous-espace générique de dimension $D = 5$ préserve le nombre et les dimensions des n sous-espaces de mouvement. Dit autrement, pour que deux mouvements différents soient distinguables, les sous-espaces de mouvement n'ont pas besoin d'être différents dans les 4 dimensions, il suffit qu'ils soient différents dans une dimension. Ainsi toutes les configurations possibles peuvent être traitées : de n

mouvements de dimension 2 partiellement dépendants à n mouvements de dimension 4 entièrement indépendants.

Pour des scènes plus complexes contenant par exemple des mouvements articulés, un sous-espace de dimension $D = 5$ peut ne pas suffire. Dans l'algorithme LSA (« Local Subspace Affinity ») de Yan et Pollefeys [Yan 06], le sous-espace de projection est de dimension $D = \text{rang}(\mathbf{W}) \leq 4n$. La valeur de D est déterminée en utilisant une technique de sélection de modèle [Vidal 04a]. Les points projetés dans $\mathbb{R}^D$ sont projetés à nouveau sur une hypersphère $\mathbb{S}^{D-1}$ en fixant leur norme à 1.

6.2.3 Estimation des sous-espaces et clustering

Vidal et Hartley [Vidal 04b] procèdent à une estimation globale des sous-espaces en s'appuyant sur l'algorithme d'analyse en composantes principales généralisée (« Generalized Principal Component Analysis » (GPCA))[Vidal 05]. Cet algorithme est une méthode algébrique de partitionnement de données en plusieurs sous-espaces. L'idée principale est d'établir une correspondance entre un ensemble de n sous-espaces et un ensemble de polynômes de degré n, dont les dérivées en un point donnent un vecteur normal au sous-espace contenant ce point. Le partitionnement des données est alors obtenu par clustering de ces vecteurs normaux, réalisé en appliquant l'algorithme de clustering spectral [Ng 01]. Les détails de cet algorithme sont donnés à la section suivante 6.3.

L'algorithme LSA [Yan 06], quant à lui, ajuste à chaque point et ses k plus proches voisins un sous-espace local dont la dimension est déterminée par une technique de sélection de modèle. Pour chaque paire de ces P sous-espaces locaux, une mesure de similarité est définie en fonction des angles principaux entre les deux sous-espaces considérés. Le clustering des données est obtenu en appliquant l'algorithme de clustering spectral [Ng 01] à la matrice d'affinité contenant ces similarités.

6.2.4 Trajectoires de différentes longueurs

Toutes les trajectoires traitées par [Sugaya 04, Yan 06] sont correctes et complètes, c'est-à-dire que l'algorithme de suivi de points fonctionne parfaitement durant l'étape d'estimation des trajectoires. Toutes les trajectoires ont la même longueur, elles débutent toutes au premier instant de la séquence considérée et terminent toutes au dernier instant.

Cependant, lorsque l'on cherche à suivre un point particulier, il est très fréquent de ne pas pouvoir suivre ce point durant toute la séquence. En particulier, les occultations et disparitions du champ de vision engendrent des trajectoires incomplètes. Des changements de pose, de conditions d'illumination ou d'apparence peuvent également expliquer que l'algorithme de suivi soit mis en échec. En conséquence, chaque trajectoire de point a pour caractéristiques propres un instant de début et un instant de fin, et donc une longueur propre.

Plusieurs approches ont été proposées pour traiter de tels ensembles de trajectoires de points. Or l'inconvénient principal des approches de factorisation est que toutes les

trajectoires considérées doivent être de même longueur. En présence de trajectoires de longueurs différentes, l'extrapolation de données pour les trajectoires incomplètes ne peut pas être évitée.

Vidal et Hartley [Vidal 04b] utilisent une adaptation de la méthode de « power factorization » [Hartley 04] pour déduire les données projetées manquantes.

Rao *et al.* [Rao 08] proposent l'algorithme ALC (« Agglomerative Lossy Compression »), robuste à la présence de trajectoires de points aberrantes, incomplètes ou corrompues. En supposant qu'il existe des trajectoires complètes pour chaque cluster de mouvement, les trajectoires incomplètes sont individuellement complétées avant l'étape de séparation en sous-espaces, en utilisant aussi peu de trajectoires complètes que possible. On considère pour cela qu'il existe une combinaison linéaire des trajectoires complètes dont la restriction à l'intervalle temporel sur lequel est définie la trajectoire incomplète est égale à celle-ci. Cette combinaison linéaire n'étant généralement pas unique, on retient celle pour laquelle le vecteur de coefficients associé présente la norme la plus faible. La partie manquante de la trajectoire incomplète est calculée en utilisant la combinaison linéaire retenue non restreinte. Les trajectoires complètes utilisées pour compléter une trajectoire s'avèrent généralement appartenir à un seul des sous-espaces extraits ensuite.

6.3 Approches de clustering spectral

Les algorithmes de clustering spectral sont largement utilisés pour le clustering non paramétrique de données. Ils utilisent les vecteurs propres de matrices dérivées des données. L'algorithme de Ng *et al.* [Ng 01] se décompose de la façon suivante :

Soit un ensemble de données $\{s_1, \cdots, s_P\} \in \mathbb{R}^l$ que l'on souhaite partitionner en n sous-ensembles (n étant fixé) :

1. Construire la matrice d'affinité ou de similarité $\mathbf{A} \in \mathbb{R}^{P \times P}$ telle que :
$$A_{ij} = \begin{cases} \exp(-\|s_i - s_j\|^2 / 2\sigma^2) & \text{si } i \neq j \\ 0 & \text{si } i = j \end{cases} . \tag{6.12}$$
2. Construire la matrice diagonale $\mathbf{D}$ telle que $D_{ii} = \sum_{j=1}^{P} A_{ij}$, et en déduire la matrice $\mathbf{L} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$.
3. Calculer $\mathbf{x}_1, \cdots, \mathbf{x}_n$ les vecteurs propres associés aux n plus grandes valeurs propres de $\mathbf{L}$ (choisis orthogonaux les uns aux autres si une même valeur propre se répète), et former la matrice $\mathbf{X} = [\mathbf{x}_1 \mathbf{x}_2 \cdots \mathbf{x}_n] \in \mathbb{R}^{P \times n}$.
4. Construire la matrice $\mathbf{Y}$ en normalisant chaque ligne de $\mathbf{X}$: $Y_{ij} = X_{ij} / (\sum_j X_{ij}^2)^{1/2}$.
5. Chaque ligne de $\mathbf{Y}$ étant un point de $\mathbb{R}^n$, effectuer le clustering en n groupes des P lignes de $\mathbf{Y}$, en utilisant par exemple l'algorithme de clustering des k-moyennes.
6. Assigner le point s_i au cluster j si la i-ème ligne de $\mathbf{Y}$ appartient au cluster j.

À l'étape 1, le paramètre σ permet d'ajuster la vitesse de décroissance de la mesure de similarité en fonction de la distance entre s_i et s_j. Ng *et al.* suggèrent de procéder

à plusieurs clusterings avec différentes valeurs de σ et de déterminer automatiquement parmi celles-ci la valeur souhaitable, c'est-à-dire celle qui fournit les clusters de plus faibles dispersions. Dans l'équation (6.12), l'utilisation de la distance euclidienne nécessite de travailler sur des données de tailles égales. L'étape 5 fait appel à un autre algorithme de clustering dont on profitera des avantages et subira les inconvénients.

Azran et Ghahramani [Azran 06] proposent des algorithmes pour déterminer automatiquement la valeur du paramètre σ qui conduit à la partition la plus plausible sans que le nombre n de clusters ne soit connu *a priori*. L'approche adoptée s'inspire de l'approche de clustering spectral basée sur l'idée de marche aléatoire [Meila 01]. Azran et Ghahramani considèrent non pas une étape de marche aléatoire mais N étapes. La matrice d'affinité, déjà paramétrée par σ, dépend alors également du nombre N d'étapes de marches aléatoires. Une optimisation conjointe sur σ et N permet de déterminer automatiquement la combinaison de paramètres qui va engendrer la partition la plus plausible et donc le nombre de clusters associé à cette partition. Le paramètre σ influe sur la structure locale de voisinage au sein de la partition tandis que N révèle la structure globale.

Parmi les algorithmes de clustering spectral, on peut également citer l'algorithme de « normalized cuts » [Shi 00] qui est destiné à la segmentation d'images.

Travaillant sur les trajectoires de points pour la segmentation au sens du mouvement, Liu *et al.* [Liu 05] introduisent la corrélation normalisée comme mesure de similarité entre deux trajectoires. Ils ont recours à l'algorithme de clustering spectral [Shi 98] pour partitionner un ensemble de trajectoires en un nombre de clusters fixé. La corrélation normalisée est invariante à la fois à la direction des trajectoires et à leur amplitude. Pour deux trajectoires de points $\mathbf{u}$ et $\mathbf{v}$ de même longueur M, soit (w^x_{ut}, w^y_{ut}) et (w^x_{vt}, w^y_{vt}) les composantes horizontales et verticales des vecteurs vitesses des trajectoires $\mathbf{u}$ et $\mathbf{v}$ à l'instant t, la corrélation est alors définie entre les vecteurs vitesses complexes par :

$$\rho(\mathbf{u},\mathbf{v}) = \left| \frac{\sum_t (w^x_{ut} + i w^y_{ut})(w^x_{vt} + i w^y_{vt})}{\sqrt{\sum_t \left((w^x_{ut})^2 + (w^y_{ut})^2\right) \times \sum_k \left((w^x_{vt})^2 + (w^y_{vt})^2\right)}} \right| , \tag{6.13}$$

avec $i = \sqrt{-1}$.

Dans ces travaux, la matrice de similarité contient donc ces mesures de corrélation. Avant le clustering, une trajectoire $\mathbf{u}$ ayant moins de 5 trajectoires $\mathbf{v}$ telles que $\rho(\mathbf{u},\mathbf{v}) > 0.9$ est considérée comme aberrante et est rejetée définitivement.

6.4 Approches basées sur l'algorithme de RANSAC

6.4.1 Algorithme de RANSAC

L'algorithme de « RANdom SAmple Consensus » (RANSAC) a initialement été proposé par Fischler et Bolles [Fischler 81]. C'est une méthode itérative qui permet d'estimer les paramètres d'un modèle mathématique à partir d'un ensemble de P points

(ou observations), pouvant être corrompu par de nombreuses valeurs aberrantes. On fait l'hypothèse que la distribution des points corrects peut être expliquée par un modèle paramétrique, et on considère comme aberrants les points ne correspondant pas au modèle. L'algorithme de RANSAC est alors capable de déterminer ce modèle sans que les points aberrants ne perturbent l'estimation des paramètres. Le modèle s'ajuste uniquement aux points corrects.

Si d est le nombre minimum de points nécessaires à l'estimation d'un modèle, d points de l'ensemble de données sont sélectionnés aléatoirement pour former un sous-ensemble $\mathcal{S}$ d'échantillons (appelé parfois « Minimal Sample Set » (MSS)). On fait l'hypothèse que $\mathcal{S}$ ne contient pas de points aberrants. L'hypothèse est testée par la procédure suivante :

1. Un modèle $\mathcal{M}$ est ajusté aux d points de $\mathcal{S}$ (par exemple par la méthode des moindres carrés), c'est-à-dire que tous les paramètres du modèle sont estimés uniquement à partir de ce sous-ensemble de points.
2. Tous les autres points sont ensuite testés sur le modèle $\mathcal{M}$ précédemment estimé. Pour chaque point on estime l'erreur résiduelle au modèle $\mathcal{M}$. Si cette erreur résiduelle est inférieure à un seuil τ fixé, alors le point est considéré comme une réalisation possible de $\mathcal{M}$.
3. Si suffisamment de points ont été classés comme réalisations possibles de $\mathcal{M}$, un nouveau modèle $\mathcal{M}^*$ est estimé à partir de cet ensemble $\mathcal{S}^*$ de réalisations possibles, ensemble appelé l'ensemble de consensus de $\mathcal{M}$ (« consensus set » (CS)).

À l'étape 3, le modèle $\mathcal{M}^*$ est estimé si et seulement si $|\mathcal{S}^*|$ est supérieur à un seuil T qui dépend de la proportion de points aberrants dans l'ensemble de points initial. Cette procédure est répétée pour d autres points sélectionnés aléatoirement tant qu'à l'étape 3 on a $|\mathcal{S}^*| < T$. Si après un nombre N prédéterminé d'itérations, aucun ensemble $\mathcal{S}^*$ tel que $|\mathcal{S}^*| \geq T$ n'a été trouvé, soit on utilise l'ensemble de points $\mathcal{S}^*$ le plus grand trouvé au cours des itérations pour estimer le modèle $\mathcal{M}^*$, soit aucun modèle n'est retourné en sortie de l'algorithme.

Dans cette version la plus simple de l'algorithme de RANSAC, les trois paramètres τ, T, et N correspondent respectivement à la valeur seuil de tolérance sur l'erreur résiduelle au-dessus de laquelle un point est considéré comme aberrant pour le modèle testé, au nombre minimum de réalisations possibles requis pour affirmer qu'on vient d'estimer le modèle attendu et au nombre maximal d'itérations de l'algorithme. La proportion de points aberrants dans l'ensemble de points initial est rarement connue *a priori*. Lorsqu'une valeur approchée peut être fournie, on peut en déduire la probabilité qu'à une itération le sous-ensemble $\mathcal{S}$ ne contienne aucun point aberrant, et donc la probabilité que durant N itérations les sous-ensembles $\mathcal{S}$ ne soient jamais constitués que de points pertinents. On peut alors en déduire une valeur pertinente pour le paramètre N. Les valeurs des paramètres τ et T doivent par contre être fixées selon l'ensemble de données et les exigences spécifiques de l'application.

Dans une variante, le bon ajustement du modèle $\mathcal{M}^*$ est évalué en estimant l'erreur résiduelle moyenne des éléments de $\mathcal{S}^*$ par rapport au modèle, ce qui constitue une

étape 4. Les étapes 1 à 4 sont itérées N fois et au fil des itérations on conserve le modèle $\mathcal{M}^*$ associé à l'erreur résiduelle moyenne la plus faible. Moyennant l'ajout d'un paramètre supplémentaire, l'algorithme peut se terminer avant la N-ième itération si l'erreur résiduelle moyenne la plus faible est inférieure à un nouveau seuil fixé. Le modèle $\mathcal{M}^*$ retenu peut également être celui associé au CS de plus grande taille.

6.4.2 Variantes de l'algorithme de RANSAC pour l'estimation de plusieurs modèles

Concernant l'estimation de plusieurs modèles sur un même ensemble de données, l'algorithme itératif dit de « RANSAC séquentiel » permet à chaque étape d'ajuster un modèle au cluster dominant de la façon suivante :

1. Appliquer l'algorithme de RANSAC à l'ensemble initial de données, estimer le modèle d'intérêt et en déduire le cluster correspondant, i.e. l'ensemble de consensus du modèle d'intérêt. Toutes les données n'appartenant pas à ce cluster sont considérées comme « pseudo-aberrantes » (aberrantes par rapport au modèle d'intérêt mais plausibles pour un modèle différent).
2. Supprimer de l'ensemble des données celles correspondant au modèle d'intérêt et répéter l'étape 1 jusqu'à ce que le nombre de clusters désiré soit atteint, ou jusqu'à ce qu'il reste moins de d points dans l'ensemble de données.

Torr [Torr 98] considère des points d'intérêt détectés sur deux images éventuellement distantes. Les points sont appariés grâce à la corrélation croisée normalisée. Une approche basée sur l'algorithme de RANSAC séquentiel est utilisée pour déterminer de manière séquentielle l'ensemble des mouvements présents dans la scène. Chaque itération de l'algorithme permet l'estimation du modèle de mouvement dominant qui décrit au mieux les données.

L'inconvénient de cette approche itérative est que si des données sont détectées à tort comme réalisations possibles d'un modèle estimé à une itération, ces données sont supprimées de l'ensemble de données, ce qui contribue fortement à l'instabilité des estimations des modèles restants au cours des itérations suivantes.

Plusieurs approches ont été proposées pour remédier à ce problème. En particulier, la détection de modèles ainsi que le partitionnement peuvent être conduits de manière parallèle et non séquentielle.

Chaque itération de l'algorithme itératif dit de « multiRANSAC » [Zuliani 05] consiste à déterminer n modèles et leurs CS par l'algorithme de RANSAC séquentiel. À la fin d'une itération, les CS obtenus sont comparés aux CS obtenus à l'itération précédente. On ne conserve que les n CS disjoints et de plus grande taille. Cet algorithme a été appliqué à la détection de régions planaires par estimation d'homographies entre points d'une paire d'images.

Les travaux de Wills *et al.* [Wills 06] sont proches de ceux de Torr [Torr 98] dans la mesure où ils se basent également sur des correspondances de points issus de deux images distantes. Ils proposent une variante en 2 étapes de l'algorithme de RANSAC :

une étape de partitionnement et d'identification de modèles puis une étape d'estimation de modèles à partir de ce partitionnement.

La première étape consiste à exécuter un grand nombre N d'itérations de l'algorithme de RANSAC sans supprimer de données de manière à générer N modèles candidats. Plutôt que de retenir seulement le meilleur modèle, ces N modèles sont classés selon la taille de leurs ensembles de consensus. Les m (e.g., $m = 300$) modèles présentant les CS de plus grande taille sont conservés. Les modèles de mouvements réellement présents dans la scène ont vraisemblablement été détectés plusieurs fois, et on cherche donc à retrouver les copies de modèles parmi ces m candidats. Comparer les modèles dans l'espace des paramètres s'avère difficile. Ce sont donc leurs CS qui sont comparés. Si l'intersection de deux CS représente plus de 75% des données de ces deux CS réunis, alors le modèle avec le CS le plus grand est conservé, l'autre est supprimé.

À la seconde étape, pour chaque CS restant, on exécute l'algorithme de RANSAC avec $N = 100$ sur les données du CS comme ensemble de données initial. Lors de cette seconde étape, le paramètre τ est abaissé pour que les modèles estimés soient plus précis. Pour chaque CS, on retient cette fois uniquement le meilleur des modèles. Les copies de modèles sont à nouveau supprimées.

Si le nombre n de clusters réels augmente, la probabilité de tirer aléatoirement n MSSs composés de d données appartenant au même cluster diminue de façon exponentielle. Ce qui explique que dans la première étape N doit être très grand.

6.5 Approche de Pundlik et Birchfield

En réalité la méthode de Pundlik et Birchfield [Pundlik 08] ne repose pas directement sur des trajectoires mais sur des paires de points d'intérêt, le premier appartenant à une image de référence et le second étant le correspondant du premier dans l'image courante. Mais ces points sont tout de même suivis de manière séquentielle aussi longtemps que possible au long de la séquence, et la notion de trajectoire joue un rôle dans le maintien des groupes de points au cours du temps.

La particularité des travaux de Pundlik et Birchfield est d'estimer et partitionner un ensemble de trajectoires de points en temps réel. Le clustering se fait donc en ne connaissant que le passé, c'est-à-dire les « têtes » des trajectoires. Le nombre de clusters est inconnu et peut varier au cours de la séquence. Quelques images peuvent donc être nécessaires pour que le nombre de clusters détectés corresponde au nombre réel de clusters.

L'algorithme est initialisé en utilisant une paire d'images (pas nécessairement consécutives) et les paires de points d'intérêt détectés et suivis. Les clusters sont déterminés un par un par une méthode de croissance de région à partir d'un point non-traité sélectionné aléatoirement. L'idée est d'estimer un modèle affine de mouvement en s'appuyant sur le point sélectionné, ses voisins dans l'image de référence et leurs correspondants dans l'image courante. On fait ensuite grossir le cluster en regroupant petit à petit les voisins dont le déplacement correspond au modèle affine estimé. Le cluster est conservé si le nombre de points le constituant est supérieur à un seuil prédéfini. La procédure est

ensuite itérée à partir d'un autre point non-traité pour déterminer un second cluster, et ainsi de suite jusqu'à ce que tous les points aient été considérés. Enfin, un algorithme EM avec contraintes spatiales est appliqué pour tenir compte de l'hypothèse de continuité spatiale des régions.

Cette initialisation présente les mêmes inconvénients que les algorithmes de segmentation présentés en première partie de ce document et travaillant sur des paires d'images consécutives (section 1.2) ou distantes (section 1.3). Si les deux images sont trop proches l'une de l'autre, les objets se déplaçant lentement ne seront pas détectés. Si les deux images sont trop éloignées, l'hypothèse de mouvement affine risque d'être violée. Les résultats dépendent alors de la vitesse des différents objets constituant la scène.

Après l'étape d'initialisation, les différents groupes doivent être maintenus au cours du temps. Ceci est réalisé par une procédure incrémentale en 3 étapes. Premièrement, pour ajouter de nouveaux groupes aux groupes existants, la procédure d'initialisation par croissance de région est exécutée uniquement sur les points qui n'ont pas encore été groupés. Deuxièmement, l'algorithme d'EM avec contraintes spatiales est appliqué sur les points isolés restants pour qu'ils rejoignent les groupes existants. Troisièmement, on vérifie la cohérence de mouvement de chacun des groupes de points.

Pour cette troisième étape, si l'image de référence n'est jamais mise à jour, le nombre de points de l'image de référence ayant encore un correspondant dans l'image courante risque de décroître rapidement ce qui peut nuire à l'ajustement des modèles aux données. Si l'image de référence est mise à jour à une fréquence constante, indépendamment du comportement dynamique de chaque objet, à nouveau les résultats risquent de dépendre de la vitesse de ces objets.

L'originalité de la méthode est alors d'adapter la distance temporelle entre image de référence et image courante, pour chaque groupe, en mettant à jour l'image de référence d'un groupe lorsque le mouvement de ce groupe n'est plus jugé cohérent. La mesure de cette cohérence est réalisée à chaque image par un test χ^2. Différents groupes de points peuvent donc avoir des images de référence différentes. Si le mouvement d'un groupe est jugé incohérent, l'étape d'initialisation est appliquée aux points de ce groupe. On peut alors considérer deux possibilités : soit un seul groupe est à nouveau constitué et quelques points aberrants ne sont pas classés, auquel cas le mouvement du nouveau groupe épuré est cohérent et l'image de référence n'est pas mise à jour ; soit plusieurs groupes sont formés, ce qui correspond à la présence de plusieurs objets, auquel cas l'image de référence est mise à jour.

Notre contexte de travail ne nous impose pas de contraintes de temps réel. Par ailleurs nous avons déjà clairement justifié en introduction de cette partie notre choix d'intégrer les informations temporelles disponibles tout au long de la séquence avant de procéder au clustering de trajectoires de points. Nous ne chercherons donc pas à comparer nos travaux à ceux de Pundlik et Birchfield par la suite, mais leur méthode nous paraît réellement novatrice et le principe de mise à jour adaptée de l'image de référence selon l'objet considéré peut constituer une piste de recherches futures, par exemple pour l'élaboration d'un nouveau schéma séquentiel de segmentation proche

de notre second schéma basé sur les mosaïques de couches de référence, présenté en première partie.

6.6 Conclusions

Rappelons que nous formulons désormais notre problème de segmentation au sens du mouvement comme un problème de partitionnement d'un ensemble de trajectoires de points.

Parmi les approches hiérarchiques, les approches descendantes sont réputées spécialement coûteuses en temps de calcul et sont donc souvent écartées. Les approches ascendantes, quant à elles, reposent sur le calcul de distances entre chaque paire de trajectoires de points. Cette localité fait que les approches ascendantes existantes ne permettent que de traiter des mouvements de translation [Buzan 04] ou des mouvements parallèles au plan image [Brostow 06].

La distance DTW présente plus de souplesse que la distance euclidienne car elle permet de considérer des étirements temporels des trajectoires. La distance LCSS permet de surcroît de considérer des étirements spatiaux, ce qui la rend plus robuste au bruit. Ces étirements constituent des avantages réels pour des applications comme la synchronisation de séquences d'images [Dexter 09]. Néanmoins, d'une part, le calcul des distances DTW et LCSS demande bien plus d'opérations que le calcul de la distance euclidienne ; d'autre part, dans notre cas nous ne faisons aucune hypothèse de régularité ou de périodicité du mouvement, donc des appariements d'éléments associés à des instants différents ne sont pas réellement justifiés. Bien qu'elle ne permette pas de comparer directement deux trajectoires de longueurs différentes ou d'instants de début différents, la distance euclidienne peut être appliquée à l'intervalle temporel commun des deux trajectoires, ce qui permet alors de comparer les deux sous-trajectoires correspondantes.

Selon les conclusions de la comparaison de 4 algorithmes de factorisation réalisée par Tron et Vidal [Tron 07], l'algorithme MSL [Sugaya 04] est le plus précis mais est inutilisable en pratique car ses temps d'exécutions sont de plusieurs heures voire plusieurs jours sur les séquences testées. Les performances de l'algorithme GPCA [Vidal 04b] se dégradent dès que le nombre n de mouvements différents est supérieur à 2. L'algorithme LSA [Yan 06] est finalement le plus efficace à condition que les projections soient effectuées sur un sous-espace de dimension $D = 4n$.

Les approches de factorisation présentent néanmoins l'inconvénient majeur de ne pouvoir être appliquées qu'à un ensemble de données de même longueur. Dans le cas contraire, l'extrapolation des données manquantes ne peut être évitée.

Les approches basées sur l'algorithme de RANSAC présentent deux avantages : l'estimation robuste de modèles adaptés aux données, même en présence de bruit, et le rejet explicite des données aberrantes. En présence de plusieurs modèles, estimer les modèles les uns après les autres (tel qu'effectué par l'algorithme de RANSAC sé-

quentiel) s'avère bien moins efficace qu'estimer ces modèles en parallèle sans réduire progressivement l'ensemble de données initiales.

Dans le chapitre suivant, nous présentons la méthode que nous avons développée pour le partitionnement d'un ensemble de trajectoires de points de longueurs variables. Cette méthode s'appuie sur l'adaptation à nos trajectoires d'un algorithme combinant la robustesse de l'algorithme de RANSAC et la clarté d'interprétation des résultats des méthodes hiérarchiques. Un modèle de trajectoire affine et l'erreur résiduelle correspondante sont introduits. Quelques contraintes sont imposées pour traiter des trajectoires d'instants de début différents et de longueurs différentes sans avoir à extrapoler les données manquantes.

Chapitre 7

Clustering d'un ensemble de trajectoires de points de différentes longueurs

Ce chapitre présente notre troisième schéma de segmentation d'une séquence d'images au sens du mouvement. Contrairement aux schémas présentés en première partie de ce document, cette nouvelle approche ne fournit pas de cartes de segmentation dense. Nous nous intéressons cette fois au problème du clustering d'un ensemble épars de trajectoires de points. Chacune de ces trajectoires a ses propres instants de début et de fin, et donc sa propre longueur, en fonction des changements de pose et d'apparence de l'objet auquel elle appartient. Un tel ensemble de trajectoires est obtenu en suivant un ensemble épars de points d'intérêt sur différents intervalles temporels.

La section 7.1 propose une nouvelle formulation du problème et introduit ce que nous appelons un « modèle de trajectoire affine », ainsi que l'erreur résiduelle correspondante qui permet de traiter des trajectoires de longueurs différentes sans nécessiter d'extrapolation de données. L'algorithme de clustering est décrit à la section 7.2 et une discussion à propos du rejet explicite des données aberrantes est ébauchée. À la section 7.3, des résultats expérimentaux sur la base de données *Hopkins155* permettent une comparaison des performances de notre algorithme aux méthodes existantes pour le clustering d'un ensemble de trajectoires de points de même longueur. Nos résultats sur des ensembles de trajectoires de différentes longueurs sont montrés et discutés dans la section 7.4.

7.1 Nouvelle formulation du problème

Soit un ensemble de P points d'intérêt suivis au cours d'une séquence de F images. Notons $\mathbf{f}^{(i)} = \left[f_t^{(i)}\right]_{t=s(i)\cdots e(i)}$ la trajectoire du i-ème point d'intérêt ($i \in \{1 \cdots P\}$), où $s(i)$ et $e(i)$ sont ses instants de début et de fin, et $f_t^{(i)}$ correspond aux coordonnées (x, y) du point à l'instant t.

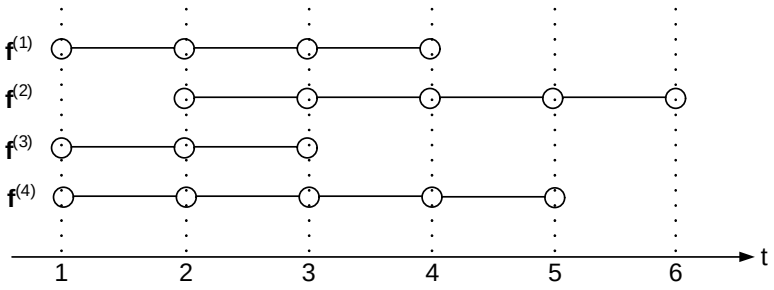


FIGURE 7.1 – Intervalle temporel commun à plusieurs trajectoires. Supposons que les trajectoires de points $\mathbf{f}^{(1)}$, $\mathbf{f}^{(2)}$, $\mathbf{f}^{(3)}$ appartiennent au même cluster. Nous ne pouvons estimer de modèle affine qu'entre les instants 2 et 3. Si la trajectoire $\mathbf{f}^{(4)}$ rejoint le cluster, la série temporelle de modèles affines peut maintenant être estimée/mise à jour entre les instants 1 et 4.

7.1.1 Modèle de trajectoire affine

Soit n le nombre clusters de mouvement à retrouver, correspondant généralement au nombre d'objets constituant la scène. Pour chaque cluster j, $j = 1, \cdots, n$, nous faisons l'hypothèse que son mouvement dans le plan image entre deux images consécutives est bien décrit par un modèle affine de mouvement. Nous introduisons alors le « modèle de trajectoire affine » et définissons celui-ci comme une série temporelle de paires de modèles de mouvement avant et arrière, affines à 6 paramètres :

$$\mathcal{M}^{(j)} = ((\mathcal{F}^{(j)}_{s(j)}, \mathcal{B}^{(j)}_{s(j)}), \cdots, (\mathcal{F}^{(j)}_{e(j)-1}, \mathcal{B}^{(j)}_{e(j)-1})) \ , \tag{7.1}$$

où $s(j)$ et $e(j)$ sont les premier et dernier instants auxquels l'objet j est visible (au moins partiellement), $\mathcal{F}^{(j)}_t$ est le modèle affine « avant » correspondant au mouvement du cluster j de l'instant t à l'instant $t+1$ et $\mathcal{B}^{(j)}_t$ est le modèle affine « arrière » correspondant au mouvement du cluster j de l'instant $t+1$ à l'instant t.

Notons que nous ne faisons aucune hypothèse de continuité ou périodicité du mouvement.

Étant donné l'ensemble de trajectoires $\mathbf{f}^{(i)}$, $i = 1, \cdots, P$ de différentes longueurs, le problème de segmentation au sens du mouvement consiste ici à partitionner l'ensemble de données en n clusters dont le mouvement est bien décrit par une série temporelle de modèles de mouvement affines, et à estimer tous les paramètres de mouvement de chaque cluster.

Puisque l'estimation d'un modèle affine nécessite de disposer d'au moins trois paires de points, nous considérons seulement les intervalles temporels communs à au moins trois trajectoires lorsque nous estimons un modèle à partir d'un ensemble de trajectoires (voir figure 7.1).

7.1.2 Erreur résiduelle de mouvement d'une trajectoire de longueur quelconque par rapport à un modèle affine de trajectoire

Dans la mesure où nous ne faisons pas d'hypothèse de continuité ou périodicité du mouvement, lors de la comparaison d'une trajectoire à un modèle de trajectoire affine, nous procédons à une comparaison « point à point » sans autoriser d'étirement temporel. Pour cela nous ne considérons que l'intervalle temporel commun à une trajectoire et à un modèle lorsque nous calculons l'erreur résiduelle de mouvement entre les deux. Cette erreur résiduelle entre trajectoire et modèle pourra également être appelée « distance au modèle » par la suite. Sur la figure 7.1, considérons qu'un modèle $\mathcal{M}$ est estimé à partir des seules trajectoires $\mathbf{f}^{(1)}$, $\mathbf{f}^{(2)}$ et $\mathbf{f}^{(3)}$. Ce modèle ne peut être défini qu'entre les instants 2 et 3 et l'erreur résiduelle entre la trajectoire $\mathbf{f}^{(4)}$ et $\mathcal{M}$ n'est calculée qu'entre ces instants.

Nous définissons l'erreur résiduelle de mouvement $r(\mathbf{f}, \mathcal{M})$ entre une trajectoire $\mathbf{f}$ d'instant de début $s(\mathbf{f})$ et d'instant de fin $e(\mathbf{f})$ et un modèle $\mathcal{M}$ d'instant de début $s(\mathcal{M})$ et d'instant de fin $e(\mathcal{M})$ comme suit :

$$r(\mathbf{f}, \mathcal{M}) = \begin{cases} \min_{t \in [s', e']} d(\mathbf{f}', \mathcal{M}'(\mathbf{f}', t)) & \text{si } e' - s' > 0 \\ +\infty & \text{sinon} \end{cases} , \tag{7.2}$$

où $\mathbf{f}'$ est la restriction temporelle de $\mathbf{f}$ à son intervalle temporel $\{s' \cdots e'\}$ commun avec $\mathcal{M}$, et où $s' = \max\big(s(\mathbf{f}), s(\mathcal{M})\big)$ et $e' = \min\big(e(\mathbf{f}), e(\mathcal{M})\big)$. $\mathcal{M}'(\mathbf{f}', t)$ est la trajectoire générée à partir des coordonnées de l'instant de référence t de la trajectoire $\mathbf{f}'$ en appliquant et accumulant les modèles affines de mouvement « avant » et « arrière » de $\mathcal{M}'$:

$$\begin{aligned} \mathcal{M}'(\mathbf{f}', t) = (\mathcal{B}_{s'}(\mathcal{B}_{s'+1}(\cdots \mathcal{B}_{t-1}(f_t)), \cdots, \mathcal{B}_{t-1}(f_t), \\ f_t, \mathcal{F}_t(f_t), \cdots, \mathcal{F}_{e'-1}(\mathcal{F}_{e'-2}(\cdots \mathcal{F}_t(f_t))) \ . \end{aligned} \tag{7.3}$$

La distance $d\big(\mathbf{f}', \mathcal{M}'(\mathbf{f}', t)\big)$ entre la trajectoire $\mathbf{f}'$ et sa transformée par le modèle $\mathcal{M}'$ est définie comme la distance géométrique moyenne :

$$d\big(\mathbf{f}', \mathcal{M}'(\mathbf{f}', t)\big) = \frac{\sqrt{\sum_{u=s'}^{e'} \|f_u - g_u\|^2}}{e' - s' + 1} , \tag{7.4}$$

où $\mathbf{g} = [g_u]_{u=s' \cdots e'}$ correspond à la trajectoire transformée $\mathcal{M}'(\mathbf{f}', t)$ pour alléger cette définition.

Pour déterminer l'adéquation entre une trajectoire et une série temporelle de modèles, nous appliquons et accumulons cette série temporelle de modèles aux coordonnées de la trajectoire correspondant à un instant de référence donné. Si ces coordonnées de référence sont erronées, l'accumulation des erreurs est inévitable, et la distance entre la trajectoire originale et la trajectoire synthétisée risque d'être forte, alors qu'un autre instant de référence aurait peut-être conduit à une distance plus faible. Pour cette raison, une minimisation est réalisée à l'équation (7.2). Pour rendre la distance au modèle

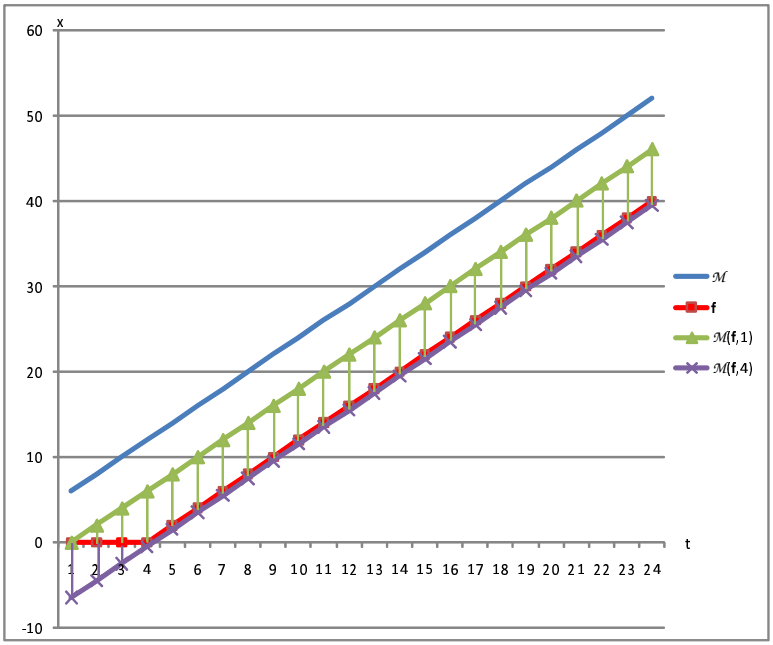


FIGURE 7.2 – Erreur résiduelle de mouvement entre le modèle $\mathcal{M}$ et la trajectoire $\mathbf{f}$, tous deux définis sur l'intervalle temporel $[1, 24]$. Sur cet exemple simple en dimension 1, $\mathcal{M}$ décrit le mouvement d'un objet en translation tel que $x_{t+1} = x_t + 2$. La trajectoire $\mathbf{f}$ appartient à cet objet mais les trois premières valeurs sont légèrement erronées. $d(\mathbf{f}, \mathcal{M}(\mathbf{f}, 1)) = 1.16$ et $d(\mathbf{f}, \mathcal{M}(\mathbf{f}, 4)) = 0.31$.

plus robuste, c'est la distance minimum entre la trajectoire considérée et chacune des trajectoires transformées par le modèle de trajectoire affine qui est retenue, plutôt que de se fier uniquement à la trajectoire transformée à partir des coordonnées (éventuellement erronées) de l'instant de début par exemple (voir figure 7.2).

7.2 Algorithme de clustering proposé

Les approches hiérarchiques ascendantes (comme par exemple [Buzan 04]) font émerger progressivement les clusters en parallèle. De telles méthodes requièrent la définition d'une distance portant seulement sur une paire d'observations parmi l'ensemble de données d'entrée. Selon la formulation précédente de notre problème, la définition d'une telle distance s'avère trop limitée : considérer deux trajectoires seulement permet d'estimer des modèles de mouvement avec au plus 4 paramètres. Les modèles plus complexes comme les modèles affines à 6 paramètres ne peuvent donc pas être utilisés dans la définition de la distance.

Les algorithmes séquentiels (comme RANSAC séquentiel) qui déterminent un cluster dominant à chaque étape ne sont pas concernés par le problème précédent. Mais l'inconvénient majeur est qu'une estimation imprécise du premier modèle peut empê-

cher la bonne détection des modèles suivants.

Récemment, l'algorithme dit de « J-linkage » [Toldo 08] a été présenté comme un algorithme de clustering hiérarchique perfectionné. Bien qu'il n'ait pas été appliqué à des trajectoires de points, cet algorithme semble être un outil séduisant pour traiter également notre problème. L'idée est de combiner la robustesse de l'algorithme de RANSAC pour l'estimation de modèle en présence d'observations aberrantes et pseudo-aberrantes, et la simplicité d'interprétation d'une approche hiérarchique ascendante. Le partitionnement n'a lieu ni dans l'espace des paramètres de modèles, ni dans l'espace des résidus, mais dans l'espace conceptuel où chaque observation est représentée par la fonction caractéristique de l'ensemble des modèles pour lesquels cette observation est une réalisation. Un avantage non négligeable mis également en avant est la détermination automatique du nombre de clusters.

En vue d'adapter cette technique à notre problème de segmentation au sens du mouvement, nous proposons d'utiliser l'erreur résiduelle de mouvement (équation (7.2)) définie à la section précédente pour une paire « trajectoire de point - modèle de trajectoire affine ». Nous proposons également une méthode pour détecter et rejeter explicitement les trajectoires aberrantes avant de lancer le processus d'agrégation.

7.2.1 Adaptation de l'algorithme de J-linkage

Comme l'estimation de chaque modèle affine de mouvement dans l'équation (7.1) requiert au moins trois paires de points correspondants à un instant donné, nous sélectionnons aléatoirement des sous-ensembles d'échantillons (« Minimal Sample Set »(MSS)) comprenant trois trajectoires de points.

Contrairement à [Toldo 08], durant cette étape de tirage aléatoire, toutes les trajectoires de l'ensemble à partitionner ont la même probabilité d'être sélectionnées. Certes, favoriser la sélection de données voisines augmenterait la probabilité qu'un MSS soit constitué uniquement de données appartenant à un même objet (si on fait l'hypothèse que les clusters sont cohérents spatialement), mais un modèle affine de mouvement est souvent plus précis s'il est estimé à partir de données spatialement distantes. Par exemple, un mouvement de rotation ne peut pas être estimé efficacement à partir des vecteurs de mouvement de trois points voisins si le centre de rotation est éloigné de ces points.

Pour gérer efficacement les différentes longueurs des trajectoires, nous imposons que chaque MSS généré vérifie la contrainte suivante, sans quoi il est définitivement rejeté : l'intervalle temporel commun aux trois trajectoires sélectionnées doit contenir au minimum deux instants (voir figure 7.1). En d'autres termes, pour que $\mathbf{f}^{(1)}$, $\mathbf{f}^{(2)}$ et $\mathbf{f}^{(3)}$ forment un MSS valide, on doit impérativement avoir :

$$\min\left(e(1), e(2), e(3)\right) - \max\left(s(1), s(2), s(3)\right) > 0 \ , \tag{7.5}$$

où $s(i)$ et $e(i)$ sont respectivement les instants de début et de fin de la trajectoire $\mathbf{f}^{(i)}$, avec $i = 1, \cdots, 3$

Nous générons M MSS remplissant cette condition. Un modèle de trajectoire affine (équation (7.1)) estimé en utilisant la méthode des moindres carrés est associé à chaque

MSS et son ensemble de consensus (« Consensus Set »(CS)) est calculé. Comme dans l'algorithme de RANSAC, le CS est composé des indices de toutes les trajectoires de points considérées comme des réalisations possibles du modèle, c'est-à-dire toutes les trajectoires de points dont la distance (équation (7.2)) au modèle de trajectoire affine considéré est inférieure à une valeur seuil λ_1.

Comme suggéré dans [Toldo 08], pour chaque trajectoire de point, nous construisons un ensemble de préférence (« Preference Set » (PS)) contenant les indices des modèles auxquels la trajectoire pourrait correspondre. Chaque trajectoire étant représentée par son PS, un clustering hiérarchique ascendant est effectué sur l'ensemble des PS. À l'initialisation, chaque PS constitue un cluster singleton. Puis l'algorithme regroupe successivement les deux clusters les plus proches, en utilisant la distance de Jaccard (équation (7.6)) définie pour une paire de PS. Après chaque regroupement, le PS du cluster résultant est calculé comme étant l'intersection des PS des deux clusters regroupés.

Étant donné deux PS $\mathcal{Q}_1$ et $\mathcal{Q}_2 \subset \{1 \cdots M\}$, la distance de Jaccard mesure la dissemblance entre les deux ensembles de la manière suivante :

$$d_J(\mathcal{Q}_1, \mathcal{Q}_2) = \frac{|\mathcal{Q}_1 \cup \mathcal{Q}_2| - |\mathcal{Q}_1 \cap \mathcal{Q}_2|}{|\mathcal{Q}_1 \cup \mathcal{Q}_2|} \ , \tag{7.6}$$

où $|.|$ désigne le cardinal d'un ensemble.

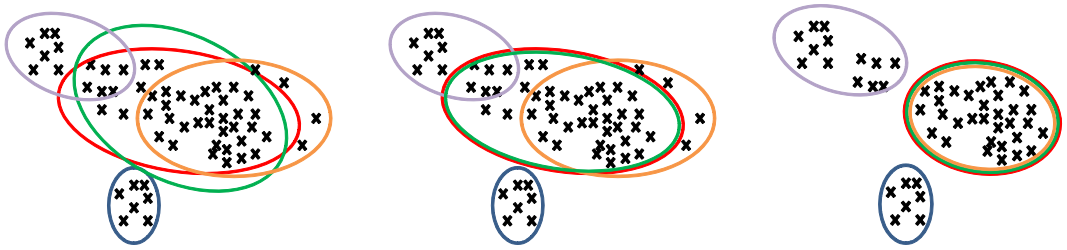
Les regroupements se poursuivent tant que la distance entre les deux clusters les plus proches est strictement inférieure à 1. Ainsi, des clusters dont les PS sont disjoints ne peuvent pas être regroupés. Soulignons que l'utilisateur n'a pas besoin de fournir le nombre de clusters à extraire, ni d'ajuster cette valeur limite. Un exemple de regroupements selon l'algorithme de J-linkage est donné à la figure 7.3.

Pour chaque cluster obtenu, le modèle est ensuite mis à jour à partir des trajectoires appartenant au cluster et en utilisant la méthode des moindres carrés. Enfin, chaque trajectoire de point est réassignée au cluster le plus proche au sens de la distance au modèle définie à l'équation (7.2). Au cours de la mise à jour des modèles, chaque modèle peut être étendu à toutes les paires d'instants consécutifs pour lesquelles nous disposons d'au moins trois trajectoires (voir figure 7.1).

7.2.2 Rejet des trajectoires aberrantes

Dans [Toldo 08], les auteurs supposent que les données aberrantes constituent des clusters de petite taille et ils proposent donc de rejeter les clusters un par un en commençant par le plus petit, jusqu'à ce que le nombre de données rejetées corresponde au nombre de données aberrantes. Cela n'est possible que si l'on dispose d'une estimation de ce nombre. Cependant, dans notre application, la proportion de trajectoires aberrantes contenues dans l'ensemble initial de trajectoires n'est pas connue et est difficile à estimer. On pourrait penser rejeter les clusters dont le cardinal est inférieur à un seuil.

Il peut arriver malgré tout en pratique qu'une donnée aberrante soit contenue dans un grand cluster. C'est le cas si le PS de cette donnée aberrante et le PS du cluster ne sont pas disjoints, ce qui correspond à une distance de Jaccard inférieure à 1. Ce



(a) (b) (c)

FIGURE 7.3 – Exemple d'agrégations par l'algorithme de J-linkage. N.B. : Chaque croix représente un modèle affine de trajectoire et non pas une trajectoire. 7.3a On dispose initialement de 5 ensembles de préférences (PS) de couleurs différentes (rouge, vert, bleu, orange, violet), chacun correspondant à un cluster de trajectoires (ou à une trajectoire) différent(e). Le PS bleu est disjoint des 4 autres et le cluster qui lui correspond ne peut donc pas être fusionné avec un autre cluster. Les PS rouge et vert sont égaux, leur distance de Jaccard est donc nulle. Les clusters correspondant sont fusionnés les premiers et le PS résultant est indiqué par le double contour rouge/vert 7.3b. Les deux PS non disjoints les plus proches sont alors le PS orange et le PS rouge/vert. 7.3c Le PS résultant de la fusion des deux clusters correspondants est l'intersection des deux PS, indiquée par le triple contour rouge/vert/orange. Les modèles n'appartenant plus à aucun PS sont supprimés. Il n'y a plus de paire de clusters non disjoints. L'algorithme a déterminé automatiquement 3 clusters de trajectoires, en l'occurrence {rouge,vert,orange}, {violet} et {bleu}.

cas est rencontré dès qu'il existe un modèle (un seul suffit) qui correspond à tous les éléments du cluster mais également à la donnée aberrante. Généralement, ce type de regroupement intervient vers la fin du processus d'agrégation. Une valeur limite inférieure à 1 pourrait éviter de tels regroupements non désirés. Cependant certains autres regroupements, corrects cette fois, deviendraient impossibles, ce qui conduirait en quelque sorte à une sur-segmentation.

Au cours du processus d'agrégation, alors que les regroupements se succèdent, le cardinal des PS diminue (en fait les PS sont progressivement réduits aux indices des modèles qui correspondent à toutes les trajectoires de points comprises dans le cluster considéré) et les distances de Jaccard entre les clusters restant augmentent. Or, la plupart des données aberrantes peuvent être identifiées comme ayant un PS de petite taille ou un PS dont les distances de Jaccard aux PS des autres données sont larges. Nous préférons donc les rejeter avant même de lancer le clustering hiérarchique, en détectant qu'il n'y a pas d'autre PS dont la distance au PS de la trajectoire considérée est inférieure à une valeur seuil λ_2.

L'algorithme complet de clustering est résumé ci-après (algorithme 4).

Algorithme 4 Algorithme de clustering proposé

Entrées: P trajectoires $\mathbf{f}^{(i)} = \left[f^{(i)}_{s(i):e(i)}\right], i = 1 \cdots P$

GÉNÉRATION DES MODÈLES HYPOTHÈSES

$m \leftarrow 1$
tantque $m <= M$ **faire** {Itération sur les modèles}
  Sélectionner aléatoirement 3 trajectoires $\mathbf{f}^{(i)}$, $\mathbf{f}^{(j)}$ et $\mathbf{f}^{(k)}$
  si l'équation (7.5) est vérifiée **alors**
    Estimer le modèle de trajectoire affine $\mathcal{M}_m$
    pour p de 1 à P **faire** {Boucle sur les trajectoires}
      si $r(\mathbf{f}^{(p)}, \mathcal{M}_m) < \lambda_1$ **alors**
        Ajouter p à $\mathcal{C}_m$ l'ensemble de consensus (CS) de $\mathcal{M}_m$ ($\mathcal{C}_m \leftarrow \mathcal{C}_m \cup \{p\}$)
        Ajouter m à $\mathcal{Q}_p$ l'ensemble de préférence (PS) de $\mathbf{f}^{(p)}$ ($\mathcal{Q}_p \leftarrow \mathcal{Q}_p \cup \{m\}$)
      fin si
    fin pour
    $m \leftarrow m + 1$
  fin si
fin tantque

REJET DES TRAJECTOIRES ABERRANTES

pour p de 1 à P **faire** {Boucle sur les trajectoires}
  si $\forall q \in [1, P] \setminus \{p\}$, $d_J(\mathcal{Q}_p, \mathcal{Q}_q) > \lambda_2$ **alors**
    Considérer la trajectoire $\mathbf{f}^{(p)}$ comme aberrante et la rejeter
  fin si
fin pour

CLUSTERING

Construire un cluster singleton par PS de trajectoire restante
tantque la distance entre les 2 clusters les plus proches est strictement inférieure à 1 **faire**
  Regrouper ces deux clusters
  Construire le PS du cluster résultant comme l'intersection des PS des deux clusters regroupés
fin tantque
Pour chaque cluster obtenu, mettre à jour le modèle de trajectoire affine
Réassigner chaque trajectoire au cluster le plus proche au sens de la distance au modèle (équation (7.2))

7.3 Résultats expérimentaux sur la base de données *Hopkins155*

La base de données *Hopkins155* présentée dans [Tron 07] (et disponible à l'adresse `http://vision.jhu.edu/data/hopkins155/`) comprend 50 séquences d'images représentant des scènes intérieures ou extérieures et contenant deux ou trois groupes de mouvement. Pour chaque séquence contenant trois groupes de mouvement, en plus de traiter le problème du partitionnement en 3 clusters, on traite également les 3 problèmes binaires subordonnés. Pour chacune de ces séquences, la base de données comprend donc trois ensembles de trajectoires supplémentaires, dérivés de l'ensemble original, qui comprennent les trajectoires de seulement deux des trois groupes de mouvements.

En fonction du contenu des séquences et du type de mouvement, on distingue trois catégories de séquences :

- les séquences extérieures de *trafic*, montrant des véhicules en mouvement dans une rue ou sur un parking,
- les séquences de mouvement *articulé* ou non-rigide, montrant des mouvements connectés par des articulations (personne bougeant la tête, piétons marchant dans la rue, engins de chantier articulés),
- les séquences intérieures de *mires*, où des motifs en damier sont collés sur des objets en mouvement de façon à faciliter le suivi de nombreux points.

Notre modélisation affine du mouvement n'est pas appropriée pour traiter les séquences de cette dernière catégorie, car elles contiennent entre autres d'importantes rotations en dehors du plan image. Pour ces séquences, nous avons obtenu fréquemment plus de trois clusters, ce qui nous empêche de comparer nos résultats à la vérité de terrain fournie.

Tous les résultats mentionnés ci-après portent donc sur les 51 ensembles de trajectoires associés aux 24 séquences des catégories *trafic* et *articulé*. Par abus de langage nous parlerons de 51 séquences par la suite. Toutes les trajectoires fournies sont complètes, donc toutes de même longueur, et il n'y a pas de trajectoires aberrantes. Le nombre moyen de points suivis et le nombre moyen d'images par séquences sont donnés dans le tableau 7.1.

Un aperçu de quelques-unes des séquences de *trafic* et de mouvement *articulé* est donné à la figure 7.4.

	2 clusters			3 clusters		
	# Séquences	Points	Images	# Séquences	Points	Images
Trafic	31	241	30	7	332	31
Articulé	11	155	40	2	122	31
Ensemble	42	218	33	9	285	31
Répartition	74% - 26%			67% - 19% - 14%		

TABLE 7.1 – Nombre de séquences, nombre moyen de points, nombre moyen d'images et répartition moyenne des points dans les différents clusters.

FIGURE 7.4 – Aperçu de la base de données *Hopkins155*. Images représentatives de séquences des catégories *trafic* (à gauche) et *articulé* (à droite). Les points suivis sont superposés.

Pour toutes nos expérimentations sur cette base de données, nous avons fixé $M = 600$ et $\lambda_2 = 1$. Dans les méthodes de factorisation déjà évaluées et comparées dans [Tron 07, Rao 08], le nombre de clusters à extraire est donné par l'opérateur. Dans notre méthode, l'opérateur n'a pas besoin de fournir cette information. Mais pour rendre possible la comparaison de nos résultats avec les résultats des méthodes de factorisation, nous avons déterminé, moyennant de rapides réglages manuels sur peu de séquences, 6 valeurs de λ_1 (le paramètre intrinsèque de RANSAC) qui ont donné à peu près les mêmes nombres de clusters que ceux indiqués par les vérités de terrain. Puis, pour chacune des 51 séquences et avec chacune des 6 valeurs de λ_1, nous avons exécuté 100 fois l'algorithme proposé. Cette démarche est choisie lors de l'évaluation des performances pour atténuer l'effet de la nature aléatoire de notre approche.

Dans le tableau 7.2, nous indiquons les taux moyens et médians d'erreur de classification, ainsi que les temps de calcul moyens de notre algorithme. Pour chaque séquence, ces statistiques ont été établies à partir des valeurs de λ_1 pour lesquelles la comparaison à la vérité de terrain a été possible. Les statistiques correspondant aux autres algorithmes viennent de [Tron 07, Rao 08].

Le taux d'erreur de classification est défini par :

$$\text{taux d'erreur} = \frac{\text{nb de points mal classés}}{\text{nb total de points}} \quad . \tag{7.7}$$

Le tableau 7.2 montre que sur l'ensemble des séquences de test et parmi les 8 algorithmes comparés, notre méthode est en moyenne la deuxième la plus précise, et la troisième la plus rapide. La méthode MSL est la plus précise mais est très lente. Les deux méthodes les plus rapides (GPCA et RANSAC) sont un peu moins précises que la nôtre en moyenne, et bien moins précises sur les séquences *Trafic* à trois groupes de mouvements.

Méthode	GPCA	RANSAC	MSL	LSA 5	LSA $4n$	ALC_5	ALC_{sp}	Proposée
Trafic (2 mouvements) (31 séquences)								
Moyenne	**1.41%**	2.55%	2.23%	2.15%	5.43%	2.58%	1.59%	1.92%
Médiane	**0.00%**	0.21%	**0.00%**	1.00%	1.48%	0.25%	1.17%	0.01%
Articulé (2 mouvements) (11 séquences)								
Moyenne	**2.88%**	7.25%	7.23%	4.66%	4.10%	6.90%	10.70%	5.38%
Médiane	**0.00%**	2.64%	**0.00%**	1.28%	1.22%	0.88%	0.95%	0.02%
Trafic (3 mouvements) (7 séquences)								
Moyenne	19.83%	12.83%	**1.80%**	27.02%	25.07%	3.52%	7.75%	4.89%
Médiane	19.55%	11.45%	**0.00%**	34.01%	23.79%	1.15%	0.49%	0.19%
Articulé (3 mouvements) (2 séquences)								
Moyenne	16.85%	21.38%	**2.71%**	23.11%	7.25%	7.25%	21.08%	20.41%
Médiane	16.85%	21.38%	2.71%	23.11%	7.25%	7.25%	21.08%	20.41%
Ensemble 2 mouvements (42 séquences)								
Moyenne	**1.80%**	3.78%	3.54%	2.81%	5.08%	3.71%	3.98%	2.83%
CPU	271ms	**138ms**	18h29m	5.744s	6.372s	n.c.	n.c.	4.457s
Ensemble 3 mouvements (9 séquences)								
Moyenne	19.17%	14.73%	**2.00%**	26.15%	21.11%	4.35%	10.71%	8.34%
CPU	439ms	**180ms**	24h54m	8.268s	10.293s	n.c.	n.c.	7.026s
Ensemble : 51 séquences								
Moyenne	4.86%	5.71%	**3.27%**	6.93%	7.91%	3.99%	5.56%	3.80%
CPU	301ms	**146ms**	19h36m	6.513s	7.064s	7m43s	15m38s	5.734s

TABLE 7.2 – Taux d'erreur de classification et temps de calcul moyens sur les séquences des catégories *Trafic* et *Articulé* de la base de données *Hopkins155*. GPCA : [Vidal 04b] ; RANSAC : RANSAC appliqué à l'ajustement de sous-espaces de dimension 4, tel que décrit dans [Tron 07] ; MSL : [Sugaya 04] ; LSA : [Yan 06] ; ALC : [Rao 08].

Nos résultats ont été obtenus sur un Pentium IV à 3,6GHz et 3GB de RAM avec un code C++ non optimisé. Toutes les expérimentations de [Tron 07] ont été réalisées sur un Intel Xeon MP à 8 processeurs à 3,66GHz et 32GB de RAM, et pour chaque simulation, chaque algorithme a exploité seulement un processeur, sans parallélisme.

7.4 Résultats expérimentaux sur des ensembles de trajectoires de différentes longueurs

Une des caractéristiques remarquables de notre approche étant sa capacité à traiter des ensembles de trajectoires de différentes longueurs et de différents instants de début, nous avons testé notre algorithme sur de tels ensembles.

Pour les séquences *Cars* et *Hand* (disponibles à l'adresse suivante : http://rvsn.csail.mit.edu/pv/), nous avons utilisé les trajectoires estimées par Sand et Teller [Sand 08] (également disponibles à la même adresse). Pour les autres séquences, les trajectoires ont été estimées en utilisant l'implémentation de Birchfield [Birchfield 07] de l'algorithme de suivi de points d'intérêt de Kanade, Lucas et Tomasi (KLT) [Lucas 81, Tomasi 91, Shi 94], avec remplacement des points perdus de manière à obtenir des trajectoires débutant après la première image.

En utilisant l'algorithme de suivi KLT, il est fréquent d'obtenir des trajectoires très courtes correspondant à des points rapidement perdus (par exemple à cause d'occultations, de rotations en dehors du plan image, de déformations, de parasites). Par ailleurs, quand on demande à l'algorithme KLT de détecter et suivre un grand nombre de points, une partie d'entre eux peut être située dans des régions uniformes et risque d'engendrer des trajectoires instables. Comme notre algorithme s'appuie uniquement sur l'information de mouvement, il n'est pas pertinent de procéder au clustering d'un ensemble de trajectoires contenant de telles trajectoires courtes et/ou non fiables.

En conséquence, et afin d'évaluer les possibilités réelles de notre système en l'absence de ces trajectoires trop gênantes, nous avons seulement retenu les P trajectoires présentant les plus forts gradients d'intensité à leur instant de début et dont la longueur est supérieure à un minimum donné $l_{\min}$.

Précisons néanmoins que les trajectoires de points mises à l'écart peuvent être classées après l'étape de clustering. La classification de chaque trajectoire se fait alors par assignation au cluster le plus proche au sens de la distance au modèle définie à l'équation (7.2), comme démontré à la section 7.5.

Pour toutes les séquences, exceptée la séquence *Carmap*, nous avons fixé $l_{\min}$ à une valeur suffisamment grande pour que tout MSS tiré aléatoirement vérifie l'équation (7.5), ce qui nous place dans une situation favorable puisque nous ne perdons pas de temps à tirer des MSS pour lesquels aucun modèle de trajectoire affine ne peut être estimé.

Pour la séquence *Carmap*, nous avons fixé $l_{\min}$ à une valeur plus faible pour que les trajectoires plus courtes correspondant à la voiture avant et après l'occultation derrière le panneau ne soient pas écartées.

Pour chacune des séquences, nous présentons les résultats du clustering avant la mise à jour des modèles et l'assignation finale après cette mise à jour, sur quelques images. Ceci explique pourquoi les résultats de « clustering » et d'« assignation finale » peuvent être légèrement différents.

Dans la séquence *Cars* (figure 7.5), il est intéressant de noter que malgré leur faible mouvement apparent, les quelques trajectoires correspondant à la voiture la plus éloi-

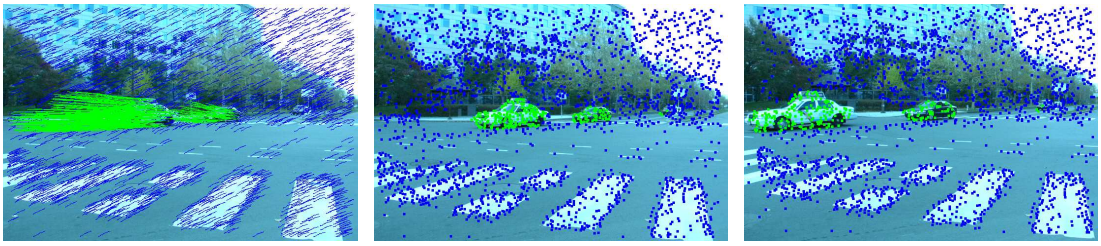


FIGURE 7.5 – Séquence *Cars*. (à gauche) Clustering automatique des trajectoires ; (au milieu, à droite) classification finale montrée sur les points des trajectoires présentes aux images 1 et 26 respectivement.

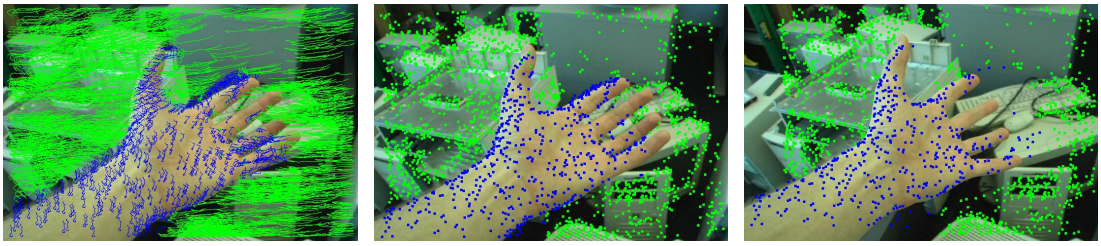


FIGURE 7.6 – Séquence *Hand*. (à gauche) Clustering automatique des trajectoires ; (au milieu, à droite) classification finale montrée sur les points des trajectoires présentes aux images 1 et 36 respectivement.

gnée ont rejoint le cluster des deux autres voitures. Notre système étant basé uniquement sur un critère de mouvement, le but n'est pas la détection d'entités individuelles. L'obtention d'un cluster indépendant pour chacune des trois voitures nécessiterait l'ajout d'un *a priori* spatial.

La figure 7.6 montre de bons résultats sur les mouvements complexes de la séquence *Hand*. Les erreurs de classification correspondent principalement à des trajectoires qui sautent de la main à l'arrière-plan (ou inversement) durant une occultation (ou une désoccultation). De telles trajectoires ne sont pas détectées comme aberrantes et ne sont donc pas rejetées car elles correspondent très bien aux modèles de mouvement estimés pendant une bonne partie des intervalles temporels considérés.

Contrairement à [Rao 08], notre système ne requiert pas de disposer pour chaque objet de plusieurs trajectoires complètes. Notre algorithme est donc capable de traiter des objets sujets à des occultations partielles telles qu'aucun point n'est visible sur l'intégralité de la séquence. La figure 7.7 illustre le cas d'un tel objet : la voiture de la séquence *Carmap*. La voiture est partiellement occultée par le panneau et tous ses points disparaissent tour à tour durant plusieurs instants consécutifs. Le fait que l'avant de la voiture réapparaisse avant que l'arrière ne disparaisse permet de grouper en un seul cluster les trajectoires correspondant à la voiture avant et après l'occultation, ceci sans avoir à compléter la moindre trajectoire.

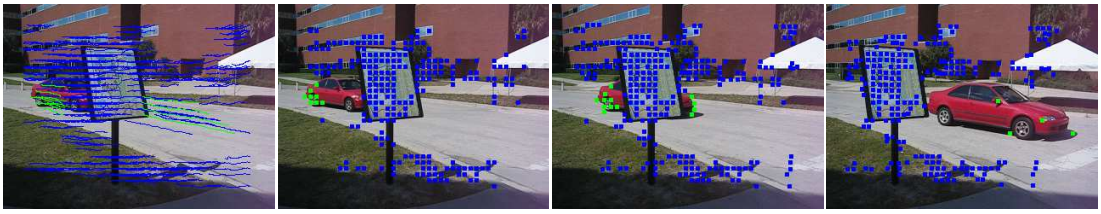


FIGURE 7.7 – Séquence *Carmap*. (de gauche à droite) Clustering automatique des trajectoires ; assignation finale montrée sur les points des trajectoires présentes aux images 1, 12 et 34 respectivement.

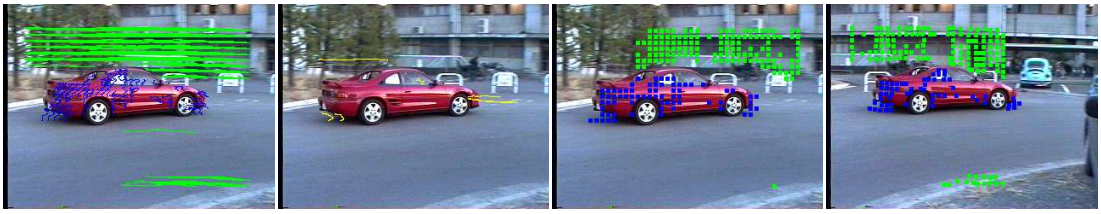


FIGURE 7.8 – Séquence *Kanatani2*. (de gauche à droite) Clustering automatique des trajectoires ; Trajectoires détectées comme aberrantes et rejetées ; assignation finale montrée sur les points des trajectoires présentes aux images 1 et 16 respectivement.

Imaginons la même scène mais avec un panneau deux ou trois fois plus large, tel qu'il occulte totalement la voiture à au moins un instant. Nous atteignons alors les limites de l'algorithme puisqu'aucun modèle de trajectoire affine ne pourra être estimé sur un intervalle temporel contenant cet instant. Nous obtiendrons donc deux clusters différents pour la voiture, l'un contenant les trajectoires de points avant l'occultation, l'autre contenant les trajectoires de points après l'occultation.

La figure 7.8 montre nos résultats sur la séquence *Kanatani2* en utilisant 300 trajectoires de différentes longueurs à la place des 63 trajectoires disponibles dans la base de données *Hopkins155*. Sept trajectoires aberrantes ont été détectées et rejetées. En moyenne, ces trajectoires avaient donné consensus à seulement 4% des modèles estimés à partir des MSS.

Seulement une trajectoire aberrante a été détectée dans la séquence *Coastguarder* (figure 7.9). Cette trajectoire correspond à un point de la surface de l'eau pour lequel le suivi a échoué sans pour autant que le point ne soit considéré comme perdu. La longueur de cette trajectoire supérieure à $l_{\min}$, ainsi que le fort gradient spatial d'intensité sur son point de début explique qu'elle n'ait pas été mise à l'écart lors de la sélection des P trajectoires jugées les plus pertinentes. Lors de son rejet avant le clustering, cette trajectoire avait donné consensus à seulement 4% des modèles.

Nous avons implémenté l'approche hiérarchique ascendante [Buzan 04] pour comparer ses résultats à ceux obtenus avec notre méthode. Comme la nôtre, l'approche de

FIGURE 7.9 – Séquence *Coastguarder*. (de gauche à droite) Clustering automatique des trajectoires ; Trajectoires détectées comme aberrantes et rejetées ; assignation finale montrée sur les points des trajectoires présentes aux images 1, 41 et 81.

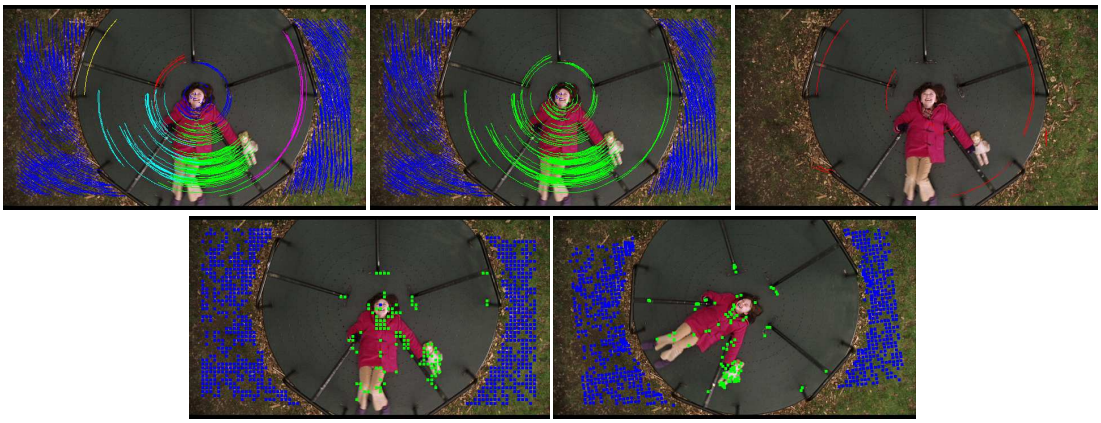


FIGURE 7.10 – Séquence *Rotation*. Première ligne : (à gauche) méthode hiérarchique ascendante [Buzan 04] ; (au milieu) notre méthode ; (à droite) Trajectoires détectées comme aberrantes et rejetées par notre algorithme. Deuxième ligne : assignation finale montrée sur les points des trajectoires présentes aux images 1 et 22 respectivement.

Buzan *et al.* traite des ensembles de trajectoires de longueurs différentes sans nécessiter l'extrapolation des données manquantes des trajectoires incomplètes. La figure 7.10 montre que cette méthode n'a pas permis de détecter et estimer des mouvements de rotation mais elle a regroupé les trajectoires comparables via un mouvement de translation. Au contraire, notre approche a détecté et estimé de manière satisfaisante ces mouvements de rotation.

Pour chaque séquence, nous indiquons dans le tableau 7.3 les valeurs des paramètres que nous avons utilisées et :

- le nombre d'images F,
- la taille P de l'ensemble de trajectoires retenues pour le clustering,
- la longueur $l_{\min}$ de la trajectoire la plus courte présente dans l'ensemble de trajectoires retenues pour le clustering,
- le pourcentage m de données manquantes dans l'ensemble de trajectoires retenues :

$$m = \left(1 - \frac{\sum_{i=1}^{P} \left(e(i) - s(i) + 1\right)}{P \times F}\right) \times 100, \tag{7.8}$$

séquence	F	P	$l_{\min}$	m	λ_1	λ_2	M	#rejets	CPU
Cars	26	2505	14	2.61%	3	0.7	600	0	112.312s
Hand	36	2285	20	6.95%	3	0.7	600	0	114.406s
Carmap	36	300	13	15.83%	1	0.7	1000	0	7.516s
Kanatani2	17	300	10	15.51%	1	0.7	600	7	1.859s
Coastguarder	81	221	50	17.76%	1	0.7	600	1	9.186s
Rotation	22	1000	15	4.34%	1	0.7	2000	9	63.796s

TABLE 7.3 – Quelques caractéristiques des séquences et des ensembles de trajectoires utilisés dans nos expérimentations, et les paramètres utilisés pour effectuer le clustering des trajectoires (voir dans le texte).

- le nombre de trajectoires détectées comme aberrantes et rejetées (#rejets),
- le temps de calcul (estimation des trajectoires non incluse) étant donné que toutes les expérimentations ont été menées sur un Pentium IV à 3,6GHz et 3GB RAM avec un code en C++ non optimisé.

Discussion sur le réglage des différents paramètres

Le nombre de trajectoires P et leur longueur minimum $l_{\min}$ ne sont pas des paramètres propres à l'algorithme de clustering. Ils permettent simplement de sélectionner parmi toutes les trajectoires de points suivis celles qui semblent les plus significatives.

Le nombre M de MSS valides à sélectionner aléatoirement est relié à la répartition des trajectoires dans les différents clusters et au pourcentage de trajectoires aberrantes. Ces données sont inconnues et difficiles à estimer dans notre cas. Ce nombre M devant être suffisamment grand pour qu'un certain nombre (au moins) de MSS ne contiennent pas de trajectoire aberrante ou pseudo-aberrante, une solution simple aurait pu être de lui donner la même valeur extrêmement élevée pour toutes les séquences. Ceci mènerait cependant à des traitements très coûteux en temps de calcul. C'est la raison pour laquelle nous avons préféré adapter la valeur du paramètre M selon la complexité de la scène, ce qui explique pourquoi cette valeur est plus élevée pour les séquences *Carmap* (occultation importante) et *Rotation*.

Le paramètre λ_2 permet la détection et le rejet de trajectoires aberrantes. Nous avons fixé $\lambda_2 = 0.7$ de manière empirique pour toutes nos expérimentations.

Le paramètre λ_1 est le paramètre de la distance au modèle, intrinsèque à l'algorithme de RANSAC. Pour un modèle donné, il permet la distinction entre les observations qui sont considérées comme des réalisations du modèle et celles qui ne le sont pas. Le paramètre λ_1 doit donc être réglé selon la complexité des séquences en fonction de la bonne description ou non des mouvements par des séries temporelles de modèles affines.

séquence	F	P	$l_{\min}$	m
Carmap	36	2450	2	54.66%
Kanatani2	17	3251	2	69.79%
Coastguarder	81	6911	2	84.49%
Rotation	22	12045	2	48.25%

TABLE 7.4 – Quelques caractéristiques (voir dans le texte, section précédente) des séquences et des ensembles de trajectoires utilisés dans nos expérimentations de classification.

7.5 Classification *a posteriori* des trajectoires mises à l'écart avant le clustering

Certaines trajectoires de points sont mises à l'écart avant le clustering, soit parce qu'elles sont trop courtes, soit parce qu'elles sont associées à des points de début qui sont les centres de patchs non suffisamment texturés. Ces trajectoires peuvent néanmoins être classées après l'étape de clustering d'une manière simple : chacune d'elles est associée au cluster le plus proche au sens de la distance au modèle définie à l'équation (7.2). Les figures 7.11 à 7.14 montrent de tels résultats de classification reposant uniquement sur le critère lié au mouvement. Toutes les trajectoires estimées sont classées, des trajectoires constituées de deux points seulement ou dont le premier point est associé à un très faible gradient aux trajectoires dont le premier point associé à un fort gradient a pu être suivi durant toute la séquence. Le tableau 7.4 montre que le pourcentage m de données manquantes (équation (7.8)) dans ces ensembles de trajectoires est bien supérieur à celui calculé sur les ensembles servant au clustering.

Les erreurs de classification apparaissent principalement dans les régions homogènes. De telles erreurs pourraient probablement être évitées en utilisant un *a priori* spatial ou en ajoutant un critère de couleur.

Mais dans la mesure où les différentes positions traversées par une trajectoire sont liées entre elles et doivent porter une unique étiquette, si le critère lié au mouvement n'a pas permis une bonne classification de ces trajectoires, la question du bon suivi des points correspondants doit être posée en priorité. Nos travaux ne portant par sur l'estimation de trajectoires de points mais sur leur clustering, les modèles de mouvement estimés pour chaque cluster peuvent être utilisés pour « redresser » certaines de ces trajectoires.

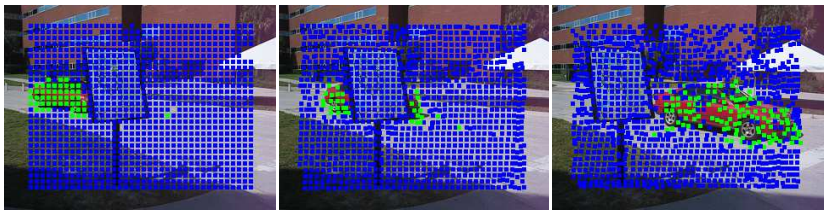


FIGURE 7.11 – Séquence *Carmap*. Classification des 2450 trajectoires estimées dont les longueurs varient entre 2 et 36. (de gauche à droite) Assignation finale montrée sur les points des trajectoires présentes aux images 1, 12 et 34 respectivement.

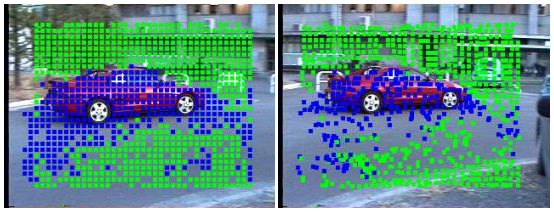


FIGURE 7.12 – Séquence *Kanatani2*. Classification des 3251 trajectoires estimées dont les longueurs varient entre 2 et 17. (de gauche à droite) Assignation finale montrée sur les points des trajectoires présentes aux images 1 et 17 respectivement.

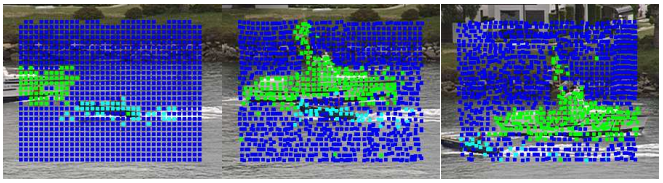


FIGURE 7.13 – Séquence *Coastguarder*. Classification des 6911 trajectoires estimées dont les longueurs varient entre 2 et 81. (de gauche à droite) Assignation finale montrée sur les points des trajectoires présentes aux images 1, 41 et 81 respectivement.

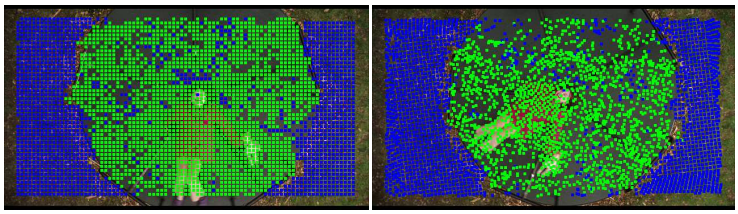


FIGURE 7.14 – Séquence *Rotation*. Classification des 12045 trajectoires estimées dont les longueurs varient entre 2 et 22. (de gauche à droite) Assignation finale montrée sur les points des trajectoires présentes aux images 1 et 22 respectivement.

Conclusion de la deuxième partie

Dans cette seconde partie nous nous sommes concentrés sur le problème du clustering d'un ensemble épars de trajectoires de points. Dans cette autre formulation du problème de segmentation au sens du mouvement, les trajectoires de points émanent de la détection de points d'intérêt et de leur suivi durant un intervalle temporel propre à chaque point.

Nous avons présenté un nouvel algorithme pour le clustering d'un ensemble de trajectoires de différentes longueurs. Nous avons proposé de modéliser les mouvements par des séries temporelles de modèles affines estimés entre instants consécutifs. L'estimation de modèles de mouvement, tout comme le calcul de l'erreur résiduelle de mouvement entre une trajectoire et un modèle, tient compte des différents intervalles temporels sur lesquels les trajectoires sont définies.

L'algorithme présenté se base exclusivement sur l'information de mouvement. Les résultats expérimentaux montrent l'efficacité de notre méthode à partitionner des ensembles de trajectoires de points sans nécessiter l'extrapolation des trajectoires incomplètes, en particulier celles qui correspondent à des points quittant le champ de vision ou n'étant pas visibles sur certaines images car étant occultés.

Lors de l'occultation d'une zone d'arrière-plan par un objet d'avant-plan, on s'attend à ce qu'une trajectoire ayant commencé par suivre un point de l'arrière-plan soit interrompue. Pourtant il arrive assez fréquemment que l'occultation ne soit pas détectée par l'estimateur et que la trajectoire se poursuive, soit en rejoignant progressivement le bord de l'objet occultant et en continuant à suivre un point de ce second objet, soit en évoluant devant le front d'occultation à une faible distance de celui-ci, comme si ce front la poussait. Ces deux cas sont rencontrés pour des points situés dans des régions très peu texturées ou sur un contour de même orientation que le mouvement de l'objet occultant. Ces trajectoires erronées ne sont pas totalement aberrantes puisqu'on peut en extraire au moins une trajectoire plus courte et correcte.

Pour celles de ces trajectoires erronées qui ont été estimées par l'algorithme de KLT, elles auraient sans doute pu être tronquées au bon moment si le suivi des points avait été fait de manière bidirectionnelle, avec vérification de la cohérence des appariements dans les deux sens.

Sans revenir sur le problème d'estimation de trajectoires, nous pensons qu'il est possible de tirer profit du clustering d'un ensemble de trajectoires (contenant tout de même une majorité de trajectoires non erronées) pour proposer un traitement de ces

trajectoires mal estimées. L'objectif serait de détecter automatiquement les trajectoires erronées, et de les découper de manière appropriée avant de procéder à la classification de leurs morceaux.

Conclusion générale

Cette étude a exploré le thème de l'analyse du mouvement dans les séquences d'images. Nous avons présenté dans deux parties différentes deux types d'approches bien distincts : la segmentation d'une séquence d'images au sens du mouvement et le clustering d'un ensemble de trajectoires de points. Dans cette conclusion générale nous proposons d'abord une synthèse des travaux effectués puis évoquons un certain nombre de perspectives.

Synthèse des travaux effectués

Dans la première partie, nous avons présenté deux nouvelles approches semi-automatiques de segmentation d'une séquence d'images au sens du mouvement. Toutes deux sont des approches séquentielles qui reposent sur la formulation du problème de segmentation comme un problème de minimisation d'énergie. Elles diffèrent par l'espacement temporel des images entre lesquelles sont estimés les mouvements utilisés pour distinguer les différents objets constituant la scène.

La première méthode séquentielle propose de segmenter l'image courante en utilisant les mouvements estimés depuis celle-ci vers l'image précédente et l'image suivante. Elle repose donc sur des mouvements estimés entre images consécutives. Pour cette méthode, nos contributions sont liées au renforcement de la cohérence spatio-temporelle de la segmentation. D'une part, une nouvelle contrainte dite de rigidité est appliquée à des objets rigides d'avant-plan et se traduit par l'ajout d'un nouveau terme d'énergie pour décourager l'insertion d'étiquettes correspondant à ces objets dans les zones qui apparaissent. D'autre part, grâce à une étape de prédiction, une majorité de pixels conservent leur étiquette prédite et seuls les pixels proches des frontières de mouvement prédites ou appartenant aux zones qui apparaissent sont réellement sujets au processus de segmentation. Ces travaux ont fait l'objet d'une première publication [Fradet 08b].

Dans la seconde méthode séquentielle proposée, la séquence est partitionnée en un ensemble d'intervalles temporels dont les limites constituent des instants de référence fixés. Chaque intervalle étant traité de manière séquentielle, les mouvements alors utilisés pour segmenter l'image courante sont ceux estimés depuis celle-ci vers chacune des images (l'une antérieure, l'autre postérieure à l'image courante) correspondant aux instants de référence bornant l'intervalle considéré. Ces mouvements sont donc estimés entre images distantes. Il est donc plus facile de les distinguer les uns des autres dans la mesure où leurs amplitudes sont vraisemblablement plus grandes que lorsque l'estima-

tion est faite entre images consécutives. Pour cette seconde méthode, nos contributions sont liées à une nouvelle exploitation de la représentation en couches. Au fil du processus séquentiel de segmentation, des mosaïques de couches sont générées progressivement et automatiquement en accumulant sur un même support à deux dimensions toutes les informations déjà traitées qui ont été affectées à une même couche. Ces mosaïques jouent à leur tour un rôle dans la segmentation des images qui n'ont pas encore été traitées. Ceci se traduit par la définition d'un nouveau terme d'énergie lié au mouvement et basé sur la différence d'intensités entre une image originale et une mosaïque. Comme avec toutes les méthodes, il arrive parfois que les résultats ne soient pas ceux espérés. Nous avons proposé un nouveau type d'interaction pour ne pas avoir à tout reprendre à zéro, ni devoir tout éditer manuellement. L'opérateur a la possibilité d'apporter des corrections aux mosaïques résultant d'une première exécution de l'algorithme. Ces mosaïques éditées sont alors passées en entrée d'une seconde exécution de l'algorithme, ce qui permet de propager à toute la séquence les indications fournies par l'opérateur et donc d'améliorer efficacement les résultats. Par ailleurs, ces mosaïques peuvent être réutilisées pour différentes applications liées par exemple à la manipulation d'objets vidéo. Ces travaux ont fait l'objet d'une seconde publication [Fradet 08a] et d'un brevet déposé en mars 2008 au nom de la société Thomson Licensing.

Dans la seconde partie, nous avons présenté un nouvel algorithme pour le clustering d'un ensemble de trajectoires de points. L'algorithme proposé combine robustesse des estimations (y compris en présence de trajectoires aberrantes), rejet explicite de ces trajectoires aberrantes, et facilité d'interprétation. Nos contributions sont liées à l'introduction d'un modèle affine de trajectoire défini comme une série temporelle de modèles affines de mouvement estimés entre instants consécutifs, et à la définition d'une erreur résiduelle de mouvement appropriée qui permet d'appréhender n'importe quelle trajectoire quelle que soit sa longueur. Le système proposé permet de traiter des ensembles de trajectoires, chacune étant définie sur un intervalle temporel qui lui est propre. Ces travaux ont fait l'objet d'une troisième publication [Fradet 09].

Perspectives

Quelques perspectives ont déjà été évoquées au sein du document, notamment dans la discussion menée à la section 5.5 et dans les conclusions des deux parties. Nous proposons ici de nouvelles perspectives, certaines d'ordre méthodologique, d'autres d'ordre plus applicatif.

Application à la compression vidéo basée sur les objets

Nos algorithmes de segmentation n'ont pour le moment pas été utilisés à des fins de codage. Leur réelle capacité à réduire l'information d'une séquence vidéo en vue de sa compression reste à étudier. Pour cette application spécifique, la localisation des frontières de mouvement ne doit pas nécessairement être extrêmement précise. Par

contre on doit s'attendre à ce que le nombre de couches idéal pour la compression ne soit pas le même que celui adapté à l'édition vidéo.

Combinaison des différentes méthodes

La combinaison des différentes méthodes constitue une des perspectives de notre étude. L'approche basée sur les trajectoires peut être utilisée pour faciliter les interventions de l'opérateur dans les deux autres approches, voire pour automatiser certaines tâches. Ainsi, les résultats du clustering de trajectoires peuvent guider l'opérateur lorsqu'il segmente une image de référence. Pour cette image de référence, il disposera effectivement d'un ensemble de pixels partitionné. Le système pourra donc lui suggérer un nombre de couches et une idée de la localisation de ces couches dans l'image. Par ailleurs, la sélection des instants de référence, elle-même, pourrait être automatisée en analysant cet ensemble de trajectoires.

Passage de l'épars au dense en conservant la notion de trajectoire de point

Sans parler de combiner les méthodes, une extension de l'approche basée sur les trajectoires permettant d'obtenir des cartes de segmentation denses serait particulièrement intéressante à notre sens si elle conservait la notion de trajectoires. Par exemple, il est envisageable d'estimer un tube dense de trajectoires par concaténation de vecteurs de mouvement issus de flots optiques estimés entre paires d'images successives. En formulant à nouveau le problème de segmentation comme un problème de minimisation d'énergie résolu par une approche de coupe minimale dans un graphe, chaque nœud du graphe correspondrait cette fois non pas à un pixel mais à une trajectoire. La définition des relations de voisinage entre trajectoires n'est pas aussi directe et demanderait une étude plus poussée afin que le graphe construit ne devienne pas exagérément complexe.

Nouvelle description des objets vidéo

La perspective la plus ambitieuse est certainement la proposition d'une nouvelle description des objets vidéo, s'appuyant sur la segmentation, l'estimation précise de mouvement sur de courts, moyens et longs intervalles temporels et les interactions d'un opérateur. L'extraction d'attributs supplémentaires (texture, forme, orientation, illumination...) permettrait une modélisation plus complète des séquences traitées.

Extension à la segmentation et à la modélisation 3D d'objets vidéo

Avec la nouvelle dimension que prennent le cinéma 3D et la télévision 3D, nous serions tentés d'ajouter à nos schémas l'information de disparité qui peut être estimée pour les séquences stéréo ou multi-vues. Cette information constituerait un critère supplémentaire sur lequel pourrait s'appuyer la segmentation. En plus d'être assurée spatialement au sein d'une même image et temporellement entre les images d'une séquence, la cohérence devrait être également assurée entre les différentes vues. Les méthodes de

coupe dans un graphe ont déjà été appliquées à la stéréovision et à la reconstruction de scène à partir de plusieurs caméras, mais jusqu'à présent très peu de travaux ont été effectués dans un cadre semi-automatique.

Glossaire

BP	Belief Propagation, propagation de croyances
CS	Consensus Set, ensemble de consensus
DFD	Displaced Frame Difference, différence inter-images déplacée
DTW	Dynamic Time Warping, déformation temporelle dynamique ou alignement temporel dynamique
ECMA	Équation de Contrainte du Mouvement Apparent
EM	Expectation Maximization, espérance-maximisation
GMM	Gaussian Mixture Model, modèle de mélange de gaussiennes
ICM	Iterated Conditional Modes, modes conditionnels itérés
IEC	International Electrotechnical Commission, commission électrotechnique internationale
ISO	International Organization for Standardization, organisation internationale de normalisation
KCC	K-Connected Component
LCSS	Longest Common SubSequence, plus longue sous-séquence commune
LBP	Loopy Belief Propagation, propagation bouclée de croyances
LMedS	Least Median of Squares, moindre médiane des carrés
MC	Moindres Carrés
MCP	Moindres Carrés Pondérés
MCPI	Moindres Carrés Pondérés Itérés
MDL	Minimum Description Length, longueur de description minimale
MPEG	Moving Picture Experts Group
MRF	Markov Random Field, champ aléatoire de Markov
MSS	Minimum Sample Set, sous-ensemble minimum aléatoire
PS	Preference Set, ensemble de préférence
RANSAC	RANdom SAmple Consensus
SSD	Sum of Square Differences, somme des différences au carré
S-VOP	Sprite Video Object Plane
TRW	Tree ReWeighted Message Passing

Bibliographie

[Alvarez 99] L. Alvarez, J. Esclarín, M. Lefébure & J. Sánchez. *A PDE model for computing the optical flow.* In Proc. XVI Congreso de Ecuaciones Diferenciales y Aplicaciones, pages 1349–1356, Las Palmas de Gran Canaria, Spain, September 1999. 64

[Aurich 95] Volker Aurich & Jörg Weule. *Non-Linear Gaussian Filters Performing Edge Preserving Diffusion.* In Proc. Deutsche Arbeitsgemeinschaft für Mustererkennung Symposium (DAGM), pages 538–545, London, UK, 1995. Springer-Verlag. 65

[Ayer 95] Serge Ayer & Harpreet S. Sawhney. *Layered Representation of Motion Video Using Robust Maximum-Likelihood Estimation of Mixture Models and MDL Encoding.* In Proc. Int. Conf. Computer Vision (ICCV), pages 777–784, 1995. 20, 22

[Azran 06] Arik Azran & Zoubin Ghahramani. *Spectral Methods for Automatic Multiscale Data Clustering.* In Proc. Conf. Computer Vision Pattern Rec. (CVPR), 2006. 137

[Baker 07] S. Baker, S. Roth, D. Scharstein, M.J. Black, J.P. Lewis & R. Szeliski. *A Database and Evaluation Methodology for Optical Flow.* In Proc. Int. Conf. Computer Vision (ICCV), pages 1–8, 2007. http ://vision.middlebury.edu/flow/. 59, 70

[Benois-Pineau 92] J. Benois-Pineau & D. Barba. *Image segmentation by region-contour cooperation as a basis for efficient coding scheme.* In Proc. of the SPIE - Visual Communications and Image Processing (VCIP), pages 1218–1229, 1992. 24

[Berndt 94] D. J. Berndt & J. Clifford. *Using Dynamic Time Warping to Find Patterns in Time Series.* In Proc. of KDD-94 : AAAI Workshop on KnowledgeDiscovery in Databases, pages 359–370, Seattle, Washington, July 1994. 131

[Besag 86] J. Besag. *On the Statistical Analysis of Dirty Pictures.* Journal of the Royal Statistical Society - Series B, vol. 48, no. 3, pages 259–302, 1986. 49, 57

[Birchfield 07] S. Birchfield. *KLT : An implementation of the Kanade-Lucas-Tomasi feature tracker*, 2007. http ://www.ces.clemson.edu/stb/klt/. 156

[Black 96a] M. J. Black & P. Anandan. *The Robust Estimation of Multiple Motions : Parametric and Piecewise-Smooth Flow Fields*. Computer Vision and Image Understanding, vol. 63, no. 1, pages 75–104, 1996. 20, 23

[Black 96b] Michael J. Black & Allan Jepson. *Estimating Optical Flow in Segmented Images using Variable-order Parametric Models with Local Deformations*. IEEE Trans. Pattern Anal. Machine Intell. (PAMI), vol. 18, pages 972–986, 1996. 20, 23

[Boult 91] T. E. Boult & Gottesfeld. *Factorization-based segmentation of motions*. In Proc. of the IEEE Workshop on Visual Motion, pages 179–186, 1991. 134

[Bouthemy 93] Patrick Bouthemy & Edouard François. *Motion segmentation and qualitative dynamic scene analysis from an image sequence*. Int. J. Computer Vision (IJCV), vol. 10, no. 2, pages 157–182, 1993. 69

[Boykov 01a] Yuri Boykov, Olga Veksler & Ramin Zabih. *Fast Approximate Energy Minimization via Graph Cuts*. IEEE Trans. Pattern Anal. Machine Intell. (PAMI), vol. 23, November 2001. 43, 45, 48

[Boykov 01b] Yuri Y. Boykov & Marie-Pierre Jolly. *Interactive Graph Cuts for Optimal Boundary & Region Segmentation of Objects in N-D Images*. In Proc. Int. Conf. Computer Vision (ICCV), volume 1, pages 105–112, July 2001. 45, 54, 55, 112

[Boykov 04] Yuri Boykov & Vladimir Kolmogorov. *An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Vision*. IEEE Trans. Pattern Anal. Machine Intell. (PAMI), vol. 26, no. 9, pages 1124–1137, September 2004. 39, 41, 42

[Brostow 06] Gabriel J. Brostow & Roberto Cipolla. *Unsupervised Bayesian Detection of Independent Motion in Crowds*. In Proc. Conf. Computer Vision Pattern Rec. (CVPR), pages I : 594–601, 2006. 130, 142

[Brox 04] T. Brox, A. Bruhn, N. Papenberg & J. Weickert. *High accuracy optical flow estimation based on a theory for warping*. In T. Pajdla & J. Matas, editeurs, Proc. Euro. Conf. Computer Vision (ECCV), volume 3024 of *LNCS*, pages 25–36, Prague, Czech Republic, May 2004. Springer. 60, 64

[Bruhn 05a] Andrés Bruhn & Joachim Weickert. *Towards Ultimate Motion Estimation : Combining Highest Accuracy with Real-Time Performance*. In Proc. Int. Conf. Computer Vision (ICCV), pages 749–755, 2005. 60, 64

[Bruhn 05b] Andrés Bruhn, Joachim Weickert & Christoph Schnörr. *Lucas/Kanade Meets Horn/Schunck : Combining Local and Global Optic Flow Methods*. Int. J. Computer Vision (IJCV), vol. 61, no. 3, pages 211–231, 2005. 63

[Bugeau 07] Aurélie Bugeau. *Détection et suivi d'objets en mouvement dans des scènes complexes, application à la surveillance des conducteurs.* Thèse de doctorat, Université de Rennes 1, Décembre 2007. 42

[Buzan 04] Dan Buzan, Stan Sclaroff & George Kollios. *Extraction and Clustering of Motion Trajectories in Video.* In Proc. Int. Conf. Pattern Rec. (ICPR), pages 521–524, 2004. 133, 142, 148, 158, 159

[Clifford 95] C. W. G. Clifford, K. Langley & D. J. Fleet. *Centre-Frequency Adaptive IIR Temporal Filters For Phase-Based Image Velocity Estimation.* Image Processing and its Applications, vol. 4-6, pages 173–177, 1995. 61

[Cohen 93] Isaac Cohen. *Nonlinear Variational Method for Optical Flow Computation.* In Proc. SCIA, pages 523–530, June 1993. 64

[Comaniciu 02] Dorin Comaniciu & Peter Meer. *Mean Shift : A Robust Approach Toward Feature Space Analysis.* IEEE Trans. Pattern Anal. Machine Intell. (PAMI), vol. 24, no. 5, pages 603–619, may 2002. 26

[Costeira 95] J. Costeira & Takeo Kanade. *A Multi-body Factorization Method for Motion Analysis.* In Proc. Int. Conf. Computer Vision (ICCV), pages 1071–1076, June 1995. 134

[Cremers 05] D. Cremers & S. Soatto. *Motion Competition : A Variational Approach to Piecewise Parametric Motion Segmentation.* Int. J. Computer Vision (IJCV), vol. 62, no. 3, pages 249–265, May 2005. 20, 24

[Criminisi 04] A. Criminisi, P. Pérez & K. Toyama. *Region filling and object removal by exemplar-based image inpainting.* IEEE Transactions on Image Processing, vol. 13, no. 9, pages 1200–1212, 2004. 109

[Dempster 77] A. P. Dempster, N. M. Laird & D. B. Rubin. *Maximum likelihood from incomplete data via the EM algorithm.* Journal of the Royal Statistical Society, Series B, vol. 39, no. 1, pages 1–38, 1977. 83

[Dexter 09] E. Dexter, P. Pérez & I. Laptev. *Multi-view synchronization of human actions and dynamic scenes.* In Proc. British Machine Vision Conf. (BMVC), London, United Kingdom, September 2009. 142

[Dupont 05] R. Dupont, N. Paragios, R. Keriven & P. Fuchs. *Extraction of layers of similar motion through combinatorial techniques.* In Proc. Int. Conf. Energy Minimization Methods on Computer Vision and Pattern Rec. (EMMCVPR), November 2005. 17, 30, 31, 82, 86, 95, 97, 98, 99

[Dupont 06a] R. Dupont. *Suivi des Parties Cachées dans une Séquence Vidéo et Autres Problèmes Soulevés par la Reconstruction Tridimensionnelle d'un Environnement Urbain.* Thèse de doctorat, Ecole Nationale des Ponts et Chaussées, Décembre 2006. 25, 30, 32, 35, 38, 77, 79, 100, 101

[Dupont 06b] R. Dupont, N. Paragios, R. Keriven & P. Fuchs. *Extraction de Couches de Même Mouvement Via des Techniques Combinatoires.* In Congrès Francophone sur la Reconnaissance des Formes et Intelligence Artificielle (RFIA), January 2006. 17, 25, 30, 31, 35, 38, 86, 95

[Dupont 06c] Romain Dupont, Olivier Juan & Renaud Keriven. *Robust Segmentation of Hidden Layers in Video Sequences.* In Proc. Int. Conf. Pattern Rec. (ICPR), volume 3, pages 75–78, Los Alamitos, CA, USA, 2006. IEEE Computer Society. 17, 25, 30, 31, 32, 35, 106

[Fablet 01] Ronan Fablet. *Modélisation statistique non paramétrique et reconnaissance du mouvement dans des séquences d'images ; application à l'indexation vidéo.* Thèse de doctorat, Université de Rennes I, Juillet 2001. 71

[Felzenszwalb 06] P.F. Felzenszwalb & D.P. Huttenlocher. *Efficient Belief Propagation for Early Vision.* Int. J. Computer Vision (IJCV), vol. 70, no. 1, pages 41–54, October 2006. 57

[Fischler 81] Martin A. Fischler & Robert C. Bolles. *Random Sample Consensus : A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography.* Communications of the ACM, vol. 24, no. 6, pages 381–395, 1981. 137

[Fleet 90] David J. Fleet & A. D. Jepson. *Computation of component image velocity from local phase information.* Int. J. Computer Vision (IJCV), vol. 5, no. 1, pages 77–104, 1990. 61

[Ford 62] L. Ford & D. Fulkerson. Flows in networks. Princeton University Press, Princeton, NJ, 1962. 42

[Fradet 08a] Matthieu Fradet, Patrick Pérez & Philippe Robert. *Semi-automatic Motion Segmentation with Motion Layer Mosaics.* In Proc. Euro. Conf. Computer Vision (ECCV), pages 210–223, december 2008. 30, 36, 38, 166

[Fradet 08b] Matthieu Fradet, Patrick Pérez & Philippe Robert. *Time-Sequential Extraction of Motion Layers.* In Proc. Int. Conf. Image Processing (ICIP), January 2008. 35, 38, 165

[Fradet 09] Matthieu Fradet, Patrick Pérez & Philippe Robert. *Clustering point trajectories with various life-spans.* In Proc. Eur. Conf. on Visual Media Production (CVMP), London, UK, November 2009. 166

[Geman 84] Stuart Geman & Donald Geman. *Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images.* IEEE Trans. Pattern Anal. Machine Intell. (PAMI), vol. 6, no. 6, pages 721–741, November 1984. 56

[Ghanbari 90] M. Ghanbari. *The cross-search algorithm for motion estimation.* IEEE Trans. on Communications, vol. 38, no. 7, pages 950–953, July 1990. 61, 62, 183

[Goldberg 86] A. V. Goldberg & R. E. Tarjan. *A new approach to the maximum flow problem*. In Proc. ACM Symposium on Theory of Computing (STOC), pages 136–146, New York, NY, USA, 1986. ACM. 43

[Greig 89] D. M. Greig, B. T. Porteous & A. H. Seheult. *Exact Maximum A Posteriori Estimation for Binary Images*. Journal of the Royal Statistical Society B, vol. 51, no. 2, pages 271–279, 1989. 39

[Harris 88] C. Harris & M. Stephens. *A Combined Corner and Edge Detection*. In Proceedings of The Fourth Alvey Vision Conference, pages 147–151, 1988. 28

[Hartley 04] R. Hartley & F. Schaffalitzky. *PowerFactorization : 3D reconstruction with missing or uncertain data*. In Japan-Australia Workshop on Computer Vision, 2004. 136

[Heeger 88] David J. Heeger. *Optical flow using spatiotemporal filters*. Int. J. Computer Vision (IJCV), vol. 1, no. 4, pages 279–302, 1988. 61

[Heel 91] J. Heel. *Temporal Surface Reconstruction*. In Proc. Conf. Computer Vision Pattern Rec. (CVPR), pages 607–612, 1991. 68

[Horn 81] Berthold K.P. Horn & Brian G. Schunck. *Determining Optical Flow*. Artificial Intelligence, vol. 17, pages 185–203, 1981. 59, 63

[Ichimura 99] Naoyuki Ichimura. *Motion Segmentation based on Factorization Method and Discriminant Criterion*. In Proc. Int. Conf. Computer Vision (ICCV), volume 1, 1999. 134

[Irani 92] Michal Irani, Benny Rousso & Shmuel Peleg. *Detecting and Tracking Multiple Moving Objects Using Temporal Integration*. In Proc. Euro. Conf. Computer Vision (ECCV), pages 282–287, London, UK, 1992. Springer-Verlag. 20, 24

[Ishikawa 03] H. Ishikawa. *Exact Optimization for Markov Random Fields with Convex Priors*. IEEE Trans. Pattern Anal. Machine Intell. (PAMI), vol. 25, no. 10, pages 1333–1336, October 2003. 47, 48

[Jain 81] J.R. Jain & A.K. Jain. *Displacement Measurement and Its Application in Interframe Image Coding*. IEEE Trans. on Commununications, vol. 29, no. 12, pages 1799–1808, December 1981. 61

[Jepson 93] A. Jepson & M. Black. *Mixture models for optical flow computation*. Technical Report RBCV-TR-93-44, Department of Computer Science, University of Toronto, April 1993. 23

[Ju 96] S. X. Ju, M. J. Black & A. D. Jepson. *Skin and Bones : Multi-layer, locally affine, optical flow and regularization with transparency*. In Proc. Conf. Computer Vision Pattern Rec. (CVPR), pages 307–314, San Francisco, June 1996. 20, 23

[Juan 06] O. Juan & Y.Y. Boykov. *Active Graph Cuts*. In Proc. Conf. Computer Vision Pattern Rec. (CVPR), volume I, pages 1023–1029, 2006. 43

[Kanatani 01] K. Kanatani. *Motion segmentation by subspace separation and model selection.* In Proc. Int. Conf. Computer Vision (ICCV), volume 2, pages 586–591, 2001. 134

[Ke 01] Qifa Ke & Takeo Kanade. *A Subspace Approach to Layer Extraction.* In Proc. Conf. Computer Vision Pattern Rec. (CVPR), pages 255–262, 2001. 25, 26, 38

[Ke 02] Qifa Ke & Takeo Kanade. *A Robust Subspace Approach to Layer Extraction.* In IEEE Workshop on Motion and Video Computing (Motion), December 2002. 25, 27, 38

[Kirkpatrick 83] S. Kirkpatrick, C. D. Gelatt & M. P. Vecchi. *Optimization by Simulated Annealing.* Science, vol. 220, no. 4598, pages 671–680, May 1983. 49, 56

[Klaus 06] Andreas Klaus, Mario Sormann & Konrad Karner. *Segment-Based Stereo Matching Using Belief Propagation and a Self-Adapting Dissimilarity Measure.* In Proc. Int. Conf. Pattern Rec. (ICPR), pages 15–18, Washington, DC, USA, 2006. IEEE Computer Society. 64

[Kohli 05] P. Kohli & P.H.S. Torr. *Efficiently Solving Dynamic Markov Random Fields Using Graph Cuts.* In Proc. Int. Conf. Computer Vision (ICCV), volume II, pages 922–929, 2005. 43

[Kohli 07] P. Kohli & P.H.S. Torr. *Dynamic Graph Cuts for Efficient Inference in Markov Random Fields.* IEEE Trans. Pattern Anal. Machine Intell. (PAMI), vol. 29, no. 12, pages 2079–2088, December 2007. 43

[Kolmogorov 04] V. Kolmogorov & R. Zabih. *What Energy Functions Can Be Minimized via Graph Cuts ?* IEEE Trans. Pattern Anal. Machine Intell. (PAMI), vol. 26, no. 2, pages 147–159, February 2004. 47

[Kolmogorov 06a] V. Kolmogorov & C. Rother. *Comparison of Energy Minimization Algorithms for Highly Connected Graphs.* In Proc. Euro. Conf. Computer Vision (ECCV), volume II, pages 1–15, 2006. 57, 58

[Kolmogorov 06b] Vladimir Kolmogorov. *Convergent Tree-Reweighted Message Passing for Energy Minimization.* IEEE Trans. Pattern Anal. Machine Intell. (PAMI), vol. 28, no. 10, pages 1568–1583, 2006. 57

[Komodakis 08] N. Komodakis, G. Tziritas & N. Paragios. *Performance vs computational efficiency for optimizing single and dynamic MRFs : Setting the state of the art with primal-dual strategies.* Computer Vision and Image Understanding (CVIU), vol. 112, no. 1, pages 14–29, October 2008. 58

[Lemonnier 93] Bruno Lemonnier. *Approche intégrative de l'analyse du mouvement et de la reconstruction 3D dans les séquences d'images.* Thèse de doctorat, Université de Rennes 1, Mai 1993. 67

[Lim 97] Kyoung Won Lim & Jong Beom Ra. *Improved hierarchical search block matching algorithm by using multiple motion vector candidates*. Electronic Letters, vol. 33, pages 1771–1772, October 1997. 62

[Liu 05] Ce Liu, Antonio Torralba, William T. Freeman, Frédo Durand & Edward H. Adelson. *Motion Magnification*. ACM Trans. on Graphics (SIGGRAPH), 2005. 25, 28, 35, 38, 137

[Lucas 81] B.D. Lucas & T. Kanade. *An Iterative Image Registration Technique with an Application to Stereo Vision*. In Proc. DARPA Image Understanding Workshop, pages 121–130, April 1981. 59, 63, 156

[Meila 01] M. Meila & J. Shi. *A random walks view of spectral segmentation*. In Int. Workshop on Artificial Intelligence and Statistics (AISTATS), 2001. 137

[Mémin 02] E. Mémin & P. Pérez. *Hierarchical estimation and segmentation of dense motion fields*. Int. J. Computer Vision (IJCV), vol. 46, no. 2, pages 129–155, February 2002. 20, 23

[Min 08] Changki Min & Gérard Medioni. *Inferring Segmented Dense Motion Layers Using 5D Tensor Voting*. IEEE Trans. Pattern Anal. Machine Intell. (PAMI), vol. 30, no. 9, pages 1589–1602, 2008. 30

[Morier 97] F. Morier, J. Benois-Pineau, D. Barba & H. Sanson. *Robust segmentation of moving image sequences*. In Proc. Int. Conf. Image Processing (ICIP), 1997. 24

[Moscheni 98] Fabrice Moscheni, Sushil Bhattacharjee & Murat Kunt. *Spatiotemporal Segmentation Based on Region Merging*. IEEE Trans. Pattern Anal. Machine Intell. (PAMI), vol. 20, no. 9, pages 897–915, 1998. 24

[Nagel 86] H. H. Nagel & W. Enkelmann. *An investigation of smoothness constraints for the estimation of displacement vector fields from image sequences*. IEEE Trans. Pattern Anal. Machine Intell. (PAMI), vol. 8, no. 5, pages 565–593, 1986. 64

[Nam 95] K.M. Nam, J.S. Kim, R.H. Park & Y.S. Shim. *A fast hierarchical motion vector estimation algorithm using mean pyramid*. IEEE Trans. on Circuits and Systems for Video Technology, vol. 5, no. 4, pages 344–351, August 1995. 62

[Ng 01] A. Ng, M. Jordan & Y. Weiss. *On spectral clustering : Analysis and an algorithm*. In Advances in Neural Information Processing Systems (NIPS), 2001. 135, 136

[Odobez 95] J.-M. Odobez & P. Bouthemy. *Robust multiresolution estimation of parametric motion models*. Journal of Visual Communication and Image Representation, vol. 6, no. 4, pages 348–365, December 1995. 24

[Odobez 98] J.-M. Odobez & P. Bouthemy. *Direct incremental model-based image motion segmentation for video analysis.* Signal Processing, vol. 66, no. 2, pages 143–155, 1998. 20, 24

[Orchard 91] Michael Orchard & Charles Bouman. *Color Quantization of Images.* IEEE Transactions on Signal Processing, vol. 39, pages 2677–2690, 1991. 83

[Paris 06] Sylvain Paris & Frédo Durand. *A Fast Approximation of the Bilateral Filter Using a Signal Processing Approach.* In Proc. Euro. Conf. Computer Vision (ECCV), pages 568–580, 2006. 67

[Pedersini 96] Federico Pedersini, Augusto Sarti & Stefano Tubaro. *Combined Motion and Edge Analysis for a Layer-based Representation of Image Sequences.* In Proc. Int. Conf. Image Processing (ICIP), 1996. 20, 23

[Potts 52] Renfrey B. Potts. *Some Generalized Order-Disorder Transformations.* In Proceedings of the Cambridge Philosophical Society, volume 48, pages 106–109, 1952. 48

[Pundlik 08] Shrinivas J. Pundlik & Stanley T. Birchfield. *Real-Time Motion Segmentation of Sparse Feature Points at Any Speed.* IEEE Transactions on Systems, Man, and Cybernetics, Part B, vol. 38, no. 3, pages 731–742, 2008. 140

[Rao 08] Shankar Rao, Roberto Tron, René Vidal & Yi Ma. *Motion segmentation via robust subspace separation in the presence of outlying, incomplete, or corrupted trajectories.* In Proc. Conf. Computer Vision Pattern Rec. (CVPR). IEEE Computer Society, 2008. 134, 136, 154, 155, 157

[Rav-Acha 08] Alex Rav-Acha, Pushmeet Kohli, Carsten Rother & Andrew Fitzgibbon. *Unwrap Mosaics : A new representation for video editing.* ACM Trans. on Graphics (SIGGRAPH), vol. 27, no. 3, pages 17 :1–17 :11, august 2008. 110

[Rissanen 83] J. Rissanen. *A Universal Prior for Integers and Estimation by Minimum Description Length.* Annals of Statistics, vol. 11, no. 2, pages 416–431, 1983. 22, 82

[Rother 04] C. Rother, V. Kolmogorov & A. Blake. *Grabcut - Interactive Foreground Extraction using Iterated Graph Cuts.* ACM Trans. on Graphics (SIGGRAPH), 2004. 45, 55, 56, 58, 82, 83, 112

[Roy 98] Sébastien Roy & Ingemar J. Cox. *A Maximum-Flow Formulation of the N-Camera Stereo Correspondence Problem.* In Proc. Int. Conf. Computer Vision (ICCV), 1998. 47

[Sand 08] P. Sand & S. Teller. *Particle Video : Long-Range Motion Estimation using Point Trajectories.* Int. J. Computer Vision (IJCV), 2008. http ://rvsn.csail.mit.edu/pv/. 156

[Schoenemann 08] T. Schoenemann & D. Cremers. *High Resolution Motion Layer Decomposition using Dual-space Graph Cuts*. In Proc. Conf. Computer Vision Pattern Rec. (CVPR), Anchorage, Alaska, June 2008. 25, 29, 30, 38

[Schrijver 03] Alexander Schrijver. Combinatorial optimization - polyhedra and efficiency, volume A of *Algorithms and Combinatorics*. Springer, 2003. 42

[Shi 94] Jianbo Shi & Carlo Tomasi. *Good Features to Track*. In Proc. Conf. Computer Vision Pattern Rec. (CVPR), pages 593–600, 1994. 156

[Shi 98] Jianbo Shi & J. Malik. *Motion segmentation and tracking using normalized cuts*. In Proc. Int. Conf. Computer Vision (ICCV), pages 1154–1160, 1998. 25, 28, 38, 137

[Shi 00] Jianbo Shi & Jitendra Malik. *Normalized Cuts and Image Segmentation*. IEEE Trans. Pattern Anal. Machine Intell. (PAMI), vol. 22, no. 8, pages 888–905, August 2000. 137

[Sugaya 04] Yasuyuki Sugaya & Kenichi Kanatani. *Geometric structure of degeneracy for multi-body motion segmentation*. In Workshop on Statistical Methods in Video Processing, pages 1–2. Springer-Verlag, 2004. 134, 135, 142, 155

[Sun 05] J. Sun, L. Yuan, J. Jia & H-Y Shum. *Image Completion with Structure Propagation*. ACM Trans. on Graphics (SIGGRAPH), 2005. 110

[Talbot 04] Justin F. Talbot & Xiaoqian Xu. Implementing grabcut. 2004. 83

[Toldo 08] Roberto Toldo & Andrea Fusiello. *Robust Multiple Structures Estimation with J-Linkage*. In Proc. Euro. Conf. Computer Vision (ECCV), 2008. 149, 150

[Tomasi 91] C. Tomasi & T. Kanade. *Detection and tracking of point features*. Rapport technique, Carnegie Mellon University, April 1991. 63, 156

[Tomasi 98] C. Tomasi & R. Manduchi. *Bilateral Filtering for Gray and Color Images*. In Proc. Int. Conf. Computer Vision (ICCV), Washington, DC, USA, 1998. IEEE Computer Society. 65

[Torr 98] P. H. S. Torr. *Geometric Motion Segmentation and Model Selection*. Phil. Trans. Royal Society of London A, vol. 356, no. 1740, pages 1321–1340, 1998. 139

[Tron 07] Roberto Tron & René Vidal. *A Benchmark for the Comparison of 3-D Motion Segmentation Algorithms*. In Proc. Conf. Computer Vision Pattern Rec. (CVPR), volume 0, pages 1–8, Los Alamitos, CA, USA, 2007. IEEE Computer Society. http ://www.vision.jhu.edu/motion.htm. 134, 142, 153, 154, 155

[Van Wijk 99] Jarke J. Van Wijk & Edward R. Van Selow. *Cluster and Calendar Based Visualization of Time Series Data*. In INFOVIS '99 : Proc.

of the 1999 IEEE Symp. on Information Visualization, Washington, DC, USA, 1999. IEEE Computer Society. 130

[Vidal 04a] R. Vidal, Y. Ma & J. Piazzi. *A new GPCA algorithm for clustering subspaces by fitting, differentiating and dividing polynomials*. In Proc. Conf. Computer Vision Pattern Rec. (CVPR), pages I : 510–517, 2004. 135

[Vidal 04b] René Vidal & Richard Hartley. *Motion Segmentation with Missing Data Using PowerFactorization and GPCA*. In Proc. Conf. Computer Vision Pattern Rec. (CVPR), volume 2, pages 310–316, Los Alamitos, CA, USA, 2004. IEEE Computer Society. 134, 135, 136, 142, 155

[Vidal 05] R. Vidal, Y. Ma & S. Sastry. *Generalized Principal Component Analysis (GPCA)*. IEEE Trans. Pattern Anal. Machine Intell. (PAMI), vol. 27, no. 12, pages 1945–1959, December 2005. 135

[Vlachos 02a] M. Vlachos, G.Kollios & D. Gunopulos. *Discovering Similar Multidimensional Trajectories*. In Proc. of 18th International Conference on Data Engineering (ICDE), pages 673–684, 2002. 133

[Vlachos 02b] M. Vlachos, D. Gunopulos & G. Kollios. *Robust Similarity Measures for Mobile Object Trajectories*. In Proc. of 13th Database and Expert Systems Applications (DEXA) 5th International Workshop "Mobility in Databases and Distributed Systems" (MDDS), pages 721–726, Aix-en-Provence, France, 2002. 133

[Vlachos 03] Michail Vlachos, Marios Hadjieleftheriou, Dimitrios Gunopulos & Eamonn Keogh. *Indexing multi-dimensional time-series with support for multiple distance measures*. In Proc. of the ACM SIGKDD Int. Conf. on Knowledge Discovery and Data mining, pages 216–225, 2003. 132

[Wainwright 05] M.J. Wainwright, T.S. Jaakkola & A.S. Willsky. *MAP estimation via agreement on (hyper) trees : message-passing and linear programming approaches*. IEEE Trans. on Information Theory, vol. 51, no. 11, pages 3697–3717, November 2005. 57

[Wang 94a] John Y. A. Wang & Edward H. Adelson. *Representing Moving Images with Layers*. IEEE Trans. on Image Processing, vol. 3, no. 5, pages 625–638, September 1994. 19, 20, 21, 22, 23, 29, 183

[Wang 94b] John Y. A. Wang & Edward H. Adelson. *Spatio-Temporal Segmentation of Video Data*. Rapport technique 262, M.I.T. Media Laboratory Vision and Modeling Group, February 1994. 20, 21

[Wang 08] Zeng-Fu Wang & Zhi-Gang Zheng. *A region based stereo matching algorithm using cooperative optimization*. In Proc. Conf. Computer Vision Pattern Rec. (CVPR), volume 0, pages 1–8, Los Alamitos, CA, USA, 2008. IEEE Computer Society. 64

[Weber 95] Joseph Weber & Jitendra Malik. *Robust Computation of Optical Flow in a Multi-Scale Differential Framework*. Int. J. Computer Vision (IJCV), vol. 14, no. 1, pages 67–81, 1995. 61

[Weickert 01] Joachim Weickert & Christoph Schnörr. *A Theoretical Framework for Convex Regularizers in PDE-Based Computation of Image Motion*. Int. J. Computer Vision (IJCV), vol. 45, no. 3, pages 245–264, 2001. 64

[Weiss 97] Y. Weiss. *Smoothness in Layers : Motion Segmentation Using Nonparametric Mixture Estimation*. In Proc. Conf. Computer Vision Pattern Rec. (CVPR), pages 520–526, 1997. 71

[Wills 03] Josh Wills, Sameer Agarwal & Serge Belongie. *What Went Where*. In Proc. Conf. Computer Vision Pattern Rec. (CVPR), volume 1, pages 37–454, Madison, WI, June 2003. 25, 38

[Wills 06] Josh Wills, Sameer Agarwal & Serge Belongie. *A Feature-based Approach for Dense Segmentation and Estimation of Large Disparity Motion*. Int. J. Computer Vision (IJCV), vol. 68, no. 2, pages 125–143, 2006. 139

[Wu 96] L. Wu, J. Benois-Pineau, P. Delagnes & D. Barba. *Spatio-temporal segmentation of image sequences for object-oriented low bit-rate image coding*. Signal Processing : Image Communication, vol. 8, no. 6, pages 513–543, September 1996. 24

[Xiao 04] J. Xiao & M. Shah. *Motion layer extraction in the presence of occlusion using graph cut*. In Proc. Conf. Computer Vision Pattern Rec. (CVPR), 2004. 86

[Xiao 05a] J. Xiao & M. Shah. *Accurate Motion Layer Segmentation and Matting*. In Proc. Conf. Computer Vision Pattern Rec. (CVPR), June 2005. 98

[Xiao 05b] J. Xiao & M. Shah. *Motion Layer Extraction in the Presence of Occlusion Using Graph Cuts*. IEEE Trans. Pattern Anal. Machine Intell. (PAMI), vol. 27, no. 10, pages 1644–1659, October 2005. 25, 27, 30, 35, 38, 77, 95, 97, 98, 99, 100, 101, 106, 115

[Xiao 06] J. Xiao, H. Cheng, H.S. Sawhney, C. Rao & M. Isnardi. *Bilateral Filtering-Based Optical Flow Estimation with Occlusion Detection*. In Proc. Euro. Conf. Computer Vision (ECCV), pages I : 211–224, 2006. 65, 66

[Xu 08] L. Xu, J.N. Chen & J.Y. Jia. *A Segmentation Based Variational Model for Accurate Optical Flow Estimation*. In Proc. Euro. Conf. Computer Vision (ECCV), pages I : 671–684, 2008. 64

[Yan 06] Jingyu Yan & Marc Pollefeys. *A General Framework for Motion Segmentation : Independent, Articulated, Rigid, Non-rigid, Degenerate and Non-degenerate*. In Proc. Euro. Conf. Computer Vision (ECCV), pages 94–106, 2006. 134, 135, 142, 155

[Zach 07] C. Zach, T. Pock & H. Bischof. *A Duality Based Approach for Real-time TV-L1 Optical Flow*. In Pattern Recognition (Proc. DAGM), pages 214–223, Heidelberg, Germany, 2007. 64

[Zhang 05] Y. Zhang, J. Xiao & M. Shah. *Motion Layer Based Object Removal in Videos*. In IEEE Workshop on Application on Computer Vision, January 2005. 109

[Zhu 00] Shan Zhu & Kai-Kuang Ma. *A new diamond search algorithm for fast block-matching motion estimation*. IEEE Trans. on Image Processing, vol. 9, no. 2, pages 287–290, February 2000. 61

[Zuliani 05] M. Zuliani, C. S. Kenney & B. S. Manjunath. *The Multiransac Algorithm and its Application to Detect Planar Homographies*. In Proc. Int. Conf. Image Processing (ICIP), September 2005. 139

Table des figures

Résumé

De nombreuses applications en vision par ordinateur nécessitent la distinction et le suivi des différents objets vidéo constituant une scène dynamique. Dans le contexte de la post-production, la qualité visuelle des résultats est une contrainte si forte qu'un opérateur doit pouvoir intervenir facilement et rapidement pour guider efficacement les traitements. Le but de cette thèse est de proposer de nouveaux algorithmes de segmentation au sens du mouvement. Ce document est décomposé en deux parties. Dans la première partie, deux nouvelles méthodes séquentielles et semi-automatiques de segmentation de séquences d'images au sens du mouvement sont proposées. Toutes deux exploitent la représentation d'une scène par un ensemble de couches de mouvement. L'extraction de ces dernières repose sur différents critères (mouvement, couleur, cohérence spatio-temporelle) combinés au sein d'une fonctionnelle d'énergie minimisée par coupe minimale/flot maximal dans un graphe. La seconde partie présente une nouvelle méthode pour le partitionnement automatique d'un ensemble de trajectoires de points d'intérêt. Chaque trajectoire est définie sur un intervalle temporel qui lui est propre et qui correspond aux instants auxquels le point considéré est visible. Comparée à un mouvement estimé entre deux images, l'information de mouvement fournie par une trajectoire offre un horizon temporel étendu qui permet de mieux distinguer des objets dont les mouvements sont différents. Les méthodes sont validées sur différentes séquences aux contenus dynamiques variés.

Mots clés : analyse du mouvement, segmentation au sens du mouvement, trajectoire de point d'intérêt, approches semi-automatiques, coupe minimale/flot maximal.

Abstract

Many computer vision applications require the extraction and tracking of the different video objects that compose a dynamic scene. In the context of post-production, the constraint of visual quality of the results is so high that a user must be able to add inputs easily and quickly in order to efficiently help the process. This thesis aims at proposing new motion-based segmentation algorithms. It is composed of two main parts. First, two sequential and semi-automatic methods are proposed for segmenting a video shot in regions of similar motion. Both exploit the multi-layered motion representation of a dynamic scene. It is based on different criteria (motion, colour, spatio-temporal coherence) that are combined into an objective function minimized with Graph Cuts. The second part introduces a new approach to automatically cluster sets of feature point trajectories. Each trajectory has its own life-span. Compared to motion estimated between two images only, motion information provided by a trajectory offer an extended temporal horizon that allows a better distinction of objects whose motions are different. The algorithms are validated on several different sequences with various dynamic contents.

Keywords : motion analysis, motion-based segmentation, feature point trajectory, semi-automatic approaches, graph cuts.