Segmentation de personnes dans les images et les vidéos

Cyrille Migniot

UNIVERSITÉ DE GRENOBLE

THÈSE

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ DE GRENOBLE

Spécialité : **Signal, Image, Parole, Télécommunications**

Arrêté ministérial : 7 août 2006

Présentée par

Cyrille MIGNIOT

Thèse dirigée par **Jean-Marc CHASSERY**
et codirigée par **Pascal BERTOLINO**

préparée au sein **du laboratoire Grenoble Image Parole Signal et Automatique**
et de **l'école doctorale Électronique, Électrotechnique, Automatique, Télécommunications, Signal**

Segmentation de personnes dans les images et les vidéos

Thèse soutenue publiquement le **17 janvier 2012**,
devant le jury composé de :

Mr Bill TRIGGS
Directeur de recherche au laboratoire Jean Kuntzmann de Grenoble, Président
Mme Sylvie PHILIPP-FOLIGUET
Professeur des universités à l'ENSEA, Rapporteur
Mr Jean-Louis DILLENSEGER
Maître de conférences à l'université de Rennes 1, Rapporteur
Mr François BREMOND
Directeur de recherche à l'INRIA Sophia Antipolis, Examinateur
Mr Jean-Marc CHASSERY
Directeur de recherche au Gipsa-Lab, Directeur de thèse
Mr Pascal BERTOLINO
Maître de conférences au Gipsa-Lab, Co-Directeur de thèse

Résumé

La segmentation de personnes dans les images et les vidéos est une problématique actuellement au coeur de nombreux travaux. Nous nous intéressons à la segmentation de personnes debout. Pour cela, nous avons mis au point deux méthodes originales :
La première est une continuation d'une méthode de détection efficace. On réalise une pré-segmentation en associant aux segments de contour de l'image une valeur de vraisemblance en tant qu'élément d'une silhouette humaine par une combinaison d'histogrammes de gradients orientés (HOG) et de machines à vecteurs de support (SVM) prises à l'échelle des ces segments. Une recherche d'arbre optimal dans un graphe intégrant les données de la pré-segmentation permet de reconstruire la silhouette de la personne. Enfin, une utilisation itérative de ce processus permet d'en améliorer la performance.
La seconde méthode prend en compte l'interaction de l'utilisateur pour une image. Une coupe de graphe est guidée par un gabarit non binaire représentant une silhouette humaine. Nous proposons également un gabarit par parties pour s'adapter à la posture de la personne. Nous avons enfin transposé cette méthode à la segmentation de vidéos et la réalisation automatique de *trimaps*.

Mots clés : segmentation de personnes, analyse de segments de contour, segmentation interactive, coupe de graphe, gabarit non-binaire, génération automatique de trimap.

Abstract

Human segmentation in images and videos is currently a difficult task. We are interested in the upright people class. We have realized two original methods. After a review of the state of the art, this thesis presents these two methods.
The first one is the continuation of an effective segmentation method. The union of Histogramms of Oriented Gradients based descriptors (HOG) and of a Support Vector Machine (SVM) classifier at the contour segment scale provides a likelihood degree of being part of a human silhouette. The shortest path in a graph created from this data provides the segmentation.
The second method is interactive. A graph cut is guided by a non-binary template of silhouette that represents the probability of each pixel to be a part of the person. In a second time, a part-based template is computed to be adapted to the person posture. This method can be transformed to segment videos or automaticaly produce trimaps.

Keywords : silhouette segmentation, contour segment analysis, interactive segmentation, graph cut, template of silhouette, part-based template, trimap.

Remerciements

Ces travaux de thèse ont été réalisés au sein du département images et signaux du laboratoire GIPSA-lab à Grenoble sous la direction de Messieurs Jean-Marc Chassery et Pascal Bertolino.
Je tiens tout d'abord à les remercier pour leurs conseils et leur disponibilité ainsi que pour m'avoir permis de travailler sur ce sujet passionnant.
Je les remercie également pour la confiance qu'ils m'ont portée et pour la libertée scientifique dont j'ai disposée.
Je tiens à remercier tout particulièrement Madame Sylvie Philipp-Foliguet et Messieurs Bill Triggs, Jean-Louis Dillenseger et François Bremond d'avoir accepté de faire partie de mon jury et pour l'intérêt qu'ils ont bien voulu porter à mes travaux de thèse.

Merci à Messieurs Jean-Marc Chassery, Jean-Michel Dion et Jean-Marc Thiriet de m'avoir accueilli dans leur laboratoire.
Merci également à Jean-Marc Sache pour son soutien technique et sa bonne humeur.
Je tiens aussi à remercier Laurent Bonnaud, Alice Caplier, Rémy Chollet, Franck Corset, Yves Delnondedieu, Cédric Gerot, Hayate Khenouf, Annick Montanvert et Cécile Roisin pour leur aide et leur confiance durant mes vacations au sein de l'ENSE3, de PHELMA, de l'IUT1 et de l'IUT2.
Merci aux membres du conseil de laboratoire pour m'avoir acceuilli et m'avoir fait partager la vie administrative du laboratoire.

Un grand merci aux coincheurs du midi, aux rollistes, au GipsaDoc et à l'équipe de foot du Gipsa pour tous les bons moments passés.
Je n'oublie pas non plus de remercier tous les doctorants du GIPSA-lab pour leur contribution à l'ambiance de ce laboratoire : Bastien, Jeanne, Luiz, Olivier, Vincent, Jérémie, Jonathan, David, Mathieu, Éric, Benjamin, Damien, Rodrigo, Antoine, Gailene, Tomasz, Thibaud, Loïc, Émilie, Gabriel, Chloé, Lionel, Christian, Fabien, Hussein, Bertrand, Cédric et Nicolas.
Une pensée aussi aux gourmets de l'entracte : Antoine, Seb, Claudio, Philippe, Romain, Luc, Ali, Andy, Anthony, Joachim, Hilal, Klodjan, Nathalie, Juliette, Laure, Céline, Aurélie, Rachel, Richard, Pierrick, Nicolas, Delphine et Samuel. Une autre pensée pour les rollistes : Marc, Aurore, Édouard, Gaétan, Yoyo, Hadrien, Thomas et Louis.

Je remercie finalement mes parents et mes soeurs chez qui j'ai toujours pu trouver soutien et réconfort.

Table des matières

Chapitre 1

Introduction

L'homme est mortel par ses
craintes, immortel par ses désirs.

PYTHAGORE

Sommaire

Ces dernières années, le domaine du traitement et de l'analyse numérique des images a connu un essor considérable engendrant un nombre conséquent de travaux de recherche. En effet, la récente expansion des possibilités des ordinateurs a grandement facilité le traitement de masse d'informations numériques. De gigantesques quantités de calculs peuvent désormais être traités dans des temps toujours plus courts. Des traitements plus complexes sont alors accessibles, ce qui ouvre considerablement les perspectives.

L'image est aujourd'hui un objet des plus importants dans notre société. Elle est un média d'information (journaux papiers et télévisuelles, internet), publicitaire, artistique (cinéma, photographie,...) et social (Facebook, Picasa,...) parmi les plus utilisés. Beaucoup de travaux visent donc à en améliorer et en faciliter l'accessibilité. Que ce soit au niveau de son acquisition (appareil photographique, caméra numérique, webCam), de sa capacité de stockage et d'échange (formats, compression, ...) ou de son édition (par des logiciels tels que Photoshop ou Gimp pouvoir corriger des yeux rouges ou recadrer une photo).

Mais l'image est aussi un moyen de récupérer de l'information. Une machine doit souvent être la plus autonome possible. Il faut minimiser le travail de l'utilisateur. Pour cela, la machine doit connaître l'environnement qu'elle manipule. Elle prend alors l'information nécessaire par l'intermédiaire d'un certain nombre de capteurs. L'information visuelle donnée par une caméra en fait partie. Elle a notamment l'avantage de ne pas nécessiter de contact avec l'objet étudié. Il faut alors tirer l'information pertinente de l'image.
La reconnaissance et la segmentation des éléments d'une classe est alors un domaine très important du traitement de l'image. Elles permettent de reconnaître, de localiser et d'isoler l'objet d'intérêt. Ces informations sont nécessaires et suffisantes aux fonctionnements de nombreuses applications.
Pour cela, la machine a besoin d'un apprentissage. C'est-à-dire d'une connaissance sur la classe recherchée qui permette sa reconnaissance et guide sa segmentation. Elle peut, par exemple, prendre pour forme une base de données d'images correspondant à la classe recherchée. La taille de cet apprentissage dépend de la complexité de la classe.
Au vu de l'intérêt évident des communications entre la machine et l'homme, l'étude de la classe des personnes est l'une des plus importantes. Elle est donc le sujet d'un très grand nombre de travaux. Elle se trouve être aussi l'une des plus complexes du fait de la variété d'apparence et de posture qu'une personne peut prendre.

1.1 Applications

La détection et la segmentation de personnes dans les images et les vidéos est un domaine en pleine expansion qui regroupe bon nombre de projets de recherche. Cela s'explique en grande partie par le nombre important d'applications pratiques et industrielles qui en découlent. De nombreuses entreprises investissent ainsi dans ce domaine au vue des perspectives offertes.

1.1.1 Traitement temps réel

Une contrainte essentielle du traitement de données est le temps de traitement acceptable accordé à la machine. En effet, de nombreuses applications nécessitent un traitement temps réel afin d'avoir une interaction immédiate avec l'environnement. Cela permet de réagir à une action avant qu'elle ne soit finie ou bien d'anticiper un événement afin de l'empêcher d'advenir.
Parfois un décalage avec le réel peut être accepté mais le système sera plus attractif et plus agréable à l'utilisation s'il donne un résultat en un temps réduit.

La vidéo surveillance

Dans un lieu public, comme un hall de gare ou d'aéroport, il est utile de surveiller le comportement des personnes afin de prévenir les actes dangereux. De même, dans un espace privé comme une maison, une surveillance permet d'agir contre les intrusions indésirables. À partir du visionnage et de l'archivage des images issues d'un système de caméras, la vidéo-surveillance permet de répondre à ces problématiques essentielles car traitant de la sécurité (figure 1.1(a)).

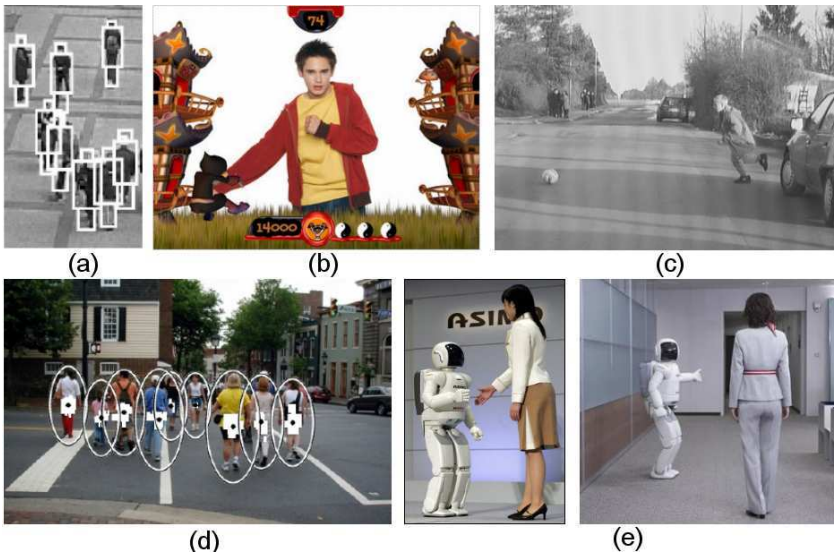

FIG. 1.1 – De nombreuses méthodes nécessitent un traitement temps réel. Que ce soit la vidéo surveillance (en (a) selon la méthode de Beleznai et al.[Beleznai et al., 2004]), les jeux vidéos avec intégration du joueur dans un environnement virtuel (en (b) "Eye-Toy" de PlayStation), l'assistance de conduite pour éviter les obstacles et les piétons imprudents (en (c)), le comptage de personnes (en (d) avec la méthode de Tuzel et al.[Tuzel et al., 2007]) ou l'intelligence artificielle (en (e) le robot Asimo de Honda).

L'acquisition se réalise à partir d'une caméra fixe ou dont le mouvement est controlé. De cette façon, l'arrière-plan est connu (ou facilement déterminé) et les régions d'intérêts que sont les différents premiers plans peuvent être aisément isolées. La détection est facilitée puisque l'arrière plan ne vient pas encombrer le signal et la segmentation est assez souvent déjà réalisée par l'isolement des régions d'intérêts continues.

Néanmoins, afin d'avoir une vue assez globale et de rester assez discrètes, les caméras de vidéo surveillance sont dans la plupart des cas placées en hauteur. Les silhouettes des personnes sont alors déformées (effet de contre champ) et de petite taille. De plus, les caméras de vidéo surveillance réalisent souvent des acquisitions de mauvaise qualité (pour des raisons de coût et de stockage des données). Enfin, dans les lieux publics, l'étude des scènes de foules implique de nombreuses occultations et de nombreux mouvements.

Système d'assistance de conduite

La sécurité des personnes est un domaine plus prioritaire que la sécurité des biens. C'est pourquoi l'industrie automobile investit beaucoup d'argent dans la recherche en vue de rendre les voitures plus sûres pour le conducteur mais aussi pour les piétons [Zaklouta, 2012]. Un système d'assistance de conduite permet de détecter la présence d'un piéton imprudent sur la route et donc d'éviter la collision (figure 1.1(c)). Mitsubishi a mis en oeuvre certains systèmes permettant cette opération en embarquant une caméra sur le devant du véhicule comme montré dans les articles [Gavrila and Philomin, 1999] [Gavrila and Giebel, 2002] [Elzein et al., 2003] [Gavrila and Munder, 2007].

La principale difficulté du domaine réside dans le changement rapide de l'arrière plan lorsque le véhicule se déplace et particulièrement lorsqu'il réalise un virage. De plus, le temps de réaction du système doit être très rapide (très proche du temps réel) puisqu'il faut le temps à l'utilisateur (ou à un système interne au véhicule) de définir puis de réaliser la manoeuvre qui permettra d'éviter la collision.
Néanmoins, le constructeur a ici la possibilité de choisir le système d'acquisition de l'image. Une image de bonne qualité peut permettre une détection avec peu de "non détection" ou de "fausse alarme". De plus, utiliser un système de caméra stéréo (avec deux caméras filmant la même scène avec un petit décalage spatial qui permet d'obtenir une information sur la profondeur des éléments visualisés) permet d'obtenir rapidement et sûrement les différentes régions d'intérêts (premiers plans). Le traitement et l'analyse de l'information sont alors grandement facilités et le temps de calcul diminué.

Comptage de personnes

Le comptage de personnes (figure 1.1(d)) permet de déterminer le nombre de personnes se trouvant dans un lieu ou bien le nombre de personnes passant à travers un passage (une porte, un pont, un couloir...). Cette application est utile pour évaluer le succès d'une exposition, l'affluence dans une salle, le nombre de personnes entrées dans un bus...
Pour ce genre d'application, les caméras du système d'acquisition peuvent être fixes et les divers premiers plans observés sont alors facilement isolés. Néanmoins, dans une foule, il est difficile de voir chaque élément de celle-ci en intégralité. Il y a souvent d'importantes occultations.
Dans ce cas, un suivi des personnes est nécessaire. En effet, ce traitement est essentiel pour éviter de compter plusieurs fois une même personne sur deux *frames* différentes. Dans le cas d'un passage, ce suivi permet aussi de s'assurer qu'un individu a bien traversé ce passage et ne s'est pas seulement arrêté devant. Il permet également de faciliter la détection. En effet,compte tenu des occultations, il est plus probable de réussir à réunir l'ensemble de la personne sur un ensemble de *frames* que sur une image fixe.

Jeux vidéo

Dans le domaine des loisirs, l'industrie des jeux vidéos est assez active pour offrir à sa clientèle des nouveautés attractives. Avec "EyeToy", PlayStation a, par exemple, intégré l'image du joueur dans le jeu afin qu'il apparaisse sur l'écran de sa télévision. Le principe est de placer une caméra sur son téléviseur, l'image du joueur est alors isolée et placée dans un environnement virtuel correspondant au jeu (figure 1.1(b)). Le joueur peut alors interagir avec cet environnement en déplaçant ses bras.

Ici aussi, le constructeur a la possibilité de choisir son système d'acquisition et donc d'utiliser la stéréo afin de délimiter facilement le premier plan. Le système a aussi l'a-priori que le joueur doit se situer devant la caméra, à une distance connue afin d'apparaître au centre de son téléviseur. Le traitement est plus simple, ce qui lui permet d'être temps réel. La seule difficulté est de bien segmenter la personne. Il ne faut pas, en effet, qu'une partie d'un membre manque ou bien qu'un objet de l'arrière plan soit ajouté lors de l'intégration dans le nouvel environnement.

Intelligence artificielle

Le but de la robotique est de concevoir des robots effectuant des tâches déterminées en s'adaptant à leur environnement (figure 1.1(e)). Une des perspectives les plus novatrices est la création d'une intelligence artificielle dans le sens où le robot prend des décisions de façon indépendante. Dans cette optique, il est important qu'il puisse évaluer la situation dans laquelle il se trouve et donc analyser son signal visuel. Détecter et reconnaître une personne est alors une tâche prioritaire puisque le but final est qu'il puisse interagir avec un être humain pour pouvoir l'aider.
Par exemple, Honda a créé le robot Asimo qui sait adapter son comportement avec celui des personnes présentes dans son environnement. Pour cela il doit initialement détecter les personnes et reconnaître leur comportement et leurs actions.

Dans la robotique, le système d'acquisition est libre mais le traitement doit être temps réel. De plus, la détection doit être certaine puisque le robot a pour fonction d'être en contact avec des personnes. Un mauvais comportement de sa part peut s'avérer dangereux.

1.1.2 Post-traitement

Pour d'autres applications, le temps de calcul peut être plus important. C'est le cas de toutes les méthodes utilisant un post-traitement. Après l'acquisition de l'image ou de la vidéo, un traitement plus long sur l'ensemble du signal est réalisé. Dans la plupart des cas, l'aboutissement du travail est d'améliorer la qualité du rendu ou bien de transformer la réalité pour créer une image ou une vidéo irréalisable autrement.
Une interaction de l'utilisateur est souvent nécessaire mais par commodité, productivité et besoin de précision elle se doit d'être la plus réduite possible.

FIG. 1.2 – Exemple de post-traitements. L'indexation de vidéos cliquables permet de lier une action au passage de la souris sur un objet d'une vidéo (en (a) cliquer sur le pull donne accès au site de vente de ce pull). La création d'avatar donne de bons effets de cinéma (en (b) des capteurs sont placés sur un acteur pour réaliser le mouvement d'un personnage virtuel). Enfin le photo montage est un effet très utilisé (en (c) un présentateur météo est filmé sur un fond bleu, puis est segmenté et enfin rajouté sur la carte qu'il commente).

Indexation de vidéos et vidéos cliquables

L'objectif est ici prioritairement d'enrichir le contenu de la vidéo pour des vues publicitaires ou autre. L'idée est de créer une interaction possible de l'utilisateur avec les éléments composant une vidéo (figure 1.2(a)). Une vidéo cliquable est une vidéo sur laquelle l'utilisateur peut sélectionner des éléments la composant et où ces éléments sont associés à un lien. L'aboutissement est qu'une personne qui visionne une vidéo sur internet et qui repère un objet qui l'intéresse, puisse cliquer sur cet objet pour être redirigé sur une page Web dont l'objet cliqué est le sujet (une personne clique sur un vêtement et est redirigée vers le site de vente en ligne où il peut acheter ce vêtement). L'objet en question peut être une personne (par exemple une personne célèbre dont le lien associé mènerait à sa biographie).
Une grande précision n'est pas obligatoire car l'utilisateur a tendance à cliquer au centre de l'objet et non à ses extrémités. Par contre, le plus souvent, le traitement doit se faire sur des vidéos existantes et ni le format ni le système d'acquisition ne peuvent alors être choisis. Le traitement doit alors s'adapter à tout genre de vidéos.

Effets spéciaux pour le cinéma

Dans le monde du cinéma, il existe une forte demande en effets spéciaux. Pour faire croire à des situations ou des personnages de plus en plus insensés, il est indispensable que la modification de l'image soit la plus réaliste possible.
Assez couramment, un personnage doit être inséré dans un autre décor (si le décor ne peut être construit ou, par exemple, s'il n'est pas à la même échelle que le personnage). Pour cela est réalisé un vidéo-montage. C'est-à-dire que le premier plan d'une vidéo doit être superposé à une seconde vidéo.
Ce procédé est également utilisé lors de la réalisation d'une émission météorologique (figure 1.2(c)). Le présentateur est filmé puis ajouté à la carte affichant le temps ou les températures.

Afin de réaliser une segmentation d'un premier plan en vue d'un vidéo-montage, la méthode la plus couramment utilisée est celle dite du fond bleu [Smith and Blinn, 1996]. L'élément à segmenter est placé devant un fond de couleur bleu lors de l'acquisition. L'arrière plan étant précisément connu, il est assez aisé de séparer le premier plan de l'arrière plan.
Pour être réaliste, la segmentation doit très précise car une petite erreur sera très visible et décrédibilisera l'effet. Une méthode souvent utilisée est le *matting*. Aux bords du premier plan, le vidéo-montage sera un mélange des couleurs du premier plan et du nouvel arrière plan. Ainsi la transition est plus douce et l'effet plus réaliste.

Création d'avatar

Un autre effet assez apprécié dans le cinéma est la création de personnages virtuels. La création et la mise en animation de personnages entièrement réalisés par ordinateur ouvrent les perspectives de personnages plus originaux. Une des difficultés majeures de ce procédé est le mouvement donné à ce personnage. Pour qu'il soit crédible, le mouvement humain est imité. Dans cette optique, un squelette du personnage est créé avec un certain nombre d'articulations qui définissent les mouvements possibles. Une personne

est ensuite filmée réalisant les mouvements que le personnage virtuel doit réaliser. En associant à chaque articulation du squelette une articulation de la personne filmée et en enregistrant le mouvement de la personne au niveau de ses articulations, le mouvement de la personne est transmis au personnage virtuel (figure 1.2(b)). Ce personnage virtuel est appelé un avatar.
Pour faciliter ce procédé, des capteurs sont placés sur la personne au niveau des articulations recherchées. La détection est alors évidente. Une méthode automatique pourrait cependant être créée afin de fournir une application tous publics.

1.2 Objectif

L'objectif de cette thèse est de mettre au point et d'implémenter en vue de tests, de nouvelles méthodes permettant la segmentation de personnes dans les images fixes et les séquences vidéo. Les thèmes traités sont donc l'étude des caractéristiques discriminantes de la classe des personnes, la reconnaissance des ses éléments puis leur segmentation précise.

Les méthodes créées se doivent d'avoir un maximum de robustesse.

- Le logiciel de tests doit s'approcher le plus souvent des résultats optimaux sans interactions (ou avec très peu) avec l'utilisateur. Un novice doit aussi pouvoir utiliser la méthode sans avoir à évaluer les valeurs des paramètres les plus aptes à optimiser le traitement.
- Le système d'acquisition de l'image ou de la séquence vidéo doit être très simple. Pas de caméra stéréo ou haute définition facilitant le traitement. La méthode doit pouvoir traiter correctement une acquisition d'une caméra amateur ou d'une petite caméra de surveillance (donnant un rendu de faible qualité).
- Le sujet étudié doit être soumis au minimum de contraintes possibles : pas d'arrière plan fixe ou connu, une personne immobile, marchant ou courant, pas d'habillage spécial ou de spécification physique (couleur de peau)...

Au final, le but est de réaliser des méthodes fonctionnant correctement pour le plus de cas possibles. En effet, pour que les méthodes puissent être utilisées pour un usage familial ou sur des images et des vidéos déjà acquises (et dont on n'a pas pu contrôler l'acquisition), il est important d'avoir le moins de contraintes possibles.

1.2.1 Principales difficultés inhérentes à cet objectif

Selon les situations, la qualité de l'acquisition ou l'environnement autour du sujet étudié, la difficulté pour un algorithme de reconnaître une certaine classe est plus ou moins importante. Voici les obstacles les plus fréquemment rencontrés :

- Une partie de l'objet peut être cachée par un élément de l'environnement se trouvant entre le sujet recherché et le système d'acquisition. C'est ce qu'on appelle une occultation. Si la méthode se base sur une forme, il faut qu'elle puisse la reconnaître même si elle est tronquée. Le problème peut aussi causer l'échec de la détection si cette dernière se base sur la reconnaissance de toutes les sous-parties.

- La caméra peut aussi bouger lors de l'acquisition. L'arrière plan n'est plus fixe et il est alors difficile de trouver une référence et de focaliser l'étude sur une partie de l'image en particulier. De plus, certaines sous-parties de l'arrière plan peuvent avoir des mouvements rapides et complexes comme par exemple le mouvement des feuilles d'un arbre secouées par le vent.
- Il se peut également que l'arrière plan soit particulièrement chargé, comme par exemple dans une scène urbaine. Il est alors compliqué d'isoler certaines régions uniformes cohérentes. De même, les intempéries (pluie, neige...) peuvent nuire à la clarté de l'image et donc à la détection.
- Les couleurs d'un objet sont modifiées selon l'éclairage qu'il subit. Il est donc difficile d'associer une partition de couleur à une classe. De plus, si l'objet à reconnaître est de couleur proche du fond (camouflage) la détection sera d'autant plus délicate.
- Enfin, selon l'usage, il peut être nécessaire que le système de détection fonctionne rapidement, voire en temps réel. Par exemple pour l'assistance de conduite, il est nécessaire que le système prenne en compte l'obstacle avant de l'avoir percuté. Le temps de traitement doit alors être minimisé.

De toutes les classes, celle des personnes est l'une des plus complexes. En effet, de multiples difficultés sont inhérentes à la forme et au comportement humain.

- Tout d'abord, fort de ces 250 degrés de liberté, l'homme est un objet non rigide et non isotrope qui peut prendre un nombre important de positions. Lui associer une forme fixe est donc délicat.
- Dans de nombreux cas (surveillance de rue ou de hall), les personnes se regroupent pour former une foule. Une difficulté supplémentaire est alors de différencier chaque individu.
- Enfin, la variété de couleurs et de textures des vêtements pouvant être portés par une personne rend ces informations peu discriminantes. De plus, une personne peut avoir des accessoires, comme une mallette ou un parapluie, qui modifient sa silhouette et peuvent cacher des régions utiles à la détection.

Tous ces points démontrent la difficulté de la détection et de la segmentation de personnes. Voila pourquoi la recherche de caractéristiques discriminantes pour des détecteurs efficaces et précis est assez répandue et influence beaucoup d'études.

1.2.2 Précision de l'objectif

Ces dernières années, un nombre conséquent de travaux sur la détection de personnes ont été effectués. De multiples points de vue, modélisations et méthodes ont été testés. Certains ont donné des résultats plutôt satisfaisants.
En utilisant une combinaison de machines à vecteurs de support (SVM) et d'histogrammes de gradients orientés (HOG), Dalal a mis au point une méthode efficace et précise [Dalal, 2006]. En effet, cette méthode permet une détection dans 89% des cas avec un taux de fausse détection par fenêtre de 1 pour 10000. Néanmoins cette méthode ne permet pas la segmentation de la personne détectée. Un cadre (ou fenêtre de détection) est seulement placé au lieu de détection. Nous reviendrons plus en détail sur cette méthode au chapitre 3.
Pour segmenter un objet dans une image, il existe de nombreuses méthodes tel que le *watershed* ou le *snake*. Ces méthodes visent à regrouper des entités cohérentes ou à suivre la continuité d'un contour. Néanmoins, si un tel procédé était utilisé en sortie d'une méthode

de détection, il ne prendrait pas en compte la connaissance *a priori* sur la classe recherchée et ne serait pas amélioré par l'apprentissage effectué. En outre, la classe des personnes étant, comme nous l'avons vu, particulièrement délicate à traiter, ces méthodes devraient s'avérer inefficaces. En effet, une personne étant le groupement d'éléments très différents (manque de cohérence) comme des habits, des membres avec de la peau, des cheveux... ces méthodes tendraient à délimiter ces différents éléments et non la silhouette dans son ensemble.
Il existe néanmoins des méthodes traitant simultanément de la détection et la segmentation (section 2.2). Toutefois, elles se basent sur des principes de détection qui ne sont pas forcement les plus efficaces (comparaison avec un gabarit, détection de sous-parties...)
Dans ce document, nous limitons notre étude aux cas des personnes en position debout. Nous avons choisi, dans un premier temps, de partir d'une méthode de détection parmi les plus performantes et les plus éprouvées, puis de lui associer une nouvelle méthode de segmentation. Les outils utilisés par les deux méthodes sont les mêmes afin d'optimiser la cohérence ainsi que le temps de traitement.
La segmentation de personnes est souvent utilisée pour des post-traitements. Un second objectif est alors de permettre une interaction ergonomique et cohérente avec l'utilisateur. Nous avons donc décidé de réaliser également une méthode originale de segmentation de personne répondant à ces contraintes.

1.3 Vue globale de notre approche

Dans un premier temps, nous avons focalisé notre étude sur l'élaboration de la segmentation à partir des éléments calculés par la méthode de détection de Dalal [Dalal, 2006]. Cette méthode utilise les histogrammes de gradients orientés comme descripteur et les machines à supports de vecteurs comme classifieur.
Les histogrammes de gradients orientés relèvent les concentrations locales de pixels de contours d'une image selon différentes orientations. L'ensemble des pixels de contour peut être modélisé en une suite de segments. Un segment est défini par sa localisation et son orientation. Une première étape est alors d'utiliser le couplage descripteur-classifieur de façon locale afin d'évaluer les segments les plus vraissemblables en tant qu'élement d'une silhouette humaine. On réalise alors une pré-segmentation.
Une silhouette peut être modélisée comme une suite bouclée de segments. Un graphe est alors créé où les noeuds sont les segments de contours. Les arcs représentent les liaisons possibles entre segments spatialement proches et sont pondérés par les valeurs calculées lors de la pré-segmentation. L'algorithme de plus court chemin de Dijkstra permet alors de déterminer le cycle qui donne les limites de la silhouette recherchée. Le processus est ensuite réalisé de façon itérative pour éliminer les erreurs et donc améliorer la précision.

Dans un second temps, nous avons voulu permettre l'interaction avec l'utilisateur. La méthode de coupe de graphe de Boykov [Boykov and Jolly, 2001] est une des méthode de segmentation avec interaction de l'utilisateur les plus performantes. Dans le graphe, chaque noeud est un pixel de l'image. Des arêtes de voisinage relient ces noeuds pour modéliser l'adjacence des pixels. Deux noeuds particuliers appelés source et puits sont rajoutés au graphe afin de représenter le premier et l'arrière plan. Des arêtes de liaison relient chaque pixel au puits et à la source. Des pondérations sont associées à toutes les

arêtes. La coupe du graphe qui minimise la somme des pondérations des arêtes coupées est alors calculée et sépare les pixels de premier et d'arrière plan. Nous introduisons un gabarit non binaire donnant la forme d'une personne, qui est bien discriminante de la classe, dans la pondération des arêtes de liaison afin de spécifier la méthode à la segmentation de personnes. Puis, un gabarit par parties est déterminé par des coupes de graphe successives sur des sous-parties de l'image.

Vue comme un objet tri-dimensionnel, une séquence vidéo peut être segmentée par le même principe. Enfin, en recherchant la région de transition entre premier et arrière plan, la méthode permet la génération automatique d'un trimap précis.

1.4 Structure du document

Le travail effectué dans cette thèse a bien évidemment commencé par une recherche bibliographique des avancées techniques réalisées dans le domaine étudié. Dans le chapitre 2, une présentation de l'avancée des recherches et un état de l'art des méthodes existantes de détection et de segmentation est fourni. Ils permettent de justifier notre choix de partir d'une méthode de détection efficace pour se concentrer sur la segmentation. Les deux méthodes proposées peuvent alors être présentées. La première se décompose en deux étapes. Dans un premier temps, une pré-segmentation permet de donner aux segments de contour une valeur relative à leur vraisemblance en tant qu'élément d'une silhouette humaine. Le couplage HOGs SVMs utilisé par Dalal [Dalal, 2006] pour la détection est pris de façon locale pour définir une information plus précise spatialement. Le chapitre 3 présente cette étape ainsi que son implémentation et les résultats qu'elle forme.
Ensuite, la silhouette doit être reconstituée à partir des segments de contours favorisés par l'opération précédente. Un graphe est alors créé dont les noeuds représentent les différents segments de contours de l'image et dont les arcs sont pondérés par les valeurs obtenues lors de l'étape de pré-segmentation. Le cycle de segments obtenu par un algorithme de recherche de plus court chemin dans ce graphe définit notre première méthode de segmentation de personnes. Les détails de cette étape ainsi que son implémentation et les résultats obtenus sont présentés dans le chapitre 4.
Notre seconde méthode a l'avantage de proposer une interaction facile et efficace avec l'utilisateur. Une coupe de graphe telle que définie par Boykov [Boykov and Jolly, 2001] est utilisée comme méthode de segmentation. Nous introduisons alors des gabarits non binaires comme information sur la forme de la personne. Le chapitre 5 présente et évalue cette nouvelle méthode de segmentation de personnes. Cette méthode étant efficace, nous avons dans le chapitre 6 élargi le principe à de nouvelles applications : tout d'abord l'étude de séquences vidéo où l'usage du gabarit pour chaque *frame* demande une adaptation pour être efficace, ensuite la recherche automatique de trimap dans le but de réaliser un photomontage réaliste par matting. L'information de contour préalablement utilisée pour la continuité de la segmentation est désormais discriminante de la région à segmenter. Ces deux applications sont aussi évaluées dans ce chapitre.
Enfin, le chapitre 7 tire une conclusion du travail effectué et annonce les perspectives possibles avec nos méthodes.

Chapitre 2

État de l'art

> Le contraire d'une vérité banale, c'est une erreur stupide. Le contraire d'une vérité profonde, c'est une autre vérité profonde.
>
> Niels Bohr

Sommaire

La détection d'objets, et en particulier des personnes, dans les images et les vidéos a concentré beaucoup de travaux de recherche en traitement de l'image et reconnaissance de formes ces dernières années. Il s'agit d'un problème d'importance utile dans de nombreuses applications. Ce chapitre vise à passer en revue les différentes méthodes de détection automatique d'objet et de leur segmentation, en centrant particulièrement sur la classe des personnes.

Dans un premier temps, une étude sera faite des méthodes existantes de détection seule d'éléments d'une classe dans des images puis des vidéos dans le paragraphe 2.1. Ensuite, le paragraphe 2.2 réalise une énumération des principales approches de détection et segmentation simultanées. Enfin le paragraphe 2.3 délivre une conclusion de l'état des connaissances déjà acquises et présente les motivations qui ont guidé la spécification du sujet de thèse.

2.1 Détection de personnes dans les images

La reconnaissance de personnes dans les images est un problème peu évident. En effet, la variété de poses, de textures et de couleurs que possède cette classe est très importante. Certaines méthodes ont cependant été mises au points afin de résoudre ce problème.
Un principe qui fonctionne efficacement, et qui est donc le plus souvent utilisé, est celui d'associer un descripteur avec un classifieur. Le descripteur isole les informations les plus discriminantes de la classe recherchée et le classifieur compare ces informations avec une base de données d'exemples d'éléments de la classe recherchée afin de déterminer si l'élément testé ressemble ou non aux exemples. Cette approche est expliquée en section 2.1.1. Puis, en section 2.1.2, sont présentées les méthodes n'utilisant pas cette vision mais étant néanmoins performantes.

2.1.1 Structure descripteur-classifieur

Un grand nombre de publications définissent des méthodes de détection de personnes utilisant la même structure de fonctionnnement (cf figure 2.1).

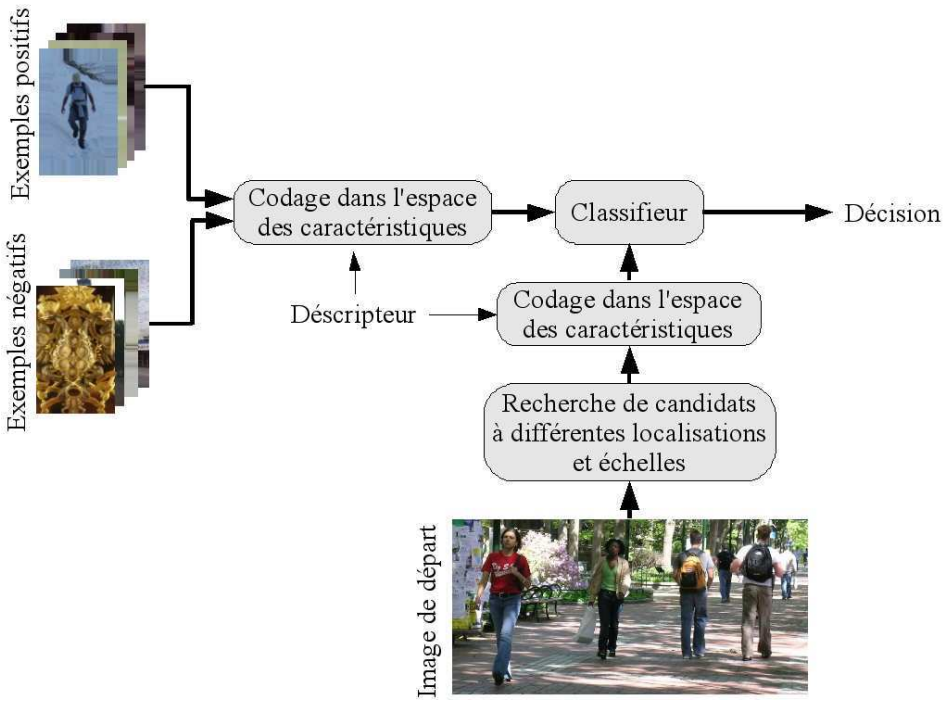

Fig. 2.1 – Fonctionnement de la structure descripteur-classifieur

Le classifieur est réalisé à partir d'une base de données d'exemples positifs et négatifs. Un exemple positif est un élément de l'ensemble à tester (par exemple une image d'une certaine taille) qui contient un individu de la classe recherchée (par exemple une personne). Un exemple négatif est un élément de l'ensemble à tester qui ne contient pas d'individu de la classe recherchée. L'objectif du classifieur est alors de comparer un élément à tester aux exemples de la base de données pour savoir s'il se rapproche plus des exemples positifs (et contient donc certainement un élément de la classe recherchée) ou des exemples négatifs (et ne contient certainement pas d'élément de la classe recherchée).

La quantité d'information contenue dans une image est très importante et souvent redondante. Parmi ces informations, la plus grande partie n'aide pas à caractériser un élément de la classe recherchée. Or l'abondance d'informations dégrade les performances de classification. L'objectif du descripteur est alors d'isoler les informations qui sont discriminantes de la classe recherchée.

Recherche de candidats

L'aboutissement de la détection est de déterminer la localisation spatiale d'une personne. Dans la grande majorité des cas, celle-ci est définie par une fenêtre de détection encadrant la personne. La première étape de la structure présentée dans ce paragraphe est donc de rechercher des fenêtres candidates à contenir une personne.

La solution la plus simple est de prendre en considération toutes les fenêtres possibles dans l'image [Dalal and Triggs, 2005][Sharma and Davis, 2007], c'est-à-dire à toutes les positions et échelles possibles, par un balayage sur l'ensemble de l'image initiale. Cette solution est la plus sûre pour ne pas oublier de candidats mais peut prendre un temps important car elle demande une quantité de tests très importante.
Pour réduire le nombre de candidats, il est possible de réaliser une soustraction d'arrière plan [Lin et al., 2007a][Lu et al., 2008]. Ainsi les premiers plans sont isolés. Les formes qui sont alors trouvées sont des candidats possibles. Il se peut néanmoins qu'une forme contienne plusieurs objets et soit donc un ensemble de candidats ou qu'un même objet soit représenté par plusieurs formes disjointes. Il faut alors arriver à fissionner ou à fusionner ces formes par exemple par une méthode de clustering.
Si l'étude se base sur des séquences vidéo, les objets en mouvement sont alors de bons candidats qui sont facilement isolés par la différence d'intensité entre *frames* successives [Rodriguez and Shah, 2007]. Néanmoins, si la caméra n'est pas fixe lors de l'acquisition, l'arrière plan lui aussi est en mouvement. L'arrière plan doit donc être préalablement estimé.
Enfin, si le système peut disposer d'une caméra double (stéréo) [Kang and whan Lee, 2002][Zhao and Thorpe, 2000], il est possible de déterminer très rapidement la profondeur des objets filmés et donc de les isoler pour les utiliser directement en tant que candidats (une personne est dans son ensemble sur un même plan). Ce procédé permet un traitement très rapide mais nécessite un système d'acquisition spécial. Par exemple pour l'assistance de conduite, la voiture peut être munie d'une caméra spéciale de façon à minimiser le temps de calcul.

Caractéristiques de l'image

Une fois que les candidats ont été localisés, les caractéristiques principales doivent être extraites. L'aboutissement est d'isoler les caractéristiques qui vont décrire le mieux une personne et qui sont les plus propres à la classe "personne".

Comme la couleur et la texture ne peuvent pas raisonnablement être prises en compte puisque les vêtements d'une personne peuvent prendre une gamme trop importante de ces caractéristiques, l'idée première et de prendre en compte les caractéristiques de forme. En effet, la silhouette humaine est assez discriminante tout en restant assez

stable. Dans la plupart des cas, l'étude se fera alors en isolant les contours par étude des gradients (par exemple par filtrage de Canny [Alonso et al., 2007]) puis en comparant avec le modèle d'une silhouette humaine.
Le mouvement est aussi un facteur caractéristique des personnes. Pour cela doit être pris en compte soit le flot optique [Sidenbladh, 2004], soit le mouvement des différentes parties de l'anatomie humaine [Mori et al., 2004]. Avec le flot optique, l'étude se portera sur le mouvement global de la silhouette. Pour le mouvement des différentes parties, un suivi est d'abord réalisé pour former un squelette de l'objet étudié (un modèle). C'est sur le mouvement des composantes de ce modèle que portera l'étude.
La texture donne de précieuses informations [Munder et al., 2008] mais ne suffit pas à caractériser une personne. Elle peut néanmoins être associée à une autre caractéristique pour en améliorer la pertinence. Brox et Weickert [Brox and Weickert, 2004] utilisent le flot variationnel total pour déterminer l'échelle locale et donc la taille des régions. Associé aux caractéristiques données par la matrice du moment d'ordre deux de l'intensité, il donne une mesure bien descriptive qui permet la segmentation des régions.
Plusieurs autres approches ont été testées mais sont moins reprises. La périodicité du mouvement de balancier des jambes [Wohler et al., 1998] [Shashua et al., 2004] est assez caractéristique d'une personne mais nécessite que ladite personne soit en phase de marche ou de course (néanmoins cette méthode permet de déterminer dans laquelle de ces deux phases l'individu se trouve) et qu'elle marche ou court plus ou moins perpendiculairement à l'axe optique. Les symétries (surtout verticales) peuvent être prises en compte, mais elles dépendent de l'orientation du sujet.
Des paramètres liés à la forme de la silhouette comme la concentration ($\frac{perimetre^2}{aire}$) ou son allure comme la minceur (projection verticale du squelette sur projection horizontale du squelette) sont aussi utilisables [Dai et al., 2005].
Enfin, si l'acquisition est réalisée par une caméra infra-rouge, la notion de chaleur peut rentrer en compte [Bertozzi et al., 2004] [Dai et al., 2005]. Or la chaleur corporelle est bien propre aux créatures vivantes. Cette notion ne se suffit par contre pas à elle-même et doit être associée à des caractéristiques de formes.

Les descripteurs

Une fois le choix des caractéristiques à prendre en compte effectué, un modèle doit être trouvé pour les acquérir dans l'image. Le but ici est de trouver une méthode pour transformer une image en un vecteur de caractéristiques. Celui-ci définit les caractéristiques choisies de l'image.
Une simple liste des intensités de chaque pixel peut être choisie, mais certains procédés permettent une meilleur caractérisation.

- **Les ondelettes de Haar** sont assez souvent utilisées dans le traitement d'image. L'idée est de projeter l'image sur une base d'ondelettes de Haar. L'image est décomposée en une image d'approximation et en images représentant les détails perdus (cf figure 2.2).
  En obtenant une image de moins en moins détaillée, les détails superflus sont éliminés pour donner une allure qui caractérise mieux les formes. Ainsi la quantité d'information est grandement réduite (ce procédé est d'ailleurs aussi utilisé pour la compression) pour ne garder que les caractéristiques de formes. Voila pourquoi

elle est assez souvent utilisée [Oren et al., 1997] [Papageorgiou and Poggio, 2000] [Mohan et al., 2001] [Elzein et al., 2003].

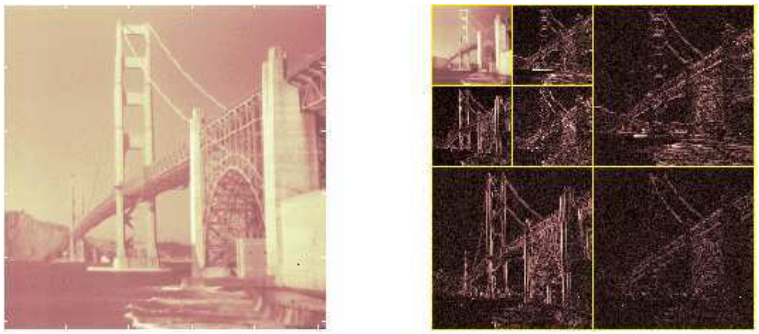

FIG. 2.2 – À gauche : image originale. À droite : décomposition selon les ondelettes de Haar en une image d'approximation (en haut à gauche) et en images des détails selon certaines directions et échelles.

- **Les histogrammes de gradients orientés.** L'image est dans un premier temps divisée en cellules qui sont elles-mêmes regroupées en blocs. Il est possible (et même recommandé) de faire se chevaucher les blocs. Chaque bloc est normalisé en intensité. Un filtre pour obtenir le gradient est alors appliqué et la moyenne des orientations des gradients est associée à chaque cellule. Le vecteur des caractéristiques est ensuite formé par la suite, pour chaque bloc, des histogrammes des gradients orientés des cellules. Ce vecteur permet de bien caractériser la forme de l'objet. Les histogrammes de gradients orientés (ou HOG) sont couramment utilisés [Dalal and Triggs, 2005] [Mikolajczyk et al., 2004] [Zhu et al., 2006] [Abd-Almageed et al., 2007] [Bertozzi et al., 2007] [Alonso et al., 2007] [Shet et al., 2007] [Corvee and Bremond, 2010] [Felzenszwalb et al., 2010] [Zaklouta, 2012]. Il est aussi possible de remplacer dans cette méthode les gradients par le flot optique entre *frames* successives et ainsi de caractériser le mouvement ou même d'associer les deux notions.

- **Analyse en composantes principales.** L'analyse en composantes principales (ACP) est une méthode mathématique d'analyse des données qui consiste à rechercher les directions de l'espace qui représentent le mieux les corrélations entre n variables aléatoires. Lorsqu'on veut compresser un ensemble de N variables aléatoires, les n premiers axes de l'ACP sont un meilleur choix, du point de vue de l'inertie. Cette analyse permet donc de réduire le vecteur de caractéristiques de l'image [Yoon and Kim, 2004] [Munder et al., 2008].

- **Champ de Riemann** Pour réaliser une bonne comparaison, une corrélation est efficace. Néanmoins les caractéristiques à utiliser ici se présentent sous forme de vecteur (intensité, contour, ...). La corrélation donne donc une matrice qui ne peut être utilisée par les classifieurs classiques. Le champ de Riemann permet de réduire la matrice en un vecteur de caractéristiques qui peut ensuite être utilisé [Tuzel et al., 2007].

- **Descripteurs SIFT** (Scale Invariant Feature Transform) Ils transforment une image en une collection de vecteurs de caractéristiques locales correspondant aux points pertinents des formes principales. Chacun d'eux est invariant par translation, rotation, changement d'échelle et illumination de l'image. La sélection effectuée est très proche de celle effectuée par la vision des primates.
  Cette méthode a été introduite et appliquée à la reconnaissance d'objet par Lowe[Lowe, 1999].

- **Descripteurs de Fourier.** Chaque point du contour d'une forme est passée en complexe (une des coordonnées en x ou y pour les réels et l'autre pour les complexes) pour former une fonction. Les descripteurs de Fourrier sont les coefficients de la transformée de Fourier discrète de cette fonction. En ne gardant que les premiers coefficients, une bonne caractérisation des formes est effectuée. Cette méthode est utilisée par Toth et Aach [Toth and Aach, 2003].

Les classifieurs

La fonction du classifieur est de donner une décision sur l'appartenance du candidat à la classe recherchée. Elle repose sur une base de données. En prenant en entrée les caractéristiques des exemples contenant un individu de la classe (ici une personne) et des exemples ne contenant pas d'individu de la classe, le classifieur doit déterminer de qui les caractéristiques de l'image candidate sont les plus proches.
Dans la plupart des cas, cette étape est la dernière du processus puisqu'une fois reconnues par le classifieur, il suffit d'afficher les fenêtres de détection.

- **Machine à vecteurs de support (SVM)** Cette méthode élaborée par Vapnik[Vapnik, 1995] vise à déterminer un hyperplan séparateur entre les espaces des deux classes à séparer. L'idée est de maximiser la marge, c'est-à-dire la distance entre la frontière de séparation et les échantillons les plus proches. Pour cela, l'algorithme transforme l'espace de représentation des données d'entrées en un espace de plus grande dimension, dans lequel il est probable qu'il existe une séparatrice linéaire.
  Du fait de sa très grande efficacité, cette méthode est très couramment utilisée notamment dans le domaine de la détection de personnes [Oren et al., 1997] [Papageorgiou and Poggio, 2000] [Mohan et al., 2001] [Kang and whan Lee, 2002] [Yoon and Kim, 2004] [Sidenbladh, 2004] [Karam et al., 2004] [Dalal and Triggs, 2005] [Dai et al., 2005] [Zhu et al., 2006] [Alonso et al., 2007] [Bertozzi et al., 2007] [Shet et al., 2007].

- **AdaBoost** (adaptative boosting) [Freund and Schapire, 1995] est un algorithme d'apprentissage assez utilisé [Viola and Jones, 2001] [Mikolajczyk et al., 2004] [Laptev, 2006]. Il peut être associé avec un autre algorithme d'apprentissage pour en améliorer la performance [Shashua et al., 2004] [Zhu et al., 2006].
  Le principe est issu de la combinaison de classifieurs. Par itérations successives, la connaissance d'un classifieur faible est ajoutée au classifieur final. Le classifieur ajouté est pondéré par la qualité de sa classification : plus il classe bien, plus il sera important. Les exemples mal classés sont boostés pour qu'ils aient davantage d'importance vis à vis de l'apprenant faible au prochain tour, afin qu'il pallie ce manque.

- **Les réseaux de neurones** [Hérault and Jutten, 1994] représentent un système d'apprentissage par expérience s'inspirant du fonctionnement de neurones humains. Si les valeurs observées sont représentées par le neurone sous forme d'un vecteur, le neurone réalise alors un découpage de son espace d'entrée (l'espace vectoriel auquel appartient le vecteur d'observation) en deux zones : la zone d'activité dont les vecteurs donnent une sortie égale à 1 et la zone d'inactivité dont les vecteurs donnent une sortie égale à 0. Comme le calcul effectué est en fait linéaire, la séparation l'est aussi. Les coefficients synaptiques et le seuil définissent l'équation d'un hyperplan qui est la frontière de la séparation entre les deux zones.
  Les réseaux de neurones permettent ainsi la classification de personnes [Wohler et al., 1998] [Zhao and Thorpe, 2000] [Gavrila and Giebel, 2002] [Toth and Aach, 2003] [Gavrila and Munder, 2007] [Munder et al., 2008].

Variation de la méthode

La structure descripteur-classifieur présentée précédemment est la plus couramment utilisée. Néanmoins, elle est parfois reprise avec quelques variantes qui en changent quelque peu le fonctionnement ou qui précisent certains points.

- **Détection par parties** Plutôt que de faire une étude globale de la forme d'une personne, plusieurs méthodes [Mohan et al., 2001] [Mikolajczyk et al., 2004] [Ioffe and Forsyth, 2001] [Ramanan et al., 2005] [Alonso et al., 2007] [Corvee and Bremond, 2010] préfèrent chercher à détecter tout d'abord les sous-parties de l'anatomie humaine comme par exemple la tête, le torse, les bras, les jambes voire les cuisses et les avant-bras. Leur détection se fait d'une façon identique à la détection globale (figure 2.3). Ensuite, la logique de leur placement et les relations entre sous-parties permettent de donner une décision sur la présence ou non d'une personne.
  Une détection par parties définit un modèle bien déformable qui demande souvent un apprentissage compliqué. Felzenszwalb [Felzenszwalb et al., 2010] réalise une détection par HOG de parties reliées selon une organisation en étoile. Un algorithme de SVM dit latent donne un apprentissage itératif de ces parties.
  Une recherche de chaque sous-partie par des classifieurs différents est aussi possible. Alonso et al. [Alonso et al., 2007] font, dans cette optique, une comparaison des performances obtenues pour différents classifieurs sur chacune des sous-parties.

- **Cascades de classifieurs** Une série de classifieurs peut être plus efficace qu'un seul. Une cascade de classifieurs [Viola and Jones, 2001] [Viola et al., 2003] [Mikolajczyk et al., 2004] [Zhu et al., 2006] est une suite de classifieurs successifs (figure 2.4). Le passage d'un classifieur au suivant ne se fait que si le classifieur précédant donne une décision positive. Plusieurs caractéristiques peuvent ainsi être prises en compte successivement. Après avoir passé ces tests éliminatoires de plus en plus restrictifs (les premiers tests seront grossiers et peu discriminants puis les tests seront de plus en plus précis), la décision sera d'autant plus précise et les fausses détections seront mieux éliminées.

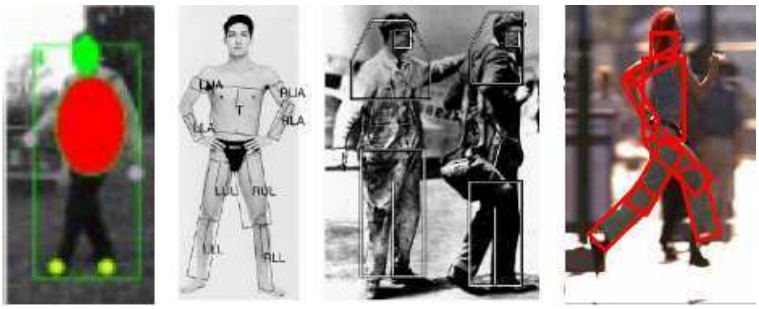

FIG. 2.3 – Exemples de détection par parties avec, de gauche à droite, les méthodes de Haritaoglu et al.[Haritaoglu et al., 1998], Ioffe et Forsyth [Ioffe and Forsyth, 2001], Mikolajczyk et al. [Mikolajczyk et al., 2004] et Ramanan et al. [Ramanan et al., 2005].

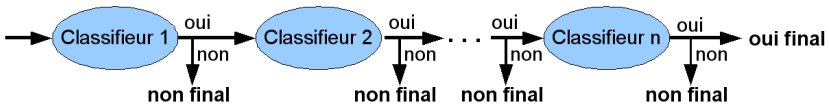

FIG. 2.4 – Une succession de classifieurs permet d'obtenir une décision plus restrictive et donc plus sûre.

- **Constitution d'une meilleure base de données par utilisation de l'alogithme** L'algorithme fonctionne à partir d'une base de données d'exemples positifs et négatifs. Cette dernière peut être améliorée facilement. Il suffit de réaliser la méthode à partir d'un petit nombre d'exemples négatifs. Des images ne contenant pas de personne sont placées en entrée de la méthode. Toutes les détections trouvées sont alors des fausses détections. En ajoutant ces fausses détections à la liste des exemples négatifs le classifieur s'adapte automatiquement aux exemples qu'il n'avait pas bien traités et devient donc de plus en plus discriminant (figure 2.5). Ceci permet une meilleure disparité des exemples [Oren et al., 1997].

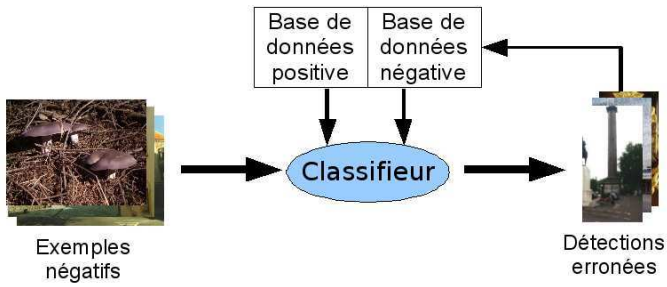

FIG. 2.5 – En rajoutant à la base de données négatives, les détections erronées réalisées sur des images sans personne, la classification est rendue plus efficace.

2.1.2 Autres démarches de détection

Si la structure descripteur-classifieur présentée dans le point précédent est celle utilisée le plus couramment, il existe d'autres démarches possibles qui sont très différentes.

Reconnaissance de mouvement

Une personne peut réaliser un nombre très important d'actions possibles. Il existe néanmoins certaines actions communes qui se retrouvent assez souvent (au moins assez pour apparaître une fois dans une vidéo) comme s'asseoir, marcher, courir ou se lever. La détection de personnes peut alors s'effectuer par une recherche de ces actions clés [Bobick et al., 2001].

Un des mouvements les plus caractéristiques des humains est le balancement des jambes. Une détection des jambes peut se faire (par recherche des lignes de contours verticales par exemple) ou du moins une détection d'un mouvement périodique [Polana and Nelson, 1994]. Ensuite une étude de la fréquence permet de définir s'il s'agit ou non d'une personne [Wohler et al., 1998][Cutler and Davis, 2000] [Ran et al., 2005].

Modèle probabiliste Bayésien

L'étude probabiliste est parfois utilisée [Fablet and Black, 2002] [Zhao and Nevatia, 2003] [Eng et al., 2004]. Une personne est decrite par un modèle (par exemple par un ensemble d'ellipses entourant certaines parties importantes du corps tels que le tête ou le torse). Les paramètres θ de ce modèle (position, taille et orientation de chaque ellipse) sont suivis au cours du temps. Connaissant les caractéristiques de l'image I, l'estimation des paramètres du modèle est calculée : $\widehat{\theta} = \arg max P(\theta/I)$.
La fonction de probabilité se détermine à partir de la ressemblance de l'image au modèle, des valeurs de paramètres préalablement déterminés et de la dynamique temporelle du système. De plus, certains phénomènes tels que les changements brusques de positions sont pénalisés.

Étude de la silhouette

Une fois une forme d'intérêt détectée, une étude de la silhouette de la forme en question peut être effectuée en considérant la silhouette comme une courbe. Plusieurs caractéristiques de cette courbe peuvent être prises en compte.
Tout d'abord une simple comparaison en distance de chanfrein de la silhouette avec un catalogue de silhouettes humaines est possible [Gavrila and Philomin, 1999]. Avec plusieurs orientations possibles cela représente une quantité de calcul importante. Voila pourquoi une hiérarchisation des modèles de silhouettes est utilisée.
Une détection de toutes les articulations (épaules, genoux, coudes, aisselles, mains, pieds,...) peut être réalisée par une étude des convexités et des concavités de la silhouette [Haritaoglu et al., 1998].
Xu et Fujimura [Xu and Fujimura, 2003] entourent la silhouette par une ellipse puis réduisent la taille de celle-ci selon l'orientation où l'union de l'ellipse et de la silhouette est la plus fréquente jusqu'à représenter le torse.
Enfin, il est possible de prendre en compte la distance de chaque point d'interêt à l'axe principal (déterminé à partir de la détection de la tête) [Haritaoglu et al., 1999], la densité de projection verticale et horizontale de la silhouette [Bertozzi et al., 2003] ou l'approximation par un polynôme de l'évolution de la courbe silhouette [Lee et al., 2004].

Carte des distances de Mahalanobis

L'image candidate est découpée en blocs. La moyenne x et la variance σ de chaque bloc sont calculées. La distance de Mahalanobis entre deux blocs i et j est définie par : $d(i,j) = \frac{(x_i - x_j)^2}{\sigma_i + \sigma_j}$. La carte des distances de Mahalanobis est la matrice comportant les distances de Mahalanobis pour toutes les combinaisons de blocs.

Utsumi et Tetsutani [Utsumi and Tetsutani, 2002] définissent ensuite un espace discriminant qui ne sélectionne que les parties de la carte des distances qui sont propres à la classe recherchée. Un seuillage du produit de la carte des distances avec la matrice de projection dans l'espace discriminant donne une décision.
He et al. [He et al., 2008] utilise lui les motifs locaux binaires. Pour chaque pixel est associée une combinaison binaire indiquant si chacun des pixels voisins lui est supérieur. Un pixel est uniforme si la combinaison possède au plus deux changements.

Autres méthodes

Enfin il existe certaines méthodes qui explorent des visions originales qui ne donnent pas forcement les meilleurs résultats. Ohba et Ikeuchi [Ohba and Ikeuchi, 1997] cherchent à reconnaître une sous-partie de l'image à partir d'un modèle et sélectionne les régions d'intérêts selon la détectabilité, l'unicité et la fiabilité. Ce système est peu robuste aux changements d'orientations et de poses.
Haga et al.[Haga et al., 2004] réalisent un passage dans l'espace formé par les axes unicité spatiale - unicité temporelle - continuité temporelle. Beleznai et al.[Beleznai et al., 2004] réalisent une décomposition en clusters par l'algorithme du Mean Shift. Li et al.[Li et al., 2004] utilisent un filtrage adaptatif d'échelle d'objets orientés (ORSAF).

Le suivi : une aide à la détection

Le suivi désigne l'apprentissage du mouvement de l'objet détecté. L'utilité première du suivi est la construction d'un avatar [Bregler and Malik, 1998] ou la reconnaissance d'un mouvement [Ramanan et al., 2005]. Néanmoins, comme la détection est nécessaire au suivi (au moins pour l'initialisation), le suivi peut aider la détection [Andriluka et al., 2008]. En effet, le suivi peut suivre le mouvement de chaque pixel [Wren et al., 1997]. La détection peut alors être nécessaire que pour les premières *frames* d'une vidéo.
Néanmoins, dans la plupart des cas, c'est le suivi d'un modèle qui est réalisé (d'où l'utilisation d'un avatar). Ce suivi aide cependant à donner une logique spatio-temporelle à la détection et donc à limiter les fausses détections. Ce suivi se réalise par une prédiction des états futurs calculée par filtrage de Kalman [Gavrila and Giebel, 2002] [Bertozzi et al., 2003] [Bertozzi et al., 2004].
Enfin, le suivi est utile lorsque deux personnes se croisent. Le suivi permet alors de séparer plus facilement les deux personnes grâce à la connaissance de leurs mouvements respectifs [Haritaoglu, 1998].

2.2 Détection et segmentation simultannée

Le but de la détection est de localiser automatiquement une personne dans une image ou une vidéo et de rendre en sortie l'image ou la vidéo avec une boite englobante encadrant chaque personne détectée (voir Figure 2.6).
Une perspective assez intéressante, et développée dans un certain nombre d'études récentes, est d'arriver à une détection précise, c'est-à-dire à segmenter la personne (voir Figure 2.6). La personne détectée ne sera alors plus repérée par une boîte englobante mais par ses propres contours. Ceci a de nombreuses applications pratiques comme on l'a vu dans l'introduction.
Il est possible de réaliser à la suite et indépendamment la détection et la segmentation.

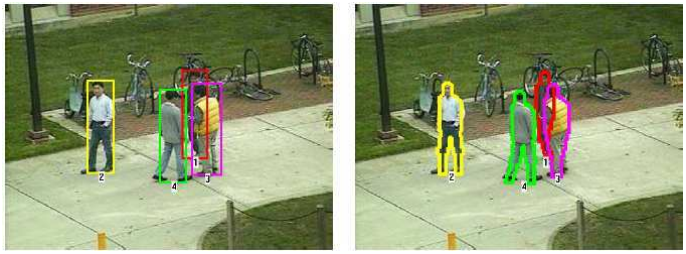

FIG. 2.6 – La détection seule encadre la personne détectée dans une boîte englobante (à gauche) alors que la segmentation donne les contours précis de celle-ci (à droite).

Néanmoins, la segmentation peut permettre une amélioration des résultats de la détection. Il est alors intéressant de coupler les deux procédés en une méthode.

2.2.1 Par analyse de régions d'interêts

Il est parfois facile d'isoler dans l'image un certain nombre de régions d'intérêts. Le plus facile étant lorsque le système d'acquisition peut être choisi. En effet, avec une caméra stéréo, le paramètre de profondeur peut être facilement calculé. Ainsi, la forme de la personne est directement donnée sans aucun processus de segmentation [Gavrila and Munder, 2007] [Xu and Fujimura, 2003].
Mori et al.[Mori et al., 2004] utilisent une segmentation par coupe normalisée (voir Figure 2.7) qui sépare l'image en un nombre prédéfini de régions cohérentes puis repère les sous-parties du corps humains grâce à leurs contraintes physiques. Lu et al.[Lu et al., 2008] présentent aussi un algorithme de mise en relation de régions appartenant à une même personne.

FIG. 2.7 – L'algorithme de segmentation en coupe normalisée sépare l'image en un nombre donné de régions (ici 40).

2.2.2 Par comparaison à un catalogue de gabarits

De nombreuses méthodes utilisent un catalogue de gabarits afin de détecter les personnes présentes sur une image [Munder and Gavrila, 2006] [Wang and Cohen, 2005]. Il s'agit en fait d'une liste regroupant le panel le plus varié d'exemples correspondant à la classe recherchée. Une comparaison est alors réalisée entre l'image testée et l'ensemble des éléments du catalogue. Si un des membres du catalogue est vraiment très proche, alors il y a détection de la classe.

Intégration de la segmentation

Dans la grande majorité des cas, le catalogue utilisé est un catalogue de gabarits de formes (voir Figure 2.8). En effet, la forme est la caractéristique la plus discriminante de la classe des personnes. La méthode doit donc soit isoler des contours par un filtrage comme celui de Canny [Zhao and Davis, 2005], soit faire des associations de sous régions [Mori et al., 2004] puis comparer aux membres du catalogue.

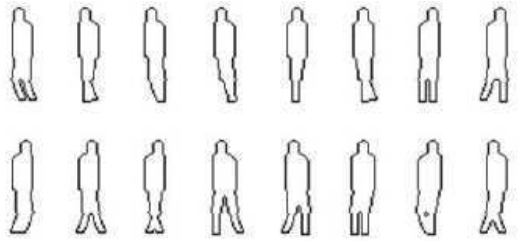

Fig. 2.8 – Exemple de catalogue de gabarits

Un modèle de silhouette proche de celui de la personne présente sur l'image testée est alors obtenu. Selon les caractéristiques (couleurs) de l'image, la silhouette est alors adaptée pour bien correspondre aux contours de la personne [Rodriguez and Shah, 2007] par un alignement des centroïdes. Ce qui réalise finalement la segmentation.
Il est aussi possible à partir d'exemples de gabarits proches, de rentrer dans chaque gabarit la notion de déformabilité qui donnera les parties du modèle qui sont les plus variables [Ferrari et al., 2007].
Le catalogue peut contenir autre chose que des silhouettes. Liu et Sarkar[Liu and Sarkar, 2004] y listent des projections sur des espaces propres et [Murai et al., 2007] des matrices représentatives des gradients dans l'espace spatio-temporel.

Avec une hiérarchie

Comparer les caractéristiques de l'image testée avec l'ensemble des gabarits d'un catalogue prend beaucoup de temps. Une solution pour réduire la quantité de calcul est d'utiliser une classification hiérarchique des gabarits [Gavrila and Philomin, 1999] [Gavrila and Giebel, 2002] [Shotton et al., 2008b]. L'idée est de retenir un petit nombre de gabarits les plus différents possibles et d'associer à chacun d'eux un autre petit nombre de gabarits proches mais différents entre eux et ainsi de suite sur plusieurs étages pour

former une pyramide (voir Figure 2.9). À chaque étage, une comparaison est faite avec chacun des gabarits pour déterminer le chemin dans la pyramide qui mène au gabarit le plus proche de la silhouette étudiée. Celui-ci est trouvé avec un nombre réduit de comparaisons.

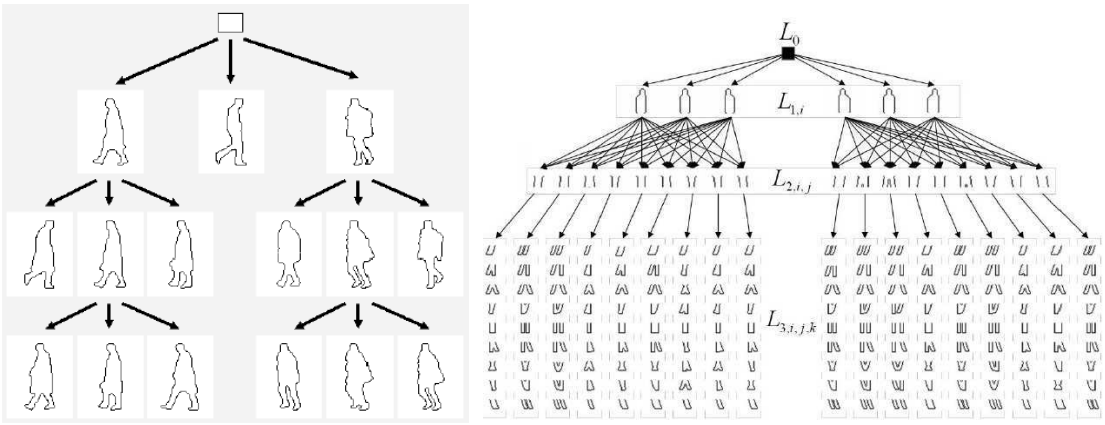

FIG. 2.9 – À gauche : classification hiérarchique de gabarits. À droite : classification par parties de Lin et al.[Lin et al., 2007a].

Lin et al. [Lin et al., 2007a] présentent une légère variante en découpant chaque gabarit en trois parties : la partie haute, le bassin et les jambes. Puis ils réalisent une pyramide où chaque étage représente tous les états possible d'une partie par rapport aux parties déjà sélectionnées aux étages précédents (voir Figure 2.9).

2.2.3 Par analyse de segments

La forme étant la caractéristique la plus discriminante, le contour est souvent étudié. Une idée est de le décomposer en segments (droits ou courbes) et de reconnaître ces segments. Si les contours d'un objet sont détectés, alors la segmentation est aussi effectuée.

Shotton et al. [Shotton et al., 2008a] démontrent que la reconnaissance d'un petit nombre de segments bien choisis, car caractéristiques, permet de détecter les éléments de certaines classes. Il suffit alors de reconnaître les segments un à un et de vérifier leur ordonnancement et leurs relations. Une cascade de classifieurs pour reconnaître chacun des segments les plus importants est introduite par Wu et Nevatia [Wu and Nevatia, 2007].

La notion de cycle peut être ajoutée au problème. Il s'agit d'essayer de former un cycle de segments de contours en reliant les segments par une valeur de continuité de distance, d'orientation et d'intensité. Sharma et Davis [Sharma and Davis, 2007] proposent une coupe de graphe d'un modèle caché de Markov reliant les cycles à leur probabilité d'appartenir à une personne afin d'obtenir le cycle de contour donnant la silhouette de la personne détectée. Cette méthode, comme celle que nous proposons en section 4, cherche les cycles de segments de contour pour reconstruire la silhouette recherchée. Mais, contrairement à notre méthode, la connaissance de la classe n'est pas présente dans l'étude du graphe mais intégrée dans une seconde étape. Ferrari et al.

[Ferrari et al., 2006] décomposent le modèle de la silhouette de référence en une suite ordonnée de segments. Cette suite est ensuite recherchée dans le même ordre dans l'image testée. Cette méthode ne fonctionne que sur des classes dont la forme des éléments la composant varie peu.

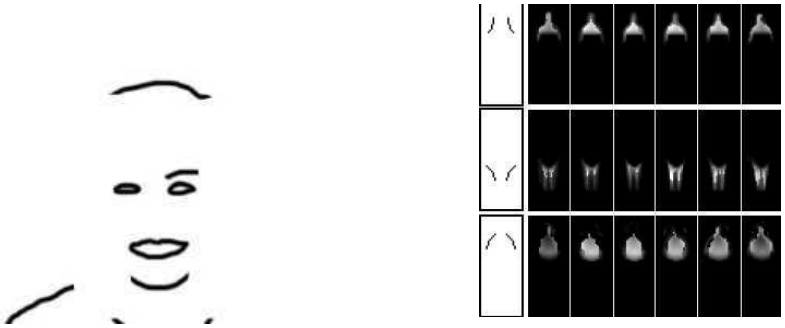

FIG. 2.10 – Á gauche : segments caractéristiques permettant la reconnaissance selon Shotton et al.[Shotton et al., 2008a]. À droite : segments reconnus par les classifieurs de Wu et Nevatia[Wu and Nevatia, 2007].

2.2.4 Par d'autres méthodes

Il existe d'autres méthodes qui prolongent celles présentées ci-dessus. La notion de mouvement peut par exemple être prise en compte. Par ACP, Liu et Sarkar [Liu and Sarkar, 2004] réalisent une décomposition en espaces propres de silhouettes en tronçons de mouvement (voir Figure 2.11). Une projection des caractéristiques de l'image testée sur les premiers espaces du modèle le plus proche recompose le contour. Yilmaz et Shah [Yilmaz and Shah, 2004] présentent une méthode de suivi de contours à partir des couleurs et de la vélocité.

Zhao et Davis [Zhao and Davis, 2005] présentent une méthode qui utilise itérativement une méthode de comparaison à un catalogue de gabarits avec une méthode de séparation sur les couleurs. Le gabarit choisi détermine les modèles de couleurs et la séparation des couleurs affine la forme à comparer aux gabarits. La segmentation est plus précise à chaque itération (voir Figure 2.11).

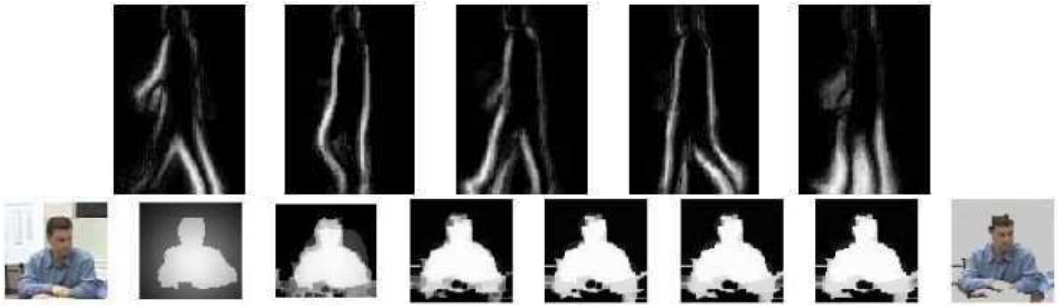

FIG. 2.11 – Au dessus : exemples d'espaces propres par Liu et Sarkar[Liu and Sarkar, 2004]. En dessous : progression des itérations de Zhao et Davis [Zhao and Davis, 2005].

Détection et segmentation peuvent être intégrées dans un cadre probabiliste par un modèle de forme implicite (ISM) [Leibe et al., 2004]. Un catalogue d'apparences locales contient la structure locale de la classe. L'ISM spécifie les positions relatives de ces informations. La comparaison des régions entourant les points d'interêts avec ce modèle par une transformée de Hough donne une formulation probabiliste qui permet la détection. Une vue probabiliste au niveau du pixel donne ensuite la segmentation.

2.2.5 Validation de la détection par la segmentation

La segmentation s'obtient après la détection. Néanmoins, la segmentation peut être utilisée pour valider la détection et ainsi réduire son taux de fausse détection. Pour certaines méthodes de détection, une segmentation grossière est calculée afin d'obtenir de meilleurs résultats.

Ramanan [Ramanan, 2007] crée un modèle spatial d'une classe. Une coupe de graphe guidée par les histogrammes de luminance produits par ce modèle donne une segmentation. Puis l'apprentissage d'un classifieur SVM est réalisé à partir des segmentations ainsi obtenues pour des exemples contenant un élément de la classe et d'autres n'en contenant pas. Ce classifieur permet finalement de valider la détection.
Ott et Everingham proposent une étape de segmentation intégrée dans le processus de détection par HOG [Ott and Everingham, 2009]. Pour chaque bloc (voir section 3.3.1), une segmentation non binaire par discriminant de Fisher est réalisée par une projection en couleur qui maximise la séparation entre premier et arrière-plan selon quatre positions de potentiels de référence. L'histogramme de gradients orientés de cette segmentation (appelé CHOG) est ajouté au HOG pour le rendre plus discriminant.

2.3 Conclusions

Comme nous l'avons vu, les techniques de **détection** de personnes sont nombreuses. Les méthodes par apprentissage grâce à des classifieurs performants comme les machines à vecteurs de support permettent de réaliser rapidement des comparaisons sur des quantités importantes d'informations à partir de base de données de grande taille. Ainsi, un panel se rapprochant de l'exhaustif des représentants possibles de la classe recherchée peut être pris en compte. De plus, parmi les nombreux descripteurs testés, certains ont prouvé leur qualité à caractériser le plus fidèlement possible les spécificités de la classe des personnes. Des résultats fiables et performants peuvent ainsi être obtenus.
Dalal et Triggs [Dalal and Triggs, 2005] arrivent à obtenir un taux de détection (rapport du nombre de personnes bien détectées sur le nombre de personnes à detecter) de 89% pour un taux de fausse détection (rapport du nombre de détections erronées sur le nombre totale de détections) par fenêtre de détection de 1 pour 10000. Par détection par partie, Alonso et al.[Alonso et al., 2007] attestent sur des séquences vidéo, d'un taux de détection de 99% pour un taux de fausse détection de 5%. Ces résultats montrent la fiabilité de ces méthodes qui détectent efficacement avec un faible risque de fausse détection. Certaines méthodes réalisent la segmentation simultanément à la détection mais ne se basent pas forcement sur les méthodes de détection les plus efficaces. Par exemple Sharma et Davis [Sharma and Davis, 2007] ont pour des images fixes, un taux

de fausse détection par fenêtre de 1 pour 10000, un taux de détection de 78.4%. Même si ces résultats sont plutôt bons, il serait intéressant d'apprécier la segmentation obtenue à partir d'une des méthodes de détection les plus performantes et les plus utilisées. Dans les chapitres 3 et 4 nous présentons une méthode de segmentation de personnes que nous avons développé en se basant sur les résultats et les outils de la méthode de détection de Dalal [Dalal, 2006] que nous avons choisi pour sa grande efficacité [Enzweiler and Gavrila, 2009].
Pour obtenir une segmentation précise, la coupe de graphe est une des méthodes les plus efficaces. Elle permet de plus une interaction performante avec l'utilisateur. Dans les chapitres 5 et 6 nous présentons une méthode de segmentation de personnes que nous avons développé à partir d'une coupe de graphe.

Chapitre 3

Pré-segmentation par recherche de segments de contour pertinents

> Je n'ai pas peur des ordinateurs. J'ai peur qu'ils viennent à nous manquer.
>
> Isaac Asimov

Sommaire

Une segmentation d'objet peut se réaliser par recherche d'un cycle de segments de contours pertinents. Une des principales difficultés réside dans l'évaluation de la pertinence des segments. Le problème est d'autant plus intéressant quand l'étude repose sur une classe bien particulière (dans notre cas celle des personnes). Dans ce chapitre, une pré-segmentation est présentée qui associe à chaque segment de contour d'une fenêtre de détection positive, une valeur qui correspond à sa vraisemblance en tant qu'élément de la

silhouette de la personne présente dans la fenêtre de détection.
Dans un premier temps, nous présentons dans la section 3.1 les objectifs de cette pré-segmentation ainsi que les raisons de baser notre étude sur les segments de contour. Puis nous présentons dans la section 3.2 les différents outils utilisés dans la méthode de détection sur laquelle nous nous basons. Ils auront un rôle prépondérant dans notre méthode. Les détails sur le fonctionnement de notre méthode de pré-segmentation sont décrits dans la section 3.3. Enfin, les résultats obtenus à partir de l'implémentation de notre méthode se trouvent dans la section 3.4. Les conclusions sur notre méthode de pré-segmentation sont tirées en section 3.5.

3.1 Objectifs

La détection vise à localiser un élément d'une classe. La plupart des méthodes donnent en sortie une fenêtre de détection. Les seules informations obtenues sont donc le lieu et la taille de l'élément trouvé. Pour réaliser une segmentation, ces informations ne suffisent pas. Il faut des informations plus précises sur les éléments présents dans la fenêtre. En outre, nous désirons utiliser les mêmes données que celles calculées lors de la détection. Il faut donc trouver lesquelles peuvent être utilisées de façon plus locale pour aider à la segmentation.

Il est important de différencier la segmentation de la recherche de sous-parties. Notre but n'est pas de reconnaître les différentes parties de l'anatomie de la personnes détectées (tête, bras, jambes, torse ...) mais de déterminer, sur la fenêtre de détection, quels sont les pixels appartenant à la personne et quels sont les pixels correspondant à l'arrière plan. Le but est de créer un masque binaire donnant l'appartenance ou non de chaque pixel à la personne recherchée.
Pour obtenir ce masque, soit on détermine pour chaque pixel s'il appartient à la silhouette, soit on recherche une entité caractéristique de la classe recherchée regroupant plusieurs pixels. Dans le premier cas, le nombre de calcul réalisé est très important. De plus, une classe ne peut être bien représentative à une si petite échelle. Le second choix est donc préférable.
Il est possible de regrouper les pixels en régions pour essayer de reconstituer la forme réalisée par la silhouette. Mais la variation de couleur et de texture des éléments composant une personne est très importante (par exemple une personne portant un habit de couleurs multiples engendre un grand nombre de régions indépendantes). Une autre possibilité est de chercher la délimitation entre les deux régions (premier et arrière plan). Cette délimitation est composée, pour le plus grand nombre de cas, d'une ligne unique se bouclant. Cependant, parfois, plusieurs boucles peuvent apparaître. Par exemple, une personne posant ses mains sur ces hanches crée une délimitation avec deux nouvelles boucles sous ses bras.
L'objectif est de reconstituer cette délimitation à partir d'un ensemble fini d'entités. Ces entités peuvent être des lignes, mais une courbe est difficile à définir et donne des caractéristiques trop complexes. Ce n'est pas le cas des segments qui sont juste définis par deux points et caractérisés par une orientation et une localisation. L'utilisation des segments est donc plus pertinente. De plus, l'usage des segments est cohérent avec l'usage des histogrammes de gradients orientés comme nous le verrons en section 3.2.2

Les différents contours de l'image sont donc modélisés par des segments. Nous recherchons le cycle de ces contours qui représente le mieux la silhouette souhaitée. La carte des contours est obtenue en appliquant l'algorithme de Canny sur chaque fenêtre de détection (figure 3.1(b)). L'algorithme de Canny permet également d'obtenir l'orientation de chaque contour. Ces contours sont ensuite délimités en lignes continues (figure 3.1(c)). Puis, ces lignes sont divisées en tronçons de droite les plus grands possibles dont la variation d'orientation ne dépasse pas un seuil. Chacun de ces tronçons est alors modélisé en un segment à partir de son point de départ et d'arrivé ainsi que de son orientation moyenne. L'ensemble de tous ces segments de contour (figure 3.1(d)) représente l'ensemble des candidats possibles pour faire partie du cycle représentant la silhouette recherchée.

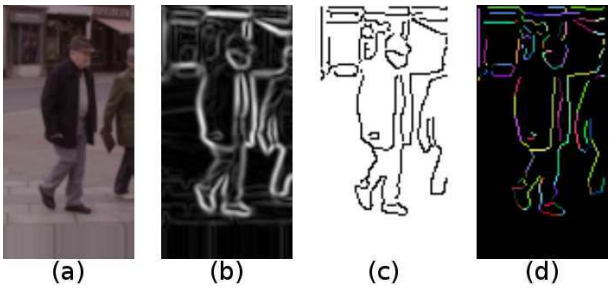

FIG. 3.1 – (a) Image initiale. (b) Magnitude de l'image par un filtrage de Canny. (c) Contours obtenus après suppression des non-maxima. (d) Vectorisation des contours en segments de contour.

3.2 Outils utilisés

Nous avons choisi comme contrainte que les éléments calculés lors de la détection doivent servir à la pré-segmentation. Les outils utilisés dans la méthode de détection choisie doivent donc être définis. Les histogrammes de gradients orientés sont un descripteur qui permet d'isoler les informations d'intérêts de l'image. Les machine à vecteurs de support sont un classifieur qui, par comparaison à une base de données, donne une décision sur l'appartenance d'un élément testé à la classe étudiée. Cette section présente les détails sur ces deux outils.

3.2.1 Machines à vecteurs de support

Les machines à vecteurs de support (ou SVM) définies par Vapnik en 1995 [Vapnik, 1995], sont un processus réalisant la classification de données. La classification se réalise sur des vecteurs à N dimensions. Dans notre cas, ces vecteurs doivent représenter des caractéristiques des images. L'élément testé ainsi que les éléments de la base de données sont dans ce format pour permettre les comparaisons.
La première partie du procédé consiste à créer un modèle à partir de la base de données. Il s'agit de l'apprentissage de la classe. La base de données se décompose en un ensemble d'éléments positifs (contenant un élément de la classe) et un ensemble d'éléments négatifs (ne contenant pas d'élément de la classe). Un hyperplan séparant les éléments de chacun des deux ensembles est calculé de façon à maximiser la marge (figure 3.2), c'est-à-dire la distance entre les échantillons et l'hyperplan. Pour cela, l'espace d'étude est retranscrit

sur un espace de plus grande dimension où l'existence d'une séparatrice linéaire est possible. Finalement, l'ensemble des exemples positifs se trouve d'un coté de l'hyperplan tandis que l'ensemble des exemples négatifs se trouve de l'autre coté.

Dans la seconde partie du procédé, ce modèle permet la décision. Si le vecteur testé se trouve du coté de l'hyperplan relatif au exemples positifs, alors il s'agit d'un élément de la classe. Dans le cas contraire, il ne s'agit pas d'un élément de la classe.
On peut remarquer que la distance du vecteur de caractéristiques à l'hyperplan donne une évaluation de la fiabilité de la décision. En effet, si cette distance est très réduite, la décision sera moins tranchée car l'exemple est très proche des deux classes.

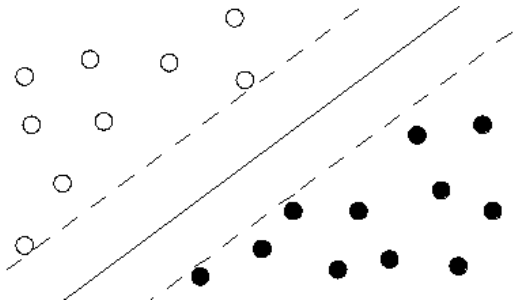

FIG. 3.2 – Illustration du principe de fonctionnement des SVMs pour un cas linéaire simple. Les éléments d'une classe (ronds blancs) sont séparés des éléments d'une autre classe (rond noirs) par l'hyperplan séparateur (trait plein) qui maximise la marge (traits en pointillés).

3.2.2 Histogrammes de gradients orientés

Un histogramme de gradients orientés (ou HOG), est un outil introduit par Dalal en 2005 [Dalal and Triggs, 2005]. Il définit dans une région les proportions de pixels dont l'orientation du gradient appartient à un certain intervalle. Ces proportions caractérisent la forme présente dans cette région. L'information est plus caractéristique si les HOGs sont pris sur plusieurs unités de lieu. Elle peut alors être utilisée en tant que descripteur de forme.

Carte des gradients

Les contours d'une image donnent une bonne représentation des formes qui y sont présentes. Le calcul du gradient, notamment par la méthode de Canny [Canny, 1986], permet une détection performante des contours.

Un filtrage gaussien à deux dimensions est tout d'abord appliqué à la luminosité de l'image (figure 3.3(b)) afin de réduire le bruit et de limiter le nombre de contours. Ensuite, pour chaque pixel p de l'image, on applique un masque $[-1\ 0\ 1]$ puis un masque $[\text{-}1\ 0\ 1]^T$. On obtient les valeurs G_p^x et G_p^y. La somme des valeurs absolues des images obtenues en sortie de ces deux filtres ($|G_p^x| + |G_p^y|$) donne une image qui correspond à la magnitude du gradient (figure 3.3(c)). En effet, cette valeur représente les écarts de nuances selon les orientations verticale et horizontale.

Ensuite, l'orientation du gradient est obtenue par l'arc tangente du rapport entre la sortie du filtre horizontale sur la sortie du filtre verticale ($arctan(\frac{G_p^x}{G_p^y})$) comme sur la figure 3.3(d)). Pour n'obtenir que les contours, les non-maxima sont supprimés. Les contours sont "fermés" par un seuillage par hystérésis.

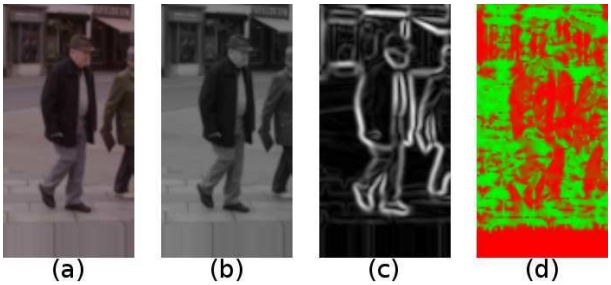

FIG. 3.3 – (a) Image initiale. (b) Luminance de l'image initiale. (c) Magnitude du gradient par filtrage de Canny. (d) Orientation du gradient par filtrage de Canny. Les nuances rouges représentent les orientations verticales et les nuances vertes représentent les orientations horizontales.

Histogramme

Un histogramme est un outil permettant de représenter la répartition d'une variable sur une population. L'espace généré par cette variable est divisé en intervalles. L'histogramme est la quantité, pour chaque intervalle, d'élément de la population dont la variable appartient à l'intervalle. Dans notre cas, la population est l'ensemble des pixels de l'image et la variable est l'orientation du gradient.
Prendre l'orientation du gradient entre 0 et 2π radians permet de prendre en compte le sens du contour et donc la transition en luminosité. Néanmoins, selon les cas, le premier plan est plus clair que l'arrière plan ou le contraire. Le sens de l'orientation du gradient n'est donc pas caractéristique de la classe des personnes. De plus, il est préférable d'avoir un descripteur de petite dimension. Nous considérons donc comme espace généré par l'orientation du gradient l'intervalle $[0, \pi]$.
Cet espace est divisé en N_{bin} intervalles. Ces intervalles sont de mêmes tailles ($\frac{\pi}{N_{bin}}$). L'ensemble Δ des intervalles d'orientation du gradient est défini par :

$$\Delta = \left\{ \mathcal{I}_k = \left[\frac{2k-1}{2N_{bin}}\pi, \frac{2k+1}{2N_{bin}}\pi \right], k \in [1, N_{bin}] \right\} \tag{3.1}$$

Il existe deux possibilités pour déterminer le nombre de pixels concernés par chaque intervalle d'orientation :

- Soit les contours sont pris en compte. Dans ce cas, le nombre de pixels concernés par un intervalle est le nombre de pixels appartenant à un contour et dont l'orientation du gradient appartient à l'intervalle. Cette solution est cohérente avec l'utilisation des segments de contour et la recherche de silhouette.
- Soit le gradient est pris en compte. Dans ce cas, le nombre de pixels concernés par un intervalle est le nombre de pixels dont l'orientation du gradient appartient à l'intervalle pondéré par la magnitude du gradient. Cette solution permet une étude plus souple des variations de gradient.

Notre objectif étant, dans un second temps, de caractériser les segments de contour, nous prenons en compte la première possibilité.

Le décompte des pixels dont l'orientation du gradient appartient à un intervalle est binaire. Il est incrémenté de 1 si l'orientation appartient à l'intervalle et de 0 dans le cas contraire. Or, un pixel dont l'orientation du gradient est proche de la valeur moyenne de l'intervalle est plus caractéristique de celui-ci. Un pixel dont l'orientation du gradient est éloigné de la valeur moyenne de l'intervalle est également caractéristique de l'intervalle voisin. Une interpolation est alors réalisée pour adoucir les contraintes du décompte.
Soit D_k, le décompte des pixels pour l'intervalle $\mathcal{I}_k$. Sans interpolation, le décompte pour chaque pixel de l'image d'orientation du gradient θ est réalisé par :

$$\text{pour tout k} \in [1, N_{bin}], D_k := \begin{cases} D_k + 1 & \text{si } \theta \in I_k \\ D_k & \text{sinon} \end{cases} \tag{3.2}$$

Soit $\theta \in \left[\frac{l\pi}{N_{bin}}, \frac{(l+1)\pi}{N_{bin}}\right]$ l'orientation du gradient d'un pixel de l'image. Avec interpolation, la règle de décompte devient :

$$\text{pour tout k} \in [1, N_{bin}], D_k := \begin{cases} D_k + \dfrac{\theta - \frac{l\pi}{N_{bin}}}{\frac{\pi}{N_{bin}}} & \text{si } k = l \\ D_k + \dfrac{\frac{l\pi}{N_{bin}} - \theta}{\frac{\pi}{N_{bin}}} & \text{si } k = l+1 \\ D_k & \text{sinon} \end{cases} \tag{3.3}$$

Ces décomptes correspondent aux fréquences d'occurrences des pixels de l'image dont l'orientation du gradient appartient à un intervalle de Δ. Soit ν_{I_k}, la fréquence d'apparition relative à l'intervalle I_k. L'histogramme de gradients orientés de la région est alors l'ensemble de ces fréquences pour tous les intervalles de Δ.

$$HOG = \left\{\nu_I^{region}, I \in \Delta\right\} \tag{3.4}$$

Les effets d'illumination rendent moins efficace les vecteurs de caractéristiques. Il faut donc uniformiser ces derniers sur l'ensemble des exemples. Ceux-ci sont donc normalisés (voir section 3.3.1).

3.2.3 Vue générale de la méthode de Dalal et Triggs

L'aboutissement de la méthode présentée dans ce chapitre est d'adapter une méthode de détection efficace afin de réaliser la segmentation. La méthode choisie est celle détaillée par Dalal dans [Dalal and Triggs, 2005]. Dans cette méthode, les histogrammes de gradients orientés sont utilisés comme descripteurs et les machines à vecteurs de support sont utilisées comme classifieur. Des détails sur son fonctionnement permettent d'appréhender son adaptation.

Afin d'obtenir une information sur la forme, une utilisation spatiale des HOGs est réalisée. La fenêtre de détection étudiée est découpée en un certain nombre de sous-parties. En réalisant la concaténation des HOGs de chacune des sous-parties, on obtient un vecteur de caractéristiques qui est bien discriminant de la forme présente.
Dalal teste plusieurs découpages (découpage régulier ou se concentrant sur certaines régions qui ont le plus d'intérêts) de l'image afin de choisir le plus performant. Un découpage selon un quadrillage avec recouvrement (expliqué en section 3.3.1) est utilisé dans cette méthode car très efficace.
Le découpage divise la fenêtre en sous-parties nommées cellules. Un HOG est calculé pour chaque cellule. Les cellules sont regroupées en blocs. Afin de réduire les effets d'éclairages locaux, une normalisation est effectuée sur les blocs. Un recouvrement des blocs permet de rendre l'information plus discriminante. Le vecteur de caractéristiques finalement obtenu est la suite pour tous les blocs des concaténations de HOGs des cellules composant chacun de ces blocs.

Sur une image candidate, la méthode teste toutes les fenêtres de détection possibles à toutes les échelles et localisations. Pour chacune de ces fenêtres de détection, une décision sur son appartenance ou non à la classe recherchée est obtenue par l'association HOGs et SVM.
Pour une personne présente sur l'image initiale, il existe une fenêtre l'encadrant de la façon la plus précise. Néanmoins, les fenêtres proches spatialement ou en échelle risquent de donner aussi une classification positive. On obtient alors une constellation de fenêtres de détection positives autour d'une même personne détectée. La méthode de Dalal utilise l'algorithme du Mean-Shift [Comaniciu, 2003] pour déterminer la position et la taille idéale de la fenêtre de détection relative à la personne détectée (figure 3.4).

FIG. 3.4 – Pour une personne dans l'image, une multitude de fenêtres sont détectées (à gauche). La dernière étape consiste alors à déterminer laquelle est celle encadrant le mieux la personne (à droite).

3.3 Étude locale de la fenêtre de détection

La méthode de Dalal réalise une reconnaissance des fenêtres de détection positives. Elle part d'une fenêtre et renvoie une information binaire (cette fenêtre contient ou non une personne). C'est insuffisant pour réaliser la segmentation. Des informations plus précises sont obtenues par une étude plus locale.

Avec l'étude globale, un vecteur de caractéristiques $V^{globale}$ est calculé pour chaque fenêtre de détection testée. Ce vecteur est défini par :

$$V^{globale} = \{\{HOG_{cellule}, cellule \in bloc\}, bloc \in fenetre\} \tag{3.5}$$

L'information est bien descriptive mais concerne la fenêtre de détection dans sa globalité. Pour une fenêtre candidate, le vecteur de caractéristiques est placé en entrée d'une SVM qui renvoie une décision. Le principe est résumé sur la figure 3.5. Une seule information binaire par fenêtre est obtenue.

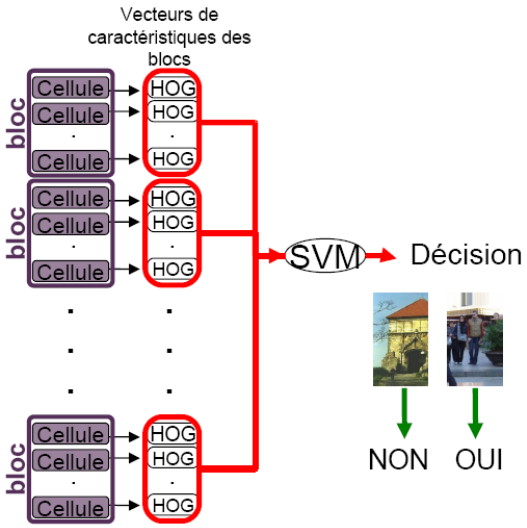

FIG. 3.5 – Principe de détection globale. Une seule information (la décision sur l'appartenance à la classe) est au final obtenue.

Dans l'étude locale que nous proposons, une décision est obtenue pour une région plus réduite et localisée dans la fenêtre : le bloc. Le vecteur de caractéristiques relatif au *bloc* est défini par :

$$V_{bloc}^{locale} = \{HOG_{cellule}, cellule \in bloc\} \tag{3.6}$$

Chacun de ces vecteurs de caractéristiques est placé en entrée d'une SVM qui donne une valeur S_{bloc}^{SVM}. Cette valeur représente la vraisemblance des éléments présents dans ce bloc en tant qu'élément d'une silhouette humaine au lieu relatif au bloc. Une valeur par bloc est donc obtenue comme le montre la figure 3.6. Enfin, ces valeurs sont associées aux contours de l'image pour obtenir une valeur de vraissemblance des segments en tant qu'élément de la silhouette.
La suite de cette partie rentre plus en détails sur le fonctionnement de la méthode. La section 3.3.1 explique la façon dont les caractéristiques de l'image sont utilisées par le descripteur. La section 3.3.2 décrit comment les SVM renvoient les valeurs de sortie S_{block}^{SVM}. Enfin la section 3.3.3 présente le passage des détections locales à la valorisation des segments de contour.

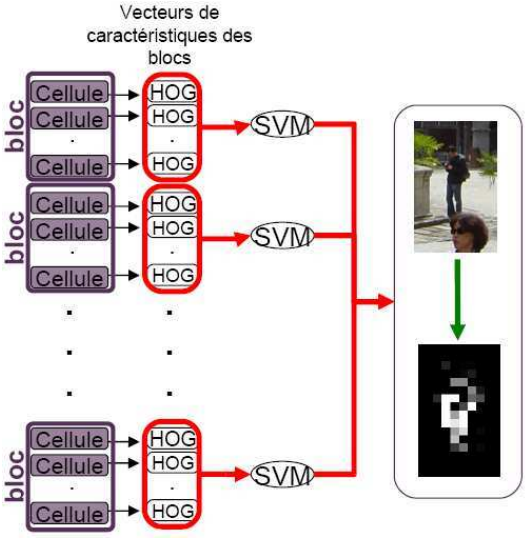

FIG. 3.6 – Vue d'ensemble du principe de la détection précédente adaptée à une étude localisée. Cette fois-ci une information est obtenue pour chaque bloc.

3.3.1 Le descripteur

Découpage de l'image

Le découpage de l'image en cellules se fait selon un quadrillage carré (figure 3.7). En effet Dalal [Dalal, 2006] démontre que ce découpage est le plus performant pour la détection. De plus, il est le plus simple à mettre en oeuvre et permet une répartition spatiale régulière pertinente avec l'équation 3.13.
L'image est divisée de façon dense en $n_{colonne} \times n_{ligne}$ cellules carrées. Le nombre de lignes et de colonnes et la taille des cellules sont alors les variables de cette première partie de l'algorithme.

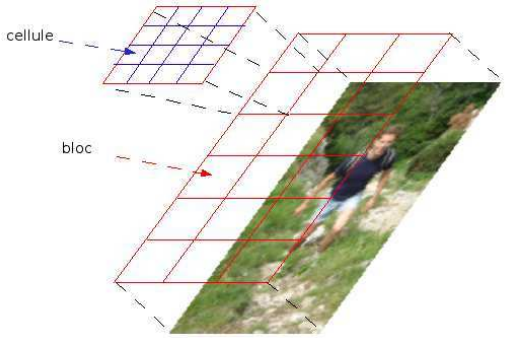

FIG. 3.7 – Découpage de l'image en cellules (en bleu) et en blocs (en rouge). Le découpage par un quadrillage est l'un des plus efficaces.

Recouvrement

Un bon compromis doit être trouvé entre la taille des cellules et la taille des blocs. En effet, si le nombre de cellules dans un bloc est élevé, le descripteur est plus discriminant.

Mais la taille des cellules est alors plus petite. La taille des cellules doit cependant rester assez importante car, si les cellules sont trop petites, les informations qu'elles contiennent sont trop peu nombreuses et donc peu pertinentes.
De plus, un grand nombre de blocs donne un résultat final plus précis. En effet, avec un nombre de blocs important, l'information en sortie des SVMs concerne une zone plus petite. Les segments reconnus sont donc mieux localisés et il y a moins de segments favorisés à tort car se trouvant à proximité de segments vraisemblables. Or, en gardant une taille de cellule suffisante, augmenter le nombre de blocs revient à diminuer le nombre de cellules par bloc.

Néanmoins, le recouvrement des blocs permet d'augmenter le nombre de blocs sans pour autant diminuer le nombre de cellules par bloc.

Le principe du recouvrement des blocs est de faire se chevaucher les blocs les uns sur les autres (voir figure 3.8). Les cellules peuvent appartenir à plusieurs blocs et non forcement à un seul comme dans le cas sans recouvrement. Ainsi, avec un même nombre de cellules, le nombre de blocs est plus important. Le calcul des HOGs a montré que, sans recouvrement, l'image est découpée en $\frac{n_{colonne}}{n_{taille.bloc}} \times \frac{n_{ligne}}{n_{taille.bloc}}$ blocs. Avec un taux de recouvrement de r, on a alors : $\left(\frac{n_{colonne}}{(1-r)n_{taille.bloc}} - \frac{r}{1-r}\right) \times \left(\frac{n_{ligne}}{(1-r)n_{taille.bloc}} - \frac{r}{1-r}\right)$ blocs.

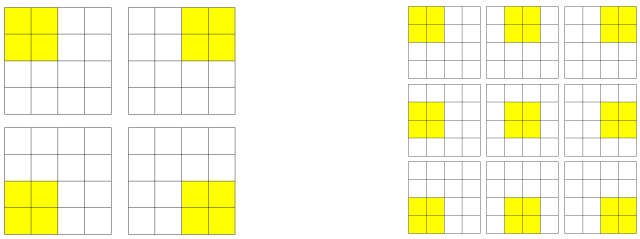

FIG. 3.8 – Illustration du principe de recouvrement des blocs sur un exemple simple. Soit une image de 4×4 cellules. Sans recouvrement des blocs (à gauche), il y a 4 blocs de 2×2 cellules possibles. Avec recouvrement des blocs de $\frac{1}{2}$ (à droite), il y a 9 blocs de 2×2 cellules possibles.

Le recouvrement des blocs permet d'obtenir des informations en sorties des SVMs plus précises. En outre, les résultats de reconnaissance de segment (section 3.4) sont bien meilleurs avec recouvrement. Le recouvrement des blocs est donc appliqué dans la suite de l'étude.

Normalisation

Selon l'éclairage ou le contraste entre le premier et l'arrière plan, les valeurs des HOGs peuvent varier de façon importante, bien que la personne ait la même posture. Les éléments de la base de données positive sont alors moins proches et le descripteur est alors moins performant.

Afin d'uniformiser les vecteurs de caractéristiques, une normalisation du vecteur de caractéristiques V_{bloc} est réalisée. Soit N la taille du vecteur. Les différentes normalisations testées sont :

- Norme L1 : moyennage par la norme 1 du vecteur :

$$V_{bloc}^{L1}(i) = \frac{V_{bloc}(i)}{\sum_{k=1}^{N} V_{bloc}(k)} \tag{3.7}$$

- Norme L1sqrt : la racine carré de la norme L1 est prise afin de traiter le vecteur de caractéristiques comme une densité de probabilité.

$$V_{bloc}^{L1_{sqrt}}(i) = \sqrt{\frac{V_{bloc}(i)}{\sum_{k=1}^{N} V_{bloc}(k)}} \tag{3.8}$$

- Norme L2 : moyennage par la norme 2 du vecteur :

$$V_{bloc}^{L2}(i) = \frac{V_{bloc}(i)}{\sqrt{\sum_{k=1}^{N} V_{bloc}(k)^2}} \tag{3.9}$$

- Norme 2 à hystérésis : des changements non linéaires d'illumination peuvent causer des saturations lors de l'acquisition et provoquer des changements brutaux de magnitude. L'influence des gradients de hautes magnitudes est ici réduite en seuillant à 0,2 après une normalisation L2. Une normalisation unitaire est finalement réalisée pour obtenir la norme L2 à hystérésis.

Les effets de ces différentes normes sont explicités par les résultats de la section 3.4.

Algorithme

Les précisions données plus haut permettent de déterminer l'algorithme relatif au calcul du descripteur dont le résumé est présenté dans l'algorithme 1.

Algorithm 1 Algorithme de réalisation du descripteur caractérisant chaque bloc

Entrée : fenêtre de détection candidate
Sortie : vecteurs de caractéristiques de chaque bloc

- calcul de la luminance de la fenêtre de détection
- filtrage de Canny $\longrightarrow$ carte des contours
- vectorisation des contours $\longrightarrow$ liste des segments de contours
- calcul du HOG de chaque cellule de la fenêtre
- regroupement des cellules en blocs
  - création du vecteur de caractéristiques
  - normalisation du vecteur de caractéristiques

3.3.2 Le classifieur

Apprentissage

L'apprentissage est réalisé à partir d'une base de données d'exemples positifs et d'exemples négatifs annotés. La classification se réalise par comparaison du candidat aux exemples de ces deux classes. Comme nous réalisons maintenant une classification SVM sur chaque bloc, il faut réaliser un apprentissage pour chacun des N_{bloc} blocs. On réutilise, dans un premier temps, les images de la base de données de l'INRIA que l'on découpe pour former les N_{bloc} bases de données (figure 3.9).

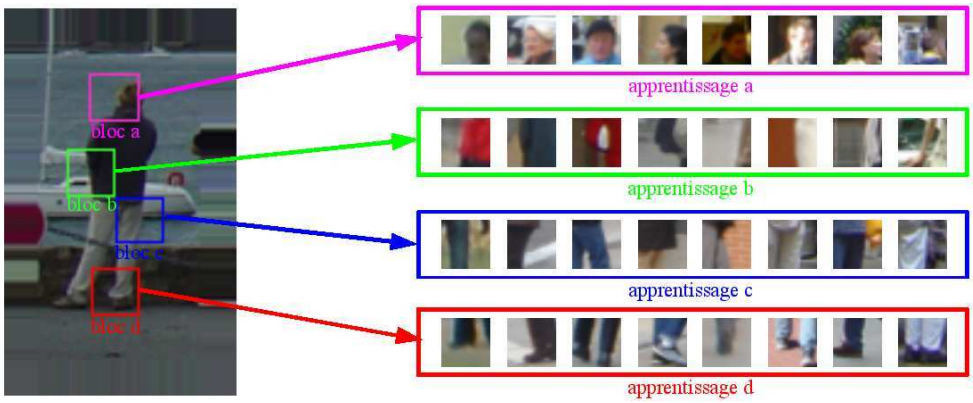

FIG. 3.9 – Réalisation des N_{bloc} bases d'apprentissage : pour chaque fenêtre de la base de données de Dalal, les blocs sont ajoutés à la base d'apprentissage correspondante.

Cependant, sur un élément de la base de données positives de l'INRIA, certains blocs ne contiennent pas d'élément de la silhouette de la personne. C'est surtout le cas pour les blocs se trouvant sur les bords de la fenêtre. Si ces élément sont utilisés pour l'apprentissage de l'étude locale, le modèle obtenu donne une classification bien moins performante. C'est pourquoi, pour réaliser nos N_{bloc} bases de données positives, nous avons décidé de partir non plus d'une base de données d'images, mais d'une base de données de silhouettes. Ainsi, chaque bloc contient un élément de la silhouette ou est vide. L'apprentissage est alors réalisé convenablement. La création de cette base de données de silhouettes est expliquée en détail en annexe A.

Le classifieur SVM

Les SVMs utilisées dans notre étude pour classifier les différents blocs de l'image sont ceux développés par Thorsten Joachims en 1999 [Joachims, 1999]. L'algorithme détermine l'hyperplan séparateur divisant l'espace des vecteurs de caractéristiques possibles en deux sous-espaces. L'un contient tous les vecteurs des exemples positifs. L'autre contient tous les vecteurs des exemples négatifs. L'hyperplan est calculé de façon à minimiser sa marge avec les exemples positifs et négatifs. Chaque nouveau vecteur testé correspond à une fenêtre de détection positive s'il se trouve du même côté de l'hyperplan que les vecteurs des exemples positifs.
L'algorithme de Joachims renvoie une valeur relative à la position du vecteur vis à vis de l'hyperplan. Si cette valeur est nulle, le vecteur appartient à l'hyperplan. Si elle est

positive, le vecteur correspond à un élément de la classe. Sinon, il ne correspond pas à un élément de la classe. Plus la valeur absolue de cette valeur est grande, plus le vecteur est éloigné de l'hyperplan. Si le vecteur se trouve dans la marge de l'hyperplan, alors la valeur retournée est comprise entre -1 et 1.

Avec le recouvrement des blocs que nous effectuons, une cellule peut appartenir à plusieurs blocs. Le calcul de la vraisemblance se base sur la répartition du segment sur les blocs et la valeur de sortie des SVMs. Or, avec recouvrement, il se peut qu'un pixel soit spatialement sur deux blocs simultanément. Définissons alors la valeur de sortie de la SVM pour une cellule qui est la moyenne des valeurs de sortie des SVMs relatives aux blocs comprenant cette cellule.

$$S^{SVM}_{cellule} = \underset{bloc \ni cellule}{\text{moyenne}}(S^{SVM}_{bloc}) \tag{3.10}$$

3.3.3 Vraissemblance des segments de contours

L'étude locale nous permet d'obtenir une valeur pour chaque bloc. Mais, comme nous l'avons expliqué dans nos objectifs, notre volonté est d'obtenir une valeur pour chaque segment de contour.
Les contours sont étudiés car ils définissent la silhouette. La vitesse de transition n'intervient pas. Donc, la magnitude du gradient n'est prise en compte que pour la détection des contours.

Soit un segment de contour seg et t_c le nombre de pixels de ce segment dans la cellule c. Alors une première évaluation de la vraisemblance de ce segment peut être la somme des sorties des SVMs pondérées par le nombre du pixel dans la cellule correspondante. Elle est donc définie par :

$$P_{seg} = \frac{\sum_{k=1}^{N_{cellule}} t_{c_k} S^{SVM}_{c_k}}{\sum_{k=1}^{N_{celule}} t_{c_k}} \tag{3.11}$$

Un segment est défini par sa taille, sa localisation et son orientation. Cette dernière information est très influente dans la classification mais n'apparaît pas dans l'équation 3.11. Par ailleurs, il est assez logique que les segments qui ont le plus contribué à l'élaboration du HOG, qui est le point de départ du processus de reconnaissance avec les SVMs, soient favorisés dans le choix des segments vraisemblables. Or, le HOG définit l'influence de chaque orientation sur l'ensemble des contours. Soit $\nu_I^{cellule}$ la fréquence d'occurrence de pixels de contours dont l'orientation appartient à l'intervalle I dans c. Le HOG correspondant à cette cellule est alors défini par :

$$HOG_{cellule} = \left\{\nu_I^{cellule}, I \in \Delta\right\} \tag{3.12}$$

Cette valeur s'extrait facilement des HOGs calculés dans l'étude globale réalisée pour la détection et ne demande donc pas de calcul supplémentaire. Soit seg un segment de

contour d'orientation appartenant à l'intervalle I, cette valeur de vraisemblance est alors définie par :

$$P_{seg} = \frac{\sum_{k=1}^{N_{cellule}} t_{c_k} HOG_{ck}(I) S_{c_k}^{SVM}}{\sum_{k=1}^{N_{cellule}} t_{c_k}} \tag{3.13}$$

Cette valeur caractérise correctement la vraisemblance de chaque segment en tant qu'élément d'une silhouette humaine. Plus cette valeur est importante, plus le segment a des chances d'appartenir au contour de la silhouette recherchée.

3.4 Résultats

Dans cette section, une évaluation de notre méthode de pré-segmentation est présentée. Une série de tests a été réalisée pour obtenir des résultats dont la qualité est évaluée. L'évaluation est qualitative et quantitative. Dans le premier cas, les résultats seront appréciés visuellement. Dans le second cas, des valeurs numériques estiment leur qualité.

Ces tests ont été réalisés à partir d'un algorithme que nous avons codé en langage C. Les SVMs sont déterminés à partir du code de *SVMlight* réalisé par Thorsten Joachims pour l'article [Joachims, 1999] et disponible sur le site : *http ://svmlight.joachims.org/*. L'ensemble des N_{bloc} modèles est obtenu sous forme de fichier .doc annexe aux deux logiciels d'apprentissage et de classification.

Les images utilisées pour les tests et pour l'apprentissage sont issues de la base de données *INRIA Static Person Data Set* réalisée par Navneet Dalal pour son étude [Dalal and Triggs, 2005] et disponible sur *http ://pascal.inrialpes.fr/data/human/*.
Pour l'apprentissage, les images négatives de la base de données sont conservées et les images positives sont transformées en silhouettes comme présenté en annexe A pour les raisons soulevées en section 3.3.2. Dans nos tests, l'apprentissage est réalisé à partir de 200 images positives et 200 images négatives. Nos tests avec de plus larges bases de données n'ont pas amélioré la pré-segmentation. Enfin les fenêtres candidates testées sont des images positives de la base de données de l'INRIA non présentes dans la base d'apprentissage.

3.4.1 Modes d'évaluations des résultats

L'évaluation de la qualité de nos résultats a nécessité plusieurs outils. Des appréciations qualitatives (avec des visualisations de résultats mettant bien en avant les apports et les failles de la méthode) et des appréciations quantitatives (pour chiffrer la qualité) sont essentielles à une bonne évaluation.

Image de sorties des SVMs

Cette image permet de visualiser les valeurs de sorties des SVMs. Sans recouvrement des blocs, chaque bloc est représenté rempli d'une nuance de gris proportionnelle à la valeur S_{bloc}^{SVM}. Avec recouvrement, chaque cellule est représentée remplie d'une nuance de gris proportionnelle à la valeur $S_{cellule}^{SVM}$. La nuance noire correspond au minimum des S_{bloc}^{SVM} (ou des $S_{cellule}^{SVM}$) et donc aux blocs les moins vraisemblables. La nuance blanche correspond au maximum des S_{bloc}^{SVM} (ou des $S_{cellule}^{SVM}$) et donc aux blocs les plus vraisemblables. Ainsi plus le bloc (ou la cellule) est de nuance élevée, plus les éléments qui le composent sont probables en tant qu'éléments d'une silhouette humaine et donc plus le contour de la silhouette a de chance de passer par ce bloc (ou cette cellule). Cette image est obtenue à la fin du processus de classification par SVM. Elle permet alors d'apprécier cette première étape et donc de vérifier que la reconnaissance est pertinente. Des exemples sont donnés figure 3.10 avec les S_{bloc}^{SVM} sans recouvrement (quatrième colonne) puis avec les $S_{cellule}^{SVM}$ et recouvrement des blocs (seconde colonne).

Segments vraissemblables

La finalité de la pré-segmentation est de donner une valeur de vraisemblance en tant qu'élément d'une silhouette humaine aux segments de contour d'une image. Cette valeur est obtenue selon l'équation 3.13. Il est alors intéressant de pouvoir visualiser quels sont les segments de contours les plus vraisemblables. La visualisation des segments vraisemblables montre l'ensemble des segments de contours de l'image avec une nuance proportionnelle à cette valeur. Les nuances sont recadrées pour permettre une meilleure visualisation. Ainsi le segment le plus vraisemblable est codé à 255 (blanc) tandis que le moins vraisemblable est codé à 0 (noir). Cette visualisation permet de voir l'aboutissement de la pré-segmentation. De plus, elle permet d'évaluer si la pré-segmentation est assez précise pour permettre la segmentation. C'est donc la représentation la plus importante.

Critère de quantification

Les visualisations présentées ci-dessus sont des résultats qualitatifs de la méthode. Elles permettent à un utilisateur de voir immédiatement si le résultat obtenu est proche de celui attendu et, en cas de problème, permet de repérer rapidement le point bloquant ou les phénomènes particuliers qui représentent les limites de la méthode.
Cependant, les tests effectués sur notre algorithme doivent déterminer les valeurs des différentes variables qui permettent d'optimiser les performances de l'algorithme. Or, une petite variation sur une variable ne modifira pas beaucoup le résultat. Des comparaisons qualitatives ne sont donc pas efficaces. En outre, il est intéressant d'évaluer l'amélioration apportée par un choix pour, dans le cas d'un compromis entre plusieurs améliorations possibles, savoir si ce choix est judicieux et si ses avantages prévalent sur ses inconvénients. Voilà pourquoi une valeur quantitative des résultats, c'est-à-dire une valeur numérique correspondant à la performance de la méthode, doit être trouvée.

Pour les méthodes de détection de l'état de l'art de la section 2, les classifieurs sont binaires. Les mesures de performance de ces méthodes sont alors déterminées par les diagrammes ROC (Receiver Operating Characteristic) [Fawcett, 2006] qui représentent graphiquement une courbe qui donne le taux de classifications correctes parmi un groupe

de tests (ou bonnes détections) en fonction du nombre de classifications incorrectes (ou fausses détections) pour ce même groupe. Si un bon taux de détection (c'est-à-dire que beaucoup d'éléments à détecter le sont) entraîne un nombre élevé de fausse détection, alors la méthode de classification est peu performante. Par contre, si la méthode détecte un fort pourcentage des éléments à détecter sans détecter trop d'éléments n'appartenant pas à la classe, alors la méthode est efficace. Mais la valeur attribuée ici aux segments de contour n'est pas une valeur binaire (reconnue ou rejetée) mais une valeur réelle. Par conséquent le diagramme ROC ne peut être ici utilisé. Il faut alors définir un nouveau critère de quantification pour évaluer les performances de notre méthode.

Pour une évaluation efficace des valeurs correspondant à la vraisemblance de chacun des segments de contour, il faut tout d'abord que les segments de contour correspondant au contour de la silhouette aient la valeur P_{seg} définie par l'équation 3.13 la plus élevée possible. Soit fdp une fenêtre de détection positive. Soit $Silhouette$, l'ensemble des segments de contour correspondant au contour de la silhouette et $\overline{Silhouette}$ l'ensemble des segments de contour ne correspondant pas au contour de la silhouette. La somme des valeurs attribuées aux segments correspondant à la silhouette est notée :

$$S_+^{fdp} = \sum_{seg \in Silhouette} P_{seg} \tag{3.14}$$

Une valeur importante de S_+^{fdp} décrit une détection efficace. Mais, comme dans la représentation ROC, on désire également le moins de fausses détections possible. C'est-à-dire ici que l'algorithme favorise le moins possible les segments de contour ne correspondant pas à la silhouette. L'évaluation de l'importance des fausses détections est définie par la valeur :

$$S_-^{fdp} = \sum_{seg \in \overline{Silhouette}} P_{seg} \tag{3.15}$$

Une bonne représentation de la qualité d'une méthode se réalise sur un nombre important d'exemples testés afin de prendre en compte le plus vaste choix de configurations différentes. Prenons une base de données de fenêtres de détection positives $\triangle_{fdp}$. Une première évaluation du critère de quantification peut alors être :

$$C_q^{(1)} = \underset{fdp \in \triangle_{fdp}}{moyenne} \left(\frac{S_+^{fdp}}{S_-^{fdp}} \right) \tag{3.16}$$

Néanmoins, avec ce critère, une proportion moins importante de segments correspondant à la silhouette par rapport au nombre de segments ne correspondant pas à la silhouette engendre, quelles que soient les valeurs attribuées par la classification aux segments, une augmentation du critère. Ce phénomène n'est pas logique et l'évaluation de la méthode ne doit pas dépendre du nombre de segments dans une classe ou l'autre. Un arrière plan chargé ne doit pas influencer le critère. Les moyennes des valeurs attribuées aux segments correspondant à la silhouette puis ne lui correspondant pas sont définies par les valeurs :

$$\begin{cases} m_+^{fdp} = \underset{segment \in Silhouette}{moyenne} (P_{seg}) \\ m_-^{fdp} = \underset{segment \in \overline{Silhouette}}{moyenne} (P_{seg}) \end{cases} \tag{3.17}$$

Un second critère plus pertinent est alors défini par :

$$C_q^{(2)} = \underset{fdp \in \triangle_{fdp}}{moyenne} \left(\frac{m_+^{fdp}}{m_-^{fdp}} \right) \tag{3.18}$$

Mais la taille des segments n'est pas fixe. En effet certains contours peuvent être rectilignes sur une grande distance (notamment le long d'une jambe ou d'un bras) et d'autres sont courbés et sont donc modélisés par de petits segments de contours (comme par exemple autour de la tête). Avec le critère de l'équation 3.18, un segment de grande taille modifie autant le calcul du critère qu'un petit segment alors qu'il influence bien plus l'allure de la silhouette. Notons $T_{segment}$ la taille de *segment*. Sont alors définies, pour une fenêtre de détection positive fdp, la moyenne des valeurs attribuées aux pixels appartenant au contour de la silhouette et la moyenne des valeurs attribuées aux pixels n'appartenant pas au contour de la silhouette :

$$\begin{cases} M_{pos}^{fdp} = \dfrac{\underset{segment \in Silhouette}{moyenne} (P_{segment} \times T_{segment})}{\underset{segment \in Silhouette}{moyenne} (T_{segment})} \\ M_{neg}^{fdp} = \dfrac{\underset{segment \in \overline{Silhouette}}{moyenne} (P_{segment} \times T_{segment})}{\underset{segment \in \overline{Silhouette}}{moyenne} (T_{segment})} \end{cases} \tag{3.19}$$

Le critère de quantification le plus pertinent est finalement :

$$C_q = \underset{fdp \in \triangle_{fdp}}{moyenne} \left(\frac{M_{pos}^{fdp}}{M_{neg}^{fdp}} \right) \tag{3.20}$$

Plus ce critère est élevé, plus la performance de la méthode est élevée. Ce critère quantifie la qualité de la méthode et peut donc permettre de comparer des résultats.
Si la classification correspond parfaitement à la théorie, alors le critère vaut ∞. Si la classification attribue des valeurs nulles à tout les segments correspondants à la silhouette, alors le critère vaut 0. Donc, le critère est une valeur réelle croissante avec la qualité de la méthode. Il est utilisé comme critère de quantification de la pré-segmentation pour la suite.

3.4.2 Optimisation de la méthode

L'étude théorique réalisée dans la section 3.3 a permis de déterminer le principe de la méthode et le fonctionnement de l'algorithme. Certaines variables influencent les performances de l'algorithme. Des tests doivent donc être réalisés afin de déterminer les valeurs à choisir pour optimiser l'algorithme.

Effet du recouvrement

Le recouvrement des blocs doit permettre d'obtenir une plus grande précision sans nuire à la performance de la détection (voir section 3.3.1). Les valeurs du critère d'évaluation C_q de l'équation 3.20 sont très proches avec et sans recouvrement des blocs.

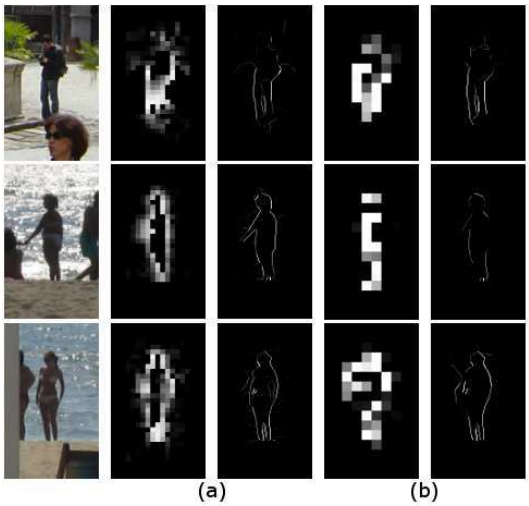

FIG. 3.10 – De gauche à droite : image originale, (a) sorties des SVMs et segments vraisemblables avec recouvrements des blocs et (b) mêmes résultats sans recouvrement des blocs. Le recouvrement des blocs élimine moins efficacement les segments périphériques mais donne des résultats plus fins.

Néanmoins l'approche qualitative révèle des différences (voir figure 3.10). Le recouvrement des blocs élimine moins facilement les segments sur les bords de la fenêtre. En effet, comme la valeur de sortie des SVMs d'une cellule dépend de tous les blocs qui la contiennent, elle est plus rarement proche de 0. Mais, le recouvrement permet de faire une étude plus fine spatialement. Or, le principe de cette méthode est de réaliser une étude plus locale. Le recouvrement des blocs permettant d'obtenir des résultats plus localisés (sur des ensembles plus petits), les performances en sont améliorées. En sortie des SVMs, la résolution est meilleur avec recouvrement des blocs que sans. La reconstitution de la silhouette est plus complète et les segments proches de la silhouette mais ne lui appartenant pas sont plus facilement écartés.

Effet de la taille des cellules et des blocs

La taille des cellules réalise le compromis entre la précision spatiale des sorties de SVMs et la pertinence des zones étudiées. En effet, une cellule large donne un même résultat sur une plus grande zone. La représentation de la silhouette en sortie des SVMs est donc plus grossière. De même, une cellule trop petite contient des formes peu caractéristiques et donne alors des informations sans grand intérêt. Une taille de bloc élevée donne un descripteur plus efficace (plus d'information) mais rend des résultats spatialement moins précis.
Sur la figure 3.11, le critère d'évaluation est calculé sur des tests réalisés avec la norme L2. Les blocs de 4 et de 16 cellules réalisent des recouvrements de $\frac{1}{2}$ alors que les blocs de 9 cellules réalisent un recouvrement de $\frac{1}{3}$. Les blocs de 4 cellules donnent les meilleurs résultats. Une information plus localisée est donc préférable. Les résultats pour les différentes tailles de cellules sont assez stables, mais, pour l'ensemble des autres tests, une taille de 25 pixels est utilisée car c'est celle qui répond le mieux au critère d'évaluation.

Ces résultats sont assez proches de ceux obtenus par Dalal [Dalal, 2006]. Les tailles de blocs et de cellules à favoriser y sont approximativement les mêmes. Il y a donc une bonne cohérence entre la détection et la segmentation. La transition se réalisera donc sans problème.

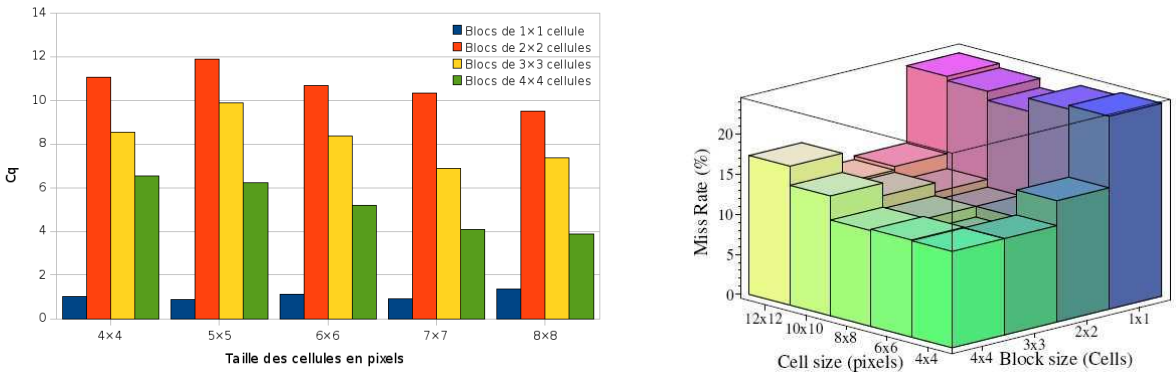

FIG. 3.11 – À gauche : caractéristiques des résultats obtenus pour différentes tailles de blocs et de cellules. Les cellules étant des carrés, la valeur affichée en abcisse est le nombre de pixels en composant le côté. À droite : effet de la taille des blocs et des cellules sur le taux d'erreur de l'algorithme de détection de Dalal [Dalal, 2006]. Les résultats pour la détection et la segmentation sont cohérents.

Effet de la normalisation

Pour éviter la dispersion entre les éléments d'une même classe provoquée par les variations d'éclairage, les HOGs sont normalisés.
Pour obtenir les résultats de la figure 3.12, les tests ont été réalisés avec des blocs de 4 cellules de 25 pixels avec recouvrement. Sont visualisés la moyenne des M_{pos}^{fdp}, la moyenne des M_{neg}^{fdp} et le critère d'évaluation. Les différentes normes testées sont celles définies en section 3.3.1.

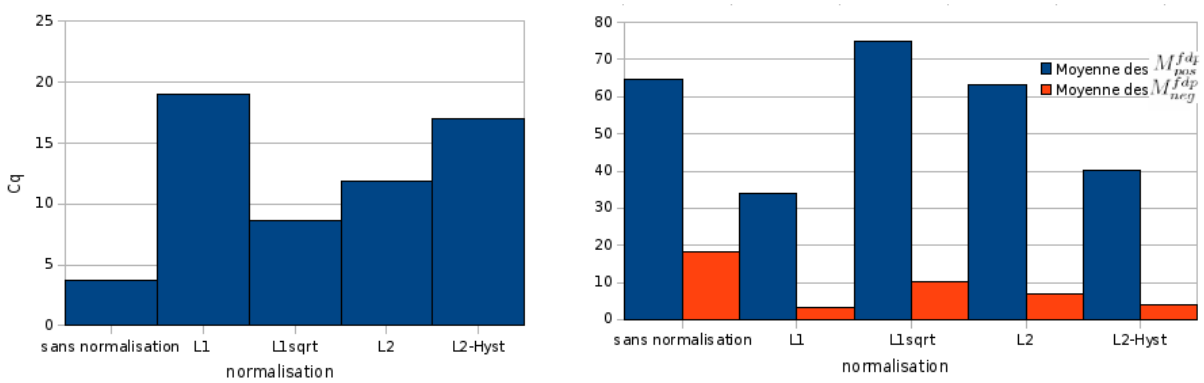

FIG. 3.12 – Evaluation des résultats obtenus pour différentes normalisations par le critère de quantification C_q puis par ses composantes M_{pos}^{fdp} et M_{neg}^{fdp}.

Sans normalisation, les variations entre les éléments d'une même classe sont plus importantes. Donc le descripteur est moins discriminant et la réponse au critère d'évaluation est moins bonne. C'est en effet ce qu'on observe en pratique. La normalisation est donc nécessaire pour obtenir les résultats optimaux.
Pour les différentes normes, le critère d'évaluation valorise les normes L1 et L2 à hystérésis. Néanmoins pour ces normalisations, la moyenne des M_{pos}^{fdp} est assez basse. Ce qui signifie qu'avec ces normes, la méthode est précise mais peu performante. En effet, une faible moyenne des M_{pos}^{fdp} signifie que les segments de la silhouette sont en moyenne peu favorisés et donc sont peu reconnus. Les segments n'appartenant pas à la silhouette sont donc bien éliminés. Mais les segments recherchés ne sont pas tous trouvés. Il y a alors des manques pour la suite de la méthode. La norme L2 semble un bon compromis.

3.4.3 Cas particuliers

Certaines particularités des images candidates peuvent provoquer des erreurs de calcul qui engendrent des dysfonctionnements de notre méthode. L'étude des formes permet de réaliser une reconnaissance assez robuste à de nombreuses variations. Mais ces cas particuliers sont les limites de la méthode et doivent être repérés.

- Pour réaliser la segmentation d'une personne, il est problématique que tous les contours de la silhouette ne soient pas dans la carte des contours calculée (figure 3.13a). En effet, localement, si la transition n'est pas assez franche entre la personne et l'arrière plan, certains contours de la silhouette peuvent manquer. N'apparaissant pas dans la liste des segments de contour de l'image, aucune valeur de vraisemblance ne leur est attribuée. Ils n'apparaissent pas dans la liste des segments candidats à appartenir au cycle de segments correspondant à la silhouette recherchée. Si cela n'a aucune influence sur les calculs du reste de l'image, la méthode de regroupement des segments pour la segmentation en est perturbée. En effet, la recherche du cycle se réalise de proche en proche. Ces manques obligent alors à rechercher des segments consécutifs plus éloignés spatialement.

- Une difficulté classique de la détection et de la segmentation est l'occultation (figure 3.13b). Un objet en premier plan cache une partie de la personne étudiée. Pour la détection, la silhouette est incomplète et est donc moins facilement reconnaissable. Pour la segmentation, il est plus difficile de délimiter un objet n'étant pas d'un seul tenant. Néanmoins, avec notre étude locale, chaque zone est étudiée séparément. Ainsi chaque segment peut être localement vraisemblable même s'il n'est pas rattaché aux autres segments vraisemblables. L'occultation ne défavorise pas la reconnaissance d'une sous-partie détachée du reste du corps (comme les pieds sur la figure 3.13b).

- Si l'image étudiée comporte un arrière plan trop chargé (figure 3.13c), les contours sont trop nombreux. Les proportions des segments de contour selon une orientation sont alors biaisées et donc moins bien représentatives de la forme qu'elles représentent. Les HOGs sont alors moins discriminants. La reconnaissance est dégradée et moins précise.

- Enfin l'illumination peut poser problème. En effet, les ombres créent des contours ne correspondant à aucun objet. Sur l'exemple (d) de la figure 3.13, seule une partie de la jambe est éclairée. Il y a un contour correspondant à la séparation de la jambe avec l'arrière plan et un contour correspondant à la séparation de la zone éclairée avec la zone à l'ombre. Ces deux contours sont très proches. Reconnaître lequel correspond à la silhouette est donc difficile.

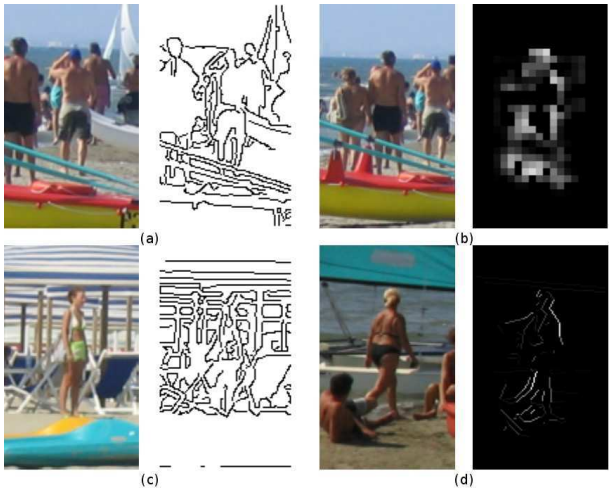

FIG. 3.13 – (a) Si des segments de la silhouette n'apparaissent pas dans le contour (bras droit), la segmentation est difficile. (b) En cas d'occultation, l'élément n'est pas favorisé mais ne perturbe pas le calcul sur les autres zones (même ici les pieds). (c) Un arrière plan chargé augmente la difficulté. (d) Les changements d'illumination donnent deux contours parallèles au niveau de la cuisse qui sont difficilement dissociés.

3.5 Conclusions

La méthode présentée dans ce chapitre réalise, à partir d'une fenêtre de détection positive, une pré-segmentation adaptée à la classe des personnes. Une valeur de vraisemblance en tant qu'élément de la silhouette est associée à chaque segment de contour de la fenêtre. Les SVMs, utilisés sur la fenêtre entière pour la détection, sont utilisés plus localement sur des zones plus petites appelées blocs. Des informations sur les éléments composant la fenêtre sont alors obtenues en vue de permettre la segmentation.

Les différents tests réalisés sur l'algorithme codé à partir de notre méthode ont donné des résultats cohérents où les segments de la silhouette sont bien reconnus tout en gardant un taux de fausse erreur (de valorisation de segments n'appartenant pas à la silhouette) assez bas.
Néanmoins, nous avons mis en avant en section 3.4.3 certaines limites de cette méthode. En effet, plusieurs situations entraînent des baisses de performance importantes. C'est le cas lorsqu'il y a présence de surcharge d'information dans l'arrière plan ou absence de certains segments de la silhouette dans la carte des contours. L'algorithme est cependant robuste à de nombreuses difficultés tel l'occultation ou la variation en couleur, texture et position.

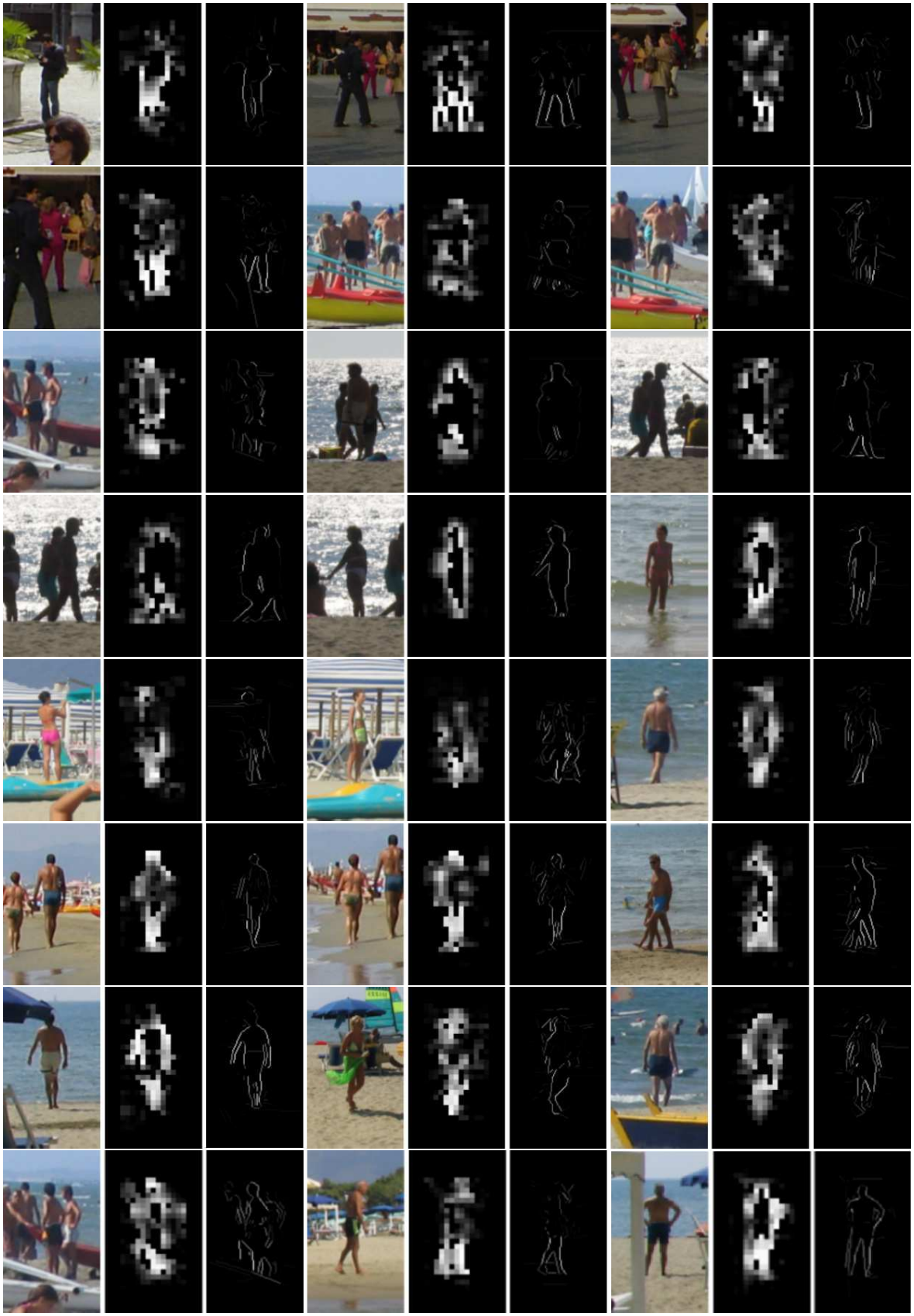

FIG. 3.14 – Première colonne : images de la base de données de L'INRIA. Seconde colonne : sortie des SVMs. Troisième colonne : segments vraisemblables. Ces résultats sont obtenus avec recouvrement, une normalisation L2 et des blocs de 4 cellules de 25 pixels.

Dans la plupart des cas, les résultats obtenus sont satisfaisants (figure 3.14). Ils doivent donc permettent de passer à l'étape suivante, à savoir la segmentation à proprement parler. En effet, l'objectif est désormais de trouver le cycle de segments de contour qui est associé à la silhouette de la personne recherchée. Notre pré-segmentation favorise les segments de la silhouette et défavorise ceux n'appartenant pas à la silhouette. La reconnaissance du cycle optimal est donc possible. Ce problème est traité dans le chapitre 4. Les travaux présentés dans ce chapitre ont été publiés pour la conférence VISAPP2010 [Migniot et al., 2010].

Chapitre 4

Reconstitution de la silhouette par recherche de plus court chemin à partir de la pré-segmentation

> Se demander si un ordinateur peut penser est aussi intéressant que de se demander si un sous-marin peut nager.
>
> Edsger Dijkstra

Sommaire

La segmentation d'un élément d'une image consiste à déterminer quels pixels appartiennent à cet élément et quels pixels appartiennent à l'arrière plan. Dans le cas de la segmentation de personnes, l'élément à segmenter est d'un seul tenant (ou en nombre fini de partie si un autre objet crée une occultation séparant la vision de l'ensemble de la personne).
Le contour de transition entre les pixels du premier et de l'arrière plan est caractéristique de la classe des personnes. Ce contour peut être divisé en segments. Ces segments forment un cycle qui entoure la personne recherchée. Chaîner ces segments revient alors à obtenir la segmentation voulue.
La méthode présentée dans ce chapitre utilise cette approche. Les segments correspondant à la silhouette sont choisis à partir des valeurs de vraisemblance obtenues par la pré-segmentation du chapitre 3.
La section 4.1 présente de façon précise la notion de segments de contour et les perspectives de cet outil. Le cycle de segments correspondant à la silhouette est obtenu grâce à un algorithme de recherche du plus court chemin dans un graphe comme expliqué en section 4.2. Cet algorithme doit cependant être adapté à notre étude. Les modifications nécessaires sont précisées en section 4.3. La section 4.4 montre que les résultats sont améliorés avec un processus itératif de notre méthode. Enfin la section 4.5 tire les conclusions des résultats obtenus.

4.1 Utilisation des segments de contour pour la segmentation

La pré-segmentation du chapitre 3 assigne à chaque segment de contour de la fenêtre une valeur de vraisemblance en tant qu'élément de la silhouette humaine. Ces données sont utilisées afin de déterminer le cycle de segments qui permet de reconstituer la silhouette.
Cette section établit la pertinence de l'utilisation des segments de contour pour la segmentation. Un état de l'art des méthodes exploitant cette approche justifie notre choix d'utiliser un algorithme de recherche de plus court chemin.

4.1.1 Caractérisation et notations des segments de contour

Les segments de contour sont une modélisation des contours d'une image en une suite de segments. Ils sont définis par leur taille, leur orientation et leur localisation. Ils sont donc une caractérisation locale du contour plus consistante que le pixel de contour.
Un cycle doit être formé à partir des segments de contour. Or, comme l'ensemble des contours de la silhouette ne sont pas forcement présents dans la carte des contours, deux segments successifs dans le cycle ne sont pas forcement contigus. Des transitions inter-segments sont donc nécessaires pour reconstituer la silhouette. Le sens du segment est donc important. Un segment de contour est alors, dans notre étude, caractérisé par son point de départ p^d, son point d'arrivé p^a (qui sont ses deux extrémités) et sa vraisemblance notée v obtenue par la pré-segmentation. Chaque segment de contour seg est alors caractérisé par le vecteur :

$$V_{seg} = \{p^d_{seg}, p^a_{seg}, v_{seg}\} \tag{4.1}$$

Ces caractéristiques donnent facilement la taille du segment T_{seg} ainsi que son orientation θ_{seg}. La vectorisation de la carte des contours donne alors un ensemble Δ_v de N_s segments de contour.

4.1.2 Méthodes utilisant des éléments du contour pour la segmentation

La forme de la silhouette est discriminante de nombreuses classes. Les contours sont donc très couramment utilisés pour la détection et la segmentation. Les sous-parties des contours d'une image sont des outils interessants que de nombreuses méthodes exploitent.

Shotton [Shotton et al., 2008a] utilise les fragments de contour (lignes continues du contour) pour la reconnaissance d'objet. Il montre qu'un nombre restreint de ces fragments de contour permet une bonne classification, s'ils sont bien choisis. Il introduit un modèle composé de fragments caractérisant différentes sous-parties définies par leur distance au centroïde de l'objet et une valeur d'incertitude (représentant la variation de la posture des éléments de la classe recherchée). À partir d'un centroïde fixé et d'une échelle, la méthode recherche, via une comparaison par distance de chanfrein aux éléments d'un catalogue de fragments de contours, les éléments de l'image testée correspondant à chaque sous-partie. La vraisemblance de l'ensemble des comparaisons donne une décision sur la présence de l'élément de la classe. Une application de l'algorithme du Mean Shift permet de ne garder qu'une détection par objet de l'image.
Ferrari [Ferrari et al., 2006] découpe les contours en segments. Puis, il réalise un réseau de segments en reliant les segments spatialement proches. À partir d'un segment de départ, il cherche la succession de segments la plus pertinente pour reconstruire le contour recherché. Pour cela, il prend en compte la vraisemblance de la continuité entre segments ainsi qu'un *a priori* sur la forme donné par un apprentissage.
Haugeard [Haugeard and Philipp-Foliguet, 2010] réalise un graphe relationnel attribué (ARG) dont les noeuds sont les segments de contours d'une image. Il définit ensuite une mesure de similarité entre le graphe d'une image et un graphe de référence en comparant les chemins du graphe de l'image avec les chemins partant de chaque noeud du graphe de référence. La sélection des images contenant un élément de la classe s'effectue à l'aide d'un classifieur SVM.

Wu [Wu and Nevatia, 2007] réalise une double classification : une pour la détection et une pour la segmentation. La première recherche la localisation de parties du contour caractéristiques de la classe. La seconde, en fonction des parties de contour localisées, donne l'appartenance d'un pixel au premier ou à l'arrière plan. Ainsi détection et segmentation sont réalisées simultanément.

Elder [Elder and Zucker, 1996] cherche lui à fermer des contours, c'est-à-dire à trouver les segments qui permettent d'entourer complètement les différentes formes d'une image. Il utilise pour cela les tangentes aux contours qui modélisent des ensembles de pixels de contours successifs et d'orientations proches. À partir d'un modèle bayésien, il exprime la probabilité de deux tangentes de se succéder dans un contour. Ce modèle prend notamment en compte les différents cas possibles de transition entre deux segments consécutifs pour une certaine forme. Ce modèle est explicité en détail dans l'annexe C.
La liaison entre ces tangentes est réalisée par un algorithme de recherche de plus court chemin, à partir d'un graphe dont les noeuds sont les tangentes et les liaisons sont pondérées par les résultats du modèle bayésien. Un certain nombre de contraintes sont aussi présentées afin d'éviter des topologies absurdes (croisement du cycle de contour en forme de "8" par exemple).
Sharma [Sharma and Davis, 2007] utilise lui les segments de contour pour la segmentation de personnes. Il définit les segments comme un ensemble de deux points extrêmes p_1 et p_2, de la magnitude moyenne du gradient le long du segment E_{mag} et de la composante moyenne du flot optique le long du segment ν_x et ν_y ($seg = [p_1, p_2, E_{mag}, \nu_x, \nu_y]$). Il cherche ensuite les cycles les plus probables de la même façon qu'Elder [Elder and Zucker, 1996]. Puis, il réalise un champ de Markov aléatoire à partir des segments de contours avec des relations dépendantes de la probabilité qu'on les deux segments de se suivre (voir section 4.3.2) mais aussi du nombre de cycles pertinents auxquels les segments appartiennent. Une coupe de graphe permet finalement d'obtenir le masque réalisant la segmentation.

L'idée de rechercher les cycles de plus court chemin est très intéressante. [Elder and Zucker, 1996] ne l'utilise que pour la fermeture de contour et ne prend donc en compte que la continuité. [Sharma and Davis, 2007] l'utilise pour trouver des cycles complets mais n'intègre aucune connaissance sur la classe recherchée. Il ne trouve alors que l'ensemble des formes de l'image. L'*a priori* obtenu par le chapitre 3 peut être intégré dans le graphe étudié. Les cycles obtenus suivent alors uniquement les contours de la silhouette recherchée. Il suffit d'augmenter le poids relatif aux arêtes des segments peu vraisemblable.
Le choix que nous prenons est donc de réaliser, à partir de la vectorisation de l'image testée, un graphe dont les noeuds sont les segments de contour de l'image, les arêtes sont les liaisons possibles entre segments pour la création d'un cycle. Les poids liés aux arêtes dépendent à la fois de la continuité (spatiale et d'orientation) des deux segments mais aussi de la vraisemblance des segments d'appartenir à la silhouette (pré-segmentation).

4.2 Recherche de plus court chemin

Le cycle optimal est obtenu à partir d'un graphe orienté sur lequel est appliqué un algorithme de recherche du plus court chemin. Ce problème est traité depuis longtemps

et des méthodes donnant de très bons résultats avec un temps de traitement très court ont été développées. Voici une rapide présentation du problème et de ces solutions.

La recherche du plus court chemin est une application de la théorie des graphes. Soit $\mathcal{G}$ un graphe composé de noeuds et d'arêtes. Les noeuds sont un ensemble de points représentant les données que le graphe étudie. Les noeuds qui ont une certaine relation sont liés par une arête. Un poids est associé à chaque arête qui représente une quantification de la relation entre les deux éléments représentés par les noeuds relatifs à cette arête. Un chemin est une liste ordonnée d'arêtes successives qui peuvent ête utilisées pour aller d'un noeud de départ à un noeud d'arrivée.
Dans un graphe orienté, une arête (appelée alors arc) réalise le passage d'un noeud (appelé alors sommet) à un autre. Un arc possède donc un sens. Si deux arcs sont consécutifs dans un chemin, alors le noeud d'arrivée du premier arc est le noeud de départ du second arc.
La recherche du plus court chemin dans un graphe orienté est :

- Soit le chemin optimal d'un sommet i à un sommet f ; c'est-à-dire le chemin qui, à partir du sommet i, mène au sommet f, et dont la somme des poids des arcs traversés est minimale.
- Soit l'arbre optimal à partir d'un sommet source s. Cet arbre contient tous les arcs des chemins optimaux de s vers tous les autres sommets.

Cette approche permet de résoudre de nombreux problèmes dans des domaines très différents comme l'informatique (algorithmique, base de données, réseau...) ou les mathématiques (optimisation combinatoire, probabilité, algèbre...).

4.2.1 Algorithme de Dijkstra

L'algorithme de Dijkstra permet de déterminer le plus court chemin dans un graphe connexe orienté dont les poids liés aux arcs sont positifs ou nuls. Il porte le nom de son inventeur et a été publié en 1959 [Dijkstra, 1959].
L'arbre optimal se détermine progressivement en assignant les sommets par ordre croissant de la somme des poids des arcs traversés pour le rejoindre à partir du sommet source.
Soit $\mathcal{S}= [s_i, i \in [1, N]]$, l'ensemble des sommets du graphe et $\mathcal{A}= [a_{ij}, (i, j) \in [1, N]^2]$ l'ensemble des arcs. a_{ij} est l'arc réalisant le passage du sommet s_i au sommet s_j. Il est pondéré par le poids ω_{ij}.
À chaque sommet s_i est associé un poids ω_i initialisé à ∞. Le sommet source est assigné et associé à un poids égale à 0. Les sommets sont ensuite assignés successivement selon l'itération suivante :

- Pour tout arc a_{ij} réalisant le passage entre un sommet s_i déjà assigné et un sommet s_j non encore assigné, le poids $\omega_j = \omega_i + \omega_{ij}$. Le poids ω_j du sommet s_j est remplacé par ω_j si celui-ci lui est inférieur ($\omega_j \rightarrow \omega$ si $\omega < \omega_j$).
- Le sommet non encore assigné dont le poids associé est le plus faible est assigné à son tour et le dernier arc utilisé pour le rejoindre est ajouté à l'arbre. Si un autre chemin permettait d'obtenir un poids moins important, il faudrait qu'il passe par un sommet non encore assigné. Mais comme parvenir à ces sommets demande déjà un poids plus important et que les poids associés aux arêtes sont positifs, il s'agit bien du chemin optimal pour ce sommet.

À chaque itération, un nouveau sommet est assigné. L'arbre contient tous les arcs du chemin optimal de ce segment à la source. Lorsque tous les sommets sont assignés, l'arbre optimal est réalisé et l'algorithme s'arête. L'ensemble est résumé dans l'algorithme 2.

Algorithm 2 Algorithme de recherche de l'arbre optimal par Dijkstra

Entrées : Graphe orienté et sommet source
Sorties : Ensemble des arcs empruntés par l'arbre optimal

Initialisation :
$\forall i \neq source, \omega_i = \infty$
$\omega_{source} = 0$, $S = \{source\}$, $A = \{\emptyset\}$

Tant qu'il existe un sommet s vérifiant $s \notin S$:
- Pour tout arc a_{ij} tel que $i \in S$ et $j \notin S$, si $\omega = \omega_i + \omega_{ij} < \omega_j$ alors $\omega_j = \omega$
- Si $s = arg\min_i\{\omega_i, i \notin S\}$, alors :
  $S = S \cup \{s\}$
  $A = A \cup \{a_{.s}\}$

4.2.2 Algorithme de Dantzig-Ford

Cet algorithme, créé par [Dantzig et al., 1956], fonctionne pour la recherche du plus court chemin (ou du plus grand) dans un graphe orienté pouvant être avec ou sans circuit et dont les poids des arcs peuvent être positifs ou négatifs (contrairement à l'algorithme de Dijkstra). L'algorithme améliore un arbre jusqu'à obtenir celui qui est optimal.
Soit un arbre du graphe déterminé aléatoirement. Le poids ω_i est le cumul des poids des arcs traversés pour aller de la source au sommet s_i par le chemin désigné par l'arbre. Le sommet a pour antécédent dans $\mathcal{A}$ le sommet s_i^a.
À chaque itération, l'algorithme teste chacun des sommets. Pour le sommet s_j, s'il existe un arc a_{ij} allant d'un sommet s_i à un sommet s_j tel que $\omega_i + \omega_{ij} < \omega_j$, alors le sommet s_i devient l'antécédent du sommet s_j et le poids ω_j devient égale à $\omega_i + \omega_{ij}$.
Les chemins menant à chaque sommet dans l'arbre sont donc plus courts à chaque itération. Si aucun antécédent n'a été modifié pendant une itération, l'algorithme s'arête. L'arbre optimal $\mathcal{A}$ est formé des arcs allant jusqu'à un sommet s_i à partir de son antécédent s_i^a. Le processus dure au maximum un nombre d'itérations égale au nombre de sommets. Il est résumé dans l'algorithme 3.

4.2.3 Conclusion et choix de l'algorithme de recherche de l'arbre optimal

Comme les méthodes de la section 4.1.2 le montrent il est plus judicieux, avec l'utilisation des segments comme noeuds du graphe, de chercher les segments successifs un par un. C'est le cas avec l'algorithme de Dijkstra. De plus, ce dernier est le moins gourmand en calcul. Enfin, l'algorithme de Dantzig Ford nécessite un premier arbre à établir initialement. Notre choix se porte alors sur l'algorithme De Dijkstra que nous utiliserons pour la suite.

Algorithm 3 Algorithme de recherche de l'arbre optimal par Dantzig Ford
Entrées : Graphe orienté et sommet source
Sorties : Arbre optimal

Initialisation : À partir d'un arbre aléatoire du graphe, calcul pour chaque sommet s_i de son antécédant s_i^a et du poids ω_i

Tant qu'au moins un antécédant a été modifié dans l'itération précédente :
- Pour tout sommet s_j
  si $\exists a_{ij} | \omega_i + \omega_{ij} < \omega_j$, alors $\omega_j = \omega_i + \omega_{ij}$ et $s_j^a = s_i$

Néanmoins, le cumul des poids entraîne une tendance de l'algorithme à vouloir minimiser le nombre de sommets traversés. Ceci entraîne l'oubli de certains membres (bras, jambes) lors de l'évaluation du plus court chemin (voir section 4.3.5). Nous avons donc testé une recherche de chemin optimal qui ne minimise pas le cumul des poids mais leur moyenne. Ainsi, l'algorithme peut favoriser sans problème un chemin traversant un nombre important d'arcs, si ceux-ci sont vraisemblables. Le passage par les bras et les jambes n'est alors pas défavorisé.
Ce processus est impossible à obtenir pour l'algorithme de Dijkstra. En effet, comme l'ajout d'un nouveau terme peut diminuer la moyenne et donc le poids assigné à un sommet, l'assignation en fin d'itération n'est pas certaine. Le principe de l'algorithme de Dijkstra n'est donc pas respecté.
L'expérimentation montre cependant que, si l'algorithme minimise moins le nombre de sommets traversés, il favorise maintenant des chemins maximisant le nombre de segments. Les résultats présentés en section 4.3.5 montrent qu'au final, l'algorithme de Dijkstra donne de meilleurs résultats. Nous restons donc sur le choix de ce dernier.

4.3 Adaptation de l'algorithme pour notre approche

Soit $\mathcal{G}$ le graphe qui nous permettra de résoudre notre problème. Ses sommets sont les segments de contour de l'image. Deux segments susceptibles de se succéder dans un cycle doivent former un arc dans le graphe. Cet arc est pondéré par les valeurs de vraisemblances des segments données par la pré-segmentation. Il est aussi pondéré par la continuité entre ces deux segments. Le chemin le plus court pour passer d'un segment à un autre peut être déterminé par l'algorithme de Dijkstra.

Néanmoins, certaines particularités de notre problème font que de nouvelles contraintes doivent être appliquées à l'algorithme tel que défini en section 4.2.1 pour certifier des résultats cohérents. L'algorithme est alors légèrement transformé.

4.3.1 Réalisation du graphe

L'élément d'étude est le segment. Le segment est défini par deux points, il ne s'agit pas d'une donnée ponctuelle comme usuellement dans les résolutions par graphe. Un contour est formé à partir d'une succession de segments. Mais ce contour dépend à la fois de l'ordre dans lequel les segments sont pris mais aussi du sens dans lequel ils sont pris. En

effet, une transition entre deux vecteurs peut partir d'une des deux extrémités du premier vers une des deux extrémités du second. Le sens de passage d'un segment change donc le contour obtenu (figure 4.1).

FIG. 4.1 – Selon le sens dans lequel un segment est pris, le contour obtenu est grandement modifié.

Un sommet du graphe représente donc le passage par un segment de contour selon un sens. Le nombre de sommets du graphe est ainsi le double du nombre de segments de contour dans l'image. Pour éviter un contour réalisant des boucles, lorsqu'un sommet est assigné (et ne peut donc plus être choisi par la suite), le sommet représentant le passage par ce même segment mais dans l'autre sens est aussi assigné.

Il existe quatre transitions possibles pour aller d'un segment S_d à un segment S_a (figure 4.2). Le contour recherché est une suite de segments et de transitions. Une transition relie donc une extrémité du segment précédant S_d à une extrémité de S_d. La transition vers S_a commence donc par son autre extrémité. Seule la transition la plus courte vers une extrémité de S_a est finalement gardée et forme un arc dans $\mathcal{G}$.
Il y a donc un arc entre chaque sommet et la moitié des autres sommets. Donc si l'image

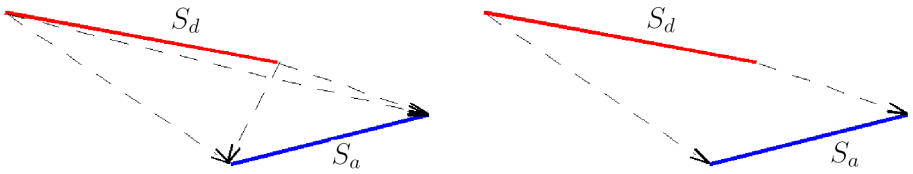

FIG. 4.2 – La transition du segment rouge au segment bleu peut se faire de quatre façons possibles (à gauche). Elle commence de l'extrémité non encore utilisé du premier segment et va à l'extrémité du second segment la plus proche (à droite).

comporte N_s segments de contour, le graphe est composé de N_s sommets et $2N_s(N_s - 1)$ arcs. Ce dernier nombre est trop important. Les calculs seraient trop importants et le processus trop lent et moins efficace. Seuls les segments spatialement proches ont une forte probabilité de se suivre dans le contour. Seules les transitions dont la taille est inférieure à un seuil fixé forment donc un arc dans $\mathcal{G}$. Le nombre d'arcs est ainsi grandement réduit. Pour finir, la valeur d'affinité, définie dans la section 4.3.2, détermine le poids de chaque arc du graphe.

4.3.2 Affinité entre segments

Le poids associé aux arcs d'un graphe correspond à la difficulté du passage d'un chemin par cet arc. Pour défavoriser le passage d'un sommet à un autre, la transition

les liant est associé à un poids élevé. Par conséquent, dans notre graphe, le poids d'un arc reliant deux segments est inversement proportionnel à la probabilité que les deux segments se suivent dans le contour recherché. Cette probabilité est appelée **affinité**. Elle est en relation avec deux caractéristiques :

- **La continuité :** il s'agit de la cohérence entre les deux segments. Pour que le contour formé ne soit pas trop abrupt, il faut que les deux segments aient une orientation proche. Le contour est aussi plus pertinent si les deux segments sont spatialement proches. Cette notion ne prend pas en compte la connaissance sur la classe recherchée.

- **La vraisemblance :** il s'agit de l'intégration de la connaissance de la classe recherchée. Si le segment d'arrivée est vraisemblable en tant qu'élément d'une silhouette humaine, le passage par ce segment est favorisé. Ainsi, le contour suit la forme de la silhouette.

L'affinité dans la litterature

Sharma [Sharma and Davis, 2007] recherche les cycles probables d'une image sans spécification sur une classe en particulier. Sa valeur d'affinité ne prend donc en compte que la continuité. Elle est définie pour un couple de deux segments seg_1 et seg_2 par :

$$Aff(seg_1, seg_2) = e^{-\frac{r}{\sigma_r}}.e^{-\frac{\beta}{\sigma_\beta}}.e^{-\frac{\Delta}{\sigma_\Delta}} \tag{4.2}$$

où r représente la distance minimale entre deux points de chacun des deux segments, $\beta = \theta_1^2 + \theta_2^2 - \theta_1\theta_2$ et $\Delta = |I_{seg_1} - I_{seg_2}|$. θ_1 est l'angle entre le segment seg_1 et la transition liant seg_1 et seg_2. θ_2 est la valeur correspondante pour le segment seg_2 (figure 4.3). I_{seg_1} et I_{seg_2} sont les moyennes de la magnitude du gradient le long des segments seg_1 et seg_2. Les constantes σ_r, σ_β et σ_Δ permettent de régler l'influence que doivent avoir chacun des trois termes dans le calcul. Dans un second temps, Sharma rajoute un terme pour introduire le mouvement.

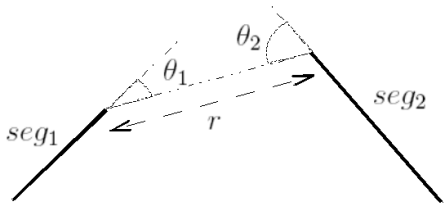

FIG. 4.3 – Caractéristiques utiles pour le calcul de l'affinité entre deux segments : la distance r entre les deux segments et les angles θ_1 et θ_2 relatifs à leurs orientations.

Elder [Elder and Zucker, 1996] n'utilise pas non plus d'*a priori* sur la classe recherchée. Il recherche uniquement la meilleure façon de fermer les contours. Pour cela, il réalise également un graphe et pondère les arcs par la probabilité qu'ont deux segments (en vérité dans son étude il s'agit de deux tangentes aux contours) de se succéder. Le calcul de ces pondérations se réalise selon une étude bayésienne qui prend notamment en compte trois configurations possibles de passage entre deux segments : la courbure

(où le contour se courbe doucement), l'interruption du contour (occultation) et le coin (où la courbure du contour est franche). Le calcul considère l'orientation des segments, leur longueur, la distance les séparant et les moyennes d'intensité de l'image le long de chaque côté du segment. Les détails du calcul se trouvent en annexe C.

Quel que soit le graphe, un arc de forte pondération a moins de chance de faire partie du plus court chemin. La pondération est logiquement décroissante avec l'affinité. Dans la littérature, l'évaluation du poids ω_{ij} d'un arc reliant les deux segments seg_i et seg_j est réalisée par :

$$\omega_{ij} = -log(Aff(seg_i, seg_j)) \tag{4.3}$$

Affinité choisie

Pour des formes humaines vues à petites échelles, l'orientation du contour le long de la silhouette risque d'être très abrupte (au niveau des aisselles, des mains, des pieds ou de la tête). La continuité d'orientation est alors peu pertinente. La continuité en intensité du gradient correspond à la différence d'intensité entre premier et arrière plan. Elle est intéressante lorsque la couleur du premier plan est presque uniforme. Or, la variation en couleur sur une personne est élevée (notamment au niveau des habits).
Mais d'autres données sont plus pertinentes :

- Si deux segments sont spatialement éloignés, il est moins probable qu'ils se succèdent dans un contour.
- La vectorisation donne des segments de tailles irrégulières. Or, il faut moins de segments pour aller d'un point à un autre si ceux ci sont grands. Ils doivent donc être défavorisés.

Soit T_{seg} la taille du segment seg. L'affinité est tout d'abord définie par :

$$Aff(seg_i, seg_j) = e^{-\frac{r}{\sigma_r}}.e^{-\frac{T_{seg_j}}{\sigma_T}} \tag{4.4}$$

Cette définition ne prend pas en compte l'apprentissage de la classe. La valeur de vraisemblance de la pré-segmentation noté $P(seg)$, d'après l'équation 3.13, représente la probabilité qu'un segment a d'appartenir à la silhouette recherchée. Le passage par un segment probable doit être favorisé. L'affinité liée à une transition vers ce segment doit donc être élevée.
Le contour formé par un chemin est constitué des segments mais aussi des transitions. Si la transition forme un contour improbable, l'affinité doit être réduite. On soumet donc les transitions aux SVMs déja calculées. Soit $trans_{ij}$ la transition du segment seg_i au segment seg_j. L'équation 3.13 lui associe la vraisemblance $P(trans_{ij})$. L'affinité est alors finalement définie par :

$$Aff(seg_i, seg_j) = e^{-\frac{r}{P(seg_j)\sigma_r}}.e^{-\alpha\frac{T_{seg_j}}{P(trans_{ij})\sigma_T}} \tag{4.5}$$

où α représente le rapport d'influence sur la pondération entre le terme correspondant à la transition et le terme correspondant au segment.
La pondération est déterminée par l'équation 4.3.

4.3.3 Difficultés propres au problème

Le graphe est constitué. Le problème peut maintenant être résolu. Pour cela nous utilisons l'algorithme de recherche de plus court chemin de Dijkstra. Celui-ci doit être quelque peu modifié pour s'adapter aux spécificités du problème. De meilleures performances sont acquises en rajoutant certaines contraintes.

Format des sommets

Un sommet représente un segment pris dans un sens. Lorsqu'un sommet est assigné, le sommet relatif au même segment mais pris dans l'autre sens doit aussi être assigné. Par contre, la transition empruntée est la seule ajoutée à l'arbre.

Deux chemins pour réaliser le cycle

L'algorithme de Dijkstra fonctionne sur le cumul des poids d'un chemin. Un chemin passant par peu de sommets est donc plus probablement le plus court. Or, nous recherchons un cycle, c'est-à-dire un chemin allant d'un sommet à lui-même. Le plus court chemin a donc plus de chance de prendre quelques segments autour de celui pris pour source, même si ceux-ci sont peu vraisemblables, que de faire le tour de la silhouette, même si les segments correspondants ont une forte valeur de vraisemblance. Le cycle de plus court chemin risque de n'être qu'une partie de la silhouette.
Il faut donc mieux rechercher le plus court chemin entre deux segment éloignés. Le cycle est donc décomposé en deux chemins. Deux segments sources s_1 et s_2 assez éloignés sont choisis. L'algorithme de Dijkstra est ensuite utilisé deux fois pour trouver le plus court chemin de s_1 à s_2 puis de s_2 à s_1. La concaténation des deux chemins forme le cycle.
Les deux segments sources doivent être représentatifs de la classe, spatialement stables dans cette classe et le plus éloignés possible. La figure 4.4 montre la moyenne des contours des images Pour une meilleur continuité, les ordres de passage des extrémités de s_1 et s_2

FIG. 4.4 – Par filtrage de Canny, les contours de 1000 images de la base statique de personnes de l'INRIÀ sont calculés. Cette image est la moyenne des 1000 cartes de contour obtenues.

dans les deux chemins sont inversés.
Les deux chemins ne doivent pas se chevaucher (au risque d'obtenir une forme d'aire nulle). Le second chemin ne doit donc pas emprunter les sommets du premier. Pour cela, au début de l'algorithme de plus court chemin, tous les sommets du premier chemin (excepté s_1) sont assignés.

Choix des segments sources

Pour des raisons d'autonomie des traitements, les deux segments sources doivent être définis automatiquement. Soit s_t le segment correspondant au sommet de la tête et s_p le sommet correspondant aux plats des pieds. Le premier chemin est celui allant de s_t à s_p. Ces deux segments sont choisis parmi tout ceux donnés par la vectorisation de la carte des contours. Il faut qu'ils soient à la bonne position, à la bonne orientation et qu'ils soient vraisemblables selon la pré-segmentation. Pour un segment de contour s, notons x et y les coordonnées de son point milieu, θ son orientation et v la valeur de vraisemblance fournie par la pré-segmentation. Nous modélisons la probabilité que ce segment soit un des segments sources par une gaussienne :

$$\begin{cases} P(s = s_t) = e^{-\frac{(x-\mu_{x_t})^2}{2\sigma_{x_t}^2}}.e^{-\frac{(y-\mu_{y_t})^2}{2\sigma_{y_t}^2}}.e^{-\frac{(\theta-\mu_\theta)^2}{2\sigma_\theta^2}}.e^{-\frac{(v-\mu_v)^2}{2\sigma_v^2}} \\ P(s = s_p) = e^{-\frac{(x-\mu_{x_p})^2}{2\sigma_{x_p}^2}}.e^{-\frac{(y-\mu_{y_p})^2}{2\sigma_{y_p}^2}}.e^{-\frac{(\theta-\mu_\theta)^2}{2\sigma_\theta^2}}.e^{-\frac{(v-\mu_v)^2}{2\sigma_v^2}} \end{cases} \tag{4.6}$$

Les différentes valeurs μ et σ utilisées dans ces formules correspondent à la position optimale et à la dispersion possible des variables. Ces valeurs sont déterminées à partir de la figure 4.4. Par exemple, la dispersion verticale est moins importante que celle horizontale. σ_v représente lui uniquement l'influence de la vraisemblance du segment par rapport aux autres caractéristiques. Les différentes valeurs que nous avons calculées puis utilisées par la suite sont répertoriées dans le tableau de la figure 4.5
Les segments sources sont ceux maximisant ces probabilités.

	x_t	y_t	x_p	y_p	θ	v
μ	47	32	47	122	0	1
σ	2,7	1,2	3,3	1,8	0,5	0,27

FIG. 4.5 – Variables des gaussiennes liées aux différentes caractéristiques.

4.3.4 Algorithme complet

L'algorithme complet de recherche du cycle optimal que nous avons retenu est le suivant.
Le graphe $\mathcal{G}$ est réalisé comme décrit dans la section 4.3.1. Puis, les segments sources sont obtenus par l'équation 4.6. L'algorithme de Dijkstra calcule ensuite l'arbre de plus court chemin à partir de s_t. L'algorithme s'arrête dès que s_p est assigné et donne l'arbre $\mathcal{A}_1$. Comme, dans un arbre, un sommet n'a qu'un seul antécédent, le chemin $\mathcal{C}_1$ de s_t à s_p est reconstitué (en commençant par la fin). Le processus de Dijkstra est réinitialisé et tous les segments traversés par $\mathcal{C}_1$ sont assignés. L'algorithme de Dijkstra à partir du segment s_p, arêté dès l'assignation de s_t, donne l'arbre $\mathcal{A}_2$. Comme précédemment, le chemin $\mathcal{C}_2$ de s_p à s_t est déterminé. Le cycle optimal $\mathcal{C}$ est alors la concaténation de $\mathcal{C}_1$ et $\mathcal{C}_2$.

4.3.5 Premiers résultats

Le premier résultat à évaluer est le choix des segments sources. Pour la grande majorité des cas, ce choix automatique selon les équations 4.6 donne des résultats très satisfaisants. Les segments appartiennent bien à la silhouette et correspondent bien aux parties du corps espérées. Néanmoins, lorsque le sujet a les jambes bien écartées (par exemple s'il court) et qu'un contour se trouve entre ses jambes avec une orientation horizontale, alors ce contour est choisi pour être s_p. En effet, les segments représentant ces contours entre les jambes ont souvent une vraisemblance moyenne et sont proches de la localisation favorisée. La segmentation obtenue contient alors l'espace entre les deux jambes au lieu de parcourir l'intérieur des jambes (figure 4.6(a)).
L'algorithme de Dijkstra est cumulatif, donc les contours courts sont favorisés. Lorsque des ombres forment des contours verticaux au niveau du torse, les bras sont parfois coupés (figure 4.6(b)).
Le seuil déterminant si la transition entre deux segments est suffisamment longue pour former un arc dans le graphe (voir section 4.3.1) est difficile à fixer. S'il est trop élevé, le nombre d'arcs est trop important et l'algorithme est moins performant. S'il est trop faible, le contour correspondant à la silhouette peut ne correspondre à aucun chemin possible du graphe. La forme obtenue réalise alors des détours non pertinents (figure 4.6(c)).
La segmentation obtenue est performante mais est souvent mise en défaut. La méthode peut être améliorée. La section 4.4 présente une version itérative de notre approche. Les défauts d'une itération sont corrigés dans la suivante grâce à la pré-segmentation.

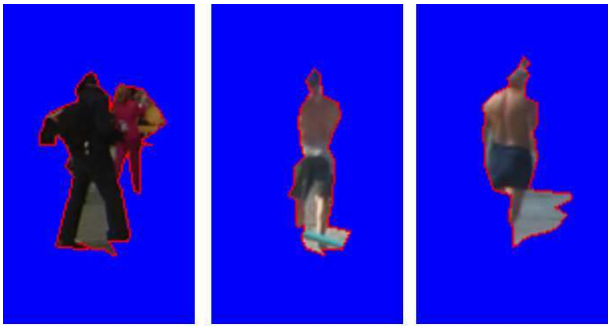

Fig. 4.6 – Plusieurs résultats du cycle obtenu par notre algorithme (en rouge) dans des cas qui posent problème. (a) Les jambes de la personne étant écartées, le segment s_p est mal évalué. Le cycle passe par s_p entre les jambes et ne suit donc pas l'intérieur des jambes. (b) Vu la mauvaise évaluation des contours, le bras de la personne a été tronqué. (c) La distance autorisée entre deux segments étant trop petite, le cycle fait un détour non pertinent même si la vraisemblance à la silhouette handicape celui-ci.

4.4 Processus itératif

Comme nous l'avons vu dans la section 4.3.5, la méthode que nous utilisons est souvent perturbée ce qui engendre des résultats parfois aberrants. En intégrant une nouvelle fois les informations d'*a priori* que nous possédons, les résultats sont améliorés. Nous allons transformer notre méthode en un processus itératif.

4.4.1 Raisons et aboutissements

Pour minimiser la taille du graphe $\mathcal{G}$, seules les transitions dont la taille est inférieure à un seuil forment des arcs. Ce seuil représente donc la distance maximale entre deux segments consécutifs. Le contour formé par le cycle est constitué des segments et des transitions. Mais seuls les segments sont des éléments de l'image. Il faut donc éviter que les transitions représentent une proportion trop importante du contour. De plus, de petites transitions favorisent le suivi des segments vraisemblables. Prendre un seuil assez bas est donc judicieux.
Néanmoins, la carte des contours est souvent incomplète. Des éléments du contour de la silhouette peuvent manquer. Dans ce cas, si les transitions longues sont exclues, l'algorithme cherche un chemin possible même s'il doit passer par des segments peu vraisemblables. Il réalise alors des détours illogiques.
Mais ces digressions sont peu courantes, localisées et facilement repérables. La solution est de réaliser une nouvelle itération du processus sur les parties du contour posant problème avec une valeur du seuil augmentée. Ainsi, la précision d'un seuil bas est obtenue sans l'inconvénient des détours inopportuns.
Les segments traversés par le cycle de l'itération précédente et reconnus comme invraisemblables sont défavorisés dans la nouvelle itération. Ainsi le cycle peut s'améliorer même si la hausse du seuil ne suffit pas.
Nous utilisons une nouvelle fois les données de la pré-segmentation pour reconnaître les parties invraisemblables du contour. Réutiliser les données d'apprentissage rend le processus plus discriminant.

4.4.2 Fonctionnement

Nous testons ici une approche itérative de notre méthode. À chaque itération, un nouveau cycle est calculé. Le contour associé correspond de mieux en mieux à la silhouette. Pour cela, les suites de segments qui semblent mal correspondre au cycle espéré sont repérées. L'algorithme de plus court chemin est appliquée sur chacune de ces suites en adaptant certaines caractéristiques du graphe pour remplacer les segments douteux.

Première itération

La première itération du processus et identique à celle présentée en section 4.2. Si aucune suite de segments ne pose problème, le processus s'arrête. Sinon, un premier cycle est obtenu que les itérations suivantes vont améliorer.

Évaluation des segments du cycle précédent

Au début d'une itération, un cycle est disponible dont il faut trouver les segments aberrants. Le cycle forme une silhouette. Nous réalisons une image binaire à partir de cette silhouette (silhouette blanche sur un fond noir). Cette image permet une évaluation de la pré-segmentation du chapitre 3. Elle est divisée en blocs et en cellules. Les HOGs des cellules sont calculés. Chaque SVM renvoie une valeur SVM^{bloc}. L'équation 3.10 donne les valeurs SVM^{cell}. Enfin, l'équation 3.13 associe à chaque segment du contour une valeur de vraisemblance. C'est cette valeur qui détermine les segments posant problème dans le cycle.

Les classifications SVM étant déjà réalisées, les calculs supplémentaires sont faibles. De plus, comme la base d'apprentissage positive est réalisée à partir d'images de silhouettes binaires (voir section 3.3.1 et annexe A), utiliser la méthode à partir d'une image de silhouette binaire est cohérent.

Recherche des sections posant problème

Pour que la nouvelle itération n'altère pas les résultats de la précédente, seuls les segments posant problèmes sont modifiés. On désigne comme problématiques, les segments dont la valeur de vraisemblance est en dessous d'un seuil.
Soit une suite de segments problématiques dans le cycle. Soit s_1 le segment les précédant et s_2 le segment les succédant. Ni s_1 ni s_2 ne sont problématiques. À partir du graphe $\mathcal{G}$, le plus court chemin de s_1 à s_2 est calculé. Les segments du chemin alors obtenu remplacent les anciens. Quelques modifications sur le graphe sont réalisées pour améliorer les performances comme nous le montrons ci-après.

Nouvelle recherche de plus court chemin

Pour chaque itération k, on réalise un nouveau graphe $\mathcal{G}^k$. Le seuil de la taille des transitions augmente avec k. La sélection des arcs est donc moins sélective et ceux-ci sont donc plus nombreux dans le graphe.
Un nouveau chemin est recherché pour remplacer celui peu vraisemblable. Ce chemin doit éviter d'emprunter les segments empruntés par l'ancien. Soit un segment s de vraisemblance P_s inférieur au seuil $S^{(k)}$ à l'itération k. Le poids ω^{k-1} associé à tout arc menant à s dans $\mathcal{G}^k$ est réactualisé en ω^k :

$$\omega^k = \frac{1 + S^{(k)}}{1 + P_s + \varepsilon}\omega^{k-1} \tag{4.7}$$

Comme P_s a valeur dans $[-1, S_v]$, ε est un réel positif très petit empêchant la division par 0. Plus le segment est invraisemblable, plus il est défavorisé.
Enfin, le nouveau chemin ne doit pas traverser les segments non problématiques de l'itération précédente. Ces derniers sont donc assignés en initialisation de l'algorithme de Dijkstra.

Évaluation de la nouvelle itération

Le nouveau cycle obtenu est évalué. La vraisemblance des segments du cycle est calculée comme vu précédemment. On choisit comme critère d'évaluation la moyenne, sur tous les segments du cycle, de la vraisemblance pondérée par la taille du segment.
Si ce critère est supérieur à celui calculé à l'itération précédente, alors l'itération a amélioré la segmentation et une nouvelle itération est effectuée. Dans le cas contraire, le processus est stoppé et le cycle de l'itération précédente donne la segmentation finale.

Résultats

Si le cycle est pertinent dès la première itération, le processus s'arrête et le résultat est le même que précédemment.
Sinon, le passage par plusieurs itérations réduit fortement les parties non pertinentes.

L'évolution du contour est parfois abrupte mais la silhouette est au final plus proche de la réalité.
L'évaluation précise (et notamment quantitative) du processus se trouve en section 4.5.

4.5 Résultats

4.5.1 Critères d'évaluation

Les segmentations obtenues par notre méthode sont évaluées, par l'intermédiaire de tests, dans cette section. Les performances sont jugées selon une évaluation qualitative mais aussi une évaluation quantitative. Cette dernière permet en effet une comparaison de résultats proches. Elle permet aussi de vérifier la robustesse de la méthode en manipulant les résultats de segmentation sur un nombre important d'exemples. Un critère d'évaluation doit alors être désigné.

L'algorithme testé fournit, pour chaque exemple, une segmentation. Soit M le masque qui, pour un pixel, renvoie 1 si la segmentation assigne ce pixel au premier plan (la personne) et renvoie 0 si elle l'assigne à l'arrière plan.
Pour quantifier la qualité d'une segmentation, il faut la comparer à une réalité terrain qui correspond à la segmentation idéale. Nous créons cette réalité terrain manuellement pour chaque exemple testé. Soit G le masque qui, pour un pixel, renvoie 1 si la vérité terrain assigne le pixel au premier plan et renvoie 0 si elle l'assigne à l'arrière plan. Soit N le nombre de pixels dans l'image testée.
Dans de nombreux papiers, le critère retenu est le *taux d'erreur* de la segmentation ou la *précision* de la segmentation.
Le *taux d'erreur* ε d'une segmentation, notamment utilisé par [Blake et al., 2004], représente le pourcentage de pixels mal classifiés par la segmentation étudiée. Plus ce taux est faible, plus la segmentation est de bonne qualité.

$$\varepsilon = \frac{card\{x|M(x) \neq G(x)\}}{N} \tag{4.8}$$

La *précision* γ d'une segmentation représente la proportion de pixels correctement classifiés au premier ou à l'arrière plan. Elle est très fortement liée à la proportion précédente. Elle est notamment utilisée par [Kumar et al., 2005], [Lin et al., 2007b] et [Winn and Shotton, 2006]. Plus la segmentation est performante et plus la précision est élevée.

$$\gamma = 1 - \frac{card\{x|M(x) \neq G(x)\}}{N} \tag{4.9}$$

Ces valeurs donnent une bonne évaluation. Néanmoins, elles sont incomplètes. Certains pixels mal attribués altèrent plus une segmentation que d'autres. Ce point n'est pas intégré dans ces critères.
Philipp-Foliguet [Philipp-Foliguet and Guigues, 2006] réalise un état de l'art des critères d'évaluation d'une segmentation parmi lesquels nous en avons sélectionné trois pertinents.

F-Measure

La segmentation est une détection des pixels du premier plan. Des mesures usuelles pour la détection sont la précision et le rappel. La précision représente la proportion d'éléments correctement attribués à la classe recherchée par rapport au nombre d'éléments qui ont été attribués à cette classe. Une bonne précision indique que les pixels attribués par la méthode à la silhouette le sont avec une bonne fiabilité.

$$précision = \frac{card\{x|M(x) = G(x) = 1\}}{card\{x|M(x) = 1\}} \tag{4.10}$$

Le rappel représente la proportion d'éléments correctement attribués à la classe recherchée par rapport au nombre d'éléments qui appartiennent réellement à cette classe. Un bon rappel signifie que la majeure partie des pixels appartenant à la silhouette ont bien été assignés par notre méthode.

$$rappel = \frac{card\{x|M(x) = G(x) = 1\}}{card\{x|G(x) = 1\}} \tag{4.11}$$

Une bonne segmentation est alors une segmentation qui donne une précision et un rappel élevé. La F-measure, définie par [Rijsbergen, 1979], est un indicateur liant ces deux critères. Elle est définie par :

$$F_{measure} = \frac{2 \times precision \times rappel}{precision + rappel} \tag{4.12}$$

Dans le meilleur des cas, les deux masques sont identiques et la précision et le rappel valent 1. Par conséquent, la F-measure vaut 1. Au contraire, dans le pire des cas, les deux masques sont opposés. La précision et le rappel valent donc 0. La F-measure tend alors vers 0. La $F_{measure}$ a valeur dans $[0, 1]$ et est croissante avec la performance de la segmentation.

La mesure de Martin

La mesure de Martin [Martin, 2002] est au départ une mesure pour évaluer la cohérence entre deux segmentations réalisées par deux méthodes. Il suffit ici de l'utiliser avec une segmentation idéale. L'intérêt de cette mesure est qu'elle sépare les différentes configurations : fausse détection, non détection ... La mesure vérifie que chacune d'elles est performante. Soit un pixel x. Une erreur du masque de segmentation par rapport au masque idéal donne :

$$E(x) = \frac{card\{y|G(y) = G(x), M(y) \neq M(x)\}}{card\{y|G(y) = G(x)\}} \tag{4.13}$$

Puis une erreur du masque idéal par rapport au masque de segmentation donne :

$$E'(x) = \frac{card\{y|M(y) = M(x), G(y) \neq G(x)\}}{card\{y|M(y) = M(x)\}} \tag{4.14}$$

La mesure de Martin détermine finalement la dissimilarité entre ces deux masques par :

$$M_{Martin} = \frac{1}{N} \sum_{i=0}^{N} \min\{E(x_i), E'(x_i)\} \tag{4.15}$$

Cette mesure a valeur dans $[0, 1]$ et est décroissante avec la performance de segmentation.

La mesure de Yasnoff

La mesure de Yasnoff [Yasnoff et al., 1979] se base sur le principe qu'un pixel éloigné de la région auquel il devrait appartenir perturbe plus le résultat qu'un pixel moins éloigné. En effet, si le premier plan obtenu est légèrement plus mince que voulu, le rendu est moins affecté que si un élément manque ou est ajouté.
En notant $d(x, y)$ la distance euclidienne entre deux pixels x et y, la distance $d(x)$ du pixel x au pixel le proche de même région dans la vérité terrain est définie par : $d(x) = arg\min_d\{d(x, y)|G(y) = M(x)\}$ (figure 4.7). La mesure de Yasnoff est définie par :

$$M_{Yasnoff} = \frac{100}{N}\sqrt{\sum_{i=0}^{N} d(x_i)^2} \tag{4.16}$$

Cette mesure a valeur dans $[0, +\infty[$ et est décroissante avec la performance de la segmentation.

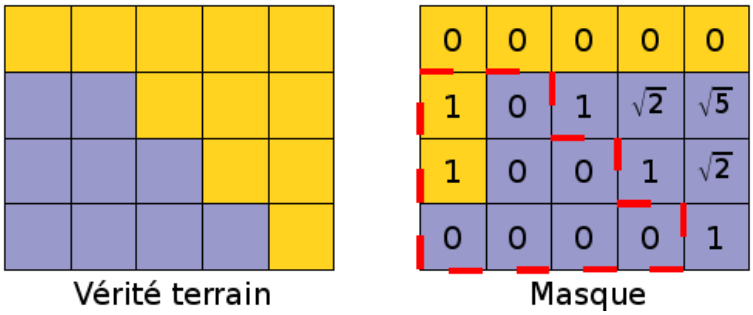

FIG. 4.7 – Calcul de la distance $d(x)$ de la mesure de Yasnoff pour le cas simple d'un image 5×4 pixels : à gauche les deux régions (premier et arrière plan) de la vérité terrain et à droite les deux régions du masque de segmentation avec la distance d relative à chaque pixel.

4.5.2 Évaluation de la méthode

Nous réalisons la segmentation de 400 images issues de la base de données statiques de personnes de l'INRIA. La base de données représentant un échantillon assez représentatif de la classe recherchée. Nous avons réalisé une segmentation manuelle pour chaque image testée qui correspond à la vérité terrain et sert de référence (disponible librement sur le site : *http ://www.gipsa-lab.inpg.fr/ cyrille.migniot/recherches.html*). Pour chaque segmentation, les trois critères d'évaluation sont calculés. L'évaluation de la méthode est donnée par la moyenne de chacun des critères sur l'ensemble des tests.

4.5.3 Choix de l'algorithme de recherche du plus court chemin

Il faut justifier notre choix de l'algorithme de recherche du plus court chemin utilisé. Nous avons testé les algorithmes de Dijkstra (section 4.2.1) et de Dantzig Ford selon la moyenne (section 4.2.2).

La pondération étant cumulative pour l'algorithme de Dijkstra, le processus favorise toujours les chemins comportant le moins de segments possible. Si une petite perturbation arrive (ombre, absence de détection de certains segments,...), alors toute inflection au trajet, comme le passage par les bras, est supprimée et certains éléments de la silhouette sont alors oubliés (figure 4.6(b)).
Au contraire, l'algorithme de Dantzig-Ford avec la moyenne favorise les chemins comportant le plus de segments possibles. Des détours sans pertinence sont alors obtenus (figure 4.8).
La distance euclidienne maximale entre deux segments successifs est fixé à 10. Les

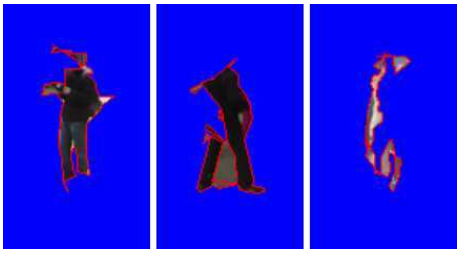

FIG. 4.8 – En utilisant l'algorithme de Dantzig-Ford, les chemins passant par un maximum de segments sont favorisés. Des détours aberrants sont alors obtenus.

moyennes des différents critères d'évaluation pour l'utilisation des deux algorithmes se trouvent dans le tableau 4.9. L'algorithme de Dijkstra donne des meilleurs résultats selon les trois critères. Il est donc le plus pertinent. C'est pourquoi nous continuons de l'utiliser par la suite.

Mesure	Dijkstra	Dantzig-Ford
F-measure	0,7443	0,6772
Martin	0,0773	0,0973
Yasnoff	1.4258	2,2845

FIG. 4.9 – Résultats des évaluations obtenues avec plusieurs algorithmes de recherche de plus court chemin. Les trois mesures montrent la supériorité de l'algorithme de Dijkstra.

4.5.4 Seuil à optimiser

Le seuil de distance acceptable entre deux segments pour créer un arc (voir section 4.3.1), permet d'assurer la continuité spatiale du contour formé par le cycle. En effet, avec un seuil bas, seul les segments proches peuvent se succéder dans le cycle. Le contour suit donc mieux les informations de l'image. Néanmoins comme certains segments de la silhouette peuvent ne pas être détectés lors du processus de recherche des contours, un passage de proche en proche peut s'avérer impossible. Pour relier un segment à un autre, le cycle devra alors réaliser des détours (figure 4.6(c)) même si les segments traversés sont reconnus comme improbables.
Les critères d'évaluation permettent de déterminer la valeur de ce seuil qui optimise les performances de notre méthode.
Des séries de tests sont réalisées pour différentes valeurs du seuil. Les segments *initial* et $final$ sont déterminés automatiquement. Les résultats obtenus sont présentés sur la figure

4.10. La $F_{measure}$ et la mesure de Martin donnent les meilleures performances pour les petites valeurs du seuil. Néanmoins, la mesure de Yasnoff, qui prend en compte l'importance de la localisation des pixels mal attribués et qui est donc plus proche des impressions visuelles, valorise le seuil de 9. Ce critère valorise un seuil de façon bien plus franche. Cette valeur de 9 est ainsi conservée pour la suite.

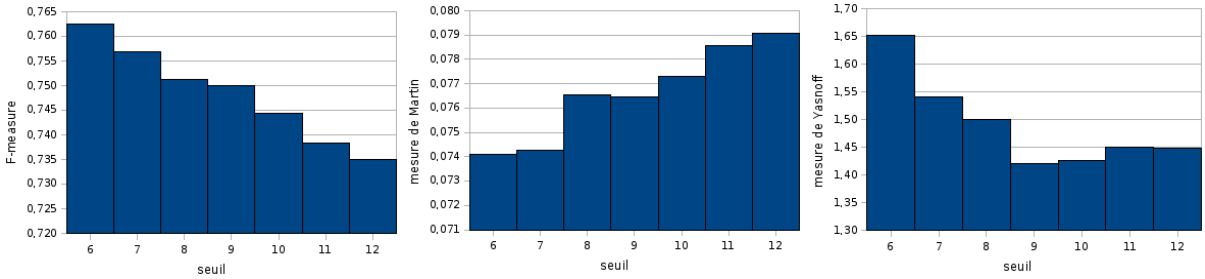

FIG. 4.10 – Performances obtenues par notre méthode pour différentes valeurs du seuil par trois critères d'évaluation. De gauche à droite sont visualisées la F-measure et les mesures de Martin et de Yasnoff.

4.5.5 Pondération des arcs

La seconde variable de l'algorithme est le paramètre α, utilisé dans l'équation 4.5. Elle représente le rapport d'influence, dans la pondération de chaque arc, entre le passage dans un segment de contour et le passage dans la transition. Les résultats de la figure 4.11 sont obtenus à partir d'une série de 400 tests.
Si $\alpha = 0$ et que les transitions ne sont pas prises en compte lors de la pondération, les performances sont clairement inférieures. À partir d'une valeur de 4, les résultats sont assez constants. Il faut donc que le terme concernant la transition soit légèrement plus important que le terme concernant le segment. C'est assez logique puisque, pour que le cycle suive bien le contour, il faut qu'il s'adapte en priorité aux caractéristiques de l'image. Néanmoins l'autre terme est aussi important et doit donc être pris en compte. La meilleure valeur selon la $F_{measure}$ et la mesure de Yasnoff est $\alpha = 5$. Nous gardons donc cette valeur pour la suite.

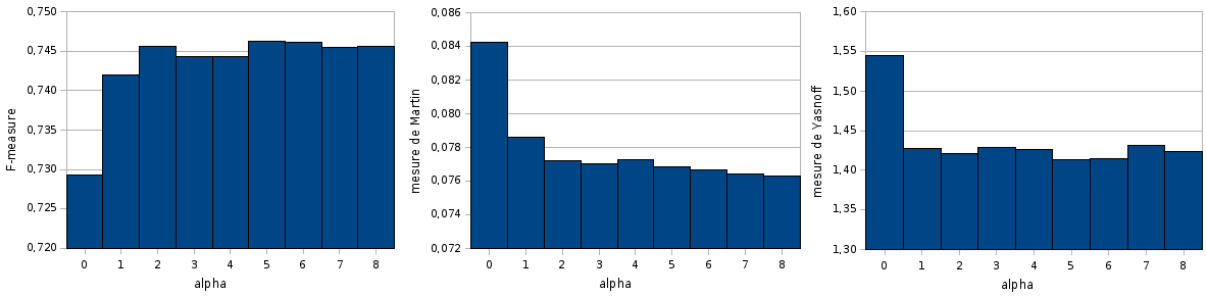

FIG. 4.11 – Influence du paramètre α sur les performances de la segmentation selon les trois critères d'évaluation.

4.5.6 Choix des segments extrêmes

Le premier choix à réaliser pour commencer la recherche du cycle optimal est celui des segments sources s_t et s_p. Pour la grande majorité des cas, un choix automatique selon les équations 4.6 donne des résultats très satisfaisants.
Si le sujet étudié a les jambes bien écartées, la détection du segment source s_p peut être erronée. Le cycle ne prend alors pas en compte le dessous des jambes (figure 4.6(a)).
Pour évaluer l'effet engendré par la détection automatique de ces segments nous comparons, dans le tableau 4.12, les moyennes des différents critères d'évaluation sur 30 exemples avec une détection automatique des segments sources tout d'abord, puis avec une désignation manuelle (réalisée par l'utilisateur) de ces segments. Les performances sont moins élevées dans le second cas selon nos trois critères. Le choix plus astucieux fait par l'estimation automatique compense donc avantageusement les erreurs dues au cas présenté plus haut.

Mesure	Automatique	Manuelle
F-measure	0,7932	0,7740
Martin	0,0581	0,0601
Yasnoff	0,9685	1,0988

FIG. 4.12 – Mesures des évaluations obtenues avec une désignation automatique puis manuelle des segments sources. La désignation automatique est la meilleure selon les trois critères.

4.5.7 Le processus itératif

Une première estimation de la silhouette est obtenue. Lorsque celle-ci est insuffisante, réaliser plusieurs itérations du processus (comme expliqué section 4.4) améliore la segmentation.

Chaque itération repère les défauts évidents du cycle résultant de l'itération précédente et le modifie pour mieux correspondre à la silhouette. Une première itération est donc réalisée avec un seuil fixé à 10 comme vu plus haut. Puis, dans la suite, les rectifications se réalisent à partir d'un seuil plus élevé donnant des contraintes plus souples. Ainsi la première itération donne une première estimation assez fiable, puis les suivantes éliminent les erreurs dues au manque de certains éléments du contour de la silhouette dans la carte des contours calculée. Comme la figure 4.13 le montre, les détours incohérents sont alors éliminés.
En utilisant les critères d'évaluation sur un ensemble de 400 images tests, les moyennes obtenues donnent les résultats du tableau 4.14. Les trois critères démontrent alors bien l'amélioration apportée par cette modification de la méthode. La baisse importante de la mesure de Yasnoff montre que le rendu est visuellement beaucoup plus proche du résultat attendu.
Sur les 400 images testées, le nombre moyen d'itérations nécessaires pour obtenir la segmentation finale est de $2,26$.

La méthode complète est résumée sur l'organigramme de la figure 4.15.

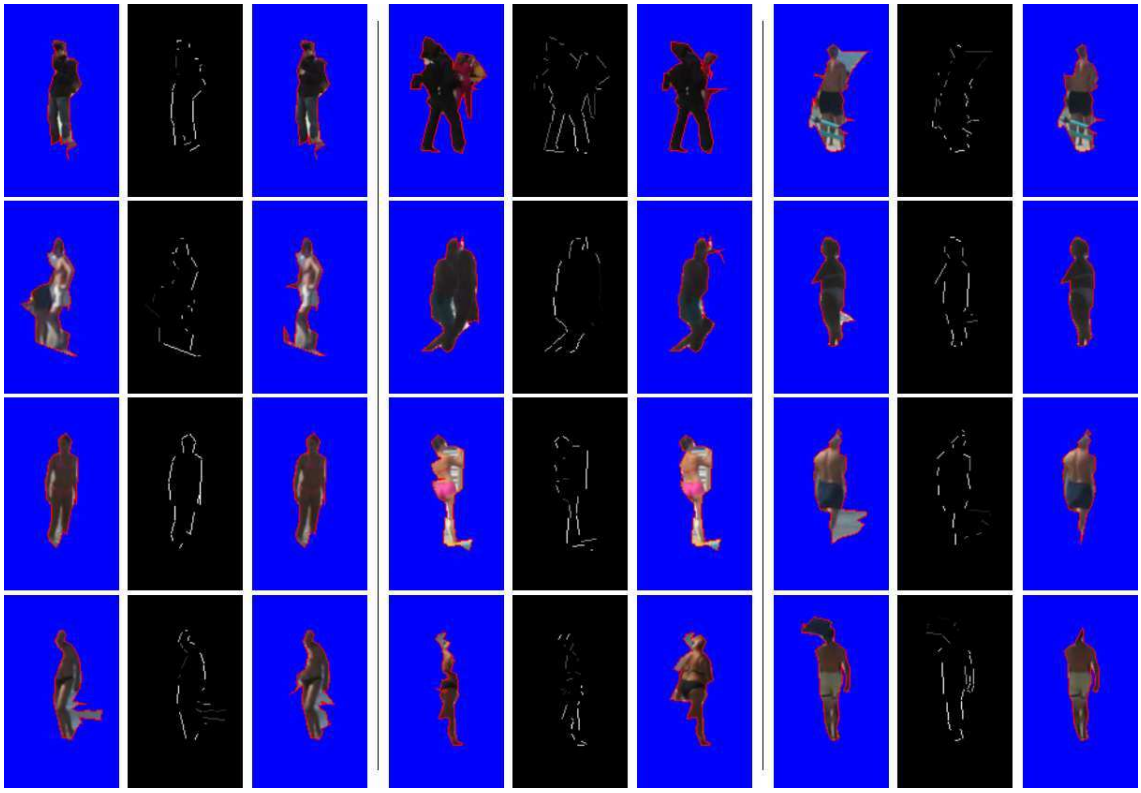

FIG. 4.13 – Effet du passage de une à plusieurs itérations du processus sur les performances de la segmentation. Utiliser plusieurs itérations permet d'éliminer les éléments mal attribués lors de la première. Pour chaque triplet : segmentation obtenue avec une seule itération (à gauche), vraisemblance des segments du cycle obtenue par la première itération (au centre) et segmentation obtenue avec plusieurs itérations (à droite).

Mesure	Une itération	Plusieurs itérations
F-measure	0,7443	0,8028
Martin	0,0774	0,0657
Yasnoff	1,4258	0,8476

FIG. 4.14 – Mesures des évaluations obtenues avec une seule itération puis avec plusieurs itérations.

4.6 Passage à la vidéo : un nouvel apport d'information

Jusqu'à présent, notre étude s'est basée sur des images fixes. Dans cette section, nous adaptons notre étude au cas de la vidéo.

Deux objectifs sont alors visés :

- Réussir à minimiser le temps de calcul pour la segmentation dans une séquence. En effet, les *frames* successives d'une séquence comportent une forte redondance. Réaliser le processus de segmentation pour chaque *frame* est alors long et inapproprié. Les *frames* précédentes (ou suivantes) étant assez proches d'une *frame*, elles peuvent alors guider sa segmentation.

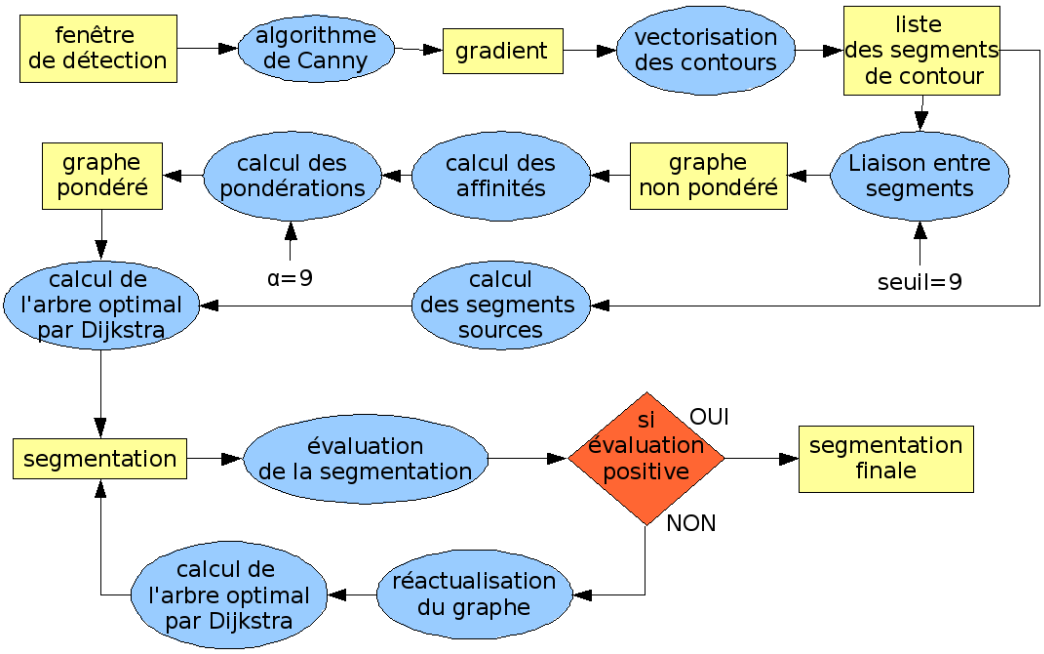

FIG. 4.15 – Organigramme de la méthode complète présentée dans ce chapitre avec la valeur optimale des différents paramètres.

- Ajouter aux processus de détection et de segmentation les informations de mouvement qui sont assez descriptifs de la classe recherchée. En effet, le mouvement régulier de balancement des jambes d'une personne qui se déplace perpendiculairement à l'axe optique est bien caractéristique des personnes.

Dans un premier temps, nous avons discuté dans l'état de l'art de l'utilisation du mouvement dans le contexte de la détection de personnes. Mais l'analyse du mouvement a de nombreuses applications dans l'étude de la classe des personnes. Tout d'abord, il est possible, en plus de réaliser la détection, de reconnaître le mouvement de la personne (s'asseoir, courir, sauter...) [Sidenbladh et al., 2000][Polana and Nelson, 1994][Bobick et al., 2001][Laptev, 2006].
Ensuite, l'étude du mouvement peut permettre à un véhicule de reconnaître un piéton et d'évaluer le temps restant avant la collision [Elzein et al., 2003]. Elle peut aussi permettre de déterminer le mouvement d'un avatar imitant une personne filmée [Bregler and Malik, 1998].
En restant dans le cadre de la détection et de la segmentation, la vidéo dispense une aide précieuse. Une différence entre *frames* successives donne une liste réduite de candidats [Zhao and Thorpe, 2000][Zhao and Nevatia, 2003]. Zaklouta compare les fenêtres de détection relatives à une même personne sur plusieurs *frames* par une corrélation des vecteurs de caractéristiques [Zaklouta, 2012]. Il repère et élimine ensuite les candidats immobiles pour réduire le nombre de fausses détections.
Associer informations d'apparence (statique) et de mouvement (dynamique) améliore souvent les résultats. Cela est possible en ajoutant des dimensions au vecteur de caractéristiques d'un descripteur [Viola et al., 2003][Abd-Almageed et al., 2007] (par exemple avec des histogrammes de flots optiques [Dalal et al., 2006][Laptev, 2006]). [Andriluka et al., 2008] fusionne, lui, les données statiques et dynamiques dans un modèle

de Markov caché. [Fablet and Black, 2002] et [Wu and Yu, 2006] rajoutent un terme fonction de la dynamique temporelle à un système probabiliste basé sur la ressemblance statique [Fablet and Black, 2002][Wu and Yu, 2006].
L'étude du mouvement a aussi pour but de minimiser le temps de calculs. Appliquer une méthode de détection ou de segmentation à chaque *frame* d'une séquence demande beaucoup de calculs. Or, une estimation de la localisation et de la pose de la personne peut être réalisée à partir des états précédents [Haritaoglu et al., 1999][Polana and Nelson, 1994]. Cette estimation peut être réalisée à partir d'un filtrage de Kalman [Bertozzi et al., 2003][Bertozzi et al., 2004].
Enfin, certaines méthodes changent l'espace d'étude. [Murai et al., 2007] réalise une détection par reconnaissance de la matrice gradient 3D spatio-temporelle (x,y,t). [Haga et al., 2004], lui, reporte les informations sur une espace basé sur le mouvement. Il utilise en effet un espace dont les axes sont l'unicité spatiale du mouvement, l'unicité temporelle du mouvement et la continuité temporelle.
Cette partie vise à expliquer comment nous avons incorporé le mouvement dans notre travail. Une présentation du flot optique, l'outil qui nous permet de quantifier le mouvement, est faite en section 4.6.1. Ensuite, la section 4.6.2 explique comment nous intégrons ces informations dans le descripteur pour modifier les performances de reconnaissance des segments de contours. Puis, la section 4.6.3 commente l'utilisation du mouvement pour guider le processus de segmentation. Enfin, la section 4.6.4 tire les conclusions de cette partie.

4.6.1 Le flot optique

Afin d'utiliser le mouvement des éléments d'une vidéo pour la détection ou la segmentation, il faut réussir à quantifier le mouvement. Le flot optique permet d'évaluer le mouvement de chaque pixel d'une *frame*.
Le calcul du flot optique se base sur l'étude de couples de *frames* successives dans la séquence. La figure 4.16 donne quelques exemples de ces couples de *frames*.

FIG. 4.16 – Suites de deux *frames* consécutives utilisées pour le calcul du flot optique.

Problématique

Le flot optique est un champ qui permet de définir le déplacement d'un point (pixel) de l'image entre deux instants. Pour obtenir une évaluation de celui-ci, l'hypothèse est faite que la couleur d'un point de la scène est indépendante du temps. Nous considérons le flot comme pixelique.
Donc, le vecteur $[\nu_x, \nu_y]$ le long duquel la dérivée de l'image $I_t(x, y)$ est nulle (avec x et y la position du point) est recherché.

Il vérifie l'équation :

$$\frac{\partial I_t(x,y)}{\partial x}\nu_x + \frac{\partial I_t(x,y)}{\partial y}\nu_y + \frac{\partial I_t(x,y)}{\partial t} = 0 \tag{4.17}$$

Cette équation ne permet pas de déterminer de manière unique le flot optique. En effet, en chaque point de la scène, une seule contrainte scalaire est disponible pour déterminer un vecteur de deux coefficients (v1,v2). De nouvelles contraintes sont imposées pour estimer cette information.

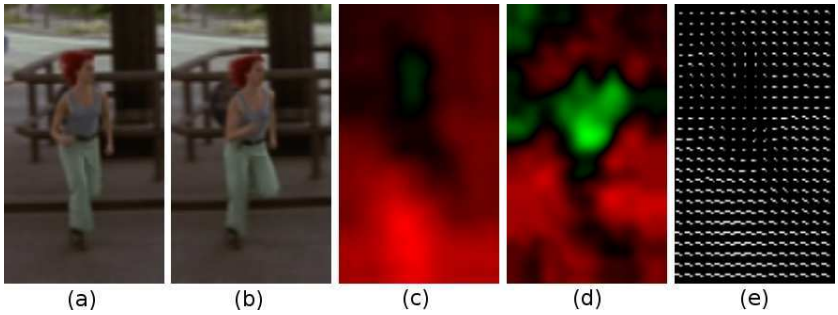

FIG. 4.17 – Flot optique calculé à partir de la méthode de Bruno et Pellerin [Bruno and Pellerin, 2002]. À partir de deux *frames* successives (a) et (b), un vecteur de flot optique est obtenu pour chaque pixel. Il est défini par une composante du mouvement selon l'axe horizontal (c) et d'une composante du mouvement selon l'axe vertical (d). (e) représente le déplacement de certains pixels entre les deux *frames* à partir des deux composantes du flot optique.

Méthode d'estimation du flot optique

Pour Bouguet [Bouguet, 2000], la contrainte choisie est, pour chaque pixel p de la *frame*, de choisir le vecteur de vélocité (les composantes du flot optique) qui minimise la différence d'intensité entre les pixels de la première *frame* dans une fenêtre d'étude centrée sur le pixel p et les pixels de la seconde *frame* dans la même fenêtre d'étude centrée sur le pixel p et translatée du vecteur de vélocité. Le mouvement décrit est donc celui de groupes de pixels. Néanmoins, une bonne précision nécessite une étude très locale qui est mise en défaut lors de mouvement rapide. Pour obtenir à la fois une bonne précision et une bonne robustesse de l'estimation, l'étude se réalise de façon pyramidale. Les *frames* sont étudiées à différentes échelles. En commençant à une petite échelle, la robustesse au mouvement rapide est assurée. Ensuite, par passage à des échelles de plus en plus grandes, la précision est augmentée. Les détails des calculs de cette méthode sont explicités en annexes D.1.

[Bruno and Pellerin, 2002] réalise un filtrage spatio-temporel sur chaque canal de couleur des *frames* afin de rajouter des équations pour pouvoir résoudre le problème. Le problème est alors sur-contraint pour optimiser certaines caractéristiques de l'estimation. Les filtrages utilisés sont des filtrages de Gabor réalisant une estimation plus stable car complexe. Une optimisation par les moindres carrés donne finalement l'estimation voulue. La figure 4.17 montre les résultats obtenus. Les détails de cette méthode sont disponibles en annexe D.2.

4.6.2 Un complément au descripteur

Le mouvement est une information bien descriptive de la classe des personnes. Par conséquent, rajouter des informations de mouvement au vecteur de caractéristiques défini en section 3.3.1 peut permettre d'améliorer le descripteur.

Codage de l'information

Les pixels correspondant à un même élément de la séquence ont un mouvement très proche. Donc, la variation de flot optique entre les pixels de la silhouette est assez faible. Par contre, le mouvement de la personne est souvent différent de celui de l'arrière plan. La variation de flot optique entre un pixel du premier plan et un pixel de l'arrière plan est souvent plus importante. L'information pertinente est donc la variation de flot optique entre deux pixels spatialement voisins. Si celui-ci est grand, les chances de se trouver sur le contour de la silhouette sont plus élevées. L'information est assez similaire au gradient qui est la variation d'intensité entre deux pixels spatialement voisins. L'information de mouvement peut donc être codée par des histogrammes de variations de flots optiques.

L'évaluation de la variation du flot optique n'est pas simple. Comme l'arrière plan peut aussi se déplacer, une compensation permet de normaliser le mouvement et donc de le rendre puis descriptif (la variation entre les exemples positifs est plus restreinte). Dans [Dalal et al., 2006], Dalal définit un certain nombre de méthodes de codage pour obtenir cette variation.
Avant de les définir, fixons certaines notations. Soit $\mathcal{I}^x$ et $\mathcal{I}^y$ les composantes selon les axes x et y du flot optique. Les dérivées partielles sont alors notées : $\mathcal{I}^x_x = \frac{d}{dx}\mathcal{I}^x$, $\mathcal{I}^x_y = \frac{d}{dy}\mathcal{I}^x$, $\mathcal{I}^y_x = \frac{d}{dx}\mathcal{I}^y$ et $\mathcal{I}^y_y = \frac{d}{dy}\mathcal{I}^y$.

Une première étude utilise uniquement le mouvement des contours. Les images $\mathcal{I}^x$ et $\mathcal{I}^y$ sont prises indépendamment, leurs gradients sont calculés (magnitude et orientations) ce qui permet de réalisé deux HOGs. Le mouvement des contours est alors décrit. Notons que l'utilisation de deux HOGs indépendamment (selon x et y) est plus descriptif qu'une association des deux. C'est le codage MBH (*Motion Boundary Histograms*).
Étant donné que l'arrière plan se déplace en même temps que les autres plans, il est intéressant d'étudier le mouvement relatif entre les différents éléments d'une scène. C'est ce qu'on appelle le codage IMH (*Internal Motion Histograms*).
Pour cela, le décompte pour chaque intervalle d'orientation est réalisé à partir des pairs $(\mathcal{I}^x_x, \mathcal{I}^y_x)$ et $(\mathcal{I}^x_y, \mathcal{I}^y_y)$ et la magnitude est dépendante de différences spatiales pas forcement voisines. De ces valeurs résultent les histogrammes orientés basés sur les directions de flots relatifs :

- **IMHdiff** utilise simplement $(\mathcal{I}^x_y, \mathcal{I}^y_y)$ avec une taille de filtre pouvant varier.
- **IMHcd** prend un bloc de 3×3 cellules et réalise 8 histogrammes prenant pour différences spatiales celle entre un pixel de la cellule centrale et le pixel correspondant dans chacune des cellules voisines.
- **IMHmd** prend un bloc de 3×3 cellules et réalise 9 histogrammes avec la différence entre le pixel de chacune des cellules et la moyenne des pixels correspondants pour les 9 cellules.
- **IMHwd** est semblable au IMHcd en utilisant des ondelettes de Haar.

Le mouvement est caractérisé selon plusieurs directions pour obtenir une bonne représentation. Les histogrammes de ces variations peuvent alors être ajoutés au descripteur pour une meilleure précision. Les mouvements reconnus par chaque codage sont visibles sur les moyennes obtenues par ces codages sur un ensemble de 100 couples de *frames* (figure 4.18). On vérifie donc que chaque histogramme caractérise le mouvement d'une ou plusieurs parties du corps d'une personne. Ils ne sont donc pas totalement redondants.

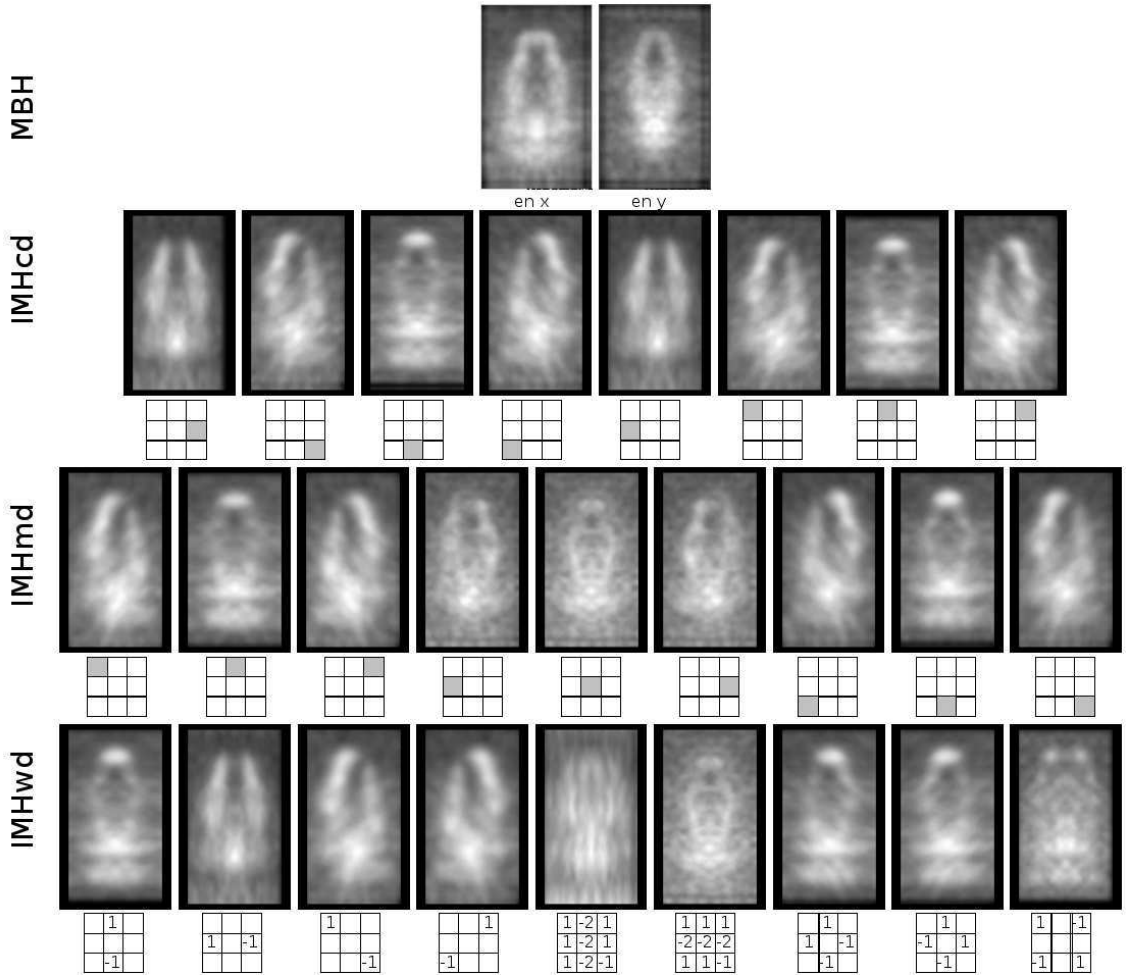

FIG. 4.18 – Moyennes des variations obtenues selon les différents codage. Le codage IMHcd réalise la différence spatiale entre la cellule centrale et celle grisée sur le schéma en dessous de l'image. Le codage IMHcd réalise la différence spatiale entre la moyenne des cellules et la cellule grisée sur le schéma en dessous de l'image. Enfin le codage IMHwd utilise les ondelettes de Haar selon le filtre représenté sous l'image. Pour chaque codage, les différents histogrammes prennent en compte la caractérisation de différentes parties du corps de la personne. Ils sont donc peu redondants et complémentaires pour réaliser une caractérisation complète du mouvement.

Résultats

Les codages cités dans la section précédente permettent de quantifier le mouvement et d'obtenir une information descriptive de la classe recherchée. Ces ajouts améliorent-ils

le descripteur ? La base de données servant à l'apprentissage est maintenant constituée de 77 couples de *frames* contenant des personnes en mouvement (exemples positifs) et de 60 couples de *frames* ne contenant pas de personnes. La base de données étant assez restreinte, la classification est donc moins efficace. Le but est néanmoins seulement d'évaluer s'il y a amélioration du processus. Les tests sont réalisés sur un ensemble de 10 couples de *frames* (fenêtre de détection) successives contenant des personnes et réalisant des mouvements variés (surtout au niveau des jambes : vitesse de balancement et orientation) comme sur la figure 4.16. Le résultat étudié est alors la moyenne du critères C_q (comme en section section 3.4.1) sur l'ensemble de ces tests.
Pour réaliser les descripteurs, sont utilisées soit les informations statiques (celles déterminées en section 3.3.1) soit les informations dynamiques liées au mouvement dans l'image (vu en section 4.6.2) soit la concaténation des deux. Les résultats obtenus sont présentés dans le tableau 4.19.

Codage	MBH	IMHdiff	IMHcd	IMHmd	IMHwd
Statique	2,23				
Dynamique	1,77	1,86	1,80	1,86	1,90
Statique + Dynamique	2,06	2,06	1,82	1,87	1,92

FIG. 4.19 – Valeurs du critère de quantification C_q obtenues pour différents codages et un descripteur réalisé à partir des informations statiques, dynamiques ou des deux. Ces résultats montrent que l'information dynamique est moins efficace que l'information statique et que l'association des deux informations est moins efficace que l'utilisation de l'information statique seule.

Ces résultats montrent que l'ajout, ou l'utilisation seule, des informations dynamiques altère la reconnaissance des segments vraisemblables. Cela signifie que les informations récoltées ne sont, à une petite échelle, pas assez descriptives de la classe recherchée.
La périodicité du mouvement des jambes est la partie la plus descriptive. Mais la périodicité s'observe sur un nombre conséquent de *frames* et non sur un couple comme dans notre cas. La partie haute du corps a un mouvement peu caractéristique et donc sans grand intérêt. L'information de mouvement est peu pertinente dans notre étude.
De plus, un descripteur relatif à des vecteurs de dimension plus élevé nécessite une base de données plus importante pour être efficace. Les résultats du tableau 4.19 montrent que les codages BMH ou IMHdiff qui se basent sur deux vecteurs de caractéristiques au lieu de 8 ou 9 pour les autres donnent les meilleures performances.

4.6.3 Un guide à la segmentation

La section précédente a montré que le mouvement n'était pas assez descriptif pour la **pré-segmentation**. Cette section va maintenant tester l'influence de l'insertion des informations de mouvement dans la segmentation présentée dans la section 4.
La **segmentation** est particulièrement difficile lorsque la concentration de segments de contour est élevée. Des détails sur des éléments comme des motifs sur un vêtement ou des fenêtres d'une maison, peuvent provoquer ce phénomène. Ces contours correspondent à des séparations entre objets d'un même plan. Ils ne peuvent appartenir à la silhouette recherchée.

L'ensemble des parties d'une personne se déplace en même temps. Un contour de transition entre deux objets réalisant un mouvement différent a donc plus de chance de correspondre à la silhouette de la personne.

Réalisation

Pour chaque pixel de contour p, un indice d'unicité du mouvement avec son entourage $i_{um}(p)$ est déterminé. Cet indice est la somme des variations de flot optique entre le pixel p et les pixels dans un voisinage 4-connexité V_4 de p :

$$i_{um}(p) = \sum_{v \in V_4(p)} (\mathcal{I}^x(p) - \mathcal{I}^x(v))^2 + (\mathcal{I}^y(p) - \mathcal{I}^y(v))^2 \tag{4.18}$$

Un indice correspondant à la moyenne des indices attribués aux pixels de chaque segment lui est affecté. Pour un segment seg de taille N pixels, cette indice est :

$$i_{um}(seg) = \frac{1}{N} \sum_{p \in seg} i_{um}(p) \tag{4.19}$$

Le passage par les segments d'indice élevé doit être favorisé. Comme vu en section 4.3.2, ceci se réalise au niveau de la pondération des arcs du graphe orienté. Cette pondération est fonction de l'affinité du passage entre deux segments. Notons ω^s la pondération statique calculée dans l'équation 4.3. On utilise le même graphe que dans le cas statique en réactualisant les poids ω^s par ω^d. Pour le passage d'un segment seg_i à un segment seg_j, la pondération dynamique ω^d est :

$$\omega^d_{ij} = \frac{\omega^s_{ij}}{\sqrt{i_{um}(seg_j)}} \tag{4.20}$$

Résultats

Un découpage manuel des silhouettes sur un ensemble de 10 couples de *frames* successives est réalisé afin d'obtenir les évaluations présentées en section 4.5.1. Le processus est réalisé sans insertion des informations de mouvement, puis avec. Les valeurs obtenues pour les différents critères sont présentées dans le tableau de la figure 4.20. Les trois critères montrent que l'insertion des informations de mouvements améliore la segmentation. Beaucoup de segments non pertinents ont été défavorisés et le suivi du contour est donc plus fidèle et moins dispersé.

Critères	Avec mouvement	Sans mouvement
F-measure	0,7956	0,7853
Martin	0,0743	0,0741
Yasnoff	0,9150	0,9342

FIG. 4.20 – Évaluation de la segmentation obtenue sans et avec ajout de l'information de mouvement. Les trois mesures montrent que l'ajout de l'information de mouvement améliore la segmentation.

4.6.4 Conclusions

Le mouvement doit permettre de réduire le temps de calcul et d'affiner la détection et la segmentation. Nous avons surtout étudié le second point. La réduction du temps de calcul peut être réalisée par un suivi de la silhouette ou une estimation par un filtrage de Kalman. Il s'agit d'un problème différent et non d'une adaptation de notre méthode.
Nous avons montré que, à une échelle locale (spatialement et temporellement), le mouvement d'une personne n'est pas caractéristique. L'ajout de cette information au niveau du descripteur ne permet donc pas d'améliorer la pré-segmentation.
Le mouvement permet d'isoler des régions intérêts. Ce qui est utile pour la détection (section 2.1). Nous utilisons ce même principe pour détecter les segments de contours séparant deux régions de mouvements différents. En favorisant ces segments dans le graphe, la segmentation est améliorée.

4.7 Conclusion

Dans cette section, nous avons présenté une méthode originale de segmentation de personnes dans les images fixes. Elle est associée à une méthode de détection efficace. Elle utilise la pré-segmentation de la section 3.
La silhouette de la personne est reconstituée à partir des segments de contour les plus vraisemblables. Pour cela, un algorithme de recherche de plus court chemin est appliqué à un graphe représentant les segments de contour de l'image. L'*a priori* sur la classe donné par la pré-segmentation est incorporé au processus par l'intermédiaire des pondérations des arcs.
Une segmentation correcte est obtenue dans la plupart des cas. Néanmoins, certaines images donnent une mauvaise segmentation. Nous avons montré que réitérer le processus permettait d'améliorer grandement les segmentations dans ces cas complexes.
Nous avons ensuite élargi notre étude au cas des séquences vidéos. L'information de mouvement n'est localement pas caractéristique et n'améliore pas la pré-segmentation. En revanche, un segment séparant deux régions dont les mouvements sont différents a plus de chances d'être un contour de la silhouette. La segmentation est améliorée en utilisant cette information.

De nombreuses perspectives d'améliorations de notre méthode existent :

- La pré-segmentation traite correctement le cas des occultations. Mais la silhouette obtenue par notre méthode de segmentation ne peut être que d'un seul tenant. Dans ces cas, plusieurs cycles pourraient être recherchés.

- D'autres codages de l'information de mouvement peuvent être plus descriptifs.

- La segmentation obtenue donne une silhouette plus ou moins pertinente. Cette information peut améliorer le processus de détection personnes en confirmant la détection et donc en réduisant le taux de fausse détection.

- La modélisation du contour de la silhouette en segments est un peu abrupte. Prendre en considération des objets permettant l'arrondi pourrait être intéressant.

La dilatation du contour obtenu peut aussi donner un *trimap* (figure 5.1(b)) qui peut permettre en post-traitement une segmentation précise sans *a priori*. Un photo-montage réaliste peut enfin être obtenu en remplaçant cette segmentation de post-traitement par un matting.

- Une intégration d'interaction de la part de l'utilisateur permettrait un plus grand controle de la segmentation.

S'agissant de ces deux derniers points, la méthode de la coupe de graphe [Boykov and Jolly, 2001] permet une segmentation au niveau du pixel intégrant une interaction pertinente avec l'utilisateur. Nous avons alors réalisé une nouvelle méthode de segmentation de personnes utilisant ce procédé. Nous la présentons dans le chapitre suivant.

Chapitre 5

Segmentation automatique par coupe de graphe et gabarits

> Le commencement de toutes les sciences, c'est l'étonnement de ce que les choses sont ce qu'elles sont.
>
> ARISTOTE

Sommaire

Pour obtenir une segmentation précise ou bien pour que l'utilisateur ait un contrôle plus important sur le résultat, il est intéressant que l'utilisateur puisse avoir une interaction avec la méthode qu'il utilise. Si cette interaction est préjudiciable dans certaines applications temps réel comme la vidéo-surveillance ou l'assistance de conduite, où l'on comprend aisément que l'utilisateur n'a pas le temps d'influencer le calcul, elle est fortement recommandée pour tous les travaux de post-traitements. Cependant l'interaction doit rester la plus mince possible pour minimiser le travail à effectuer par l'utilisateur. De plus, l'interaction doit bien s'intégrer dans le principe de la méthode pour que l'information donnée par l'utilisateur soit la plus profitable au temps de calcul et à la précision du résultat.

En opposition aux chapitres 3 et 4, nous avons développé une méthode favorisant l'interaction avec l'utilisateur et la précision de la segmentation aux dépens de la continuité avec la méthode de détection. Dans la littérature, la méthode de segmentation interactive de la coupe de graphe proposée par Boykov [Boykov and Jolly, 2001] est très souvent reprise pour son efficacité et sa simplicité. Nous proposons dans ce chapitre une méthode originale de segmentation de personnes basée sur ce principe. Nous introduisons des gabarits non binaires, représentant la probabilité de chaque pixel d'appartenir à une silhouette humaine, pour représenter dans le graphe la connaissance que nous possédons sur la classe recherchée.
Nous présentons tout d'abord le principe de la coupe de graphe qui permet la segmentation interactive dans la section 5.1. Puis notre méthode est expliquée dans la suite. Un gabarit unique, valable pour l'ensemble des silhouettes humaines étudiées, est dans un premier temps étudié dans la section 5.2. Pour avoir un gabarit qui s'adapte mieux à la posture de la personne étudiée, un gabarit par parties, où les quatre membres et la tête sont considérés séparément, est mis en place dans la section 5.3. Ces deux principes sont comparés et évalués dans la section 5.4. L'interaction de l'utilisateur peut remplacer la phase de détection. L'étude n'est alors plus restreinte aux fenêtres dont la taille est fixée par la méthode de détection. Les modes d'interactions adéquats et l'adaptation de notre méthode aux fenêtres de tailles variées sont présentés en section 5.6. Une conclusion est finalement tirée en section 5.7.

5.1 La coupe de graphe : une segmentation interactive

La segmentation d'images peut se classer en trois catégories :

- La séparation de l'image en régions partageant une même caractéristique (une couleur proche par exemple). Ce découpage se fait automatiquement et seul le niveau de découpage (le nombre de régions à séparer dans l'image par exemple) est réglable par l'utilisateur.
- Le découpage d'un objet du choix de l'utilisateur. L'algorithme n'a aucune connaissance sur la nature de l'objet en question. Donc, l'utilisateur doit donner un certain nombre d'informations pour que le programme sache quel est l'élément de l'image qu'il veut segmenter. Le nombre de segmentation possible étant très important, l'interaction de l'utilisateur doit alors être importante et précise.

- Le découpage des objets d'une classe bien particulière. Un apprentissage sur la classe étudiée permet de détecter les objets pertinents dans une image et donne donc suffisamment d'informations pour se passer de l'interaction avec l'utilisateur.

Notre étude sur la segmentation de personne se situe dans la troisième catégorie. Aucune interaction avec l'utilisateur n'est donc nécessaire. Néanmoins, que ce soit pour donner des résultats plus précis ou bien pour réduire le temps de calcul, une légère interaction de l'utilisateur peut permettre d'obtenir de meilleures performances.

5.1.1 Méthode d'intéraction

Les modes d'interaction de l'utilisateur peuvent être multiples. Plusieurs procédés sont utilisées dans la littérature.
L'utilisateur peut tout d'abord réaliser un trimap (figure 5.1(b)). Il s'agit d'une carte de l'image à segmenter où chaque pixel est assigné à une des trois régions : pixels appartenant avec certitude au premier plan (l'élément à segmenter), pixels appartenant avec certitude à l'arrière plan et pixels dont l'appartenance au premier ou à l'arrière plan est incertaine. La méthode doit alors assigner les pixels de la troisième région. Cette région est souvent la plus fine et se situe le long des contours de l'objet. C'est une contrainte forte demandée par l'utilisateur mais assez fastidieuse à réaliser. Ce principe permet une grande précision (et la possibilité de réaliser un *matting* plutôt qu'une segmentation). Cependant, les connaissances sur l'objet à segmenter sont inutiles puisque la forme globale est déjà donnée par l'utilisateur. Ce procédé ne nous intéresse donc pas dans cette étude. Il est néanmoins beaucoup utilisé dans la segmentation sans *a priori* [Ruzon and Tomasi, 2000][Hillman et al., 2001][Chuang et al., 2001][Sun et al., 2004] [Huart, 2007].

L'utilisateur peut aussi situer approximativement l'objet à segmenter par exemple en l'encadrant dans une fenêtre [Rother et al., 2004] (figure 5.1(c)). La connaissance de la classe recherchée est alors nécessaire pour bien savoir, dans cette fenêtre, quels sont les éléments appartenant à l'objet souhaité. L'interaction est ici très rapide et facile mais donne de précieuses informations. Lin [Lin et al., 2007a] réalise une carte de probabilité d'appartenance au premier plan initialisée par une gaussienne centrée dans la fenêtre. Ensuite, un algorithme EM paramétré par un estimateur de densité de Kernel [Scott, 1992] affine itérativement la carte jusqu'à obtenir la segmentation. La fenêtre sert alors de point de départ pour le calcul des premiers paramètres. Bray [Bray et al., 2006] recherche par l'algorithme EM le squelette correspondant à la personne présente dans la fenêtre. Celui-ci initialise un champ aléatoire de Markov. Une coupe de graphe de ce champ donne la segmentation. Borenstein [Borenstein and Malik, 2006] recherche des sous-parties de la classe recherchée bien placées dans la fenêtre. Puis, il découpe l'image en régions cohérentes et réalise une pyramide à partir de ces découpages et des informations précédemment calculées. Une remontée de la pyramide engendre la segmentation. L'interaction sert ici à définir une vraisemblance de localisation des sous-parties.
Cependant, la détection des personnes est de nos jours assez efficace. Il est alors assez facile d'obtenir des fenêtres de détection aussi précises que celles que donnerait l'utilisateur.

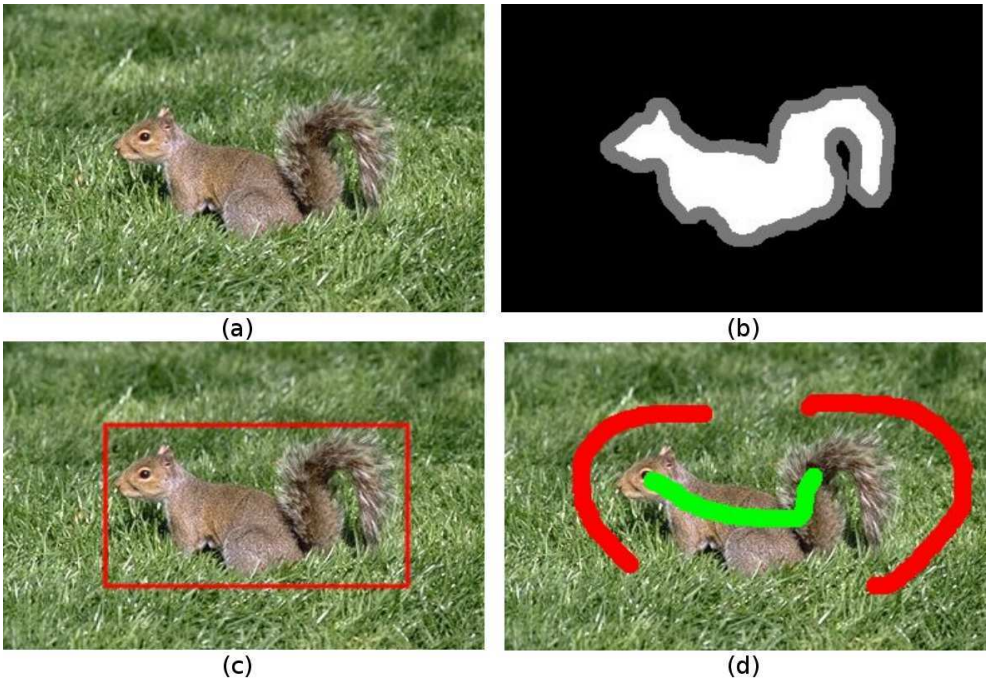

FIG. 5.1 – Différents modes d'interaction de l'utilisateur. (a) image initiale où l'écureuil doit être segmenté. (b) trimap où les pixels appartenant certainement au premier plan sont en blancs, les pixels appartenant certainement à l'arrière plan sont en noirs et les pixels incertains sont en gris. Ce trimap est compliqué à obtenir. (c) l'utilisateur a juste encadré l'objet d'intérêt pour le localiser. (d) par des traits (vert pour le premier plan et rouge pour l'arrière plan), l'utilisateur a assigné certains pixels à la région à laquelle ils appartiennent.

Enfin, l'utilisateur peut juste tracer quelques traits sur l'image (figure 5.1(d)) pour fixer des pixels appartenant au premier et à l'arrière plan [Li et al., 2004][Guan et al., 2006]. Encore une fois le travail à réaliser par l'utilisateur est très restreint et rapide. Cette assignation est parfaitement adaptée aux résolutions par coupe de graphe [Boykov and Jolly, 2001][Kumar et al., 2005]. Il est en effet facile de forcer des pixels à appartenir à une des deux régions en jouant sur les poids des arcs du graphe (voir section 5.1.2). De plus, il est très facile de remodifier une première segmentation en marquant des pixels des parties mal attribuées. La coupe de graphe est donc une méthode très flexible qui semble la plus à même de convenir à notre étude.

5.1.2 Coupe de graphe

Soit un graphe $\mathcal{G}$ composé de noeuds reliés par des arcs. Le noeud de départ s est appelé source et le noeud d'arrivé p est appelé puits. Au passage par un arc d'un noeud à un autre est associé un poids. Une coupe de $\mathcal{G}$ est représentée par un plan séparant les noeuds en deux sous-parties. L'une doit contenir la source et l'autre le puits. L'énergie associé à cette coupe est la somme des poids des arcs croisés par la coupe. Des algorithmes [Boykov and Kolmogorov, 2004] permettent de déterminer la coupe minimale, c'est-à-dire la coupe qui donne l'énergie la plus faible possible (figure 5.2).

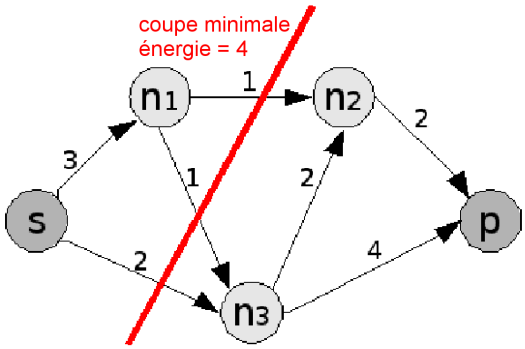

FIG. 5.2 – Graphe orienté représentant le trajet d'un noeud source s à un noeud puits p. Les valeurs sur les arcs sont les poids associés. La coupe minimale (en rouge) est la coupe qui sépare les noeuds en une partie contenant s et une partie contenant p et dont la somme des poids des arcs croisés est la plus faible.

Boykov [Boykov and Jolly, 2001] propose une méthode pour adapter la coupe de graphe au traitement d'image et en particulier à la segmentation d'objet. Il construit un graphe $\mathcal{G}$ où chaque pixel de l'image étudiée est représenté par un noeud. Un pixel est relié à chacun de ses voisins par un arc. Il rajoute alors deux noeuds qu'il définit comme la source et le puits (figure 5.3). La source représente le premier plan et le puits représente l'arrière plan. Ainsi, lors de la coupe du graphe, les pixels de l'image sont répartis en deux sous-parties. L'une contient la source et appartient donc au premier plan et l'autre contient le puits et appartient donc à l'arrière plan. Pour permettre à chacun des pixels d'appartenir à l'une ou l'autre des deux parties, chaque noeud est relié par un arc aux deux noeuds s et p.

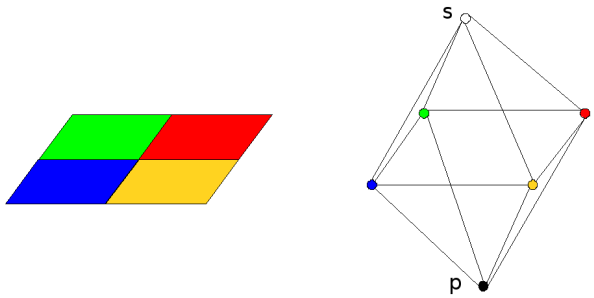

FIG. 5.3 – Pour une image (à gauche), chaque pixel est représenté par un noeud et est relié par un arc à ses voisins, à la source (représentant le premier plan) et au puits (représentant l'arrière plan).

La coupe minimale passe par les arcs dont le poids est le plus faible. Dans notre cas, la coupe représente une délimitation entre deux régions. Par conséquent, si une transition entre deux pixels voisins est abrupte, elle a de grande chance d'appartenir à un contour et doit donc être représentée par un poids faible. Soit i un pixel et I_i l'intensité de l'image au pixel i. L'arc entre les noeuds correspondant aux pixels i et j de l'image est alors associé à un poids :

$$\omega_{i,j} = \frac{1}{d_{i,j}} . e^{-\frac{(I_i - I_j)^2}{2\sigma^2}} \tag{5.1}$$

où $d_{i,j}$ représente la distance spatiale entre i et j et σ représente la tolérance à la discontinuité associée à la résolution de l'image.
Les poids relatifs aux arcs reliant des noeuds pixel au noeuds s et p permettent l'interaction avec l'utilisateur. En effet, si un pixel est manuellement assigné au premier plan par l'utilisateur, le poids reliant ce pixel à la source est mit à $+\infty$ (la coupe minimale ne le croisera donc pas) et le poids reliant ce pixel au puits est mis à 0 (la coupe minimale le croisera donc). Ainsi, dans la segmentation, le pixel assigné appartient forcément au premier plan (figure 5.4). L'intégration des informations de l'utilisateur est ainsi très facile et efficace.
Ensuite, les poids relatifs aux liaisons entre les noeuds des pixels non assignés et les noeuds s et p correspondent à la vraisemblance de ces pixels à appartenir au premier ou à l'arrière plan. À partir des pixels assignés au premier plan, l'histogramme des valeurs d'intensité H_f est calculé. De même l'histogramme H_b est calculé à partir des pixels assignés à l'arrière plan. Le poids entre le noeud d'un pixel i et la source est alors défini par :

$$\omega_{i,s} = -ln(H_f) \tag{5.2}$$

Le poids entre le noeud d'un pixel i et le puits est de même défini par :

$$\omega_{i,p} = -ln(H_b) \tag{5.3}$$

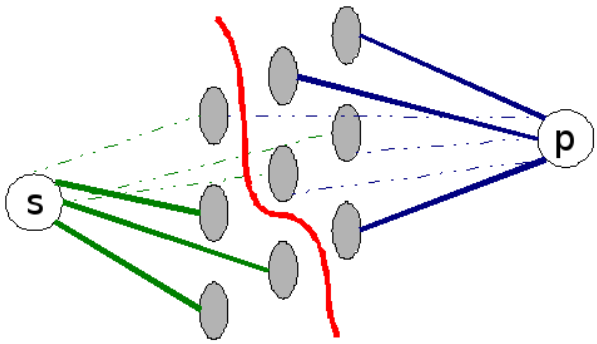

FIG. 5.4 – Pour un pixel assigné au premier plan par l'utilisateur, seul le passage par le lien avec le puits (traits bleus épais) est possible. Pour un pixel assigné à l'arrière plan par l'utilisateur, seul le passage par le lien avec la source (traits verts épais) est possible. Pour les autres pixels, les liaisons avec la source et le puits sont possibles (traits en pointillés). La coupe obtenue (en rouge) respecte bien les contraintes de l'utilisateur.

5.2 Segmentation par un gabarit unique

La méthode de Boykov [Boykov and Jolly, 2001] est très efficace mais a le désavantage de ne pas s'adapter à une classe en particulier. La totalité de la connaissance sur l'objet à segmenter est donnée par les caractéristiques des pixels marqués par l'utilisateur. Nous voulons ici développer une méthode qui utilise les nombreux avantages de la coupe de graphe tout en étant spécialement adaptée à une classe en particulier. Comme précédemment nous nous intéressons à la classe des personnes.

L'objectif est d'obtenir une segmentation où l'interaction n'est qu'une possibilité donnée à l'utilisateur de corriger les erreurs réalisées par une méthode de segmentation automatique. Nous proposons donc une méthode où l'interaction est toujours possible et bien intégrée au principe de calcul, mais où l'apprentissage de la classe étudiée permet d'obtenir un premier résultat cohérent sans aide extérieure.

Nous voulons intégrer notre connaissance de la forme de la personne dans le graphe. Plusieurs méthodes ont déjà été proposées pour y parvenir. Le processus original de la coupe de graphe a été modifié par un ajout d'un troisième terme (les deux premiers étant le terme de donnée correspondant aux arêtes de liaison et le terme de lissage correspondant aux arêtes de voisinage) à sa fonction d'énergie afin de prendre en compte la forme par des *Level Set* [Freedman and Zhang, 2005]. Les termes de donnée ou de lissage (ou les deux) de la fontion énergie ont été modifiés [Wang and Zhang, 2010] [Slabaugh and Unal, 2005] [Das et al., 2009]. Malcolm propose une solution intéressante mais coûteuse en temps [Malcolm et al., 2007] où une pré-image, obtenue à partir d'un apprentissage avec une analyse en composantes principales (ACP) est itérativement remise à jour.
L'originalité de notre méthode est d'introduire directement l'information de la forme dans le terme de donnée (les pondérations des arrêtes de liaisons). Pour cela nous utilisons des gabarits non-binaires. Un gabarit est un modèle utilisé pour caractériser la forme des éléments d'une classe. Dans la classe des personnes debout à laquelle on s'intéresse, les postures possibles sont assez variées. Un gabarit représente par un masque binaire la silhouette pour une de ces postures types. Un catalogue de gabarits est réalisé contenant toutes les postures que la personne peut prendre. Ces gabarits sont ensuite comparés un par un aux caractéristiques de l'image (souvent les contours [Zhao and Davis, 2005]). Si la comparaison est positive, une personne est détectée et la silhouette correspondant à sa posture est évaluée. La segmentation est finalement obtenue en adaptant légèrement cette silhouette aux contours de l'image [Rodriguez and Shah, 2007]. Pour diminuer le temps de calcul, [Gavrila and Giebel, 2002] réalise une répartition hiérarchique des gabarits. Ainsi, les comparaisons ne sont pas effectuées avec tous les gabarits (le choix des gabarits dépend du résultat des comparaisons précédentes). Enfin, [Lin et al., 2007a] décompose le corps en trois parties (le torse, le bassin et les jambes) et cherche le gabarit correspondant à chaque sous-partie.
Ramanan propose un gabarit assez proche du notre [Ramanan, 2007]. Mais il l'utilise pour définir des modèles de luminance du premier et de l'arrière-plan, alors que nous l'utilisons pour intégrer directement dans le graphe l'infomation de forme qui est la plus caractéristique de la classe des personnes.

Comme précédemment, la segmentation se base sur une fenêtre de détection de taille fixée contenant une personne centrée. Le graphe est réalisé à partir d'une de ces fenêtres englobant une personne issue de la détection. Une source F et un puits B représentent le premier et l'arrière plan. Chaque pixel de la fenêtre forme un noeud dans le graphe. Chaque pixel est relié dans le graphe aux pixels voisins par des *arêtes de voisinage* et à F et B par des *arêtes de liaisons* comme vu dans la section 5.1.2. L'information sur les caractéristiques de l'image est alors introduite par les pondérations de ces arêtes.

5.2.1 Arêtes de voisinage

Les arêtes de voisinage représentent la possibilité que la transition entre deux pixels voisins soit un contour de la silhouette découpée par le graphe. Il est donc logique de les associer au gradient de l'image. [Boykov and Jolly, 2001] utilise la différence d'intensité pour pondérer ces arêtes :

$$\nabla_{p_1,p_2}^{Boykov} = |I_{p_1} - I_{p_2}| \tag{5.4}$$

où p_1 et p_2 sont les deux pixels de la transition et I_p l'intensité du pixel p.
Compte tenu de la petite taille des images que nous étudions, et des discontinuités rapides qu'on y trouve, il est, dans notre cas, préférable d'utiliser le gradient calculé selon les deux directions. Soit $I_{x,y}$ l'intensité du pixel p de coordonnées x, y,le gradient ∇_p est défini par :

$$\nabla_p = |I_{x,y} - I_{x+1,y}| + |I_{x,y} - I_{x,y+1}| \tag{5.5}$$

La pondération associée à l'arête reliant les pixels p à q est alors définie par :

$$\omega_{p,q} = \beta e^{-\frac{\nabla_p^2}{2\sigma^2}} \tag{5.6}$$

où β exprime l'importance des arêtes de voisinage et σ le filtrage opéré par l'exponentielle. En effet, l'évolution de l'exponentielle ne nous intéresse que sur l'intervalle où elle n'est ni trop lente ni trop rapide. La valeur de σ permet de juxtaposer les gradients à mettre en exergue dans le graphe avec cet intervalle.
Les valeurs faibles représentent les fortes probabilités de contour (figure 5.5). Les coupures des arêtes correspondant aux transitions de gradients élevés sont donc favorisées.

FIG. 5.5 – Valeurs de pondération des arêtes de voisinage. Elles correspondent bien aux contours de l'image.

5.2.2 Arêtes de liaisons

Les arêtes de liaison rattachent chacun des pixels à F et B. Pour un pixel, la coupe passe par une et une seule de ces deux arêtes, ce qui désigne à quelle région est assigné le pixel. La pondération de ces arêtes doit donc correspondre à la probabilité du pixel d'appartenir au premier et à l'arrière plan. [Boykov and Jolly, 2001] relie cette probabilité à la distribution en couleurs. Mais la grande variété de couleurs dans la classe des personnes et dans le fond rend cette caractéristique peu discriminante. La localisation spatiale est une information plus pertinente. On réalise alors un gabarit non binaire représentant la probabilité de chaque pixel d'appartenir à une silhouette humaine. Pour le réaliser, on utilise la base de données de silhouettes de personnes debout présentée en annexe A et réalisée pour la méthode de pré-segmentation de la section 3. Chaque pixel de ce gabarit est alors la proportion d'occurrence de ce pixel dans les 200 silhouettes humaines de la base de données. Ce gabarit est visualisé sur la figure 5.6. Les valeurs sont plus élevées au niveau de la tête et du torse que des jambes et des bras. En effet, la position de ces parties est beaucoup plus stable. Le balancement des jambes lors de la marche fait que leur position est bien moins déterminée.

FIG. 5.6 – Image moyenne d'une base de 200 silhouettes. La position de la tête et du torse est bien plus stable que celle des bras et surtout que celle des jambes

Soit t_p la valeur de ce gabarit pour le pixel p. Comme t_p est une probabilité, elle est comprise entre 0 et 1. Les pondérations attribuées aux arêtes de liaison sont alors définies par :

$$\omega_p^F = -\alpha.ln(t_p) \tag{5.7}$$

$$\omega_p^B = -\alpha.ln(1 - t_p) \tag{5.8}$$

où α exprime l'importance des arêtes de liaison.
α et β expriment l'importance respective des arêtes de liaison et de voisinage. Comme seul le rapport entre leurs pondérations est théoriquement utile, on pourrait ne garder que le rapport $\frac{\alpha}{\beta}$ associé aux arêtes de liaisons. Néanmoins, en pratique, le logiciel fourni par [Boykov and Jolly, 2001] et que nous utilisons demande des pondérations entières avec des valeurs pas trop grandes. Pour limiter les approximations, les deux paramètres α et β sont nécessaires.

5.3 Segmentation avec un gabarit par parties

Le gabarit unique présenté précédemment prend en compte toutes les postures mais défavorise celles d'occurrences les plus faibles (notamment lorsque les jambes sont écartées). Nous proposons alors d'utiliser un gabarit adapté à la posture rencontrée. Il serait possible de créer un gabarit relatif à toutes les postures qu'une personne debout peut prendre puis de réaliser une coupe de graphe avec chacun de ces gabarits. Mais le nombre de postures possibles étant très élevé, un temps de traitement très important serait nécessaire. La solution que nous proposons est de traiter indépendamment les différentes parties de la silhouette. Nous découpons l'image en cinq parties : tête, partie droite du torse, partie gauche du torse, jambe droite et jambe gauche. Pour chacune de ces parties, nous construisons un certains nombre de sous-gabarits représentant les divers postures possibles (figure 5.7).

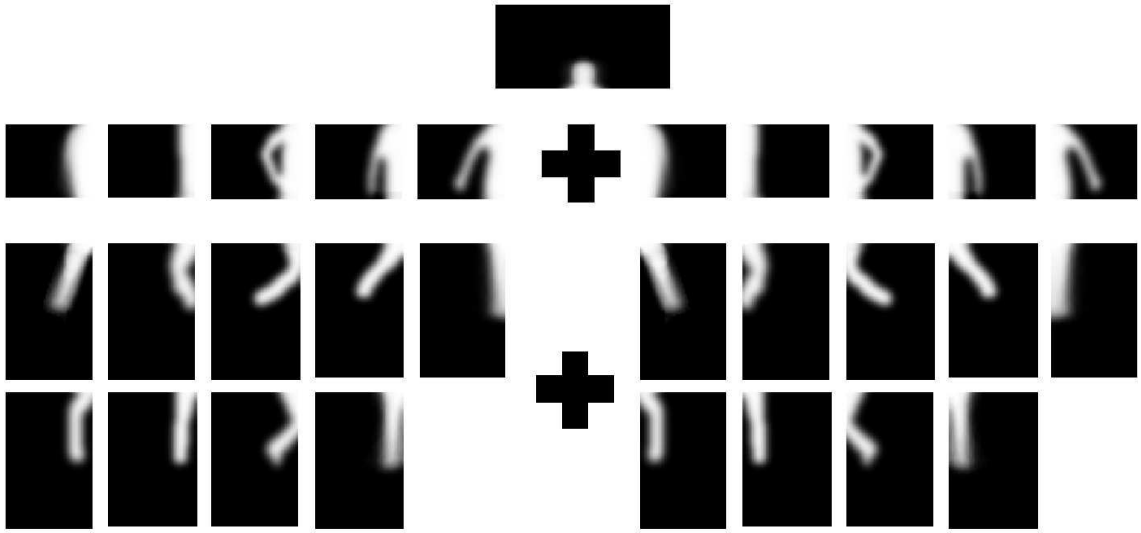

Fig. 5.7 – Ensemble des sous-gabarits relatifs aux cinq sous-parties de l'image.

Pour chaque partie, une coupe de graphe est réalisée à partir de chacun des sous-gabarits. L'énergie d'une coupe est la somme des pondérations des arêtes coupées. Ainsi, si cette énergie est faible, c'est que les arêtes coupées étaient faiblement pondérées et donc pertinentes. Il est donc préférable d'obtenir l'énergie la plus petite possible. Si une coupe avec un gabarit donné, donne une énergie faible, c'est donc que le gabarit s'adapte bien aux caractéristiques de l'image.
Le gabarit par parties est alors la concaténation des sous-gabarits ayant donné les énergies de coupes les plus faibles pour chaque partie (figure 5.8). Enfin, une coupe de graphe sur l'image entière à partir du gabarit par parties obtenu donne la segmentation finale.
Un nombre de sous-gabarits restreint est suffisant au traitement. En effet, comme les arêtes de voisinage adaptent la coupe au contour, les gabarits doivent seulement favoriser un état (bras décollé, jambe pliée,...) et non parfaitement correspondre à la silhouette de la personne. Le gabarit est seulement une allure de la silhouette. En outre, le modèle proposé permet de tester $1 \times 5 \times 5 \times 9 \times 9 = 2025$ gabarits différents tout en ne réalisant que 28 coupes de graphe sur des sous-images de tailles plus petites. Le temps de traitement est donc bien plus court.

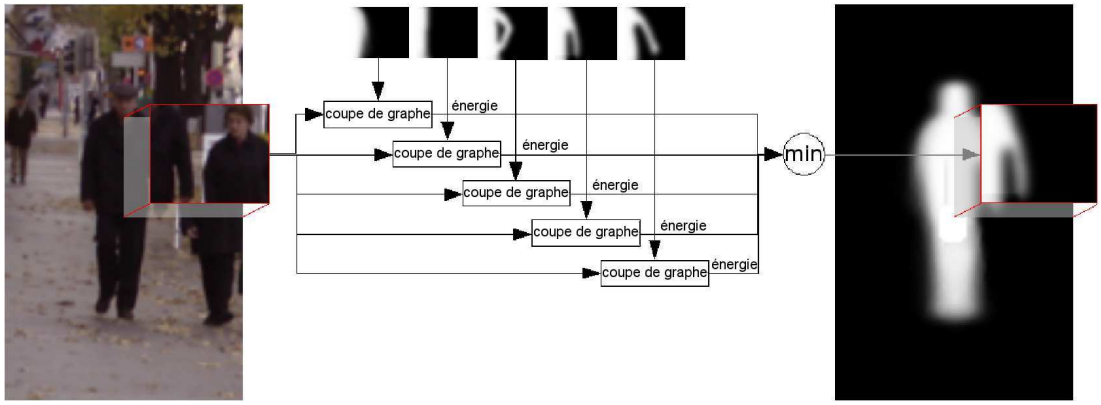

FIG. 5.8 – Pour la partie de l'image correspondant à la partie droite du torse, des coupes de graphe sont réalisées à partir de plusieurs sous-gabarits. Celui associé à la coupe donnant l'énergie minimal est alors ajouté au gabarit par parties.

5.4 Résultats

Pour évaluer la méthode, nous utilisons comme précédemment la $F_{measure}$ et les mesures de Martin et de Yasnoff [Philipp-Foliguet and Guigues, 2006] sur un ensemble de 400 images de la base de données de l'INRIA par comparaison avec une réalité terrain réalisée manuellement. Les temps de traitement sont observés à partir d'une implémentation C++ non optimisée sur un processeur Pentium D 3GHz. Pour faciliter la lecture des résultats, le signe (↓) signifie qu'une mesure est à minimiser (la mesure de Yasnoff) et le signe (↑) qu'elle est à maximiser (la $F_{measure}$).

5.4.1 Optimisation du processus

Pour optimiser le processus, nous devons déterminer les valeurs des paramètres α, β et σ qui produisent les meilleures performances. Des tests ont donc été réalisés avec différentes valeurs de ces paramètres. Les résultats donnés dans la figure 5.9 sont obtenus avec un gabarit unique mais ceux obtenus avec un gabarit par parties sont très semblables. Pour obtenir la meilleure segmentation, c'est-à-dire minimiser la mesure de Yasnoff et maximiser la $F_{measure}$, nous choisissons de garder pour la suite les valeurs : $\sigma = 10$, $\alpha = 12$ et $\beta = 150$.

5.4.2 Gabarit unique ou par parties ?

Puisque nous avons introduit deux procédés différents, il faut comparer leurs performances respectives. Tout d'abord le traitement avec un gabarit unique est logiquement plus rapide avec, pour le traitement d'une image 96×160 pixels, une moyenne de 12 ms contre une moyenne de 70ms avec le gabarit par parties. En gardant les valeurs des paramètres désignées précédemment, on obtient les résultats du tableau 5.1.

Le gabarit par parties donne de meilleures $F_{measure}$ et mesure de Yasnoff. Mais les résultats sont assez proches notamment avec la mesure de Yasnoff. En regardant les seg-

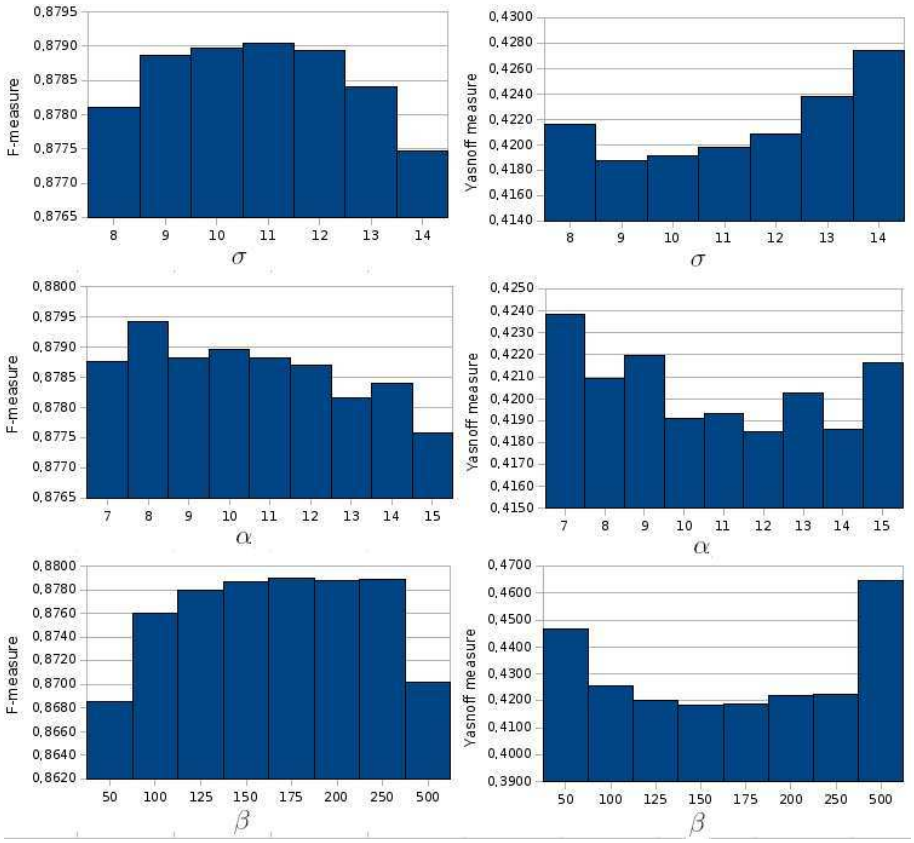

FIG. 5.9 – Résultats des tests réalisés pour optimiser les valeurs des paramètres σ, α et β. Les mesures $F_{measure}$ (↑) et de Yasnoff (↓) permettent d'évaluer la qualité de la segmentation. Les tests pour σ ont été réalisés avec $\alpha = 10$ et $\beta = 150$. Les tests pour α ont été réalisés avec $\sigma = 10$ et $\beta = 150$. Les tests pour β ont été réalisés avec $\sigma = 10$ et $\alpha = 12$. Pour optimiser la mesure de Yasnoff (↓) tout en prenant des hautes valeurs pour la $F_{measure}$, les valeurs $\sigma = 10$, $\alpha = 12$ et $\beta = 150$ sont utilisées pour la suite.

Mesure	Gabarit unique	Gabarit par parties
$F_{measure}$ (↑)	0,8787	0,8851
Yasnoff (↓)	0,4185	0,4162

TAB. 5.1 – Performances de segmentation sur un ensemble de 400 tests avec un gabarit unique et avec un gabarit par parties. Les meilleurs résultats sont obtenus avec le gabarit par parties.

mentations obtenues, on voit qu'un sous-gabarit inadapté provoque des segmentations aberrantes auxquelles la mesure de Yasnoff est très sensible (figure 5.10). Néanmoins, dans la grande majorité des cas, le gabarit par parties donne une segmentation mieux adaptée et plus précise (figure 5.11).

5.5 Ajout de l'information de luminance

Pour définir les pondérations des arêtes de liaisons, il faut déterminer la probabilité qu'un pixel appartienne au premier et à l'arrière plan. Dans la version usuelle du graph-

FIG. 5.10 – Un mauvais choix de gabarit par parties peut engendrer des segmentations aberrantes.

cut définie par Boykov [Boykov and Jolly, 2001], ces probabilités sont liées à l'information de luminance. Mais cela nécéssite une forte intéraction de l'utilisateur pour définir les modèles de référence du premier et de l'arrière plan. C'est pourquoi nous avons introduit nos gabarits. Mais notre méthode ne prend alors plus en compte l'uniformité de couleur (et donc de luminance) des éléments d'une région. Dans cette section, nous présentons un aménagement de notre méthode où une première itération utilise uniquement les gabarits comme précédemment et une seconde itération intègre la luminance pour affiner la segmentation.

5.5.1 Présentation de la méthode

Les deux itérations suivent le même format. Un graphe est réalisé où les noeuds sont les pixels de l'image. Les arêtes de voisinage sont pondérées par une fonction exponentielle de la variation d'intensité. Une coupe de graphe sur chaque sous-partie permet de reconstituer le gabarit par parties. Une coupe de graphe sur l'image entière à partir du gabarit par parties donne la segmentation. Les variations entre les deux itérations résident uniquement dans la pondération des arêtes de liaisons.
Pour la première itération seule la valeur t_p du gabarit pour le pixel p est prise en compte. Les pondérations des arêtes de liaisons sont définies par les équations 5.7 et 5.8.

Pour pouvoir utiliser la luminance comme Boykov [Boykov and Jolly, 2001], il faut connaître un certain nombre de pixels de chaque région. Mais la segmentation de la première itération donne une évaluation de la répartition des pixels du premier et de l'arrière plan. On définit alors les histogrammes H_F et H_B donnant la répartition des pixels d'une région selon leur valeur de luminance. Soit $a_p \in \{F, B\}$ l'assignation du pixel p au premier où à l'arrière plan selon la segmentation de la première itération et $I_p \in [0, 255]$ la luminance du pixel p, alors les histogrammes sont définis par :

$$H_F = \{\frac{card\{p|I_p = i, a_p = F\}}{card\{p|a_p = F\}}, i \in [0, 255]\} \tag{5.9}$$

$$H_B = \{\frac{card\{p|I_p = i, a_p = B\}}{card\{p|a_p = B\}}, i \in [0, 255]\} \tag{5.10}$$

Comme ni la couleur ni la luminance ne sont discriminantes de la classe des personnes, ces informations ne suffisent pas. Il faut alors leur associer les informations spatiales. Les pondérations des arêtes de liaisons dans la seconde itération sont définies par :

$$\omega_p^F = -\alpha(ln(t_p) + \gamma ln(H_F(I_p))) \tag{5.11}$$

$$\omega_p^B = -\alpha(ln(1 - t_p) + \gamma ln(H_B(I_p))) \tag{5.12}$$

où γ gère l'importance du gabarit face à la répartition en luminance.

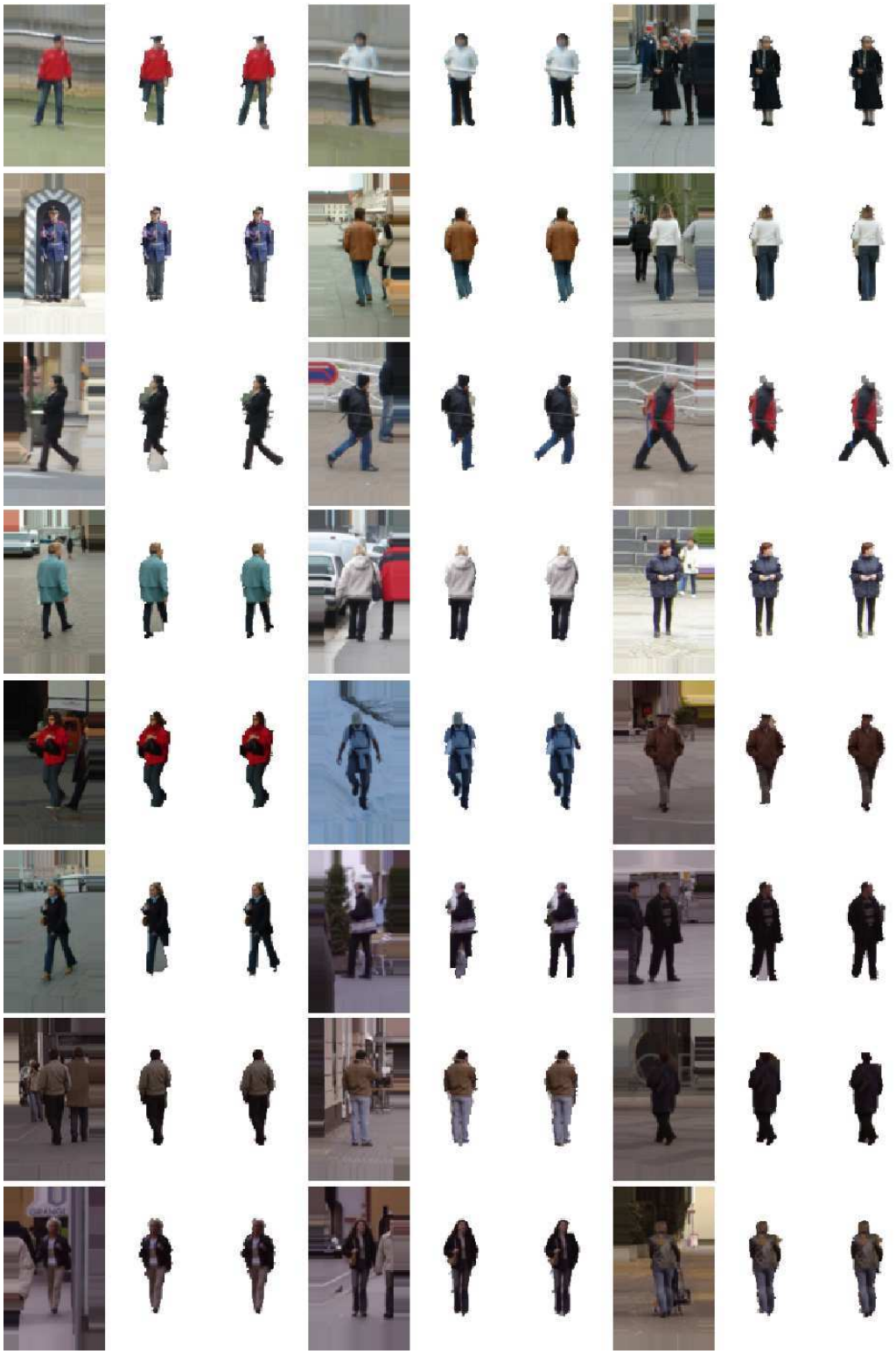

FIG. 5.11 – De gauche à droite : image initiale, segmentation obtenue avec un gabarit unique et avec un gabarit par parties. Dans la plupart des cas, la segmentation est plus précise avec le gabarit par parties. Ceci est bien visible lorsque les jambes sont écartées (exemples des lignes 3 et 6).

5.5.2 Résultats

Le paramètre γ est le seul nouveau paramètre. On teste les segmentations obtenues sur une base de 400 images tests pour différentes valeurs de ce paramètres. D'après les résultats visibles sur la figure 5.12, on choisit $\gamma = 0,7$ comme valeur optimale que l'on garde pour la suite.

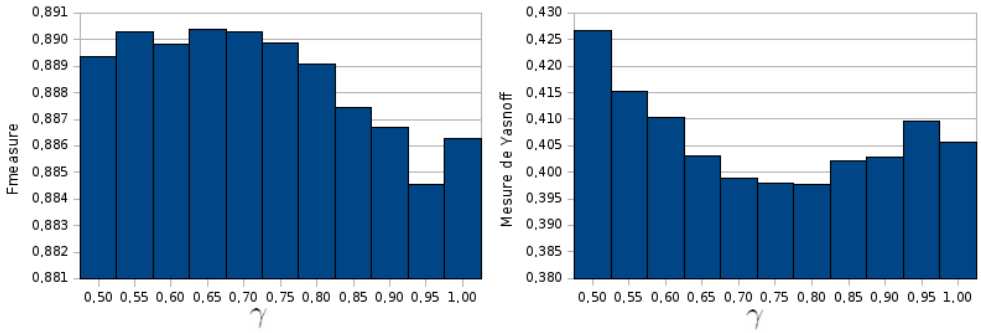

FIG. 5.12 – Évaluation de la qualité des segmentations en fonction du paramètre γ.

Comme nous le voyons dans le tableau 5.2, la seconde itération apporte une importante amélioration de la segmentation. Les exemples de la figure 5.13 confirment que la seconde itération permet bien d'affiner la segmentation notamment en retrouvant des membres perdus ou en suivant plus précisément les contours.

Nous prenons en considération la luminance comme le fait Boykov. Mais il serait possible d'utiliser aussi la couleur dans ses trois dimensions. Bugneau [Bugneau and Perez, 2007] et Kulkarni [Kulkarni and Nicolls, 2009] modélisent l'interaction de l'utilisateur par des modèles de mélanges de gaussiennes (GMM). Lattari [Lattari et al., 2010] définit les couleurs les plus souvent associées au premier ou à l'arrière plan par *K-means*. Néanmoins, la couleur ne devant nous permettre que d'améliorer le résultat de la segmentation, prendre une information à plusieurs dimensions n'est pas pertinent. De plus, l'utilisation de la luminance comme vu précédemment fait déjà doubler le temps de traitement de notre méthode.

Mesure	1 itération	2 itérations
$F_{measure}$ (↑)	0,8851	0,8911
Yasnoff (↓)	0,4162	0,3965

TAB. 5.2 – Performances de segmentation avec une seule itération puis avec une seconde intégrant la luminance. La seconde itération améliore la segmentation selon les deux mesures.

5.6 Fenêtres de tailles quelconques

L'association de la coupe de graphe avec les gabarits nous permet de réaliser une segmentation efficace des personnes dans des fenêtres de détection. Nous avons pour

FIG. 5.13 – Exemples de segmentation avec une itération (à gauche) ou avec deux itérations (à droite). La seconde itération permet notamment de récupérer des parties perdues lors de la première itération.

l'instant restreint notre étude à ces fenêtres normalisées centrées sur la personne et de taille 96×160 pixels fixée par la base de tests. Ces conditions sont imposées par le processus de détection de Dalal [Dalal, 2006] que nous utilisons. Cependant, il serait intéressant d'élargir le sujet à des fenêtres de tailles diverses. Il est certes possible de changer la taille des fenêtres pour les faire correspondre à celle du protocole d'étude, mais, en particulier pour les images plus grandes, une étude sur l'image dans sa vraie taille devrait permettre une segmentation plus fine ("au pixel près"). De plus, nous utilisons la coupe de graphe pour sa grande adaptativité aux interactions avec l'utilisateur. Or, les informations de l'utilisateur peuvent remplacer la détection. Il peut même être demandé à l'utilisateur d'effectuer cette étape qui est d'ailleurs assez coûteuse en temps puisqu'elle nécessite de tester l'ensemble des fenêtres à toutes les localisations et échelles possibles.

Dans cette section, nous verrons donc dans un premier temps comment adapter notre méthode pour qu'elle fonctionne avec toute taille de fenêtres. Puis, nous verrons comment l'utilisateur peut remplacer le processus de détection. Notons que seule la segmentation des fenêtres plus grandes que le format de base peut être améliorée. Nous n'étudierons alors ici uniquement le cas de ces fenêtres.

5.6.1 Segmentation de fenêtres de tailles quelconques

Soit une fenêtre de détection contenant une personne. La personne y est centrée et les proportions de la fenêtre et de la personne qu'elle contient sont proportionnelles aux proportions données par les fenêtres de détection de Dalal utilisées jusqu'à présent. Notre objectif est ici de faire en sorte que notre méthode traite efficacement cette fenêtre.

Méthode $M1$: agrandissement du masque de segmentation

Dans cette première méthode, l'image est tout d'abord réduite au format 96×160 pixels. Puis la segmentation s'effectue comme précédemment. Le masque binaire obtenu est alors agrandi au format initial par interpolation. Cette méthode est simpliste et ne prends pas en compte les détails perdus par la première réduction.

Méthode $M2$: agrandissement du gabarit par parties

La structure de la coupe de graphe permet de s'adapter à n'importe quelle taille d'images puisqu'elle se base sur chaque pixel. Mais il faut alors adapter le gabarit pour correspondre à la taille de l'image puisque celui-ci indique la localisation probable des pixels du premier plan. Or, les sous-gabarits donnent juste une allure. Agrandir leur taille n'apporte donc pas plus d'information pertinente. Il est donc hors de question d'augmenter la taille des gabarits.
Nous proposons alors le protocole suivant : la fenêtre contenant la personne est réduite au format de base (96×160 pixels). Les coupes de graphes sur chaque partie nous donnent le gabarit par parties au format de base. Ce gabarit par parties est alors agrandi à la taille initiale de la fenêtre par une interpolation bilinéaire. Enfin, une coupe de graphe est réalisée à partir de la fenêtre et du gabarit par parties à l'échelle initiale de la fenêtre (figure 5.14). Ainsi, le choix du gabarit par parties est réalisé dans un format qui respecte le besoin de n'avoir qu'une allure et la coupe de graphe au format réel donne une segmentation plus fine.

Méthode $M3$: la segmentation comme gabarit

Lorsqu'il est de grande taille, un gabarit est pertinent s'il est assez précis. Or nous voulons juste ici obtenir une segmentation plus fine qu'avec le format de base. Nous proposons alors un second protocole. La fenêtre est d'abord réduite au format de base. Puis, à ce format, la segmentation est réalisée. Un nouveau gabarit est obtenu en réalisant un filtrage gaussien du masque binaire de la segmentation. Ce gabarit est alors très proche de la posture de la personne. Il est ensuite agrandi au format initial de la fenêtre. Une dernière coupe de graphe avec la fenêtre et le nouveau gabarit au format réel permet d'obtenir une segmentation plus fine (figure 5.15).

Résultats

Pour obtenir une évaluation de nos procédés, nous avons créé une nouvelle base de données de 50 images de personnes centrées avec des tailles variées. Ces images sont issues des images des bases de données de l'INRIA et de PennFudanPed relatif au

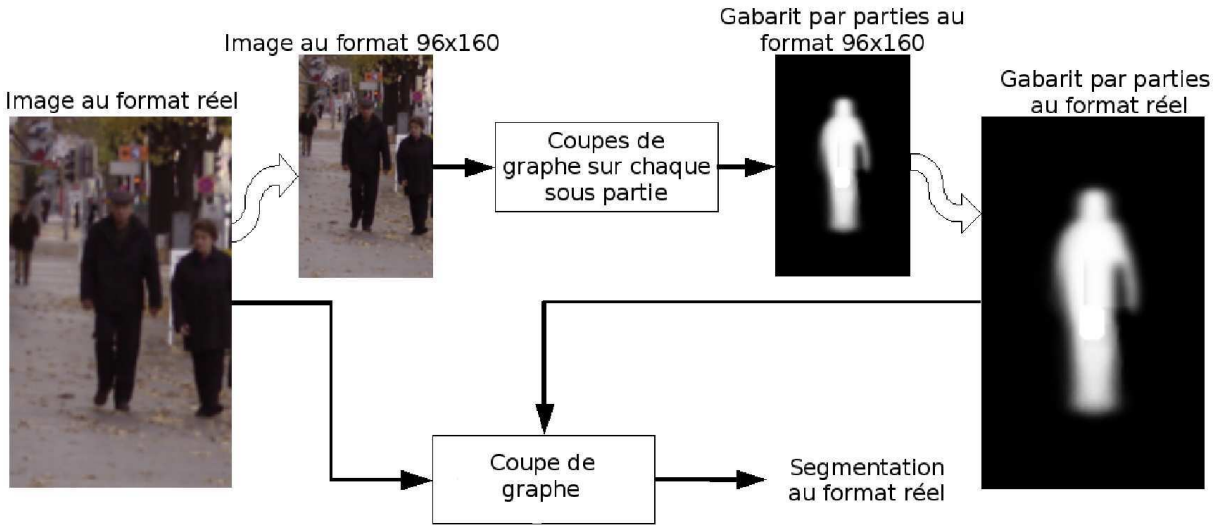

FIG. 5.14 – Première adaptation de notre méthode à des images de tailles supérieures. L'image est réduite au format de base (96×160 pixels). À partir de cette image normalisée le gabarit par partie correspondant est obtenu comme précédemment. Ce gabarit par parties est ensuite agrandi à la taille réelle de l'image. Enfin une coupe de graphe finale est réalisée avec l'image et le gabarit par parties au format réel.

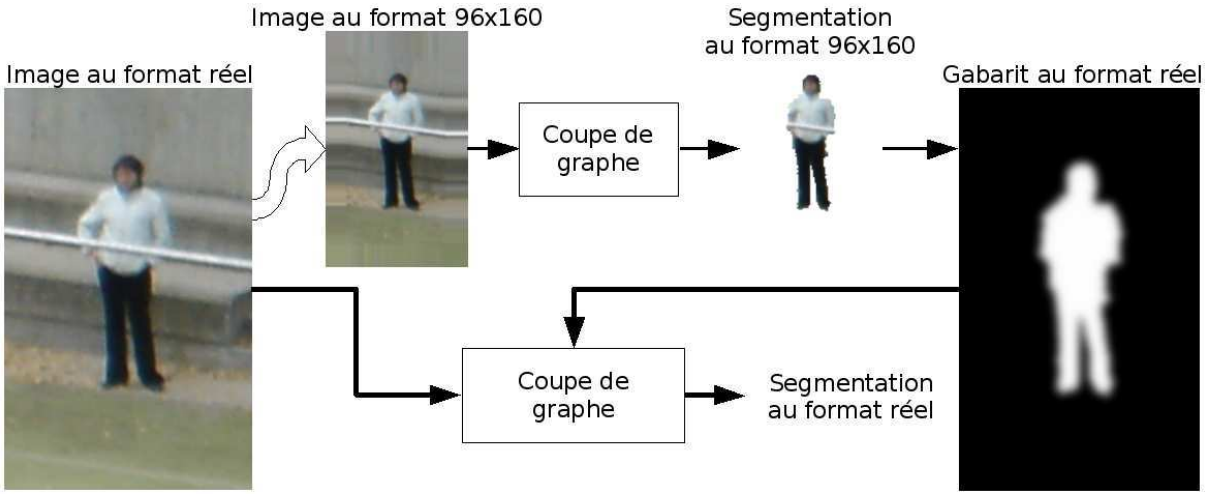

FIG. 5.15 – Seconde adaptation de notre méthode à des images de tailles supérieures. La segmentation est réalisée sur l'image au format de base. Puis un gabarit au format réel est réalisé à partir de cette segmentation. Enfin une dernière coupe de graphe au format réel donne une segmentation plus fine.

travail de Wang [Wang et al., 2007a]. Elles ont une hauteur variant de 200 à 600 pixels environ. Cette base de données nous permet une évaluation qualitative de nos procédés. Pour l'évaluation quantitative, nous utilisons la $F_{measure}$ et les mesures de Martin et de Yasnoff. Pour cela, il faut une segmentation de référence. Une segmentation manuelle des 50 images a donc été réalisée pour représenter une vérité terrain. Les valeurs données sont les moyennes sur les 50 images.

Nous testons ici les trois méthodes présentées plus haut. Nous obtenons alors les valeurs du tableau 5.3.

Mesure	$M1$	$M2$	$M3$
$F_{measure}$ (↑)	0,8517	0,8207	0,8517
Martin (↓)	0,0520	0,0646	0,0516
Yasnoff (↓)	0,4367	0,4791	0,1882

TAB. 5.3 – Performances de segmentation sur un ensemble de 50 images de tailles différentes selon les trois méthodes : agrandissement du masque de segmentation ($M1$), agrandissement du gabarit par parties ($M2$) et segmentation utilisée comme gabarit ($M3$).

$M2$ donne de bien moins bons résultats. Cela s'explique par le fait que le gabarit par parties doit lui aussi rester peu précis. Il n'est en fait qu'une allure grossière de la posture de la personne et ne fait que guider sa segmentation afin qu'elle puisse bien s'adapter aux contours de l'image. Augmenter la taille du gabarit rajoute des informations non pertinentes. La segmentation a alors tendance a être trop influencée par le gabrit plutôt que par les caractéristiques de l'images.
Les résultats des méthodes $M1$ et $M2$ sont très proches quantitativement. Visuellement, on remarque que la segmentation de $M3$ est beaucoup plus lisse et vraisemblable. En fait, comme $M3$ réalise une dernière coupe de graphe avec un gabarit formé à partir de la segmentation déjà obtenue, la segmentation est peu modifiée (donc quantitativement proche) mais juste mieux adaptée aux caractéristiques de l'image dans sa taille d'origine (donc un contour plus lisse).

5.6.2 Modes d'interactions

Jusqu'à présent, l'interaction de l'utilisateur permettait uniquement de faire des retouches à la segmentation. Nous allons maintenant voir comment l'utilisateur peut aider initialement à la segmentation en remplaçant la phase de détection. Celle-ci est en effet coûteuse en temps et l'utilisateur peut ne vouloir s'intéresser qu'à une personne en particulier dans l'image. L'interaction doit être la plus simple et la plus courte possible. Elle doit en priorité localiser la fenêtre de détection englobant la personne à étudier puisque c'est ce dont notre méthode de segmentation a besoin. Mais elle peut aussi apporter d'autres informations qui amélioreront la segmentation.
Nous proposons alors trois protocoles différents d'interaction dont nous détaillerons les avantages et les inconvénients.

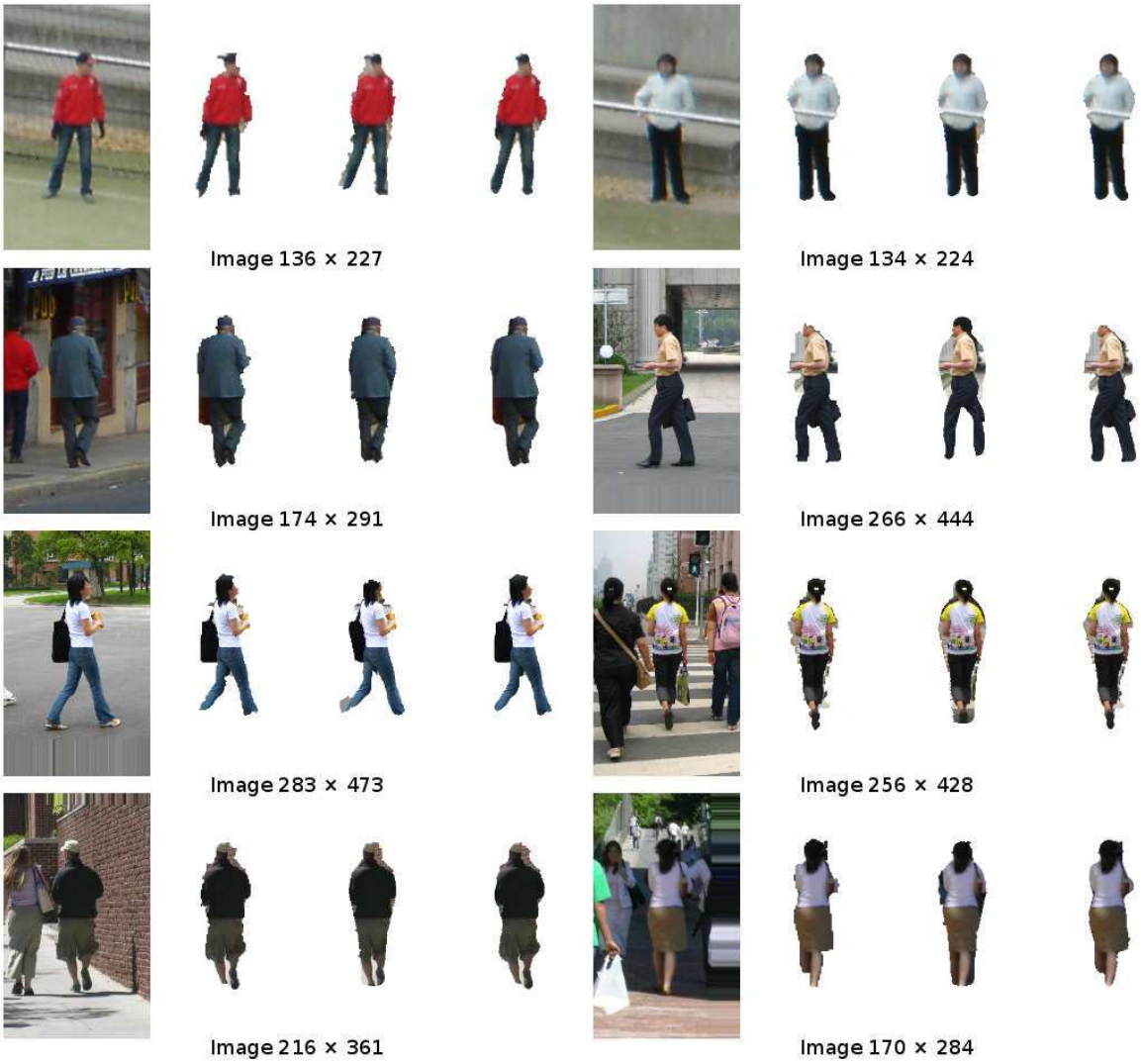

FIG. 5.16 – Segmentations obtenues par les trois méthodes sur des images de tailles différentes (indiquées en dessous). De gauche à droite : image et segmentations obtenues par les méthodes $M1$, $M2$ et $M3$.

5.6.3 Encadrement

Comme l'objectif est de réaliser une fenêtre de détection, le mode d'interaction qui semble le plus simple et le plus évident est l'encadrement de la personne (du haut du crâne jusqu'au dessous des pieds). L'utilisateur, par deux clics, encadre la personne à segmenter (figure 5.17(a)).
Dans les fenêtres dont notre méthode a besoin, la hauteur de la personne (donnée par la hauteur du cadre) représente les $\frac{5}{8}$ de la hauteur de la fenêtre. La personne est centrée dans cette fenêtre.
L'avantage de ce procédé est sa simplicité et sa rapidité. Cependant, seules la taille est la localisation de la personne sont données par l'utilisateur. Ce procédé est notamment utilisé par Rother pour son *Grabcut* [Rother et al., 2004].

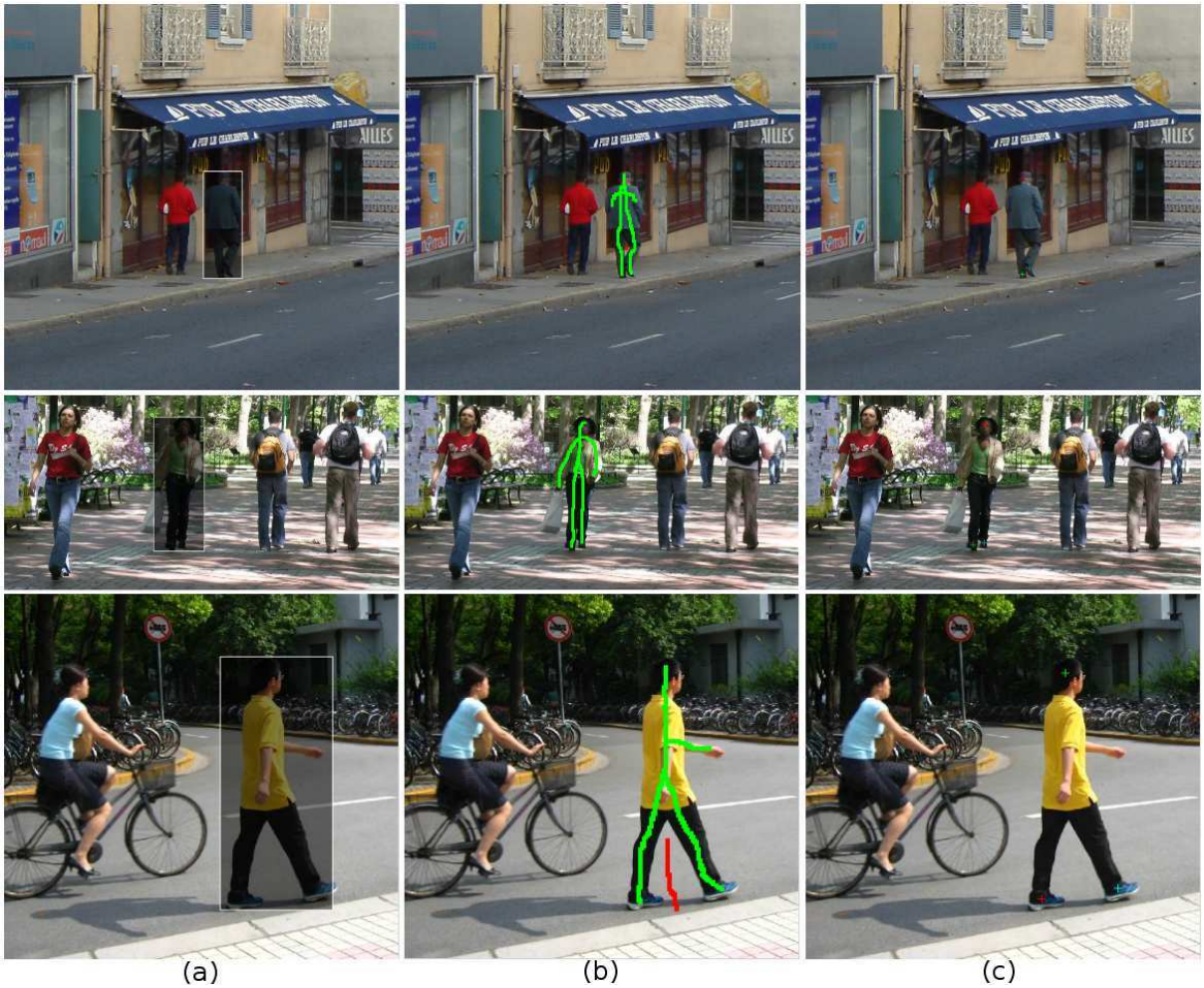

FIG. 5.17 – Trois différents modes d'interaction avec l'utilisateur : encadrement de la personne (a), marqueurs correspondants au premier (vert) et à l'arrière plan (rouge) (b) et pointeurs sur la tête et les deux jambes (c).

5.6.4 Marqueurs

L'utilisateur marque par des traits colorés des pixels de l'images appartenant au premier et à l'arrière plan. Ce procédé est souvent utilisé avec les coupes de graphes pour donner des informations de couleurs sur les deux régions à séparer [Boykov and Jolly, 2001]. Ici, l'utilisateur marque le plus de parties possible de la personne (figure 5.17(b)). La position moyenne des pixels marqués au premier plan c et leurs extrema selon la verticale sont déterminés afin de déterminer la hauteur h de la personne. La fenêtre de détection recherchée est obtenue comme précédemment.
Ce procédé est un peu plus exigeant vis à vis de l'utilisateur vu que celui-ci doit marquer un nombre important de pixels. Néanmoins, la segmentation est alors plus précise. En effet, tous les pixels marqués par l'utilisateur au premier et à l'arrière plan sont forcés (par les pondérations des t_{link}) à appartenir dans la segmentation à la région correspondante. Il y a donc moins de possibilité d'erreur.

5.6.5 Pointage de sous-parties

Enfin, dans ce mode, l'utilisateur pointe le centre de la tête et des deux pieds (figure 5.17). Le point le plus haut représente alors la tête et les deux autres les pieds. Le centre de la fenêtre est calculé avec la moyenne selon l'horizontale des points et la moyenne selon la verticale entre la tête et le plus bas des pieds. La taille de la personne est modélisée comme proportionnelle avec la distance entre les points de la tête et du pied le plus éloigné. La fenêtre de détection recherchée est alors obtenue comme précédemment.

Ce procédé est peu contraignant pour l'utilisateur et donne plus d'information pour la coupe du graphe. En effet, les pixels voisins des pixels pointés sont forcés, pour la segmentation, à appartenir au premier plan.

Résultats

Pour réaliser une évaluation des résultats, nous nous basons sur l'ensemble d'images utilisés par Dalal et Triggs [Dalal, 2006] et par Wang et al. [Wang et al., 2007a] qui ont permis d'obtenir les images de la base de données de personnes présentée précédemment. Nous avons donc une vérité terrain disponible. Les tests ont été réalisés avec une interaction effectuée par un utilisateur. Ils sont donc dépendants de cet utilisateur et sont légèrement différents à chaque essai. Ils donnent néanmoins une première idée de l'efficacité de chacun de ces modes.

Mesure	Encadrement	Marqueurs	Pointage
$F_{measure}$ (↑)	0,8248	0,8454	0,8025
Martin (↓)	0,0627	0,0566	0,0708
Yasnoff (↓)	0,4679	0,4202	0,5329

Tab. 5.4 – Performances de segmentation obtenues par trois modes d'interaction. Les marqueurs donnent une meilleure segmentation mais demandent une forte intéraction de l'utilisateur. L'encadrement est un bon compromis.

Le tableau 5.4 montre les valeurs obtenues. Les marqueurs donnent les meilleurs résultats. Ces résultats sont logiques puisque la méthode des marqueurs profite d'une interaction complexe et de beaucoup plus d'information. En revanche, l'encadrement donne de meilleurs résultats que le pointage alors que l'utilisateur donne moins d'information.
En fait, la segmentation dépend des informations supplémentaires obtenues mais en priorité du bon choix de la fenêtre de détection. Notons, pour une fenêtre de détection, h_r sa hauteur théorique de référence (qui est proportionel à sa largeur) et h sa hauteur selon un des trois modes d'interaction. La valeur $\frac{|h_r - h|}{h_r}$ donne une première idée sur la capacité de l'interaction de produire une fenêtre de taille adéquate. En moyenne sur les 50 cas, cette valeur vaut $0,020745$ pour l'encadrement, $0,045733$ pour les marqueurs et $0,035223$ pour le pointage. L'encadrement donne les fenêtres de tailles les plus précises.
En conclusion, étant la méthode la plus intuitive, l'encadrement permet d'obtenir la fenêtre de détection la plus précise. Cette fenêtre est un peu moins bien trouvée avec les marqueurs, mais le surplus d'information permet d'obtenir une segmentation plus fiable

et plus contrôlée. Elle demande cependant un travail important à l'utilisateur. Enfin, le pointage des sous-parties donne une segmentation moins précise malgré les informations supplémentaires apportées.

5.7 Conclusion

Nous avons, dans ce chapitre, présenté une nouvelle méthode de segmentation automatique de personnes permettant une interaction pertinente avec l'utilisateur. Cette interaction permet d'améliorer la segmentation obtenue automatiquement. Elle est également bien intégrée dans le processus de segmentation. Pour cela, le principe de la coupe de graphe, très couramment utilisé, permet de réaliser de bonnes segmentations semi-automatiques. Nous avons également introduit des gabarits non binaires représentant la probabilité d'un pixel d'appartenir à la silhouette recherchée. Le gabarit est soit unique (où il s'adapte à l'ensemble des silhouettes possibles), soit par parties (où il s'adapte plus précisément à la posture de la personne étudiée). Ces gabarits permettent d'introduire la connaissance sur la classe de façon cohérente puisqu'ils s'insèrent dans le principe même de la coupe de graphe. La segmentation obtenue est efficace et robuste (figure 5.11). Ces travaux ont donnés lieu à des articles présentés lors des conférences ICIP 2011 [Migniot et al., 2011a] et GRETSI 2011 [Migniot et al., 2011b].

Cette méthode est réalisée pour compléter la détection effectuée par la méthode de Dalal [Dalal, 2006]. Néanmoins, nous avons montré que la méthode s'adaptait bien à tout format de fenêtre de détection. La phase de détection peut notamment être remplacée par une interaction de l'utilisateur. La méthode n'est alors plus automatique mais gagne beaucoup en temps de calcul sans perdre en qualité.

Chapitre 6

Adaptation de la coupe de graphe sur les séquences vidéo et pour la recherche de trimaps

> Et si tout n'était qu'illusion et que rien n'existait ? Dans ce cas, j'aurais vraiment payé mon tapis beaucoup trop cher.
>
> Woody Allen

Sommaire

Le chapitre 5 présentait une segmentation automatique de personnes sur les images fixes par coupe de graphe. La coupe de graphe permet une interaction cohérente et efficace avec l'utilisateur. Mais cette méthode peut être adaptée pour d'autres applications.
Tout d'abord, nous avons vu dans la section 1.1 que de nombreuses applications nécessitent l'étude de séquences vidéo. Une méthode de segmentation sur image fixe peut être réalisée sur chaque *frame*. Néanmoins, le temps de traitement est alors

très important alors que l'information contenue dans des *frames* successives est très redondante. De plus, le mouvement des différents objets de la séquence est une donnée qui peut aider à la segmentation. Enfin, traiter chaque *frame* séparément peut nuire à la continuité temporelle de la segmentation. Dans la section 6.1, nous proposons plusieurs méthodes utilisant la coupe de graphe pour segmenter les personnes dans les séquences vidéo : la première optimise l'interaction de l'utilisateur, la seconde découpe la séquence temporellement en fenêtres glissantes puis recherche un gabarit par parties à appliquer aux *frames* d'une fenêtre glissante et la dernière utilise un gabarit spatio-temporel.
Le mouvement des jambes est le plus caractéristique de la classe des personnes. Nous nous intéressons donc principalement au cas des personnes qui marchent perpendiculairement à l'axe optique car c'est celui où ce mouvement est le plus visible.

Ensuite, pour obtenir une segmentation très précise et vraisemblable, il est utile de déterminer le canal alpha (ou transparence) : c'est le *matting*. Or, la plupart des méthodes de *matting* nécessite un trimap de l'objet à segmenter. Mais la réalisation d'un trimap est un travail long et fastidieux. Nous proposons alors en section 6.2, une méthode de réalisation automatique de trimap de personnes.

6.1 Étude de séquences vidéo

6.1.1 Considération 3D

L'analyse de vidéo est un travail différent de celui des images fixes. Considérer chaque *frame* séparément est un mauvaise façon de procéder. Tout d'abord l'information comprise dans les autres *frames* n'est pas prise en compte. Ensuite le temps de calcul est important alors qu'il pourrait être réduit. En effet, la redondance d'information entre deux *frames* successives est importante et la segmentation entre ces deux instants est assez semblable. Il est donc intéressant de travailler sur une vidéo dans son ensemble. Une démarche est de considérer la vidéo comme un objet en trois dimensions. Aux deux dimensions spatiales de l'image est rajoutée une dimension temporelle. Ainsi, un pixel d'une *frame* y est voisin avec les pixels spatialement situés au même endroit mais à l'instant précédent et suivant. La vidéo peut se représenter sur ses trois dimentions comme sur la figure 6.1(a). La segmentation de la vidéo donne l'ensemble des segmentations pour chaque *frame*. Ainsi, le premier plan n'est plus comme précédemment une surface mais maintenant un volume.

De nombreuses méthodes utilisent cette considération spatio-temporelle. Kellokumpu et al. [Kellokumpu et al., 2008] découpent l'image 3D en cubes spatio-temporels. Ils calculent ensuite un histogramme de texture dans chacun de ces cubes pour obtenir un descripteur qui, associé à un classifieur modèle de Markov caché (HMM), permet de reconnaître une activité humaine.
Blank et al. [Blank et al., 2005] étudient l'enveloppe de la segmentation spatio-temporelle pour reconnaître des activités (figure 6.1(c)). L'équation de Poisson donne pour chaque pixel la longueur moyenne d'un chemin aléatoire pour atteindre un point de cette enveloppe. Ces valeurs, ainsi que leurs matrices hessiennes, permettent de reconnaître le torse et les membres et donc de définir le mouvement.

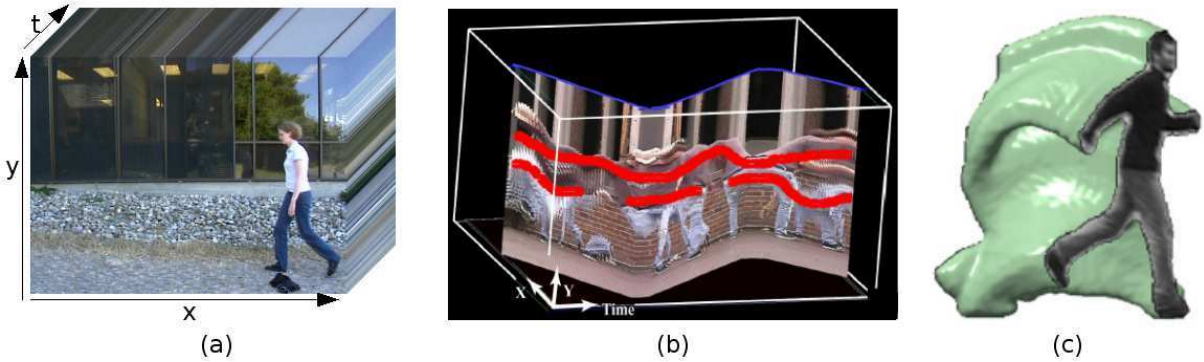

FIG. 6.1 – Visualisation et manipulation 3D de vidéos. (a) La vidéo peut être représentée par un objet 3D avec les deux dimensions spatiales de l'image et la dimension temporelle. (b) Wang et Cohen [Wang and Cohen, 2005] réalisent une coupe de la représentation 3D pour que l'utilisateur puisse facilement annoter la vidéo. (c) Blank et al. [Blank et al., 2005] étudient l'enveloppe réalisée par la segmentation d'une personne à travers le temps. La forme de cette enveloppe leur permet de reconnaître l'activité de cette personne.

Hernandez et al. [Hernández et al., 2010] réalisent la détection de personne par Dalal et Triggs [Dalal and Triggs, 2005]. Les pixels en dehors de la fenêtre de détection spatio-temporelle sont assignés à l'arrière plan et les pixels de peau sont assignés au premier plan. Une coupe de graphe itérative permet ensuite la segmentation.
La coupe de graphe 3D est en effet possible. [Boykov and Jolly, 2001] adapte sa méthode sur les images 2D en utilisant la représentation spatio-temporelle et en reliant les noeuds des pixels voisins spatialement mais aussi temporellement. La coupe du graphe est effectuée semblablement à l'étude 2D. Kohli et Torr proposent néanmoins une méthode dynamique [Kohli and Torr, 2007] par coupe de graphe 2D sur chaque *frame* où le graphe (respectivement la coupe) est obtenu à partir de la *frame* précédente en adaptant les pondérations (respectivement le flot). Néanmoins, le nombre de pixels et de liaisons devenant très important, le calcul devient plus complexe et plus long. Wang et Cohen [Wang and Cohen, 2005] réalisent un pré-traitement visant à réunir les volumes cohérents (où le changement de couleur est faible) pour les considérer comme un seul noeud et ainsi réduire notablement la complexité du problème. La précision perdue par ce procédé est ensuite récupérée par un post-traitement où le principe de coupe de graphe est appliqué uniquement sur une région restreinte correspondant à la dilatation de l'enveloppe séparant les deux régions dans la segmentation de la première étape. La complexité est là aussi réduite. Le second point d'intérêt de ce papier est une nouvelle façon de réaliser l'interaction avec l'utilisateur. Assigner à la main les pixels du premier et à l'arrière plan est difficile puisque l'étude se réalise sur un objet 3D et la représentation du logiciel ne peut être qu'en deux dimensions. La solution est de découper le volume de la vidéo 3D. La tranche obtenue (appelée coupe) est une représentation 2D pouvant être manipulée par l'utilisateur (figure 6.1(b)). L'idée de coupe est aussi utilisée par Niyogi et Adelson [Niyogi and Adelson, 1994]. Ils remarquent que le motif d'une coupe obtenue par une découpe au niveau des jambes suit un certain gabarit. Cette modélisation leur permet de déterminer la segmentation sur cette coupe. Ils prolongent alors par continuité sur le reste de la vidéo 3D afin de réaliser la segmentation du volume.

6.1.2 Choix de la coupe 2D de la vidéo disponible pour l'interaction

La coupe de graphe est une méthode efficace de segmentation interactive. Le travail de l'utilisateur est réduit (quelques traits pour l'assignation de pixels) et la segmentation est facilement modifiable. En dehors de la taille du graphe, le passage de l'étude d'une image à une vidéo sous forme d'un objet 3D ne change presque rien au procédé. La méthode est très pertinente pour la segmentation interactive de vidéo. La difficulté reste néanmoins de permettre une interaction facile et pratique à l'utilisateur. Il faut en effet que celui-ci assigne un certain nombre de pixels appartenant au premier et à l'arrière plan. Sur une image fixe, cela est très aisé. Il suffit de tracer sur l'image un trait. Les pixels appartenant à ce trait sont assignés à une des deux régions (figure 5.1(d)). Cependant, le problème se pose désormais pour les séquences vidéo. La représentation de l'objet à trois dimensions xyt sur un écran d'ordinateur est impossible. Il faut donc trouver un affichage permettant à l'utilisateur de visualiser et de choisir les pixels d'intérêt de l'objet 3D.
Une première possibilité est de faire choisir une ou plusieurs *frames* de la vidéo à l'utilisateur (il n'est pas concevable de les lui faire toutes manipuler une par une) pour qu'il les manipule (en marquant certains pixels). Par continuité temporelle d'intensité, l'information se transmet dans le temps. Cependant, l'utilisateur n'a ici aucune emprise sur le domaine temporel. Si la vidéo est suffisamment longue, la propagation d'information se disperse et la segmentation se dégrade. Il est donc intéressant d'avoir une visualisation spatio-temporelle.

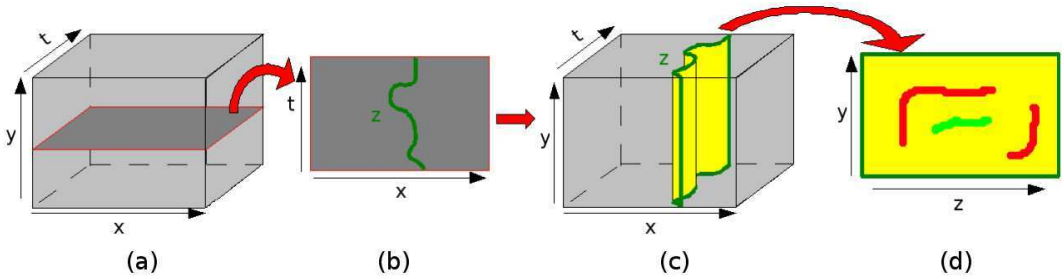

FIG. 6.2 – Principe d'interaction avec l'utilisateur de la méthode de Wang et Cohen [Wang and Cohen, 2005]. (a) Une coupe selon l'axe y est choisie. (b) un axe z est tracé sur cette coupe en relation avec la présence de l'élément d'intérêt. (c) cet axe est prolongé sur toute les coupes selon y. (d) sur la coupe YZ, l'utilisateur peut assigner des pixels en les marquant (ici par des traits verts et rouges).

Wang et Cohen[Wang and Cohen, 2005] proposent de faire une coupe spatio-temporelle dans l'objet 3D afin d'avoir une représentation 2D à visualiser (figure 6.1(b)). Ils réalisent déjà une coupe de la vidéo selon la hauteur (y constant) pour obtenir une visualisation XT (figure 6.2(a)). Puis l'utilisateur trace une ligne représentant un axe $z(x,t)$ suivant l'élément d'intérêt de l'image (figure 6.2(b)). Enfin, l'algorithme affiche la représentation 2D ZY où l'utilisateur peut marquer les pixels (figure 6.2(d)).
Cette interaction n'est pas forcément très intuitive pour l'utilisateur. Il faut savoir quelle coupe choisir et arriver à tracer l'axe z le plus pertinent. Néanmoins cette idée de réaliser des coupes pour permettre une bonne visualisation à l'utilisateur est bien fonctionnelle. Le point suivant traite de l'utilisation de la connaissance de la classe recherchée pour obtenir automatiquement une coupe performante.

Coupe relative au mouvement humain

Un des aspects les plus délicats de la segmentation de personnes est la segmentation des membres et en particulier des jambes qui sont plus souvent détachées du corps. Lorsque les jambes sont écartées, le contour de la silhouette change de direction de façon plus abrupte et le suivi de ce contour est donc plus complexe. Dans notre méthode de segmentation, il s'agit d'un artefact très gênant du processus car il oblige à passer par un plus grand nombre de noeuds du graphe. Il est donc bien pertinent de faire disposer à l'utilisateur d'un contrôle sur le comportement spatio-temporel des jambes.

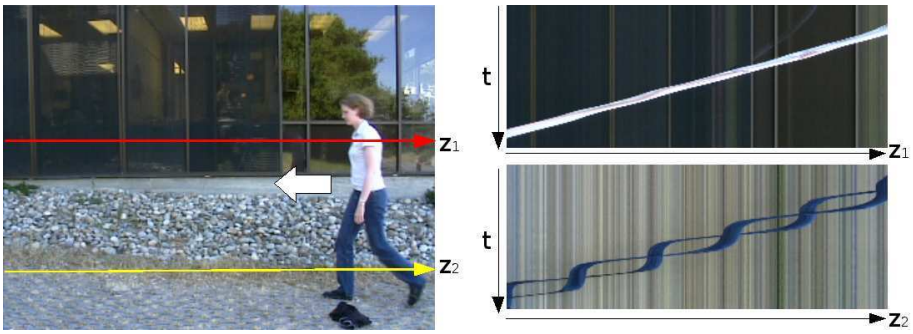

FIG. 6.3 – Sur une vidéo d'une personne se déplaçant par translation de la droite vers la gauche, des coupes sont réalisées au niveau du torse (z_1) et des jambes (z_2). La visualisation ZT des coupes montrent un mouvement linéaire du torse et un mouvement caractéristique des jambes.

Concerant ce problème, Niyogi et Adelson [Niyogi and Adelson, 1994] basent leurs travaux sur l'étude des coupes spatio-temporelles de séquences vidéo au niveau des jambes (figure 6.4(a)). La coupe transversale selon l'axe Y donne un motif très particulier qui permet de bien définir le mouvement des jambes. Ce mouvement est naturellement très fortement relié au déplacement de la personne et au mouvement de la caméra. Pour définir le déplacement de la personne dans l'image, une seconde coupe est réalisée au niveau du torse. Le déplacement du torse étant le même que le mouvement global de la personne, il est possible de recadrer la séquence et ainsi compenser le mouvement de la caméra. Niyogi, lui, se place dans le cas le plus simple du déplacement rectiligne et régulier d'une personne. De plus, pour bien visualiser l'étendue de la marche, le déplacement se réalise selon la perpendiculaire à l'axe de la caméra. Le motif de la coupe au niveau du torse donne alors un trait rectiligne (figure 6.3 en haut à droite). Ce trait permet notamment déterminer la vitesse du déplacement.
Cette droite permet de mieux analyser la coupe au niveau des jambes (figure 6.3 en bas à droite) qui suit le même déplacement avec des écarts correspondant au balancement périodique des jambes. Niyogi modélise alors le motif de cette coupe selon un gabarit (figure 6.4(b)) dépendant de trois variables : l'amplitude A correspondant à l'étendue d'une enjambée, la période T de déplacement des jambes et le pas p lié à la durée d'une enjambée.
Niyogi utilise cette coupe, en dehors du fait que les jambes sont une des principales difficultés de la segmentation, parce qu'elle est hautement caractéristique de la classe des personnes. La présence de ce motif dans une vidéo donne un indice fort sur la présence d'une personne. Il l'utilise donc aussi pour la détection de personnes.

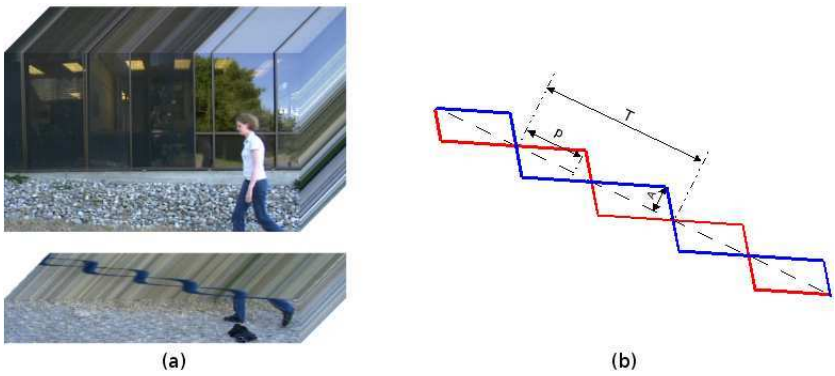

FIG. 6.4 – Le mouvement des jambes est très particulier. Pour le visualiser, il suffit de réaliser une coupe selon Y à leur niveau (a). Cette coupe présente un motif qui peut être modélisé par un gabarit présenté en (b) défini par trois variables : la période T, le pas p et l'amplitude A. Les courbes rouge et bleu représentent le mouvement des deux jambes et le trait en pointillé représente la vitesse de la personne (calculée à partir de la coupe au niveau du torse).

L'intérêt de la coupe au niveau des jambes est donc double. Premièrement, elle permet de donner à l'utilisateur un accès direct sur le mouvement des jambes, qui est une des plus grosses difficultés de la segmentation, et donc d'y avoir une emprise plus efficace. Ensuite, elle affiche un élément très discriminant de la classe étudiée. Par conséquent, elle donne un accès pertinent aux informations et s'adapte bien à l'objet étudié. Le mouvement de la personne est finalement assez bien décrit par l'union des deux coupes aux niveaux des jambes et du torse. Pour notre première méthode, nous choisissons alors ces deux représentations pour l'interaction avec l'utilisateur afin qu'il assigne un certain nombre de pixels au graphe. Son travail est alors assez réduit et plutôt aisé puisqu'il suffit de marquer deux images qui représente un motif régulier. Le choix de ces représentations permet, malgré la petite quantité d'information demandée à l'utilisateur, d'obtenir la meilleure performance de segmentation.

Protocole et résultats

Nous présentons ici notre première méthode où la segmentation d'une séquence vidéo est réalisée par la méthode du *graph-cut* de Boykov[Boykov and Jolly, 2001] avec initialisation de l'utilisateur réalisée à partir de coupes de l'objet vidéo 3D au niveau des jambes et du torse. Les coupes permettent d'obtenir les meilleurs résultats avec peu d'interaction.
Le graphe est réalisé tel que décrit section 5.1. Deux noeuds (premier et arrière plan) définissent la source et le puits. Puis un noeud est créé pour chaque pixel de l'objet vidéo 3D. Pour chacun de ces noeuds, un arc est créé entre lui et la source et le puit avec une énergie calculée selon les équations 5.2 et 5.3, et avec chacun de ses voisins spatio-temporels selon l'équation 5.1. L'utilisateur assigne dans l'objet vidéo 3D un certain nombre de pixels sur des représentations des coupes. Ensuite, l'algorithme de recherche de coupe minimale de Boykov [Boykov and Kolmogorov, 2004] calcule la coupe de l'objet vidéo 3D voulue et réalise ainsi la segmentation de la personne sur l'ensemble des *frames* de la séquence.

Pour réaliser les tests, nous prenons une séquence de 50 images réalisée par Sidenbladh [Sidenbladh et al., 2000]. Cette séquence est de résolution 320 × 240 pixels. Elle contient une personne qui se déplace de façon régulière et rectiligne de la droite de la *frame* vers la gauche.
Pour rester en cohérence avec le travail précédent, nous continuons de travailler sur des fenêtres de détection de taille 96 × 160. Nous recadrons donc l'ensemble des *frames* pour obtenir un suivi de la personne dans une fenêtre de taille contrôlée (figure 6.5(b)). Comme la détection se réalise de façon efficace à partir de la méthode de Dalal et Triggs [Dalal and Triggs, 2005], le suivi se réalise ici de façon tout aussi efficace par la méthode de Dalal et al. [Dalal et al., 2006]. Le principe de segmentation basée sur une détection performante est alors conservé.

Le recadrage des *frames* permet également de réduire la taille de l'objet vidéo 3D. En effet, les trois dimensions engendrent un nombre important de noeuds et un nombre encore plus important d'arcs. La complexité du problème peut très vite devenir trop importante et demander un temps de calcul trop important. Pour réduire encore cette complexité, nous réalisons les liaisons entre le noeud d'un pixel et seulement les noeuds des 6 pixels les plus proches (figure 6.5(a)) spatio-temporellement (et non des 26). Cela suffit à réaliser des transitions correctes.

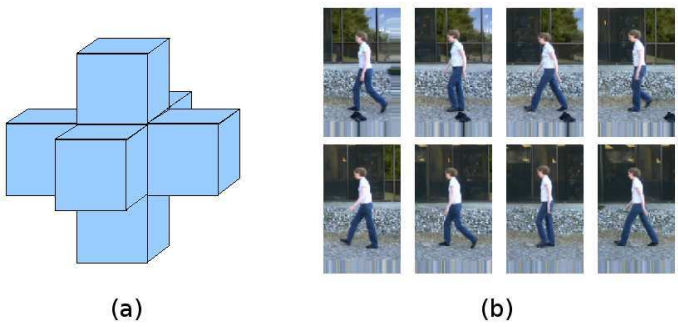

FIG. 6.5 – (a) représentation des 6 voisins spatio-temporels les plus proches d'un pixel. (b) exemples de *frames* de la séquence étudiée recadrées pour rentrer dans notre cadre d'étude.

Comme pour chaque *frame* la personne est centrée dans la fenêtre, les positions des jambes et du torse sont connues. Dans notre cas, pour laisser à l'utilisateur un cadre d'étude plus large, une troisième coupe est disponible au niveau de la tête. Cette coupe n'est pas nécessaire mais la tête possède des caractéristiques en couleur très spéciales et en donner l'accès à l'utilisateur peut être intéressant. Le recadrage provoqué par le suivi conserve bien la particularité du motif présent sur la coupe au niveau des jambes comme on le voit figure 6.6. Sur cette visualisation des trois coupes, l'utilisateur marque des pixels par un trait d'une couleur pour les pixels du premier plan ou d'une autre pour les pixels de l'arrière plan, ce qui réalise leur assignation dans le graphe. Il est nécessaire qu'un nombre suffisant de pixels soient attribué. Pour cela, les pixels les plus à l'extérieur du cadre, qui n'ont aucune chance d'appartenir à la silhouette, sont assignés d'office à l'arrière plan.

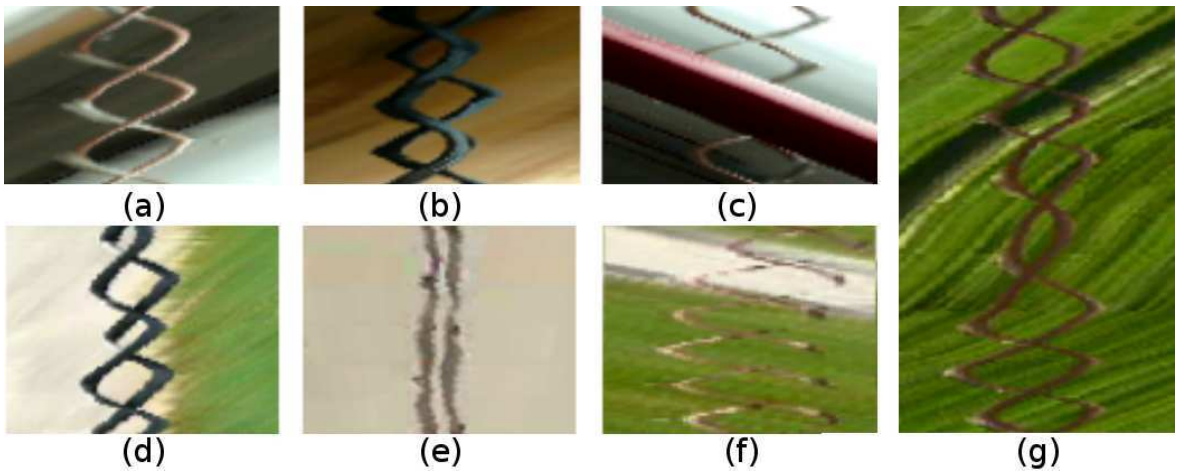

FIG. 6.6 – Coupe au niveau des jambes sur plusieurs séquences vidéo de la base de donnée de l'annexe B. Le motif caractéristique du mouvement humain y est bien reconnaissable. De plus, certaines particularités sur leurs formes donnent d'autres informations sur le mouvement réalisé par la personne. On reconnaît des marches perpendiculaires à l'axe de la caméra pour (a), (b), (c), (f) et (g), oblique (avec un angle de 45°) pour (d) et de face pour (e). De plus, (c) fait apparaître une occultation au cours du mouvement (passage derrière un siège) et (f) représente une personne courant. Enfin, (g) correspond à une personne qui change de vitesse qui provoque la discontinuité du motif.

Pour quantifier la performance de segmentation, nous utilisons une nouvelle fois les trois critères utilisés dans la partie précédente ($F_{measure}$, mesure de Martin et de Yasnoff). La segmentation est réalisée sur la séquence de Sidenbladh présentée plus haut. Les performances sont données par la moyenne de ces critères sur l'ensemble des *frames* de la séquence. Pour bien analyser les résultats et rendre compte de l'amélioration (en effet, l'objectif est de déterminer si l'accès et la manipulation des coupes apporte une amélioration des performances par rapport à des méthodes plus classiques), plusieurs protocoles sont testés :

- Protocole A : l'assignation est réalisée à partir de la première *frame*. L'information doit alors se propager à travers le temps. Ce procédé est un des plus courant mais ne prend aucunement en compte le mouvement.

- Protocole B : l'utilisateur visualise les trois coupes et les marque par des traits colorés. C'est, au final, le protocole que nous proposons.

- Protocole C : l'utilisateur assigne des pixels sur les coupes et sur la première *frame*. Il s'agit d'un mélange des deux premiers protocoles.

- Protocole D : l'assignation des coupes se fait sur l'ensemble des pixels par la vérité terrain. Cette interaction n'est pas vraiment envisageable car elle demande un travail trop important à l'utilisateur. Ce protocole donne cependant une référence pour juger les autres protocoles.

- Protocole E : au protocole D on ajoute l'assignation de la première *frame*.

Les résultats obtenus sont affichés dans le tableau 6.1.

Mesure	Protocole A	Protocole B	Protocole C	Protocole D	Protocole E
$F_{measure}$ (↑)	0,8335	0,9149	0,9147	0,9156	0,9185
Martin (↓)	0,0504	0,0286	0,0286	0,0288	0,0279
Yasnoff (↓)	1,7659	0,4272	0,4273	0,4272	0,4203

TAB. 6.1 – Évaluation des performances de segmentation sur la séquence étudiée selon trois critères avec plusieurs protocoles d'assignation. Parmi les protocoles nécessitant une intéraction raisonnable, celui que nous proposons donne les meilleurs résultats.

Tout d'abord, nous remarquons que les performances sont assez élevées et démontrent l'efficacité de la segmentation par coupe de graphe (il faut cependant remarquer que nos résultats se basent sur une seule séquence assez simple). Ensuite, comme nous l'espérions, le protocole B (celui que nous proposons) donne selon les trois critères de bien meilleurs résultats que le protocole A (le plus instinctif et usuellement utilisé). Avec le protocole C, l'ajout de l'assignation de la première *frame* n'aide finalement pas à la segmentation du protocole B et le dégrade même. Le protocole D sert à analyser l'influence du marquage sur les performances. Le marquage est ici "idéal" car l'ensemble des pixels des coupes étudiées sont assignés selon la vérité terrain. Il est en effet impossible de réaliser plus complète indication au programme. Bien entendu, les trois critères prouvent la supériorité de cette assignation. Néanmoins l'amélioration n'est pas très élevée. Cela démontre la robustesse de la coupe de graphe puisqu'une interaction rapide et simple comme le marquage par quelques traits permet d'obtenir de bons résultats. Enfin, le dernier protocole allie les protocole A et D. Les critères donnent ici les meilleures performances. Néanmoins l'amélioration, ici aussi, est assez faible et, par contre, le travail à effectuer par l'utilisateur est très important. Ces résultats montrent donc que, pour notre séquence, le simple marquage est suffisant et que l'utilisation des coupes permet bien d'avoir une interaction plus pertinente qui améliore les performances de la segmentation.
Pour le calcul des poids entre pixels voisins, la valeur de σ permet de définir la tolérance d'écart de couleur qui est directement lié à la résolution de l'image. Sur une image basse résolution, le passage entre deux pixels voisins peut être plus abrupte, ce qui ne doit pas défavoriser sa coupe dans le graphe. Pour notre séquence, par des tests sur un ensemble de valeur, nous trouvons une valeur optimale de σ égale à $10,5$.

6.1.3 Attribution d'un gabarit par parties sur les frames d'une fenêtre glissante

Nous présentons dans cette section une nouvelle méthode de segmentation de personnes dans les séquences vidéo en reprenant la méthode de coupe de graphe associée avec des gabarits présentée en section 5.2.

Principe

Le gabarit unique présenté dans la section 5.2 s'utilise pour toute les fenêtres de détection 2D. Pour l'utiliser sur une séquence 3D, il suffit donc de l'associer à chacune

des *frames* qui la composent.
En revanche, le gabarit par parties présenté en section 5.3 s'adapte à la posture de la personne. Or, au cours de la séquence, la personne change de posture (figure 6.7). Le même gabarit ne peut donc pas être appliqué à l'ensemble de la séquence.
Néanmoins, dans une séquence, la posture d'une personne ne change pas brutalement.

FIG. 6.7 – Dans une séquence, la posture d'une personne change. Il est donc peu performant d'utiliser un même gabarit pour toute la séquence.

Ainsi, sur une petite suite de *frames* consécutives, il est possible d'utiliser un seul et même gabarit. L'idée est alors de réaliser une fenêtre glissante afin de traiter la séquence morceau par morceau. Chaque fenêtre sélectionne un gabarit par parties qu'elle associe à chacune des *frames* la composant.

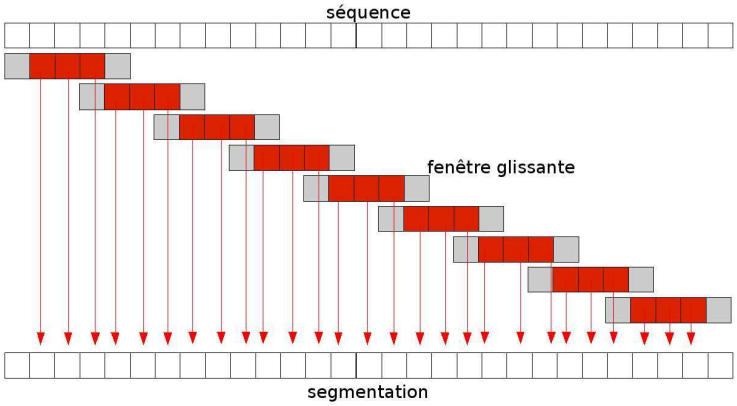

FIG. 6.8 – La fenêtre glissante découpe la séquence en sous-séquences. Chaque fenêtre est traitée séparément. Pour garder une bonne continuité temporelle, les segmentations des *frames* de bords de la fenêtre glissante (en gris) ne sont pas gardées. En revanche, les segmentations des *frames* de segmentation de la fenêtre glissante (en rouge) sont incorporées dans la segmentation de la séquence. Pour obtenir une segmentation de la séquence entière, il faut donc réaliser un recouvrement des fenêtres glissantes.

Une fenêtre glissante est composée de n_f *frames*. La fenêtre se déplace le long de la séquence pour traiter toutes les *frames* de la séquence (figure 6.8). Une coupe de graphe est alors réalisée sur chaque fenêtre glissante. Les *frames* se trouvant sur les bords de la fenêtre glissante (appelées *frame* de bords) ont peu ou pas de voisines. Afin de garder la continuité temporelle, les segmentations de celles-ci ne sont pas conservées pour la séquence complète (seul les *frames* centrales, appelées *frames* de segmentation, le sont) (figure 6.9). Afin d'obtenir la segmentation de la séquence dans son ensemble, un recouvrement des fenêtres glissantes est effectué. Soit, pour une fenêtre glissante, n_r le nombre de *frames* de bords

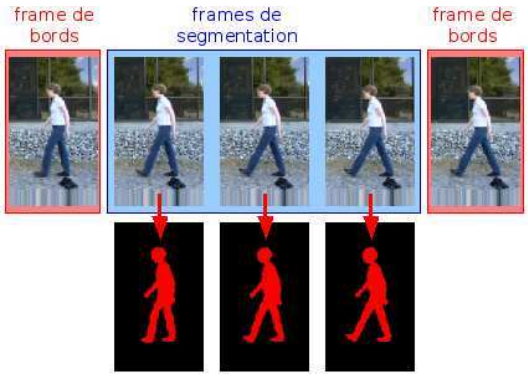

FIG. 6.9 – Seules les segmentations des *frames* de segmentation de la fenêtre glissante sont répercutées dans la segmentation de la séquence compléte. Les *frames* de bords servent à maintenir une continuité temporelle.

et n_s le nombre de *frames* de segmentation. On a donc $n_f = n_s + 2n_r$ *frames* dans une fenêtre. Soit N le nombre de *frames* dans la séquence, on a alors besoin de $\frac{N-2n_r}{n_s}$ fenêtres glissantes.

Continuité entre fenêtres glissantes

Les coupes de graphe des différentes fenêtres glissantes sont effectuées indépendamment. Dans la segmentation de la séquence complète, la continuité entre les segmentations relatives à deux fenêtres glissantes successives n'est pas assurée. Ce manque de continuité est très visible par l'utilisateur.

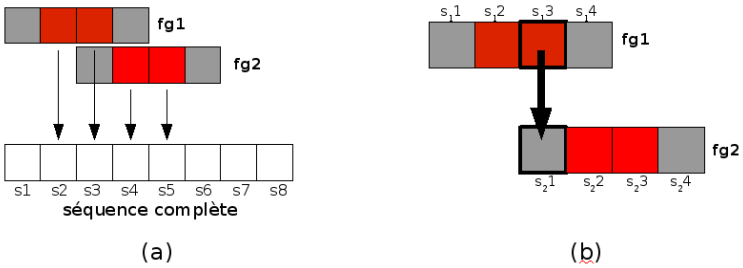

FIG. 6.10 – Comme les coupes de graphe pour les fenêtres glissantes $fg1$ et $fg2$ sont indépendantes, les segmentations des frames $s3$ (provenant de $fg1$) et $s4$ (provenant de $fg2$) ne sont pas nécessairement continues temporellement (a). Pour remedier à cela, la segmentation de la frame $s_1 3$ de la fenêtre glissante $fg1$ est imposée comme segmentation de la frame $s_2 1$ de la fenêtre glissante $fg2$ (b).

Pour remédier à ce problème, la segmentation de la première *frame* de la fenêtre glissante (qui fait partie des *frames* de bords et n'est donc pas prise en compte pour la segmentation de la séquence) est forcée à correspondre à la segmentation de la *frame* correspondante dans la séquence (figure 6.10). En effet, cette dernière a été calculée avec la fenêtre glissante précédente. Pour cela, le poids des arêtes de liaisons du graphe correspondantes sont mis à ∞ (comme avec l'interaction de l'utilisateur). Un rendu bien plus agréable et vraisemblable est alors obtenu.

Performances

Des tests sur les séquences de la base de vidéos que nous avons créée (annexe B) permettent d'évaluer qualitativement les performances du processus. Les évaluations quantitatives ont été réalisées sur la séquence de la fille marchant issue du travail de Sidenbladh [Sidenbladh et al., 2000] et disponible sur *http ://www.cs.brown.edu/ black/* ainsi que sur les séquences *ThibaudCroisement*, *Luiz* et *JonathanOblique* de notre base. Quatre exemples ne sont pas représentatifs de la classe mais donnent une première évaluation.
Les sous-gabarits utilisés pour réaliser les gabarits par parties sont ceux de le figure 5.7.

Réglages et compromis : le principal réglage à effectuer est la taille des fenêtres glissantes et donc la valeur des paramètres n_s et n_r. En prenant un nombre de *frames* de recouvrement (n_r) supérieur à 1, on n'obtient pas, selon nos tests, un meilleur résultat. Ensuite, augmenter n_r revient à agrandir la taille des fenêtres glissantes sans en réduire le nombre. Cela revient alors à augmenter sensiblement le temps de calcul sans apporter d'amélioration aux performances. Enfin, le nombre de *frames* segmentées (n_s) permet d'assurer un peu plus la continuité de la segmentation. Cependant, sur un nombre trop important de *frames* consécutives, la posture d'une personne change et ne peut donc correspondre à un unique gabarit par parties. Par expérience on favorise une valeur $n_s = 2$. Les performances obtenues sur les séquences testées sont présentées sur le tableau 6.2.

séquence	n_s n_r	3 1 ● ○ ○ ○ ●	2 1 ● ○ ○●	1 1 ● ○ ●	2 2 ● ● ○ ○ ●●
Fille	$F_{measure}$ (↑)	**0,8030**	0,7999	0,7886	0,7782
	Martin (↓)	**0,0542**	0,0554	0,0680	0,0665
	Yasnoff (↓)	**0,4740**	0,4814	0,5282	0,5250
Thibaud	$F_{measure}$ (↑)	0,7785	**0,7856**	0,7797	0,7396
	Martin (↓)	0,0585	**0,0563**	0,0576	0,0688
	Yasnoff (↓)	0,5321	**0,5012**	0,5210	0,6160
Luiz	$F_{measure}$ (↑)	0,7175	**0,7272**	0,7137	0,7153
	Martin (↓)	0,0675	**0,0621**	0,0666	0,0649
	Yasnoff (↓)	0,8571	**0,7918**	0,8348	0,8390
Jonathan	$F_{measure}$ (↑)	0,7864	**0,7870**	0,7859	0,7855
	Martin (↓)	0,0659	**0,0652**	0,0648	0,0655
	Yasnoff (↓)	0,5983	**0,5967**	0,6025	0,5627

Tab. 6.2 – Effet de la taille des fenêtres glissantes sur les performances de segmentation des quatre séquences testées. Pour chaque séquence, les meilleurs résultats sont en gras. Les ronds représentent la position des *frames* de segmentation (ronds blancs) et des *frames* de bords (ronds noirs) dans une fenêtre glissante. L'association $n_s = 2$ et $n_r = 1$ donne les meilleurs résultats sauf pour la première séquence.

Protocoles d'interaction : que l'étude porte sur la 2D ou la 3D, le *graph cut* permet l'interaction de l'utilisateur par marquage de pixels du premier et de l'arrière plan. Comme précédemment, nous proposons plusieurs protocoles d'interaction :

- Seule la première *frame* à segmenter est marquée par l'utilisateur. Comme la coupe de graphe maintient une continuité temporelle, l'information est propagée dans le temps. Dans ce cas, l'information est rapidement réduite et les segmentations des dernières *frames* ne tiennent plus compte de l'interaction de l'utilisateur (figure 6.11).
- L'utilisateur marque une coupe spatio-temporelle au niveau des jambes. Dans ce cas, comme les jambes sont souvent peu variantes en couleur et texture, l'information de l'utilisateur se diffuse plus efficacement. De plus, le marquage est aisé car le motif à marquer est très facilement reconnaissable. Seules les jambes sont alors annotées par l'utilisateur, mais c'est au niveau des jambes que se repèrent les plus importantes erreurs (lorsque le gabarit choisi n'est pas le bon).

Le second protocole est le plus pertinent.

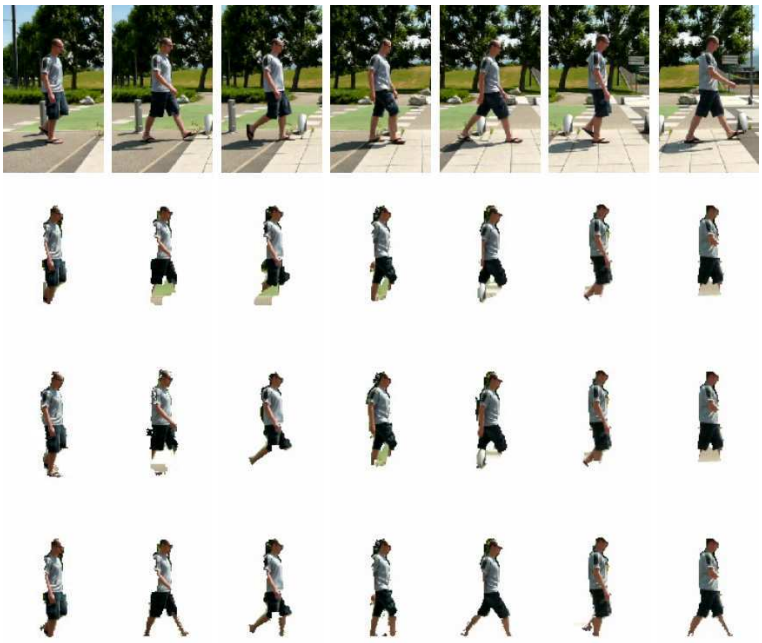

FIG. 6.11 – Effets du choix du mode de marquage interactif de l'utilisateur. De haut en bas : séquence initiale, segmentation sans interaction, segmentation avec marquage de la première *frame* et segmentation avec marquage sur une coupe spatio-temporelle au niveau des jambes.

6.1.4 Gabarit spatio-temporel

Les gabarits utilisés plus haut sont uniquement spatiaux. Ils sont appliqués à chaque *frame* d'une fenêtre. Mais le problème vient du fait que la posture change avec le temps et qu'un même gabarit spatial ne correspond pas forcement à une suite de *frames*. L'idée développée ici est de réaliser un gabarit spatio-temporel. Le gabarit, comme la séquence vidéo, est alors un objet 3D tel que défini en section 6.1.1.

Le gabarit spatial définit une allure de la posture de la personne. Un gabarit spatio-temporel définit une allure du déplacement de la personne. Comme le mouvement de la personne est bien caractéristique de la classe qu'il représente, le nouveau gabarit est d'autant plus pertinent.

Synchronisation du gabarit spatio-temporel avec la séquence

Comme elles pressentent la plus grande variation de posture, les jambes sont la partie la plus difficile à segmenter. Néanmoins, leur mouvement est caractéristique. Le gabarit spatio-temporel utilise ce mouvement pour guider la segmentation.
Le balancement des jambes donne un mouvement périodique. Un gabarit spatio-temporel est défini sur une période entière pour pouvoir représenter chaque phase de la marche (deux jambes rapprochées, genou levé, jambes écartées...). Pour une segmentation cohérente, la séquence vidéo et le gabarit doivent être alignés temporellement. C'est-à-dire qu'à une *frame* de la séquence est associé un gabarit spatial où les jambes sont dans une position correspondante (même phase du mouvement). Pour faire correspondre la séquence et le gabarit, il faut reconnaître, dans la séquence, la *frame* qui correspond au début du mouvement pour le gabarit et que appelons *frame* de synchronisation.

Nous choisissons comme position de synchronisation celle correspondant à l'instant où les deux jambes sont le plus rapprochées. En effet, quelles que soient la vitesse et l'amplitude du mouvement, cette position apparaît toujours. Nous avons alors collecté plusieurs *frames* correspondant à cette position. Puis nous avons réalisé la moyenne des gradients, obtenus par l'algorithme de Canny, de ces *frames* afin d'obtenir un modèle de référence.

Pour obtenir les positions de synchronisation dans la séquence testée, le gradient par l'algrithme de Canny de chaque *frame* de la séquence est calculé. Soit Δ_j une fenêtre de 34×54 pixels contenant les jambes de la personne. Soit, pour un pixel p, $G(p)$ le gradient de la frame et $G^r(p)$ la valeur du modèle de référence. Alors, pour chaque *frame*, la valeur V^d représentant la vraisemblance de la *frame* en tant que *frame* de synchronisation est définie par :

$$V^d = \underset{p \in \Delta_j}{\text{moyenne}} \left(G(p) G^r(p) \right) \tag{6.1}$$

Les *frames* recherchées sont celles donnant les maxima locaux de V^d. Pour éviter l'apparition d'une multitude de maxima locaux, l'évolution de V^d est lissée dans le temps. Les courbes obtenues (figure 6.12) montrent bien une évolution périodique. Les maxima sont facilement détectables et bien réguliers dans le temps. Ils correspondent bien aux *frames* recherchées.
Si la personne est de face (figure 6.12 (f)), alors les jambes de la personne semblent toujours rapprochées. L'évolution de V^d n'est donc pas pertinente. Sur la figure 6.12(f), l'évolution de V_d donne une période bien trop longue. Dans ce cas, un gabarit spatial fixe est utilisé pour l'ensemble de la séquence. La coupe peut donc se faire sans difficulté.
Il existe des méthodes efficaces pour déterminer la période du balancement des jambes [Polana and Nelson, 1994][Cutler and Davis, 2000][Ran et al., 2005]. Dans notre cas, il suffit de prendre le nombre de *frames* séparant deux *frames* successives détectées précédemment. Cette période donne aussi la vitesse de déplacement de la personne.

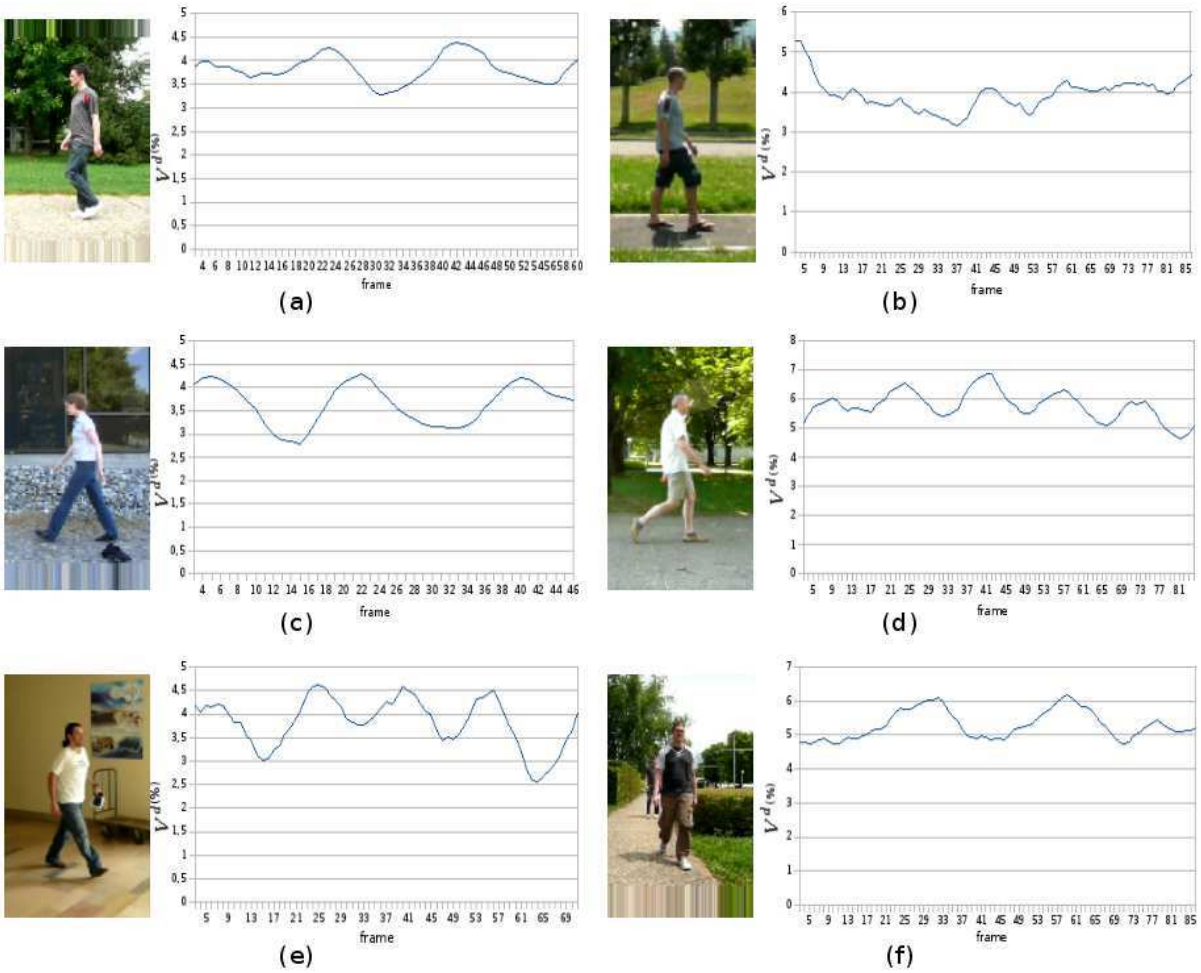

FIG. 6.12 – Évolution du paramètre V^d au cours du temps pour plusieurs séquences vidéo. Pour (a)(c)(d) et (e), le mouvement est bien périodique. Les maxima locaux sont réguliers et facilement repérables. Pour (b) l'évolution est plus chaotique. Néanmoins certains maxima plus importants correspondent bien aux *frames* recherchées. Pour (f), la personne est de face. La périodicité du mouvement des jambes n'est pas détectable. La période trouvée ne correspond pas à une période de la marche (elle est d'ailleurs trop longue).

Intégration du gabarit spatio-temporel dans le graphe

Nous formons un ensemble de 8 gabarits spatio-temporels correspondant aux différents angles de vue (figure 6.13). Nous créons nos gabarits sur une période de 18 *frames* (durée moyenne d'une période de marche observée sur un ensemble de séquences). Une interpolation linéaire 3D (somme pondérée des 8 pixels voisins spatio-temporellement) est ensuite faite pour qu'ils possèdent la même période que celle obtenue dans la séquence. Pour le cas de la vue de face, le gabarit unique statique est appliqué à chaque frame du gabarit spatio-temporel.

La réalisation et la résolution du graphe se réalisent comme avec la fenêtre glissante mais sur une période entière et avec un gabarit différent pour chaque *frame*.

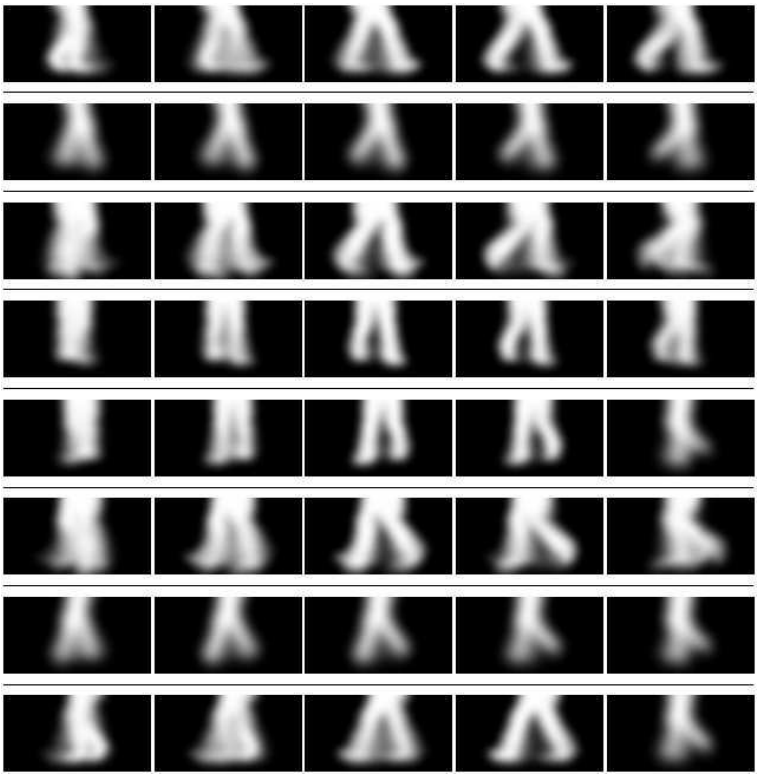

FIG. 6.13 – Recadrage au niveau des jambes (partie la plus importante) de 5 *frames* issues de nos 8 gabarits spatio-temporels. Ces gabarits modélisent la marche selon différents angles de vue.

Les segmentations obtenues suivent bien le mouvement présenté par le gabarit. Néanmoins, les artefacts et les flous provoqués par le mouvement des personnes perturbent l'utilisation des gradients pour la pondération des arêtes de voisinage. Les segmentations ne suivent alors pas parfaitement les contours. Plusieurs exemples de segmentation sur notre base de données de vidéos se trouvent sur la figure 6.14.

Nous prenons les mêmes séquences que celles utilisées pour l'évaluation des performances dans la section 6.1.3. Nous ne prenons en compte que 2 périodes (soit 36 *frames*) pour la séquence *fille* et une période (16, 20 et 18 *frames*) pour les autres. Les mesures quantitatives obtenues sont dans le tableau 6.3. La figure 6.15 montre que les performances avec le gabarit spatio-temporel sont proches de celles avec les fenêtres glissantes de la section 6.1.3. Néanmoins, comme il existe une transition entre les gabarits dans le temps, le rendu donne une continuité temporelle bien plus agréable et réaliste.

séqeunce	Fille	Thibaud	Luiz	Jonathan
$F_measure$ (↑)	0,7942	0,8178	0,8711	0,7714
Martin (↓)	0,0688	0,0607	0,0511	0,0634
Yasnoff (↓)	0,5878	0,4491	0,4112	0,6292

TAB. 6.3 – Évaluation de la segmentation avec un gabarit spatio temporel sur quatre séquences vidéo.

FIG. 6.14 – Segmentations obtenues par coupe de graphe avec un gabarit spatio-temporel sur 4 séquences vidéo.

6.1.5 Conclusion

Une séquence vidéo peut être représentée par un objet en trois dimensions, la troisième dimension étant le temps. Cette représentation permet une adaptation facile et cohérente du principe de la coupe de graphe pour les séquences vidéo. Le mouvement humain étant caractéristique il est tout d'abord possible de définir une interaction de l'utilisateur particulièrement efficace pour la classe des personnes. Nous avons vu que marquer une coupe spatio-temporelle au niveau des jambes permet une segmentation sans apprentissage plus performante.
Mais le but est d'intégrer notre connaissance de la classe dans le graphe. L'utilisation du gabarit n'est pas le même qu'avec les images fixes. Nous avons alors proposé deux méthodes. Dans la première, une fenêtre glissante temporellement permet de proposer un gabarit identique sur une petite suite de *frames*. La segmentation est alors assez discontinue temporellement. Dans la seconde méthode, le graphe utilise un gabarit spatio-temporel. La continuité temporelle est alors de qualité et la segmentation légèrement plus précise.
Nos méthodes donnent des segmentations de moins bonne qualité que sur de images fixes. Cela vient en partie du fait que notre base de données de vidéos possède des artefacts et des effets de flou provoqués par le mouvement. La recherche des contours est donc moins aisée. De plus, travailler sur des séquences augmente fortement le nombre de pixels (multiplié par le nombre de *frame*). Le nombre de noeuds et surtout d'arêtes augmente donc singulièrement ce qui altère le fonctionnement de la recherche de la coupe minimale.

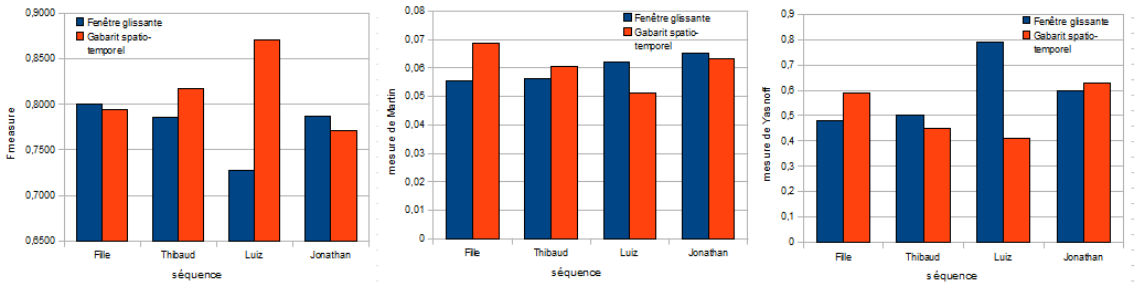

FIG. 6.15 – Comparaison des segmentations obtenues avec une fenêtre glissante et avec un gabarit spatio-temporel par trois critères de quantification sur quatre séquences vidéo. Sur la séquence "'Luiz"', la segmentation est bien meilleure avec avec un gabarit spatio-temporel. Sinon les résultats sont très proches.

6.1.6 Segmentation adaptative à l'echelle

Comme nous l'avons fait sur les images fixes en section 5.6.1, nous allons maintenant étudier le cas des images de taille quelconque.

Avec une fenêtre glissante

Soit une séquence vidéo $\mathcal{S}$ de n_f *frames* contenant une personne. La première étape est de détecter la personne sur l'ensemble des *frames*. Ceci peut être fait par une détection automatique [Dalal et al., 2006] ou par une détection manuelle par l'utilisateur. Il y a alors une fenêtre de détection de taille $l_i^f \times h_i^f$ pour chaque *frame* f. Chacune de ces n_f fenêtres de détection est alors recadrée au format 96×160 pixels. L'ensemble de ces fenêtres redimensionnées forme la séquence recadrée $\mathcal{S}^r$.
On se retrouve alors dans le même cas que précédemment. La méthode de la fenêtre glissante permet d'obtenir la segmentation $\mathcal{S}_s^r$
$\mathcal{S}_s^r$ forme un masque binaire. Une interpolation linéaire permet de redimensionner chaque *frame* f de $\mathcal{S}^r$ au format $l_i^f \times h_i^f$. Puis un filtrage gaussien permet d'obtenir un gabarit non binaire qui donne le gabarit de la frame entière. Une dernière coupe de graphe est réalisée sur la séquence $\mathcal{S}$ entière avec ce gabarit pour obtenir la segmentation finale (figure 6.16).

Avec un gabarit spatio-temporel

La segmentation de la séquence $\mathcal{S}$ peut aussi se réaliser de façon similaire mais en segmentant $\mathcal{S}^r$ avec un gabarit spatio-temporel comme en section 6.1.4.

Résultats

Comme pour les images, itérer la coupe de graphe à la taille réelle augmente la précision de la segmentation. Mais réaliser une coupe sur la séquence entière l'améliore aussi.
Étudier une séquence dans son ensemble plutôt qu'une *frame* à la fois permet d'améliorer la continuité temporelle de la segmentation. Réaliser une nouvelle coupe de graphe sur la séquence $\mathcal{S}$ entière réduit de même toute les discontinuités temporelles provoquées par les transitions entre deux fenêtres glissantes (pour la première méthode) ou périodes (pour la seconde méthode). Le rendu est alors bien plus agréable.

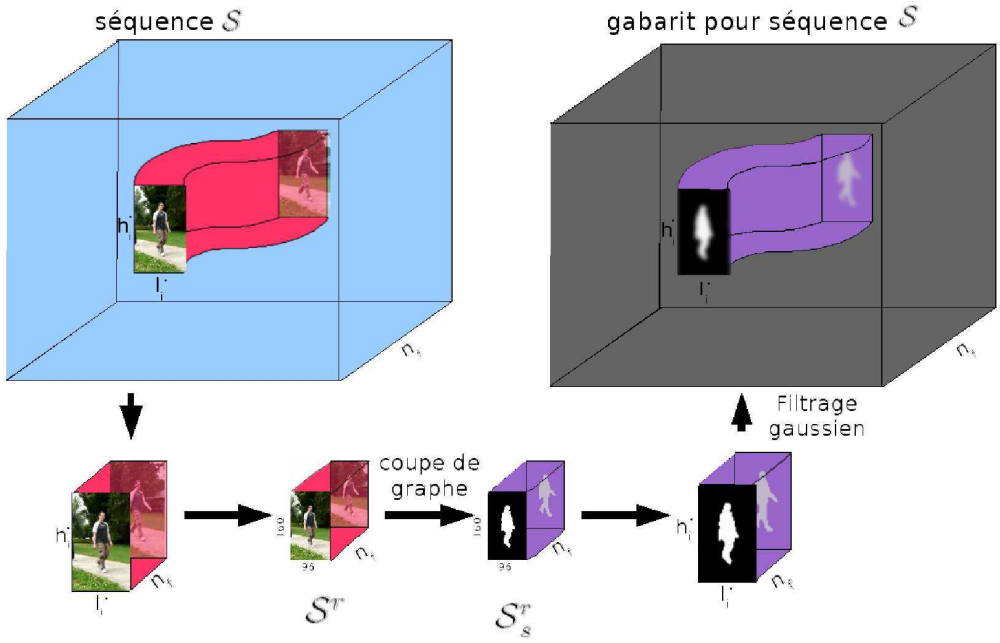

FIG. 6.16 – Processus de segmentation : la séquence $\mathcal{S}$ est d'abord recadrée et redimensionnée pour former $\mathcal{S}^r$. Une coupe de graphe réalise la segmentation de $\mathcal{S}^r$. Cette segmentation est redimensionnée puis passée par un filtrage gaussien. Elle est enfin recadrée pour former le gabarit de $\mathcal{S}$ pour une dernière coupe de graphe.

Dans $\mathcal{S}^r$, la personne est toujours centrée. C'est l'arrière plan qui se déplace. Or c'est le mouvement du premier plan qui est important et dont la continuité doit être assurée par l'étude spatio-temporelle. La coupe de graphe sur la séquence $\mathcal{S}$ est donc bien pertinente.
Néanmoins, le temps de traitement augmente car le graphe sur $\mathcal{S}$ est très grand. De plus, la coupe de graphe assurant la continuité, une erreur de segmentation se répercute dans la suite de la séquence. Enfin, si la personne effectue des mouvements trop rapides, la discontinuité déjà évoquée peut provoquer des erreurs de segmentation.

6.2 Recherche d'un trimap précis

Le **matting** est une variation de la segmentation qui calcule l'opacité α du premier plan. En effet, la couleur d'un pixel peut être un mélange de celle du premier et de l'arrière plan. C'est le cas avec des objets transparents, semi-transparents ou en mouvement (provoquant un effet de flou) ainsi qu'au niveau des transitions entre deux objets. Soit, pour un pixel, I la couleur de l'image, F celle du premier plan et B celle de l'arrière plan. L'expression du matting est alors définie par l'équation :

$$I = \alpha.F + (1 - \alpha)B \tag{6.2}$$

Dans certaines applications (photo-montage), le matting permet un rendu plus réaliste (figure 6.18(d)). La segmentation est un matting avec un α binaire.

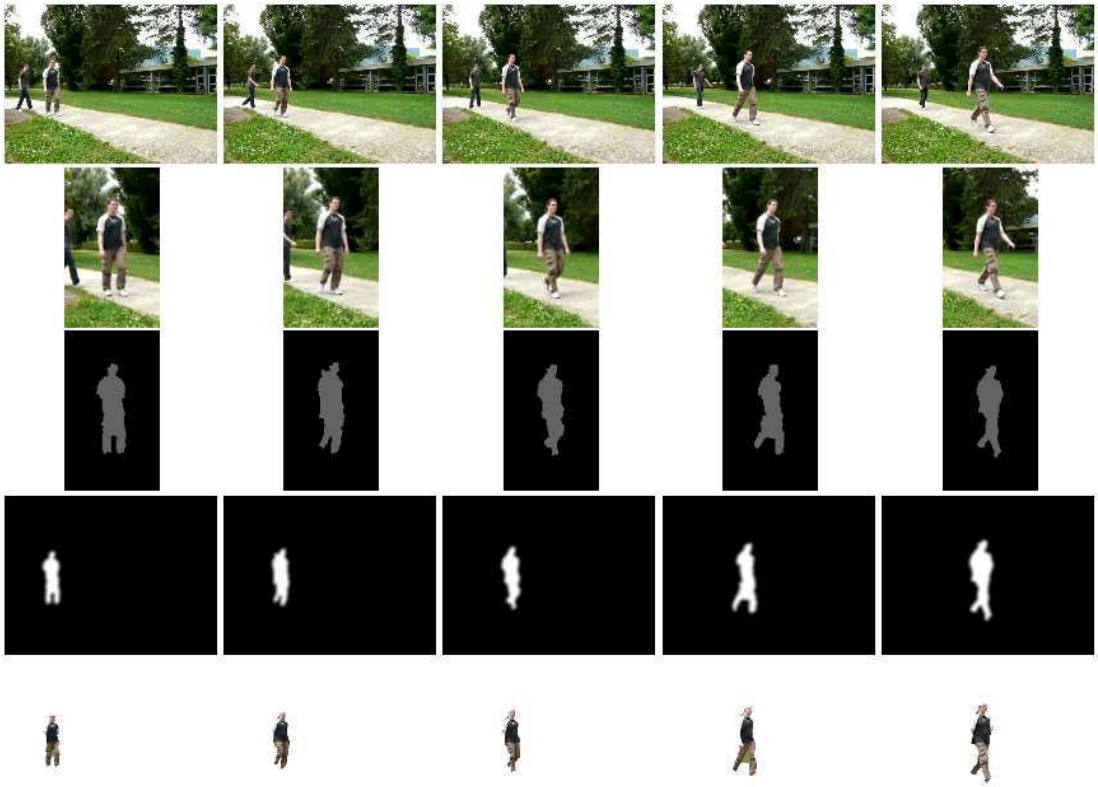

FIG. 6.17 – Exemple de segmentation d'une séquence vidéo. Première ligne : *frames* de la séquence $\mathcal{S}$. Seconde ligne *frames* recadrées correspondantes de la séquence $\mathcal{S}^r$. Troisième ligne : segmentation des *frames* de $\mathcal{S}^r$. Quatrième ligne : gabarit des *frames* de $\mathcal{S}$. Dernière ligne : segmentation des *frames* de $\mathcal{S}$.

Un **trimap** est une vue d'une image dans un processus de segmentation ou de matting. Il permet à l'utilisateur de donner l'information sur ce qu'il veut segmenter dans l'image. Un trimap est composé de trois régions. La région Ω_F correspond aux pixels qui appartiennent avec certitude au premier plan. La région Ω_B correspond aux pixels qui appartiennent avec certitude à l'arrière plan. Enfin Ω_U est la région inconnue dont l'appartenance au premier ou à l'arrière plan des pixels n'est pas déterminée (figure 6.18(b)). Le travail à réaliser est de déterminer l'assignation ou la valeur du canal α des pixels de Ω_U. Cela est notamment assez souvent réalisé par apprentissage du premier et de l'arrière plan à partir de Ω_F et Ω_B.

Pour des images de personnes en haute résolution, un photo-montage est bien plus réaliste si un matting est réalisé plutôt qu'une simple segmentation. Il existe plusieurs méthodes de matting efficaces mais celles-ci ne sont pas guidées par la connaissance d'une classe. Elles nécessitent l'aide d'un trimap réalisé par l'utilisateur.
Nous proposons ici une méthode réalisant automatiquement un trimap pertinent de personnes. C'est-à-dire un trimap qui contienne les informations de la segmentation grossière tout en gérant l'incertitude aux contours. Il s'agit en réalité de la méthode duale à celle de la segmentation puisque les contours ne permettent pas la séparation mais sont la partie à segmenter.

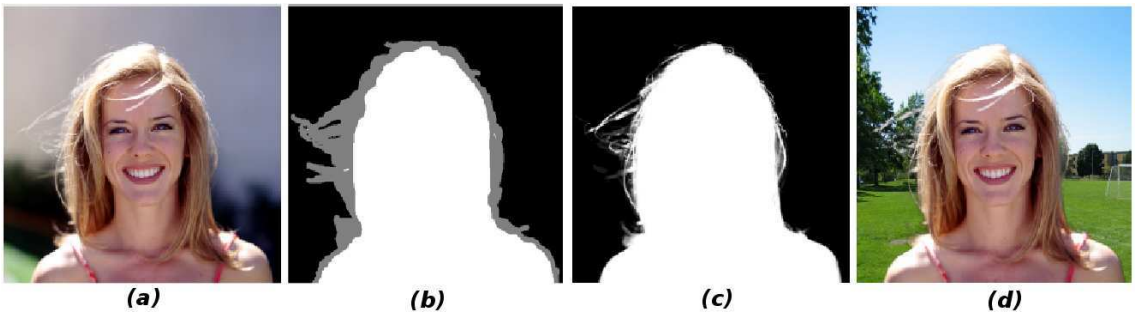

FIG. 6.18 – Exemple de *matting* : à partir de l'image originale (a), un trimap est réalisé (b). La région blanche représente Ω_F, la région grise Ω_U et la région noire Ω_B. Une méthode de *matting* bayésien [Wang et al., 2007b] permet d'obtenir le masque alpha (c) où chaque pixel possède une luminance proportionelle à la valeur de α telle que définie par l'équation 6.2. Un photo-montage peut alors être réalisé à partir de ce masque (d).

6.2.1 Recherche automatique de Trimap

La réalisation manuelle des trimaps est notamment inconcevable pour les séquences vidéo. En effet, réaliser un trimap pour chaque *frame* prend un temps très important alors que le contenu de *frames* voisines est fortement corrélé. Il est alors courant de demander à l'utilisateur le trimap de certaines *frames* clés puis de propager ceux-ci sur les *frames* voisines. Cette propagation spatio-temporelle est réalisée en fonction des caractéristiques des *frames* afin de rendre les trimaps les mieux adaptés. Guillemaut et al. [Guillemaut et al., 2010] modélisent chaque région du trimap par des modèles de mélange de Gaussiennes (GMM), qui sont calculées à partir des trimaps déterminés des *frames* voisines, pour déterminer la région de chaque pixel d'une *frame*. Chuang et al. [Chuang et al., 2002] propagent le trimap en fonction du flot optique de la séquence.
Pour obtenir un trimap automatiquement, il est aussi possible d'user d'appareillages spéciaux. Par exemple, Hyun et Al. [Hyun et al., 2008] prennent plusieurs prises légèrement décalées de la scène à segmenter. Ils réalisent leur trimap en seuillant la variance des images obtenues par ces prises. L'arrière plan est en effet bien plus sensible au changement lors des différentes prises de vue. Les régions Ω_F et Ω_B peuvent aussi être évaluées en séparant les objets de profondeurs différentes (distants différemment de l'objectif) grâce à un appareillage stéréoscopique [Zitnick et al., 2004] ou par des prises avec des focales différentes [McGuire et al., 2005]. La région intermédiaire forme la région Ω_U.

La génération automatique du trimap d'une image fixe simple est un problème difficile et peu traité. Juan [Juan and Keriven, 2005][Juan, 2006] propose une méthode à partir d'un Graph Cut [Rother et al., 2004]. Une segmentation rapide est d'abord réalisée à partir de pixels rapidement assignés au premier et à l'arrière plan par l'utilisateur. Les arêtes de voisinage entre deux pixels sont pondérées par l'inverse de la distance spatiale séparant les deux pixels. Les arêtes de liaisons sont calculées à partir des modèles de mélanges de gaussiennes des trois régions. Les gaussiennes relatives au premier et à l'arrière plan sont obtenues à partir de la première segmentation. On note, pour la gaussienne i relative à la région X, μ_i^X la moyenne et Σ_i^X la covariance. Comme la région inconnue est fortement corrélée au deux autres régions par la relation 6.2, les gaussiennes

qui lui sont relatives sont formées à partir des associations entre gaussiennes du premier et de l'arrière plan. La gaussienne mélangeant la gaussienne i du premier plan avec la gaussienne j de l'arrière plan a les caractéristiques suivantes :

$$\mu_{ij}^{U} = \frac{\mu_i^F + \mu_j^B}{2} \tag{6.3}$$

$$\Sigma_{ij}^{U} = \frac{1}{3}(\Sigma_i^F + \Sigma_j^B) + \frac{1}{12}(\mu_j^B - \mu_i^F)(\mu_j^B - \mu_i^F)^T \tag{6.4}$$

Deux noeuds sont alors nécessaires pour chaque pixel afin de permettre l'étude des trois régions. La coupe de graphe donne alors directement le trimap recherché. Rhemann et al.[Rhemann et al., 2008] donnent une version très proche en rajoutant dans l'expression de l'énergie du graphe des termes permettant notamment de gérer l'épaisseur de Ω_U ou de s'adapter à certaines particularités de l'image.

6.2.2 Coupe de graphe duale

Pour obtenir un trimap, nous réalisons une nouvelle coupe de façon duale à celle présentée dans le chapitre 5. En effet, les contours ne sont plus les séparateurs entre les deux régions mais une des régions. Il faut donc adapter la méthode.
Dans le nouveau graphe, la source représente donc les pixels de la région Ω_U et le puits représente les pixels des régions Ω_F et Ω_B.

Arêtes de liaison

Un pixel a de fortes probabilités d'être dans la région incertaine Ω_U si il représente une séparation entre premier et arrière plan et donc si il représente un contour. Pour un pixel i soit $\bigtriangledown_i$ le gradient normalisé de l'image au niveau de ce pixel. Alors le poids de l'arête de liaison reliant i au premier plan est :

$$\omega_i^F = -\alpha.ln(\bigtriangledown_i) \tag{6.5}$$

Il faut, de plus, que les contours autres que ceux correspondant à la séparation premier/arrière plan ne soit pas sélectionnés. Cette information apparaît sur les gabarits dont nous parlerons plus loin. Pour un pixel i, soit t_i la valeur de ce gabarit (comparable à une probabilité d'appartenir à un contour) pour ce pixel, alors le poids de l'arête de liaison reliant i à l'arrière plan est :

$$\omega_i^B = -\beta.ln(1 - t_i) \tag{6.6}$$

Arêtes de voisinage

Pour obtenir un trimap performant, il faut, dans les régions plus certaines, que la séparation inconnue entre premier et arrière plan soit la plus étroite possible. Ainsi, si la transition entre ces deux régions est franche, alors la zone inconnue est étroite et inversement. Soit Δ_i le laplacien de l'image au pixel i, alors le poids associé à une arête de voisinage partant de i vers j est :

$$\omega_{ij} = \gamma e^{-\frac{\Delta_i^2}{2\sigma^2}} \tag{6.7}$$

Notons que α, β, γ et σ sont les paramètres qui permettent de gérer l'influence de chaque poids dans le graphe.

Fermeture de Ω_U

La coupe de graphe permet d'obtenir la région correspondant à la région inconnue Ω_U. Cette région doit séparer Ω_F et Ω_B de telle façon qu'elles soient deux composantes connexes distinctes. Ces deux régions ne doivent pas être contigues en aucun endroit. Il faut donc que Ω_U soit fermée.
Plus β est élevé plus le nombre de pixels de Ω_U est important (ce qui est logique puisque β représente l'influence du gabarit et que plus il est élevé plus le gabarit prend de l'importance face au gradient). Ainsi, des coupes de graphe sont réalisées avec des valeurs croissantes de β (figure 6.19), jusqu'à ce que Ω_U soit fermée.
On cherche ensuite à affiner Ω_U. Soit N_β le nombre de valeurs de β nécessaires pour fermer Ω_U. Soit, pour un pixel, $\delta \in [0, N_\beta]$ le nombre de fois où ce pixel à fait partie de Ω_U dans les N_β itérations précédentes. Ω_U est décomposée en régions connexes de même valeur δ. Pour un indice i allant de 0 à N_β, toute région vérifiant $\delta = i$ et n'étant pas nécessaire à la fermeture de Ω_U est supprimée de Ω_U (figure 6.20). Ainsi la région fermée est la plus fine possible.

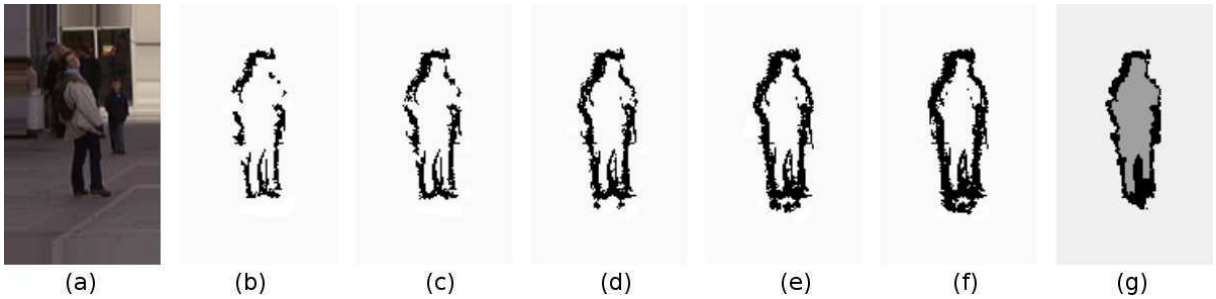

FIG. 6.19 – Plusieurs valeurs de β sont utilisées pour fermer la région Ω_U. À partir de l'image initiale (a) on réalise une coupe de graphe avec $\alpha = 90$, $\gamma = 10$, $\sigma = 7$ et $\beta = 300$ (b) puis $\beta = 350$ (c) puis $\beta = 400$ (d) puis $\beta = 450$ (e) puis $\beta = 500$ (f) pour réaliser le trimap (g).

FIG. 6.20 – Exemple de région Ω_U affinée : par δ croissant, les régions sont supprimées si la fermeture de Ω_U est conserevée.

Utilisation des gabarits

On a vu que la pondération des arêtes de liaisons vers l'arrière plan se base sur un gabarit. Comme dans le chapitre 5, on introduit un gabarit unique basé sur une position moyenne de la personne et un gabarit par parties adapté à la posture de la personne. Le

gabarit unique est visible sur la première image de la figure 6.21. Il a été réalisé par une moyenne des contours sur une base de données de 400 silhouettes humaines.
Les images de la figure 6.21 monttrent les différents sous-gabarits que nous avons introduits pour les jambes. Le gabarit par parties est la concaténation des sous-gabarits décrivant le mieux la posture de la personne.

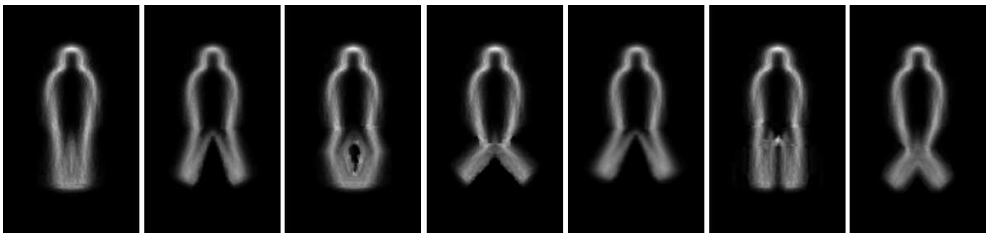

FIG. 6.21 – Des gabarits permettent de définir pour plusieurs postures la probabilité d'un pixel de séparer le premier de l'arrière plan et donc de définir la région Ω_U. Pour le gabarit unique on prend le gabarit de gauche. Les autres gabarits donnent les sous-gabarits au niveau des jambes.

6.2.3 Résultats et conclusion

FIG. 6.22 – Exemples de trimaps obtenus par notre méthode.

Un bon trimap possède une région inconnue Ω_U petite avec une fiabilité sur les régions Ω_F et Ω_B élevée. On évalue donc la performance de la méthode en comparant ces deux notions.
Soit N le nombre de pixels de l'image et Ω_{FGT} l'ensemble des pixels du premier plan dans la vérité terrain obtenue manuellement. On définit tout d'abord le *taux d'incertitude* qui correspond à la proportion de pixels incertains :

$$\tau_i = \frac{card\{i|i \in \Omega_U\}}{N} \tag{6.8}$$

On définit ensuite le *taux d'erreur* comme le pourcentage de pixels de F et B mal attribués :

$$\tau_\varepsilon = \frac{card\{i|i \in (\Omega_F \cap \overline{GT}) \cup (\Omega_B \cap GT)\}}{card\{i|i \in \Omega_F \cup \Omega_B\}} \tag{6.9}$$

Ces deux valeurs sont liées. Si τ_i est élevé, Ω_F et Ω_B sont plus petits. Donc il y a moins d'erreurs et τ_ε est plus faible. On joue donc sur les différents paramètres pour tracer le graphe $\tau_\varepsilon = f(\tau_i)$. Les courbes les plus proches des axes sont celles qui représentent les meilleures performances.

Nous comparons les résultats obtenus par trois méthodes de génération de *trimap* :

- (M1) Une silhouette est calculée par seuillage de la moyenne d'une base de données de silhouettes humaines. Le trimap est obtenu par dilatation du contour de cette silhouette. Cette solution ne prends pas en compte les caractéristiques de l'image testée.

- (M2) Le contour du masque de segmentation obtenu par une coupe de graphe (section 5.3) est dilaté.

- (M3) Notre nouvelle méthode présentée dans la section 6.2.2.

La figure 6.23 montre que, logiquement, la méthode (M1) est la moins efficace, et que les deux autres méthodes sont plus ou moins efficaces selon l'attente. Pour avoir une forte précision, notre méthode (M3) est la plus efficace (ce qui est logique puisqu'elle s'adapte aux caractéristiques de l'image) tandis que la méthode (M2) est plus efficace pour obtenir une petite région Ω_U (ce qui s'explique du fait que Ω_U suit le contour par la segmentation).

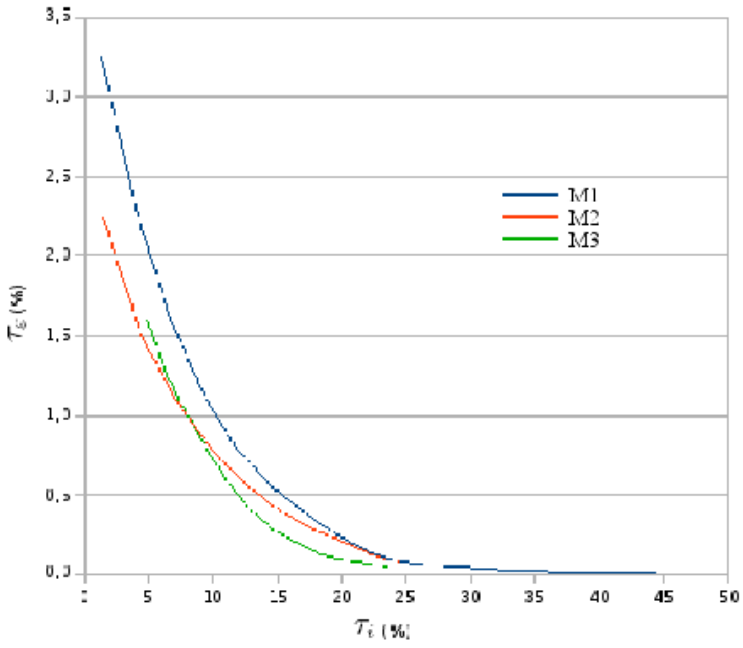

FIG. 6.23 – Courbes *taux d'erreur* en fonction du *taux d'incertitude* pour une dilatation d'une segmentation moyenne non adaptée à l'image (M1), pour une dilatation de la segmentation (M2) et enfin par notre nouvelle méthode (M3). La méthode M1 est logiquement inférieure. La méthode M2 favorise la finesse de Ω_U alors que la méthode M3 favorise la précision.

La figure 6.24 montre que le gabarit unique et le gabarit par parties donnent des résultats similaires. Une idée est de tester une dilatation du Ω_U obtenue par notre méthode (M3) pour vérifier que la finesse de Ω_U ne contraigne pas à des transitions trop rapides. Au contraire, la figure 6.25 montre que la dilatation dégrade la réalisation du trimap.

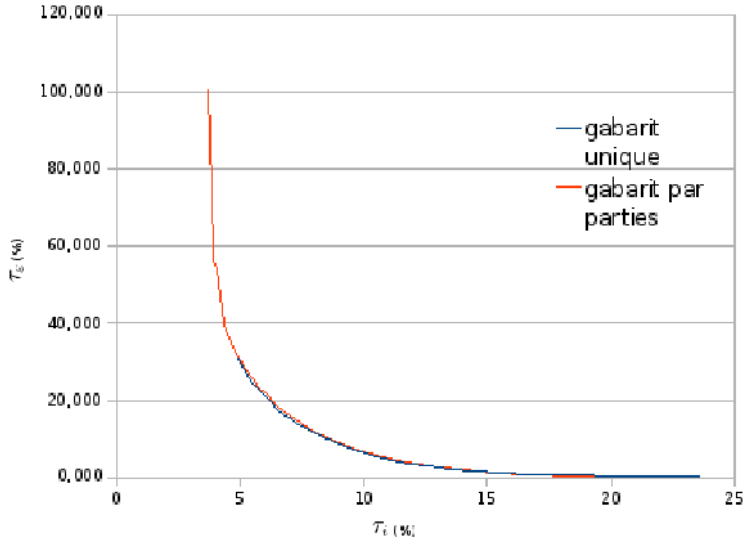

FIG. 6.24 – Courbes *taux d'erreur* en fonction du *taux d'incertitude* avec un gabarit unique et avec un gabarit par parties. Les résultats sont très semblables.

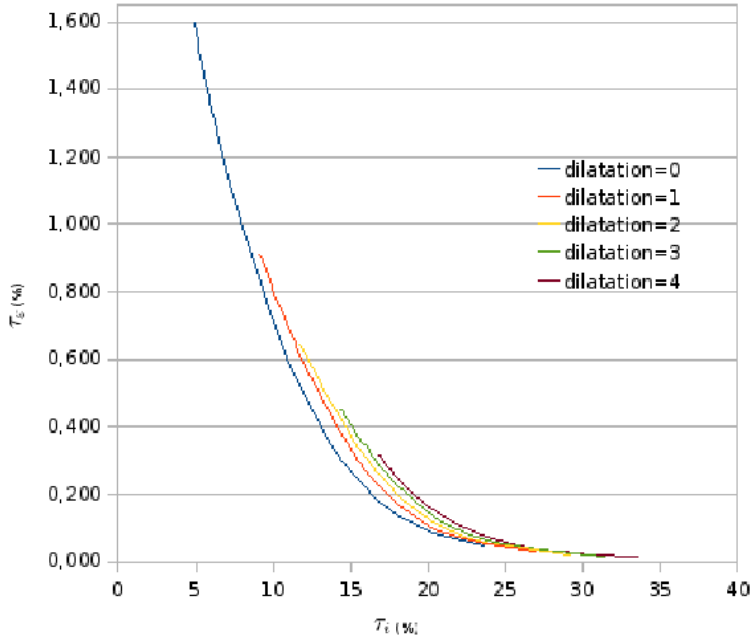

FIG. 6.25 – Courbes *taux d'erreur* en fonction du *taux d'incertitude* en dilatant la région Ω_U. La dilatation dégrade les performances.

La méthode de réalisation automatique de trimaps que nous venons de présenter donne des résultats intéressants pour obtenir des taux d'incertitude faible. Néanmoins, la méthode ne s'améliore presque pas lorsque la posture est déterminée par un gabarit par parties. Elle atteint donc rapidement ses limites. Le problème réside dans le fait que la coupe de graphe est efficace pour la segmentation d'objets concaves et compacts. Le contour étant une région longiligne, elle n'est pas la mieux adaptée au procédé. De plus, l'usage du gabarit donne juste une allure qui doit guider la séparation tout en la laissant bien le contour s'adapter aux caractéristiques de l'image. Une région longiligne n'est donc ici pas non plus la mieux adaptée. Il faudrait alors réaliser la coupe avec un nombre très important de gabarits donnant précisément l'ensemble des postures possibles. Cela demanderait alors un temps de traitement très important.

Chapitre 7

Conclusion

> Il faut faire de la vie un rêve et faire d'un rêve une réalité.
>
> Pierre Curie

Sommaire

Dans ce document, nous avons traité de la segmentation des personnes dans les images et les vidéos. Généralement, la détection est associée à la segmentation. Les deux processus sont alors souvent réalisés simultanément. Nous avons pris le parti de concentrer notre étude sur la segmentation. Dans cette optique, nous avons mis au point deux algorithmes fonctionnant à partir de la détection mais selon des conditions différentes :

- La première méthode réalise un traitement automatique. C'est-à-dire que l'utilisateur n'intervient en aucune façon dans le déroulement du calcul. Cela permet de réduire au minimum le travail de l'utilisateur mais rend le logiciel peu adaptable.

- La seconde méthode réalise un traitement interactif. C'est-à-dire que l'algorithme prend en compte les indications données par l'utilisateur pour réaliser le traitement. Si l'utilisateur a un travail plus important à réaliser, il possède néanmoins une meilleure emprise sur le résultat. Nous avons alors optimisé l'interaction de l'utilisateur pour qu'elle soit la plus simple et la plus intuitive possible, et non pas seulement accessible aux spécialistes du traitement de l'image. Seuls des marquages à la souris sont à réaliser sur des images s'interprétant facilement.

Ces deux méthodes fournissent finalement un masque qui sépare le premier plan recherché de l'arrière plan. Une intégration sur un nouvel arrière plan est alors possible pour réaliser un photo (ou vidéo) montage.

7.1 Principales contributions

Notre travail a apporté plusieurs innovations qui peuvent être reprises et permettre de nouvelles avancées :

- Une méthode originale de pré-segmentation réalisant une recherche d'information locale sur une fenêtre de détection a été développée afin d'utiliser les données de la détection à une échelle plus réduite. Ainsi, des informations concernant la détection spatiale des parties de la silhouette humaine dans la fenêtre de détection sont obtenues. Cette pré-segmentation est expliquée dans le chapitre 3. Ces données visent à permettre la segmentation et s'adaptent parfaitement à la détection puisque les outils utilisés sont les mêmes.

- Un algorithme classique de recherche de plus court chemin a été adapté dans le chapitre 4 pour obtenir la segmentation d'une personne dans une fenêtre de détection positive à partir des données calculées lors de la pré-segmentation. Le principe combine l'utilisation des segments de contours avec la pré-segmentation pour rechercher, dans un graphe, le chemin correspondant à la succession de segments qui permet de reconstruire la silhouette de la personne.

- Une méthode originale de segmentation de personnes par utilisation de coupe de graphe a été réalisée. Un nouveau type de gabarit non binaire a été développé pour introduire dans le graphe notre connaissance de la classe.

- Une méthode pour déterminer le gabarit adaptés à la silhouette recherchée, que nous appelons gabarits par parties, a permis une segmentation plus efficace sans trop augmenter le temps de calcul.

- Deux adaptations de la méthode du point précédent en utilisant une fenêtre glissante puis en utilisant un gabarit spatio-temporel ont étendues l'étude au cas des séquences vidéos.

- L'interaction de l'utilisateur sur une séquence vidéo pour initialiser une coupe de graphe par l'intermédiaire du marquage d'une coupe spatio-temporelle au niveau des jambes a été testée et a donné de bonnes performances.

- Une méthode de génération automatique de *trimap* par utilisation de coupe de graphe a égallement été developpée et testée.

- Enfin, pour l'apprentissage ainsi que pour les tests nécessaires aux méthodes citées plus haut, nous avons réalisé deux bases de données (annexes A et B). L'une contient des images de silhouettes de personnes. L'autre contient des séquences vidéo de personnes marchant. Dans les deux cas, le format de l'image ou de la *frame* et la taille de la personne qu'elles contiennent sont identiques pour correspondre à la base de données de personnes de l'INRIA. En effet, cette base de données est très couramment utilisée. Fournir d'autres bases de données selon un format similaire permet de réaliser des comparaisons entre procédés. Ces nouvelles bases de données sont libres et peuvent donc être réutilisées pour d'autres travaux et d'autres recherches.

7.2 Contexte et limites

Pour apprécier une méthode, il faut connaître le contexte dans lequel on peut l'utiliser et les conditions nécessaires à une efficacité maximale. Dans la plupart des cas, un compromis doit en effet être fait entre la précision et la robustesse. Il faut bien expliciter le choix qui a été pris. Celui-ci définit les principales limites de la méthode.
Tout d'abord, les méthodes sont faites pour la segmentation de la seule classe des personnes. Elles pourraient être utilisées pour la segmentation d'autres classes comme par exemple celles d'animaux ou de véhicules. Néanmoins, dans notre étude, seule la forme est prise en compte car elle seule est bien représentative de la classe des personnes. Mais pour les classes nommées plus haut, la forme n'est pas la meilleure caractéristique. La couleur et la texture peuvent cette fois aider à la reconnaissance car elles ne varient que peu dans ces classes. Ainsi, notre méthode automatique n'est pas la mieux appropriée pour d'autres classes. De même, le mouvement des voitures est rectiligne et donc loin d'être caractéristique. Ces informations ne peuvent donc pas être prises en compte.
Au niveau de notre première méthode, il est à noter que l'utilisation des contours pose un problème certain. Le traitement dépend étroitement de la détection de contours. Une détection trop fine trouve un nombre trop important de contours qui empêchent la reconnaissance. Cependant, dans le cas contraire, la méthode est assez sensible au problème de camouflage. Si la rupture de couleur entre le premier et l'arrière plan n'est pas assez prononcée, le contour est absent et altère le traitement.
Du point de vue de l'implémentation de la pré-segmentation, bien que les vecteurs de caractéristiques soient de dimension bien inférieure à ceux utilisés pour la détection, le nombre de SVMs (et donc de modèles liés) est assez élevé. La complexité de l'algorithme est alors assez important. De plus, sur l'ensemble des méthodes, nous avons introduit de nombreux paramètres. Il n'est alors pas évident de régler toutes ces inconnues pour obtenir la combinaison de paramètres qui donnent les meilleurs performances. Cependant, des tests sur un certain nombre de ces variables isolées nous permettent d'avancer certaines valeurs qui donnent de bons résultats de façon suffisamment stable.
Enfin, pour la seconde méthode, l'algorithme de la coupe de graphe est moins efficace lorsque le nombre de noeuds du graphe est trop important. La taille de l'objet étudié est donc limitée (en particulier avec les séquences vidéo). Le nombre de gabarits est lui-aussi limité pour des raisons de complexité et de temps de calcul. Les positions les plus inhabituelles d'une silhouette (par exemple lors d'une extension dans un sport) ne peuvent pas être traitées convenablement.

7.3 Perspectives

Les méthodes que nous avons développées durant cette thèse répondent à certains choix et à certaines contraintes. Les modifier peut ouvrir des perspectives :

- **Des méthodes plus récentes** : pour réaliser nos traitements, nous avons utilisé un certain nombre de méthodes existantes. Le choix de ces méthodes s'est effectué assez tôt et nous avons décidé de ne pas revenir ensuite dessus. Néanmoins, l'utilisation de méthodes présentées plus récemment pourrait améliorer nos résultats.

De nouvelles méthodes de détection de personnes ont vu le jour. Certaines développent de nouvelles idées comme l'adaptation de l'algorithme à la structure de la scène [Joshi and Porikli, 2010] [Wang and Wang, 2011], d'autres améliorent d'anciennes méthodes (par exemple [Tosato et al., 2010] améliore la méthode de [Tuzel et al., 2007]). L'association HOG-SVM que nous utilisons dans le chapitre 3 a, elle aussi, été améliorée [Leibe et al., 2008] [Pang et al., 2011] [Wang et al., 2009]. Pour la détection des contours, nous avons choisi l'algorithme de Canny [Canny, 1986] couramment utilisé mais qui est maintenant surpassé [Hariharan et al., 2011].
Dans la section 3.2.2, nous avons vu que l'utilisation du flot optique ne permettait pas d'amélioration du descripteur. Il pourrait être intéressant de le vérifier avec des méthodes d'évaluation du flot optique plus récentes [Adato et al., 2011] [Jia et al., 2011]. Enfin, dans les chapitres 5 et 6, nous utilisons la méthode de coupe de graphe définie par [Boykov and Jolly, 2001]. Les coupes de graphe définies par [Komodakis and Tziritas, 2007] ou [Gorelick et al., 2011] donneraient peut-être un traitement plus rapide.

- **De nouveaux choix** : nous avons effectué dans le chapitre 3 des détections sur les blocs plutôt que sur les fenêtres de détection afin de parvenir à une pré-segmentation. Ces détections ont été réalisées en utilisant des histogrammes de gradients orientés comme descripteur et des machines à vecteurs de support comme classifieur. Ce choix s'est basé sur l'efficacité du processus de détection testé par Dalal. Nous avons néanmoins montré dans l'état de l'art qu'il existait un grand nombre de méthodes de détection. Des tests comparatifs de ces méthodes pour la détection au niveau des blocs pourrait déterminer laquelle est la plus performante pour réaliser la pré-segmentation.
  Ensuite, dans le chapitre 4, nous avons développé une méthode de recherche du plus court chemin qui permet d'utiliser la pré-segmentation pour la segmentation. Notre méthode a l'inconvénient de modéliser la silhouette par une suite de segments. Une nouvelle méthode pourrait utiliser les informations de notre pré-segmentation d'une façon permettant d'obtenir des courbes mieux arrondies. Cela pourrait se faire, par exemple, par une intégration efficace de ces données au niveau des arêtes de voisinage pour une coupe de graphe ou bien par une recherche de plus court chemin dans un graphe dont les nœuds ne seraient plus associés à des segments de contour mais à des parcelles de contour courbes (dans ce cas il faudra déterminer de nouveaux critères pour quantifier la probabilité que deux de ces éléments se suivent et assurer la continuité du contour).

- **Des sytèmes d'acquisition particuliers** : les méthodes que nous avons développées réalisent la segmentation en se basant sur les résultats d'une détection préalable. Nous avons choisi de concentrer notre étude sur des images et des vidéos obtenues par des systèmes d'acquisition classiques afin de rester accessibles à tous les usages. Or le temps de traitement des méthodes de détection dans ce cas de figure est assez important (il peut prendre plusieurs secondes pour une image). Il est souvent demandé de minimiser ce temps de traitement. Il peut être alors intéressant de tester des modes d'acquisition particuliers.
  Lorsque l'arrière plan est connu (par exemple dans le cas d'une vidéo prise par une caméra fixe), des régions d'intérêt sont obtenues par soustraction de l'arrière plan

à l'image. Ces régions d'intérêt sont aussi facilement obtenues par une acquisition thermographique ou stéréoscopique. Limiter le nombre de régions d'intérêt réduit considérablement le temps de traitement de la détection. Mais ces procédés apportent bien plus d'informations qui peuvent être intégrées à nos deux méthodes pour améliorer leur performance. Par exemple le fait de connaître une bonne partie de l'arrière plan élimine nombre d'erreurs possibles.
La stéréoscopie permet également d'obtenir la profondeur, c'est-à-dire la distance séparant les éléments de la scène de l'objectif. Nous avons déjà guidé une coupe de graphe par un gabarit 2D (chapitre 5) puis par un gabarit 3D prenant le temps comme troisième dimension (section 6.1.4). Avec la stéréoscopie, il serait possible de guider la coupe de graphe par un gabarit 3D prenant la profondeur comme troisième dimension.

- **Un algorithme chaîné** : dans la coupe de graphe guidée par un gabarit, un compromis est réalisé entre l'importance donnée au gabarit et celle donnée aux contours de l'image. En effet, comme le gabarit n'est qu'une allure de la silhouette, ce sont les contours qui délimitent véritablement la segmentation. Mais comme la méthode donne la posture de la personne, il serait possible de réaliser de nouvelles itérations de la coupe de graphe pour les sous-parties de la silhouette avec différents gabarits plus précis correspondant à la posture trouvée. Par exemple seraient proposées des gabarits de la tête avec plusieurs types de coiffure ou des gabarits de la jambes avec la position du pied ou avec une jupe, ou des gabarits des bras avec la position de la main. Ainsi, à chaque itération, on obtiendrait une évaluation plus précise de la posture de la personne de la part du gabarit.

- **La reconnaissance de pose** : c'est un domaine très intéressant qui donne lieu à de nombreuses recherches. Or notre méthode de coupe de graphe guidée par un gabarit par parties réalise une estimation de la posture de la personne. Si l'on supprime les contraintes de la segmentation et que l'on élargit le choix des postures proposées par les gabarits par parties, notre méthode pourrait être adaptée à la reconnaissance de pose.

Bibliographie

[Abd-Almageed et al., 2007] Abd-Almageed, W., Hussein, M., Abdelkader, M., and Davis, L. (2007). Real-time detection and tracking from mobile vehicles. In *Intelligent Transportation Systems Conference*, pages 149–154.

[Adato et al., 2011] Adato, Y., Zickler, T., and Ben-Shahar, O. (2011). A polar representation of motion and implications for optical flow. In *Computer Vision and Pattern Recognition*, pages 1145–1152.

[Alonso et al., 2007] Alonso, I. P., Llorca, D. F., Sotelo, M. A., Bergasa, L. M., Toro, P. R. D., Nuevo, J., Ocania, M., and Garrido, M. A. G. (2007). Combination of feature extraction methods for svm pedestrian detection. *Transactions on Intelligent Transportation Systems*, 8 :292–307.

[Andriluka et al., 2008] Andriluka, M., Roth, S., and Schiele, B. (2008). People tracking by detection and people detection by tracking. In *Computer Vision and Pattern Recognition*, pages 1–8.

[Beleznai et al., 2004] Beleznai, C., Fruhstuck, B., and Bischof, H. (2004). Human detection in groups using a fast mean shift procedure. *International Conference on Image Processing*, 1 :349.

[Bertozzi et al., 2003] Bertozzi, M., Broggi, A., Chapuis, R., Chausse, F., Fascioli, A., and Tibaldi, A. (2003). Shape-based pedestrian detection and localization. In *Intelligent Transportation Systems Conference*, pages 328–333.

[Bertozzi et al., 2004] Bertozzi, M., Broggi, A., Fascioli, A., Tibaldi, A., Chapuis, R., and Chausse, F. (2004). Pedestrian localization and tracking system with kalman filtering. In *Intelligent Vehicles Symposium*, pages 584–589.

[Bertozzi et al., 2007] Bertozzi, M., Broggi, A., Rose, M. D., Felisa, M., Rakotomamonjy, A., and Suard, F. (2007). A pedestrian detector using histograms of oriented gradients and a support vector machine classifier. In *Intelligent Transportation Systems Conference*, pages 143–148.

[Blake et al., 2004] Blake, A., Rother, C., Brown, M., Pérez, P., and Torr, P. H. S. (2004). Interactive image segmentation using an adaptive gmmrf model. *European Conference in Computer Vision*, pages 428–441.

[Blank et al., 2005] Blank, M., Gorelick, L., Shechtman, E., Irani, M., and Basri, R. (2005). Actions as space-time shapes. *International Conference in Computer Vision*, pages 1395–1402.

[Bobick et al., 2001] Bobick, A. F., Davis, J. W., Society, I. C., and Society, I. C. (2001). The recognition of human movement using temporal templates. *Transactions on Pattern Analysis and Machine Intelligence*, 23 :257–267.

[Borenstein and Malik, 2006] Borenstein, E. and Malik, J. (2006). Shape guided object segmentation. In *Computer Vision and Pattern Recognition*, pages 969–976.

[Bouguet, 2000] Bouguet, J.-Y. (2000). Pyramidal implementation of the Lucas Kanade feature tracker description of the algorithm.

[Boykov and Jolly, 2001] Boykov, Y. and Jolly, M.-P. (2001). Interactive graph cuts for optimal boundary and region segmentation of objects in n-d images. *International Conference in Computer Vision*, 1 :105–112.

[Boykov and Kolmogorov, 2004] Boykov, Y. and Kolmogorov, V. (2004). An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *Transactions on Pattern Analysis and Machine Intelligence*, 26 :1124–1137.

[Bray et al., 2006] Bray, M., Kohli, P., and Torr, P. H. (2006). Posecut : Simultaneous segmentation and 3d pose estimation of humans using dynamic graph-cuts. *European Conference in Computer Vision*, pages 642–655.

[Bregler and Malik, 1998] Bregler, C. and Malik, J. (1998). Tracking people with twists and exponential maps. *Computer Vision and Pattern Recognition*, pages 8–15.

[Brox and Weickert, 2004] Brox, T. and Weickert, J. (2004). A tv flow based local scale measure for texture discrimination. In *European Conference in Computer Vision*, volume 3022, pages 578–590.

[Bruno and Pellerin, 2002] Bruno, E. and Pellerin, D. (2002). Robust motion estimation using spatial gabor-like filters. *Signal Processing*, 82 :297–309.

[Bugneau and Perez, 2007] Bugneau, A. and Perez, P. (2007). Joint tracking and segmentation of objects using graph cuts. In *International Conference on Advanced Concepts for Intelligent Vision Systems*, pages 628–639.

[Canny, 1986] Canny, J. (1986). A computational approach to edge detection. *Transactions on Pattern Analysis and Machine Intelligence*, 8 :679–714.

[Chuang et al., 2002] Chuang, Y.-Y., Agarwala, A., Curless, B., Salesin, D. H., and Szeliski, R. (2002). Video matting of complex scenes. *International Conference on Computer Graphics*, 21 :243–248.

[Chuang et al., 2001] Chuang, Y.-Y., Curless, B., Salesin, D. H., and Szeliski, R. (2001). A bayesian approach to digital matting. *Computer Vision and Pattern Recognition*, 2 :264–271.

[Comaniciu, 2003] Comaniciu, D. (2003). An algorithm for data-driven bandwidth selection. *Transactions on Pattern Analysis and Machine Intelligence*, 25 :281–288.

[Corvee and Bremond, 2010] Corvee, E. and Bremond, F. (2010). Body parts detection for people tracking using trees of histogram of oriented gradient descriptors. *International Conference on Advanced Video and Signal-Based Surveillance*, pages 469–475.

[Cutler and Davis, 2000] Cutler, R. and Davis, L. (2000). Robust real-time periodic motion detection, analysis and applications. *Transactions on Pattern Analysis and Machine Intelligence*, 22 :781–796.

[Dai et al., 2005] Dai, C., Zheng, Y., and Li, X. (2005). Layered representation for pedestrian detection and tracking in infrared imagery. In *Computer Vision and Pattern Recognition*, page 13.

[Dalal, 2006] Dalal, N. (2006). *Finding People in Images and Videos*. PhD thesis, Mathématiques, Sciences et Technologie de l'Information.

[Dalal and Triggs, 2005] Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition*, volume 1, pages 886–893.

[Dalal et al., 2006] Dalal, N., Triggs, B., and Schmid, C. (2006). Human detection using oriented histograms of flow and appearance. *European Conference in Computer Vision*, 3954 :428–441.

[Dantzig et al., 1956] Dantzig, G., Ford, L., and Fulkerson, D. (1956). A primal-dual algorithm for linear programms. *Linear Inequalities and Related Systems*, pages 171–181.

[Das et al., 2009] Das, P., Veksler, O., Zavadsky, V., and Boykov, Y. (2009). Semiautomatic segmentation with compact shape prior. *Image and Vision Computing*, 27 :206–219.

[Dijkstra, 1959] Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. 1 :269–271.

[Elder and Zucker, 1996] Elder, J. H. and Zucker, S. W. (1996). Computing contour closure. In *European Conference in Computer Vision*, pages 399–412.

[Elzein et al., 2003] Elzein, H., Lakshmanan, S., and Watta., P. (2003). A motion and shape-based pedestrian detection algorithm. In *Intelligent Vehicles Symposium*, pages 500–504.

[Eng et al., 2004] Eng, H.-L., Wang, J., Kam, A. H., and Yau, W.-Y. (2004). A bayesian framework for robust human detection and occlusion handling using human shape model. In *International Conference on Pattern Recognition*, pages 257–260.

[Enzweiler and Gavrila, 2009] Enzweiler, M. and Gavrila, D. (2009). Monocular pedestrian detection : Survey and experiments. volume 31, pages 2179–2195.

[Fablet and Black, 2002] Fablet, R. and Black, M. J. (2002). Automatic detection and tracking of human motion with a view-based representation. In *European Conference in Computer Vision*, pages 476–491.

[Fawcett, 2006] Fawcett, T. (2006). An introduction to roc analysis. *Pattern Recognition Letters*, 27(8) :861–874.

[Felzenszwalb et al., 2010] Felzenszwalb, P., Girshick, R., McAllester, D., and Ramanan, D. (2010). Object detection with discriminatively trained part-based models. volume 32, pages 1627–1645.

[Ferrari et al., 2007] Ferrari, V., Jurie, F., and Schmid, C. (2007). Accurate object detection with deformable shape models learnt from images. In *Computer Vision and Pattern Recognition*.

[Ferrari et al., 2006] Ferrari, V., Tuytelaars, T., and Gool, L. V. (2006). Object detection with contour segment networks. In *European Conference in Computer Vision*.

[Freedman and Zhang, 2005] Freedman, D. and Zhang, T. (2005). Interactive graph cut based segmentation with shape priors. In *Computer Vision and Pattern Recognition*, pages 755–762.

[Freund and Schapire, 1995] Freund, Y. and Schapire, R. E. (1995). A decision-theoretic generalization of on-line learning and an application to boosting. In *Proceedings of*

the Second European Conference on Computational Learning Theory, pages 23–37. Springer-Verlag.

[Gavrila and Philomin, 1999] Gavrila, D. and Philomin, V. (1999). Real-time object detection for smart vehicles. *International Conference in Computer Vision*, 1 :87–93.

[Gavrila and Giebel, 2002] Gavrila, D. M. and Giebel, J. (2002). Shape-based pedestrian detection and tracking. In *Intelligent Vehicles Symposium*, volume 1, pages 8–14.

[Gavrila and Munder, 2007] Gavrila, D. M. and Munder, S. (2007). Multi-cue pedestrian detection and tracking from a moving vehicle. *International Journal of Computer Vision*, 73 :41–59.

[Gorelick et al., 2011] Gorelick, L., Delong, A., Veksler, O., and Boykov, Y. (2011). Recursive mdl via graph cuts : Application to segmentation. In *International Conference in Computer Vision*.

[Guan et al., 2006] Guan, Y., Chen, W., Liang, X., Ding, Z., and Peng, Q. (2006). Easy matting - a stroke based approach for continuous image matting. volume 25, pages 567–576.

[Guillemaut et al., 2010] Guillemaut, J., Sarim, M., and Hilton, A. (2010). Stereoscopic content production of complex dynamic scenes using a wide-baseline monoscopic camera set-up. pages 9–12.

[Haga et al., 2004] Haga, T., Sumi, K., and Yagi, Y. (2004). Human detection in outdoor scene using spatio-temporal motion analysis. In *International Conference on Pattern Recognition*, pages 331–334.

[Hariharan et al., 2011] Hariharan, B., Arbelaez, P., Bourdev, L., Maji, S., and Malik, J. (2011). Semantic contours from inverse detectors. In *International Conference in Computer Vision*.

[Haritaoglu, 1998] Haritaoglu, I. (1998). W4s : A real time system for detecting and tracking people in 2.5 d. In *European Conference in Computer Vision*, pages 877–892.

[Haritaoglu et al., 1998] Haritaoglu, I., Harwood, D., and Davis, L. S. (1998). Ghost : A human body part labeling system using silhouettes. In *International Conference on Pattern Recognition*, pages 77–82.

[Haritaoglu et al., 1999] Haritaoglu, I., Harwood, D., and Davis, L. S. (1999). Hydra : Multiple people detection and tracking using silhouettes. In *Conference on Image Analysis and Processing*, page 6.

[Haugeard and Philipp-Foliguet, 2010] Haugeard, J.-E. and Philipp-Foliguet, S. (2010). Recherche d'objets par appariement de graphes combinant contours et régions. In *RFIA : Reconnaissance des Formes et Intelligence Artificielle*, Caen.

[He et al., 2008] He, X., Li, J., and Chen, Y. (2008). Local binary patterns with mahalanobis distance maps for human detection. *Congress on Image and Signal Processing*, 2 :520–524.

[Hérault and Jutten, 1994] Hérault, J. and Jutten, C. (1994). *Réseaux Neuronaux et Traitement du Signal*.

[Hernández et al., 2010] Hernández, A., Reyes, M., Escalera, S., and Radeva, P. (2010). Spatio-temporal grabcut human segmentation for face and pose recovery.

[Hillman et al., 2001] Hillman, P., Hannah, J., and Renshaw, D. (2001). Alpha channel estimation in high resolution images and images sequences. In *Computer Vision and Pattern Recognition*, volume 1, pages 1063–1068.

[Huart, 2007] Huart, J. (2007). *Extraction et analyse d'objets-clés pour la structuration d'images et de vidéos*. PhD thesis, Institut National Polytechnique de Grenoble.

[Hyun et al., 2008] Hyun, M., Kim, S., and Ho, Y. (2008). Multi-view image matting and compositing using trimap sharing for natural 3-d scene generation. *3DTV Conference*, pages 397–400.

[Ioffe and Forsyth, 2001] Ioffe, S. and Forsyth, D. A. (2001). Probabilistic methods for finding people. *International Journal of Computer Vision*, 43 :45–68.

[Jia et al., 2011] Jia, K., Wang, X., and Tang, X. (2011). Optical flow estimation using learned sparse model. In *International Conference in Computer Vision*.

[Joachims, 1999] Joachims, T. (1999). Making large-scale svm learning practical. In Schlkopf, B., Burges, C., and A. Smola, M.-P., editors, *Advances in Kernel Methods - Support Vector Learning*.

[Joshi and Porikli, 2010] Joshi, A. and Porikli, F. (2010). Scene-adaptive human detection with incremental active learning. In *International Conference on Pattern Recognition*, pages 2760–2763.

[Juan, 2006] Juan, O. (2006). *Quelques Extensions des Level Sets et des Graph Cuts et Leurs Applications $\tilde{A}$ $\hat{A}$ la Segmentation d'Images et de Videos*. PhD thesis, École Nationale des Ponts et Chaussées.

[Juan and Keriven, 2005] Juan, O. and Keriven, R. (2005). Trimap segmentation for fast and user-friendly alpha matting. pages 186–197.

[Kang and whan Lee, 2002] Kang, S. and whan Lee, S. (2002). Real-time pedestrian detection using support vector. In *International Workshop on Pattern Recognition with Support Vector Machines*, page 268.

[Karam et al., 2004] Karam, W., Mokbel, C., Greige, H., Pesquet-Popescu, B., and Chollet, G. (2004). Un système de détection de visage et d'extraction de paramètres basé sur les svm et des contraintes.

[Kellokumpu et al., 2008] Kellokumpu, V., Zhao, G., and Pietikainen, M. (2008). Human activity recognition using a dynamic texture based method. In *British Machine Vision Conference*.

[Kohli and Torr, 2007] Kohli, P. and Torr, P. (2007). Dynamic graph cuts for efficient inference in markov random fields. 29 :2079 – 2088.

[Komodakis and Tziritas, 2007] Komodakis, N. and Tziritas, G. (2007). Approximate labeling via graph-cuts based on linear programming. volume 29, pages 1436–1453.

[Kulkarni and Nicolls, 2009] Kulkarni, M. and Nicolls, F. (2009). Interactive image segmentation using graph cuts. *Pattern Recognition Association of South Africa*, pages 99–104.

[Kumar et al., 2005] Kumar, M. P., Torr, P. H. S., and Zisserman, A. (2005). Obj cut. *Computer Vision and Pattern Recognition*, 1 :18–25.

[Laptev, 2006] Laptev, I. (2006). Improvements of object detection using boosted histograms. In *British Machine Vision Conference*, pages 949–958.

[Lattari et al., 2010] Lattari, L., Conci, A., Montenegro, A., Clua, E., Mota, V. F., and Vieira, M. B. (2010). Colour based human skin segmentation using graph-cuts. In *International Conference on Systems, Signals and Image Processing*.

[Lee et al., 2004] Lee, D.-J., Pengcheng, Z., Thomas, A., and Schoenberger, R. (2004). Shape-based human detection for threat assessment. *Conference on Visual Information Processing*, 5438 :81–91.

[Leibe et al., 2004] Leibe, B., Leonardis, A., and Schiele, B. (2004). Combined object categorization and segmentation with an implicit shape model. *European Conference in Computer Vision*, 1 :1–16.

[Leibe et al., 2008] Leibe, B., Leonardis, A., and Schiele, B. (2008). Robust object detection with interleaved categorization and segmentation. *International Journal of Computer Vision*, 77 :259–289.

[Li et al., 2004] Li, Y., Sun, J., Tang, C.-K., and Shum, H.-Y. (2004). Lazy snapping. *Transactions on Graphics*, 23 :1–8.

[Lin et al., 2007a] Lin, Z., Davis, L. S., Doermann, D., and DeMenthon, D. (2007a). Hierarchical part-template matching for human detection and segmentation. In *International Conference in Computer Vision*.

[Lin et al., 2007b] Lin, Z., Davis, L. S., Doermann, D., and DeMenthon, D. (2007b). An interactive approach to pose-assisted and appearance-based segmentation of humans. *International Conference in Computer Vision*, pages 1–8.

[Liu and Sarkar, 2004] Liu, Z. and Sarkar, S. (2004). Challenges in segmentation of human forms in outdoor video. *Computer Vision and Pattern Recognition*, pages 43–50.

[Lowe, 1999] Lowe, D. G. (1999). Object recognition from local scale-invariant features. In *International Conference in Computer Vision*, pages 1150–1157.

[Lu et al., 2008] Lu, H., Jia, C., and Zhang, R. (2008). An effective method for detection and segmentation of the body of human in the view of a single stationary camera. In *International Conference on Pattern Recognition*, pages 1–4.

[Lucas and Kanade, 1981] Lucas, B. D. and Kanade, T. (1981). An iterative image registration technique with an application to stereo vision. *Proceedings of Imaging Understanding Workshop*, pages 121–130.

[Malcolm et al., 2007] Malcolm, J., Rathi, Y., and Tannenbaum, A. (2007). Graph cut segmentation with nonlinear shape priors. *International Conference on Image Processing*, 4 :365–368.

[Martin, 2002] Martin, D. R. (2002). *An Empirical Approach to Grouping and Segmentation*. PhD thesis, University of California, Berkeley, USA.

[McGuire et al., 2005] McGuire, M., Matusik, W., Pfister, H., Hughes, J. F., and Durand, F. (2005). Defocus video matting. *Transactions on Graphics*, 24 :567–576.

[Migniot et al., 2010] Migniot, C., Bertolino, P., and Chassery, J.-M. (2010). Contour segment analysis for human silhouette pre-segmentation. *International Conference on Computer Vision Theory and Applications*.

[Migniot et al., 2011a] Migniot, C., Bertolino, P., and Chassery, J.-M. (2011a). Automatic people segmentation with a template-driven graph cut. *International Conference on Image Processing*, pages 3210–3213.

[Migniot et al., 2011b] Migniot, C., Bertolino, P., and Chassery, J.-M. (2011b). Segmentation automatique de personnes par coupe de graphe et gabarits. *Gretsi*.

[Mikolajczyk et al., 2004] Mikolajczyk, K., Schmid, C., and Zisserman, A. (2004). Human detection based on a probabilistic assembly of robust part detectors. *European Conference in Computer Vision*, 3021/2004 :69–82.

[Mohan et al., 2001] Mohan, A., Papageorgiou, C., and Poggio, T. (2001). Example-based object detection in images by components. *Transactions on Pattern Analysis and Machine Intelligence*, 23 :349–361.

[Mori et al., 2004] Mori, G., Ren, X., Efros, A. A., and Malik, J. (2004). Recovering human body configurations : Combining segmentation and recognition. *Computer Vision and Pattern Recognition*, pages 326–333.

[Munder and Gavrila, 2006] Munder, S. and Gavrila, D. (2006). An experimental study on pedestrian classification. *Transactions on Pattern Analysis and Machine Intelligence*, 28 :1863–1868.

[Munder et al., 2008] Munder, S., Schnorr, C., and Gavrila, D. (2008). Pedestrian detection and tracking using a mixture of view-based shape-texture models. *Transactions on Intelligent Transportation Systems*, 9 :333–343.

[Murai et al., 2007] Murai, Y., Fujiyoshi, H., and Kanade, T. (2007). Combined object detection and segmentation by using space-time patches. In *Asian Conference in Computer Vision*, pages 915–924.

[Niyogi and Adelson, 1994] Niyogi, S. A. and Adelson, E. H. (1994). Analyzing and recognizing walking figures in xyt. In *Computer Vision and Pattern Recognition*, pages 469–471.

[Ohba and Ikeuchi, 1997] Ohba, K. and Ikeuchi, K. (1997). Detectability, uniqueness, and reliability of eigen windows for stable verification of partially occluded objects. *Transactions on Pattern Analysis and Machine Intelligence*, 19 :1043–1048.

[Oren et al., 1997] Oren, M., Papageorgiou, C., Sinha, P., Osuna, E., and Poggio, T. (1997). Pedestrian detection using wavelet templates. In *Computer Vision and Pattern Recognition*, pages 193–199.

[Ott and Everingham, 2009] Ott, P. and Everingham, M. (2009). Implicit color segmentation features for pedestrian and object detection. In *International Conference in Computer Vision*, pages 723–730.

[Pang et al., 2011] Pang, Y., Yuan, Y., Li, X., and Pan, J. (2011). Efficient hog human detection. *Signal Processing*, 91 :773–781.

[Papageorgiou and Poggio, 2000] Papageorgiou, C. and Poggio, T. (2000). A trainable system for object detection. *International Journal of Computer Vision*, 38 :15–33.

[Philipp-Foliguet and Guigues, 2006] Philipp-Foliguet, S. and Guigues, L. (2006). Evaluation de la segmentation d'images : Etat de l'art, nouveaux indices et comparaison. *Traitement du Signal*, 23 :109–124.

[Polana and Nelson, 1994] Polana, R. and Nelson, A. (1994). Low level recognition of human motion (or how to get your man without finding his body parts. In *Computer Society Workshop on Motion of Non-Rigid and Articulated Objects*, pages 77–82.

[Ramanan, 2007] Ramanan, D. (2007). Using segmentation to verify object hypotheses. In *Computer Vision and Pattern Recognition*, pages 1–8.

[Ramanan et al., 2005] Ramanan, D., Forsyth, D. A., and Zisserman, A. (2005). Strike a pose : Tracking people by finding stylized poses. In *Computer Vision and Pattern Recognition*, pages 271–278.

[Ran et al., 2005] Ran, Y., Zheng, Q., Weiss, I., S.Davis, L., Abd-Almageed, W., and Zhao, L. (2005). Pedestrian classification from moving platforms using cyclic motion pattern. *International Conference on Image Processing*, 2 :854–857.

[Rhemann et al., 2008] Rhemann, C., Rother, C., Rav-Acha, A., and Sharp, T. (2008). High resolution matting via interactive trimap segmentation. *Computer Vision and Pattern Recognition*, pages 1–8.

[Rijsbergen, 1979] Rijsbergen, C. V. (1979). *Information Retrieval*. Dept. of Computer Science, University of Glasgow, second edition.

[Rodriguez and Shah, 2007] Rodriguez, M. D. and Shah, M. (2007). Detecting and segmenting humans in crowded scenes. In *ACM Transactions on Graphics*, pages 353–356.

[Rother et al., 2004] Rother, C., Kolmogorov, V., and Blake, A. (2004). "grabcut" : Interactive foreground extraction using iterated graph cuts. *Transactions on Graphics*, 23 :309–314.

[Ruzon and Tomasi, 2000] Ruzon, M. A. and Tomasi, C. (2000). Alpha estimation in natural images. *Computer Vision and Pattern Recognition*, pages 18–25.

[Scott, 1992] Scott, D. (1992). *Multivariate Density Estimation*. Wiley Inter-Science.

[Sharma and Davis, 2007] Sharma, V. and Davis, J. W. (2007). Integrating appearance and motion cues for simultaneous detection and segmentation of pedestrians. In *International Conference in Computer Vision*, pages 1–8.

[Shashua et al., 2004] Shashua, A., Gdalyahu, Y., and Hayun, G. (2004). Pedestrian detection for driving assistance systems : Single-frame classification and system level performance. In *Intelligent Vehicles Symposium*, pages 1–6.

[Shet et al., 2007] Shet, V., Neumann, J., Ramesh, V., and Davis, L. (2007). Bilattice-based logical reasoning for human detection. pages 1–8.

[Shotton et al., 2008a] Shotton, J., Blake, A., and Cipolla, R. (2008a). Multi-scale categorical object recognition using contour fragments. In *Transactions on Pattern Analysis and Machine Intelligence*, pages 1270–1281.

[Shotton et al., 2008b] Shotton, J., Johnson, M., and Cipolla, R. (2008b). Semantic texton forests for image categorization and segmentation. In *Computer Vision and Pattern Recognition*, pages 1–8.

[Sidenbladh, 2004] Sidenbladh, H. (2004). Detecting human motion with support vector machines. In *International Conference on Pattern Recognition*, volume 2, pages 188–191.

[Sidenbladh et al., 2000] Sidenbladh, H., Black, M. J., and Fleet, D. J. (2000). Stochastic tracking of 3d human figures using 2d image motion. In *European Conference in Computer Vision*, pages 702–718.

[Slabaugh and Unal, 2005] Slabaugh, G. and Unal, G. (2005). Graph cuts segmentation using an elliptical shape prior. *International Conference on Image Processing*, 2 :1222–1225.

[Smith and Blinn, 1996] Smith, A. R. and Blinn, J. F. (1996). Blue screen matting. In *SIGGRAPH*, pages 259–268, New York, NY, USA.

[Sun et al., 2004] Sun, J., Jia, J., Tang, C.-K., and Shum, H.-Y. (2004). Poisson matting. *Transactions on Graphics*, 23.

[Tosato et al., 2010] Tosato, D., Farenzena, M., Cristani, M., and Murino, V. (2010). A re-evaluation of pedestrian detection on riemannian manifolds. In *International Conference on Pattern Recognition*, pages 3308–3311.

[Toth and Aach, 2003] Toth, D. and Aach, T. (2003). Detection and recognition of moving objects using statistical motion detection and fourier descriptors. In *International Conference on Image Analysis and Processing*, page 430.

[Tuzel et al., 2007] Tuzel, O., Porikli, F., and Meer, P. (2007). Human detection via classification on riemannian manifolds. *Computer Vision and Pattern Recognition*, 0 :1–8.

[Utsumi and Tetsutani, 2002] Utsumi, A. and Tetsutani, N. (2002). Human detection using geometrical pixel value structures. In *International Conference on Automatic Face and Gesture Recognition*, pages 34–39.

[Vapnik, 1995] Vapnik, V. (1995). *The Nature of Statistical Learning Theory*. Springer-Verlag.

[Viola and Jones, 2001] Viola, P. and Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. *Computer Vision and Pattern Recognition*, 1 :511–518.

[Viola et al., 2003] Viola, P., Jones, M., and Snow, D. (2003). Detecting pedestrians using patterns of motion and appearance. In *International Conference in Computer Vision*, pages 734–741.

[Wang and Zhang, 2010] Wang, H. and Zhang, H. (2010). Adaptive shape prior in graph cut segmentation. *International Conference on Image Processing*, pages 3029–3032.

[Wang and Cohen, 2005] Wang, J. and Cohen, M. F. (2005). An iterative optimization approach for unified image segmentation and matting. In *International Conference in Computer Vision*, pages 936–943.

[Wang et al., 2007a] Wang, L., Shi, J., Song, G., and fan Shen, I. (2007a). Object detection combining recognition and segmentation. *Asian Conference in Computer Vision*, pages 189–199.

[Wang and Wang, 2011] Wang, M. and Wang, X. (2011). Automatic adaptation of a generic pedestrian detector to a specific traffic scene. In *Computer Vision and Pattern Recognition*, volume 1, pages 401–3408.

[Wang et al., 2007b] Wang, O., Finger, J., Yang, Q., Davis, J., and Yang, R. (2007b). Automatic natural video matting with depth. *Pacific Graphics*, pages 469–472.

[Wang et al., 2009] Wang, X., Han, T., and Yan, S. (2009). An hog-lbp human detector with partial occlusion handling. *International Conference in Computer Vision*, pages 32–39.

[Winn and Shotton, 2006] Winn, J. and Shotton, J. (2006). The layout consistent random field for recognizing and segmenting partially occluded objects. In *Computer Vision and Pattern Recognition*, volume 1, pages 37–44.

[Wohler et al., 1998] Wohler, C., Anlauf, J. K., Portner, T., and Franke, U. (1998). A time delay neural network algorithm for real-time pedestrian recognition. In *International Conference on Intelligent Vehicle*, pages 247–251.

[Wren et al., 1997] Wren, C., Azarbayejani, A., Darrell, T., and Pentland, A. (1997). Pfinder : Real-time tracking of the human body. *Transactions on Pattern Analysis and Machine Intelligence*, 19 :780–785.

[Wu and Nevatia, 2007] Wu, B. and Nevatia, R. (2007). Simultaneous object detection and segmentation by boosting local shape feature based classifier. In *Computer Vision and Pattern Recognition*, volume 0, pages 1–8.

[Wu and Yu, 2006] Wu, Y. and Yu, T. (2006). A field model for human detection and tracking. *Transactions on Pattern Analysis and Machine Intelligence*, 28 :753–765.

[Xu and Fujimura, 2003] Xu, F. and Fujimura, K. (2003). Human detection using depth and gray images. In *Conference on Advanced Video and Signal Based Surveillance*, pages 115–121.

[Yasnoff et al., 1979] Yasnoff, W. A., Galbraith, W., and Bacus, J. (1979). Error measures for objective assessment of scene segmentation algorithms. *Analytical and Quantitative Cytology*, 1 :107–121.

[Yilmaz and Shah, 2004] Yilmaz, A. and Shah, M. (2004). Contour-based object tracking with occlusion handling in video acquired using mobile cameras. In *Transactions on Pattern Analysis and Machine Intelligence*, volume 26, pages 1531–1536.

[Yoon and Kim, 2004] Yoon, S. M. and Kim, H. (2004). Real-time multiple people detection using skin color, motion and appearance information. In *International Workshop on Robot and Human Interactive Communication*, pages 331–334.

[Zaklouta, 2012] Zaklouta, F. (2012). *Multiclass Object Recognition for Driving Assistance Systems and Video Surveillance*. PhD thesis, Ecole Nationale Supérieure des Mines de Paris.

[Zhao and Davis, 2005] Zhao, L. and Davis, L. S. (2005). Closely coupled object detection and segmentation. *International Conference in Computer Vision*, pages 454–461.

[Zhao and Thorpe, 2000] Zhao, L. and Thorpe, C. (2000). Stereo and neural network-based pedestrian detection. *Transactions on Intelligent Transportation Systems*, 1 :148–154.

[Zhao and Nevatia, 2003] Zhao, T. and Nevatia, R. (2003). Bayesian human segmentation in crowded situations. *Computer Vision and Pattern Recognition*, 2 :459–466.

[Zhu et al., 2006] Zhu, Q., Yeh, M.-C., Cheng, K.-T., and Avidan, S. (2006). Fast human detection using a cascade of histograms of oriented gradients. *Computer Vision and Pattern Recognition*, 2 :1491–1498.

[Zitnick et al., 2004] Zitnick, C., Kang, S., ans S. Winder, M. U., and Szeliski, R. (2004). High-quality video view interpolation using a layered representation. *Transactions on Graphics*, 3 :600–608.

Appendices

Annexe A

Bases de données statiques

Afin de réaliser les apprentissages des machines à vecteurs de support lors du processus de classification mais aussi pour réaliser les tests en vue d'évaluer les performances de nos algorithmes, des bases de données d'images ont été nécessaires. L'objectif est d'avoir un ensemble d'éléments d'une classe la représentant le mieux possible dans sa diversité de forme, de couleur, de texture ... afin de pouvoir efficacement réaliser un modèle de cette classe. Dans cette optique, nous avons utilisé deux bases de données que nous présentons dans cette annexe.

A.1 La base de données statique de personnes de l'INRIA

Cette base de données, réalisée par Navneet Dalal afin de réaliser sa méthode de détection de personnes [Dalal and Triggs, 2005], est une base de données libre disponible sur le site *http ://pascal.inrialpes.fr/data/human/*. Elle est composée de deux sous-ensembles :

- **les exemples négatifs :** il s'agit d'un ensemble de 912 images en couleurs ne contenant pas d'élément de la classe des personnes (figure A.1). Il contient une grande diversité de paysages et d'environnements urbains et ruraux (dans le but de décrire tous les arrières plans possibles) mais aussi certains objets comme des vélocipèdes ou des voitures (dans le but de décrire les différents premiers plans que l'algorithme doit reconnaître comme n'appartenant pas à la classe des personnes).

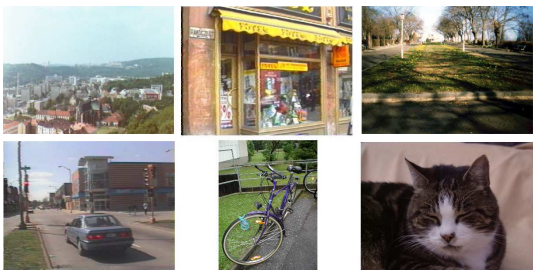

FIG. A.1 – Exemples d'éléments négatifs de la base de données statiques de personnes de l'INRIA.

- **les exemples positifs :** il s'agit d'un ensemble de 2141 images en couleurs contenant des éléments de la classe des personnes (figure A.2). Les images sont de taille 96×160 pixels contenant une personne centrée sur l'image et de taille approximative 35×100 pixels. Les personnes présentes dans ces images sont dans toutes les postures debout possibles et portent des vêtements avec une grande variété de couleurs et de textures. Il y a des photos de neige (où les personnes portent des habits chauds très épais) et des photos de plage (où les personnes portent uniquement des maillots de bains) afin de bien faire varier l'épaisseur des silhouettes et d'être bien représentatif. A chaque image est ajouté son symétrique selon l'axe verticale afin de ne pas favoriser une orientation de la prise de vue en particulier.

La base de données négatives est utilisée pour l'apprentissage et la base de données positives est utilisée pour les différents tests et pour la réalisation de la base de données de silhouettes.

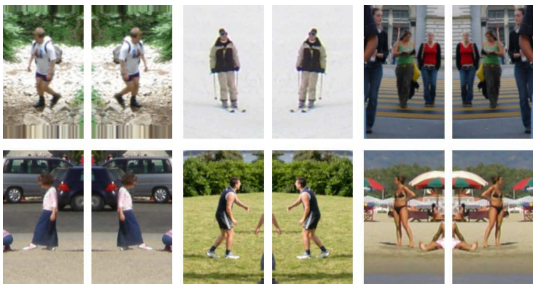

FIG. A.2 – Exemples d'éléments positifs de la base de données statiques de personnes de l'INRIA.

A.2 La base de données de silhouettes de personnes

Notre méthode de pré-segmentation présente une étude locale. La classification se réalise sur la zone de l'image restreinte qu'est le bloc. Pour l'étude globale de la détection, chaque fenêtre positive contient une personne et l'information est donc pertinente. Les images peuvent donc être utilisées comme telles. Mais, dans notre approche, une étude locale des images est effectuée. Or, certaines régions des images positives ne contiennent pas d'élément de la silhouette recherchée (figure A.3) et ne doivent donc pas intégrer la liste des exemples positifs de l'apprentissage. Voila pourquoi une nouvelle base de données ne contenant que les silhouettes a été créée. Ces silhouettes sont réalisées manuellement à partir des exemples positifs de la base de données de l'INRIA.

Les silhouettes obtenues sont en rouge sur un fond noir (figure A.4). Cette base de données contient 400 images et est disponible librement sur le site : $http://www.gipsa-lab.inpg.fr/\sim cyrille.migniot/$.

Pour cette thèse, cette base de données a non seulement servie pour l'apprentissage de notre méthode de pre-segmentation [Migniot et al., 2010] (chapitre 3) mais aussi pour définir la vérité terrain nécéssaire pour évaluer les résultats de notre méthode de segmentation par coupe de graphe guidée par un gabarit [Migniot et al., 2011a][Migniot et al., 2011b] (chapitre 5).

FIG. A.3 – Lors du découpage de la fenêtre de détection positive, certains blocs ne contiennent aucun élément de la silhouette de la personne. Il ne faut donc pas que ces blocs vides d'information pertinente influent sur l'apprentissage.

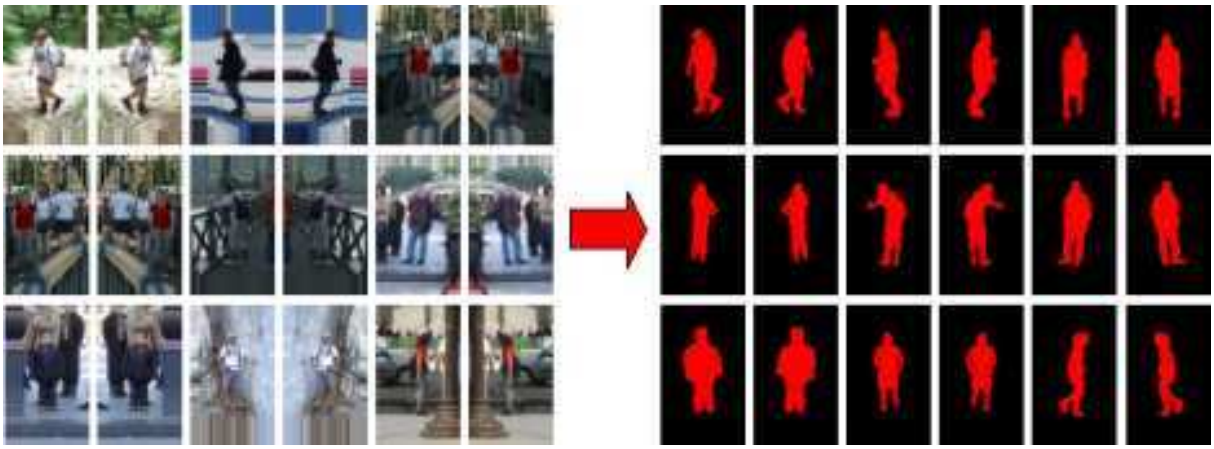

FIG. A.4 – A partir des images de personnes (à gauche), on réalise une base de données de silhouettes rouges sur un fond noir (à droite). Cette base de données permet l'apprentissage pour la pré-segmentation.

Annexe B

Bases de données de séquences vidéo

La segmentation de personnes dans les vidéos est un problème bien particulier. Que ce soit pour réduire le temps de calcul ou bien pour utiliser le mouvement et la redondance entre *frames* comme caractéristiques descriptives de la classe recherchée, le problème nécessite un traitement spécifique.
Une vidéo est découpée en plans pour conserver la continuité spatio-temporelle. Ces plans sont représentés sous la forme d'objet 3D XYT (les deux dimensions spatiales de la *frame* et la dimension temporelle).

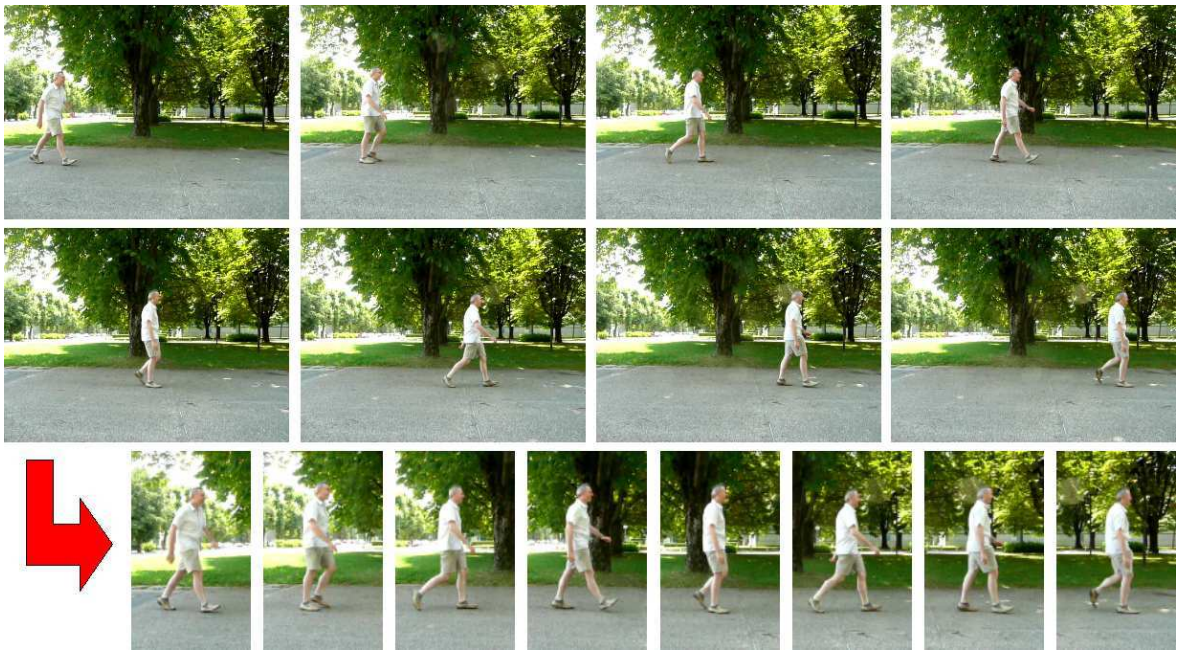

FIG. B.1 – Par cohérence avec les fenêtres de détection, les plans vidéo (en haut) sont recadrées (en bas) pour bien suivre la personne étudiée.

Pour tester les algorithmes développés, évaluer leurs performances et repérer leurs défaillances, il faut alors une série de plans vidéo tests représentatifs de la classe étudiée et des difficultés qu'elle représente. La marche et en particulier le mouvement des jambes, est discriminante des personnes. Nous avons donc réalisé une base de données de plans vidéo où des personnes marchent.

Les séquences de départ ont été réalisées à partir d'une caméra tenue à la main. Les *frames* sont de dimension 640×480 pixels et les séquences durent entre 3 et 14 secondes (soit entre environ 75 et 350 *frames*).
Néanmoins, dans notre étude, par cohérence avec les fenêtres de détection utilisées, nous avons recadré les séquences selon un format 96×160 pixels avec la personne étudiée centrée et d'une taille de 100 pixels de haut environ (figure B.1). Le suivi et le positionnement de ces fenêtres à partir de la séquence de départ peut se réaliser facilement à partir de [Dalal et al., 2006] par exemple. Les séquences suivent alors la personne et c'est l'arrière plan qui défile et se trouve donc en mouvement. Notons que le déplacement de la fenêtre dans la *frame* initiale ainsi que la vitesse de la personne s'évaluent à partir d'une coupe XT au niveau du torse.

La base de données représente les principales variations de la classe des personnes :

- **Différents angles de vue** : la base de données comporte des déplacements selon diverses prises de vues (figure B.2) qui font varier la forme de la silhouette ainsi que le motif sur une coupe XT au niveau des jambes (section 6.1.2).

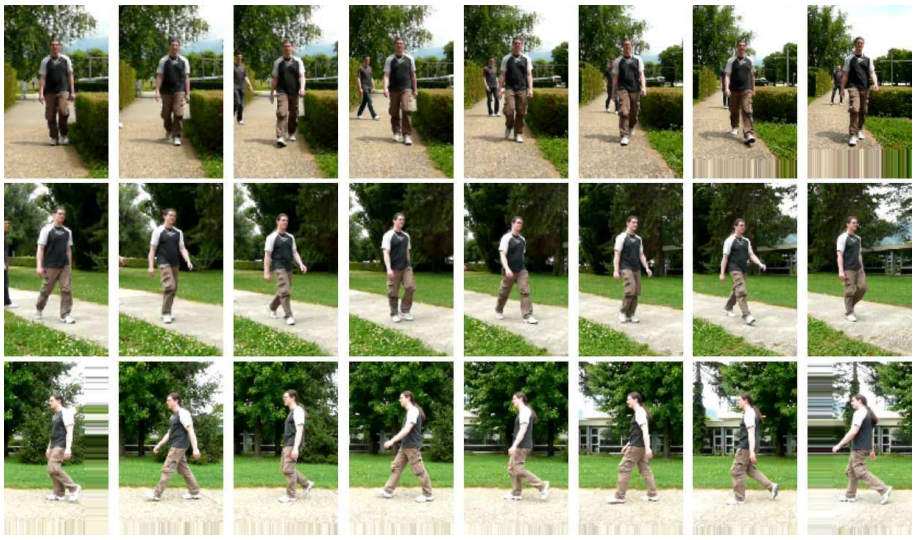

FIG. B.2 – La marche d'une personne est visualisée différemment selon la prise de vue.

- **Occultations** : la base de données comporte plusieurs séquences où la personne est partiellement cachée par un objet (un siège ou une haie). En effet la segmentation est plus difficile si un objet vient se placer entre la personne et l'objectif (figure B.3). Néanmoins, le processus est facilité si, dans du plan, la personne est entièrement visible sur certaines *frames*.

- **Intérieur/extérieur** : la luminosité d'une prise peut accentuer ou diminuer les contrastes et donc plus ou moins faciliter la segmentation. Pour s'en rendre compte, la base de données possède des séquences en plein air et d'autres à l'intérieur (figure B.4).

FIG. B.3 – Séquence présentant des occultations. Un objet cache une partie de la personne.

FIG. B.4 – Séquence selon différentes luminosités.

- **Vitesse de déplacement** : si une personne court, elle a un comportement différent que si elle marche. Plusieurs séquences de course sont alors ajoutées. Des séquences où la personne change de comportements (arrêt, marche, course) sont aussi présentes pour vérifier la continuité de la segmentation et l'effet du bon suivi (figure B.5).

FIG. B.5 – Il est interessant de tester des séquences de courses (en haut) ou avec des ruptures de vitesse (en bas).

- **Habillements variés** : la base de données contient des personnes habillées avec des tenues différentes pour bien rendre compte de la variété de couleurs de la classe des personnes (figure B.6).

FIG. B.6 – Les séquences contiennent des personnes habillées avec des teintes différentes.

- **Cas particuliers** : la base de données contient enfin des plans avec un arrière plan en mouvement rapide (une voiture passant derrière la personne), avec le croisement de plusieurs personnes et avec une marche avec un mouvement excessif des bras (figure B.7).

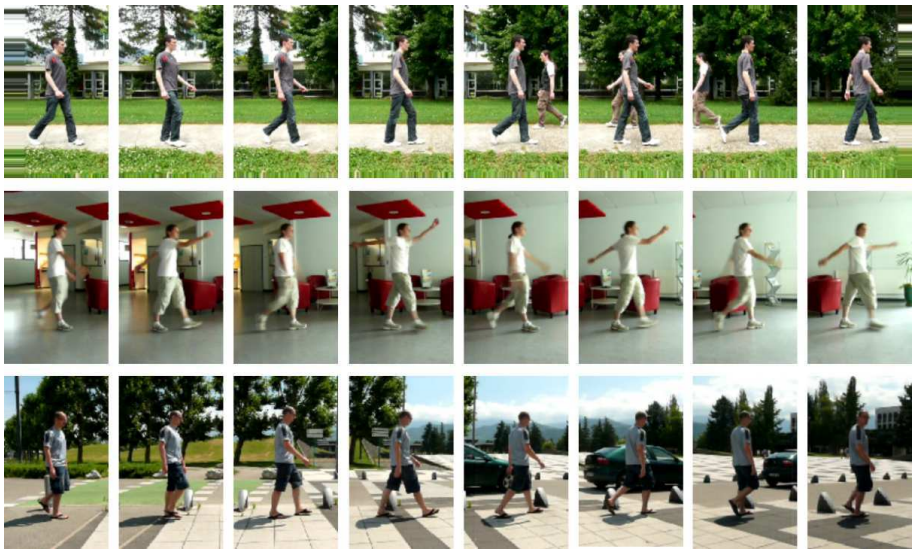

FIG. B.7 – Cas particuliers où des personnes se croisent (en haut), où la personne fait de grands gestes avec ses bras (au centre) et où l'arrière plan bouge par l'intermédiaire d'une voiture (en bas).

La base de données contient 22 séquences non recadrées et 27 séquences recadrées. Elle est libre et disponible sur le site : *http* : *//www.gipsa − lab.inpg.fr/* ∼ *cyrille.migniot/recherches.html*.

Annexe C

Fermeture de contour

C.1 Introduction

Cette annexe présente la méthode de fermeture de contour de James H Elder [Elder and Zucker, 1996]. Cet article ne concerne pas la segmentation d'une classe en particulier.
La fermeture de contour est fortement liée à notre étude de recherche de cycle optimal guidée par une pré-segmentation (sections 4.2 et 4.3). Lors du calcul des contours de la fenêtre de détection, certains segments de la silhouette recherchée peuvent manquer (figure C.1). Les résultats de notre pré-segmentation en sont perturbés (section 3.4.3). Le seuil de la taille des transitions dans le graphe est alors plus difficile. Fermer les contours peut remédier à ce phénomène.
Elder met en œuvre une méthode utilisant les segments de contours. Il définit un calcul d'affinité entre deux segments. Nous avons fait de même dans notre étude (section 4.3.2).

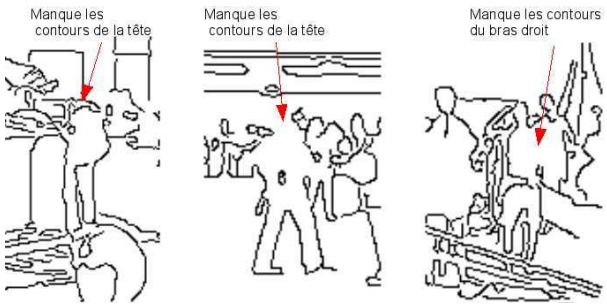

Fig. C.1 – Suite au filtrage de Canny est obtenue une carte des contours. La plupart d'entre eux ne sont cependant pas fermés ce qui pause dans notre cas problème.

C.2 Calcul de l'affinité

Pour fermer les contours, la méthode de Elder réalise un graphe dont l'arbre optimal donne les contours à relier.

C.2.1 Détermination des nœuds du graphe

Un filtrage simple (Canny) permet d'obtenir la carte des contours de l'image. Ces contours sont modélisés en une suite de segments-tangentes : les zones du contour où la tangente varie le moins (contours rectilignes). Par continuité, les parcelles de variations de la tangente de plus en plus élevée sont déterminées. Ces segments-tangentes forment les nœuds du graphe.

C.2.2 Description d'un couple de segments voisins

Les caractéristiques qui vont définir chaque liaison entre couple de segments tangentes sont déterminées afin de décrire la probabilité que ces deux segments tangentes se suivent.

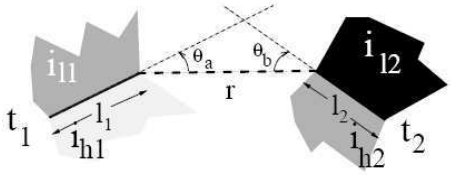

FIG. C.2 – Schéma représentant les caractéristiques décrivant la relation entre les segments tangentes 1 et 2

Toutes ces caractéristiques sont schématisées sur la Figure C.2 :
- **r** : la distance minimale entre les deux segments-tangentes
- l_1 **et** l_2 : les longueurs des segments-tangentes
- θ_a **et** θ_b : les angles entre les segments-tangentes
- i_{l1} **et** i_{l2} : la moyenne de l'intensité des pixels voisins du segment-tangente du côté le plus sombre
- i_{h1} **et** i_{h2} : la moyenne de l'intensité des pixels voisins du segment-tangente du côté le plus clair

Les valeurs $\Delta_{ih} = i_{h1} - i_{h2}$ et $\Delta_{il} = i_{l1} - i_{l2}$ et, pour les cas d'occultation, la longueur de fossé normalisée : $r' = \frac{r}{min(l_1,l_2)}$ sont utilisées pour définir la transition.

Le vecteur $o = \{r \text{ (ou r')}, \theta_a, \theta_b, \Delta_{ih}, \Delta_{il}\}$ caractérise alors la liaison.

C.2.3 Probabilité que deux segments tangentes se suivent

La probabilité que deux segments tangentes 1 et 2 se suivent étant donné leurs caractéristiques ($p(t_1 \rightarrow t_2|o)$) définit leur transition.
Or, d'après Bayles :

$$p(t_1 \rightarrow t_2|o) = \frac{p(o|t_1 \rightarrow t_2)p(t_1 \rightarrow t_2)}{p(o)} \tag{C.1}$$

De plus :

$$p(o) = p(o|t_1 \rightarrow t_2)p(t_1 \rightarrow t_2) + p(o|t_1 \nrightarrow t_2)p(t_1 \nrightarrow t_2) \tag{C.2}$$

Donc :

$$p(t_1 \rightarrow t_2|o) = \frac{p(o|t_1 \rightarrow t_2)p(t_1 \rightarrow t_2)}{p(o|t_1 \rightarrow t_2)p(t_1 \rightarrow t_2) + p(o|t_1 \nrightarrow t_2)p(t_1 \nrightarrow t_2)} \tag{C.3}$$

En posant $L = \frac{p(o|t_1 \nrightarrow t_2)}{p(o|t_1 \rightarrow t_2)}$ et $P = \frac{p(t_1 \nrightarrow t_2)}{p(t_1 \rightarrow t_2)}$, l'équation devient :

$$p(t_1 \rightarrow t_2|o) = \frac{1}{1+LP} \tag{C.4}$$

La variable P est expérimentalement fixée à 50 pour permettre des courbures assez importantes. Reste donc à déterminer L qui correspond au ratio de ressemblance selon que les deux segments-tangentes se suivent ou non.

Ressemblance si les deux segments-tangentes sont connexes

Trois cas de succession de segments-tangentes sont acceptés :
- la courbure : le contour se courbe doucement (r,θ_a et θ_b sont faibles)
- l'interruption : une interruption du contour apparaît (θ_a et θ_b sont faibles)
- le coin : la courbure est franche (r faible)

La probabilité est décomposée en :

$$\begin{aligned} p(o|t_1 \rightarrow t_2) =& p(o|t_1 \rightarrow t_2, courbure)p(courbure) + \\ & p(o|t_1 \rightarrow t_2, interruption)p(interruption) + \\ & p(o|t_1 \rightarrow t_2, coin)p(coin) \end{aligned} \tag{C.5}$$

Par expérience : $p(courbure) = 0.9$, $p(interruption) = 0.05$ et $p(coin) = 0.05$.
Pour le reste, les ressemblances, quand les segments tangentes se suivent, se modélisent par une gaussienne pour chaque caractéristique de la liaison o (elles sont indépendantes) définie par :

$$p(o_i|t_1 \rightarrow t_2) = \frac{\sqrt{2}}{\sqrt{\pi}\sigma_{o_i}} e^{-\frac{o_i^2}{2\sigma_{o_i}^2}} \tag{C.6}$$

Où les constantes d'échelle sont définies dans le tableau C.1.

cas étudié	σ_r	$\sigma_{r'}$	σ_{θ_a}=σ_{θ_b}	$\sigma_{\Delta b}$=$\sigma_{\Delta d}$
courbure	2	-	10	20
interruption	-	0.5	10	20
coin	2	-	90	20

TAB. C.1 – Constante d'échelle à appliquer pour chacune des caractéristiques.

Ressemblance si les deux segments-tangentes ne sont pas connexes

Soit l la longueur maximale de l'image, les probabilités recherchées sont définies par :

$$\begin{cases} p(r|t_1 \nrightarrow t_2) \approx \frac{4r}{l^2} \\ p(\theta_a|t_1 \nrightarrow t_2) = p(\theta_b|t_1 \nrightarrow t_2) = \frac{1}{\pi} \\ p(\Delta_{ih}|t_1 \nrightarrow t_2) = \frac{2}{255}(1 - \frac{\Delta_{ih}}{255}) \\ p(\Delta_{il}|t_1 \nrightarrow t_2) = \frac{2}{255}(1 - \frac{\Delta_{il}}{255}) \end{cases} \tag{C.7}$$

C.3 Utilisation de l'affinité

Ces valeurs de probabilité sont calculées pour toutes les liaisons entre segments-tangentes. Mais, pour un nœuds du graphe, seules les six meilleures forment un arc.
Le poids de l'arc est donné par $log(p(t_1 \rightarrow t_2|o)$.

Le chemin obtenu avec la résolution du problème du plus court chemin est fermé. Mais il se peut que le contour obtenu se croise (une boucle ou un chemin en forme de huit).
Pour empêcher ces phénomènes, la boucle trouvée est contrainte par l'équation :

$$\frac{1}{2\pi \sum_{1}^{n} (\theta_{a_i} + \theta_{b_i})} = \pm 1 \tag{C.8}$$

Annexe D

Calcul du flot optique

Dans la section 4.6, le mouvement est pris en compte pour ajouter de l'information caractéristique. Pour cela, des couples de *frames* successives sont étudiés et le flot optique correspondant est calculé. Celui-ci correspond, pour chaque pixel des *frames*, au vecteur $[\nu_x, \nu_y]$ le long duquel la dérivée de l'image $I_t(x, y)$ est nulle (avec x et y la position du point) :

$$\frac{\partial I_t(x,y)}{\partial x}\nu_x + \frac{\partial I_t(x,y)}{\partial y}\nu_y + \frac{\partial I_t(x,y)}{\partial t} = 0 \tag{D.1}$$

Dans cette annexe, deux méthodes d'estimation du flot optique sont présentées.

D.1 Méthode pyramidale de Lucas et Kanade

Cette méthode a été définie par Jean-Yves Bouguet dans [Bouguet, 2000]. Pour un point de coordonnée $u = [u_x, u_y]$, le vecteur $\nu = [\nu_x, \nu_y]$ est la vélocité du mouvement, c'est-à-dire la représentation de son mouvement. Lucas et Kanade [Lucas and Kanade, 1981] proposent de trouver la vélocité qui minimise la différence entre un bloc de taille $2\omega_x \times 2\omega_y$ et son semblable décalé spatialement de la vélocité dans l'image suivante. La contrainte est alors de minimiser :

$$\epsilon(\nu) = \sum_{x=u_x-\omega_x}^{u_x+\omega_x} \sum_{y=u_y-\omega_y}^{u_y+\omega_y} (I_t(x,y) - I_{t+1}(x+\nu_x, y+\nu_y))^2 \tag{D.2}$$

Pour obtenir une bonne estimation, il faut maximiser la précision et la robustesse du procédé. Pour la précision, il faut choisir des valeurs de ω_x et ω_y petites. Mais, dans ce cas, le système est très sensible aux mouvements rapides et la robustesse est plus faible. Voila pourquoi une version pyramidale est utilisée. L'image est étudiée sur un certain nombre d'échelles. Sur la première échelle, l'image est petite. La fenêtre d'étude est donc suffisamment grande. Ensuite, puisque la vélocité est évaluée à l'itération précédente, chaque étape ne fait qu'affiner l'estimation. La taille de la fenêtre d'étude suffit même sur l'image finale. Ainsi précision et robustesse sont optimisées.

Soit une pyramide à L niveaux. I^0 est l'image de départ et I^{L-1} l'image à la plus basse résolution. À chaque niveau la largeur et la hauteur de l'image sont divisées par deux. Ainsi le point étudié est réévalué en : $u^l = \frac{u}{2^l}$. Soit $g^l = [g^l_x, g^l_y]$ l'estimation faite

au niveau l de la vélocité (affinée à chaque nouveau niveau). Alors la minimisation de l'équation D.2 au niveau l devient :

$$\epsilon^l(\nu^l) = \sum_{x=u_x^l-\omega_x}^{u_x^l+\omega_x} \sum_{y=u_y^l-\omega_y}^{u_y^l+\omega_y} (I_t^l(x,y) - I_{t+1}^l(x+\nu_x^l+g_x^l, y+\nu_y^l+g_y^l))^2 \tag{D.3}$$

L'estimation est initialisée à $g^{L-1} = [0,0]$ puis réactualisée à chaque niveau par : $g^{l-1} = 2(g^l + \nu^l)$.

Il faut maintenant, pour un niveau l, trouver le vecteur ν^l.

Notons $\nabla I^l = \begin{bmatrix} I_x^l \\ I_x^l \end{bmatrix} = \begin{bmatrix} \frac{\partial I_{t+1}^l(x+g_x^l, y+g_y^l)}{\partial x} \\ \frac{\partial I_{t+1}^l(x+g_x^l, y+g_y^l)}{\partial x} \end{bmatrix}$. Pour minimiser l'équation D.3, celle-ci est dérivée :

$$\frac{\partial \epsilon^l(\nu^l)}{\partial \nu^l} = -2 \sum_{x=u_x^l-\omega_x}^{u_x^l+\omega_x} \sum_{y=u_y^l-\omega_y}^{u_y^l+\omega_y} (I_t^l(x,y) - I_{t+1}^l(x+\nu_x^l+g_x^l, y+\nu_y^l+g_y^l))\nabla I^{lT} \tag{D.4}$$

En réalisant un développement de Taylor du premier ordre autour de $\nu^l = [0,0]^T$ (puisque cette valeur est petite), on obtient :

$$\frac{\partial \epsilon^l(\nu^l)}{\partial \nu^l} \approx -2 \sum_{x=u_x^l-\omega_x}^{u_x^l+\omega_x} \sum_{y=u_y^l-\omega_y}^{u_y^l+\omega_y} (I_t^l(x,y) - I_{t+1}^l(x+g_x^l, y+g_y^l) - \nabla I^{lT}\nu^l)\nabla I^{lT} \tag{D.5}$$

En notant $\delta I^l(x,y) = I_t^l(x,y) - I_{t+1}^l(x+g_x^l, y+g_y^l)$, on a :

$$\frac{1}{2}\frac{\partial \epsilon^l(\nu^l)}{\partial \nu^l} \approx \sum_{x=u_x^l-\omega_x}^{u_x^l+\omega_x} \sum_{y=u_y^l-\omega_y}^{u_y^l+\omega_y} (\nabla I^{lT}\nu^l - \delta I^l(x,y))\nabla I^{lT}) \tag{D.6}$$

Donc :

$$\frac{1}{2}\left[\frac{\partial \epsilon^l(\nu^l)}{\partial \nu^l}\right]^T \approx \sum_{x=u_x^l-\omega_x}^{u_x^l+\omega_x} \sum_{y=u_y^l-\omega_y}^{u_y^l+\omega_y} \left(\begin{bmatrix} I_x^{l2} & I_x^l I_y^l \\ I_x^l I_y^l & I_y^{l2} \end{bmatrix} \nu^l - \begin{bmatrix} \delta I^l I_x^l \\ \delta I^l I_y^l \end{bmatrix} \right) \tag{D.7}$$

En notant $G^l = \sum_{x=u_x^l-\omega_x}^{u_x^l+\omega_x} \sum_{y=u_y^l-\omega_y}^{u_y^l+\omega_y} \begin{bmatrix} I_x^{l2} & I_x^l I_y^l \\ I_x^l I_y^l & I_y^{l2} \end{bmatrix}$ et $b^l = \sum_{x=u_x^l-\omega_x}^{u_x^l+\omega_x} \sum_{y=u_y^l-\omega_y}^{u_y^l+\omega_y} \begin{bmatrix} \delta I^l I_x^l \\ \delta I^l I_y^l \end{bmatrix}$, on a finalement : $\frac{1}{2}\left[\frac{\partial \epsilon^l(\nu^l)}{\partial \nu^l}\right]^T \approx G^l\nu^l - b^l$. Le vecteur ν^l finalement trouvé est donc défini par :

$$\nu^l = G^{l^{-1}} b^l \tag{D.8}$$

Si l'erreur réalisée lors du développement de Taylor est importante, l'estimation de la vélocité trouvée n'est pas suffisamment précise. Néanmoins l'image réajustée est forcement plus proche de l'image de départ. Un procédé d'estimation itératif de ν^l permet alors d'obtenir une meilleure valeur.

Cette méthode permet d'obtenir la vélocité et donc le flot optique.

D.2 Estimation robuste du mouvement par un filtrage spatial proche de Gabor

Voici une autre méthode [Bruno and Pellerin, 2002] de calcul du flot optique. On se base une nouvelle fois sur l'équation D.1. L'idée est ici de surcontraindre cette équation en adoptant une stratégie multi-canals. C'est-à-dire qu'un filtrage spatio-temporel est réalisé sur chaque canal. Cette décomposition permet de rajouter des équations et donc de résoudre le problème. L'hypothèse de constance de la vélocité sur un voisinage est donc nécessaire.
Soit une banque de filtre $G_i(x,y), i \in [1,N]$, l'équation devient :

$$\int\int (\nu_x \frac{\partial I_t(x,y)}{\partial x} + \nu_y \frac{\partial I_t(x,y)}{\partial y} + \frac{\partial I_t(x,y)}{\partial t}) G_i(x-x_0, y-y_0) dxdy = 0 \tag{D.9}$$

En posant pour tout i, $\Omega_i^p = \frac{\partial I * G_i}{\partial p}$, on obtient :

$$\begin{bmatrix} \Omega_1^x \Omega_1^y \\ \Omega_1^x \Omega_1^y \\ \vdots \\ \Omega_N^x \Omega_N^y \end{bmatrix} \begin{bmatrix} \nu_x \\ \nu_y \end{bmatrix} = - \begin{bmatrix} \Omega_1^t \\ \Omega_2^t \\ \vdots \\ \Omega_N^t \end{bmatrix} \tag{D.10}$$

Pour la banque de filtres, la méthode se base sur des filtres de Gabor. En effet, ceux-ci sont des filtres complexes qui donnent des estimations plus stables de la vélocité que les filtres réels. De plus, les images filtrées sont assez indépendantes et une analyse de l'image par échelle est permise. Enfin le calcul est assez rapide.
Un filtre de Gabor est de la forme :

$$G_i(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} e^{2\pi j f_0 (x.cos\theta_i + y.sin\theta_i)} \tag{D.11}$$

f_0 est la fréquence centrale, θ_i l'orientation et σ l'échelle. Il est possible de simplifier le calcul en un filtre de Gabor récursif du premier ou du troisième ordre. Un filtre du premier ordre est plus rapide mais engendre une erreur résiduelle plus importante.
Pour une bonne estimation, les canaux doivent être aussi peu redondants les uns aux autres que possible. Pour cela, l'échelle doit être importante (mais assez faible pour que le mouvement soit constant localement), la fréquence centrale doit être haute (bien que limitée par l'estimation de la dérivée spatiale) et le nombre de filtres utilisés doit être faible (mais assez important pour contraindre suffisamment l'équation de départ). Les variables des filtres sont alors déterminées. Préfiltrer les images par des filtres de différence de Gaussiennes (DOG) permet de réduire la redondance pour des échelles petites.

La technique des moindres carrés permet d'estimer la vélocité à partir de l'équation D.10. En limitant le déplacement maximal permis, la vélocité est estimée avec une précision plus grande. Pour cela, une étude multi-résolution est réalisée. C'est-à-dire que l'étude sur une pyramide de résolution de l'image en partant de l'image de plus faible résolution (où le mouvement est le plus faible) est effectuée. A chaque étage l'estimation est améliorée tout en étudiant des mouvements toujours restreints.