Développement d'un système de tracking vidéo sur caméra robotisée

Thomas Penne

N° d'ordre : D. U : 2167
E D S P I C : 536

UNIVERSITÉ BLAISE PASCAL - CLERMONT-FERRAND II

ÉCOLE **D**OCTORALE
SCIENCES **P**OUR L'**I**NGÉNIEUR DE **C**LERMONT-**F**ERRAND

T H È S E

Présentée par

Thomas PENNE

pour obtenir le grade de

DOCTEUR D'UNIVERSITÉ

Spécialité : Informatique

Développement d'un système de tracking vidéo sur caméra robotisée

---

Thèse dirigée par Vincent BARRA

préparée au sein de la société PRYNεL,
en collaboration avec la Fédération TIMS de Clermont-Ferrand

Soutenue publiquement le 14 octobre 2011 devant le jury :

M. Frédéric Lerasle	Rapporteur	Professeur à l'Université de Toulouse
M. Vincent Charvillat	Rapporteur	Professeur à l'Université de Toulouse
M. Thierry Chateau	Examinateur	Professeur à l'Université Blaise Pascal
Mme. Laure Tougne	Examinateur	Professeur à l'Université de Lyon
M. Jérôme Brossais	Visiteur	Chef de projet de la société Prynεl
M. Christophe Tilmant	Co-Directeur de Thèse	Professeur à l'Université Blaise Pascal
M. Vincent Barra	Directeur de Thèse	Professeur à l'Université Blaise Pascal

À mes parents et grands-parents,

Remerciements

Je tiens tout d'abord à remercier Vincent Barra et Christophe Tilmant, qui m'ont accompagné pendant ces trois années de thèse, pour leur disponibilité incomparable, leur véracité sans pareil, le soutien de tous les instants qu'ils m'ont témoigné, leurs innombrables satisfecit et surtout la motivation implacable qu'ils m'ont transmise lors de chacun de nos échanges.
Le sujet de ma thèse n'aurait pu exister sans les sociétés Prynɛl® et TEB®. Je souhaite remercier, par conséquent, tout particulièrement M. Stéphane Bidault pour avoir permis d'assouvir un peu plus ma soif de défis et de connaissances en m'accueillant au sein de sa société. Je souhaite également remercier tout spécialement l'ensemble du personnel de la société Prynɛl® avec lequel j'ai pu partager énormément durant ces trois années : Jérôme Brossais, acteur majeur de cette thèse qui, fort de son incroyable alacrité, a su m'accueillir à bras ouverts au sein de sa formidable équipe ; Laurent Mignot qui a été capable, sans admonitions ni admonestations, de supporter ma loquacité prononcée avec une grande mansuétude et de m'inculquer les rudiments de la vengeance méphistophélique ; Thomas Chalumeau, mon collègue attitré de traitements d'images, avec qui j'ai partagé de très nombreuses facondes durant les pauses café (ou non), avec qui j'ai pu, à de très nombreuses reprises, ratiociner sur des thèmes divers et variés et auprès de qui je m'excuse si j'ai pu jouer inconsciemment le matamore ; Anthony Monopoli, Olivier Pfauwathel, Olivier Roger et Julien Bernard qui ont fait preuve à mon égard d'une incroyable munificence et qui peuvent être considérés comme des parangons dans bien des domaines ; et enfin toute l'équipe "électronique" (David qui a parfois pu m'abhorrer mais a toujours su me pardonner mes peccadilles, Antoine, Vincent, Jean-Luc et Clément) qui m'a fait découvrir l'envers mécanico-électronique du suivi d'objet.
Je remercie également les équipes du LIMOS et du LASMEA parmi lesquelles j'ai pu comprendre la complexité du suivi d'objet et plus généralement du traitement d'images. Ma reconnaissance va plus particulièrement à Thierry Chateau qui a toujours su m'éclairer de ses sages lumières, apporter des réponses à nombre de mes questions et enfin a permis, sans ambages, de m'éviter de faire de ce manuscrit un amphigouri.
Ma gratitude va également à Frédéric Lerasle et Vincent Charvillat, qui m'ont fait l'honneur d'être les rapporteurs de cette thèse. Merci également à Laure Tougne qui a accepté de prendre part à mon jury de thèse.
Je ne pourrais finir ces remerciements sans une pensée personnelle à celle qui partage ma vie et aux membres de ma famille. Je remercie tout d'abord Audrey, qui m'a apporté un énorme soutien quand j'en avais besoin, qui a tout fait pour faciliter l'achèvement de cette thèse, notamment en réalisant, à ma place et de manière impromptue, diverses tâches vespérales et qui a consacré une partie de son temps à la relecture de ce manuscrit. Merci également du fond du coeur à mon frère et à mes parents pour leur soutien inconditionnel, à ma mère pour sa relecture très appréciée et à mes parents et grands-parents pour avoir fait de moi qui je suis aujourd'hui.

Table des matières

Table des figures

Liste des tableaux

Liste des Algorithmes

Introduction

Les systèmes de vidéo-surveillance se développent rapidement car ils répondent à un besoin de sécurisation des biens et des lieux. Une dizaine de millions de caméras de vidéo-surveillance est, à ce jour, installée dans le monde. La vidéo-surveillance des lieux publics et des résidences privées est en forte progression et d'une manière plus générale, selon une étude récente du cabinet Frost et Sullivan [1], le marché mondial de la vidéo-surveillance est en croissance importante, de 10 à 20 % par an ces dernières années. Les systèmes de vidéo-surveillance ont longtemps été rudimentaires et, à leurs débuts, ils se résumaient à un ensemble de caméras et d'écrans de visualisation situés dans un poste de contrôle. Encore aujourd'hui, les méthodes d'exploitation restent très opérateur-dépendantes et les informations temps-réel fournies par les caméras sont, de fait, sous-exploitées. La multiplication de sites couverts par ces caméras rend, de plus, le marché de la vidéo-surveillance très volumineux, d'où la nécessité d'une aide à la supervision automatique et temps-réel.

La société TEB®, basée à Corpeau en Bourgogne, tente d'apporter une solution à cette nécessité en construisant et distribuant des matériels de capture (caméras fixes ou mobiles) et d'enregistrement (enregistreur numérique) toujours plus innovants et intégrant une intelligence de traitement toujours plus avancée. Dans cette optique, la société développe, via son bureau d'études (la société Prynɛl®), depuis plusieurs années une suite de traitements d'images simples ou complexes au sein de son produit phare, l'enregistreur numérique Digipryn©.

De son côté, le problème du suivi d'objets est devenu, en quelques années, un problème récurrent dans de nombreux domaines allant de la vidéo-surveillance à l'asservissement visuel en passant par l'analyse comportementale. Partant de ce constat, la société TEB® a décidé d'enrichir son panel de fonctionnalités en intégrant à Digipryn© une solution de suivi automatique d'objet.
Cependant, malgré l'intérêt qu'elle suscite, la problématique de suivi n'est, à l'heure actuelle, toujours pas résolue de manière unifiée pour toutes les situations et tous les environnements d'observation. L'intégration au Digipryn© ajoute à cette dernière de nombreuses contraintes spécifiques. Il s'agit alors de développer un système de suivi d'objet sur caméra robotisée (site, azimut, zoom) tenant compte de la contrainte temps-réel (à 25 images par secondes) imposée par l'enregistreur.

Issu d'une étroite collaboration entre la société Prynɛl et les laboratoires LIMOS (Laboratoire d'Informatique, de Modélisation et d'Optimisation des Systèmes) et LASMEA (LAboratoire des Sciences et Matériaux pour l'Electronique et l'Automatique) réunis au sein de la fédération TIMS de Clermont-Ferrand (Technologies de

1. http ://www.frost.com/prod/servlet/report-toc.pag ?repid=N7AC-01-00-00-00

l'Information de la Mobilité et de la Sûreté), ce mémoire présente le développement d'un système de tracking vidéo temps-réel sur caméra robotisée.

Ce mémoire est structuré en quatre chapitres.
Le premier chapitre reprend la définition du suivi d'objet et décompose la problématique en deux étapes : la détection d'objet et le suivi de celui-ci. Pour chacune de ces étapes, il présente, de manière exhaustive, un ensemble de méthodes existantes et décrit les avantages et inconvénients qu'elles exposent au regard de notre contexte. Enfin il introduit la méthode qui a particulièrement retenu notre attention.
Le deuxième chapitre est l'occasion de rentrer dans les détails de la méthode choisie. Outre une présentation et une analyse détaillée de cette méthode, ce chapitre présente un ensemble de résultats publiés par l'auteur et extrait de ceux-ci un ensemble de limitations auxquelles nous apportons une solution dans le chapitre suivant.
Le troisième chapitre propose une nouvelle méthode basée sur la méthode décrite dans le chapitre deux et permettant de pallier ses limitations. Elle intègre notamment un filtre particulaire capable d'estimer un état caché intégrant différentes informations sur l'objet et sur son observabilité.
Enfin le quatrième et dernier chapitre présente un ensemble de résultats qualitatifs et quantitatifs illustrant le bon fonctionnement du suivi et comparant son efficacité avec celle de méthodes dîtes classiques
Après avoir réalisé une synthèse des différents travaux effectués pendant ces trois années, nous présentons en conclusion les perspectives qu'ils laissent entrevoir.

Chapitre 1

Détection et suivi d'objet

Sommaire

De nombreux ouvrages de référence et notamment [Yilmaz 2006] définissent le tracking dans sa forme la plus simple de la manière suivante :

Définition 1.1 *Le tracking, dans sa forme la plus simple, est l'estimation de la trajectoire d'un objet en mouvement dans le plan image.*

En d'autres termes, le tracking détermine la position d'un objet cible dans chaque image d'une séquence vidéo. Toujours selon [Yilmaz 2006], toutes les méthodes de tracking comportent deux couches techniques : la première permet de détecter l'ensemble des candidats potentiels (objets similaires à la cible suivie) dans chaque image de la séquence et la seconde effectue la mise en correspondance d'une image à l'autre d'un de ces candidats avec la cible afin de maintenir la cohérence du suivi au fil du temps.
Nous présentons ici, dans un premier temps, les différentes catégories de méthodes permettant la détection d'objets dans les différentes images de la séquence vidéo et, dans un second temps, les types d'algorithmes de mise en correspondance de ces détections.
Le lecteur intéressé pourra consulter [Yilmaz 2006] qui présente, conjointement à la catégorisation des méthodes des deux couches techniques, plusieurs possibilités de représentation des objets, plusieurs types de caractéristiques d'image ainsi que les difficultés majeures rencontrées par les algorithmes de tracking.

1.1 Détection d'objet

La détection d'objet(s) représente le fondement de tout algorithme de tracking. L'approche commune de détection des différents objets à suivre n'utilise l'information contenue que dans une image de la séquence à chaque itération. Il est cependant possible d'utiliser une information temporelle calculée à partir d'un ensemble d'images afin de réduire le nombre de fausses détections. On peut distinguer quatre ensembles d'algorithmes de détection : la détection de points dits d'intérêt, la soustraction de fond, la segmentation d'image et enfin la classification supervisée.

1.1.1 Détection de points

La détection de points permet de localiser dans l'image l'ensemble des points d'intérêt, ceux-ci étant définis comme points de l'image possédant une information localement discriminante. Les méthodes de détection les plus présentes dans la littérature sont l'opérateur d'intérêt de Moravec [Moravec 1979], le détecteur de Harris [Harris 1988], le détecteur KLT [Shi 1994] et enfin le détecteur SIFT [Lowe 2004]. Le lecteur intéressé pourra se référer à l'étude comparative de Mikolajczyk et Schmid [Mikolajczyk 2005] sur ces différents détecteurs de points d'intérêt.
L'opérateur de Moravec [Moravec 1979] considère, pour chaque pixel de l'image, une fenêtre locale centrée, détermine les variations moyennes d'intensité subies par cette fenêtre lors de petits déplacements suivant les quatre directions majeures (verticale, horizontale, diagonale et anti-diagonale) et sélectionne comme valeur représentative de la fenêtre la valeur minimale des variations subies suivant ces différents déplacements. Finalement, un point est considéré intéressant lorsque sa valeur représentative se trouve être un maximum local dans un voisinage de taille donnée. Malgré de très bonnes performances calculatoires, l'opérateur de Moravec possède trois limitations. Les calculs sont effectués pour un nombre limité de directions, ce qui rend la réponse de l'opérateur anisotropique et par conséquent l'opérateur lui même non invariant en rotation. Il est, de plus, considéré dans la littérature comme ayant une réponse bruitée du fait de l'utilisation d'une fenêtre binaire et rectangulaire. Enfin, il a été démontré qu'il est, du fait de l'utilisation d'images en niveaux de gris, très sensible au bruit présent dans l'image d'entrée.
Le détecteur de Harris [Harris 1988] tente de pallier ces limitations. La fenêtre rectangulaire et binaire utilisée par l'opérateur de Moravec est remplacée par une fenêtre circulaire de type gaussienne afin de réduire le bruit au niveau de la réponse du détecteur. L'équation régissant la variation d'intensité subie par un pixel est analytiquement développée au voisinage de l'origine du déplacement afin de couvrir tous les petits déplacements possibles et ainsi de rendre le détecteur invariant en rotation. Harris calcule donc, dans un premier temps, les dérivées partielles du premier ordre de l'image I_x dans la direction horizontale et I_y dans la direction verticale. Une matrice des moments du second ordre M est ensuite évaluée pour chaque pixel

dans un voisinage local :

$$M = \begin{pmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{pmatrix} \tag{1.1}$$

Les valeurs propres de cette matrice permettant de classer les régions suivant leur type (bord, coin ou fond), le repérage des points d'intérêt potentiels s'effectue via le calcul d'une mesure basée sur le déterminant et la trace de M :

$$R = Det(M) - k\, Tr^2(M) \tag{1.2}$$

où k représente une constante à fixer. Finalement, les points d'intérêt sont identifiés par seuillage de R et application d'un algorithme de suppression des non-optimaux. Bien qu'il soit plus efficace que l'opérateur de Moravec, le détecteur de Harris n'en est pas moins sans limitations. Comme dans la plupart des cas, une meilleure efficacité s'accompagne d'une augmentation significative des temps de calcul. L'algorithme est, de plus, du fait de l'utilisation d'informations de gradient, sensible au bruit lui aussi. Il possède également des taux de détection très faibles sur certains types de jonction et enfin, il n'est, lui non plus, pas invariant en rotation, du fait de la limitation à l'utilisation des gradients horizontaux et verticaux. Par ailleurs, le détecteur de Harris engendre la définition de trois paramètres : le paramètre k utilisé dans la formulation de R, le rayon du voisinage utilisé lors de la suppression des non-optimaux ainsi que la valeur de seuil.
Le détecteur KLT (Kanade-Lucas-Tomasi) est, lui aussi, basé sur la matrice M définie en 1.1. La différence entre les deux algorithmes réside dans leur utilisation de cette matrice pour construire une mesure de confiance et ainsi réperer les pixels intéressants. Le détecteur KLT base, pour sa part, sa mesure de confiance sur la plus petite valeur propre de M. Les pixels sont triés dans l'ordre décroissant de leur plus petite valeur propre (après seuillage). Finalement, les points d'intérêt sont identifiés après application d'un algorithme permettant de supprimer les pixels spatialement proches les uns des autres (seul le pixel de valeur propre minimale la plus haute est conservé dans chaque amas).
Le détecteur KLT produit donc une liste de pixels de coin séparés les uns des autres. De par leur définition de base identique, les détecteurs de Harris et KLT possèdent les mêmes limitations. De plus, tout comme le détecteur de Harris, le détecteur KLT requiert la définition de plusieurs paramètres : la valeur de seuillage utilisée ainsi que le(s) paramètre(s) nécessaire(s) à la définition du voisinage dans l'algorithme de suppression des pixels proches.
Le détecteur SIFT (Scale Invariant Feature Transform) tente de pallier les problèmes d'invariance des autres détecteurs. L'image est tout d'abord convoluée avec un ensemble de filtres gaussiens à différentes échelles. Des images de différences de gaussiennes sont ensuite calculées relativement à ces convolutions. Les points d'intérêt sont identifiés, dans un premier temps, comme les extremums locaux (voisinage spatio-temporel fixé) de ces images de différences. Leur position est ensuite affinée par interpolation des composantes couleur du voisinage. Enfin les pixels à contraste

faible ainsi que ceux présents le long d'un contour image sont supprimés pour ne laisser place qu'aux seuls points de réel intérêt. Le détecteur SIFT génère un nombre de pixels de coin plus important que les algorithmes précédents. Ceci est dû à l'accumulation des points d'intérêt à différentes échelles et différentes résolutions.
Les méthodes de détection de points permettent d'obtenir un ensemble de points d'intérêt ayant une texture particulière dans leur voisinage respectif. Bien qu'invariantes à de nombreuses transformations et déformations de l'image, ces méthodes semblent peu applicables dans notre cas. En effet, la diversité des cibles et des environnements de suivi rendent la modélisation des objets par un ensemble de points très difficile à mettre en place. De plus, ces méthodes sont peu adaptées au suivi d'objets déformables (comme par exemple des personnes).

1.1.2 Soustraction de fond

Afin de détecter un objet, il est possible de construire une représentation de la scène appelée "modèle de fond" et de repérer ensuite les variations par rapport à ce modèle pour chaque nouvelle image. Chaque changement significatif vis à vis du modèle représente un objet mobile. Les pixels de la région subissant ce changement sont marqués pour être traités. Un algorithme de liaison des composantes connexes est en général appliqué pour obtenir des régions connectées correspondant aux objets. Cette technique de détection est appelée soustraction de fond. Il existe une multitude d'algorithmes de soustraction de fond construisant des modèles plus ou moins complexes. Un descriptif de certaines méthodes et une comparaison de leurs performances peuvent être trouvés dans l'étude de Piccardi [Piccardi 2004].
Wren et al. [Wren 1997] ont proposé une modélisation indépendante du fond pour chaque position dans l'image. Leur modèle est basé sur l'ajustement et la mise à jour temporelle des paramètres (moyenne et écart-type) d'une fonction de densité de probabilité gaussienne en fonction des n dernières valeurs des pixels de l'image. Un pixel est alors classé comme "hors-fond" ou objet dès lors que sa valeur sort d'un intervalle défini relativement à ces paramètres. Bien que très performante en besoins mémoire et en temps de traitement, cette technique est peu adaptée aux scènes extérieures. En effet, dans ce genre de scène, plusieurs fonds peuvent se compléter (par ex. branchages mus par le vent et cachant partiellement un bâtiment), rendant la modélisation par une seule gaussienne inefficace.
Stauffer et Grimson [Stauffer 1999] introduisent la modélisation de chaque pixel (fond et objet) par une mixture de gaussiennes. La probabilité d'observer une valeur de pixel X_t particulière à l'instant t est définie au moyen d'une mixture de K gaussiennes multi-variées (valeurs rouge, vert et bleu) pondérées, chaque gaussienne modélisant un mode de fond ou d'objet du pixel :

$$P(X_t) = \sum_{i=1}^{K} \omega_{i,t} \ \eta(X_t, \mu_{i,t}, \Sigma_{i,t}) \tag{1.3}$$

où $\omega_{i,t}$ représente l'estimation du poids de la $i^{\text{ème}}$ gaussienne de la mixture à l'instant t, $\mu_{i,t}$ sa valeur moyenne à l'instant t, $\Sigma_{i,t}$ sa matrice de covariance à l'instant t et

$\eta(.,.,.)$ une fonction de densité de probabilité gaussienne multi-variée.
Pour toute nouvelle image, la valeur de chaque pixel est analysée relativement à la mixture correspondante. En cas de correspondance avec l'une des gaussiennes, le pixel est classé comme le mode représenté par la gaussienne et les paramètres de la distribution sont mis à jour. En cas de non-correspondance, une nouvelle gaussienne centrée en ce pixel est créée (poids faible et variance élevée).
Une approche similaire, introduite par Elgammal et al. [Elgammal 2000], modélise la distribution du fond via un modèle non-paramétrique basé sur une estimation de la densité de noyau sur le buffer des n dernières valeurs de fond. La fonction de densité de probabilité du fond est basée sur une somme de noyaux gaussiens centrés sur les n valeurs de fond les plus récentes X_i :

$$P(X_t) = \frac{1}{n} \sum_{i=1}^{n} \mathcal{N}(X_t, X_i, \Sigma_t) \tag{1.4}$$

où $\mathcal{N}(X_t, X_i, \Sigma_t)$ représente une fonction de densité de probabilité gaussienne multi-variée centrée en X_i et dont la matrice de covariance est Σ_t à l'instant t.
La classification des pixels résulte simplement du seuillage de P et la mise à jour du modèle découle naturellement de l'intégration des dernières données. Enfin, l'algorithme a la particularité de prendre également en compte une information spatiale en considérant un voisinage associé à chaque pixel. L'avantage de cette technique sur la précédente est sa capacité à gérer les micro-secousses caméra et les petits mouvements du fond.
Seki et al. [Seki 2003] exploitent les cooccurrences spatiales des variations de l'image en se basant sur l'idée qu'un bloc de pixels voisins appartenant au fond doit subir des variations dans le temps similaires. Leur algorithme, qui travaille sur des blocs de pixels voisins plutôt que sur les pixels eux même, se divise en deux étapes. Durant l'apprentissage, la moyenne temporelle de l'image, ses variations vis à vis de cette moyenne ainsi que la matrice de covariance relative à la moyenne sont calculées. Une transformation basée sur les vecteurs propres de cette matrice est ensuite appliquée. Afin de classifier un bloc de pixels donné, l'algorithme considère ses blocs voisins. Ceux-ci sont exprimés relativement à leurs plus proches voisins dans l'espace propre et les coefficients d'interpolation sont réutilisés pour interpoler le bloc courant. Pour finir, le bloc courant est considéré comme fond s'il s'avère proche de son estimation (variation similaire). Bien que naturellement robuste aux faibles variations d'illumination, cet algorithme, sans mise à jour du modèle, semble incapable de gérer des variations plus importantes.
Oliver et al. [Oliver 2000] proposent une approche, nommée "fonds propres", basée elle aussi sur une décomposition sur l'espace propre, mais cette fois-ci au niveau pixel. La phase d'apprentissage ne diffère de la précédente que par le stockage des meilleurs vecteurs propres dans une matrice de vecteurs propres. Lors de l'étape de classification, l'image est d'abord projetée sur cette matrice. Elle est ensuite reprojetée dans l'espace image. Enfin les différences entre l'image initiale et la double projection sont seuillées afin de détecter les pixels changeants (considérés comme

objets). Une fois de plus l'auteur ne mentionne aucune étape de mise à jour du modèle afin de gérer les variations d'illumination.
De nombreuses méthodes actuelles de suivi utilisent la soustraction de fond afin de détecter les régions d'intérêt. En effet, ces méthodes permettent désormais de modéliser les changements d'illumination, le bruit ainsi que les mouvements périodiques des objets du fond. Elles sont, de plus, très efficaces en temps de traitement. Néanmoins ces méthodes sont généralement uniquement adaptées aux caméras fixes, les mouvements de caméra distordant le modèle de fond. Deux techniques de base permettent de gérer les caméras mobiles. Cependant, ces techniques ne permettent de traiter que des scènes planes ou de petits mouvements de caméra. L'utilisation de ces méthodes de soustraction de fond nous est donc impossible, le contexte de mise en oeuvre limité allant à l'encontre de nos objectifs.

1.1.3 Segmentation

Le but des algorithmes de segmentation est de partitionner l'image en régions perceptuellement similaires. On distingue trois grandes familles d'algorithmes de segmentation : les algorithmes basés sur l'approche Mean Shift, les algorithmes utilisant des coupes de graphes et les algorithmes basés sur des contours dits "actifs".
La segmentation basée sur l'algorithme Mean Shift [Cheng 1995], proposée en 2002 par Comaniciu et Meer [Comaniciu 2002b] dans le cadre de l'analyse d'un espace de caractéristiques, permet de trouver les différentes régions dans l'espace joint spatial/colorimétrique. L'algorithme initialise, aléatoirement sur les données, un grand nombre d'hypothèses de position des centres de régions. Ces centres sont ensuite déplacés sur la moyenne des données contenues dans l'ellipsoïde multidimensionnelle centrée en le centre de région et le vecteur de déplacement ainsi défini est appelé vecteur Mean Shift. Finalement, ce vecteur est calculé itérativement jusqu'à stabilité des centres (fusions possibles). Les approches basées Mean Shift se révèlent en général efficaces en termes de ressources de traitement. Cependant elles s'avèrent difficiles à "calibrer" au vu du nombre de paramètres à fixer.
La segmentation peut également être considérée comme un problème de partitionnement de graphes dans lequel les pixels (sommets) sont séparés en plusieurs sous-graphes (régions) disjoints par élagage des arêtes pondérées (poids calculés relativement à la similarité de couleur, d'illumination ou encore de texture séparant les sommets). La somme des poids des arêtes élaguées est alors appelée "coupe".
Wu et Leahy [Wu 1993] tentent d'identifier les partitions minimisant une coupe. Les poids sont calculés relativement à la similarité de couleur. La limitation majeure de cette approche est le biais qu'elle introduit vis à vis de la sursegmentation.
Afin de surmonter ce problème, Shi et Malik [Shi 2000] proposent "la coupe minimum normalisée". Les coupes ne dépendent plus uniquement de la somme des poids des arêtes mais également du rapport entre le poids total de connexion des noeuds de chaque partition à tous les noeuds du graphe. Les poids correspondent ici au produit de la similarité couleur et de la proximité spatiale entre les noeuds. Les ressources nécessaires à la mise en place des méthodes de segmentation par coupes de graphes

sont en général plus importantes que les ressources utilisées par les approches Mean Shift. Cependant elles mettent en oeuvre un nombre plus réduit de paramètres à fixer manuellement.
La segmentation par contours actifs permet de détecter les contours des régions en faisant évoluer un contour jusqu'aux limites des objets. La fonctionnelle d'évolution du contour fait intervenir un ensemble de contraintes de régularisation (basée sur la courbure et la continuité), une énergie basée sur l'apparence dans l'image (calculée globalement ou localement) ainsi que différentes contraintes supplémentaires. Paragios et Deriche [Paragios 2002] ont proposé une énergie image utilisant une combinaison convexe du gradient et d'énergies basées régions afin de compenser les défauts de chaque terme, utilisé auparavant séparément dans la littérature. L'initialisation du contour peut se faire de deux manières différentes suivant l'énergie image choisie : il peut être placé à l'extérieur de l'objet et rétrécir au fur et à mesure pour correspondre aux limites de l'objet ou alors être placé à l'intérieur de l'objet et grandir. Enfin le contour peut être représenté de deux manières : explicitement, par un ensemble de points de contrôle (les relations entre points sont définies par un ensemble d'équations **splines**), ou implicitement, par une grille de distances (chaque valeur de la grille représente la distance du point au contour le plus proche, l'évolution du contour correspondant alors au changement des valeurs de la grille). Le principal défaut des méthodes basées contours actifs est qu'elles requièrent une information a priori sur la position de l'objet. En effet le contour doit être initialisé, suivant la méthode choisie, à l'intérieur ou à l'extérieur de l'objet à segmenter. Le temps de traitement nécessaire à l'utilisation de ces méthodes est très variable puisqu'il dépend de la fonctionnelle d'évolution choisie.
Les méthodes de segmentation sont parfois utilisées dans un contexte du suivi d'objet [Comaniciu 2003] [Paragios 2000] [Xu 2002] [Allili 2009]. Elles peuvent s'avérer très efficaces, peu coûteuses en termes de ressources et très faciles à paramétrer. Cependant ces trois avantages ne sont jamais associés en même temps. De plus la méthode des contours actifs requiert une information a priori sur la position de l'objet. L'utilisation de ces méthodes nous est donc impossible.

1.1.4 Apprentissage supervisé

La détection d'objet peut s'effectuer grâce à l'apprentissage automatique de différentes vues de l'objet à partir d'un ensemble d'exemples au moyen d'un mécanisme d'apprentissage supervisé. Etant donné un ensemble d'exemples d'apprentissage, les méthodes d'apprentissage supervisé génèrent une fonction faisant correspondre les entrées aux sorties souhaitées. Dans la formulation du problème de classification, la forme des sorties peut varier : valeurs continues (on parle alors de régression) ou étiquettes de classe (classification). En détection d'objet, les exemples d'apprentissage sont composés de caractéristiques objet et d'une étiquette de classe associée. Les méthodes d'apprentissage supervisé pouvant servir de base au suivi d'objet sont généralement séparées en deux catégories : les méthodes permettant la construction directe d'une frontière inter-classes (réseaux de neurones, machines à vecteurs

supports, ...) et les méthodes de combinaisons de classifieurs (boosting, bagging, stacking, ...).

1.1.4.1 Construction directe d'une frontière inter-classes

Les méthodes de construction directe d'une frontière inter-classes tentent de construire, directement à partir des exemples d'apprentissage, une fonction permettant de séparer les exemples d'une classe des exemples des autres classes. Les deux méthodes principales de cette catégorie sont les réseaux de neurones et les machines à vecteurs supports.
Les réseaux de neurones peuvent être vus comme des systèmes de calcul massivement parallèles contenant un très grand nombre de processeurs simples inter-connectés suivant une analogie inspirée du fonctionnement de vrais neurones (humains ou non). Le premier modèle de neurone imitant un neurone réel fut développé en 1943 par McCulloch et Pitts [McCulloch 1943]. Il correspond au passage dans une fonction d'activation de la somme pondérée de ses entrées. Les réseaux de neurones sont une organisation particulière (en couches principalement) de ces neurones formels dans laquelle les entrées d'un neurone correspondent aux sorties de la couche précédente. Tout le principe de l'apprentissage est donc de calculer les valeurs des coefficients synaptiques (poids) en fonction des exemples d'apprentissage disponibles.
Les caractéristiques principales des réseaux de neurones sont leur capacité d'apprentissage de relations d'entrée/sortie non linéaires et complexes, leur capacité d'utilisation de procédures d'apprentissage séquentielles ainsi que leur capacité d'adaptation aux données. Cependant ces méthodes souffrent, outre la définition complexe de leur paramètres parfois obscurs, du phénomène dit de sur-apprentissage. Si l'on oblige un réseau à répondre de façon quasi-parfaite relativement à des exemples d'apprentissage biaisés ou bruités (comme c'est le cas généralement), on peut obtenir un réseau basé sur des valeurs erronées. Bien qu'attirantes, ces méthodes semblent donc difficiles à mettre en place relativement à nos objectifs.

Les SVM (Support Vector Machines, Machines à vecteurs supports en français), introduites par Boser et al. [Boser 1992], sont utilisées pour séparer les données en classes en trouvant l'hyperplan marginal maximal qui sépare une classe des autres. La marge de l'hyperplan (que l'on tente de maximiser) est définie comme la distance entre l'hyperplan et les points de données les plus proches de celui-ci. Ce sont ces points qui sont appelés "vecteurs supports". Dans le cas de la détection d'objet, les classes considérées sont la classe "objet" et la classe "non-objet". Malgré l'aspect linéaire de ce classifieur, il est possible d'utiliser les SVM de manière non linéaire en appliquant aux données d'entrée une fonction noyau (en général polynomiale ou à base radiale de type gaussienne ou sigmoïde). Cet artifice est utilisé pour emmener des données d'entrée non linéairement séparables dans un espace de plus grande dimension où la séparabilité est plus probable. L'inconvénient majeur de cet artifice

est la paramétrisation du noyau et plus particulièrement les temps de traitement engendrés par la complexité de ce dernier. En effet, il est bien entendu possible de définir un noyau capable de s'adapter aux différentes observations disponibles. Cependant, la complexité nécessaire à cette définition engendre en général des temps de traitement venant à l'encontre de notre principe d'utilisation temps-réel.

1.1.4.2 Combinaison de classifieurs

Les méthodes de combinaison de classifieurs permettent d'améliorer les performances de classification des classifieurs dits "de base". Un grand nombre de schémas de combinaison a été proposé dans la littérature. Un schéma classique est de combiner les résultats de chaque classifieur pris individuellement pour obtenir la décision finale. De nombreux schémas de combinaison diffèrent de par leur architecture, les caractéristiques de leur algorithme, et la sélection des classifieurs individuels. On distingue cependant trois méthodes de combinaison qui se démarquent : le bagging, le stacking et le boosting adaptatif.

Le bagging [Breiman 1996] est une technique de combinaison de classifieurs basée sur la décomposition de l'ensemble des exemples d'apprentissage en plusieurs sous-ensembles (de taille inférieure ou égale à celle de l'ensemble initial). Etant donné un ensemble de données d'apprentissage, l'algorithme de bagging va construire un certain nombre de nouveaux ensembles par tirages avec remise. Ces nouveaux ensembles vont ensuite être utilisés pour apprendre les différents classifieurs utilisés. Enfin, lors de l'étape de classification, les décisions de chacun de ces classifieurs sont combinées par moyenne (régression) ou par vote (classification). Cet algorithme permet de réduire la variance de classification et d'éviter le problème de sur-apprentissage.
Le stacking [Wolpert 1992] est une méthode de combinaison basée sur la division de l'ensemble des données d'apprentissage en deux sous-ensembles. Le premier sous-ensemble va servir à apprendre les classifieurs utilisés. Les performances de ces classifieurs sur le second sous-ensemble sont ensuite utilisées pour déterminer la manière de les combiner afin d'obtenir de meilleures performances. Cet algorithme peut être vu comme une version plus sophistiquée de la validation croisée, en exploitant une méthode plus sophistiquée que la technique "du vainqueur prend tout" de cette dernière.
Le boosting adaptatif, introduit par Freund et Schapire [Freund 1995], est une méthode itérative de combinaison permettant de construire un classifieur très efficace par combinaison d'un ensemble de classifieurs de base (modérément efficaces). Le boosting construit, de manière incrémentale, un ensemble de classifieurs en entraînant chaque nouveau classifieur de manière à ce qu'il se concentre sur les exemples d'apprentissage mal classés par le classifieur précédent. L'algorithme de boosting adaptatif de référence est l'algorithme Adaboost. La première étape de cet algorithme est de construire une distribution de poids pour l'ensemble des exemples d'apprentissage (poids uniformes initialement). Le mécanisme de boosting tire ensuite aléatoirement un ensemble d'exemples d'apprentissage relativement à cette

distribution de poids et apprend un classifieur de base sur cet ensemble. L'erreur générée par ce classifieur est ensuite utilisée pour modifier la distribution des poids, en se concentrant sur les exemples mal classés. Ainsi le poids des exemples mal classés est augmenté et celui des exemples bien classés est diminué. Enfin le processus est répété jusqu'à atteindre un certain seuil d'erreur d'apprentissage. De cette manière, à chaque itération, un exemple mal classé aura plus de chance de servir à l'apprentissage du nouveau classifieur. La décision de classification de l'ensemble est donnée par la somme pondérée (poids relatif à l'erreur) des décisions de classification de chacun des classifieurs de base construits. La variante réelle [Schapire 1999] de l'algorithme Adaboost est présenté plus en détails dans l'algorithme 1.1.

Entrées : $K = k_{max}$ Un critère d'arrêt (nombre d'itérations)
$D = \{(\mathbf{x_i}, y_i), i \in [1, N]\}$ Un ensemble d'apprentissage contenant
- $\{\mathbf{x_i}, i \in [1, N]\}$ les données d'apprentissage
- $\{y_i, i \in [1, N]\}$ les étiquettes de données (classes)

Sorties : $\{(C_k, \alpha_k), k \in [1, k_{max}]\}$ L'ensemble des paires (classifieur créé, poids du classifieur)

début

Initialisation

Distribution de poids : $W_1(i) = \frac{1}{N}, \forall i \in [1, N]$

Indice de boucle : $k \leftarrow 0$

répéter

$k \leftarrow k + 1$

Apprentissage de C_k en utilisant D échantillonné relativement à W_k

Erreur de C_k : $E_k \leftarrow$ erreur d'apprentissage de C_k mesurée sur D relativement à W_k

Poids de C_k : $\alpha_k \leftarrow \frac{1}{2} \ln[\frac{1-E_k}{E_k}]$

Mise à jour de la distribution de poids :

$$W_{k+1}(i) \leftarrow \frac{W_k(i)}{Z_k} \times \begin{cases} e^{-\alpha_k} & \text{si } h_k(\mathbf{x_i}) = y_i \text{ (correctement classé)} \\ e^{\alpha_k} & \text{si } h_k(\mathbf{x_i}) \neq y_i \text{ (incorrectement classé)} \end{cases}$$

avec Z_k une constante de normalisation et $h_k(\mathbf{x_i})$ l'étiquette retournée par C_k sur la donnée $\mathbf{x_i}$

jusqu'à $k = K$

retourner $\{(C_k, \alpha_k), \forall k \in [1, K]\}$

fin

Algorithme 1.1: Algorithme Adaboost réel

Naturellement le critère de terminaison utilisé peut être remplacé par un seuillage sur l'erreur d'apprentissage globale des classifieurs. La règle de classification globale sur une donnée $\mathbf{x}$ est alors donnée par le signe de :

$$g(\mathbf{x}) = \sum_{k=1}^{k_{max}} \alpha_k h_k(\mathbf{x}) \tag{1.5}$$

Bien qu'un grand k_{max} puisse, en principe, conduire à un sur-apprentissage, les simulations par expérimentation ont démontré que le sur-apprentissage ne survient que très rarement, même lorsque k_{max} est très grand. Tout ceci combiné à la simplicité d'utilisation de la méthode ainsi qu'aux différents résultats obtenus dans de très nombreux domaines d'applications, et notamment la détection d'objets avec les travaux de référence de Viola et Jones [Viola 2001] ou encore les travaux de Grabner et al. [Grabner 2008] ou Stalder et al [Stalder 2009] rendent cette méthode tout à fait adaptée à notre problématique, et ce malgré sa forte sensibilité envers les données marginales.

1.1.5 Méthode choisie

La détection non limitée (non limitée à un type de cible), de même que la détection de piétons, sont des domaines de travail où les formes, les couleurs, les textures, ainsi que toutes les autres caractéristiques d'une même cible (ou d'une cible à une autre) sont très diverses. De la même manière, les environnements de détection de ces cibles sont totalement hétérogènes. Ces disparités rendent la modélisation des objets à suivre par un ensemble de points très difficile à mettre en place.
Bien que certaines méthodes actuelles de soustraction de fond permettent de modéliser le fond sur une caméra mobile, elles n'en restent pas moins limitées à un contexte d'utilisation très précis (scène plane, petits mouvements de caméra). Ces limitations, et ce malgré des temps de traitement très intéressants, rendent leur utilisation impossible dans notre cas. En effet, les mouvements de caméra étant asservis sur la vitesse de déplacement de l'objet suivi (piéton par exemple), l'hypothèse de mouvements lents n'est pas applicable.
Enfin, les méthodes de segmentation possèdent trois avantages majeurs non négligeables. Cependant, aucune des méthodes présentées ne permet de bénéficier de ces trois avantages simultanément. Leur utilisation engendrerait donc soit une difficulté de paramétrage très difficilement contournable (chaque paramétrage s'avèrerait inefficace sur une autre cible), soit des ressources (temps et mémoire) trop élevées relativement à nos objectifs, soit une efficacité modérée (ce qui n'est évidemment pas le but), soit enfin la nécessité d'utilisation d'une information a priori impossible à obtenir.
Le choix de méthode s'est donc tout naturellement porté sur les méthodes d'apprentissage supervisé et plus particulièrement sur les méthodes de combinaison au vu des différents résultats d'utilisation de ces méthodes dans notre domaine d'application. En terme de détection/suivi d'objets, les résultats les plus probants ont été observés avec la méthode de boosting adaptatif et plus particulièrement son algorithme de référence Adaboost. Outre ces résultats, la méthode s'avère très simple d'utilisation et très rarement affectée par le phénomène de sur-apprentissage. Nous nous sommes donc concentrés sur l'algorithme Adaboost et plus particulièrement sur une méthode de référence de la littérature utilisant Adaboost : l'Ensemble Tracking, développé en 2007 par Shaï Avidan [Avidan 2007]. Cette méthode, présentée plus en détails dans le chapitre 2, considère le suivi comme un problème de classification binaire.

Adaboost est utilisé pour construire un ensemble de classifieurs capable de discerner l'objet du fond. Cet ensemble est mis à jour au fil de la séquence.
La méthode de détection est la base de tout algorithme de suivi d'objet. Elle est en général utilisée en complément d'une méthode de suivi assurant la cohérence temporelle.

1.2 Suivi d'objet

Le but du suivi est de générer la trajectoire d'un objet dans le temps en déterminant sa position dans chaque image d'une séquence vidéo. Le suivi peut aussi fournir la région de l'image occupée par l'objet à chaque instant. Les tâches de détection de l'objet et de mise en correspondance des instances détectées (suivi) peuvent être conduites séparément ou de manière jointe. Dans le cas de tâches séparées, l'algorithme de détection fournit un ensemble de régions possibles et l'algorithme de suivi fait correspondre ces détections. Dans le cas de tâches jointes, la région objet et la mise en correspondance sont estimées conjointement par mise à jour itérative de la position objet et de l'information région obtenues dans les images précédentes. On distingue, de la même manière que pour les méthodes de détection, quatre ensembles d'algorithmes de suivi : le suivi de noyau, le suivi de silhouette, le suivi de points déterministe et enfin le suivi de points probabiliste.

1.2.1 Suivi de noyau

Le suivi de noyau s'effectue en calculant le mouvement de l'objet, représenté par une primitive, d'une image sur l'autre. Il existe un grand nombre de méthodes de suivi de noyau. Celles-ci diffèrent dans la forme du mouvement calculé (paramétrique, champ de déplacement dense, ...), dans leur représentation de l'apparence de l'objet, dans le nombre d'objets suivis ou encore dans la méthode utilisée pour estimer le mouvement. On peut cependant regrouper ces méthodes en deux catégories : les méthodes basées sur des modèles génériques ou des modèles basés sur une densité de probabilité et les méthodes basées sur une représentation multi-vues de l'objet.
Les méthodes basées sur des modèles génériques ou des modèles basés sur une densité de probabilité ont été largement utilisés par le passé du fait de leur simplicité et de leur faible coût calculatoire.
Bien que le coût de la mise en correspondance brute de modèles (génériques ou basés sur des histogrammes de couleurs ou des modèles à base de mixtures) soit possible à réduire, celui-ci n'en reste pas moins très élevé, rendant ce type de méthode inutilisable.
Au lieu d'utiliser une méthode de force brute, Comaniciu et Meer [Comaniciu 2003] (respectivement Comaniciu [Comaniciu 2002a]) ont utilisé l'algorithme Mean-Shift(cf. section 1.1.3) afin de maximiser itérativement la similarité d'apparence (mesurée par distance de Bhattacharya) entre l'histogramme de couleurs (resp. l'histogramme joint espace/couleurs) pondéré représentant l'objet et l'histogramme correspondant représentant la position hypothétique. L'avantage de cette méthode est la réduction considérable du temps d'estimation de la position objet.
Shi et Tomasi [Shi 1994], quant à eux, se sont basés sur une méthode similaire à la construction du flot optique introduite par Lucas et Kanade [Lucas 1981] pour proposer le suivi KLT. Celui-ci calcule itérativement la translation d'une région centrée en un point d'intérêt. La qualité de la nouvelle position obtenue est mesurée en

calculant les paramètres de la transformation affine permettant de passer de l'ancienne région à la nouvelle. Le point d'intérêt n'est alors conservé que si la somme des carrés des différences entre la région courante et sa projection est petite.
Afin de pallier le problème d'invalidité du modèle lorsque la vue de l'objet est modifiée de manière très importante, les méthodes basées sur une représentation multivues de l'objet construisent leur modèle par apprentissage d'un ensemble de vues différentes de ce dernier.
Black et Jepson [Black 1998] ont proposé en 1998 une méthode basée sur un sous-espace propre afin de calculer la transformation affine permettant de passer de l'objet dans l'image courante à l'objet dans l'image reconstruite à partir des vecteurs propres. Le sous-espace de représentation de l'apparence de l'objet est calculé par analyse en composantes principales et la transformation par minimisation de la différence entre l'image d'entrée et l'image reconstruite. Le suivi est effectué en estimant itérativement les paramètres de la transformation qui rendent la différence d'images minimale.
Avidan [Avidan 2004] a, quant à lui, utilisé une machine à vecteurs supports (cf. chapitre 1.1.4.1). La classe des exemples positifs (objet) est peuplée d'images de l'objet à suivre alors que la classe des négatifs est peuplée d'images de fond (régions pouvant être confondues avec l'objet). La méthode de suivi d'Avidan consiste alors à maximiser le score de classification du SVM sur des régions de l'image afin d'estimer la position de l'objet. Ce type d'approche intègre explicitement la connaissance du fond de la scène au sein du suivi.

1.2.2 Suivi de silhouette

Les méthodes basées silhouette modélisent de manière efficace les objets à la forme complexe en utilisant l'information encodée à l'intérieur de la région objet. Leur but est de déterminer la région occupée par l'objet au moyen d'un modèle généré en utilisant les images précédentes. On distingue deux types de méthodes de suivi de silhouette : la correspondance de formes et le suivi de contours.
Les méthodes de correspondance de formes se basent sur la recherche effective de l'objet dans l'image courante. La recherche est effectuée en calculant la similarité entre l'objet et le modèle de silhouette hypothétique basé sur les images précédentes, la mise à jour du modèle permettant de prendre en compte les changements d'apparence. Huttenlocher et al. [Huttenlocher 1993] ont proposé une représentation basée sur les crêtes afin de parvenir à une correspondance de formes. La distance de Hausdorff (mesurant ici les crêtes les moins correspondantes) est utilisée pour construire une surface de corrélation dont le minimum est considéré comme la nouvelle position de l'objet.
La correspondance de formes peut aussi s'effectuer en calculant la distance qui sépare les modèles objets associés aux silhouettes détectées dans deux images consécutives. Ces modèles sont généralement sous la forme d'une fonction de densité, d'un contour objet, d'un ensemble de crêtes objet ou un mélange de ces modèles.
Kang et al. [Kang 2004] ont utilisé des histogrammes de couleurs et de crêtes

construits à partir de cercles concentriques de rayon différent centrés en un ensemble de points de contrôle pris sur un cercle de référence, ce qui permet l'encodage implicite d'une information spatiale. Les histogrammes résultants sont invariants à de nombreuses transformations et le score de correspondance est calculé par distance de Bhattacharya et divergence de Kullback-Leibler.
Le suivi de silhouette peut également s'effectuer par calcul du vecteur de flot de chaque pixel à l'intérieur de la silhouette. Le flot dominant est alors utilisé pour générer la trajectoire de la silhouette.
Sato et Aggarwal [Sato 2004] ont proposé de générer la trajectoire des silhouettes en appliquant une transformée de Hough dans l'espace des vitesses aux silhouettes objets dans les images consécutives. Une fenêtre spatio-temporelle autour de chaque région en mouvement est utilisée pour appliquer la transformée de Hough dans le but de calculer deux matrices de flot (vertical et horizontal). Ces matrices interviennent dans la composition d'une image 4D (position, matrices) appelée TSV (Temporal Spatio-Velocity) encodant le mouvement principal d'une région en mouvement ainsi que la vraisemblance de ce mouvement. Il s'agit ici d'une mise en correspondance de mouvements.
Outre les méthodes de mise en correspondance (de formes, de mouvements), le suivi de silhouette peut être effectué via un ensemble de méthodes dîtes de suivi de contours. Ces méthodes font évoluer un contour initial dans l'image précédente vers sa nouvelle position dans la nouvelle image, cette évolution nécessitant l'existence d'un chevauchement de la région objet d'une image à l'autre.
Différentes méthodes se basent sur des modèles d'espace d'état. L'état d'un objet est défini en terme de forme et de paramètres de mouvement du contour, l'état étant mis à jour de manière à maximiser la probabilité a posteriori du contour.
Isard et Blake [Isard 1998] (resp. MacCormick et Blake [MacCormick 2000]) définissent l'état de l'objet (resp. des objets) en termes de paramètres de forme de spline et de paramètres de mouvement affine. Les mesures consistent en un ensemble de crêtes calculé le long de la normale au contour. Enfin l'état est mis à jour via un filtre à particules (resp. un filtre à particules intégrant un principe d'exclusion multi-cibles afin de gérer les occlusions) (cf. section 1.2.4.4).
Enfin, d'autres méthodes font évoluer le contour en minimisant l'énergie de ce dernier (énergie représentée en termes d'informations temporelles de type gradient temporel ou statistiques d'apparence).
Bertalmio et al. [Bertalmio 2000] ont utilisé la contrainte de cohérence d'illumination du flot optique pour faire évoluer le contour dans les images successives en utilisant une représentation par courbes de niveau. Les auteurs utilisent deux fonctionnelles énergétiques : une fonctionnelle de suivi minimisée simultanément à une fonctionnelle de modelage de l'intensité qui minimise les changements d'intensité d'une image à la suivante.
Yilmaz et al. [Yilmaz 2004] ont modélisé la forme de l'objet ainsi que ses changements au moyen d'un modèle de forme basé sur des courbes de niveau. Dans ce modèle, les points des courbes de niveau contiennent les moyennes et écarts-types des distances des points aux contours de l'objet.

1.2.3 Suivi de points déterministe

Le problème de suivi peut être formulé comme un problème de mise en correspondance d'objets représentés par des points d'une image sur l'autre. On distingue deux types de suivi de points : le suivi déterministe et le suivi probabiliste. Les méthodes déterministes définissent un coût d'association entre les objets à l'image précédente et chaque objet unique à l'image courante, le but étant bien entendu de minimiser le coût total engendré. Le coût est en général défini comme une combinaison de contraintes de type proximité, rigidité, mouvement commun, vitesse maximale, ...
Salari et Sethi [Salari 1990] résolvent, dans un premier temps, le problème de correspondance entre les points détectés par une approche gloutonne basée sur les contraintes de proximité et de rigidité et étendent ensuite le suivi des objets manquants par l'ajout de nouveaux points hypothétiques.
Veenman et al. [Veenman 2001] ont introduit en 2001 la contrainte de mouvement commun. Celle-ci fournit une contrainte forte afin de suivre de manière cohérente un ensemble de points appartenant au même objet. L'algorithme est initialisé par la génération de la trajectoire initiale en utilisant un algorithme à deux passes et la fonction de coût est minimisée par l'algorithme d'assignation hongrois sur deux images successives.
Shafique et Shah [Shafique 2003] ont proposé une approche multi-images afin de préserver la cohérence temporelle de la vitesse et de la position des points. Le problème de correspondance est défini comme un problème de théorie des graphes dans lequel la problématique devient la détection du meilleur chemin (à travers la séquence d'images) unique pour chaque point (avec positions manquantes en cas d'occlusion ou de mauvaise détection). Pour résoudre ce problème, les auteurs utilisent un algorithme glouton.

1.2.4 Suivi de points probabiliste

Au vu des différents arguments présentés en section 1.2.5, les méthodes de suivi de points probabiliste ont retenu notre attention. Par conséquent, et afin de pouvoir réutiliser par la suite les notations, les sections ci-dessous présentent en détail le vocabulaire, le contexte général et bien entendu le principe de ces méthodes.

1.2.4.1 Vocabulaire et contexte général

Les méthodes de suivi de points probabilistes se basent sur la prise en compte des incertitudes de mesure et de modélisation pour effectuer le suivi. Elles utilisent l'approche espace d'état pour modéliser les propriétés de l'objet. Ces méthodes introduisent les notations suivantes :

Définition 1.2 *On appelle* ***vecteur d'état****, noté* $\mathbf{X}_k$ *à l'instant* k*, le vecteur contenant toutes les informations relatives à la description du système (par ex. les caractéristiques cinématiques).*

Définition 1.3 *On appelle* ***vecteur de mesure*** *ou* ***d'observation****, noté* $\mathbf{Z}_k$ *à l'instant* k*, le vecteur représentant l'observation bruitée de l'état du système.*

Définition 1.4 *On appelle* ***modèle du système****, défini par la fonction notée* $\mathbf{f}_k$ *à l'instant* k*, le modèle décrivant l'évolution de l'état au cours du temps (ce modèle est généralement disponible sous forme probabiliste).*

Définition 1.5 *On appelle* ***modèle de mesure*** *ou* ***modèle d'observation****, défini par la fonction notée* $\mathbf{h}_k$ *à l'instant* k*, le modèle permettant de relier les observations bruitées à l'état (ce modèle est, lui aussi, généralement disponible sous forme probabiliste).*

L'approche Bayésienne tente de construire la fonction de densité de probabilité postérieure de l'état (solution complète au problème d'estimation) en fonction de l'ensemble des observations reçues. Une approche par filtrage récursif signifie que les données peuvent être traitées séquentiellement plutôt que par lot. Il n'est donc pas nécessaire de stocker toutes les données, ni de retraiter les données existantes à l'arrivée d'une nouvelle observation. Un tel filtre comporte deux étapes :

Définition 1.6 *On appelle* ***étape de prédiction*** *l'étape permettant, à partir du modèle du système, de prédire la fonction de densité de probabilité d'état d'une observation à l'observation suivante.*

Définition 1.7 *On appelle* ***étape de mise à jour*** *l'étape permettant, à partir de la dernière observation, de modifier la fonction de densité de probabilité prédite. Cette étape est réalisée grâce au théorème de Bayes.*

Le problème de suivi considère l'évolution de la séquence des états de la cible $\{\mathbf{X}_k, k \in \mathbb{N}\}$ donnée par :

$$\mathbf{X}_k = \mathbf{f}_k(\mathbf{X}_{k-1}, \mathbf{V}_{k-1}) \tag{1.6}$$

où $\mathbf{f}_k : \mathbb{R}^{n_X} \times \mathbb{R}^{n_V} \to \mathbb{R}^{n_X}$ est une fonction possiblement non linéaire de l'état $\mathbf{X}_{k-1}$, $\{\mathbf{V}_{k-1}, k \in \mathbb{N}\}$ représente la séquence des bruits de traitement et n_X (resp. n_V) est la dimension du vecteur d'état (resp. de bruit). L'objectif du suivi est d'estimer récursivement $\mathbf{X}_k$ à partir des observations :

$$\mathbf{Z}_k = \mathbf{h}_k(\mathbf{X}_k, \mathbf{N}_k) \tag{1.7}$$

où $\mathbf{h}_k : \mathbb{R}^{n_X} \times \mathbb{R}^{n_N} \to \mathbb{R}^{n_Z}$ est une fonction possiblement non linéaire, $\{\mathbf{N}_k, k \in \mathbb{N}\}$ représente la séquence des bruits de mesure et n_Z (resp. n_N) est la dimension du vecteur de mesure (resp. de bruit). En particulier, on recherche des estimés filtrés de $\mathbf{X}_k$ basés sur l'ensemble des mesures disponibles $\mathbf{Z}_{1:k} = \{\mathbf{Z}_i, i = 1, ..., k\}$.
Du point de vue Bayésien, le problème de suivi est de calculer récursivement un certain degré de confiance sur l'état $\mathbf{X}_k$ à l'instant k. Il est donc nécessaire de construire la fonction de densité de probabilité $p(\mathbf{X}_k|\mathbf{Z}_{1:k})$. On suppose, bien entendu, la fonction initiale $p(\mathbf{X}_0|\mathbf{Z}_0) \equiv p(\mathbf{X}_0)$ connue. La fonction de densité de

probabilité $p(\mathbf{X}_k|\mathbf{Z}_{1:k})$ peut alors, en principe, être obtenue récursivement grâce aux deux étapes décrites en Définition 1.6 et Définition 1.7.
On suppose la fonction à $k-1$, $p(\mathbf{X}_{k-1}|\mathbf{Z}_{1:k-1})$, disponible. L'étape de prédiction implique d'utiliser le modèle système (1.6) pour obtenir la fonction de densité de probabilité a priori de l'état à l'instant k via l'équation de Chapman-Kolmogorov :

$$p(\mathbf{X}_k|\mathbf{Z}_{1:k-1}) = \int p(\mathbf{X}_k|\mathbf{X}_{k-1})p(\mathbf{X}_{k-1}|\mathbf{Z}_{1:k-1})d\mathbf{X}_{k-1} \tag{1.8}$$

Le modèle probabiliste d'évolution de l'état $p(\mathbf{X}_k|\mathbf{X}_{k-1})$ est défini par l'équation système (1.6) et la statistique connue de $\mathbf{V}_{k-1}$.
A l'instant k, la nouvelle observation $\mathbf{Z}_k$ peut être utilisée pour mettre à jour la fonction de densité de probabilité a priori (étape de mise à jour) via la règle de Bayes :

$$p(\mathbf{X}_k|\mathbf{Z}_{1:k}) = \frac{p(\mathbf{Z}_k|\mathbf{X}_k)p(\mathbf{X}_k|\mathbf{Z}_{1:k-1})}{p(\mathbf{Z}_k|\mathbf{Z}_{1:k-1})} \tag{1.9}$$

où la constante de normalisation

$$p(\mathbf{Z}_k|\mathbf{Z}_{1:k-1}) = \int p(\mathbf{Z}_k|\mathbf{X}_k)p(\mathbf{X}_k|\mathbf{Z}_{1:k-1})d\mathbf{X}_k \tag{1.10}$$

dépend de la fonction de vraisemblance, $p(\mathbf{Z}_k|\mathbf{X}_k)$, définie par le modèle de mesure (1.7) et la statistique connue de $\mathbf{N}_k$.
Les relations de récurrence (1.8) et (1.9) forment la base de la solution optimale Bayésienne. Cette propagation récursive de la densité a posteriori n'est, cependant, qu'une solution conceptuelle dans le sens où, en général, elle ne peut être déterminée analytiquement.
On distingue, à ce niveau, deux types d'algorithmes : les algorithmes dits optimaux qui permettent de déterminer une telle solution sous contraintes restrictives et les algorithmes dits sous-optimaux qui permettent d'approximer cette solution lorsque celle-ci est intraitable.

1.2.4.2 Méthodes optimales

Le filtre de Kalman [Kalman 1960] se base sur plusieurs suppositions :
- $p(\mathbf{X}_k|\mathbf{Z}_k)$ est gaussienne à chaque instant k et donc paramétrée par une moyenne et une covariance.
- $\mathbf{V}_{k-1}$ et $\mathbf{N}_k$ sont tirés relativement à des distributions gaussiennes de paramètres connus.
- $\mathbf{f}_k(\mathbf{X}_{k-1}, \mathbf{V}_{k-1})$ est une fonction linéaire de $\mathbf{X}_{k-1}$ et $\mathbf{V}_{k-1}$ connue.
- $\mathbf{h}_k(\mathbf{X}_k, \mathbf{N}_k)$ est une fonction linéaire de $\mathbf{X}_k$ et $\mathbf{N}_k$ connue.

Les modèles (1.6) et (1.7) peuvent alors être réécrits :

$$\mathbf{X}_k = F_k\mathbf{X}_{k-1} + \mathbf{V}_{k-1} \tag{1.11}$$

$$\mathbf{Z}_k = H_k\mathbf{X}_k + \mathbf{N}_k \tag{1.12}$$

avec F_k et H_k des matrices connues définissant les fonctions linéaires et $\mathbf{V}_{k-1}$ et $\mathbf{N}_k$ statistiquement indépendants, de moyenne nulle et de covariance respective Q_{k-1} et R_k.
L'algorithme du filtre de Kalman peut être vue comme la relation de récurrence suivante :

$$\left\{ \begin{array}{rcl} p(\mathbf{X}_{k-1}|\mathbf{Z}_{1:k-1}) & = & \mathcal{N}(\mathbf{X}_{k-1}; m_{k-1|k-1}; P_{k-1|k-1}) \\ p(\mathbf{X}_k|\mathbf{Z}_{1:k-1}) & = & \mathcal{N}(\mathbf{X}_k; m_{k|k-1}; P_{k|k-1}) \\ p(\mathbf{X}_k|\mathbf{Z}_{1:k}) & = & \mathcal{N}(\mathbf{X}_k; m_{k|k}; P_{k|k}) \end{array} \right. \tag{1.13}$$

où

$$\left\{ \begin{array}{rcl} m_{k|k-1} & = & F_k m_{k-1|k-1} \\ P_{k|k-1} & = & F_k P_{k-1|k-1} F_k^T + Q_{k-1} \\ m_{k|k} & = & m_{k|k-1} + K_k(\mathbf{Z}_k - H_k m_{k|k-1}) \\ P_{k|k} & = & P_{k|k-1} - K_k H_k P_{k|k-1} \end{array} \right. \tag{1.14}$$

et où $\mathcal{N}(X; m; P)$ représente une densité de distribution gaussienne de paramètre X, de moyenne m et de covariance P. Enfin :

$$\left\{ \begin{array}{rcl} S_k & = & H_k P_{k|k-1} H_k^T + R_k \\ K_k & = & P_{k|k-1} H_k^T S_k^{-1} \end{array} \right. \tag{1.15}$$

représentent respectivement la covariance du terme d'innovation $\mathbf{Z}_k - H_k m_{k|k-1}$ et le gain de Kalman.
La figure 1.1 illustre l'algorithme de construction des paramètres du filtre de Kalman et la table 1.1 reprend le rôle de chacune des variables et chacun des paramètres que ce dernier introduit.

Variable/Paramètre	Rôle	Dimensions
$\mathbf{X}_k$	État	$n_X \times 1$
$\mathbf{Z}_k$	Observation	$n_Z \times 1$
F_k	Gain d'état	$n_X \times n_X$
H_k	Gain de sortie	$n_Z \times n_X$
$\mathbf{V}_{k-1}$	Bruit de traitement	$n_X \times 1$
Q_{k-1}	Covariance des bruits de traitement	$n_X \times n_X$
$\mathbf{N}_k$	Bruit de mesure	$n_Z \times 1$
R_k	Covariance des bruits de mesure	$n_Z \times n_Z$
$P_{k\|k-1}$	Covariance a priori	$n_X \times n_X$
$P_{k\|k}$	Covariance a posteriori	$n_X \times n_X$
S_k	Covariance du terme d'innovation	$n_Z \times n_Z$
K_k	Gain de Kalman	$n_X \times n_Z$

TABLE 1.1 – Variables et paramètres du filtre de Kalman.

Le filtre de Kalman a été très largement utilisé dans un contexte de suivi. On peut citer par exemple les travaux de Broida et Chellappa [Broida 1986] dans lesquels les auteurs utilisent le filtre pour suivre un ensemble de points dans des images bruitées.

Les méthodes basées grille fournissent la récursion optimale lorsque l'espace d'états est discret et consiste en un nombre fini d'états. Supposons que l'espace d'états à

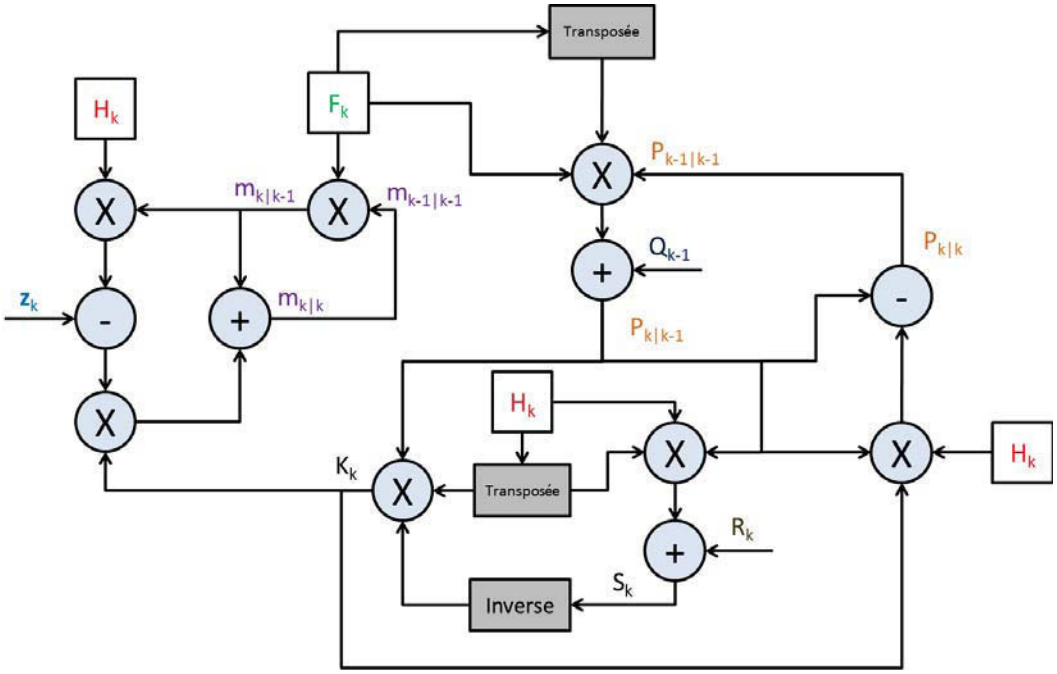


Figure 1.1 – Relations de récurrence du filtre de Kalman.

l'instant $k-1$ se compose des états discrets $\mathbf{X}_{k-1}^i, i = 1, ..., N_s$. Pour chaque $\mathbf{X}_{k-1}^i$, on note la probabilité conditionnelle de l'état étant données toutes les observations jusque $k-1$, $Pr(\mathbf{X}_{k-1} = \mathbf{X}_{k-1}^i | \mathbf{Z}_{1:k-1}) = w_{k-1|k-1}^i$. La fonction de densité de probabilité a posteriori à l'instant $k-1$ peut s'écrire :

$$p(\mathbf{X}_{k-1}|\mathbf{Z}_{1:k-1}) = \sum_{i=1}^{N_s} w_{k-1|k-1}^i \delta(\mathbf{X}_{k-1} - \mathbf{X}_{k-1}^i) \tag{1.16}$$

où $\delta(.)$ représente la mesure de Dirac. Les équations de prédiction et de mise à jour deviennent alors :

$$\left\{ \begin{array}{rcl} p(\mathbf{X}_k|\mathbf{Z}_{1:k-1}) & = & \sum_{i=1}^{N_s} w_{k|k-1}^i \delta(\mathbf{X}_k - \mathbf{X}_k^i) \\ p(\mathbf{X}_k|\mathbf{Z}_{1:k}) & = & \sum_{i=1}^{N_s} w_{k|k}^i \delta(\mathbf{X}_k - \mathbf{X}_k^i) \end{array} \right. \tag{1.17}$$

où

$$\left\{ \begin{array}{rcl} w_{k|k-1}^i & \triangleq & \sum_{j=1}^{N_s} w_{k-1|k-1}^j p(\mathbf{X}_k^i|\mathbf{X}_{k-1}^j) \\ w_{k|k}^i & \triangleq & \frac{w_{k|k-1}^i p(\mathbf{Z}_k|\mathbf{X}_k^i)}{\sum_{j=1}^{N_s} w_{k|k-1}^j p(\mathbf{Z}_k|\mathbf{X}_k^j)} \end{array} \right. \tag{1.18}$$

1.2.4.3 Méthodes sous-optimales

Lorsque la linéarité des fonctions de (1.6) et (1.7) n'est pas vérifiée, une linéarisation locale des équations peut être une description suffisante de la non-linéarité. Le filtre de Kalman étendu se base sur ce constat en utilisant le premier terme dans la décomposition de Taylor des fonctions non-linéaires. Rosales et Sclaroff [Rosales 1999]

ont utilisé ce type de filtre pour estimer la trajectoire 3D d'un objet à partir d'un mouvement 2D.
Plus récemment est apparu le filtre de Kalman "Unscented" [Wan 2000]. Celui-ci considère un ensemble de points sélectionnés de manière déterministe selon une approximation gaussienne de $p(\mathbf{X}_k|\mathbf{Z}_{1:k})$. Ces points sont alors propagés via la non-linéarité réelle et les paramètres de l'approximation gaussienne sont alors ré-estimés.
Lorsque l'espace d'état est continu mais peut être décomposé en cellules, une méthode basée grille peut être utilisée pour approximer la densité a posteriori. Incorporant aux équations présentées dans la méthode optimale l'intégration des probabilités sur les différentes cellules, la méthode est en général simplifiée et l'évaluation des poids n'est effectuée qu'au centre de chaque cellule. Les modèles de Markov cachés sont une application de ce type de méthode.

1.2.4.4 Cas non gaussien : le filtrage particulaire

L'algorithme séquentiel d'échantillonnage par importance (noté en général SIS pour Sequential Importance Sampling et aussi connu sous le nom de filtrage particulaire [Carpenter 1999]) est une technique d'amélioration du filtre Bayésien récursif par simulations de Monte-Carlo. L'idée est de représenter la fonction de densité de probabilité a posteriori par un ensemble d'échantillons aléatoires associés à un poids et de calculer les estimés à partir de ces échantillons pondérés.
Soit $\left\{\mathbf{X}^i_{0:k}, w^i_k\right\}_{i=1}^{N_s}$ une mesure aléatoire caractérisant la fonction de densité de probabilité a posteriori $p(\mathbf{X}_{0:k}|\mathbf{Z}_{1:k})$ avec $\left\{\mathbf{X}^i_{0:k}, i = 1, ..., N_s\right\}$ l'ensemble des points supports de poids associés $\left\{w^i_k, i = 1, ..., N_s\right\}$ et $\mathbf{X}_{0:k} = \{\mathbf{X}_j, j = 0, ..., k\}$ l'ensemble de tous les états jusqu'à l'instant k. Les poids sont normalisés de sorte que $\sum_i w^i_k = 1$.
La densité a posteriori à l'instant k peut alors être approximée par

$$p(\mathbf{X}_{0:k}|\mathbf{Z}_{1:k}) \approx \sum_{i=1}^{N_s} w^i_k \delta(\mathbf{X}_{0:k} - \mathbf{X}^i_{0:k}) \tag{1.19}$$

Les poids sont choisis selon le principe de l'échantillonnage par importance [Doucet 2000]. Ce principe se base sur le constat suivant : Soit $p(X) \propto \pi(X)$ une fonction de densité de probabilité à partir de laquelle il est difficile d'obtenir des échantillons mais pour laquelle $\pi(X)$ peut être facilement évalué. Soit $X^i \sim q(X), i = 1, ..., N_s$ des échantillons facilement générés à partir d'une proposition $q(.)$ appelée **densité d'importance**. Alors, une approximation pondérée de la densité $p(.)$ est donnée par

$$\left\{ \begin{array}{rcl} p(X) & \approx & \sum_{i=1}^{N_s} w^i \delta(X - X^i) \\ w^i & \propto & \frac{\pi(X^i)}{q(X^i)} \end{array} \right. \tag{1.20}$$

où w^i représente le poids normalisé de la particule i.
Par conséquent, si les échantillons $\mathbf{X}^i_{0:k}$ sont tirés à partir d'une densité d'importance $q(\mathbf{X}_{0:k}|\mathbf{Z}_{1:k})$ alors les poids seront définis comme

$$w^i_k \propto \frac{p(\mathbf{X}^i_{0:k}|\mathbf{Z}_{1:k})}{q(\mathbf{X}^i_{0:k}|\mathbf{Z}_{1:k})} \tag{1.21}$$

Pour en revenir au cas séquentiel, à chaque itération, il est possible d'obtenir des échantillons servant d'approximation pour $p(\mathbf{X}_{0:k-1}|\mathbf{Z}_{1:k-1})$. On souhaite alors approximer $p(\mathbf{X}_{0:k}|\mathbf{Z}_{1:k})$ avec un nouvel ensemble d'échantillons. Si la densité d'importance est choisi de sorte qu'elle puisse être factorisée de la manière suivante : $q(\mathbf{X}_{0:k}|\mathbf{Z}_{1:k}) = q(\mathbf{X}_k|\mathbf{X}_{0:k-1}, \mathbf{Z}_{1:k})q(\mathbf{X}_{0:k-1}|\mathbf{Z}_{1:k-1})$ alors il est possible d'obtenir des échantillons $\mathbf{X}^i_{0:k} \sim q(\mathbf{X}_{0:k}|\mathbf{Z}_{1:k})$ en augmentant les échantillons existants $\mathbf{X}^i_{0:k-1} \sim q(\mathbf{X}_{0:k-1}|\mathbf{Z}_{1:k-1})$ du nouvel état $\mathbf{X}^i_k \sim q(\mathbf{X}_k|\mathbf{X}_{0:k-1}, \mathbf{Z}_{1:k})$.
Si, de plus, $q(\mathbf{X}_k|\mathbf{X}_{0:k-1}, \mathbf{Z}_{1:k}) = q(\mathbf{X}_k|\mathbf{X}_{k-1}, \mathbf{Z}_k)$ alors la densité d'importance ne devient plus dépendante que de $\mathbf{X}_{k-1}$ et $\mathbf{Z}_k$. Dans de tels scénarios, il n'est plus nécessaire de stocker le chemin $\mathbf{X}^i_{0:k-1}$, ni l'historique des observations $\mathbf{Z}_{1:k-1}$. La densité a posteriori filtrée $p(\mathbf{X}_k|\mathbf{Z}_{1:k})$ peut alors être approximée par

$$\begin{cases} p(\mathbf{X}_k|\mathbf{Z}_{1:k}) & \approx & \sum_{i=1}^{N_s} w^i_k \delta(\mathbf{X}_k - \mathbf{X}^i_k) \\ w^i_k & \propto & w^i_{k-1} \frac{p(\mathbf{Z}_k|\mathbf{X}^i_k)p(\mathbf{X}^i_k|\mathbf{X}^i_{k-1})}{q(\mathbf{X}^i_k|\mathbf{X}^i_{k-1}, \mathbf{Z}_k)} \end{cases} \tag{1.22}$$

Il peut être montré que si $N_s \to \infty$ alors cette approximation approche la densité a posteriori $p(\mathbf{X}_k|\mathbf{Z}_{1:k})$ réelle.
Doucet et al. [Doucet 2000] ont démontré que la variance des poids ne peut qu'augmenter au fil du temps. Ce constat, appelé phénomène de **dégénérescence**, implique qu'après quelques itérations, toutes les particules exceptée une ont un poids négligeable. La technique de force brute pour réduire cet effet est d'utiliser un N_s très grand, ce qui n'est souvent pas applicable. Deux autres techniques ont donc été développées : le bon choix de la densité d'importance et le ré-échantillonnage.
Doucet et al. [Doucet 2000] ont montré que la densité d'importance optimale est :

$$\begin{cases} q(\mathbf{X}_k|\mathbf{X}^i_{k-1}, \mathbf{Z}_k)_{opt} & = & \frac{p(\mathbf{Z}_k|\mathbf{X}_k, \mathbf{X}^i_{k-1})p(\mathbf{X}_k|\mathbf{X}^i_{k-1})}{p(\mathbf{Z}_k|\mathbf{X}^i_{k-1})} \\ w^i_k & \propto & w^i_{k-1} \int p(\mathbf{Z}_k|\mathbf{X}'_k)p(\mathbf{X}'_k|\mathbf{X}^i_{k-1})d\mathbf{X}'_k \end{cases} \tag{1.23}$$

Bien qu'optimale, cette densité n'en est pas moins sans défaut puisqu'elle nécessite de pouvoir échantillonner à partir de $p(\mathbf{X}_k|\mathbf{X}^i_{k-1}, \mathbf{Z}_k)$ et évaluer l'intégrale sur le nouvel état, ce qui n'est pas évident a priori. Par conséquent, il est souvent plus simple et pratique d'utiliser directement la densité précédente :

$$\begin{cases} q(\mathbf{X}_k|\mathbf{X}^i_{k-1}, \mathbf{Z}_k) & = & p(\mathbf{X}_k|\mathbf{X}^i_{k-1}) \\ w^i_k & \propto & w^i_{k-1} p(\mathbf{Z}_k|\mathbf{X}^i_k) \end{cases} \tag{1.24}$$

sachant cependant que de nombreuses autres densités peuvent être utilisées et que ce choix représente l'étape cruciale de conception du filtre.
L'idée principale du ré-échantillonnage est d'éliminer les particules de poids faible et de se concentrer sur celles de poids fort. Ceci implique la génération de nouveaux échantillons $\{\mathbf{X}^{i\,*}_k\}_{i=1}^{N_s}$ grâce à N_s ré-échantillonnages (avec remise) successifs à partir d'une représentation approximée discrète de $p(\mathbf{X}_k|\mathbf{Z}_{1:k})$ donnée par

$$p(\mathbf{X}_k|\mathbf{Z}_{1:k}) \approx \sum_{i=1}^{N_s} w^i_k \delta(\mathbf{X}_k - \mathbf{X}^i_k) \tag{1.25}$$

et telle que $Pr(\mathbf{X}_k^{i\,*} = \mathbf{X}_k^j) = w_k^j$. Chaque échantillon est alors repondéré avec le poids $w_k^i = \frac{1}{N_s}$.
L'algorithme SIS présenté dans 1.2.4.4 représente la base de la plupart des filtres à particules développés jusqu'à présent. Les différentes versions présentes dans la littérature peuvent être vues comme des cas particuliers de cet algorithme obtenus grâce à un choix approprié de densité d'échantillonnage par importance et/ou une modification de l'étape de ré-échantillonnage. On distinguera notamment les filtres SIR et MCMC.
Le filtre d'échantillonnage avec ré-échantillonnage par importance (SIR pour Sampling Importance Resampling) [Gordon 1993] nécessite que les fonctions du système (définie dans 1.6) et de mesure (définie dans 1.7) soient connues. Il est aussi nécessaire de pouvoir échantillonner à partir de la distribution du bruit de traitement et de la densité précédente. Enfin la fonction de vraisemblance $p(\mathbf{Z}_k|\mathbf{X}_k)$ doit être disponible en tout point de l'ensemble de définition. L'algorithme SIR constitue une variante de l'algorithme SIS par le choix de la densité d'importance $q(\mathbf{X}_k|\mathbf{X}_{k-1}^i, \mathbf{Z}_{1:k}) = p(\mathbf{X}_k|\mathbf{X}_{k-1}^i)$ et par l'application de l'algorithme de ré-échantillonnage à chaque pas de temps. Les échantillons tirés de $p(\mathbf{X}_k|\mathbf{X}_{k-1}^i)$ sont obtenus par génération d'échantillons $\mathbf{V}_{k-1}^i$ à partir de $p_V(\mathbf{V}_{k-1})$, avec $p_V(.)$ la fonction de densité de probabilité de $\mathbf{V}_{k-1}$, et par application de $\mathbf{f}_k$. On obtient donc les échantillons $\mathbf{X}_k^i$ (ainsi que leur poids) via :

$$\left\{ \begin{array}{rcl} \mathbf{X}_k^i & = & \mathbf{f}_k(\mathbf{X}_{k-1}^i, \mathbf{V}_{k-1}^i) \\ w_k^i & \propto & w_{k-1}^i p(\mathbf{Z}_k|\mathbf{X}_k^i) \propto p(\mathbf{Z}_k|\mathbf{X}_k^i) \end{array} \right. \tag{1.26}$$

la deuxième relation de proportionnalité étant obtenue puisque l'on effectue le ré-échantillonnage à chaque instant k ($w_{k-1}^i = \frac{1}{N} \forall i$). La figure 1.2 illustre le principe du filtre SIR.
L'idée du filtre d'échantillonnage MCMC (pour "Markov Chain Monte Carlo") est de générer une chaine de Markov ayant pour distribution stationnaire une distribution donnée $f(.)$. Le principe de ce type de méthode est le suivant : on définit tout d'abord une distribution de génération de candidats et un échantillon initial. Puis, à chaque étape, on génère un nouvel échantillon suivant la distribution choisie. On décide alors d'accepter ou de rejeter ce candidat à l'aide d'une procédure d'acceptation-rejet. Enfin on supprime les premières valeurs générées (on considère que le processus approche de sa distribution stationnaire après un certain nombre d'itérations). Le premier formalisme de ce type d'approche fut introduit par l'algorithme de Metropolis-Hastings [Metropolis 1953] [Hastings 1970], algorithme qui reste aujourd'hui une référence dans la communauté de vision par ordinateur. Celui-ci se présente de la manière suivante : supposons disponibles un échantillon initial θ^0 et une distribution de génération de candidats $q(Y|\theta^k)$. A chaque itération t, on génère un nouveau candidat Y_t à partir de l'échantillon précédent θ^{t-1} via $q(Y_k|\theta^{k-1})$.

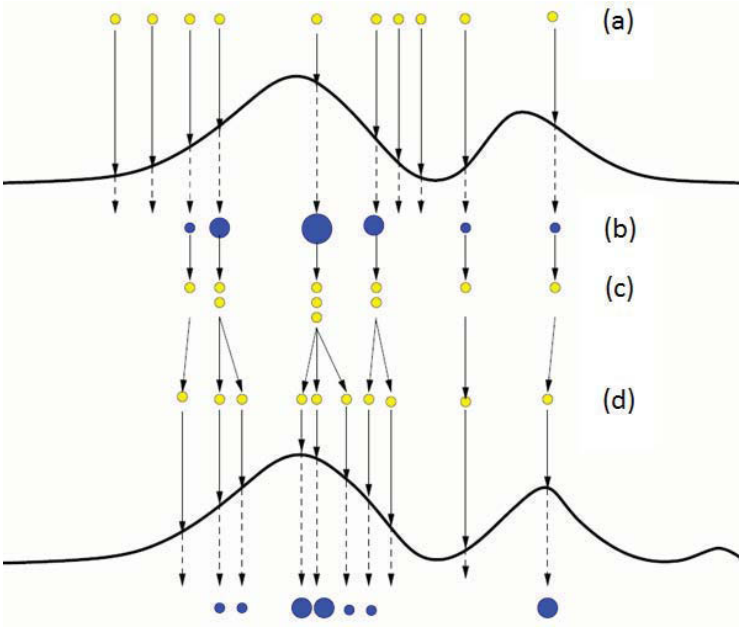


Figure 1.2 – Principe du filtre SIR. (a) : le point de départ est un ensemble de particules de même poids ; (b) : étape de pondération par importance, chaque particule se voit pondérée relativement aux observations (probabilité de l'état estimé) ; (c) : étape de ré-échantillonnage, les échantillons de poids faible sont éliminés et de nouveaux échantillons sont créés à partir de ceux de poids fort ; (d) : étape classique de prédiction et retour au cas (a).

On applique alors le schéma d'acceptation-rejet suivant :

$$\begin{cases} \theta^{t+1} = \begin{cases} Y_t & \text{avec une probabilité } \rho(\theta^t, Y_t) \\ \theta^t & \text{avec une probabilité } 1 - \rho(\theta^t, Y_t) \end{cases} \\ \rho(\theta^t, Y_t) = min(1, \frac{f(Y_t)}{f(\theta^t)} \frac{q(\theta^t|Y_t)}{q(Y_t|\theta^t)}) \end{cases} \tag{1.27}$$

Utilisé dans le cadre du filtrage particulaire, ce type de filtre permet d'éviter le problème d'appauvrissement des échantillons, problème dont souffrent la plupart des filtres particulaires. Les filtres particulaires basés MCMC sont très utilisés dans la communauté de vision par ordinateur. On peut citer, par exemple, les travaux de Khan et al. [Khan 2005] dans lesquels les auteurs utilisent un filtre basé MCMC pour suivre de multiples cibles en interaction ou encore les travaux de Jing et Vadakkepat [Jing 2010] dans lesquels les auteurs améliorent la méthode de sélection/création des échantillons du filtre MCMC afin de réduire encore l'effet d'appauvrissement et d'augmenter la vitesse de convergence du filtre.

1.2.5 Méthode choisie

Les méthodes de suivi par noyau permettent de gérer différents types de mouvement suivant l'algorithme utilisé ainsi que les occlusions partielles de la cible. Cependant, l'utilisation de formes géométriques primitives implique en général qu'une partie de l'objet réside à l'extérieur de la forme alors qu'une partie du fond se trouve à l'intérieur. Les recherches de similarité entre modèles peuvent, par conséquent, s'en retrouver faussées.
Le suivi de silhouette permet une grande flexibilité grâce à la gestion de formes diverses et variées (et même complexes). Cependant ce type de méthode ne permet pas, en général, de gérer les occlusions, ou alors sous fortes contraintes de forme, ce qui n'est pas applicable au suivi d'objet quelconque. Enfin les méthodes de suivi de silhouette possèdent de grosses difficultés dans la gestion des fusions/séparations d'objets (typiquement une personne possédant un sac et l'abandonnant).
Le suivi de points déterministe a l'avantage d'être relativement simple et par conséquent de posséder un coût de calcul très faible. Toutefois notre variété d'objets suivis implique une représentation difficilement unifiable et par conséquent une difficulté de mise en place de lois de correspondance déterministes.
Notre choix s'est donc naturellement porté vers les méthodes de suivi de points probabilistes et plus particulièrement l'utilisation de filtres particulaires (les méthodes optimales et sous optimales n'étant pas applicables au vu des contraintes et hypothèses qu'elles mettent en jeu) de type MCMC (afin d'éviter les problèmes de dégénérescence et d'appauvrissement des échantillons rencontrés par les filtres particulaires "classiques"). Ces méthodes présentent, de plus, un coût de calcul relativement faible, la possibilité d'enrichir leur vecteur d'état à souhait (selon le besoin) et une gestion des occlusions partielles et/ou totales.

Chapitre 2

Suivi par apprentissage adaptatif : Ensemble Tracking

Sommaire

L'Ensemble Tracking, développé en 2007 par Shaï Avidan [Avidan 2007], considère le suivi comme un problème de classification binaire. Un ensemble de classifieurs dits faibles ou naïfs (dont l'efficacité est au minimum légèrement supérieure à celle du hasard) est construit en ligne afin de distinguer l'objet suivi du fond de la scène. Cet ensemble est alors combiné via l'algorithme Adaboost en un classifieur dit fort, classifieur qui sera alors utilisé dans l'image suivante pour déterminer la nouvelle position de l'objet. Enfin, le modèle est maintenu à jour grâce à la mise à jour de l'ensemble des classifieurs : de nouveaux classifieurs entrainés en ligne sont ajoutés à l'ensemble alors que d'autres sont retirés.
Comme précisé très brièvement dans le chapitre 1, l'algorithme Ensemble Tracking représente l'algorithme de base de notre méthode, présentée au chapitre 3. Une description plus détaillée de cet algorithme est donc nécessaire. Dans une première partie, nous décrivons en détail les différentes phases de l'algorithme Ensemble Tracking. Nous présentons ensuite les différentes heuristiques d'implémentation utilisées par l'auteur pour augmenter l'efficacité de sa méthode. Enfin nous introduisons quelques résultats présentés par l'auteur et décrivons les différentes limites de l'algorithme de Shaï Avidan.

2.1 Analyse algorithmique

L'algorithme entraine un classifieur afin de distinguer l'objet du fond. Ceci passe par la construction d'un vecteur de caractéristiques pour chaque pixel d'une image de référence et par la construction d'un classifieur capable de séparer les pixels appartenant à l'objet des pixels appartenant au fond. Etant donnée une nouvelle image de la séquence vidéo, le classifieur est utilisé pour tester les pixels et former une carte de confiance. Le pic de la carte, estimé grâce à l'algorithme Mean Shift [Cheng 1995], correspond alors à la nouvelle position de l'objet. Enfin, puisque le tracker doit s'adapter aux changements d'apparence de l'objet et intégrer de nouvelles informations sur le fond (objet et fond n'étant pas "fixes" au fil du temps), l'intégration temporelle est maintenue grâce à l'entrainement de nouveaux classifieurs faibles et à leur ajout (parallèlement au retrait de classifieurs existants) à l'ensemble des classifieurs faibles.
Chaque classifieur faible est entrainé sur des exemples positifs et négatifs où, par convention, on appelle exemples positifs les exemples provenant de l'objet et exemples négatifs ceux provenant du fond. Le classifieur fort, construit sur l'ensemble des classifieurs faibles grâce à l'algorithme Adaboost, est alors utilisé pour classifier les pixels dans l'image suivante et construire une carte de confiance où la mesure de confiance correspond à la marge de classification du classifieur fort. Une fois la carte analysée par Mean Shift et la nouvelle position de l'objet déterminée, la mise à jour est enclenchée et tout le processus est répété à nouveau. Une vue générale et un schéma synoptique de l'algorithme sont présentés respectivement en Algorithme 2.1 et en Figure 2.1.

Données :	$I_1, ..., I_n$	$I_i : p \longrightarrow I(p)$	n images extraites
		$\Omega \subset \mathbb{Z}^2 \longmapsto \mathbb{R}$	d'une séquence vidéo
	$r_1 \subset \Omega$	Rectangle de l'objet dans la première image	
Résultats :	$r_2, ..., r_n$	Rectangles de l'objet dans les images suivantes	

Initialisation (pour l'image I_1) :
- Entraînement de T classifieurs faibles

Pour chaque nouvelle image I_j :
- Test de tous les pixels de I_j par le classifieur fort courant et création d'une carte de confiance L_j
- Analyse de la carte L_j par Mean Shift et récupération du nouveau rectangle r_j de l'objet
- Étiquetage des pixels de r_j comme pixels objet et de ceux à l'extérieur comme pixels de fond
- Conservation des K "meilleurs" classifieurs faibles
- Entraînement de T - K nouveaux classifieurs faibles sur l'image I_j

Algorithme 2.1: Vue générale de l'algorithme Ensemble Tracking

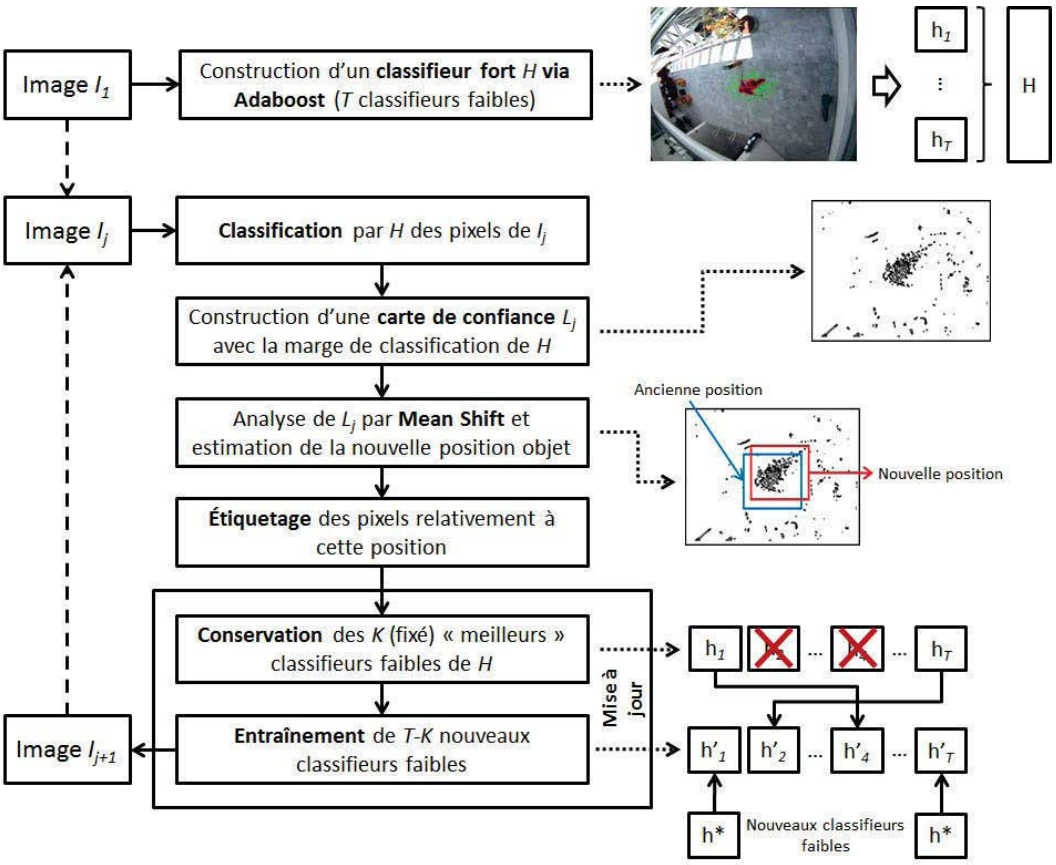


Figure 2.1 – Schéma synoptique de l'algorithme Ensemble Tracking.

L'algorithme étant basé principalement sur la construction de classifieurs, intéressons nous tout d'abord aux notations et à la définition des classifieurs fort et faible.

2.1.1 Classifieurs faibles et classifieur fort

On suppose que chaque pixel est représenté par un vecteur de caractéristiques de dimension d constitué d'informations locales. Ces informations peuvent correspondre, pour ne citer que les informations utilisées par l'auteur, à des informations de couleurs (notamment les trois composantes rouge, verte et bleue du pixel ou encore son niveau de gris), à des informations de gradients (notamment un histogramme local d'orientation des gradients seuillés à huit composantes calculé sur une fenêtre 5x5 autour du pixel (cf. chapitre 3.2.2)) ou encore à des combinaisons non-linéaires de ces caractéristiques.
Soient $\{(\mathbf{x}_i, y_i); i \in [\![1, N]\!]\}$ N exemples, de vecteur de caractéristiques $\mathbf{x}_i \in \mathbb{R}^d$ et d'étiquette $y_i \in \{-1, +1\}$. On appelle classifieur faible, noté $h(\mathbf{x}) : \mathbb{R}^d \longmapsto \{-1, +1\}$, par opposition au classifieur fort, une fonction permettant de séparer les pixels objet des pixels de fond de manière un tant soit peu meilleure qu'un simple tirage au sort. Chaque classifieur faible est entrainé sur la base d'un ensemble d'exemples d'apprentissage pondérés, le poids correspondant à l'impact de chaque exemple dans le processus d'apprentissage (le calcul de ces poids est explicité en 2.2). La définition d'un classifieur faible retenue par l'auteur est la suivante :

$$h(\mathbf{x}) = sign(\mathbf{h}^T\mathbf{x}) \tag{2.1}$$

où $\mathbf{h} \in \mathbb{R}^d$ représente un hyperplan séparateur calculé par régression des moindres carrés pondérés :

$$\mathbf{h} = (\mathbf{A}^T\mathbf{W}\mathbf{A})^{-1}\mathbf{A}^T\mathbf{W}\mathbf{y} \tag{2.2}$$

où chaque ligne de $\mathbf{A}$ notée $\mathbf{A}_i$ correspond à un exemple $\mathbf{x}_i$ augmenté de la constante 1 ($\mathbf{A}_i = [\mathbf{x}_i, 1]$), $\mathbf{W}$ correspond à la matrice diagonale des poids des exemples et $\mathbf{y}$ au vecteur des étiquettes.
Afin d'empêcher l'introduction d'un biais en cas de zone objet plus petite que la zone de fond, l'auteur préconise de normaliser la somme des poids des exemples positifs (resp. négatifs) de sorte qu'elle soit égale à 0,5.
On appelle classifieur fort la combinaison pondérée d'un ensemble de classifieurs faibles. L'algorithme Adaboost, durant le processus de construction de ce classifieur fort, attribue à chaque classifieur faible h_t un poids α_t basé sur son erreur de classification sur l'ensemble des exemples d'apprentissage. Le classifieur fort, noté $H : \mathbb{R}^d \longmapsto \{-1, +1\}$, est alors défini par :

$$\begin{cases} H(\mathbf{x}) = sign(M(\mathbf{x})) \\ M(\mathbf{x}) = \sum_{t=1}^{T} \alpha_t h_t(\mathbf{x}) \end{cases} \tag{2.3}$$

avec T le nombre de classifieurs faibles le composant.
Le classifieur fort construit grâce à l'ensemble des classifieurs faibles à chaque image est utilisé pour construire une carte de confiance correspondant à l'image suivante.

La marge de classification $M(\mathbf{x})$ du classifieur fort est convertie en mesure de confiance $c(\mathbf{x})$ en assignant aux marges négatives une valeur nulle et en rapportant les marges positives à l'intervalle $[0, 1]$:

$$c(\mathbf{x}) = \frac{\max(0, M(\mathbf{x}))}{\sup\{M(\mathbf{p}), \forall \mathbf{p}\}} \tag{2.4}$$

2.1.2 Mise à jour

Une fois la carte de confiance analysée par Mean Shift et la nouvelle position de l'objet déterminée, l'étape de mise à jour entre en jeu. L'algorithme ne conserve alors que les K "meilleurs" classifieurs faibles. Ces classifieurs sont passés à l'algorithme Adaboost afin de mettre à jour leur poids et, par la même occasion, construire une distribution d'exemples utilisée pour l'entrainement de nouveaux classifieurs à ajouter à l'ensemble.
Une attention toute particulière doit être portée, lors de l'ajout de nouveaux classifieurs ou de la repondération de classifieurs existants, à l'efficacité des classifieurs faibles. L'auteur considère que si, lors de l'étape d'ajout d'un nouveau classifieur faible à l'ensemble, celui-ci n'est pas meilleur qu'un simple tirage au sort, la boucle d'ajout doit être arrêtée et l'ensemble demeurer tel quel. De la même manière, si un classifieur faible repondéré s'avère moins efficace qu'un tirage au sort alors il se voit affecter d'un poids nul. L'auteur n'autorise cependant que deux invalidations de classifieurs faibles de cette manière, un nombre plus important de classifieurs moins efficaces qu'un tirage au sort pouvant être le signe d'une occlusion (l'ensemble ne devant par conséquent pas être modifié).

2.1.3 Algorithme global

La totalité de l'algorithme Ensemble Tracking est présentée en Algorithme 2.2.
La première phase de l'algorithme correspond à l'étape d'initialisation. De la première image de la séquence vidéo sont extraits plusieurs échantillons d'apprentissage étiquetés (1). Chaque échantillon est alors pondéré avec la même valeur (2) et le processus de construction des classifieurs faibles débute (3).
À chaque nouvelle construction d'un classifieur faible, l'ensemble des poids des échantillons est normalisé (3a) afin de pouvoir être considéré comme une densité de probabilité. Cette densité est alors utilisée lors de l'apprentissage du nouveau classifieur (3b). La qualité de l'apprentissage réalisé est ensuite mesurée par le calcul de l'erreur de classification (3c) sur l'ensemble des échantillons d'apprentissage (proportionnelle à la somme des poids des échantillons mal classés) et un poids déterminé relativement à cette erreur est attribué au classifieur faible (3d). Enfin le poids des échantillons d'apprentissage est mis à jour (3e) de sorte que le classifieur faible construit à l'itération suivante se concentre d'avantage sur les échantillons mal classés à l'itération courante (le poids diminue pour les échantillons bien classés et augmente pour les autres).
À chaque nouvelle image de la séquence vidéo, plusieurs échantillons, ici de test,

Données :	$I_1, ..., I_n$	$I_i : p \longrightarrow I(p)$	n images extraites
		$\Omega \subset \mathbb{Z}^2 \longmapsto \mathbb{R}$	d'une séquence vidéo
	$r_1 \subset \Omega$	Rectangle de l'objet dans la première image	
Résultats :	$r_2, ..., r_n$	Rectangles de l'objet dans les images suivantes	

Initialisation (pour l'image I_1) :

1. Extraction de N exemples étiquetés : $\{(\mathbf{x}_i, y_i)\}_{i=1}^N$
2. Initialisation des poids $\{w_i\}_{i=1}^N$ à $\frac{1}{N}$
3. Pour $t = 1...T$,
   (a) Transformation de $\{w_i\}_{i=1}^N$ en distribution
   (b) Entrainement du classifieur faible h_t
   (c) Calcul de son erreur $err = \sum_{i=1}^N w_i \, |h_t(\mathbf{x}_i) - y_i|$
   (d) Calcul de son poids $\alpha_t = \frac{1}{2} \log \frac{1-err}{err}$
   (e) Mise à jour du poids des exemples $w_i = w_i e^{\alpha_t |h_t(\mathbf{x}_i) - y_i|}$
4. Le classifieur fort est donné par le signe de : $\sum_{t=1}^T \alpha_t h_t(\mathbf{x})$

Pour chaque nouvelle image I_j :

1. Extraction de N exemples : $\{\mathbf{x}_i\}_{i=1}^N$
2. Test des exemples en utilisant le classifieur fort construit à l'image précédente et création d'une carte de confiance L_j
3. Analyse de la carte L_j par Mean-Shift avec r_{j-1} comme point de départ. Soit r_j la nouvelle position déterminée
4. Définition des étiquettes $\{y_i\}_{i=1}^N$ relativement au rectangle r_j
5. Initialisation des poids $\{w_i\}_{i=1}^N$ à $\frac{1}{N}$
6. Pour $t = 1...K$, (Choix et mise à jour des poids des K meilleurs classifieurs)
   (a) Transformation de $\{w_i\}_{i=1}^N$ en distribution
   (b) Choix du h_t parmi $\{h_1, ..., h_T\}$ dont l'erreur err est minimale
   (c) Mise à jour de α_t et $\{w_i\}_{i=1}^N$
   (d) Suppression de h_t de l'ensemble $\{h_1, ..., h_T\}$
7. Pour $t = K + 1...T$, (Ajout de nouveaux classifieurs faibles)
   (a) Transformation de $\{w_i\}_{i=1}^N$ en distribution
   (b) Entrainement du classifieur faible h_t
   (c) Calcul de err et α_t
   (d) Mise à jour du poids des exemples $\{w_i\}_{i=1}^N$
8. Le classifieur fort mis à jour est donné par le signe de : $\sum_{t=1}^T \alpha_t h_t(\mathbf{x})$

Algorithme 2.2: Ensemble Tracking

sont extraits également (1) et soumis pour classification au classifieur fort construit à l'image précédente. La marge de classification du classifieur fort est alors utilisée (cf. equation (2.4)) pour construire une carte de confiance (2). L'algorithme Mean-Shift, initialisé sur la dernière position connue de la cible, extrait ensuite de cette carte la nouvelle position de l'objet suivi (3). Cette dernière est alors utilisée afin de mettre à jour l'ensemble des classifieurs faibles.

En se basant sur la nouvelle position de l'objet, l'ensemble des échantillons précédemment extraits est étiqueté (4) puis equi-pondéré (5). Cet ensemble d'apprentissage est tout d'abord utilisé pour sélectionner et mettre à jour les K "meilleurs" classifieurs faibles (6) puis pour entraîner de nouveaux classifieurs afin de compléter l'ensemble (7) et construire ainsi un nouveau classifieur fort.
Durant l'étape de sélection des meilleurs classifieurs faibles, l'erreur de chaque classifieur est calculée sur l'ensemble des échantillons étiquetés et pondérés (pondération normalisée (6a) pour correspondre à nouveau à une distribution de probabilité) et le classifieur dont l'erreur est la plus faible est sélectionné (6b). Celui-ci voit alors sa pondération révisée pour correspondre à sa nouvelle erreur (6c). Enfin, à la manière de l'étape d'initialisation, la pondération des échantillons est corrigée pour permettre au classifieur choisi suivant de se concentrer d'avantage sur les échantillons "difficiles" (6c). Cette étape de sélection/mise à jour est répétée K fois. Ainsi on obtient bien les K meilleurs classifieurs faibles sur le jeu de données courant ainsi qu'une distribution de probabilité initiale permettant l'apprentissage de $T - K$ nouveaux classifieurs (7) à la manière de l'étape d'initialisation.
L'auteur précise que l'algorithme est en général utilisé avec un ensemble de trois à cinq classifieurs faibles.

2.2 Heuristiques d'implémentation

Afin de pallier les différents problèmes rencontrés durant les premières étapes de l'implémentation et afin d'améliorer encore l'efficacité de l'algorithme, l'auteur a mis en place deux heuristiques d'implémentation majeures : un mécanisme de suppression des valeurs marginales et un mécanisme de suivi multi-résolutions.

2.2.1 Suppression des valeurs marginales

Si l'objet à suivre n'est pas un rectangle pur, alors le rectangle englobant utilisé pour le suivi va nécessairement inclure des pixels étiquetés comme positifs alors qu'ils auraient dû être étiquetés comme négatifs. Sachant de plus que l'algorithme Adaboost est très sensible aux valeurs aberrantes [Freund 2001] , il est apparu nécessaire de mettre en place un mécanisme de rejet de ces valeurs.
Le principe choisi par l'auteur est de traiter les échantillons trop difficiles comme des valeurs marginales et de changer leur étiquette. Ainsi l'étape 4 de la boucle de traitement de l'algorithme (boucle répétée pour chaque image de la séquence) est modifiée de :

$$y_i = \begin{cases} +1 \text{ si } interieur(r_j, p_i) = vrai \\ \text{-1 sinon} \end{cases} \tag{2.5}$$

à :

$$y_i = \begin{cases} +1 \text{ si } (interieur(r_j, p_i) = vrai \text{ et } (w_i < \Theta)) \\ \text{-1 sinon} \end{cases} \tag{2.6}$$

où r_j représente le rectangle englobant courant, p_i la position de l'échantillon i, $interieur(r, p)$ un prédicat valant "vrai" si le pixel p se trouve à l'intérieur de r, w_i

le score de classification retourné par le classifieur fort vis à vis de l'échantillon i et Θ un seuil prédéfini, fixé par l'auteur à $\frac{3}{N}$ avec N le nombre total d'échantillons. L'auteur suppose ainsi que chaque pixel à l'intérieur du rectangle englobant est positif, à moins qu'il ne soit trop difficile à classifier, auquel cas il est changé en pixel négatif.
Afin d'illustrer cet artéfact, l'auteur présente la figure 2.2 dans laquelle la carte de confiance construite en utilisant ce mécanisme s'avère plus précise, rendant ainsi le suivi plus stable et plus performant.

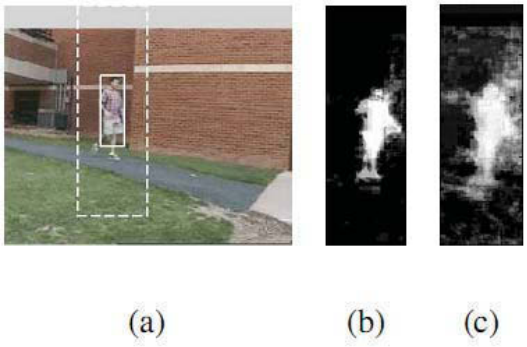


Figure 2.2 – Mécanisme de rejet des valeurs marginales. (a) Image d'entrée. Le rectangle solide correspond au rectangle englobant considéré et le rectangle en pointillés au rectangle de fond. (b) Carte de confiance avec rejet des valeurs marginales. (c) Carte de confiance sans rejet des valeurs marginales. Source [Avidan 2007].

2.2.2 Suivi multi-résolution

Afin de permettre au tracker de capturer différentes caractéristiques à différentes échelles, l'auteur propose d'exécuter l'algorithme dans un environnement multi-échelles construit sous forme de pyramide.
Un classifieur fort est maintenu, tout au long de la séquence vidéo, pour chaque niveau de la pyramide (chaque échelle d'image choisie). Chaque classifieur fort génère alors, pour toute nouvelle image, une carte de confiance. Toutes les cartes résultantes sont redimensionnées à la taille de l'image initiale et moyennées pour former la carte de confiance qui sera analysée par Mean-Shift.
La figure 2.3 présente une carte de confiance typique, obtenue par accumulation sur plusieurs échelles. Les cartes de confiance sont obtenues pour l'échelle originale, l'échelle réduite de moitié et l'échelle réduite au quart. Les cartes sont ensuite redimensionnées aux dimensions de l'image originale puis combinées en se basant sur le score global de classification de chaque classifieur fort (moyenne pondérée).

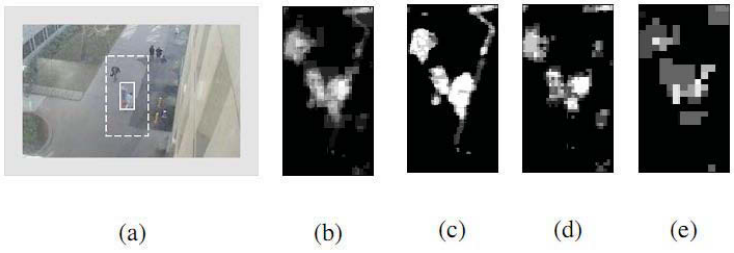


Figure 2.3 – Mécanisme de construction de carte multi-échelle. (a) Image d'entrée. Le rectangle solide correspond au rectangle englobant considéré et le rectangle en pointillés au rectangle de fond. (b) Carte de confiance calculée comme moyenne pondérée des cartes (c) à (e). (c) Carte de confiance de l'image originale. (d) Carte de confiance à l'échelle réduite de moitié. (e) Carte de confiance à l'échelle réduite au quart. Source [Avidan 2007].

2.3 Résultats et limites

La dernière partie de l'article de Shaï Avidan est consacrée aux expérimentations menées par l'auteur relativement à son algorithme et aux résultats tirés de ces expérimentations. Les différents résultats présentés mettent en avant plusieurs avantages de la méthode, en particulier l'adaptation au changement, la gestion des caméras mobiles ou encore la gestion des occlusions partielles. Cependant l'algorithme souffre aussi de quelques limites (dont certaines sont mentionnées par l'auteur).

2.3.1 Résultats présentés par l'auteur

Comme annoncé précédemment, l'auteur base la majorité de ces expérimentations sur une version de l'Ensemble Tracking utilisant cinq classifieurs faibles travaillant sur un vecteur de caractéristiques à onze dimensions (histogramme local d'orientation du gradient à huit composantes et composantes rouge, verte et bleue du pixel). Cependant, lorsque la séquence traitée ne permet pas d'utiliser ces caractéristiques (images en niveaux de gris ou images infra-rouges), l'auteur utilise une version non linéaire de l'espace des caractéristiques en élevant le vecteur de caractéristiques original au carré et au cube et réduit le nombre de classifieurs faibles à trois.
Dans chacune des expérimentations présentées par l'auteur le rectangle initial encadrant l'objet est fourni manuellement. Le traitement global est effectué en parallèle sur trois niveaux de pyramide comme présenté en section 2.2.2. Enfin l'algorithme supprime un classifieur faible à chaque image de la séquence et le remplace par un classifieur faible fraîchement entrainé (tout en autorisant la suppression de deux classifieurs faibles par image maximum, un nombre supérieur pouvant être synonyme d'occlusion).
L'auteur présente, dans un premier temps, différentes expérimentations qualitatives simples illustrant le bon fonctionnement de la méthode décrite : la figure 2.4 présente l'adaptativité des classifieurs faibles, la figure 2.5 le maintien du suivi en cas de changement de forme de la cible et enfin la figure 2.6, parallèlement au

maintien du suivi en cas de changement de forme de la cible, le maintien du suivi en cas de très légère occlusion.

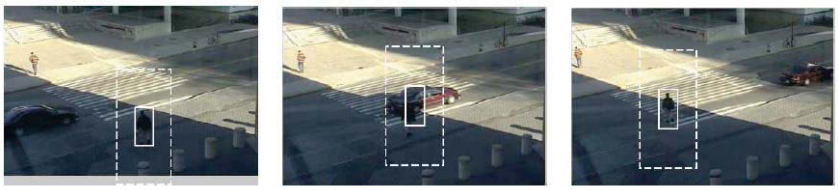


Figure 2.4 – Illustration de l'adaptation des classifieurs faibles. Les images présentées correspondent aux images 10, 40 et 70 d'une séquence de 100 images. Ce suivi illustre le fait qu'en présence de pixels de fond dont les caractéristiques de couleur sont très proches de celles des pixels objets, les classifieurs faibles s'adaptent en se concentrant sur les informations de gradient et parviennent ainsi à maintenir le suivi. Source [Avidan 2007].

Figure 2.5 – Illustration de l'adaptation au changement de forme de la cible. Les images présentées correspondent aux images 0, 20, 40 et 70 d'une séquence de 90 images. Malgré le changement de forme de la cible au fil des images, le suivi est maintenu tout au long de la séquence. Source [Avidan 2007].

Figure 2.6 – Illustration de l'adaptation au changement de forme de la cible et aux occlusions très partielles. La séquence étudiée compte 200 images. Malgré le changement de forme de la cible au fil des images ainsi que son occlusion partielle, notamment en image 37, le suivi est maintenu tout au long de la séquence. Source [Avidan 2007].

L'auteur illustre ensuite la nécessité de la mise à jour des classifieurs au travers de la figure 2.7. Contrairement au premier cas dans lequel un tracker "dynamique", c'est

à dire mis à jour selon la méthode décrite précédemment, est utilisé, le second cas utilise un tracker "statique" (non mis à jour). Ceci a pour effet une perte du suivi au niveau de l'image 30 pour le second tracker alors que le premier tracker maintient le suivi jusqu'à la dernière image de la séquence.

Figure 2.7 – Ensemble Tracking avec et sans mise à jour. (a-d) Suivi avec mise à jour des classifieurs faibles. Les classifieurs sont mis à jour suivant la méthode décrite précédemment. (e-h) Suivi sans mise à jour des classifieurs faibles. L'algorithme construit cinq classifieurs faibles sur la première image de la séquence et conserve ces classifieurs tout au long de la séquence. Source [Avidan 2007].

La nécessité de la mise à jour étant avérée, l'auteur propose alors une comparaison entre la classification par Ensemble Tracking et la classification basée sur un classifieur construit via l'algorithme Adaboost à chaque image d'une même séquence (un classifieur indépendant basé sur le même nombre de classifieurs faibles est construit de zéro à chaque image en se basant sur les données d'apprentissage utilisée par l'Ensemble Tracking pour construire le nouveau classifieur faible). Les résultats sont donnés en figure 2.8.
Comme on peut le remarquer, le classifieur construit via Adaboost surpasse le classifieur construit via Ensemble Tracking d'au maximum 8%. Cependant l'écart entre les deux algorithmes se réduit au fil de la séquence et tend à devenir nul, voire à s'inverser. De plus il est important de rappeler que l'Ensemble Tracking ne construit qu'un classifieur faible par image alors qu'Adaboost en construit cinq dans cet exemple, ce qui représente un gain en temps de traitement considérable.

L'auteur présente ensuite plusieurs résultats illustrant certains avantages notables de la méthode développée. Tout d'abord, au travers de la figure 2.9 présentant le suivi sur caméscope, l'auteur démontre que l'algorithme Ensemble Tracking gère

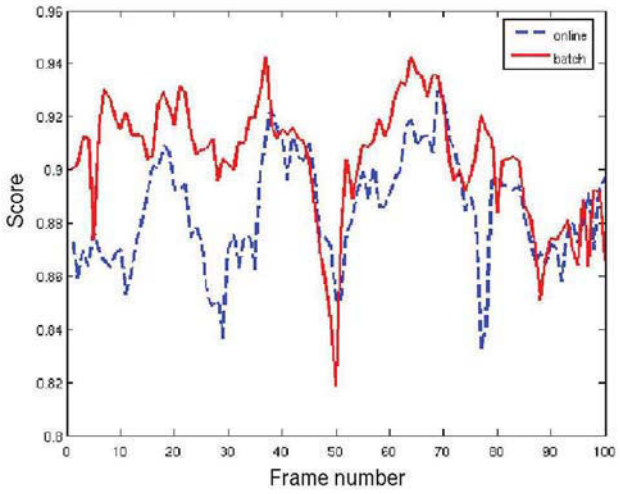


Figure 2.8 – Ensemble Tracking (en ligne) contre Adaboost (groupe). La ligne pleine correspond à l'algorithme Adaboost et la ligne en pointillés à l'algorithme Ensemble Tracking. L'abscisse représente l'index de l'image et l'ordonnée le score de classification obtenu par chaque algorithme sur chaque image. Source [Avidan 2007].

parfaitement les caméras mobiles. L'auteur insiste également sur la stabilité obtenue par l'algorithme Ensemble Tracking sur des images autres que des images couleurs (moyennant le changement d'espace de caractéristiques présenté précédemment). Ainsi la figure 2.10 (resp. 2.11) représente le suivi d'une voiture (resp. d'un zodiac) dans une séquence vidéo en niveau de gris (resp. en infra-rouge).

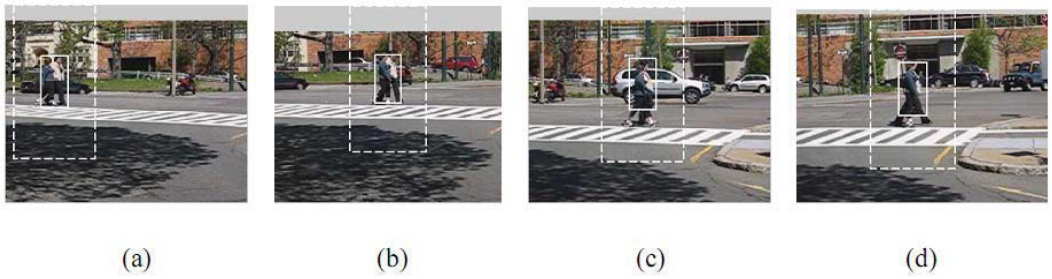


Figure 2.9 – Ensemble Tracking sur caméra mobile. Les images présentées correspondent aux images 0, 40, 68 et 80 d'une séquence de 80 images. Source [Avidan 2007].

Enfin, et pour pallier le problème rencontré par l'algorithme en cas d'occlusion longue et/ou totale, l'auteur propose l'utilisation d'une approche simple basée sur un filtre particulaire. Tant que le taux de classification (défini comme le rapport du nombre de pixels bien classés sur le nombre de pixels total) est élevé, le suivi reste tel quel. Par contre, lorsqu'il descend sous un seuil prédéfini (fixé autour de 0.5 par l'auteur), le filtrage particulaire prend le relais (mode prédictif). Le filtre échantillonne alors plusieurs positions où l'objet peut probablement réapparaitre (5

Figure 2.10 – Ensemble Tracking sur séquence en niveaux de gris. Les images présentées correspondent aux images 0, 100 et 225 d'une séquence de 225 images. Source [Avidan 2007].

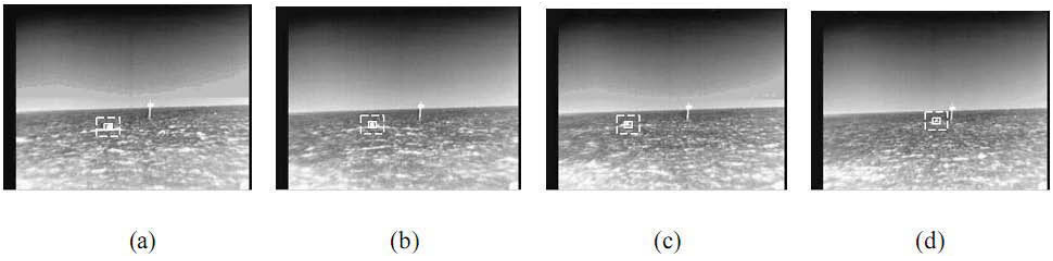


Figure 2.11 – Ensemble Tracking sur séquence en infra-rouge. Les images présentées correspondent aux images 1, 200, 400 et 671 d'une séquence de 671 images. Source [Avidan 2007].

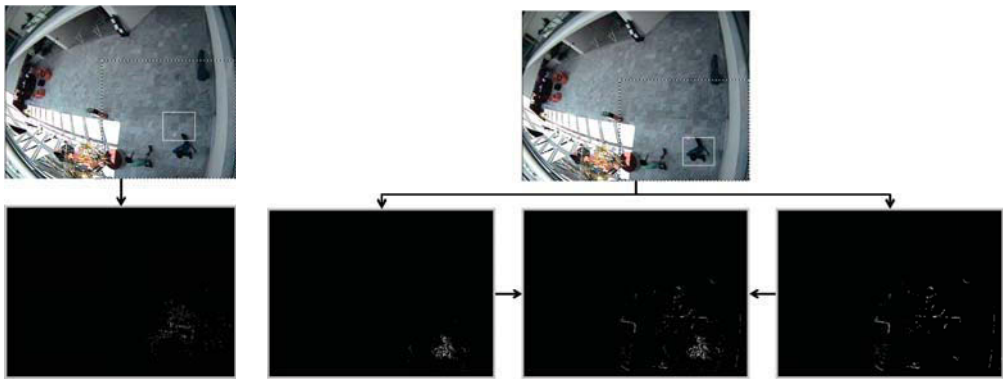
positions en général). Une carte de confiance est construite, en utilisant l'ensemble existant, pour chacune de ces positions et l'algorithme Mean-Shift est utilisé pour en déduire les nouvelles positions probables. Le score de classification de ces positions est alors calculé et si l'un d'entre eux dépasse un certain seuil prédéfini (fixé à 0.7 par l'auteur) le suivi reprend. La figure 2.12 illustre la mise en application de cette solution. Comme le précise l'auteur, l'occlusion dure environ 100 images. Durant ce laps de temps, et notamment à l'image 155, le tracker cible l'homme et non la femme. Cependant, la carte de confiance ainsi que le score de classification de la position (0.6) indiquent que le tracker sait qu'il doit continuer à chercher et ne pas réinitialiser le suivi immédiatement. Le suivi est, par contre, correctement réinitialisé à l'image 167, lorsque la femme réapparait.

Figure 2.12 – Gestion des occlusions. La séquence étudiée compte 348 images. Dans le coin supérieur droit de chaque image est présentée la carte de confiance associée au rectangle en pointillés. Source [Avidan 2007].

2.3.2 Limites de l'algorithme

La première limite de l'algorithme, présentée par l'auteur, est relative aux ressources qu'il utilise. En effet l'auteur précise que l'algorithme utilise la totalité des pixels de la zone considérée (fond et objet) afin d'effectuer la mise à jour de l'ensemble, ce qui a pour effet immédiat de réduire les performances de l'algorithme à quelques images traitées par seconde.
L'auteur mentionne également trois autres limitations. La première concerne les occlusions totales et/ou longues. En effet malgré les bons résultats obtenus avec l'artifice proposé (filtrage particulaire), il reste de nombreuses situations pour lesquelles

l'algorithme est inadapté et l'auteur insiste sur le fait qu'il semble préférable de fusionner les deux approches (ensemble tracking et filtrage particulaire) afin d'améliorer encore la méthode. Le second problème concerne la limite entre adaptation et dérive et plus particulièrement la manière de configurer au mieux l'algorithme afin qu'il s'adapte de manière optimale aux changements intervenant dans la scène sans pour autant souffrir du problème de drifting. Ce dernier correspond à un problème "classique" rencontré par les algorithmes d'apprentissage. Le modèle à apprendre évolue au fil du temps et les algorithmes d'apprentissage tentent alors de refléter cette évolution en intégrant des données la traduisant. Tout l'enjeu est alors de distinguer les données bruitées des données reflétant l'évolution du modèle. Le drifting correspond à la dérive du modèle appris par intégration d'une quantité de données bruitées trop conséquente (des données de fond dans notre cas). Enfin, la troisième limitation présentée par l'auteur est relative à l'espace des caractéristiques utilisées. Cet espace est, en l'état, relativement limité et l'algorithme pourrait gagner en efficacité s'il était enrichi, notamment par une information spatiale.
On peut également ajouter à ces limites la nécessité de configuration "manuelle" de l'espace des caractéristiques en fonction de la source vidéo utilisée. En effet l'auteur précise qu'au vu de l'instabilité constatée avec l'espace de caractéristiques original sur les vidéos en infra-rouge et en niveaux de gris, il a été nécessaire de changer manuellement cet espace pour un espace plus adapté. Ce besoin de changement manuel vient à l'encontre du caractère "tout automatique" recherché dans les méthodes de suivi.
Pour terminer, la méthode developpée par Shaï Avidan présente un inconvénient majeur : la classification des pixels s'effectue sur un espace de caractéristiques hétérogène. Les caractéristiques utilisées dans le vecteur de caractéristiques combiné peuvent ou non être fiable suivant la situation et, par conséquent, ils peuvent entraîner une augmentation significative de l'erreur Bayésienne globale. De plus, certaines caractéristiques peuvent ne pas être assez discriminantes pour un problème donné, ce qui peut conduire à un biais relativement important dans les résultats de classification. La figure 2.13 illustre parfaitement le problème et introduit en partie une solution possible : diviser l'espace des caractéristiques en plusieurs espaces homogènes. On constate que, contrairement au suivi sur espaces homogènes, le suivi par ensemble tracking perd la cible. Ceci est dû à l'information apportée par les gradients. En effet, on constate lorsque l'on regarde la carte de confiance obtenue sur l'espace des gradients que ce type d'information est inadapté à la situation courante (gradients du fond très marqués). Or cette information se retrouve automatiquement intégrée "de force" dans la carte obtenue par Ensemble Tracking.
Afin d'apporter une solution aux différentes limitations mentionnées ci-dessus, nous avons développé une nouvelle méthode, basée sur l'Ensemble Tracking : l'Ensemble Tracking Modulaire. Présentée dans le chapitre suivant, cette méthode permet de lutter contre le biais induit par les caractéristiques non déterminantes en introduisant une décomposition de l'espace des caractéristiques en sous-espaces (modules) homogènes. Un suivi basé Ensemble Tracking est appliqué sur chacun de ses modules et la problématique de suivi se transforme alors en l'estimation de la position



(a) (b)

Figure 2.13 – Hétérogénéité/Homogénéité des espaces de caractéristiques. La sous-figure (a) (resp. (b)) illustre le suivi effectué par l'algorithme Ensemble Tracking (resp. Ensemble Tracking Modulaire introduit au chapitre 3) sur une séquence vidéo (extraite de la base de données CAVIAR présentée au chapitre 3) présentant le hall d'entrée d'un laboratoire. La première ligne représente une image extraite de cette séquence sur laquelle ont été reportés le rectangle de suivi en trait plein (sortie de l'algorithme) et le rectangle de recherche (en pointillés). La seconde ligne représente différentes cartes de confiance (correspondant au rectangle de recherche) obtenues sur différents espaces de caractéristiques. De gauche à droite, on peut trouver : la carte de confiance obtenue via l'algorithme Ensemble Tracking (espace joint couleur/gradient), la carte obtenue uniquement sur l'espace des couleurs, la carte combinant les deux cartes voisines et enfin la carte obtenue uniquement sur l'espace des histogrammes d'orientation des gradients.

de l'objet ainsi que de la combinaison des cartes de confiance associées (une par module) la plus discriminante. Pour ce faire, l'Ensemble Tracking Modulaire introduit la définition d'un filtre particulaire spécifique. Outre des résultats de suivi très satisfaisants, même en cas de forte occlusion, ainsi que des temps de traitement autorisant le suivi temps-réel, l'algorithme permet, de par sa modularisation, un enrichissement de l'espace des caractéristiques quasi illimité.

Chapitre 3

Ensemble Tracking Modulaire

Sommaire

L'Ensemble Tracking Modulaire se base sur deux améliorations principales. Comme énoncé dans le chapitre précédent, l'inconvénient principal de l'algorithme Ensemble Tracking réside dans la construction de son classifieur fort. Chaque classifieur faible travaille sur un espace hétérogène de caractéristiques, ce qui peut induire un biais dans les résultats de classification. Partant de cette observation, la première modification de l'algorithme initial fut la division de l'espace hétérogène des caractéristiques en plusieurs espaces homogènes (appelés modules) et la construction d'un classifieur fort (afin de suivre l'objet par un algorithme de type Ensemble Tracking) sur chacun de ces espaces.
Partant de cette idée, chaque module est alors capable de générer une décision (plus ou moins fiable) quant à la position de l'objet à suivre. La problématique du suivi en est alors quelque peu modifiée : on cherche désormais à estimer un état caché composé, d'une part, de la position et des dimensions de l'objet suivi et, d'autre part, des paramètres de combinaison linéaire des différentes décisions conduisant à l'observation la plus discriminante.
La modification majeure de l'Ensemble Tracking réside donc dans l'apport d'une solution à cette problématique au travers de la construction d'un filtre particulaire spécifique exploitant une pondération des différents modules utilisés.

3.1 Principe de l'algorithme

L'algorithme que nous proposons ici repose sur deux aspects majeurs : la décomposition de l'espace hétérogène des caractéristiques en plusieurs espaces homogènes et l'apport d'une solution à la nouvelle problématique de suivi.
L'idée principale de la décomposition de l'espace des caractéristiques est de créer un

classifieur fort pour chaque type de vecteur de caractéristiques disponible, rendant ainsi l'algorithme Ensemble Tracking original modulaire (Ensemble Tracking Modulaire (MET)). L'espace de caractéristiques hétérogènes global est divisé en plusieurs espaces homogènes. Un algorithme basé Ensemble Tracking est alors appliqué sur chacun de ces espaces. L'ensemble des cartes de confiance construites par chacun des classifieurs forts (appelés modules) autorisent de ce fait non plus la génération d'une mais de plusieurs décisions distinctes sur la position de l'objet.
Reformulons maintenant la problématique du suivi d'objet en intégrant cet apport. Considérons un état caché X_k, variable aléatoire, dont le calcul s'effectue par un filtre Bayésien récursif. Cet état caché génère une observation Z_k sous la forme d'une scène 3D projetée dans une image. Un modèle d'observation est défini dans le but de chercher à séparer l'objet à suivre de son fond proche. Partant de la décomposition de l'espace des caractéristiques en modules présentée ci-dessus, ce modèle est tout naturellement composé d'une combinaison linéaire des cartes de confiance construites sur chacun de ces modules d'observation homogènes. Considérant, de plus, que l'importance de chaque module au sein de l'observation est variable au fil du temps (perte de pertinence relativement à un fond donné ...), la problématique du suivi peut alors être reformulée comme la détermination, à chaque instant k, de la position et des dimensions de l'objet ainsi que des paramètres de la combinaison linéaire (pondérations) des cartes de confiance construites permettant la plus meilleure "observabilité" de celui-ci (combinaison la plus discriminante). L'état caché X_k est de ce fait défini comme l'union de deux sous états, le premier X_k^{pos} correspondant à la position et aux dimensions de l'objet à suivre et le second X_k^{pon} correspondant à la pondération de chaque carte au sein de la combinaison linéaire offrant la meilleure observabilité :

$$X_k = (X_k^{pos}, X_k^{pon}) \tag{3.1}$$

Dans un cadre probabiliste, la problématique est d'estimer, à chaque nouvelle image de la séquence, la probabilité $p(\mathbf{X}_k|\mathbf{Z}_{1:k})$. Dans le cadre d'un filtrage séquentiel Bayésien, comme le précise la section 1.2.4.1, cette estimation est effectuée en deux étapes : une étape de prédiction (équation de Chapman-Kolmogorov) et une étape de mise à jour (règle de Bayes). Comme aucune hypothèse ne peut être effectuée sur la nature des distributions de probabilités, nous avons choisi de résoudre le problème en approximant les distributions par une méthode de Monte-Carlo et plus particulièrement, nous avons choisi de proposer un filtre particulaire. Deux approches étaient alors envisageables : marginaliser le vecteur d'état en une partie position et dimensions et une partie pondérations, ce qui conduit à une stratégie faisant intervenir deux filtres distincts ou intégrer tous les paramètres au sein d'un même vecteur d'état, ce qui conduit à l'utilisation d'un filtre unique explorant de manière jointe la globalité de l'espace d'état (position/dimensions/pondérations). Les deux approches ont été mises en place et testées et, au vu des différents résultats obtenus, le filtre unique à exploration jointe a été retenu. Seul ce choix est présenté ci-dessous. Cependant, la seconde approche est présentée et comparée à l'approche retenue en

chapitre 4 section 4.3.3.
L'algorithme global Ensemble Tracking Modulaire peut être résumé comme suit : l'espace de caractéristiques est divisé en plusieurs sous-espaces homogènes et un classifieur fort est construit sur chacun de ses sous-espaces à la manière de l'Ensemble Tracking. A chaque nouvelle image, l'ensemble des classifieurs forts construit un ensemble de cartes de confiance servant d'observations à un filtre particulaire dont l'objectif est d'approximer la distribution de probabilité de la position de l'objet (ainsi que ses dimensions) et des pondérations des cartes permettant de l'observer au mieux. Une position "de sortie" de l'objet est extraite de cette distribution et utilisée afin de mettre à jour les différents classifieurs forts avant le passage à l'image suivante. Le schéma synoptique 3.1 présente l'algorithme général de l'Ensemble Tracking Modulaire.

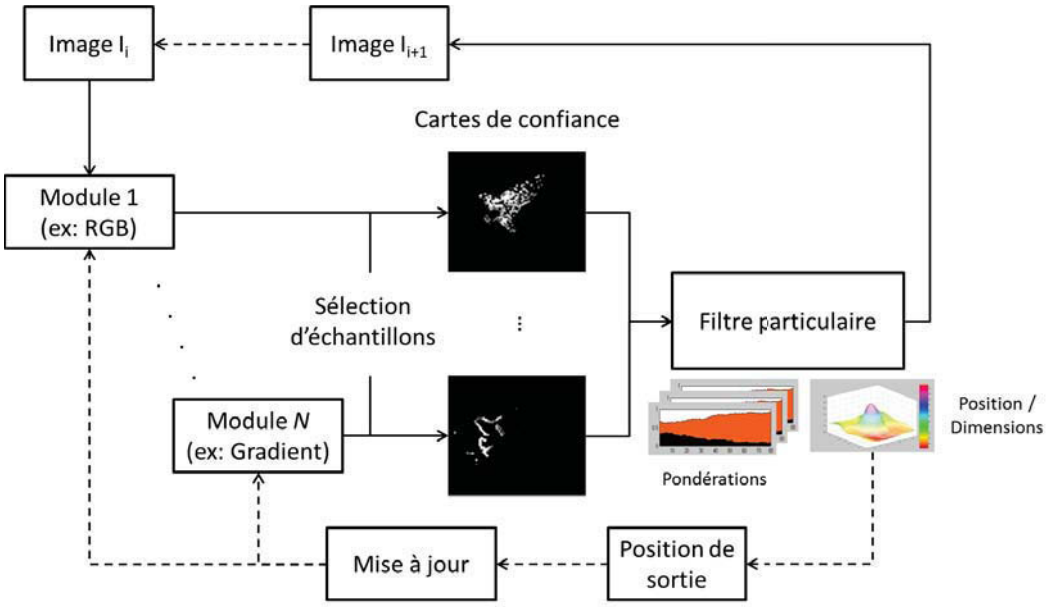


Figure 3.1 – Schéma synoptique de l'algorithme Ensemble Tracking Modulaire.

3.1.1 Échantillonnage des exemples d'apprentissage

L'étape d'initialisation de l'algorithme Ensemble Tracking Modulaire intègre pour chaque classifieur fort la même étape d'initialisation que celle de l'Ensemble Tracking, à la différence près que la totalité des exemples d'apprentissage fond/objet disponibles n'est pas utilisée. Comme indiqué en section 2.3.2, Shaï Avidan insiste sur le fait que l'algorithme Ensemble Tracking utilise l'ensemble des pixels de la région d'intérêt afin d'entraîner/mettre à jour ses classifieurs faibles, ce qui a pour conséquence directe d'empêcher l'algorithme de dépasser les quelques images traitées par seconde. Il nous a donc semblé nécessaire, afin de satisfaire au maximum

nos impératifs de traitement en temps-réel, d'introduire une méthode d'échantillonnage des exemples pour réduire les temps de traitement. La méthode d'extraction des pixels proposée ici est spécifique à chacun des modules.
Pour chaque module ne possédant aucune condition d'extraction particulière (par exemple le module de couleurs RVB), les pixels utilisés sont tirés aléatoirement selon une distribution de probabilité gaussienne multivariée centrée paramétrée de sorte que le même nombre de pixels soit extrait de l'intérieur et de l'extérieur du rectangle englobant l'objet à suivre. Ce choix permet à la zone d'apprentissage d'être sélectionnée dynamiquement et par conséquent de gérer directement un panel de fonds, que l'objet est susceptible de rencontrer, plus large.
Pour les autres modules, ceux correspondant à une information de type contours (par ex. le module travaillant sur des histogrammes d'orientation du gradient), il est aisé de constater qu'ils ne peuvent être défini que sur un ensemble de pixels de gradient fort. Par conséquent, ne sont sélectionnés pour ce type de module que les pixels dont le gradient est le plus élevé dans les différentes cellules d'une grille régulière superposée à l'image.

3.1.2 Filtre particulaire

À partir des exemples d'apprentissage extraits grâce aux méthodes présentées dans la section précédente, un classifieur fort est construit sur chaque espace de caractéristiques. À l'itération suivante, ces classifieurs forts sont utilisés, tout comme dans l'algorithme Ensemble Tracking original, pour construire une carte de confiance de la position objet. Le rôle du filtre particulaire est de maintenir au fil du temps un ensemble de particules correspondant de manière jointe aux positions (ainsi qu'aux dimensions) probables de l'objet et aux pondérations à appliquer dans la combinaison linéaire des cartes de confiance pour obtenir la meilleure observabilité. Pour ce faire, le filtre suit le schéma algorithmique classique des filtres à particules de type MCMC (cf. section 1.2.4.4), la propagation des particules s'effectuant relativement aux différentes cartes de confiance construites à l'itération courante.

3.1.2.1 Vecteur d'état

Chaque particule i à un instant t correspond à un rectangle spécifique englobant l'objet ainsi qu'à un ensemble de poids à appliquer aux cartes de confiance afin de passer de M cartes à une carte unique par moyenne pondérée (carte unique reflétant la meilleure observabilité). Ainsi :

$$\forall i \in [1, P], \forall t, X_i^t = (x_i^t; y_i^t; w_i^t; h_i^t; \left\{w_{i,m}^t\right\}_{m=1}^{M}) \tag{3.2}$$

où x_i^t et y_i^t représentent les coordonnées cartésiennes dans l'image du centre du rectangle englobant i à l'instant t, w_i^t et h_i^t représentent ses dimensions (respectivement largeur et hauteur), $w_{i,m}^t$ correspond au poids à appliquer à la carte m à l'instant t, M représente le nombre de modules utilisés et enfin P le nombre de particules utilisées.

Le score de chacune de ces particules est calculé relativement aux cartes de confiance construites par les différents modules. Or ces cartes ne sont disponibles que pour certains pixels dans le cadre image (échantillonnage). Il a donc été nécessaire, afin d'être sûr de posséder des données pour calculer le score de chaque particule, de contraindre chacune d'entre elle à ne représenter (pour la partie position/dimensions) que des rectangles dans l'image (coordonnées cartésiennes minimales et maximales ainsi que dimensions maximales).
De la même manière, un rectangle de taille trop petite ne représentera pas suffisamment de données pour un calcul cohérent du score de la particule. Une seconde contrainte a, par conséquent, été appliquée relativement aux dimensions minimales de chaque rectangle. Enfin, afin de réaliser une combinaison convexe des cartes de confiance, deux contraintes sont appliquées aux poids contenus dans chaque particule :

$$\begin{cases} \forall i \in [1, P], \forall t, \forall m \in [1, M], w_{i,m}^t \in [0, 1] \\ \forall i \in [1, P], \forall t, \sum_{i=1}^{M} w_{i,m}^t = 1 \end{cases} \tag{3.3}$$

3.1.2.2 Modèle de propagation

Le premier modèle du schéma algorithmique classique des filtres MCMC est le modèle de propagation. Celui-ci permet de passer, dans notre cas, d'un ensemble de particules représentant position, dimensions et pondérations à un instant $t-1$ à un ensemble à l'instant t. La première étape de notre algorithme de propagation est la détermination de la sous-partie du vecteur d'état impactée (position/dimensions ou pondérations) par la propagation (les deux parties ne sont jamais propagées en même temps). Celle-ci s'effectue via une marche aléatoire.
Le modèle développé comprend trois types de propagation des position/dimensions : une mise à jour des paramètres de position, une mise à jour des paramètres de dimension ou une mise à jour combinée de tous ces paramètres. Chaque type de propagation est associé à une probabilité spécifique. Suite à diverses expérimentations, ces probabilités ont été fixées à 0.75 pour la position, 0.2 pour les dimensions et 0.05 pour le combiné (l'hypothèse de mouvement lent de la cible implique une probabilité de changement de dimensions (éloignement/rapprochement) inférieure à celle de changement de position).
Dans le cas d'une propagation de la première partie du vecteur d'état, la seconde étape de la propagation consiste donc en un tirage aléatoire afin de déterminer les paramètres impactés. Les paramètres choisis sont ensuite mis à jour durant la troisième étape. La mise à jour de position correspond à l'ajout aux coordonnées du centre du rectangle englobant d'un bruit additif gaussien centré et de variance égale au carré d'une constante fixée (relativement aux dimensions de l'objet). La mise à jour des dimensions correspond, quant à elle, à l'ajout aux dimensions de ce même rectangle d'un bruit multiplicatif gaussien de moyenne un et réduit. Enfin, bien entendu, la mise à jour combinée correspond à l'application successive de ces deux bruits distincts. La dernière étape de la mise à jour vérifie alors si les différentes

contraintes appliquées aux particules ne sont pas violées et applique une transformation corrective si c'est le cas (la transformation corrective est utilisée au lieu de la limitation des intervalles de tirages gaussiens afin de ne pas modifier les densités de probabilité de chaque type de propagation).
Dans le cas où la propagation impacte non pas les position/dimensions mais la pondération, la seconde étape consiste à choisir aléatoirement quel poids sera modifié. Ce poids est alors mis à jour par l'ajout d'un bruit additif gaussien centré et de variance égale à d'une constante fixée. Enfin la somme des poids est à nouveau normalisée de manière à valoir toujours 1.
Le modèle global de propagation est repris en Algorithme 3.1.

3.1.2.3 Modèle d'observation

Le modèle d'observation est utilisé, quant à lui, afin de déterminer un score pour chaque particule. Ce score est alors utilisé par l'algorithme d'acceptation-rejet de Metropolis-Hastings (cf. section 1.2.4.4) afin de vérifier si la particule récemment construite fera ou non partie de la chaine de Markov en cours de construction. Le modèle que nous avons mis en place se base bien évidemment sur les différentes cartes de confiance construites par chacun des modules utilisés.
Pour chaque particule et chaque carte de confiance construite, deux scores moyens de classification sont calculés : le score correspondant à la moyenne des scores de classification à l'intérieur du rectangle englobant relatif à la particule courante (a) et celui correspondant à la moyenne des scores de classification à l'extérieur de ce rectangle mais à l'intérieur du rectangle de région d'intérêt (de même centre mais de dimensions trois fois supérieures) (b). Ces deux scores sont alors utilisés afin de calculer le score global du module concernant la partie position/dimensions donné via la relation

$$(a) \times (1 - (b)) \tag{3.4}$$

Le score global de la particule est finalement calculé par la somme pondérée de chacun de ces scores de module (la pondération utilisée étant celle contenue dans la particule courante).
Le modèle d'observation présenté se base sur l'ensemble des cartes de confiance construites par les différents modules en jeu. Afin de supprimer de ces cartes les valeurs aberrantes et de transformer les marges de classification des différents classifieurs forts en valeur de probabilité, une fonction sigmoïde est appliquée, en aval de la construction des cartes mais en amont de l'application du filtre particulaire, à toutes les valeurs de confiance utiles $VCU(x, y)$ contenues dans chaque carte. Ainsi, pour chaque module m à l'itération t :

$$c_m^t(x, y) = \frac{1}{1 + \exp(A_m.VCU_m^t(x, y) + B_m)}, \forall (x, y) \in \Omega \tag{3.5}$$

Données :	$X_i^{t-1} = (x_i^{t-1}; y_i^{t-1}; w_i^{t-1}; h_i^{t-1}; \left\{w_{i,m}^{t-1}\right\}_{m=1}^{M})$	Particule i à $t-1$
	dep, gro	Paramètres de déplacement et de grossissement relatifs maximaux
	$ampl$	Paramètre de variance du bruit de pondération
Résultats :	$X_i^t = (x_i^t; y_i^t; w_i^t; h_i^t; \left\{w_{i,m}^{t}\right\}_{m=1}^{M})$	Particule i propagée

- Choix du type de propagation
- Suivant (type de propagation)
  - Position/Dimensions :
    - Choix du type de propagation de position/dimensions
    - Suivant (type de propagation de position/dimensions)
      - Position :
        $x_i^t = x_i^{t-1} + (dep * \mathcal{N}(0,1)|_{[-1,1]})$
        $y_i^t = y_i^{t-1} + (dep * \mathcal{N}(0,1)|_{[-1,1]})$
      - Dimensions :
        $w_i^t = w_i^{t-1} * \mathcal{N}(1,1)|_{[1-gro,1+gro]}$
        $h_i^t = h_i^{t-1} * \mathcal{N}(1,1)|_{[1-gro,1+gro]}$
      - Combiné :
        $x_i^t = x_i^{t-1} + (dep * \mathcal{N}(0,1)|_{[-1,1]})$
        $y_i^t = y_i^{t-1} + (dep * \mathcal{N}(0,1)|_{[-1,1]})$
        $w_i^t = w_i^{t-1} * \mathcal{N}(1,1)|_{[1-gro,1+gro]}$
        $h_i^t = h_i^{t-1} * \mathcal{N}(1,1)|_{[1-gro,1+gro]}$
    - Vérification de la non-violation des contraintes et application si nécessaire d'un traitement correctif
  - Pondération :
    - $i = aleatoire(1, M)$
    - $\begin{cases} w_{p,i}^t = w_{p,i}^{t-1} + \mathcal{N}(0, ampl)|_{[-w_{p,i}^{t-1}, 1-w_{p,i}^{t-1}]} \\ \forall m \in [1,M], m \neq i, w_{p,m}^t = w_{p,m}^{t-1} \end{cases}$
    - $\forall m \in [1,M], w_{p,m}^t = \frac{w_{p,m}^t}{\sum_{i=1}^{M} w_{p,i}^t}$

Algorithme 3.1: Modèle global de propagation des particules

où Ω représente l'ensemble des pixels pour lesquels une valeur de confiance a été calculée (cf. échantillonnage de l'image selon le module utilisé), $VCU_m^t(x,y)$ représente la valeur de confiance calculée par le classifieur fort m considéré en (x,y) à t et A_m et B_m représentent les deux paramètres de sigmoïde associés au module m. Ces paramètres sont fixés, pour chacun des modules m en jeu, en optimisant une fonction d'entropie croisée [Niculescu-Mizil 2005] [Chateau 2006] sur la carte de confiance de m correspondant à la première image de la séquence :

$$argmin_{(A,B)} \left\{ - \sum_{(x,y)\in\Omega} e_{x,y} \log(p_m(x,y)) + (1-e_{x,y}) \log(1-p_m(x,y)) \right\}, \quad (3.6)$$

où

$$p_m(x,y) = \frac{1}{1+\exp(A.VCU_m(x,y)+B)} \quad (3.7)$$

et $e_{x,y} \in \{0,1\}$ représente l'étiquette du pixel utile (x,y).
Le modèle global d'observation est repris en Algorithme 3.2.

Données :	$X_*^t = (x_*^t; y_*^t; w_*^t; h_*^t; \{w_{*,m}^t\}_{m=1}^M)$	Particule à évaluer à t
	$Z^t = \{C_m^t = (\{c_m^t(x,y)\})\}_{m=1}^M$	L'ensemble des M
	avec $c_m^t(x,y) = \frac{1}{1+\exp(A_m.VCU_m^t(x,y)+B_m)}$	cartes de confiance à t
Résultats :	$p(Z^t\|X_*^t)$	Score de la particule à t

- Pour tout module m ($\in [1,M]$) utilisé
  - $S_m|_{\Omega_1} = \frac{\sum_{(x,y)\in\Omega_1} c_m^t(x,y)}{taille(\Omega_1)}$ avec $\Omega_1 = Rect(x_*^t; y_*^t; w_*^t; h_*^t)$
  - $S_m|_{\Omega_2} = \frac{\sum_{(x,y)\in\Omega_2} c_m^t(x,y)}{taille(\Omega_2)}$ avec $\Omega_3 = Rect(x_*^t; y_*^t; 3w_*^t; 3h_*^t)$ et $\Omega_2 = \Omega_3 \backslash \Omega_1$
  - $S_m = S_m|_{\Omega_1} \times (1 - S_m|_{\Omega_2})$
- $p(Z^t|X_*^t) = \sum_{m=1}^M w_{*,m}^t \times S_m$

Algorithme 3.2: Modèle d'observation des particules combinées

3.1.2.4 Mise à jour des modules

L'étape postérieure à l'étape de traitement de l'image courante par le filtre particulaire est l'étape de mise à jour des différents modules. Nous avons choisi de conserver l'algorithme de mise à jour [Avidan 2007] du classifieur fort et de l'appliquer à chacun de nos modules en jeu. Pour ne conserver que les K meilleurs classifieurs forts à chaque itération, la méthode de mise à jour de l'Ensemble Tracking se base sur la nouvelle position déterminée par Mean Shift à l'image courante. Afin d'appliquer la méthode identiquement à l'originale sur l'ensemble de nos classifieurs, il est nécessaire de déterminer une position de mise à jour unique à partir de l'ensemble des positions contenues dans les différentes particules du filtre. Chaque pixel de l'image se voit tout d'abord attribuer un score égal au nombre de particules pour lesquelles le rectangle englobant le contient. Chaque pixel de l'image est alors considéré comme

un pixel objet à partir du moment où son score est supérieur à la moitié du nombre total de particules. Autrement dit, chaque pixel image est considéré comme un pixel objet s'il se retrouve dans le rectangle englobant l'objet dans plus de la moitié des particules. Partant de ce postulat, chaque échantillon de l'ensemble courant est étiqueté relativement à son score.
Afin de contrer l'effet de drifting inhérent aux algorithmes de mise à jour dans les méthodes de classification, les échantillons courants sont utilisés en parallèle de l'ensemble des échantillons provenant de l'image initiale. A chaque mise à jour et pour chaque module, nous disposons donc de deux ensembles d'échantillons étiquetés (notés courant et initial). A partir de ces ensembles nous construisons quatre groupes d'échantillons (deux groupes de positifs et deux groupes de négatifs) par tirages aléatoires avec remise dans les ensembles courant et initial. Chaque paire de groupes (un groupe de positifs apparié à un groupe de négatifs) est finalement utilisé afin d'effectuer la mise à jour du classifieur fort ou afin de ré-estimer les paramètres de sigmoïde qui seront utilisés par le classifieur pour la construction de carte de confiance modulaire. L'algorithme général de mise à jour est repris en Algorithme 3.3 ($taille(.)$ représente la fonction qui retourne le cardinal de l'ensemble passé en paramètre, $position(s^t_{i,m})$ représente les coordonnées cartésiennes de l'échantillon $s^t_{i,m}$, $Rect(X^t_i)$ représente le rectangle de position contenu dans la particule X^t_i et enfin $maj(H^t_m)$ représente l'étape de mise à jour du classifieur fort H^t_m).
La mise à jour des modules étant effectuée en aval de la propagation de la pondération des cartes de confiance (reflétant l'efficacité relative de chaque module), le regain d'efficacité d'un module particulier suite à la mise à jour ne sera reflété qu'à partir de l'image suivante.

3.1.2.5 Algorithme global

L'algorithme global d'Ensemble Tracking Modulaire est repris en Algorithme 3.4.

3.1.3 Optimisations algorithmiques

Afin de satisfaire au maximum notre objectif de suivi en temps réel, différentes optimisations algorithmiques ont été mises en place.
La première est bien entendu l'étape d'échantillonnage de l'image qui permet de réduire considérablement la quantité d'information à traiter par les classifieurs faibles durant les phases d'apprentissage et de mise à jour.
La seconde optimisation concerne l'utilisation des procédés connus sous le nom d'image intégrale [Viola 2001] et d'histogramme intégral [Porikli 2005]. L'image intégrale est une représentation intermédiaire d'une image contenant, en chacun de ses points, la somme inclusive des pixels situés au dessus et à gauche de ce point. Ainsi, pour toute image i définie sur Ω, l'image intégrale ii de i définie également sur Ω est donnée par :

Données :	$\left\{X_i^t\right\}_{i=1}^P$	Ensemble des P particules à t
	$\forall m \in [\![1, M]\!]$, $\left\{s_{i,m}^t = (\mathbf{x}_{i,m}^t, y_{i,m}^t)\right\}_{i=1}^N$	Ensemble des N échantillons courants pour chaque module
	$\left\{H_1^t, ..., H_M^t\right\}$	Ensemble des M classifieurs forts à t
Résultats :	$\left\{H_1^{t+1}, ..., H_M^{t+1}\right\}$	Ensemble des M classifieurs forts pour $t+1$
	$\left\{(A_m^{t+1}, B_m^{t+1})\right\}_{m=1}^M$	Ensemble des paires de paramètres de sigmoïde pour $t+1$

Pour chaque module m ($\in [\![1, M]\!]$) utilisé

- $\forall i \in [\![1, N]\!]$, étiquetage de l'échantillon $s_{i,m}^t$:

$$\begin{cases} y_{i,m}^t = 1 \quad \text{si} \quad \frac{taille(\left\{X_i^t | position(s_{i,m}^t) \in Rect(X_i^t)\right\})}{P} > \frac{1}{2} \\ y_{i,m}^t = -1 \quad \text{sinon} \end{cases}$$

- Ordonnancement et séparation des échantillons courants et initiaux positifs et négatifs en 4 groupes : $G_{Pos}^{1,m}, G_{Neg}^{1,m}, G_{Pos}^{2,m}, G_{Neg}^{2,m}$
- Construction des ensembles de travail :

$$\begin{cases} \text{Construction par tirages aléatoires dans } G_{Pos}^{1,m} \text{ et } G_{Pos}^{2,m} \text{ d'un premier} \\ \text{ensemble général d'échantillons positifs : } E_{Pos}^{1,m} \\ \text{Construction par tirages aléatoires dans } G_{Neg}^{1,m} \text{ et } G_{Neg}^{2,m} \text{ d'un premier} \\ \text{ensemble général d'échantillons négatifs : } E_{Neg}^{1,m} \\ \text{Construction par tirages aléatoires dans } G_{Pos}^{1,m} \text{ et } G_{Pos}^{2,m} \text{ d'un second} \\ \text{ensemble général d'échantillons positifs : } E_{Pos}^{2,m} \\ \text{Construction par tirages aléatoires dans } G_{Neg}^{1,m} \text{ et } G_{Neg}^{2,m} \text{ d'un second} \\ \text{ensemble général d'échantillons négatifs : } E_{Neg}^{2,m} \end{cases}$$

- Mise à jour du classifieur fort : $H_m^{t+1} = maj(H_m^t)$ en utilisant $E_{Pos}^{1,m}$ et $E_{Neg}^{1,m}$
- Estimation des nouveaux paramètres de sigmoïde : (A_m^{t+1}, B_m^{t+1}) en utilisant $E_{Pos}^{2,m}$ et $E_{Neg}^{2,m}$

Algorithme 3.3: Algorithme général de mise à jour

$$\forall (x, y) \in \Omega, ii(x, y) = \sum_{0 \leq x' \leq x, 0 \leq y' \leq y} i(x', y') \tag{3.8}$$

L'avantage de cette représentation est qu'elle permet de calculer la somme des valeurs de n'importe quelle zone rectangulaire en seulement quatre accès à l'image intégrale (cf. Figure 3.2) et donc en temps constant quelle que soit la taille de la zone.

La méthode des histogrammes intégraux se base sur le concept de l'image intégrale et le généralise afin de stocker pour chaque point, non plus la somme des pixels au dessus et à gauche mais la somme des histogrammes correspondant aux points au dessus et à gauche. Un vecteur de valeurs correspondants aux différentes classes de l'histogramme est ainsi stocké pour chaque point. La construction de ce type

Données :	$I_1, ..., I_n$	$I_i : p \longrightarrow I(p)$ n images extraites
		$\Delta \subset \mathbb{Z}^2 \longmapsto \mathbb{R}$ d'une séquence vidéo
	$r_1 = (x_1; y_1; w_1; h_1) \subset \Delta$	Rectangle de l'objet dans l'image I_1
Résultats :	$\{r_j\}_{j=2}^{n}$	Rectangles de l'objet dans les images I_j

Initialisation (pour l'image I_1) :

1. Pour chaque module $m(\in [\![1, M]\!])$:
   (a) Echantillonnage de l'image $\left\{s_{i,m}^1 = (\mathbf{x}_{i,m}^1, y_{i,m}^1)\right\}_{i=1}^{N}$ (cf. section 3.1.1)
   (b) Initialisation du classifieur fort H_m^2 du module m (cf. Algorithme 2.2) qui sera utilisé sur l'image I_2
   (c) Estimation des paramètres de sigmoïde A_m^2 et B_m^2 (cf. Equation 3.5) correspondants à H_m^2
   (d) Poids initial du module : $w_m = \frac{1}{M}$
2. Initialisation du filtre particulaire :
   $X_i^1 = (x_1; y_1; w_1; h_1; \{w_m\}_{m=1}^{M}), \forall i \in [1, P]$

Pour chaque nouvelle image I_j :

1. Pour chaque module $m(\in [\![1, M]\!])$:
   (a) Echantillonnage de l'image $\left\{s_{i,m}^j = (\mathbf{x}_{i,m}^j, y_{i,m}^j)\right\}_{i=1}^{N}$
   (b) Calcul de la carte $C_m^j = (\left\{c_m^j(x, y) = \frac{1}{1+\exp(A_m.VCU_m^j(x,y)+B_m)}\right\})$ en utilisant H_m^j
2. Traitement par le filtre particulaire :
   (a) Tirage et propagation (cf. 3.1.2) d'une particule X_1^j parmi $\left\{X_i^{j-1}\right\}_{i=1}^{P}$
   (b) Calcul du score $p(Z^t|X_1^j)$ de X_1^j (cf. 3.1.2)
   (c) Pour $p = 2...P + \delta$
      i. Tirage et propagation d'une particule X_p^j parmi $\left\{X_i^{j-1}\right\}_{i=1}^{P}$
      ii. Calcul du score $p(Z^t|X_p^j)$ de X_p^j
      iii. Détermination de l'acceptation ou du rejet de X_p^j suivant l'algorithme de Metropolis-Hastings (si rejet : $X_p^j = X_{p-1}^j$)
   (d) Suppression des δ premières particules (état non stationnaire)
3. $r_j = Rect(\frac{1}{P}\sum_i x_i^j; \frac{1}{P}\sum_i y_i^j; \frac{1}{P}\sum_i w_i^j; \frac{1}{P}\sum_i h_i^j)$ uniquement pour affichage
4. Pour chaque module $m(\in [\![1, M]\!])$:
   (a) Etiquetage des échantillons courants du module m
   (b) Construction des ensembles de travail
   (c) Mise à jour du classifieur fort : $H_m^{j+1} = maj(H_m^j)$
   (d) Estimation des nouveaux paramètres de sigmoïde : (A_m^{j+1}, B_m^{j+1})

Algorithme 3.4: Algorithme global d'Ensemble Tracking Modulaire à filtre unique

d'histogramme suit le principe général de construction des images intégrales puisque pour chaque point l'histogramme est obtenu par la propagation des histogrammes

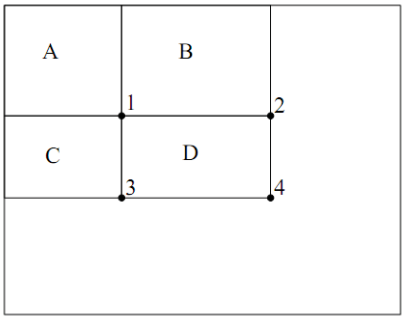


Figure 3.2 – Principe de calcul de somme en utilisant la représentation en image intégrale. La valeur de l'image intégrale en 1 est la somme des pixels du rectangle A. La valeur en 2 est la somme du rectangle $A + B$, en 3 celle du rectangle $A + C$ et en 4 celle du rectangle $A + B + C + D$. La somme des pixels du rectangle D peut donc être calculée via la relation $4 + 1 - (2 + 3)$.

déjà rencontrés (au dessus et à gauche) suivie de l'ajout de la contribution du point courant (cf. Figure 3.3).

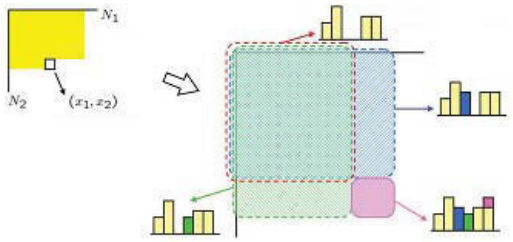


Figure 3.3 – Principe de construction d'un histogramme intégral. L'histogramme pour le point (x_1, x_2) est obtenu par propagation des histogrammes des points au dessus et à gauche suivie de l'ajout de la contribution du point courant (rose). Source [Porikli 2005].

De la même manière que pour les images intégrales, les histogrammes intégraux permettent de calculer l'histogramme de n'importe quelle zone rectangulaire en un temps constant.
Le principe de l'algorithme Ensemble Tracking implique un calcul de caractéristiques pour chacun des échantillons utilisés. Malgré notre étape d'échantillonnage qui permet de réduire le nombre de calculs nécessaires (moins d'échantillons implique moins de calcul), le calcul des caractéristiques pour les échantillons des modules basés sur des histogrammes (cf. section 3.2) reste une étape très gourmande en temps de traitement. L'utilisation, dans ce cas, d'histogrammes intégraux permet d'alléger considérablement la charge calculatoire.
Enfin la dernière optimisation est plus spécifique au matériel utilisé. Le suivi développé ici ayant pour but d'être intégré à l'enregistreur numérique de la société et

cet enregistreur étant composé de matériel sélectionné et, par conséquent, fixé (hors évolution de gamme), l'utilisation de bibliothèques de traitement spécifiques semble donc légitime. Nous nous sommes ainsi tourné vers la librairie Intel® IPP (pour Integrated Performance Primitives). Cette librairie extensible propose un ensemble de fonctions logicielles hautement optimisées (et utilisables en mode multi-coeur) pour le multimédia, le traitement de données et les applications de communication. Pour ce faire, elle fait correspondre ses algorithmes fonctionnels fondamentaux avec des optimisations bas niveau basées sur les caractéristiques disponibles des processeurs Intel® telles que les instructions SSE et d'autres jeux d'instructions optimisées.
Les temps de traitement de notre algorithme dépendent fortement du nombre de modules utilisés à chaque image. A titre d'exemple, les temps de traitement obtenus en utilisant trois modules (un module couleur de type RVB, un module d'histogrammes d'orientation du gradient et un module d'histogrammes de modèles binaires locaux) ne dépassent pas les 70 ms, ce qui permet un suivi à une fréquence supérieure à 14 images par seconde. Les temps de traitement obtenus en n'utilisant que deux des trois modules (RVB et LBP) ne dépassent pas quant à eux les 40 ms, ce qui permet un suivi à une fréquence supérieure à 25 images par seconde (temps-réel pour l'oeil humain).

3.2 Caractéristiques utilisées

L'ensemble Tracking Modulaire est un algorithme générique. En effet il se base sur la construction d'un classifieur fort par module utilisé, c'est à dire par espace de caractéristiques, et ce indépendamment de la nature de cet espace. L'unique "contrainte" régissant l'utilisation d'un espace de caractéristiques particulier est la nécessité de pouvoir extraire ou construire "rapidement" (afin de ne pas enfreindre la règle du suivi temps-réel), à partir d'une image d'entrée (couleur ou en niveaux de gris), un vecteur de caractéristiques non vide pour chacun des pixels de l'image. En se basant sur cette contrainte, tout espace peut être utilisé à partir du moment où il peut être imaginé [Dalal 2005] [Zenzo 1986] [Haralick 1973] [Ojala 2002] [Tamura 1978].
Cette généricité offre également à l'algorithme un aspect évolutif. En effet, l'algorithme permet, assez aisément, d'introduire de nouveaux modules si besoin est. Cet aspect évolutif peut également se traduire par la possibilité de combiner l'algorithme présenté précédemment avec un algorithme de détection ad-hoc (détection de piétons par exemple) permettant de récupérer, pour chaque image, une information a priori sur la position de l'objet suivi.
Comme annoncé précédemment, l'algorithme supporte une très large variété de modules. Les différents modules effectivement implémentés actuellement peuvent être séparés en trois catégories : les modules basés sur des caractéristiques colorimétriques, les modules basés sur une information relative aux contours de l'objet et les modules basés sur des informations de texture.

3.2.1 Couleurs

L'information de couleur d'une image peut être représentée de différentes manières suivant l'espace colorimétrique utilisé. Il existe de très nombreux espaces de couleurs possédant chacun leurs caractéristiques propres mais partageant le même principe de base. Chaque espace créé une décomposition, selon ses propres paramètres, de toutes les couleurs en un ensemble de composantes dites primaires. Chaque point d'une image possédant une couleur spécifique, celle-ci peut être décomposée de la sorte. A chaque point de l'image va donc correspondre un ensemble de composantes primaires résultant de la décomposition de sa couleur propre. Cette décomposition peut alors être utilisée comme ensemble de caractéristiques de classification pour les différents pixels de l'image.
Dans l'espace colorimétrique RVB, chaque couleur apparait comme une combinaison des couleurs primaires : rouge (R), vert (V) et bleu (B). Le sous-espace d'intérêt est un cube représenté en Figure 3.4 (les valeurs RVB sont normalisées entre 0 et 1).

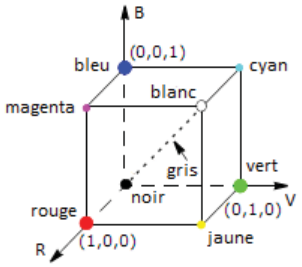


Figure 3.4 – Modèle colorimétrique RVB. Les couleurs primaires R, V et B correspondent à trois coins. Les couleurs dites secondaires : cyan, magenta et jaune correspondent à trois autres coins. La couleur noire est à l'origine du repère et la couleur blanche au coin le plus éloigné de l'origine.

Les niveaux de gris s'étendent du noir au blanc sur la diagonale séparant ces deux points. Les différentes couleurs "utiles" sont tous les points sur ou à l'intérieur du cube défini par les trois vecteurs émanant de l'origine. Ainsi toute image (resp. tout pixel) dans le modèle de couleur RVB est constitué de trois plans (resp. composantes) indépendants, un pour chaque couleur primaire.
Le modèle de couleurs YUV est le modèle de couleur de base de la diffusion audiovisuelle analogique pour télévision couleur. Le modèle de couleurs YUV est "dérivé" du modèle RVB. Il comprend une composante appelée luminance (Y) et deux autres composantes correspondants à deux différences de couleurs (U et V). La luminance peut être calculée comme une somme pondérée des composantes rouge, verte et bleue ; les différences de couleurs, appelées aussi chrominances, sont formées en ôtant la luminance aux composantes bleue et rouge. De nombreuses combinaisons de valeurs YUV résultent en des valeurs RVB invalides puisque l'ensemble des couleurs RVB possibles n'occupe qu'une partie de l'espace YUV. La Figure 3.5 montre le

bloc de couleurs dans l'espace YUV qui correspond au cube de couleurs de l'espace RVB (les valeurs RVB sont toujours normalisées entre 0 et 1).

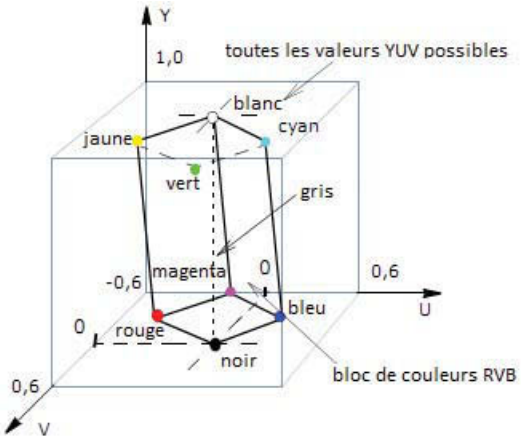


Figure 3.5 – Modèle colorimétrique YUV. Les couleurs primaires R, V et B correspondent à trois coins. Les couleurs dites secondaires : cyan, magenta et jaune correspondent à trois autres coins. La couleur noire est à l'origine du repère et la couleur blanche au coin le plus éloigné de l'origine.

Enfin le modèle de couleurs HSV (Hue, Saturation, Value en anglais ou TSV pour Teinte, Saturation, Valeur en français) a pour but la manipulation des couleurs de manière plus intuitive. La teinte (H) définie la couleur elle même. Les valeurs de l'axe de teinte varient entre 0 et 360 en commençant et terminant par le rouge tout en passant par le vert, le bleu et toutes les couleurs intermédiaires. La saturation (S) indique le degré avec lequel la teinte diffère du gris neutre. Les valeurs varient entre 0, qui indique une absence de saturation de couleur, et 1, qui représente la plus grande saturation possible pour une couleur donnée sous une illumination donnée. La composante d'intensité, appelée valeur (V), indique le niveau d'illumination. Elle varie entre 0 (noir, aucune lumière) et 1 (blanc, pleine lumière). La valeur maximale de saturation (S=1) est donc obtenue à V=1. L'espace de couleurs HSV est par nature un cylindre mais il est plus généralement représenté sous forme de cône ou encore de cône hexagonal (cf. Figure 3.6) car cette représentation définit le sous-espace de l'espace HSV correspondant à des valeurs de RVB valides.
La figure 3.7 illustre la décomposition des couleurs d'une image sur les trois espaces et leur représentation dans l'espace RVB (toutes les composantes sont ramenées à l'intervalle discret $[\![0, 255]\!]$).

3.2.2 Contours

La seconde catégorie de caractéristiques se base sur des informations de contours. La détection des contours d'une image permet de réduire de manière significative la quantité de données utiles grâce à l'élimination des informations jugées moins pertinentes, et ce tout en préservant les propriétés structurelles importantes de l'image.

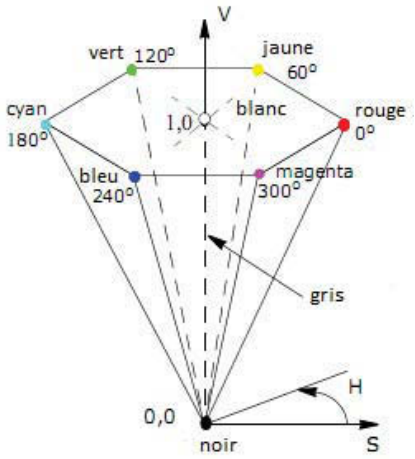


Figure 3.6 – Modèle colorimétrique HSV. L'axe vertical représente la valeur V, le sommet V=0 correspondant à la couleur noir.

(a) (b) (c)

Figure 3.7 – Décomposition des couleurs d'une image sur les trois espaces (a) RVB, (b) YUV et (c) HSV et représentation de cette décomposition dans l'espace RVB.

Au vu de la simplicité de la méthode, l'extraction d'informations de contours d'une image passe en général par la recherche des maxima locaux de l'intensité du gradient de celle-ci. Or puisque l'intensité d'une image numérique est discrète, les dérivées de la fonction intensité (et par conséquent le gradient) ne peuvent pas être définies, si ce n'est sous une hypothèse de continuité de la fonction intensité qui a été échantillonnée. Le gradient ne peut par conséquent être qu'approximé, avec plus ou moins de précision, en chaque point de définition d'échantillonnage de la fonction intensité (approximation différence). Cette approximation du gradient se traduit par l'application sur l'image de filtres (représentés sous forme de matrices de convolution).
Il existe un très grand nombre de filtres permettant, avec plus ou moins de réussite, d'approximer le gradient d'intensité en tout point de l'image. Au vu de leur simplicité, de leur rapidité d'exécution et de leurs résultats, somme toute peu précis, mais représentant le bon compromis entre temps de traitement et efficacité, nous avons choisi d'utiliser les filtres de Sobel [Sobel 1968]. Ces filtres se présentent sous la forme de deux matrices de convolution 3×3 :

$$\begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix} \begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix}$$

La convolution de chacune de ces matrices avec l'image d'entrée permet d'obtenir une des composantes (resp. horizontale et verticale) de l'approximation du gradient de l'image.
Une fois ces approximations obtenues à partir de l'image d'entrée en niveaux de gris, il est alors possible de les représenter au niveau pixel de bien des manières différentes, chacune de ces manières permettant de construire un type de caractéristiques différentes. Parmi celles-ci on peut distinguer la construction d'un histogramme d'orientation du gradient qui est relativement simple à calculer, contient une information riche et a déjà été largement utilisé avec succès par le passé à des fins de détection et de reconnaissance [Freeman 1994] [Dalal 2005]. Le principe de construction de l'histogramme local d'orientation des gradients pour un pixel donné est le suivant : sont calculés, pour chaque pixel p d'une fenêtre de taille prédéfinie (15×15 dans notre cas) englobant notre pixel P d'intérêt et centrée en ce dernier, l'approximation de la norme G du gradient en p donnée par (en notant G_x l'approximation du gradient horizontal en p et G_y l'approximation du gradient vertical en p) :

$$G = \sqrt{G_x^2 + G_y^2} \tag{3.9}$$

ainsi que l'approximation de l'orientation Θ du gradient en p donnée par :

$$\Theta = \arctan(\frac{G_y}{G_x}) \tag{3.10}$$

La valeur correspondant à la classe de l'histogramme contenant Θ est alors incrémentée de $\frac{G}{G_N}$, G_N correspondant à une constante de normalisation calculée relativement

à l'ensemble des normes des gradients sur la fenêtre de travail. On obtient finalement, pour chaque pixel P à traiter, un vecteur de caractéristiques dont les composantes (8 dans notre cas) représentent chacune la somme des normes des gradients de la fenêtre englobante locale dont l'orientation correspond à la classe représentée.
Le second type de caractéristiques que nous utilisons dans cette catégorie se base sur un type de gradient particulier appelé gradient de Di Zenzo [Zenzo 1986]. La particularité de ce gradient, outre le fait qu'il n'opère plus sur la conversion en niveaux de gris d'une image d'entrée couleurs mais directement sur les trois composantes couleurs R, V et B de cette image, est qu'il tire partie, contrairement aux méthodes dites de base pour le calcul de gradient sur une image multi-spectrale (par ex. la conservation du maximum des valeurs absolues), de la coopération entre les composantes de l'image. L'approximation de l'orientation Θ et de la norme $F(\Theta)$ du gradient pour un pixel P donné sont définis par :

$$\left\{ \begin{array}{rcl} \Theta & = & \frac{1}{2}\arctan(\frac{2g_{xy}}{g_{xx}-g_{yy}}) \\ F(\Theta) & = & \frac{1}{2}\left\{(g_{xx}+g_{yy})+\cos(2\Theta)(g_{xx}-g_{yy})+2g_{xy}\sin(2\Theta)\right\} \end{array} \right. \tag{3.11}$$

avec

$$\left\{ \begin{array}{rcl} g_{xx} & = & \left|\frac{\partial R}{\partial x}\right|^2+\left|\frac{\partial V}{\partial x}\right|^2+\left|\frac{\partial B}{\partial x}\right|^2 \\ g_{yy} & = & \left|\frac{\partial R}{\partial y}\right|^2+\left|\frac{\partial V}{\partial y}\right|^2+\left|\frac{\partial B}{\partial y}\right|^2 \\ g_{xy} & = & \left|\frac{\partial R}{\partial x}\right|\left|\frac{\partial R}{\partial y}\right|+\left|\frac{\partial V}{\partial x}\right|\left|\frac{\partial V}{\partial y}\right|+\left|\frac{\partial B}{\partial x}\right|\left|\frac{\partial B}{\partial y}\right| \end{array} \right. \tag{3.12}$$

où $\frac{\partial C}{\partial x}$ représente l'approximation du gradient horizontal "classique" sur la composante C en P et $\frac{\partial C}{\partial y}$ représente l'approximation du gradient vertical "classique" sur la composante C en P. Ce gradient est alors utilisé en lieu et place du gradient précédent pour la construction du vecteur de caractéristiques représentant les composantes de l'histogramme local d'orientation des gradients dans une fenêtre englobante donnée.

3.2.3 Texture

A l'instar de la couleur, la texture est une composante clé de la perception humaine, ce qui en fait une caractéristique essentielle à considérer dans les algorithmes de détection ou de reconnaissance. Bien que chacun soit capable de reconnaitre une texture, il est très difficile d'en donner une définition. Contrairement à la couleur, la texture se définit sur l'ensemble d'une région plutôt qu'au niveau d'un point. La texture présente de nombreuses qualités parmi lesquelles on peut citer sa faible sensibilité aux changements d'illumination. Elle peut être décrite en termes de direction, de grain, de contraste et bien d'autres encore. Cette diversité fait de la texture une facette particulièrement intéressante de l'image qui résulte en une multitude de représentations possibles. Nous avons choisi de nous intéresser à quatre d'entre elles : les caractéristiques de texture d'Haralick, les modèles binaires locaux (LBP en anglais pour Local Binary Pattern), les caractéristiques de Tamura et enfin le spectre de texture.

3.2.3.1 Caractéristiques d'Haralick

La construction des caractéristiques d'Haralick [Haralick 1973] se base sur le concept de relation inextricable entre texture et tons (niveaux de gris) dans une image. Les auteurs présentent une procédure d'extraction de propriétés de texture calculées dans le domaine spatial et tenant compte de la nature statistique de la texture. Leur procédure d'extraction se base sur l'hypothèse que toute l'information de texture d'une image est contenue dans la relation spatiale "moyenne" que les tons de gris entretiennent les uns avec les autres. Plus spécifiquement, les auteurs considèrent que l'information de texture est spécifiée de manière adéquate par un ensemble de matrices de dépendances spatiales, dîtes matrice de co-occurrence, entre tons de gris calculées pour plusieurs relations angulaires et plusieurs distances entre tons de gris voisins de l'image. Quatorze caractéristiques de texture sont alors extraites de ces matrices angulaires de dépendances spatiales entre plus proches voisins. Le calcul des différentes matrices s'effectue comme suit : pour une distance d et un angle α donnés, la composante $P(i, j, d, \alpha)$ de la matrice de fréquences $P_{d,\alpha}$ représente la fréquence avec laquelle on retrouve dans l'image un pixel de niveau de gris i et un pixel de niveau de gris j séparés d'une distance d dans la direction α. La figure 3.8 illustre la construction de quatre de ces matrices (pour les directions 0°, 45°, 90° et 135° et pour une distance de 1 sur une image fournie construite à partir de 4 tons de gris).

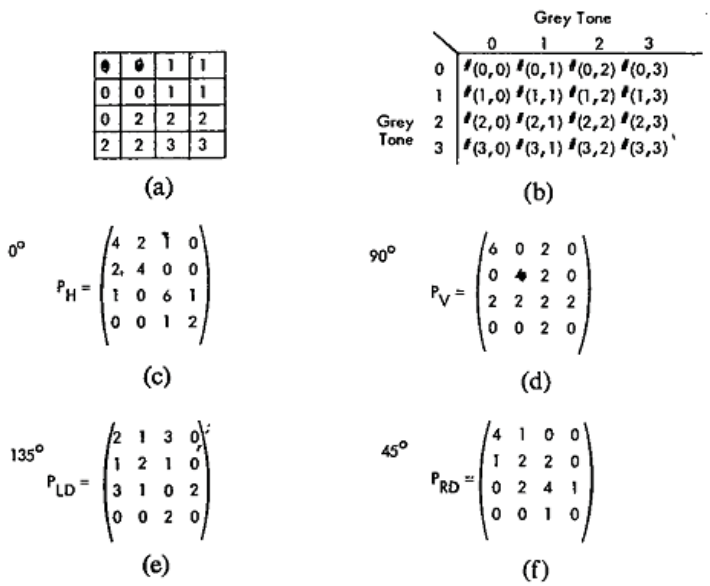


Figure 3.8 – Construction de 4 matrices de fréquences telles que présentées par l'auteur (image extraite de l'article original). (a) Image d'entrée 4×4 à 4 tons de gris (0 à 3). (b) Forme générale de toutes les matrices de dépendance spatiale ($\sharp(i, j)$ représente le nombre de fois où les tons de gris i et j sont considérés comme voisins). (c) à (f) Matrices de dépendance spatiale calculées pour les quatre directions et une distance de 1. Source [Haralick 1973].

Un ensemble de statistiques choisies est ensuite appliqué en parcourant l'ensemble des matrices afin d'obtenir les caractéristiques de texture.
L'inconvénient principal de l'utilisation de ces matrices est la quantité de ressources nécessaires à leur stockage et à leur traitement. En effet, chaque matrice compte autant de lignes et de colonnes que de niveaux de gris utiles dans l'image d'entrée, ce qui requiert un espace de stockage important. De plus, le calcul des différentes caractéristiques nécessite un parcours complet de chaque matrice, résultant en un très grand nombre d'opérations de base. La méthode de stockage développée en 2002 par Clausi et Zhao [Clausi 2002] permet de pallier ces problèmes. Elle se base sur le fait que les matrices utilisées par Haralick sont très souvent éparses et que ces valeurs nulles conduisent à effectuer de nombreux calculs non nécessaires. Elle se base également sur le fait que le stockage unique des valeurs non nulles sous forme de liste chaînée ne s'avère efficace en termes de rapidité de traitement que si cette liste est triée, or ce tri est lui aussi très coûteux. L'idée mise en place par les auteurs est de stocker les valeurs de co-occurrence non nulles (exprimées sous forme de probabilité) au sein d'une structure hybride composée d'une liste doublement chainée stockant réellement les différentes probabilités et permettant l'application très rapide des statistiques choisies et d'une table de hachage stockant, pour une paire de niveaux de gris donnée, un pointeur pointant sur l'élément correspondant à cette paire dans la liste et permettant un accès très rapide à chaque noeud de la liste précédente (cf. Figure 3.9).

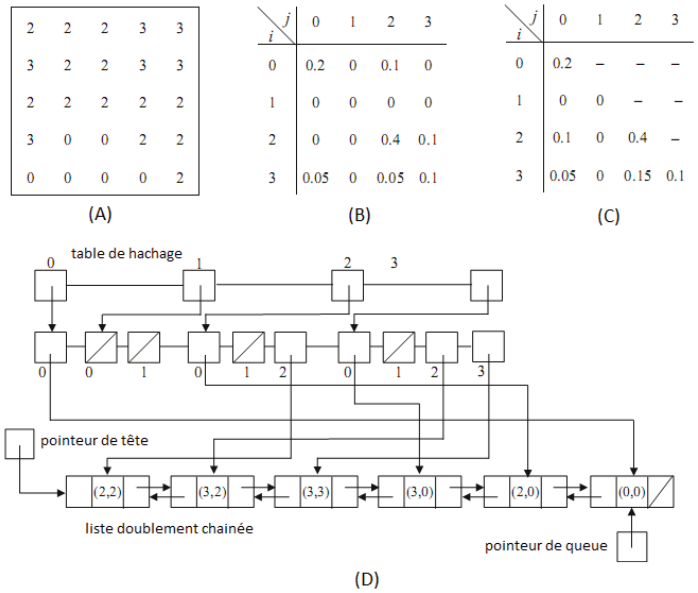


Figure 3.9 – Principe de construction de la structure hybride. (A) Image d'entrée à quatre niveaux de gris (0 à 3). (B) Matrice de co-occurrence correspondante (probabilités) à la manière d'Haralick pour une distance de 1 et une direction $\alpha = 0°$. (C) Matrice de co-occurrence correspondante pour une distance de 1 et la combinaison des deux directions $\alpha = 0°$ et $\alpha = 180°$. (D) Structure hybride correspondante. Source [Clausi 2002].

Caractéristique	Equation
Dissimilarité	$DIS = \sum_{i,j=1}^{G} C_{ij} \lvert i-j \rvert$
Uniformité	$UNI = \sum_{i,j=1}^{G} C_{ij}^2$
Entropie	$ENT = -\sum_{i,j=1}^{G} C_{ij} \log(C_{ij})$
Contraste	$CON = \sum_{i,j=1}^{G} C_{ij}(i-j)^2$
Corrélation	$COR = \sum_{i,j=1}^{G} \frac{(i-\mu_x)(j-\mu_y)C_{ij}}{\sigma_x \sigma_y}$

TABLE 3.1 – Caractéristiques utilisées au sein de l'Ensemble Tracking Modulaire. C_{ij} représente la probabilité de co-occurrence stockée dans la structure hybride pour les niveaux de gris i et j . G représente le nombre de niveaux gris présents dans l'image. Enfin (μ_x, μ_y) et (σ_x, σ_y) représentent respectivement les moyennes et écarts-type pour la ligne i et la colonne j dans la matrice de co-occurrence.

Cette structure permet à la fois de réduire l'espace nécessaire au stockage des informations et de réduire également considérablement les temps de traitement nécessaires au calcul des différentes caractéristiques.
De nombreuses études sur les caractéristiques d'Haralick et notamment une étude réalisée par Baraldi et Parmiggiani [Baraldi 1995] indiquent que seules quelques-unes de ces caractéristiques représentent effectivement des descripteurs de texture efficaces au vu de leur invariance à la translation et au changement d'échelle. La façon de calculer chacune de ces caractéristiques en utilisant la structure hybride décrite ci-dessus est reprise dans la table 3.2.3.1. La dissimilarité et le contraste mesurent le degré de régularité et d'harmonie de la texture. L'uniformité et l'entropie renseignent sur le degré de répétition au sein des paires de niveaux de gris et enfin la corrélation décrit le niveau de corrélation existant entre ces mêmes paires.

3.2.3.2 Local Binary Pattern

Le second type de caractéristiques de texture utilisé dans l'Ensemble Tracking Modulaire est un histogramme de motifs binaires locaux. Les motifs binaires locaux (LBP en anglais pour Local Binary Pattern [Ojala 2002]) permettent d'extraire certains motifs réguliers dans l'image, autrement dit une texture. La texture T d'un voisinage local d'une image en niveaux de gris est définie comme la distribution jointe des niveaux de gris de P ($P > 1$) pixels de l'image :

$$T = t(g_c, g_0, ..., g_{p-1}) \tag{3.13}$$

où g_c correspond au niveau de gris du pixel central du voisinage local et les g_p ($p \in [0, P-1]$) correspondent aux niveaux de gris de P pixels également espacés sur un cercle de rayon R ($R > 0$). En considérant les coordonnées de g_c comme origine

du repère, les coordonnées de chaque g_p sont données par $(-R\sin(\frac{2p\pi}{P}), R\cos(\frac{2p\pi}{P}))$, le niveaux de gris des voisins ne correspondant pas exactement au centre d'un pixel étant estimé par interpolation. La figure 3.10 illustre différents voisinages obtenus pour différentes valeurs du couple (P, R).

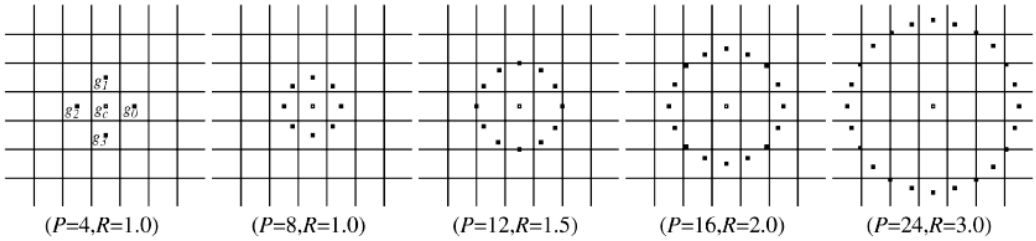


Figure 3.10 – Exemples de voisinages obtenus pour différentes valeurs de (P, R). Source [Ojala 2002].

Partant de ces définitions de texture et de voisinage, les auteurs définissent tout d'abord un motif binaire local invariant à toute transformation monotone de l'échelle des niveaux de gris, $LBP_{P,R}$. Pour chaque pixel (x, y) $(g_c = g(x, y))$:

$$LBP_{P,R}(x, y) = \sum_{p=0}^{P-1} 2^p \delta(g_p - g(x, y)) \tag{3.14}$$

où $\delta(.)$ représente la fonction de Heaviside prenant la valeur 0 en tout réel négatif et la valeur 1 partout ailleurs. Un exemple de calcul de ce motif est donné en figure 3.11.

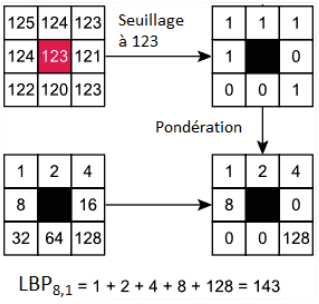


Figure 3.11 – Exemple de calcul de motif $LBP_{8,1}$. Le voisinage est tout d'abord seuillé (fonction de Heaviside) puis pondéré relativement à la position de chaque pixel. Enfin le motif est obtenu via la somme des pondérations utiles.

La définition de ce motif combiné au fait que le voisinage construit soit circulairement symétrique a permis aux auteurs de définir un second motif, lui aussi invariant à toute transformation monotone de l'échelle des niveaux de gris mais également invariant à toute rotation de l'image, $LBP^{ri}_{P,R}$. Pour tout pixel (x, y), le calcul de ce motif est donné par :

$$LBP^{ri}_{P,R}(x,y) = \min\{ROR(LBP_{P,R}(x,y),i)|i \in [\![0,P-1]\!]\} \quad (3.15)$$

où $ROR(a,i)$ correspond au résultat de i décalages circulaires successifs vers la droite des bits du nombre a (codé sur P bits). $LBP^{ri}_{P,R}$ quantifie ainsi les statistiques d'occurrence de modèles individuels invariant en rotation correspondant à certaines micro-caractéristiques de l'image. La figure 3.12 illustre les 36 uniques motifs binaires locaux invariants en rotation possibles dans le cas de $P = 8$. A titre d'exemple, le motif 0 permet de détecter les tâches claires, le motif 8 les tâches sombres et les surfaces monotones et le motif 4 les contours.

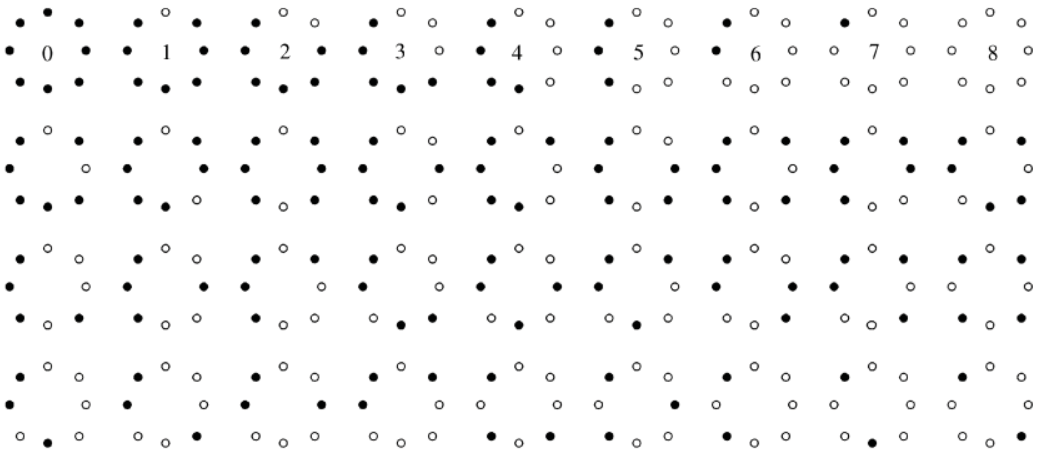


Figure 3.12 – Liste des 36 uniques motifs binaires locaux invariant en rotation possibles pour $LBP^{ri}_{8,R}$. Les points noirs et blancs correspondent respectivement aux valeurs de bit 0 et 1 dans la sortie sur 8 bits de l'opérateur $ROR(.)$. Source [Ojala 2002].

Les histogrammes d'occurrences étant des descripteurs très efficaces pour des problématiques de détection ou de reconnaissance, nous avons naturellement décidé d'utiliser ce type de caractéristiques sous la forme d'un tel histogramme. Pour chaque échantillon utilisé, un histogramme à 8 classes est construit sur une fenêtre englobante de taille 15×15 (chaque composante représente le nombre de pixels de la fenêtre englobante dont le motif LBP correspond à la classe représentée) et le vecteur de caractéristiques de cet échantillon est alors constitué des différentes composantes de cet histogramme.

3.2.3.3 Caractéristiques de Tamura

Tamura et al. ont pris le parti de séparer les caractéristiques de texture correspondant à la perception visuelle humaine. Ils ont ainsi défini six caractéristiques [Tamura 1978] et les ont comparées les unes aux autres par le biais de mesures psychologiques sur des sujets humains. Les résultats obtenus ont permis de mettre en avant deux caractéristiques : la grossièreté et le contraste. Chacune de ces caractéristiques

est calculée, pour un pixel donné, sur une fenêtre englobante (de taille 35×35 dans notre cas).
La grossièreté entretient une relation directe avec l'échelle utilisée et avec les taux de répétition. Cette caractéristique est vue par Tamura et al. comme la caractéristique de texture la plus fondamentale. Elle tente d'identifier l'échelle la plus grande pour laquelle une texture existe. Elle est calculée en plusieurs étapes. Les intensités moyennes sur des voisinages dont les dimensions sont des multiples de deux sont tout d'abord calculées. A titre d'exemple, la moyenne au point (x, y) calculée sur le voisinage de dimension $2^k \times 2^k$ est donnée par

$$M_k(x,y) = \sum_{i=x-2^{k-1}}^{x+2^{k-1}-1} \sum_{j=y-2^{k-1}}^{y+2^{k-1}-1} \frac{I(i,j)}{2^{2k}} \tag{3.16}$$

où $I(x, y)$ représente le niveau de gris du pixel aux coordonnées (x, y). Pour chaque point, les différences entre les paires de moyennes correspondant aux voisinages non-chevauchant sur les côtés opposés du point dans les directions horizontale et verticale sont alors calculées. A titre d'exemple, le cas horizontal est donné par

$$E_{k,h}(x,y) = \left| M_k(x + 2^{k-1}, y) - M_k(x - 2^{k-1}, y) \right| \tag{3.17}$$

Pour chaque point toujours, la taille k_{opt} de k maximisant E soit dans la direction horizontale soit dans la direction verticale est conservée. La mesure de grossièreté du point central de la fenêtre englobante est alors calculée comme la moyenne des valeurs k_{opt} sur tous les pixels de la fenêtre.
Le contraste tente, quant à lui, de capturer l'intervalle dynamique de niveaux de gris conjointement à la polarisation de la distribution des blancs et des noirs dans l'image. L'intervalle est mesuré en utilisant l'écart-type σ des niveaux de gris sur la fenêtre englobante et la polarisation est mesurée en utilisant le kurtosis α_4 calculé sur cette même fenêtre. La mesure de contraste est alors définie par

$$F_{cont} = \frac{\sigma}{\alpha_4^{1/4}} \tag{3.18}$$

Le kurtosis est calculé en divisant le quatrième moment autour de la moyenne par le carré de la variance

$$\alpha_4 = \frac{\mu_4}{\sigma^4} = \frac{\sum_i \sum_j (I(i,j) - \mu)^4}{N\sigma^4} \tag{3.19}$$

où μ représente la moyenne des intensités sur la fenêtre englobante, $I(i, j)$ représente l'intensité du pixel (i, j) et N représente la surface de la fenêtre. Le vecteur de caractéristique de chaque échantillon p est donc composé de la mesure de grossièreté et de la mesure de contraste calculées en p.

3.2.3.4 Spectre de texture

Le but du spectre de texture [He 1991] est l'extraction d'informations de texture locales à partir du voisinage d'un pixel. Le voisinage utilisé est ici de taille 3×3. En notant l'intensité du pixel central V_0 et l'intensité de chaque pixel voisin V_i, l'ensemble considéré comme la plus petite unité complète de l'image considérée est $V = \{V_0, V_1, ..., V_8\}$. L'unité de texture correspondant à cet ensemble est définie par $TU = \{E_1, ..., E_8\}$ avec

$$\forall i \in [\![1, 8]\!], E_i = \begin{cases} 0 & si \quad V_i \leq (V_0 - \Delta) \\ 1 & si \quad (V_0 - \Delta) < V_i \leq (V_0 + \Delta) \\ 2 & si \quad V_i > (V_0 + \Delta) \end{cases} \tag{3.20}$$

En se basant sur l'équation ci-dessus, chaque élément peut se voir assigner une des trois valeurs possibles, ce qui amène le nombre total d'unités de texture possibles pour ces 8 éléments à $3^8 = 6561$. Le numéro d'unité de texture est alors défini par l'équation suivante :

$$NUT = \sum_{i=1}^{8} E_i \times 3^{i-1} \tag{3.21}$$

avec NUT pouvant par conséquent varier entre 0 et 6560.
L'ensemble des 6561 unités de texture correspond aux relations entre niveaux de gris relatifs entre un pixel et ses voisins dans toutes les directions possibles ; c'est à dire à l'aspect de texture local d'un pixel donné relativement à son voisinage. L'idée de base de l'approche par spectre de texture est de transformer une image en utilisant les unités de texture et de caractériser sa texture globale grâce à son spectre de texture. Le spectre de texture peut être défini comme la fonction de fréquence d'occurrence de toutes les unités de texture et peut par conséquent être représenté par un histogramme d'occurrence de ces unités.
Par conséquent, pour chaque échantillon utilisé dans l'Ensemble Tracking Modulaire, un histogramme à N classes (8 dans notre cas) est construit sur une fenêtre englobante (de taille 35×35 dans notre cas). Chaque composante de cet histogramme représente alors le nombre de pixels de la fenêtre pour lesquels le numéro d'unité de texture correspond à la classe représentée. Le vecteur de caractéristiques de cet échantillon est alors constitué des différentes composantes de cet histogramme.

Nous avons présenté dans ce chapitre une nouvelle méthode de suivi d'objet basée sur la méthode Ensemble Tracking et intégrant, par rapport à cette dernière, deux améliorations principales. Nous nous sommes tout d'abord concentrés sur la séparation de l'espace hétérogène des caractéristiques en un ensemble de sous-espaces homogènes (modules) et sur l'application, sur chacun d'eux, d'un algorithme basé Ensemble Tracking.
Dans un second temps, nous nous sommes tournés vers l'apport d'une solution à la nouvelle problématique de suivi induite par cette séparation des espaces, à savoir la construction d'un filtre particulaire spécifique exploitant une pondération des

différents modules utilisés afin d'estimer à la fois, pour chaque image de la séquence, la position et les dimensions de l'objet suivi ainsi que la combinaison linéaire des différentes décisions modulaires conduisant à l'observation la plus discriminante.
Enfin, nous avons présenté l'ensemble des différents modules effectivement implémentés actuellement au sein de la méthode.

De nombreux tests d'exécution de la méthode et de comparaison de cette dernière aux méthodes dîtes classiques ont été effectués et le chapitre suivant présente un ensemble représentatif des différents résultats obtenus.
De plus, l'objectif premier de ces travaux étant leur intégration dans l'environnement industriel de la société Prynεl®, l'ensemble de la méthode présentée a été portée et validée au sein de l'enregistreur numérique de la société. L'intégralité de ce portage est présenté en Annexe A.

Chapitre 4

Expérimentations

Sommaire

Ce chapitre présente différents résultats relatifs au fonctionnement de la méthode développée. Ces résultats permettent d'illustrer le bon fonctionnement de la méthode et de comparer son efficacité avec celle de méthodes dîtes classiques.

4.1 Bases de test et méthodologie

4.1.1 Bases de test

Les séquences vidéos utilisées pour réaliser les différents tests présentés peuvent être regroupées en cinq catégories.
La séquence utilisée afin d'illustrer le bon fonctionnement de la pondération automatique des modules (section 4.2.1) est une séquence simulée présentant le mouvement ellipsoïde d'un rectangle texturé sur différents types de fond.
Les résultats de suivi sur caméra mobile (section 4.2.2) et une partie des résultats présentant l'apport de la sélection aléatoire des pixels (section 4.3.1) ont, quant à eux, été obtenus relativement à une séquence maison enregistrée sur caméra mobile dans les locaux de la société Prynɛl. Celle-ci présente le suivi d'un livre en mouvement dans une scène de bureau classique.
La troisième catégorie, correspondant à la majorité des séquences utilisées dans les différents tests présentés, regroupe un ensemble de séquences extraites de la base de

vidéos CAVIAR [1] (1^{st} et 2^{nd} sets). Ces séquences présentent deux types de scène : le hall d'entrée d'un laboratoire et le couloir principal d'un centre commercial. La base de données fournit également l'ensemble des données de vérité terrain pour chacune des séquences utilisées.
La séquence étudiée afin d'illustrer le bon fonctionnement de la mise à l'échelle sur caméra de centre optique mobile (section 4.2.2) est une séquence du cinquième jeu de données de la collection de vidéos PETS2001 [2], représentant différents véhicules en mouvement sur une autoroute filmés depuis un autre véhicule, lui aussi en mouvement.
Enfin la dernière catégorie, correspondant à la seconde moitié des résultats de comparaison aux méthodes classiques (section 4.4.2), est basée sur une séquence maison créée, utilisée et partagée [3] par S. Stalder et H. Grabner du laboratoire de vision par ordinateur de Zurich dans le cadre de leurs travaux [Stalder 2009] sur le suivi d'objet.

4.1.2 Méthodologie

Comme le laisse supposer la section précédente, trois ensembles d'expérimentations ont été effectués.
Le premier (section 4.2) démontre, en utilisant les résultats de suivi sur une séquence simulée spécifique ainsi que les images de sortie du suivi sur trois séquences (de la base CAVIAR, de la base PETS2001 et de la société Prynɛl), plusieurs propriétés spécifiques de notre algorithme : la pondération automatique des modules utilisés, la mise à l'échelle automatique de la cible sur caméra fixe et sur caméra de centre optique mobile et enfin le fonctionnement du suivi sur caméra mobile.
Le second ensemble (section 4.3) mesure l'influence de chacun des paramètres de la méthode par comparaison de divers paramètres qualitatifs du suivi choisis parmi l'ensemble suivant : la moyenne et l'écart-type de la différence entre l'abscisse du centre du rectangle englobant détecté et l'abscisse du centre issue de la vérité terrain (fournie par la base CAVIAR), la moyenne et l'écart-type de la différence entre l'ordonnée du centre du rectangle englobant détecté et l'ordonnée du centre issue de la vérité terrain, la moyenne et l'écart-type de la distance euclidienne séparant ces deux centres, la moyenne et l'écart-type des différences de largeurs entre le rectangle englobant détecté et le rectangle issu de la vérité terrain, la moyenne et l'écart-type des différences de hauteurs entre le rectangle englobant détecté et le rectangle issu de la vérité terrain, la moyenne et l'écart-type de la distance euclidienne séparant les points dont les coordonnées sont les dimensions de chacun de ces rectangles, le statut de suivi (OK pour un suivi avec succès de bout en bout et KO pour une perte de cible) ou encore les temps de traitement moyens observés par image sur l'ensemble d'une exécution. Concernant ce dernier point, l'intégralité des temps de

1. http ://homepages.inf.ed.ac.uk/rbf/CAVIARDATA1/
2. http ://www.cvg.cs.rdg.ac.uk/PETS2001/pets2001-dataset.html
3. http ://www.vision.ee.ethz.ch/boostingTrackers/videos/

traitement présentés a été obtenue par exécution avec initialisation manuelle de l'Ensemble Tracking Modulaire, implémenté en C++, sur un PC équipé d'un processeur Intel® Core 2 Duo E8500 cadencé à 3.16GHz et de 4Go de RAM DDR2.
Dans le troisième et dernier ensemble d'expérimentations (section 4.4), notre algorithme est comparé à différents algorithmes classiques sur une ou plusieurs séquences de référence : l'Ensemble Tracking Modulaire est comparé à l'Ensemble Tracking de manière numérique sur six séquences de la base CAVIAR, à l'algorithme CAMShift [Bradski 1998] de manière visuelle (en utilisant les images de sortie du suivi) sur une séquence de la base CAVIAR et aux trois algorithmes On-line Boosting [Grabner 2006], Semi-supervised On-line Boosting [Grabner 2008] et Beyond Semi-supervised Tracking [Stalder 2009] de manière visuelle sur une séquence fournie par S. Stalder et H. Grabner.

4.2 Présentation des résultats

4.2.1 Pondération des modules

La figure 4.1 illustre les résultats de suivi de l'algorithme sur une séquence simulée et plus particulièrement l'évolution de la pondération des modules au fil de la séquence. La séquence utilisée présente un rectangle texturé (parsemé de points blancs) de couleur verte se déplaçant le long d'une ellipse et traversant tour à tour trois fonds distincts : un fond uni de couleur violette, un fond uni de couleur verte (identique au vert de l'objet) et enfin un fond texturé de couleur violette (de texture identique à celle de l'objet). Les résultats présentés ont été obtenus en utilisant deux modules : un module de couleur (RVB) et un module de texture (LBP). Les images (a), (b) et (c) correspondent à trois instants clés de la séquence. L'image (d) présente les résultats de suivi à proprement parler sous la forme d'une comparaison entre la trajectoire réelle du centre du rectangle et la trajectoire "détectée" de ce même point. Enfin les courbes de l'image (e) présentent l'évolution de la pondération de chaque module au fil de la séquence.
Durant la première phase de la séquence, l'objet se rapproche de la zone de même couleur. Le nombre de pixels testés appartenant au fond et de même couleur que l'objet ne fait qu'augmenter d'une image à l'autre, diminuant de fait la fiabilité du module de couleur (sans modification de la fiabilité du module de texture). Le filtre, en se basant sur ces fiabilités, diminue donc le poids du module RVB et augmente par conséquent celui du module LBP (somme des poids normalisée) pour atteindre un maximum lors de l'entrée totale du rectangle dans la zone de même couleur (a). Lors du passage de la première zone à la zone de même texture que celle de l'objet mais de couleur différente (seconde zone (b)), la tendance s'inverse. Le module LBP devient non fiable et le module RVB regagne en fiabilité. Ce revirement conduit le filtre à échanger le poids des modules, augmentant le poids du module RVB et diminuant celui du module LBP. Lorsque l'objet s'approche finalement de la bordure entre la seconde zone et le reste de l'image (c), la fiabilité du module LBP augmente de nouveau (sans modification de la fiabilité du module RVB), ce qui

entraine une ré-augmentation progressive de son poids pour atteindre quasiment le même niveau que celui du module de couleur à la fin de la séquence. Cette séquence illustre parfaitement la gestion automatique du poids des modules par le filtre, ce qui permet automatiquement d'éviter que des informations non fiables ne viennent biaiser la prise de décision finale sur la position de l'objet.

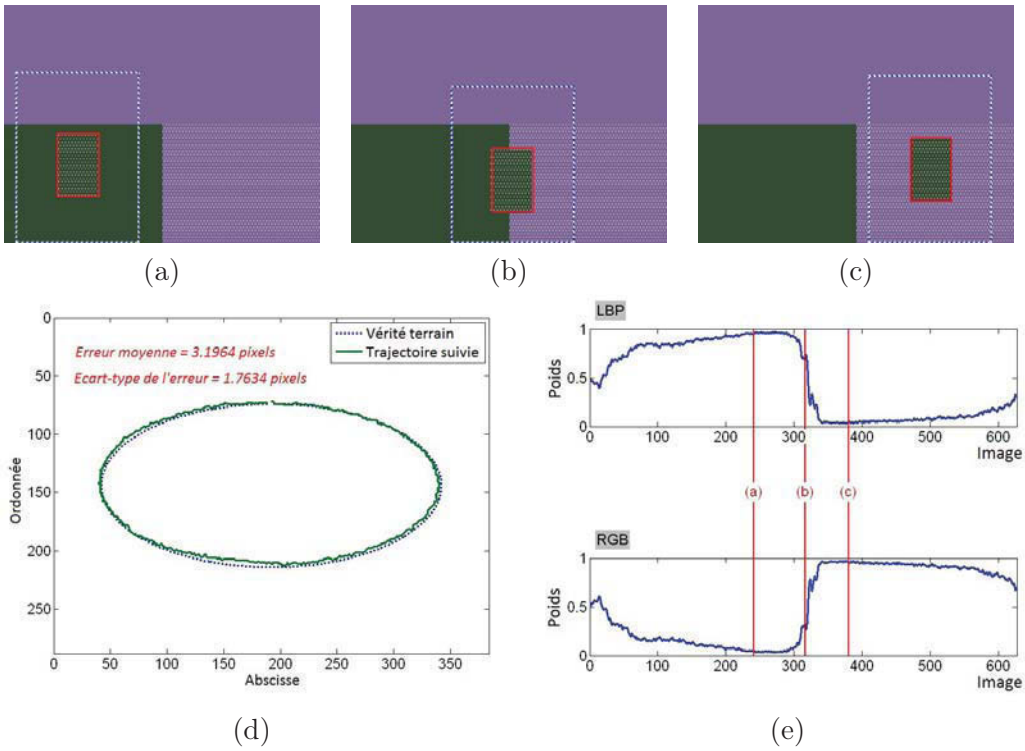


Figure 4.1 – Pondération automatique des modules. (a), (b) et (c) : résultats de suivi correspondants respectivement aux images 240, 316 et 380 de la séquence (le rectangle solide correspond à l'objet et le rectangle en pointillés à la région d'intérêt). (d) : résultat de suivi sur la séquence. La ligne en pointillés représente la trajectoire réelle du centre de l'objet dans le repère image (vérité terrain) et la ligne solide la trajectoire extraite du suivi dans ce même repère. Les paramètres de l'erreur sont calculés relativement à la distance Euclidienne séparant le centre du rectangle réel du centre observé image par image. (e) : évolution de la pondération (moyennée sur l'ensemble des particules) des modules au fil de la séquence (les images a, b et c sont reportées sur les courbes).

4.2.2 Suivi et mise à l'échelle sur caméras fixes et mobiles

La figure 4.2 (resp. 4.3) illustre la mise à l'échelle automatique de la cible lors d'un suivi par Ensemble Tracking Modulaire sur caméra fixe (resp. sur caméra de centre optique mobile).
La première correspond au suivi d'un piéton sur une séquence de la base de vidéos CAVIAR (2^{nd} set). Au fil de la séquence, le piéton suivi s'éloigne de la caméra. La seconde correspond au suivi d'une voiture en mouvement filmée depuis un autre

Figure 4.2 – Mise à l'échelle sur caméra fixe. Le rectangle en trait plein correspond au rectangle englobant la cible alors que le rectangle en pointillés correspond à la région d'intérêt.

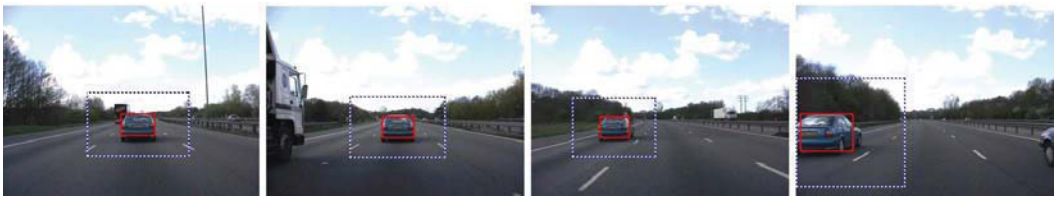


Figure 4.3 – Mise à l'échelle sur caméra de centre optique mobile. Le rectangle en trait plein correspond au rectangle englobant la cible alors que le rectangle en pointillés correspond à la région d'intérêt.

véhicule, lui aussi en mouvement sur une autoroute. Cette séquence est extraite du cinquième jeu de données de la collection de vidéos PETS2001. Tout au long de la séquence, le véhicule équipé de la caméra roule sur la voie centrale de l'autoroute tandis que la voiture suivie passe de la voie centrale, en début de séquence, à la voie de gauche en fin de séquence, tout en se rapprochant de la caméra.
Le vecteur d'état du filtre particulaire inclut les dimensions de l'objet suivi. Cette définition permet au filtre, durant la phase de propagation, d'explorer l'espace des dimensions possibles et ainsi de gérer les changements d'échelle de l'objet suivi en mettant à jour si nécessaire ses dimensions au sein des particules. Ainsi, lorsque le piéton de la première séquence s'éloigne de la caméra (resp. la voiture de la seconde séquence se rapproche de la caméra), la largeur et la hauteur de l'objet diminuent (resp. augmentent) au sein des particules, ce qui a pour effet de rétrécir (resp. d'agrandir) le rectangle englobant moyen (représenté par un trait continu).
Outre la réduction des dimensions du rectangle englobant due à un éloignement de la cible sur la première séquence, on constate également que ce même rectangle n'englobe pas entièrement la personne suivie. Cette "erreur" de détection est due à la manière dont sont définies les particules du filtre et à la manière dont est calculé le score de chacune d'entre elles. L'explication détaillée de ce phénomène est reprise dans l'analyse de la figure 4.8.

Comme illustré précédemment, l'algorithme Ensemble Tracking Modulaire permet de suivre efficacement un objet sur caméra fixe. Cependant, de par sa définition, il

est également capable de suivre sans difficultés un objet sur caméra mobile (caméra en site-azimut-zoom), comme le montre la figure 4.4.

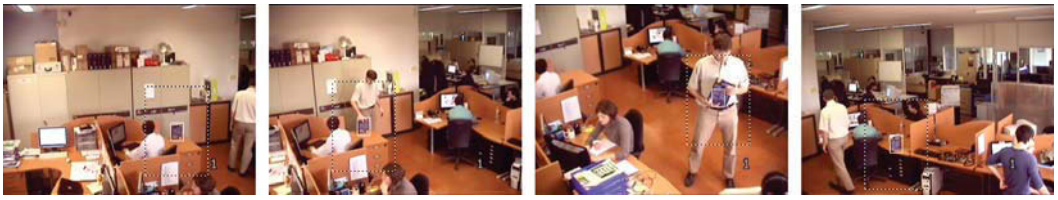


Figure 4.4 – Suivi sur caméra mobile (site-azimut-zoom). Le rectangle en trait plein correspond au rectangle englobant la cible alors que le rectangle en pointillés correspond à la région d'intérêt.

L'Ensemble Tracking Modulaire permet de suivre un objet par (re)détection de ce dernier sur les images successives d'une séquence. L'hypothèse de base de la méthode (induite par la mise en place de l'algorithme d'échantillonnage) est que la position de l'objet varie peu d'une image à l'autre. Il est, par conséquent, tout naturel d'obtenir des résultats de suivi équivalents sur caméras fixes et sur caméras en site-azimut-zoom dont la vitesse de rotation est limitée puisque mouvements de la cible et mouvements de la caméra peuvent alors être confondus.

4.3 Influence des paramètres

4.3.1 Apport de la sélection aléatoire

Les tableaux 4.1 et 4.2 reprennent les temps de traitement mesurés relativement à la variation de deux paramètres (l'activation ou non de l'étape de mise à jour mise en parallèle avec l'activation ou non de l'algorithme d'échantillonnage) lors du suivi d'un objet sur deux séquences vidéos distinctes dont l'initialisation (objet sélectionné) est illustrée respectivement sur les figures 4.5 et 4.6.
La première séquence vidéo, intitulée Grabner2, est une séquence maison fournie par S. Stalder et H. Grabner du laboratoire de vision par ordinateur de Zurich. La séquence présente un motif texturé en mouvement alternant entre visibilité totale, occlusions partielles et occlusions totales dans une scène de bureau classique.
La seconde séquence, intitulée Dictionnaire, est également une séquence maison. Enregistrée sur caméra mobile, dans les locaux de la société Prynɛl, elle présente le suivi d'un dictionnaire de couleur bleue en mouvement dans une autre scène de bureau classique.
Les résultats présentés reflètent l'évolution des temps de traitement moyens durant les étapes d'initialisation et de traitement (avec ou sans mise à jour) lors du passage d'une sélection exhaustive de l'ensemble des pixels de la région d'intérêt à un échantillonnage des pixels traités sur les deux séquences. Dans les huit cas présentés ici, et plus généralement dans 95% des tests effectués, la qualité du suivi (non présentée) est restée équivalente, quelle que soit la méthode de

Mise à jour	Sélection	Temps Initialisation	Temps Traitement
Non	Aléatoire	43.9 ms	16.6 ms
Non	Exhaustive	944.9 ms	139.1 ms
Oui	Aléatoire	40.1 ms	35.4 ms
Oui	Exhaustive	946.1 ms	998.1 ms
Graphiques			

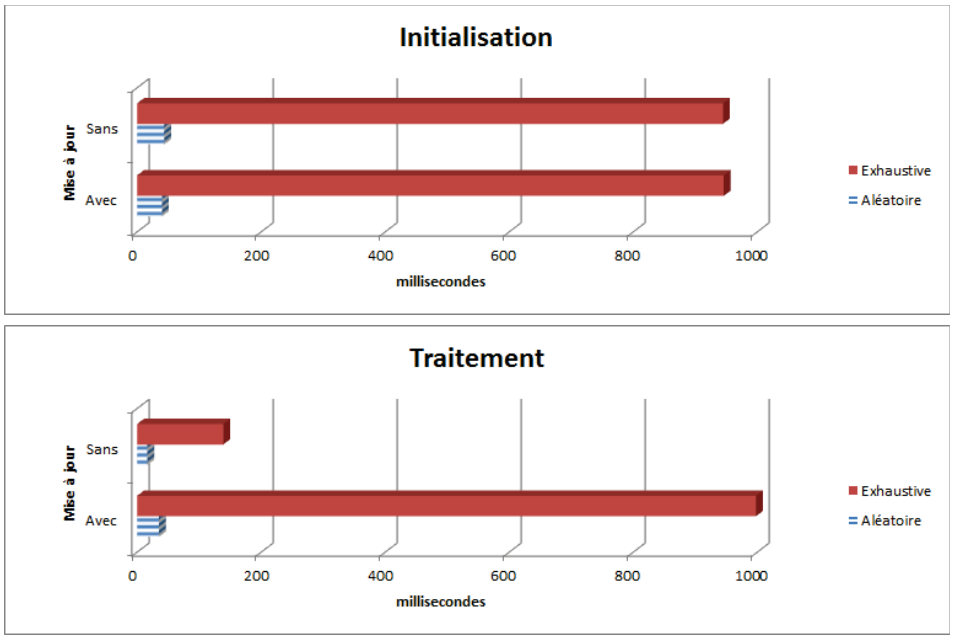


TABLE 4.1 – Temps de traitement mesurés sur la séquence Grabner2. Les temps reflètent l'activation ou non de l'étape de mise à jour (1ère colonne) et l'activation ou non de l'étape d'échantillonnage (2ème colonne). Les temps de traitement présentés correspondent respectivement à l'étape d'initialisation (3ème colonne) sur la première image de la séquence et à la moyenne des temps mesurés sur l'ensemble des autres images de la séquence (4ème colonne). Les deux graphiques reprennent l'ensemble des informations présentées et illustrent, visuellement, les ordres de grandeur obtenus (le remplissage uni rouge correspond à la sélection exhaustive et le remplissage par rayures horizontales bleues à la sélection aléatoire).

Mise à jour	Sélection	Temps Initialisation	Temps Traitement
Non	Aléatoire	46.7 ms	17.2 ms
Non	Exhaustive	950.6 ms	62.8 ms
Oui	Aléatoire	43.5 ms	36.6 ms
Oui	Exhaustive	883.7 ms	866.6 ms
Graphiques			

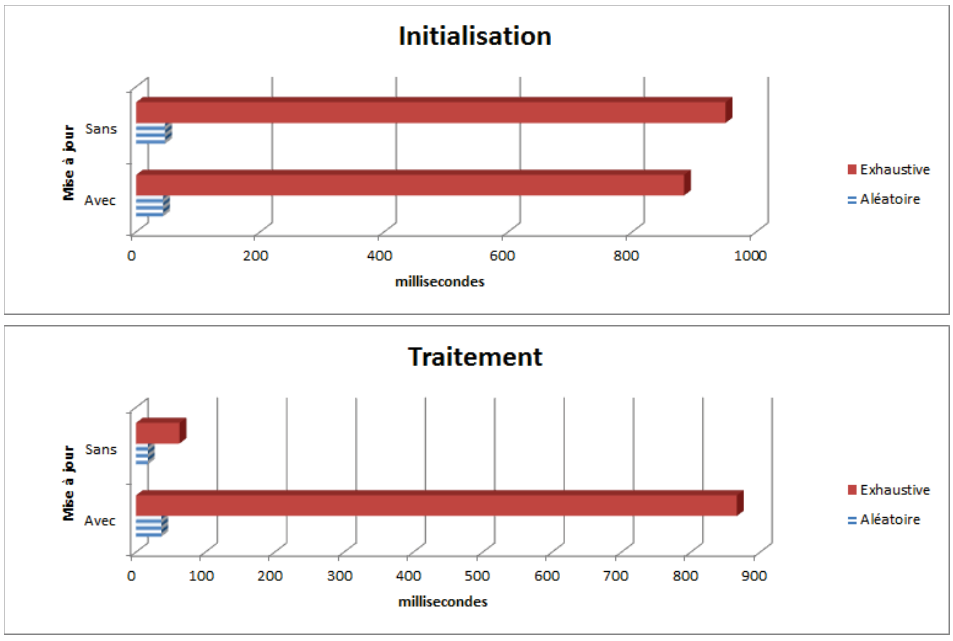


TABLE 4.2 – Temps de traitement mesurés sur la séquence Dictionnaire. Les temps reflètent l'activation ou non de l'étape de mise à jour (1ère colonne) et l'activation ou non de l'étape d'échantillonnage (2ème colonne). Les temps de traitement présentés correspondent respectivement à l'étape d'initialisation (3ème colonne) sur la première image de la séquence et à la moyenne des temps mesurés sur l'ensemble des autres images de la séquence (4ème colonne). Les deux graphiques reprennent l'ensemble des informations présentées et illustrent, visuellement, les ordres de grandeur obtenus (le remplissage uni rouge correspond à la sélection exhaustive et le remplissage par rayures horizontales bleues à la sélection aléatoire).

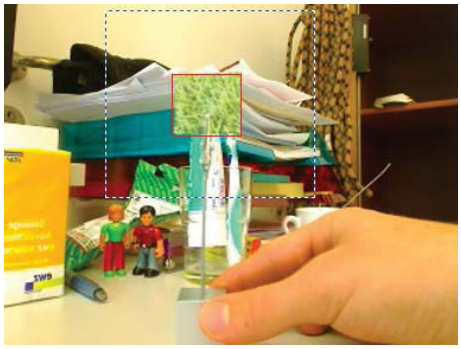


Figure 4.5 – Initialisation de l'objet suivi sur la séquence Grabner2. Le rectangle solide représente l'objet suivi et le rectangle en pointillés représente la région d'intérêt.

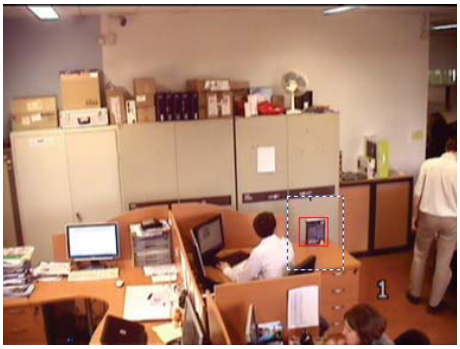


Figure 4.6 – Initialisation de l'objet suivi sur la séquence Dictionnaire. Le rectangle solide représente l'objet suivi et le rectangle en pointillés représente la région d'intérêt.

sélection des pixels choisie. On peut en conclure que l'échantillonnage ne dégrade que très faiblement et très rarement les performances qualitatives de l'algorithme et n'influe, par conséquent, que sur les ressources utilisées. Au regard des deux tableaux présentés ici, on constate très clairement que le temps de traitement nécessaire à l'étape d'initialisation se trouve réduit dans chaque cas de 95% au minimum lors du passage d'une sélection exhaustive à un échantillon de pixels. Cette baisse significative des temps de traitement s'explique naturellement au vu de l'algorithme de construction des classifieurs faibles employé (la construction de ces classifieurs étant l'étape principale de l'initialisation). Chaque classifieur faible est le résultat de plusieurs produits matriciels dans lesquels les dimensions des matrices en jeu sont directement liées au nombre de pixels utilisés. La réduction du nombre d'échantillons réduit donc la taille des matrices et, par conséquent, le temps nécessaire au calcul des différents classifieurs faibles. En ce qui concerne les temps mesurés durant l'étape de traitement, on peut distinguer deux cas. Lorsque la mise à jour des classifieurs forts est désactivée, on constate une réduction des temps de traitement de l'ordre de 80%. Celle-ci s'explique par la manière dont est calculé le score de chaque particule. Ce dernier résulte d'une combinaison entre la confiance moyenne calculée sur l'ensemble des pixels testés à l'intérieur de la région courante et la confiance moyenne calculée sur les pixels testés à l'extérieur. Pour chaque particule, on parcourt donc l'ensemble des pixels utilisés afin de discerner les pixels internes des pixels externes. Une réduction du nombre de pixels utilisés permet donc le réduire le temps de calcul du score de chaque particule et de répercuter ce gain sur l'ensemble des particules. Lorsque la mise à jour est activée, on constate une réduction de l'ordre de 95%. La mise à jour étant définie comme une suppression et un ajout de classifieurs faibles et celle-ci prenant le pas sur l'ensemble des autres traitement, on se retrouve avec un cas équivalent au cas d'initialisation et par conséquent un gain du même ordre.

La figure 4.8 présente l'influence de la taille de la région d'intérêt sur différents paramètres qualitatifs du suivi : les temps de traitement moyens observés, la moyenne et l'écart-type de la différence entre l'abscisse du centre du rectangle englobant détecté et l'abscisse du centre issue de la vérité terrain, la moyenne et l'écart-type de la différence entre l'ordonnée du centre du rectangle englobant détecté et l'ordonnée du centre issue de la vérité terrain et enfin la moyenne et l'écart-type de la distance euclidienne séparant ces deux centres. Tous les résultats présentés correspondent à une moyenne des moyennes calculée sur cinq exécutions successives (cette valeur est jugée représentative au regard de chacun des écart-types calculés, relativement à chaque paramètre mesuré et chaque taille de région d'intérêt étudiée, sur l'ensemble des cinq exécutions (cf. table 4.3)). Les temps de traitement présentés comprennent le temps nécessaire à la génération de données de log (notamment sur la position de sortie). Ce temps nécessaire est considéré comme constant au fil de la séquence et constant d'une exécution à une autre. La séquence vidéo utilisée est une séquence de la base CAVIAR intitulée TwoEnterShop2cor, restreinte pour notre utilisation aux images 740 à 1087 et pour laquelle l'initialisation est effectuée sur la personne présente au premier plan sur la première image (cf. figure 4.7).

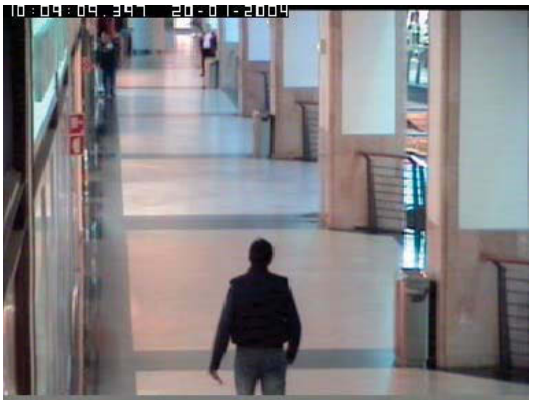


Figure 4.7 – Première image de la restriction de la séquence CAVIAR TwoEnterShop2cor étudiée. L'initialisation de l'algorithme est effectuée sur la personne présente au premier plan (ouvrier en bleu foncé).

Comme le montre le graphique présenté, la taille de la région d'intérêt n'influe de manière significative ni sur les temps de traitement mesurés, ni sur les résultats observés. Ceci s'explique simplement par le rôle de la région d'intérêt au sein de l'algorithme. En effet, celle-ci n'est utilisée que pour restreindre la zone de tirage des pixels à utiliser. A chaque itération, un tirage aléatoire est effectué afin de déterminer l'ensemble des pixels à utiliser. Les paramètres de ce tirage (moyenne et variance) sont déterminés relativement à la dernière position observée de l'objet (position + dimensions). Le tirage est ainsi paramétré de sorte que la moitié (environ) des pixels à utiliser soit tirée à l'intérieur du rectangle englobant l'objet et l'autre moitié à l'extérieur. Pour chaque pixel tiré, une étape supplémentaire vérifie sa position par rapport à la région d'intérêt et le pixel n'est conservé que s'il se

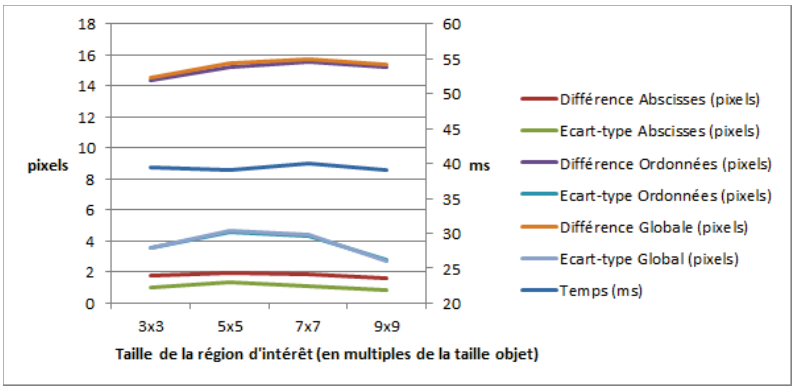


Figure 4.8 – Influence de la taille de la région d'intérêt sur le suivi. Le graphique présente, pour différentes tailles de région d'intérêt (multiples impairs de la taille objet), la moyenne des temps de traitement observés ainsi que la moyenne et l'écart-type mesurés en calculant la différence des abscisses, la différence des ordonnées et la distance euclidienne entre le centre de l'objet détecté et le centre enregistré dans la vérité terrain.

Taille région d'intérêt	Temps de traitement (ms)	Moyenne (pixels)			Ecart-Type (pixels)		
		Abs.	Ord.	Eucl.	Abs.	Ord.	Eucl.
3×3	$0,63$	$0,13$	$0,69$	$0,68$	$0,13$	$0,69$	$0,73$
5×5	$0,27$	$0,18$	$1,33$	$1,35$	$0,20$	$1,03$	$1,07$
7×7	$0,82$	$0,18$	$1,17$	$1,19$	$0,24$	$1,69$	$1,74$
9×9	$0,64$	$0,09$	$0,26$	$0,27$	$0,13$	$0,30$	$0,33$

TABLE 4.3 – Ecart-type de chacun des paramètres qualitatifs du suivi calculé sur les cinq exécutions présentées. Les faibles valeurs présentées ici témoignent de l'aspect représentatif de la moyenne des moyennes calculée sur cinq exécutions successives. Abs. représente la différence entre l'abscisse du centre du rectangle englobant détecté et l'abscisse du centre issue de la vérité terrain, Ord. la différence entre l'ordonnée du centre du rectangle englobant détecté et l'ordonnée du centre issue de la vérité terrain et Eucl. la distance euclidienne séparant ces deux centres.

trouve à l'intérieur de celle-ci. Chaque pixel écarté donne alors lieu à un nouveau tirage. La région d'intérêt permet donc uniquement de limiter la disparité des pixels utilisés en restreignant les différences d'abscisses et d'ordonnées entre le pixel tiré et le rectangle englobant l'objet.
Le graphique révèle cependant une erreur en ordonnée relativement importante sur l'ensemble des jeux de données. Cette erreur, constatée uniquement sur les ordonnées, s'explique par la définition des particules de l'algorithme. Chaque particule tente de représenter l'objet suivi par un rectangle englobant. Cette caractéristique a pour conséquence directe de favoriser les parties de l'objet dont la forme se rapproche le plus d'un rectangle. En effet, pour toute particule donnée, plus la partie de l'objet représentée se rapprochera d'un rectangle et moins la particule sera enclin à contenir des pixels de fond. La particule aura par conséquent un score plus élevé que toute les autres puisque le score particulaire reprend l'ensemble des scores de classification des pixels qui la composent. En appliquant ce constat à notre séquence, on comprend aisément que la partie supérieure (torse) de la personne suivie sera toujours favorisée. L'ordonnée du centre de la position de sortie sera toujours, de ce fait, plus élevée que l'ordonnée du centre du rectangle englobant intégralement la personne suivie.

4.3.2 Influence du nombre de particules

Les figures 4.9 et 4.10 présentent respectivement l'influence du nombre de particules du filtre sur différents paramètres qualitatifs du suivi (moyenne et écart-type de la différence entre l'abscisse du centre du rectangle englobant détecté et l'abscisse du centre issue de la vérité terrain, moyenne et écart-type de la différence entre l'ordonnée du centre du rectangle englobant détecté et l'ordonnée du centre issue de la vérité terrain et enfin moyenne et écart-type de la distance euclidienne séparant ces deux centres) et sur les temps de traitement moyens observés. Tout comme dans la section précédente, tous les résultats présentés correspondent à une moyenne des moyennes calculée sur cinq exécutions successives (valeur jugée représentative pour les mêmes raisons qu'en section précédente (cf. table 4.4)) et les temps de traitement présentés comprennent le temps nécessaire (considéré comme constant d'une image à une autre et d'une exécution à une autre) à la génération de données de log (notamment sur la position de sortie). La séquence vidéo utilisée et les conditions d'initialisation sont les mêmes que dans la section précédente (cf. figure 4.7).
Le graphique 4.9 révèle plusieurs choses. La première est que le nombre de particules affecte les différents paramètres qualitatifs du suivi de la même manière. En effet, il est aisé de remarquer que les courbes ont toutes la même tendance. La seconde est que la qualité globale du suivi semble directement liée au nombre de particules. On constate en effet qu'en règle générale les différentes mesures tendent à diminuer lorsque le nombre de particules augmente (en laissant de côté les mesures liées aux 5000 particules). Ceci s'explique par le fait que plus le nombre de particules est élevé, plus le filtre est à même d'atteindre le régime stationnaire de construction de sa chaîne particulaire de manière précise et d'y rester longuement (en nombre

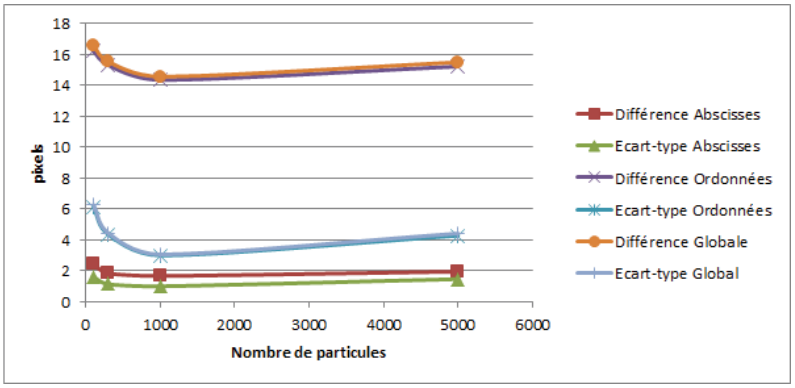


Figure 4.9 – Influence du nombre de particules sur la qualité du suivi. Le graphique présente, pour différents nombres de particules utilisées au sein du filtre, la moyenne et l'écart-type calculés sur la différence des abscisses, la différence des ordonnées et la distance euclidienne entre le centre de l'objet détecté et le centre enregistré dans la vérité terrain. Les résultats sont donnés pour 100, 300, 1000 et 5000 particules.

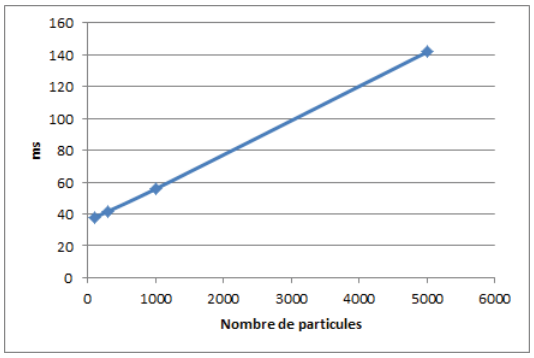


Figure 4.10 – Influence du nombre de particules sur le temps de traitement du suivi. Le graphique présente, pour différents nombres de particules utilisées au sein du filtre, la moyenne des temps de traitement observés sur l'ensemble de la séquence. Les résultats sont donnés pour 100, 300, 1000 et 5000 particules.

Taille région d'intérêt	Temps de traitement (ms)	Moyenne (pixels)			Ecart-Type (pixels)		
		Abs.	Ord.	Eucl.	Abs.	Ord.	Eucl.
3×3	$0,85$	$0,79$	$3,22$	$3,32$	$1,03$	$3,98$	$4,19$
5×5	$0,32$	$0,28$	$1,16$	$1,16$	$0,35$	$1,35$	$1,40$
7×7	$0,32$	$0,19$	$0,36$	$0,31$	$0,16$	$0,50$	$0,56$
9×9	$1,74$	$0,18$	$0,38$	$0,44$	$0,17$	$0,37$	$0,41$

TABLE 4.4 – Ecart-type de chacun des paramètres qualitatifs du suivi calculé sur les cinq exécutions présentées. Les faibles valeurs présentées ici témoignent de l'aspect représentatif de la moyenne des moyennes calculée sur cinq exécutions successives. Abs. représente la différence entre l'abscisse du centre du rectangle englobant détecté et l'abscisse du centre issue de la vérité terrain, Ord. la différence entre l'ordonnée du centre du rectangle englobant détecté et l'ordonnée du centre issue de la vérité terrain et Eucl. la distance euclidienne séparant ces deux centres.

de particules). Ce régime stationnaire plus long contribue alors de manière plus importante dans le calcul de la position de sortie (moyenne des particules), affinant par conséquent les résultats obtenus. Cependant, comme le montrent les résultats obtenus pour 5000 particules, cette précision ne peut augmenter indéfiniment et un nombre trop important de particules (entre 1000 et 5000) force à nouveau le régime oscillatoire du filtre, conduisant à des résultats moins précis.
Cette étude de l'influence du nombre de particules sur la qualité du suivi est bien entendu à mettre en parallèle avec l'étude de cette même influence sur les temps de traitement mesurés. Le graphique 4.10 met en évidence une dépendance linéaire entre le nombre de particules et le temps de traitement moyen par image. Cette dépendance linéaire reflète la manière dont l'algorithme tient compte de ce paramètre. Il n'est en effet utilisé que deux fois au sein de l'algorithme : une fois dans la construction de la chaine MCMC, un accroissement du nombre de particules augmente alors le nombre d'itération dans la boucle de construction de la chaîne et, de ce fait, le temps de traitement global (d'un temps égal au temps de construction d'une particule fois le nombre de particules supplémentaires à construire) et une fois dans la détermination de la position "moyenne" de l'objet avant mise à jour, un accroissement du nombre de particules augmente alors le temps global de calcul de la carte de vraisemblance. Cependant, ce dernier temps reste négligeable par rapport à l'ensemble des autres temps de traitement. Il en résulte donc effectivement une dépendance linéaire entre le temps de traitement global et le nombre de particules dans la chaîne MCMC.
A la vue des résultats présentés sur chacun des deux graphiques précédents, nous avons jugé qu'un bon compromis entre qualité du suivi et temps de traitement nécessaire à y parvenir est d'utiliser, au sein de la chaine MCMC, un nombre de particules égal à 300.

4.3.3 Apport du filtre unique

Comme précisé en chapitre 3, l'algorithme présenté a été au préalable comparé avec l'algorithme pour lequel deux filtres particulaires sont construits et exécutés de manière séquentielle (le vecteur d'état est marginalisé en un sous-vecteur de position/dimensions et un sous-vecteur de pondérations). Avant de présenter les différents résultats de comparaison obtenus, une présentation synthétique de cet algorithme semble nécessaire.
La création de deux filtres distincts modifie le principe général de l'algorithme de la façon suivante : les cartes de confiance construites par chacun des classifieurs forts sont utilisées, conjointement à une distribution de poids calculée sur l'image précédente, par le premier filtre particulaire (appelé filtre à particules de position (FPPos)) pour obtenir la distribution de probabilité de la position objet. La distribution de poids à appliquer aux cartes de confiance des différents modules à l'itération suivante est alors estimée par un second filtre (appelé filtre à particules de poids (FPPds)) relativement à la fonction de densité de probabilité de position

et aux cartes de confiance de l'itération courante. Le schéma synoptique 4.11 présente l'algorithme général de l'Ensemble Tracking Modulaire dans sa version à filtres distincts.

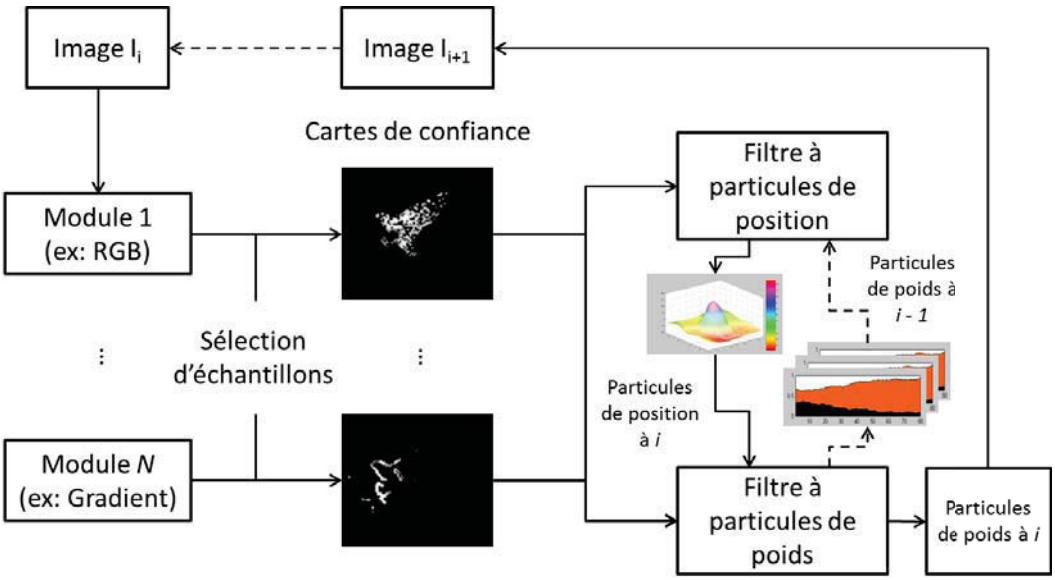


Figure 4.11 – Schéma synoptique de l'algorithme Ensemble Tracking Modulaire dans sa version à filtres distincts.

Le vecteur d'état de chaque filtre (dit marginal en référence à la marginalisation de l'état caché) correspond bien évidemment à une des deux parties (position/dimensions d'un côté et pondérations de l'autre) du vecteur d'état du filtre unique et les différentes contraintes appliquées sur chacune de ces parties au sein du filtre unique restent appliquées dans le filtre marginal correspondant.
En ce qui concerne les modèles de propagation, le modèle du filtre unique résultant une fois de plus de la fusion (par tirage aléatoire) de deux sous-modèles distincts (un correspondant à chaque partie du vecteur d'état), chacun de ses sous-modèles est utilisé comme modèle de propagation du filtre marginal correspondant.
La définition des modèles d'observation est quelque peu plus complexe. Le score de particule du modèle d'observation du filtre unique est calculé sur l'ensemble des cartes de confiance disponibles, pour une position (et des dimensions) unique(s) et un ensemble de poids à appliquer à ces cartes unique. Le filtre de position dispose de l'ensemble des cartes de confiance et, pour chaque particule, d'une position et de dimensions uniques également. Cependant il ne dispose pas d'une distribution des poids unique. Nous avons donc décidé, pour ce filtre, de calculer un score distinct à la manière du modèle défini dans le filtre unique pour chaque particule de pondération construite à l'itération précédente et de considérer le score global de la particule étudiée comme la moyenne de ces derniers. Dans le cas du filtre de pondération, le problème est inverse puisqu'ici les paramètres manquants sont une

position et des dimensions uniques. Nous avons donc décidé d'opter pour la solution inverse en calculant un score distinct pour chaque particule de position construite par le premier filtre, en utilisant la pondération courante, et de moyenner ensuite ces différents scores pour obtenir un score unique.
Afin de pouvoir comparer le principe de chacun des deux algorithmes étudiés ici, l'algorithme global d'Ensemble Tracking Modulaire à double filtres est repris en Algorithme 4.1. Chaque particule de position (respectivement chaque particule de pondération) est notée X_i^k (resp. P_i^k) à l'instant k et les constantes P_1 et P_2 correspondent respectivement au nombre de particules utilisées dans chaque filtre.

Le tableau 4.6 présente les résultats de suivi de l'algorithme Ensemble Tracking Modulaire à filtre unique et les résultats de suivi de l'algorithme à filtres séparés sur six séquences extraites de vidéos de la base CAVIAR dont les caractéristiques sont présentées en table 4.5. Pour chaque séquence et chaque algorithme, le tableau 4.6 présente la moyenne et l'écart-type des distances en abscisses, en ordonnées et globales (euclidiennes) entre le centre du rectangle observé englobant l'objet et le centre du rectangle enregistré dans la vérité terrain (fournie par la base CAVIAR), la moyenne et l'écart-type des différences de largeurs, de hauteurs et globales (distances euclidiennes en considérant les dimensions comme des coordonnées) entre ces mêmes rectangles et pour finir le statut de suivi (OK pour un suivi avec succès de bout en bout et KO pour une perte de cible).
Le premier point révélé par ces données est que la qualité du suivi fournie par l'algorithme à filtre unique est globalement meilleure que celle fournie par l'algorithme à deux filtres. En effet, il est intéressant de noter que sur l'ensemble des séquences étudiées l'algorithme à filtres séparés présente une qualité de suivi moindre (trois pertes de cible sur les trois dernières séquences contre aucune pour l'algorithme à filtre unique) ou équivalente (sur les trois premières séquences, que ce soit au niveau des coordonnées du centre ou des dimensions). La seconde observation qui peut être faite vient en complément de l'analyse effectuée sur la figure 4.8. On constate, premièrement, que la totalité des valeurs moyennes des différences d'ordonnées est relativement haute et, deuxièmement, que l'ensemble des distances basées sur les dimensions de l'objet est très élevé. Tout ceci s'explique par la combinaison de l'analyse faite en section 4.3.1 avec le fait que les "objets" suivis sont à nouveau des personnes. Le torse (partie la plus rectangulaire) de chaque individu étant privilégié, il est alors naturel de constater une différence de largeur significative (les bras sont généralement laissés de côtés), une différence de hauteur significative (même constat pour les jambes) et bien entendu une ordonnée moyenne du centre beaucoup plus haute (pour les même raisons).

4.3.4 Apport de la mise à jour

Le tableau 4.7 présente les résultats de suivi de l'algorithme Ensemble Tracking Modulaire à filtre unique avec et sans étape de mise à jour sur les six séquences définies dans la section précédente et dont les caractéristiques sont présentées en table 4.5.

Données :	$I_1, ..., I_n$	$I_i : p \longrightarrow I(p)$ n images extraites
		$\Delta \subset \mathbb{Z}^2 \longmapsto \mathbb{R}$ d'une séquence vidéo
	$r_1 = (x_1; y_1; w_1; h_1) \subset \Delta$	Rectangle de l'objet dans l'image I_1
Résultats :	$\{r_j\}_{j=2}^{n}$	Rectangles de l'objet dans les images I_j

Initialisation (pour l'image I_1) :

1. Pour chaque module $m(\in [1, M])$:
   (a) Echantillonnage de l'image (cf. section 3.1.1)
   (b) Initialisation du classifieur fort du module m (cf. Algorithme 2.2)
   (c) Estimation des paramètres de sigmoïde A_m et B_m (cf. Equation 3.5)
   (d) Poids initial du module : $w_m = \frac{1}{M}$
2. Initialisation du filtre de position : $X_i^1 = (x_1; y_1; w_1; h_1), \forall i \in [1, P_1]$
3. Initialisation du filtre de pondération : $P_i^1 = (\{w_m\}_{m=1}^{M}), \forall i \in [1, P_2]$

Pour chaque nouvelle image I_j :

1. Pour chaque module $m(\in [1, M])$:
   (a) Echantillonnage de l'image
   (b) Calcul de la carte $C_m^j = (\left\{ c_m^j(x, y) = \frac{1}{1+\exp(A_m.VCU_m^j(x,y)+B_m)} \right\})$
2. Traitement par le filtre de position :
   (a) Tirage et propagation d'une particule X_1^j parmi $\left\{ X_i^{j-1} \right\}_{i=1}^{P_1}$
   (b) Calcul du score $p(Z^t|X_1^j)$ de X_1^j
   (c) Pour $p = 2...P_1 + \delta$
      i. Tirage et propagation d'une particule X_p^j parmi $\left\{ X_i^{j-1} \right\}_{i=1}^{P_1}$
      ii. Calcul du score $p(Z^t|X_p^j)$ de X_p^j
      iii. Acceptation ou rejet de X_p^j suivant l'algorithme de Metropolis-Hastings
   (d) Suppression des δ premières particules (état non stationnaire)
3. $r_j = Rect(\frac{1}{P_1}\sum_i x_i^j; \frac{1}{P_1}\sum_i y_i^j; \frac{1}{P_1}\sum_i w_i^j; \frac{1}{P_1}\sum_i h_i^j)$
4. Traitement par le filtre de pondération :
   (a) Tirage et propagation d'une particule P_1^j parmi $\left\{ P_i^{j-1} \right\}_{i=1}^{P_2}$
   (b) Calcul du score $p(Z^t|P_1^j)$ de P_1^j
   (c) Pour $p = 2...P_2 + \delta$
      i. Tirage et propagation d'une particule P_p^j parmi $\left\{ P_i^{j-1} \right\}_{i=1}^{P_2}$
      ii. Calcul du score $p(Z^t|P_p^j)$ de P_p^j
      iii. Acceptation ou rejet de P_p^j suivant l'algorithme de Metropolis-Hastings
   (d) Suppression des δ premières particules (état non stationnaire)

Algorithme 4.1: Algorithme global d'Ensemble Tracking Modulaire à double filtres

	Séquence CAVIAR de base	Image initiale (Longueur)	Cible (couleur)
1.	Browse4	40 (112)	Homme (bleu)
2.	Fight_OneManDown	194 (198)	Combattant (noir)
3.	TwoEnterShop2cor	0 (410)	Femme (sombre)
4.	TwoEnterShop2cor	91 (441)	Travailleur (bleu foncé)
5.	TwoEnterShop2cor	791 (256)	Travailleur (bleu foncé)
6.	OneStopMoveNoEnter2cor	573 (454)	Homme (rouge)

TABLE 4.5 – Caractéristiques des six séquences vidéos utilisées

Pour chaque séquence et chaque algorithme, le tableau présente les mêmes données que dans la section précédente, à savoir la moyenne et l'écart-type des distances en abscisses, en ordonnées et globales (euclidiennes) entre le centre du rectangle observé englobant l'objet et le centre du rectangle enregistré dans la vérité-terrain (fournie par la base CAVIAR), la moyenne et l'écart-type des différences de largeurs, de hauteurs et globales (distances euclidiennes en considérant les dimensions comme des coordonnées) entre ces mêmes rectangles et pour finir le statut de suivi (OK pour un suivi avec succès de bout en bout et KO pour une perte de cible).
Les différents résultats présentés ici mettent à nouveau en avant plusieurs points. Si l'on ne considère que la partie du tableau correspondant aux coordonnées (colonne 4), on constate que la qualité de l'algorithme avec mise à jour est généralement meilleure que celle de l'algorithme sans. En effet, sur la dernière séquence, le second algorithme a perdu la cible alors que le premier non. Sur les séquences 2, 3, 4 et 5, on remarque que les indicateurs d'erreur fournis sont généralement plus élevés pour le second algorithme que pour le premier (avec parfois de fortes variations comme pour la différence d'abscisses sur la séquence 5). Enfin, sur la séquence 1, on constate que les indicateurs sont très proches pour les deux algorithmes avec cependant un léger avantage pour le second. Ces différences s'expliquent naturellement à partir du moment où l'on considère l'aspect variable de la cible et du fond. Une personne suivie, suivant l'éclairage par exemple, pourra changer de nuance de couleur. De la même manière, suivant les fonds traversés, la manière de distinguer les pixels fonds/forme pourra s'avérer différente. Ces variations seront alors bien entendu gérées automatiquement par l'algorithme avec mise à jour. Par contre, elles ne le seront par l'algorithme sans mise à jour que lorsque le changement s'avèrera minime. Les cinq premières séquences présentées ne contiennent aucun changement important. Ceci permet à l'algorithme sans mise à jour de ne pas perdre la cible (les différences de performances ne s'expliquant alors que par le ré-apprentissage à chaque image par l'algorithme avec mise à jour d'une meilleure manière de distinguer les pixels fond/forme) contrairement à la dernière séquence dans laquelle la cible traverse un fond permettant la confusion avec l'objet initial.
L'analyse des données relatives aux dimensions montre que, même si en général les performances de l'algorithme sans mise à jour restent en deçà de celles de l'algorithme avec mise à jour, il existe certains cas pour lesquels ces performances sont équivalentes voire même meilleures pour l'algorithme sans mise à jour (séquence

	Algorithme	Distances (pixels)		Différences (pixels)		Statut
1.	MET 1 filtre	Abs.	4.03 ± 1.85	Larg.	15.89 ± 5.14	OK
		Ord.	3.63 ± 2.62	Haut.	13.99 ± 6.70	
		Glb.	5.98 ± 2.33	Glb.	21.60 ± 7.45	
	MET 2 filtres	Abs.	3.97 ± 2.80	Larg.	20.43 ± 8.12	OK
		Ord.	3.46 ± 1.89	Haut.	16.26 ± 8.86	
		Glb.	5.82 ± 2.63	Glb.	26.53 ± 11.00	
2.	MET 1 filtre	Abs.	3.32 ± 2.04	Larg.	22.43 ± 14.36	OK
		Ord.	4.21 ± 2.32	Haut.	21.50 ± 9.59	
		Glb.	5.83 ± 2.43	Glb.	32.31 ± 14.83	
	MET 2 filtres	Abs.	4.28 ± 2.68	Larg.	20.96 ± 13.82	OK
		Ord.	5.01 ± 3.36	Haut.	19.54 ± 9.53	
		Glb.	7.18 ± 3.70	Glb.	30.07 ± 14.76	
3.	MET 1 filtre	Abs.	1.45 ± 1.07	Larg.	20.82 ± 6.79	OK
		Ord.	8.58 ± 5.48	Haut.	37.27 ± 17.36	
		Glb.	9.00 ± 5.23	Glb.	44.26 ± 15.27	
	MET 2 filtres	Abs.	1.29 ± 0.92	Larg.	15.93 ± 3.68	OK
		Ord.	11.29 ± 7.36	Haut.	40.76 ± 22.15	
		Glb.	11.68 ± 7.06	Glb.	46.71 ± 17.52	
4.	MET 1 filtre	Abs.	2.60 ± 2.10	Larg.	12.07 ± 7.40	OK
		Ord.	9.32 ± 4.18	Haut.	48.59 ± 16.11	
		Glb.	9.99 ± 4.30	Glb.	51.53 ± 14.29	
	MET 2 filtres	Abs.	27.01 ± 18.72	Larg.	12.63 ± 7.33	KO
		Ord.	116.28 ± 77.95	Haut.	65.32 ± 23.88	
		Glb.	121.24 ± 76.48	Glb.	67.65 ± 22.70	
5.	MET 1 filtre	Abs.	1.39 ± 0.82	Larg.	10.04 ± 2.39	OK
		Ord.	9.30 ± 4.86	Haut.	34.90 ± 8.06	
		Glb.	9.54 ± 4.73	Glb.	36.45 ± 7.95	
	MET 2 filtres	Abs.	2.59 ± 2.16	Larg.	10.87 ± 3.36	KO
		Ord.	13.26 ± 5.01	Haut.	38.73 ± 5.66	
		Glb.	13.80 ± 5.16	Glb.	40.40 ± 5.80	
6.	MET 1 filtre	Abs.	2.68 ± 1.97	Larg.	16.60 ± 7.43	OK
		Ord.	13.10 ± 6.76	Haut.	20.80 ± 7.82	
		Glb.	13.73 ± 6.85	Glb.	27.05 ± 10.53	
	MET 2 filtres	Abs.	2.99 ± 2.73	Larg.	16.63 ± 5.37	KO (fin)
		Ord.	8.35 ± 5.57	Haut.	31.30 ± 6.88	
		Glb.	9.73 ± 5.71	Glb.	36.13 ± 7.39	

TABLE 4.6 – Résultats de suivi comparatifs entre Ensemble Tracking Modulaire à filtre unique et Ensemble Tracking Modulaire à filtres séparés. La colonne 4 correspond aux distances mesurées (moyenne ± écart-type sur l'ensemble de la séquence) respectivement entre l'abscisse du centre du rectangle détecté et celle du centre du rectangle enregistré dans la vérité-terrain (simple différence), entre l'ordonnée de ces deux centres (simple différence) et enfin entre les deux coordonnées de ces deux centres (distance euclidienne). La colonne 6 correspond aux distances mesurées (moyenne ± écart-type sur l'ensemble de la séquence) respectivement entre la largeur du rectangle détecté et celle du rectangle enregistré dans la vérité-terrain (simple différence), entre la hauteur de ces deux rectangles (simple différence) et enfin entre les deux dimensions de ces deux rectangles (distance euclidienne). Enfin la colonne 7 correspond au statut global du suivi sur l'ensemble de la séquence (OK pour un suivi effectif de bout en bout et KO pour une perte de la cible).

	Mise à jour	Distances (pixels)		Différences (pixels)		Statut
1.	Oui	Abs.	4.03 ± 1.85	Larg.	15.89 ± 5.14	OK
		Ord.	3.63 ± 2.62	Haut.	13.99 ± 6.70	
		Glb.	5.98 ± 2.33	Glb.	21.60 ± 7.45	
	Non	Abs.	4.13 ± 2.08	Larg.	16.45 ± 6.16	OK
		Ord.	3.14 ± 1.78	Haut.	13.64 ± 7.16	
		Glb.	5.67 ± 1.97	Glb.	21.85 ± 6.63	
2.	Oui	Abs.	3.32 ± 2.04	Larg.	22.43 ± 14.36	OK
		Ord.	4.21 ± 2.32	Haut.	21.50 ± 9.59	
		Glb.	5.83 ± 2.43	Glb.	32.31 ± 14.83	
	Non	Abs.	4.84 ± 3.57	Larg.	20.29 ± 13.37	OK
		Ord.	4.78 ± 3.70	Haut.	18.84 ± 10.06	
		Glb.	7.38 ± 4.56	Glb.	29.10 ± 14.64	
3.	Oui	Abs.	1.45 ± 1.07	Larg.	20.82 ± 6.79	OK
		Ord.	8.58 ± 5.48	Haut.	37.27 ± 17.36	
		Glb.	9.00 ± 5.23	Glb.	44.26 ± 15.27	
	Non	Abs.	4.37 ± 2.99	Larg.	19.56 ± 8.21	OK
		Ord.	7.37 ± 4.86	Haut.	19.35 ± 11.01	
		Glb.	9.24 ± 5.05	Glb.	30.41 ± 7.64	
4.	Oui	Abs.	2.60 ± 2.10	Larg.	12.07 ± 7.40	OK
		Ord.	9.32 ± 4.18	Haut.	48.59 ± 16.11	
		Glb.	9.99 ± 4.30	Glb.	51.53 ± 14.29	
	Non	Abs.	2.62 ± 2.14	Larg.	10.29 ± 7.03	OK
		Ord.	10.25 ± 4.07	Haut.	53.20 ± 15.68	
		Glb.	10.90 ± 4.12	Glb.	55.61 ± 13.76	
5.	Oui	Abs.	1.39 ± 0.82	Larg.	10.04 ± 2.39	OK
		Ord.	9.30 ± 4.86	Haut.	34.90 ± 8.06	
		Glb.	9.54 ± 4.73	Glb.	36.45 ± 7.95	
	Non	Abs.	6.29 ± 5.48	Larg.	3.99 ± 2.11	OK
		Ord.	10.68 ± 8.00	Haut.	11.10 ± 5.75	
		Glb.	12.87 ± 9.53	Glb.	11.91 ± 6.01	
6.	Oui	Abs.	2.68 ± 1.97	Larg.	16.60 ± 7.43	OK
		Ord.	13.10 ± 6.76	Haut.	20.80 ± 7.82	
		Glb.	13.73 ± 6.85	Glb.	27.05 ± 10.53	
	Non	Abs.	4.26 ± 4.58	Larg.	19.15 ± 6.18	KO (fin)
		Ord.	12.16 ± 5.73	Haut.	27.89 ± 7.69	
		Glb.	14.19 ± 7.09	Glb.	34.64 ± 8.51	

TABLE 4.7 – Résultats de suivi comparatifs entre Ensemble Tracking Modulaire avec et sans mise à jour. La colonne 4 correspond aux distances mesurées (moyenne $\pm$ écart-type sur l'ensemble de la séquence) respectivement entre l'abscisse du centre du rectangle détecté et celle du centre du rectangle enregistré dans la vérité-terrain (simple différence), entre l'ordonnée de ces deux centres (simple différence) et enfin entre les deux coordonnées de ces deux centres (distance euclidienne). La colonne 6 correspond aux distances mesurées (moyenne $\pm$ écart-type sur l'ensemble de la séquence) respectivement entre la largeur du rectangle détecté et celle du rectangle enregistré dans la vérité-terrain (simple différence), entre la hauteur de ces deux rectangles (simple différence) et enfin entre les deux dimensions de ces deux rectangles (distance euclidienne). Enfin la colonne 7 correspond au statut global du suivi sur l'ensemble de la séquence (OK pour un suivi effectif de bout en bout et KO pour une perte de la cible).

	Algorithme	Distance (pixels)	Statut
1.	ETM	5.98 ± 2.33	OK
	ET	5.92 ± 2.97	OK
2.	ETM	5.83 ± 2.43	OK
	ET	10.28 ± 4.59	OK
3.	ETM	9.00 ± 5.23	OK
	ET	34.40 ± 19.47	KO
4.	ETM	9.99 ± 4.30	OK
	ET	92.09 ± 89.42	KO
5.	ETM	9.54 ± 4.73	OK
	ET	29.83 ± 29.27	KO
6.	ETM	13.73 ± 6.85	OK
	ET	16.64 ± 5.28	KO

TABLE 4.8 – Résultats de suivi comparatifs entre Ensemble Tracking Modulaire et Ensemble Tracking. La colonne 3 correspond à la distance euclidienne calculée entre le centre du rectangle englobant détecté et le centre du rectangle enregistré dans la vérité-terrain (moyenne ± écart-type sur l'ensemble de la séquence) et la colonne 4 correspond au statut global du suivi sur l'ensemble de la séquence (OK pour un suivi effectif de bout en bout et KO pour une perte de la cible).

2) alors même que les performances relatives à la position indiquent le contraire. Les deux informations sont donc bien entendu à mettre en parallèle et à pondérer différemment puisque qu'une erreur de position aura bien souvent de plus fâcheuses conséquences pour la suite du suivi qu'une erreur de dimensions.

4.4 Résultats comparatifs

4.4.1 Comparaison à l'Ensemble Tracking

Le tableau 4.8 présente les résultats de suivi des algorithmes Ensemble Tracking et Ensemble Tracking Modulaire à filtre unique sur les six séquences définies dans la section précédente et dont les caractéristiques sont présentées en table 4.5. Pour chaque séquence et chaque algorithme, le tableau présente la moyenne et l'écart-type des distances euclidiennes entre le centre du rectangle observé englobant l'objet et le centre du rectangle enregistré dans la vérité terrain ainsi que le statut de suivi (OK pour un suivi avec succès de bout en bout et KO pour une perte de cible). Pour chaque test, l'implémentation de l'Ensemble Tracking utilisée est celle fournie par Wei Hu dans sa discussion [Hu 2006] autour de l'algorithme de Shaï Avidan. L'algorithme Ensemble Tracking Modulaire a, quant à lui, été utilisé dans des conditions se rapprochant des conditions d'utilisation de l'ET, à savoir avec deux modules actifs, un module de couleurs RVB et un module d'histogramme d'orientation des gradients regroupés en 8 classes et calculé sur une fenêtre englobante de taille 5×5. On constate que l'algorithme Ensemble Tracking perd la cible sur l'ensemble des séquences de la série "Portugal" (série de la base CAVIAR), c'est à dire les séquences

3 à 6. Ces données indiquent que les performances de l'Ensemble Tracking sont toujours inférieures (ou au mieux équivalentes) à celles de l'Ensemble Tracking Modulaire. En effet, l'algorithme Ensemble Tracking compte quatre pertes de cible et, sur la seconde séquence, une moyenne et un écart-type bien plus élevés que ceux de l'ETM. Ces pertes peuvent s'expliquer par l'invariance d'échelle utilisée dans l'ET. Chaque changement d'échelle de l'objet créé une opportunité pour l'Ensemble Tracking de trouver une meilleure correspondance dans le reste de la région d'intérêt. Le même problème peut survenir lorsque l'objet est soumis à de grandes déformations. Il est cependant important de noter, au regard des résultats obtenus sur la première séquence, que l'Ensemble Tracking peut atteindre un niveau de performance du même ordre que celui de ETM lorsque l'environnement est approprié (pas de grande déformation, pas de grand changement d'échelle et aucun objet de caractéristiques similaires à celles de l'objet suivi aux alentours).
L'invariance d'échelle n'est cependant pas l'unique explication possible aux faibles performances de l'ET comparées à l'ETM. Un des principaux paramètres qui différencie les deux algorithmes est leur espace de caractéristiques de travail. L'espace de travail de l'Ensemble Tracking est un espace unique constitué de caractéristiques hétérogènes (couleurs et informations de contours) alors que l'espace de travail de l'Ensemble Tracking Modulaire est formé de deux sous-espaces, chacun constitué uniquement d'un ensemble de caractéristiques homogènes (un sous-espace de couleurs et un sous-espace de contours). Cette particularité de l'ETM permet une prise de décision indépendante sur chacun de ces sous-espaces et, par conséquent, une construction de décision unique de manière plus intelligente, en laissant de côté les informations non pertinentes à chaque instant. La décision de l'ET ne peut, elle, éviter de se trouver biaisée par les caractéristiques non discriminantes à un instant donné.

4.4.2 Comparaison aux méthodes "classiques"

La figure 4.12 illustre la comparaison des résultats de suivi des algorithmes CAMShift et Ensemble Tracking Modulaire. L'algorithme CAMShift, ici implémenté de par la librairie OpenCV [4], est considéré comme un des algorithmes de référence en suivi d'apparence. CAMShift est une forme modifiée de l'algorithme Meanshift (cf. chapitre 1). MeanShift opérant sur des distributions de probabilité de couleurs, CAMShift étend ce concept afin de gérer les changements dynamiques de ces distributions. La représentation en distributions de probabilités choisie par Bradski est la représentation par histogrammes de couleurs. La séquence étudiée correspond à la restriction sur l'intervalle images 1815-2336 de la séquence de la base CAVIAR 'WalkByShop1cor'.
On constate, au regard des résultats présentés que l'algorithme CAMShift fournit de mauvais résultats (images (2) et (3)), contrairement à l'ETM, lorsque la couleur des pixels de fond se rapproche de la couleur du piéton suivi (le modèle d'observation utilisé étant basé sur des histogrammes de couleurs dans l'espace HSV). L'utilisation

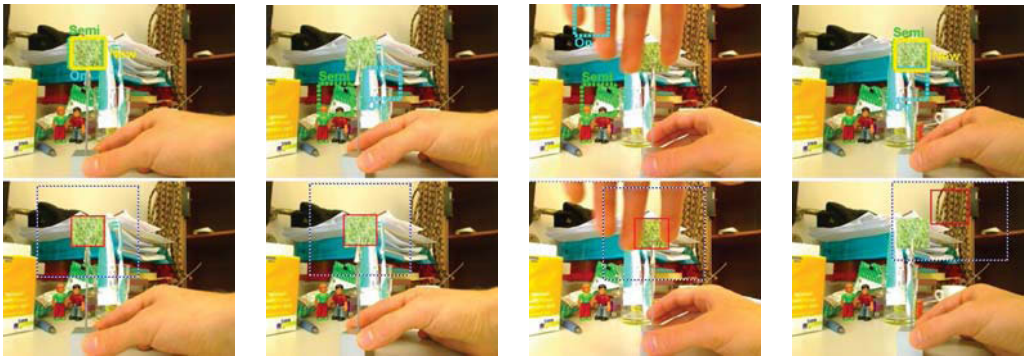
4. http ://sourceforge.net/projects/opencvlibrary/

(1) (2) (3) (4) (5)

Figure 4.12 – Comparaison des résultats de suivi des algorithmes CAMShift et ETM. La première ligne correspond à l'algorithme CAMShift et la seconde à l'ETM. Les images présentées correspondent respectivement aux images 125, 218, 229, 251 et 311 de la séquence étudiée.

de plusieurs types de caractéristiques différents (ici RVB et LBP) et la pondération de chacun des modules (ensemble de caractéristiques homogène) relativement à ses performances permet à l'algorithme ETM de ne pas souffrir des mêmes limitations que l'algorithme CAMShift.

La figure 4.13 illustre la comparaison des résultats de suivi des algorithmes On-line Boosting, Semi-supervised On-line Boosting, Beyond Semi-supervised Tracking, considérés comme des références du domaine, et Ensemble Tracking Modulaire. Le premier algorithme propose l'utilisation d'une version en ligne de l'algorithme Adaboost permettant d'adapter au fur et à mesure du suivi les caractéristiques des classifieurs. Le second propose une limitation au problème de drifting tout en restant adaptable aux changements d'apparence de la cible. Pour ce faire, les auteurs combinent les idées de l'apprentissage semi-supervisé et du boosting adaptatif en ligne pour la sélection de caractéristiques. Enfin, le troisième algorithme propose une dissociation des tâches de détection, de reconnaissance et de suivi en classifieurs séparés dans un esprit de simplification de chaque tâche de classification. La séquence étudiée est une séquence maison créée et partagée par les auteurs des trois méthodes précédentes. Cette dernière est présentée en section 4.3.1.
Les images (1) correspondent à un moment très proche du début de la séquence. Tous les algorithmes sont initialisés sur l'objet texturé rectangulaire. En (2) (resp. (3)), l'objet est soumis à une déformation (resp. une occlusion partielle). Alors que l'ETM continue de suivre l'objet avec succès, les trois autres algorithmes ont perdu la cible : l'On-line boosting (OB) et le Semi-supervised on-line boosting (SSOB) ont glissé vers un candidat plus convaincant et le Beyond semi-supervised tracking (BSST) a considéré que l'objet n'était plus présent dans l'image (aucun rectangle jaune). La méthode de recherche de l'objet des trois algorithmes OB, SSOB et BSST travaille de manière plus globale que celle de l'ETM. La conséquence majeure est que leur détection d'objet est beaucoup plus restrictive, ce qui peut occasionner des pertes de cible en cas de forte déformation (2) ou d'occlusion importante (3). Cependant,



(1) (2) (3) (4)

Figure 4.13 – Comparaison des résultats de suivi des algorithmes On-line boosting (On), Semi-supervised on-line boosting (Semi), Beyond semi-supervised tracking (New) et ETM. La première ligne correspond aux trois premiers algorithmes (cyan pointillés : On-line boosting, vert pointillés : Semi-supervised on-line boosting, jaune : Beyond semi-supervised tracking) et la seconde à l'ETM (le rectangle solide correspond à l'objet et le rectangle en pointillés à la région d'intérêt). Les images (1), (2) et (3) de la seconde ligne correspondent à une même (première) exécution de l'algorithme ETM alors que l'image (4) de cette même ligne correspond à une seconde exécution indépendante de la première.

cette globalité permet à leur suivi de fluctuer beaucoup moins que l'ETM (lié au filtre particulaire) d'une image à une autre et de pouvoir récupérer la cible après une perte beaucoup plus facilement que l'ETM, pour qui la récupération s'avère soumise à des conditions bien précises (4). Ces images illustrent parfaitement les performances équivalentes de l'ETM et des trois algorithmes présentés.

Conclusion et perspectives

Les travaux de cette thèse s'articulent autour de la problématique de suivi d'objet en temps-réel à 25 images par secondes sur caméra robotisée (site, azimut, zoom). Nous nous sommes, d'une part, intéressés aux différentes méthodes existant dans la littérature associée au domaine. Celle-ci découpe la problématique en deux étapes : la détection et le suivi. Puis, nous nous sommes concentrés sur une méthode particulièrement intéressante : l'Ensemble Tracking, qui considère le suivi comme un problème de classification binaire. Enfin nous avons développé une nouvelle méthode, l'Ensemble Tracking Modulaire, basée sur cette dernière, permettant de pallier ses limitations et de correspondre au mieux aux besoins induits par l'intégration industrielle présentée en fin de mémoire. Cette application directe dans un cadre industriel a été l'un des principaux moteurs de nos travaux, dont nous résumons ici les principales contributions.

Notre première amélioration concerne la base même de l'algorithme puisqu'elle porte sur l'espace des caractéristiques utilisé. Nous avons proposé ici une scission de l'espace hétérogène des caractéristiques introduit par Shaï Avidan en plusieurs sous-espaces homogènes afin d'éviter que des informations dont la pertinence à chaque instant peut ne pas être avérée ne viennent biaiser la décision unique prise par l'Ensemble Tracking. L'espace global est donc divisé en un ensemble de sous-espaces homogènes et un classifieur fort est alors construit sur chacun de ses sous-espaces. Cette amélioration permet, d'une part, de diviser la décision globale en un ensemble de sous-décisions et par conséquent d'en laisser certaines de côté si nécessaire et, d'autre part, de rendre l'algorithme complètement modulaire, ce qui a comme conséquence immédiate de permettre à ce dernier de présenter une très grande simplicité d'enrichissement (de nouveaux modules de caractéristiques peuvent être ajoutés à l'ensemble de manière très simple).
La seconde et principale amélioration concerne la prise de décision finale sur la position de l'objet au regard des différentes sous-décisions construites sur chacun des modules utilisé. L'Ensemble Tracking Modulaire propose la définition d'un filtre particulaire spécifique capable de maintenir au fil du temps un ensemble de particules correspondant à un état caché composé bien évidemment de la position de l'objet suivi mais également de l'ensemble des pondérations à appliquer aux différentes sous-décisions (cartes de confiance construites par les différents classifieurs forts) dans le but d'obtenir l'observation de cet état la plus discriminante. Le filtre particulaire construit est de type MCMC. Son vecteur d'état résulte de la combinaison des deux ensembles de paramètres décrits ci-dessus (position/dimensions de l'objet et pondérations) et ses différents modèles d'exécution spécifiques ont été définis relativement à ce dernier.
Enfin la troisième et dernière contribution de ce mémoire concerne la finalité des travaux proposés et plus particulièrement leur intégration au sein de l'enregistreur numérique de la société. Le but de cette intégration étant d'obtenir un suivi 100%

automatique, nous avons proposé un schéma d'intégration collaboratif alliant l'algorithme proposé à une méthode de détection de mouvement par soustraction de fond lorsque la caméra est fixe. Ceci permet, d'une part, de détacher la phase d'initialisation de son étape de sélection manuelle de la cible et, d'autre part, de conforter ou d'infirmer lorsque cela est possible la décision prise par l'Ensemble Tracking Modulaire (ceci afin de commander la caméra au plus juste).
Les différents tests effectués, et notamment les tests exposés dans ce manuscrit, révèlent un meilleur comportement et une meilleure robustesse que les méthodes dites classiques (Ensemble Tracking, Camshift, On-line Boosting, Semi-supervised On-line Boosting et Beyond Semi-supervised Tracking).

Un certain nombre d'éléments proposés dans ce manuscrit nous semblent devoir être améliorés, ou encore prolongés.
La méthode proposée est, comme le précise les paragraphes précédents, modulaire. Cette particularité permet à l'algorithme d'être très simplement enrichi de nouveaux modules de caractéristiques. Au vu des différents modules actuellement implémentés et de cette simplicité, il semble intéressant d'ajouter au panel des modules existants des modules de caractéristiques couvrant des types informations jusqu'alors non couvertes. On peut citer par exemple un module travaillant sur une information spatio-temporelle, un module travaillant sur le flot optique de chaque pixel ou encore un module construit hors-ligne et non mis à jour permettant d'intégrer une information a priori sur la forme des piétons dans l'image.
D'autre part, toute la réflexion de séparation des espaces de caractéristiques est basée sur le constat que chaque module peut ne pas être pertinent à chaque instant de la séquence. Il serait donc judicieux d'étendre la solution proposée ici en permettant à l'algorithme de modifier dynamiquement son panel de modules utilisés. En se basant sur l'efficacité de chacun des modules (révélée par la pondération contenue dans chaque particule), l'algorithme pourrait en écarter certains et en intégrer de nouveaux, appris sur l'image courante.
Enfin, deux types d'applications directes de ce genre de problématique se sont particulièrement développés ces dernières années : d'une part le suivi d'objet sur non plus une mais un réseau de caméras collaboratives et d'autre part des problématiques de contrôle d'accès basées sur une reconnaissance biométrique non coopérative (reconnaissance de visages, d'iris, ...).
Prolonger ces travaux dans ces deux directions constituent deux très intéressantes perspectives.

Annexes

Annexe A : Application industrielle

Comme annoncé en introduction, le but principal de ces travaux était leur intégration au sein de l'enregistreur numérique Digipryn© de la société TEB. La première étape de l'intégration fut la recherche d'une solution pour remplacer la sélection manuelle jusqu'alors employée afin d'initialiser l'algorithme ETM. En effet, l'une des caractéristiques majeures des différents traitements d'images intégrés au sein de l'enregistreur est leur mode de fonctionnement complètement automatique. Il a donc été nécessaire d'automatiser l'étape d'initialisation elle aussi. La caméra utilisée par l'enregistreur à des fins de suivi ne peut être commandée que par l'enregistreur lui même et uniquement lorsqu'une cible est détectée. Durant toute la durée de pré-initialisation de l'ETM, la caméra considérée peut donc être assimilée à une caméra fixe, ce qui facilite la mise en place d'une étape de soustraction de fond afin de déterminer les objets en mouvement dans la scène et ainsi les cibles potentielles. La méthode de soustraction de fond choisie est une simple différence d'images entre l'image courante et une image de fond initialisé avec l'image initiale et mise à jour à chaque image par ajout d'1% de l'image courante à 99% du fond précédent (comme l'illustre l'équation 1).

$$\begin{cases} \text{Image initiale} & : & I_0 \\ \text{Image courante} & : & I_t \\ \text{Fond initial} & : & F_1 = I_0 \\ \text{Fond courant} & : & \forall (x,y) F_{t+1}(x,y) = 0.99 \times F_t(x,y) + 0.01 \times I_t(x,y) \end{cases} \quad (1)$$

Le choix de ce type de méthode a été dicté naturellement par la condition temps réel qu'impose l'intégration à l'enregistreur. Chaque nouvelle image capturée par la caméra est donc testée relativement au fond courant et la mise à jour du fond continue tant qu'aucun mouvement de taille significative (la taille minimale étant fixée relativement aux contraintes du filtre particulaire à 25 pixels × 25 pixels et la taille maximale faisant partie des paramètres de configuration du Digipryn©) n'est détecté (seuillage de la différence d'images). Dès lors qu'un mouvement de ce type est détecté (en cas de mouvements multiples, seul le mouvement de surface maximale répondant aux critères précédents est conservé), l'algorithme ETM est initialisé en considérant l'image binarisée issue du seuillage de la différence d'image (dans laquelle un unique mouvement est conservé) comme référence pour l'étiquetage des pixels d'apprentissage. Le choix des modules utilisés est pour l'instant fixé au module de couleurs RVB et au module de texture LBP.
Le retour d'informations des caméras de la société TEB n'étant pas automatique, l'unique possibilité de détection d'un mouvement en cours sur une caméra est la

comparaison de l'image courante à l'image précédente. Or, cette information étant nécessaire afin de déterminer si le mouvement de la caméra doit ou non être stoppé, une soustraction de fond par différence image à image a dû être mise en place. De cette nécessité a émergé l'idée relativement simple de la fiabilisation du suivi par comparaison, lorsque cela est possible, avec le mouvement détecté (la soustraction de fond lorsque la caméra n'est pas commandée permet à nouveau une détection primaire du mouvement de la cible). L'initialisation de l'ETM donne donc lieu a une ré-initialisation en parallèle du fond avec l'image courante.
A chaque image capturée suivante, l'ETM indique la position présumée de l'objet suivi à la manière présentée au chapitre 3. Toujours à des fins de fiabilisation, une note est affectée à la position retournée. Cette note est calculée relativement à la confiance affectée par chacun des modules utilisés aux différents pixels étudiés, qu'ils soient à l'intérieur du rectangle considéré ou à l'extérieur ainsi qu'à la pondération moyenne fournie par le filtre à chacun des modules (à la manière du calcul de score de chaque particule). La position retournée par l'ETM, son score calculé, la position suivie précédente ainsi que la détection de mouvement effectuée relativement à l'image précédente sont alors utilisés afin de définir l'action à appliquer parmi un panel de quatre actions possibles : le pilotage de la caméra (envoi d'une commande de mouvement), l'arrêt du mouvement de la caméra, la terminaison de l'ETM et la reprise de la détection de mouvement principale (objet perdu) et la ré-initialisation de l'ETM (position retournée non fiable et mais mouvement détecté). La prise de décision se base tout d'abord sur la notion de fiabilité de l'ETM. Cette fiabilité (binaire) est définie comme la comparaison du score de la position retournée avec un seuil fixé défini par le paramétrage de l'enregistreur. Le second point de décision est la distance euclidienne séparant le centre de la position retournée au centre de la position retenue à l'image précédente (lorsque le mouvement de la caméra n'interfère pas ou peu). Une fois de plus cette notion est réduite à une définition de proximité entre ces deux centres par comparaison de la distance euclidienne avec un seuil fixé à la longueur de la diagonale du rectangle dont les dimensions correspondent à la moyenne des largeurs (resp. hauteurs) minimale et maximale. Le dernier point de décision correspond à cinq sous-points : le mouvement de la caméra (déterminé par présence d'un mouvement global), le nombre de mouvements de cible détectés, la présence d'un mouvement(aggloméré) proche (par comparaison à un seuil fixe) de la dernière position retenue, la présence d'un mouvement proche de la position actuelle retournée par l'ETM et l'égalité d'agglomération de ces deux mouvements. Une fois ces différents points déterminés, la décision finale est prise par application de la grille de décision présentée en annexe B. Cette prise de décision est bien entendu accompagnée, le cas échéant de l'enregistrement de la nouvelle position retenue précédente et chaque décision de ré-initialisation ou de terminaison de l'ETM est soumise à approbation par répétition sur plusieurs images successives afin de tenir compte du caractère aléatoire du filtre particulaire. L'algorithme global général est illustré en figure 14.
Les figures suivantes illustrent le bon fonctionnement de la méthode présentées sur caméra placée en intérieur et en extérieur.

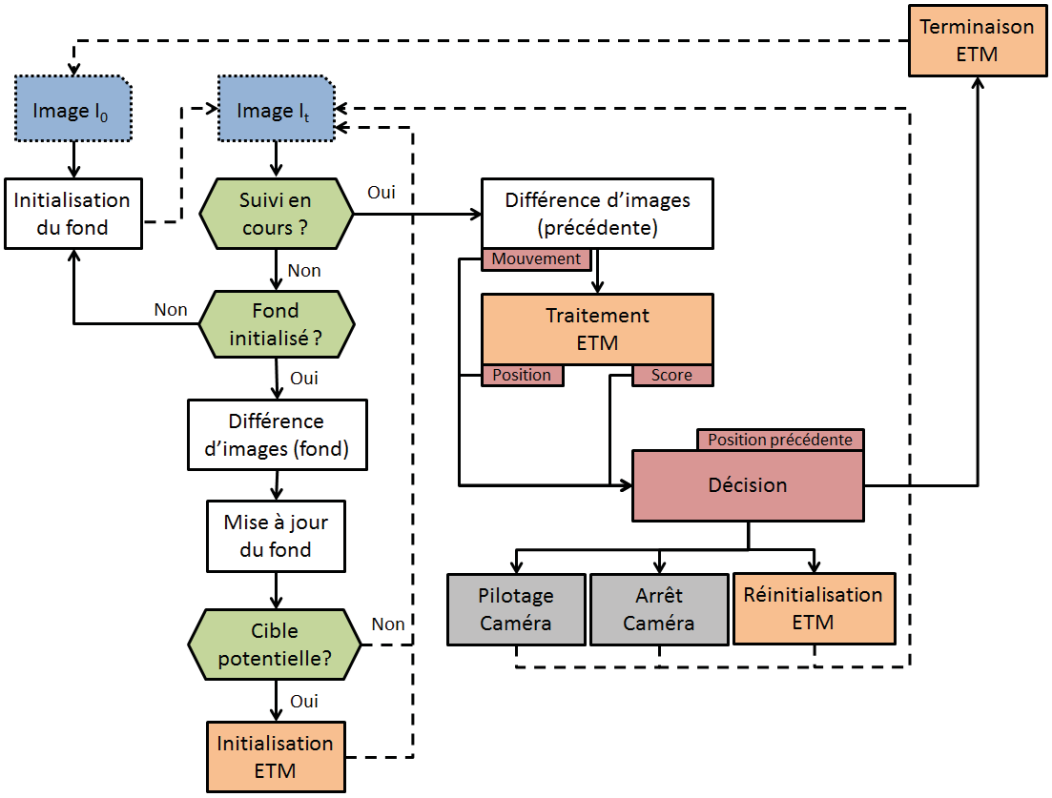


Figure 14 – Algorithme général de l'intégration d'ETM à Digipryn©.

Figure 15 – Illustration des résultats de suivi au sein de Digipryn© sur caméra intérieure. Les images présentées correspondent, de haut en bas et de gauche à droite, aux images 177, 237, 600, 653, 780, 805, 880, 1004 et 1132 de la séquence.

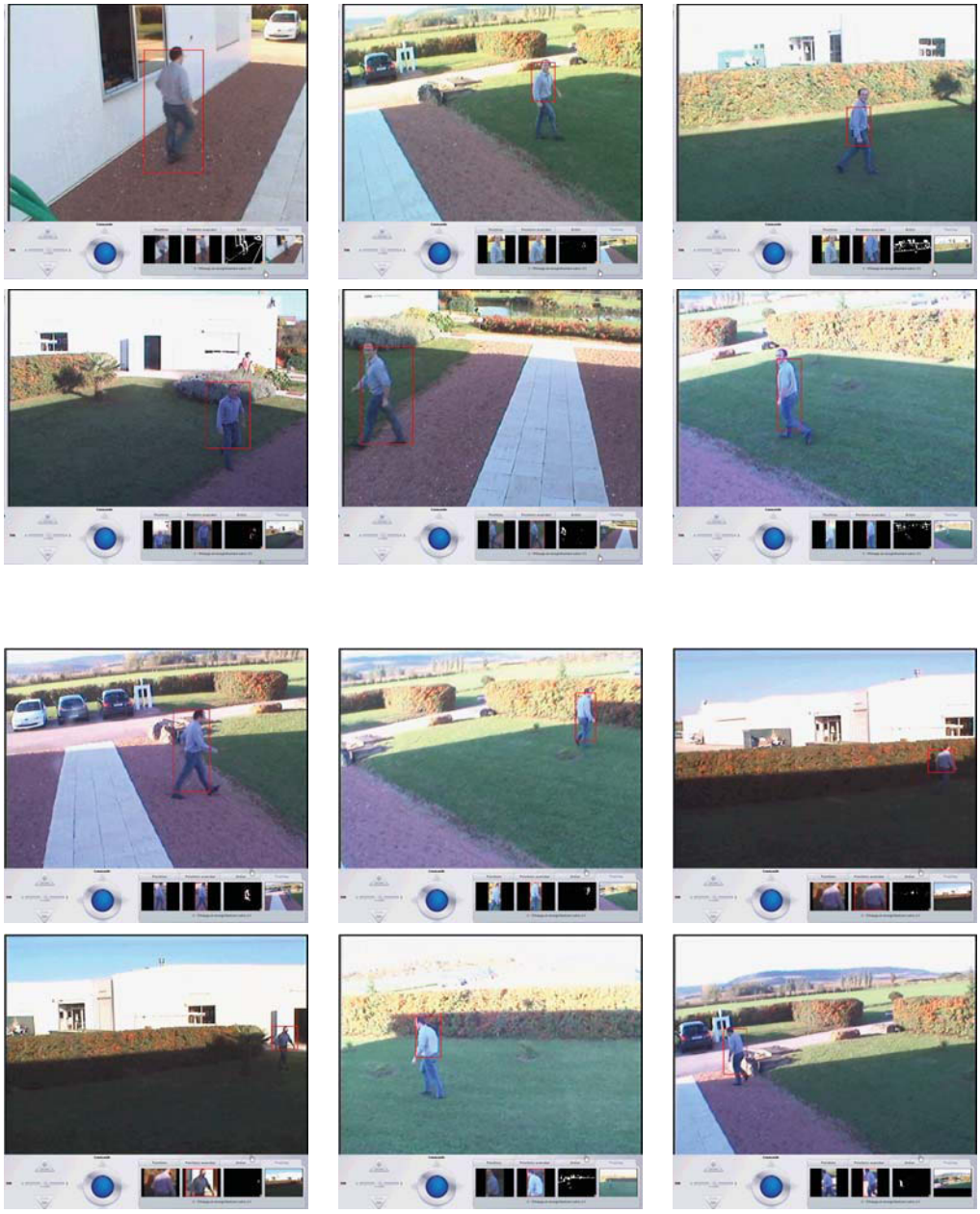


Figure 16 – Illustration des résultats de suivi au sein de Digipryn© sur caméra extérieure. Les images présentées correspondent, de haut en bas et de gauche à droite, aux images 164, 340, 551, 662, 810 et 1059 d'une première séquence puis aux images 2962, 3056, 3257, 3341, 3708 et 3827 d'une seconde séquence.

Annexe B : Grille de décision

Mvt. caméra	Fiabilité ETM	Proximité posETM posPréc	# mvts. détectés	Proximité mvt. posPréc	Proximité mvt. posETM	Egalité mvts.	Décision
Oui	Oui	Oui					Pilotage
Oui	Non	Oui					Arrêt
Oui	Oui	Non					Arrêt
Oui	Non	Non					Arrêt
Non	Oui	Oui	0				Pilotage
Non	Non	Oui	0				Terminaison
Non		Non	0				Terminaison
Non	Non	Non	>0	Non			Ré-init
Non	Non	Non	>0	Oui	Non		Ré-init
Non	Non	Non	>0	Oui	Oui	Non	Ré-init
Non	Non	Non	>0	Oui	Oui	Oui	Pilotage
Non	Non	Oui	>0	Non	Non		Ré-init
Non	Non	Oui	>0	Non	Oui		Pilotage
Non	Non	Oui	>0	Oui	Non		Ré-init
Non	Non	Oui	>0	Oui	Oui	Non	Arrêt
Non	Non	Oui	>0	Oui	Oui	Oui	Pilotage
Non	Oui	Non	>0	Non	Non		Ré-init
Non	Oui	Non	>0	Non	Oui		Pilotage
Non	Oui	Non	>0	Oui	Non		Ré-init
Non	Oui	Non	>0	Oui	Oui	Non	Ré-init
Non	Oui	Non	>0	Oui	Oui	Oui	Pilotage
Non	Oui	Oui	>0	Non	Non		Ré-init
Non	Oui	Oui	>0	Non	Oui		Pilotage
Non	Oui	Oui	>0	Oui	Non		Arrêt
Non	Oui	Oui	>0	Oui	Oui		Pilotage

TABLE 9 – Grille de décision appliquée. La grille présente, pour chaque valeur possible de chaque point de décision pris en compte, l'action à appliquer pour l'image courante. posETM correspond à la position retournée par l'ETM, posPrec à la position retenue précédente et mvt(s) signifie mouvement(s). 'Pilotage' correspond au pilotage de la caméra, 'Arrêt' à l'envoi d'une commande d'arrêt, 'Terminaison' à l'arrêt de l'ETM et à la reprise de la détection de mouvement initiale et enfin 'Ré-init' à la ré-initialisation de l'ETM sur la nouvelle cible détectée.

Bibliographie

[Allili 2009] M.S. Allili. *Effective Object Tracking by Matching Object and Background Models Using Active Contours.* In Proceedings of the 16th IEEE International Conference on Image Processing - ICIP 2009, pages 873–876, Novembre 2009.

[Avidan 2004] S. Avidan. *Support Vector Tracking.* IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 26, no. 8, pages 1064–1072, 2004.

[Avidan 2007] S. Avidan. *Ensemble Tracking.* IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 29, no. 2, pages 261–271, 2007.

[Baraldi 1995] A. Baraldi et F. Parmiggiani. *An Investigation of the Textural Characteristics Associated with Gray Level Cooccurrence Matrix Statistical Parameters.* IEEE Transactions on Geoscience and Remote Sensing, vol. 33, no. 2, pages 293–304, 1995.

[Bertalmio 2000] M. Bertalmio, G. Sapiro et G. Randall. *Morphing Active Contour.* IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, no. 7, pages 733–737, 2000.

[Black 1998] M. Black et A. Jepson. *EigenTracking : Robust Matching and Tracking of Articulated Objects using a View-Based Representation.* International Journal of Computer Vision, vol. 26, no. 1, pages 63–84, 1998.

[Boser 1992] B. Boser, I.M. Guyon et V. Vapnik. *A Training Algorithm for Optimal Margin.* In ACM Workshop on Conference on Computational Learning Theory - COLT 1992, pages 142–152, 1992.

[Bradski 1998] G.R. Bradski. *Computer Video Face Tracking for Use in a Perceptual User Interface.* Rapport technique, Intel Technology Journal, 1998.

[Breiman 1996] L. Breiman. *Bagging Predictors.* Machine Learning, vol. 24, no. 2, pages 123–140, 1996.

[Broida 1986] T. Broida et R. Chellappa. *Estimation of Object Motion Parameters from Noisy Images.* IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 8, no. 1, pages 90–99, 1986.

[Carpenter 1999] J. Carpenter, P. Clifford et P. Fearnhead. *Improved Particle Filter for Nonlinear Problems.* In IEEE Proceedings on Radar and Sonar Navigation, volume 146, pages 2–7, Février 1999.

[Chateau 2006] T. Chateau, V. Gay-Belille, F. Chausse et J.T. Lapresté. *Real-Time Tracking with Classifiers.* In IEEE Proceedings ECCV 06 Workshop on Dynamic Vision, pages 218–231, 2006.

[Cheng 1995] Y. Cheng. *Mean Shift, Mode Seeking, and Clustering.* IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 17, no. 8, pages 790–799, Août 1995.

[Clausi 2002] D. A. Clausi et Y. Zhao. *Rapid Extraction of Image Texture by Co-Occurrence Using a Hybrid Data Structure.* Computers and Geosciences, vol. 28, no. 6, pages 763–774, Juillet 2002.

[Comaniciu 2002a] D. Comaniciu. *Bayesian Kernel Tracking.* In Proceedings of the Annual Conference of the German Society for Pattern Recognition, pages 438–445, 2002.

[Comaniciu 2002b] D. Comaniciu et P. Meer. *Mean Shift : A Robust Approach Toward Feature Space Analysis.* IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 24, no. 5, pages 603–619, 2002.

[Comaniciu 2003] D. Comaniciu, V. Ramesh et P. Meer. *Kernel-based Object Tracking.* IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 25, no. 5, pages 564–575, 2003.

[Dalal 2005] N. Dalal et B. Triggs. *Histograms of Oriented Gradients for Human Detection.* In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition - CVPR 2005, volume 2, pages 886–893, Juin 2005.

[Doucet 2000] A. Doucet, S. Godsill et C. Andrieu. *On Sequential Monte Carlo Sampling Methods for Bayesian Filtering.* Statistics and Computing, vol. 10, no. 3, pages 197–208, 2000.

[Elgammal 2000] A.M. Elgammal, D. Harwood et L.S. Davis. *Non-parametric model for background subtraction.* In Proceedings of the 6th European Conference on Computer Vision - ECCV 2000, Part II, volume 1843 of *LNCS*, pages 751–767. Springer-Verlag, Juin 2000.

[Freeman 1994] W. T. Freeman et M. Roth. *Orientation Histograms for Hand Gesture Recognition.* In Proceedings of the International Workshop on Automatic Face and Gesture Recognition, pages 296–301, Juin 1994.

[Freund 1995] Y. Freund et R. Schapire. *A Decision-Theoretic Generalization of On-line Learning and an Application to Boosting.* Computational Learning Theory, pages 23–37, 1995.

[Freund 2001] Y. Freund. *An Adaptive Version Of The Boost By Majority Algorithm.* Machine Learning, vol. 43, no. 3, pages 293–318, 2001.

[Gordon 1993] N.J. Gordon, D.J. Salmond et A.F.M. Smith. *Novel Approach to Nonlinear/Non-Gaussian Bayesian State Estimation.* In IEEE Proceedings on Radar and Signal Processing, volume 140, pages 107–113, Avril 1993.

[Grabner 2006] H. Grabner, M. Grabner et H. Bischof. *Real-time Tracking via On-line Boosting.* In Proceedings of the British Machine Vision Conference, BMVC 2006, volume 1, pages 47–56, 2006.

[Grabner 2008] H. Grabner, C. Leistner et H. Bischof. *Semi-supervised On-line Boosting for Robust Tracking.* In Proceedings of the 10th European Conference on Computer Vision, ECCV 2008, volume 1, pages 234–247, 2008.

[Haralick 1973] R.M. Haralick, K. Shanmugam et I. Dinstein. *Textural Features for Image Classification.* IEEE Transaction on Systems, Man, and Cybernetics, vol. 3, no. 6, pages 610–621, Novembre 1973.

[Harris 1988] C. Harris et M. Stephens. *A Combined Corner and Edge Detector*. In Proceedings of the 4th Alvey Vision Conference - 1988, pages 147–151, 1988.

[Hastings 1970] W.K. Hastings. *Monte Carlo Sampling Methods Using Markov Chains and Their Applications*. Biometrika, vol. 57, no. 1, pages 97–109, Avril 1970.

[He 1991] D. He et L. Wang. *Texture Features Based on Texture Spectrum*. Pattern Recognition, vol. 24, no. 5, pages 391–399, 1991.

[Hu 2006] W. Hu. *Implementation and Discussion about Ensemble Tracking Algorithm*. Rapport technique, Stony Brook University. Computer Science Department, 2006.

[Huttenlocher 1993] D.P. Huttenlocher, J.J. Noh et W.J. Rucklidge. *Tracking Non-Rigid Objects in Complex Scenes*. In Proceedings of the 4th International Conference on Computer Vision - ICCV 1993, pages 93–101, Mai 1993.

[Isard 1998] M. Isard et A. Blake. *Condensation - Conditional Density Propagation for Visual Tracking*. International Journal of Computer Vision, vol. 29, no. 1, pages 5–28, 1998.

[Jing 2010] L. Jing et P. Vadakkepat. *Interacting MCMC Particle Filter for Tracking Maneuvering Target*. Digital Signal Processing, vol. 20, pages 561–574, March 2010.

[Kalman 1960] R.E. Kalman. *A New Approach to Linear Filtering and Prediction Problems*. Transactions of the American Society of Mechanical Engineers - Journal of Basic Engineering, vol. 82, no. Series D, pages 35–45, 1960.

[Kang 2004] J. Kang, I. Cohen et G. Medioni. *Object Reacquisition Using Geometric Invariant Appearance Model*. In Proceedings of the 17th International Conference on Pattern Recognition - ICPR 2004, pages 759–762, Août 2004.

[Khan 2005] Z. Khan, T. Balch et F. Dellaert. *MCMC-based Particle Filtering for Tracking a Variable Number of Interacting Targets*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 27, no. 11, pages 1805–1819, 2005.

[Lowe 2004] D.G. Lowe. *Distinctive Image Features from Scale-Invariant Keypoints*. International Journal of Computer Vision, vol. 60, no. 2, pages 91–110, Novembre 2004.

[Lucas 1981] B.D. Lucas et T. Kanade. *An Iterative Image Registration Technique with an Application to Stereo Vision*. In Proceedings of the 7th International Joint Conference on Artificial Intelligence - IJCAI 1981, volume 2, pages 674–679. Morgan Kaufmann Publishers Inc., 1981.

[MacCormick 2000] J. MacCormick et A. Blake. *Probablistic Exclusion and Partitioned Sampling for Multiple Object Tracking*. International Journal of Computer Vision, vol. 39, no. 1, pages 57–71, 2000.

[McCulloch 1943] W.S. McCulloch et W. Pitts. *A Logical Calculus of the Ideas Immanent in Nervous Activity Forms*. Bulletin of Mathematical Biophysics, vol. 9, pages 127–147, 1943.

[Metropolis 1953] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller et E. Teller. *Equation of State Calculations by Fast Computing Machines.* Journal of Chemical Physics, vol. 21, pages 1087–1092, Juin 1953.

[Mikolajczyk 2005] K. Mikolajczyk et C. Schmid. *A Performance Evaluation of Local Descriptors.* IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 27, no. 10, pages 1615–1630, Octobre 2005.

[Moravec 1979] H. Moravec. *Visual Mapping by a Robot Rover.* In Proceedings of the 6th International Joint Conference on Artificial Intelligence - IJCAI 1979, volume 1, pages 598–600. Morgan Kaufmann Publishers Inc., 1979.

[Niculescu-Mizil 2005] A. Niculescu-Mizil et R. Caruana. *Obtaining Calibrated Probabilities from Boosting.* In Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence - UAI 2005. AUAI Press, 2005.

[Ojala 2002] T. Ojala, M. Pietikäinen et T. Mäenpää. *Multiresolution Gray-Scale and Rotation Invariant Texture Classification with Local Binary Patterns.* IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 7, pages 971–987, Juillet 2002.

[Oliver 2000] N.M. Oliver, B. Rosario et A.P. Pentland. *A Bayesian Computer Vision System for Modeling Human Interactions.* IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 22, no. 8, pages 831–843, Août 2000.

[Paragios 2000] N. Paragios et R. Deriche. *Geodesic Active Contours and Level Sets for the Detection and Tracking of Moving Objets.* IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 22, no. 3, pages 266–280, 2000.

[Paragios 2002] N. Paragios et R. Deriche. *Geodesic Active Regions and Level Set Methods for Supervised Texture Segmentation.* International Journal of Computer Vision, vol. 46, no. 3, pages 223–247, 2002.

[Piccardi 2004] M. Piccardi. *Background subtraction techniques : a review.* In Proceedings of the IEEE Conference on Systems, Man and Cybernetics - SMC 2004, volume 4, pages 3099–3104. IEEE Piscataway NJ, Octobre 2004.

[Porikli 2005] F. Porikli. *Integral Histogram : A Fast Way to Extract Histograms in Cartesian Spaces.* In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition - CVPR 2005, volume 1, pages 829–836, Juin 2005.

[Rosales 1999] R. Rosales et S. Sclaroff. *3D Trajectory Recovery for Tracking Multiple Objects and Trajectory Guided Recognition of Actions.* In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition - CVPR 1999, pages 117–123, Juin 1999.

[Salari 1990] V. Salari et I.K. Sethi. *Feature Point Correspondence in the Presence of Occlusion.* IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 12, no. 1, pages 87–91, 1990.

[Sato 2004] K. Sato et J. Aggarwal. *Temporal Spatio-Velocity Transform and its Application to Tracking and Interaction.* Computer Vision Image Understanding, vol. 96, no. 2, pages 100–128, 2004.

[Schapire 1999] R.E. Schapire et Y. Singer. *Improved Boosting Algorithms using Confidence-Rated Predictions.* Machine Learning, vol. 37, no. 3, pages 297–336, 1999.

[Seki 2003] M. Seki, T. Wada, H. Fujiwara et K. Sumi. *Background Subtraction based on Cooccurrence of Image Variations.* In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition - CVPR 2003, volume 2, pages 65–72, Juin 2003.

[Shafique 2003] K. Shafique et M. Shah. *A Non-iterative Greedy Algorithm for Multi-frame Point Correspondence.* In Proceedings of the International Conference on Computer Vision - ICCV 2003, pages 110–115, 2003.

[Shi 1994] J. Shi et C. Tomasi. *Good Features to Track.* In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition - CVPR 1994, pages 593–600, 1994.

[Shi 2000] J. Shi et J. Malik. *Normalized Cuts and Image Segmentation.* IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 22, no. 8, pages 888–905, 2000.

[Sobel 1968] I. Sobel et G. Feldman. *A 3x3 Isotropic Gradient Operator for Image Processing.* In Talk at Stanford Artificial Project, 1968.

[Stalder 2009] S. Stalder, H. Grabner et L. van Gool. *Beyond Semi-Supervised Tracking : Tracking Should Be as Simple as Detection, but not Simpler than Recognition.* In IEEE Proceedings ICCV 09 Workshop on On-line Learning for Computer Vision, OLCV 2009, pages 1409 – 1416, Septembre 2009.

[Stauffer 1999] C. Stauffer et W.E.L. Grimson. *Adaptive background mixture models for real-time tracking.* In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition - CVPR 1999, volume 2, pages 246–252, 1999.

[Tamura 1978] H. Tamura, S. Mori et T. Yamawaki. *Textural Features Corresponding to Visual Perception.* IEEE Transaction on Systems, Man, and Cybernetics, vol. 8, no. 6, pages 460–473, Juin 1978.

[Veenman 2001] C. Veenman, M. Reinders et E. Backer. *Resolving Motion Correspondence for Densely Moving Points.* IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 23, no. 1, pages 54–72, 2001.

[Viola 2001] P. Viola et M. Jones. *Rapid Object Detection Using a Boosted Cascade of Simple Features.* In Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - CVPR 2001, volume 1, pages 511–518. IEEE Computer Society, Avril 2001.

[Wan 2000] E.A. Wan et R. Van der Merwe. *The Unscented Kalman Filter for Nonlinear Estimation.* In Proceedings of the Adaptive Systems for Signal

Processing, Communications and Control Symposium - ASSPCC 2000, pages 153–158, Octobre 2000.

[Wolpert 1992] D.H. Wolpert. *Stacked Generalization.* Neural Networks, vol. 5, pages 241–259, 1992.

[Wren 1997] C. Wren, A. Azarbayejani, T. Darrell et A.P. Pentland. *Pfinder : Real-time tracking of the human body.* IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 19, no. 7, pages 780–785, 1997.

[Wu 1993] Z. Wu et R. Leahy. *An Optimal Graph Theoretic Approach to Data Clustering : Theory and its Applications to Image Segmentation.* IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 11, pages 1101–1113, 1993.

[Xu 2002] N. Xu et N. Ahuja. *Object Contour Tracking using Graph Cuts Based Active Contours.* In Proceedings of the IEEE International Conference on Image Processing - ICIP 2002, pages 277–280, 2002.

[Yilmaz 2004] A. Yilmaz, X. Li et M. Shah. *Contour Based Object Tracking with Occlusion Handling in Video Acquired using Mobile Cameras.* IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 26, no. 11, pages 1531–1536, 2004.

[Yilmaz 2006] A. Yilmaz, O. Javed et M. Shah. *Object Tracking : A Survey.* ACM Computing Surveys, vol. 38, no. 4, pages 1–45, Décembre 2006.

[Zenzo 1986] S. Di Zenzo. *A Note on the Gradient of a Multi-Image.* Computer Vision, Graphics, and Image Processing, vol. 33, no. 1, pages 116–125, 1986.

Table des matières

Résumé / Abstract

Résumé

Ces dernières années se caractérisent par la prolifération des systèmes de vidéo-surveillance et par l'automatisation des traitements que ceux-ci intègrent. Parallèlement, le problème du suivi d'objets est devenu en quelques années un problème récurrent dans de nombreux domaines et notamment en vidéo-surveillance.
Dans le cadre de cette thèse, nous proposons une nouvelle méthode de suivi d'objet, basée sur la méthode Ensemble Tracking et intégrant deux améliorations majeures. La première repose sur une séparation de l'espace hétérogène des caractéristiques en un ensemble de sous-espaces homogènes appelés modules et sur l'application, sur chacun d'eux, d'un algorithme basé Ensemble Tracking.
La seconde adresse, quant à elle, l'apport d'une solution à la nouvelle problématique de suivi induite par cette séparation des espaces, à savoir la construction d'un filtre particulaire spécifique exploitant une pondération des différents modules utilisés afin d'estimer à la fois, pour chaque image de la séquence, la position et les dimensions de l'objet suivi, ainsi que la combinaison linéaire des différentes décisions modulaires conduisant à l'observation la plus discriminante.
Les différents résultats que nous présentons illustrent le bon fonctionnement global et individuel de l'ensemble des propriétés spécifiques de la méthode et permettent de comparer son efficacité à celle de plusieurs algorithmes de suivi de référence. De plus, l'ensemble des travaux a fait l'objet d'un développement industriel sur les consoles de traitement de la société partenaire.
En conclusion de ces travaux, nous présentons les perspectives que laissent entrevoir ces développements originaux, notamment en exploitant les possibilités offertes par la modularité de l'algorithme ou encore en rendant dynamique le choix des modules utilisés en fonction de l'efficacité de chacun dans une situation donnée.

Mots-clés : Suivi d'objet, classification, Adaboost, Ensemble Tracking, espaces de caractéristiques, filtrage particulaire.

Abstract

Recent years have been characterized by the overgrowth of video-surveillance systems and by automation of treatments they integrate. At the same time, object tracking has become, within years, a recurring problem in many domains and particularly in video-surveillance.
In this dissertation, we propose a new object tracking method, based on the Ensemble Tracking method and integrating two main improvements. The first one lies on the separation of the heterogeneous feature space into a set of homogenous sub-spaces called modules and on the application, on each of them, of an Ensemble Tracking-based algorithm.
The second one deals with the new tracking problem induced by this separation by building a specific particle filter. This filter weights each used module in order to estimate, for each frame in the sequence, both position and dimensions of the tracked object and the linear combination of modular decisions leading to the most discriminative observation.
The results we present illustrate the global and individual efficiency of all the specific properties of our method and allow comparing this efficiency with the one of several reference tracking algorithms. Furthermore, all this work has led to an industrial development on the treatment systems of the partner company.
In conclusion of this work, we present the prospects generated by these original developments, more particularly using the possibilities offered by the algorithm modularity or making the modules choice dynamic according to their efficiency in a given situation.

Keywords : object tracking, classification, Adaboost, Ensemble Tracking, feature spaces, particle filtering.