N° d'ordre : 3190

THÈSE

PRÉSENTÉE À

L'UNIVERSITÉ BORDEAUX I

ÉCOLE DOCTORALE DE MATHÉMATIQUES ET D'INFORMATIQUE

Par **Lionel CARMINATI**

POUR OBTENIR LE GRADE DE

DOCTEUR

SPÉCIALITÉ : INFORMATIQUE

Détection et suivi d'objets dans les scènes animées: Application à la vidéo surveillance

---

Soutenue le : 27 Juin 2006

Après avis des rapporteurs :

Franck Luthon	Professeur
Georges Quénot	Chargé de Recherche

Devant la commission d'examen composée de :

Henri Nicolas	Professeur	Président
Franck Luthon	Professeur	Rapporteur
Georges Quénot	Chargé de Recherche HDR ...	Rapporteur
Bernadette Bouchon-Meunier	Directeur de Recherche CNRS	Examinateur
Nozha Boujemaa	Directeur de Recherche INRIA	Examinateur
Jenny Benois-Pineau	Professeur	Examinateur

– 2006 –

à mes grand parents, à mes parents, à mon frère, à Émilie...

Résumé

Qu'on le veuille ou non, force est de constater que la vidéo surveillance a finalement pris le pas et s'est installée dans notre quotidien. Gare, autoroute, magasin, il n'est plus rare d'apercevoir des caméras à des endroits dits stratégiques dans le but de filmer et de surveiller une scène faisant elle-même partie d'un environnement global à surveiller. Bien que pour le moment la vidéo n'ait pas de pouvoir juridique reconnu en France -cf l'article de loi n°95-73 du 21 janvier 1995-, elle dispose toutefois d'un fort pouvoir dissuasif. Au-delà des problèmes d'éthique suscités par ce genre d'approches et notamment sur la vie privée de l'individu, la problématique que nous abordons tout au long de ce travail de thèse consiste à l'étude de l'analyse et à l'interprétation de ces flux importants d'informations visuelles en vue de l'extraction automatique d'un certain nombre d'éléments caractéristiques comme par exemple la présence de mouvement ou la présence de visages (personnes).

Dans un premier temps, nous nous sommes intéressés au problème de la détection des objets en mouvement en modélisant la fonction de densité de probabilité des valeurs de luminance des pixels au fil du temps par des mélanges de lois Gaussiennes. Une nouveauté par rapport à l'état de l'art consiste ici à proposer une méthode de ré-apprentissage temporel et une régularisation spatio-temporelle des résultats de la détection (modélisation markovienne) afin de répondre aux exigences des environnements d'extérieur et d'intérieur ainsi que de proposer des masques de mouvement les plus lisses possibles au sens spatio-temporel.

Ensuite dans des zones de mouvement détectées nous nous intéressons à l'extraction des objets particuliers. Ici des méthodes d'apprentissage et de classification par Support Vector Machines -SVM- sont proposées et combinées avec les approches classiques du traitement de l'image (multi-résolution).

Finalement, tout en restant dans le même cadre mathématique des SVM, nous proposons un suivi temporel des objets en mouvement avec le modèle affine complet à 6 paramètres. Les SVM sont employés non-seulement pour le processus de détection, mais profitant de cette connaissance de l'objet d'intérêt à suivre nous utilisons également les SVM pour le suivi avec le modèle affine complet. Ce dernier point est tout à fait intéressant, d'une part du fait de la cohérence de l'approche et d'autre part de l'originalité du traitement des mouvements complexes.

Remerciements

Je tiens à remercier en tout premier lieu Mme Jenny Benois-Pineau qui a dirigé cette thèse dans la continuité de mon stage de DEA. Tout au long de ces trois années, elle a su orienter mes recherches aux bons moments en me faisant découvrir l'analyse et l'indexation vidéo au travers de son regard de spécialiste en traitement du signal, tout en tirant partie de ma formation d'ingénieur.

Je voudrais chaleureusement remercier Mr Georges Quénot pour avoir accepté d'être un des deux rapporteurs de ce mémoire mais aussi pour notre collaboration, ensemble, à la campagne TRECVideo 2003 à la tâche "Feature Extraction". Merci à Mr Franck Luthon pour avoir également accepter d'être rapporteur de cette thèse.

Je remercie également l'ensemble des membres du jury d'avoir accepté d'être présent et de m'avoir apportés leurs remarques et corrections : merci à Mme Bernadette Bouchon-Meunier, Mme Nohza Boujemaa ainsi qu'à Mr Henri Nicolas avec qui j'ai eu l'honneur de travailler notamment sur l'installation du système de vidéo surveillance du LaBRI.

Merci également à Mr Philippe Joly pour son accueil toujours très chaleureux lors des diverses réunions Argos Technovision ainsi que les différents Groupe de Recherche du GdR ISIS qu'il a dirigé.

Je remercie cordialement toute l'équipe très active de COST292 et notamment Ebroul Izquierdo qui m'a accueilli au sein du Queen Mary dans le département "Multimedia & Vision Research Group" de l'Université de Londres pour un séjour d'un semaine qui fut riche en échange scientifique et en discussion. Merci par ailleurs à Jenny, Marta, Divna, Theresa, Emilio, Nikola pour leur accueil. Un remerciement un peu particulier pour Andrea Cavallaro qui non seulement fut mon premier "chairman" mais à qui je dois également une fière chandelle pour ses conseils photo ...

J'ai eu également le plaisir de collaborer avec d'autres ingénieurs et thésards qui ont toujours été de véritables conseils et pour certains de véritables amis. Merci à Francesca, Petra, Laurent, Nicolas, Abdelaziz, et à tout ceux que j'oublie très certainement mais avec qui j'ai eu l'honneur de partager ces trois années.

Une pensée émue pour tous les étudiants avec qui j'ai partagé une salle, un café, un repas ou une console d'ordinateur ou ceux qui ont du me supporter en tant qu'encadrant ou enseignant : Vérena, Christian, Abdenour, Plamena, Martha. Merci également à tous les étudiants de Master 1 -TER ou PdP-, d'IUT Licence Pro et de MIAGE qui ont également

travaillé chacun sur une partie de ce travail...

Ce travail de thèse n'aurait pu être possible sans le financement accordé pour une bourse Bourse de Docteur Ingénieur -BDI- par le CNRS avec la collaboration de l'entreprise Visualpix SA.

Enfin je tiens à remercier toute l'équipe de Thomson Broadcast & Multimedia à Rennes -anciennement Thalès Broadcast & Multimédia- pour m'avoir en si peu de temps intégré et m'avoir permis de corriger en temps et en heure ce mémoire de thèse. Merci à Arnaud, Yann, Samuel, Marc, Olivier et bien sûr FX...

Table des matières

Table des figures

Introduction

Les systèmes de surveillance vidéo se généralisent et occupent aujourd'hui une place de plus en plus importante dans la vie de tous les jours. Cette progression est telle que pour certains elle en devient préoccupante. Les chiffres le prouvent : à l'heure actuelle, on estime à 25 millions de caméras de surveillance vidéo en opération dans le monde et, à cause des événements que l'on connaît, leur nombre est appelé à augmenter sensiblement. Selon un article de la BBC[1], la Grande-Bretagne, premier pays européen en terme de nombre de caméras, en compterait à elle seule 2,5 millions et on estime qu'un citoyen ordinaire se promenant à Londres serait filmé près de 300 fois par jour. Rien que dans le centre de Londres il serait installé près de 327 caméras à différents endroits plus ou moins stratégiques comme les centres commerciaux, les parkings automobiles, les stations de métro ou de bus.

Avec l'émergence de solutions numériques il apparaît que ce nombre ne fera que croître. Si les caméras analogiques n'offraient que des films d'une qualité plus ou moins parfaite, les caméras numériques actuelles permettent d'obtenir des images d'une netteté surprenante tout en proposant même des solutions d'enregistrement performantes. De plus ces caméras peuvent s'interconnecter les unes entre elles, ce qui n'était pas possible avec le matériel analogique du fait des contraintes physiques. Cette propriété permet ainsi d'accroître le champ de vision d'un système de surveillance automatisé.

Depuis plusieurs années, la vidéo surveillance a connu un développement important, accéléré par les programmes de « Homeland Security » dans les pays anglo-saxons ou de « Sécurité Intérieure » en France. Les opérateurs de vidéo surveillance, seuls à même de déceler une situation critique, sont progressivement submergés par le nombre exponentiel de flux d'informations à traiter. Cette situation induit un risque croissant de non détection d'un évènement important.

La vidéo surveillance intelligente (VSI) consiste alors à filtrer les flux vidéo par un logiciel pour ne retenir que l'information pertinente. Seule cette information est envoyée à l'opérateur de vidéo surveillance pour effectuer la levée de doute et déclencher, le cas échéant, l'activation instantanée des plans d'action prévus. La détection d'intrusion ou de bagages abandonnés, le comptage d'individus ou de véhicules, se prêtent particulièrement à la VSI.

Basées principalement sur l'effet de dissuasion, ces caméras et ces réseaux de caméras

[1] http ://news.bbc.co.uk/1/hi/sci/tech/1789157.stm

n'ont pour but finalement que de sécuriser au sens de rassurer des citoyens. Il faut le reconnaître cette constante augmentation se fait sans véritablement connaître l'influence de ces technologies sur la sécurité des citoyens, notamment sur leur sphère privée. A titre anecdotique signalons également que de nombreuses associations contestataires ont donc vu naturellement le jour sur le thème de la "sous surveillance"[2].

Ceci dit, depuis les premières générations de surveillance vidéo sous forme analogique (CCTV) jusqu'aux systèmes numériques les plus avancés (WIFI CAM), ces systèmes de surveillance vidéo ont beaucoup évolué et de manière rapide. De nouveaux domaines d'application de la surveillance vidéo sont même apparus : surveillance des locaux privés comme une maison par exemple, jusqu'aux lieux publics (aéroports, gares, ...), et même semi-publics (magasins, rues,...). Il est maintenant presque impossible de ne pas voir une caméra.

La vidéo surveillance, ses enjeux, ses outils et ses méthodes ont donc été largement étudiés par la communauté scientifique travaillant sur les méthodes d'analyse vidéo et de manière plus générale par la communauté de la vision par ordinateur. Avec l'appui même de certains gouvernements, des méthodes de traitement de données et d'images sont apparues afin de permettre la manipulation de ces énormes volumes d'informations que sont les vidéos de télésurveillance. Malgré l'aspect éthique de ce type d'application, les systèmes de télésurveillance ont pris place, finalement, dans la vie de la société de tous les jours dans le but avoué de se protéger d'actes anti-sociaux et de réduire, ainsi, la criminalité. Même si les capacités des systèmes de télésurveillance s'accroissent de jour en jour du fait de la montée en puissance des systèmes calculatoires et des réseaux d'informations employés, il reste à résoudre la problématique principale : comment gérer efficacement ces volumes d'informations fournis par des centaines voire des milliers de caméras afin d'en extraire un ensemble de caractéristiques clefs qui permettent la compréhension de phénomènes dits "anormaux" par comparaison à ceux dits "normaux" ?

Bien évidemment le recours à la perception humaine a depuis longtemps fait ses preuves en terme de prise de décision mais l'approche que nous devons adopter doit automatiser ce processus par ordinateur. On visualise très bien un opérateur devant des dizaines d'écrans de surveillance à l'affût d'un événement dit "suspect". Ceci dit les caméras elles-mêmes ont leur limitation puisqu'elles n'ont finalement qu'une vue très restreinte de la scène filmée par rapport à l'environnement global à surveiller. Pour avoir une vision globale de ce même environnement, il sera donc nécessaire d'utiliser un certain nombre de caméras et, par conséquent, l'opérateur devra basculer d'une caméra à l'autre pour le visualiser correctement. Or il s'avère que dans le cas d'une supervision humaine, il a été observé que l'opérateur qui surveille doit être particulièrement entraîné et familiarisé à l'environnement et aux scènes filmés et donc il doit connaître le placement de chacune des caméras. Il s'avère également qu'un opérateur humain ne s'intéresse finalement qu'à la détection et à l'identification d'un certain nombre -limité- d'évènements particuliers comme par exemple la présence d'une foule devant une plate-forme, la présence d'individus dans des espaces protégés ou interdits, ou encore la détection d'un ensemble de comportements atypiques...

[2]http ://wearcam.org/wsd.htm

Tous ces éléments font qu'un système intelligent doit permettre de faciliter la détection automatique d'un ensemble d'évènements prédéfinis et/ou à définir en vue d'une aide à la prise de décision. Il serait hasardeux d'affirmer que ce système peut remplacer la présence humaine car il s'agit ici véritablement de l'accompagner et de faciliter sa prise de décision.

Les machines de surveillance basée vidéo peuvent tout aussi bien s'appliquer à l'étude du trafic routier, d'une banque, d'un centre ville, d'aéroports ou encore à la sécurité de locaux qui ont chacun des contraintes et des exigences différentes. Le large spectre d'applications possibles a donc favorisé l'émergence d'une variété d'approches par la communauté scientifique. Pour preuve on a pu constater l'ouverture de nombreux comités scientifiques spécialement dédiés à la vidéo surveillance, et pour ne citer qu'elle, l'ouverture d'une rubrique spéciale dans le journal IEEE Transactions on "Pattern Analysis and Machine Intelligence". La société IEEE a également sponsorisé de nombreux groupes de travail sur la vidéo surveillance dont le fameux "International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance" [3] .

Une solution automatisée doit nécessairement comprendre un certain nombre de composants effectuant chacun une opération particulière que l'on peut classer suivant le degré de complexité traité. Passant ainsi d'opérations de bas niveau, comme par exemple l'acquisition d'images par la caméra, un système intelligent doit également réaliser des opérations beaucoup plus complexes dites de haut-niveau -suivi d'un objet prédéfini à travers le temps ou encore l'interprétation visuelle par exemple. Les coûts matériels permettent aujourd'hui d'envisager économiquement parlant le déploiement de nombreuses caméras, sous forme de réseau c'est à dire inter-connectées entre elles, afin de créer un système de vidéo surveillance. De nombreuses cartes d'acquisition et d'API -Application Programming Interface- ont permis de finaliser le développement de ces réseaux de surveillance en vue de leur intégration et de nombreux systèmes numériques de surveillance vidéo ont ainsi vu le jour.

Les travaux de cette thèse sont consacrés au développement d'un système intelligent de vidéo surveillance. Partant de cette problématique très concrète présentée au chapitre 1, nous nous intéressons au problème plus général de la détection des objets en mouvement au chapitre 2. Dans cette thèse nous nous sommes intéressés à l'application des méthodes d'apprentissage au problème de la surveillance vidéo. Ainsi le chapitre 2 est consacré aux méthodes d'apprentissage non-supervisé et à la modélisation statistique pour la détection d'objets. L'apprentissage supervisé quant à lui occupe une partie plus importante -chapitre 3- dans cette thèse car il constitue une partie considérable des outils que nous avons proposés pour les méthodes de détection et de suivi des objets d'intérêt dans la vidéo de surveillance par SVM. Le chapitre 4 s'attarde un peu plus sur les méthodes de suivi par apprentissage supervisé et les optimisations possibles pour les méthodes que nous proposons. Les résultats de chacune des problématiques de cette thèse sont directement présentés dans le chapitre correspondant.

Enfin ce travail a pu voir le jour grâce à la collaboration entre le Centre National de Recherche Scientifique -CNRS- et l'entreprise régionale Visualpix SA. Cette collaboration

[3] http ://www.cbsr.ia.ac.cn/conferences/VS-PETS-2005/

s'est manifestée par une bourse de thèse de type Bourse de Doctorat Ingénieur -B.D.I.- financée par les deux parties sus-nommées. Le projet s'inscrivait dans la problématique de l'entreprise à savoir l'étude et le développement d'un système embarquée de vidéo surveillance intelligent. Ce système doit pouvoir s'interfacer directement à un système de surveillance déjà existant et très souvent de moindre coût. Ce projet a ainsi pu se dérouler normalement jusqu'à la fin de la thèse grâce à l'appui du CNRS qui a finalement repris intégralement la participation de l'entreprise.

Chapitre 1

Contexte et formulation du problème de la surveillance vidéo

Dans ce chapitre nous allons définir le problème de la surveillance vidéo et les scénarii avec lesquels nous travaillons. Des éléments de l'état de l'art sont présentés et permettent ainsi de situer notre approche par rapport à l'existant.

1.1 Contexte de surveillance vidéo

1.1.1 Définitions

Dans le cadre de ce travail, l'étude d'un système automatisé implique de *localiser* et d'*identifier* tout ou partie des objets présents dans la scène filmée. Le terme identification se réfère à une mise en correspondance entre des objets *connus et appris* et l'objet à identifier. Si l'homme est naturellement capable de percevoir le mouvement apparent des objets filmés dans la scène, par opposition au mouvement propre de la caméra, d'identifier si cet objet en mouvement est un piéton ou une voiture, et de le suivre au fil du temps, il s'agit ici de doter la machine de toutes ces aptitudes particulières.

Mais avant toute chose il convient de définir les termes d'environnement de surveillance et de scènes. Une scène est une partie de l'environnement filmée par une caméra. L'environnement est quant à lui constitué à la fois de l'ensemble des scènes filmées mais également de l'ensemble de ce qui n'est pas visible par les caméras. L'environnement est une entité globale dans laquelle interviennent des événements où les caméras situées à des endroits clefs sont censées les filmer.

L'ensemble des contributions scientifiques appliquées à la vidéo surveillance assistée par ordinateur propose des méthodes, des outils et/ou des algorithmes que l'on peut distinguer en classes en fonction du problème particulier à résoudre :

- La première classe est de celle de la **détection de mouvement**. Dans le cas d'une caméra fixe, il s'agit de repérer les zones actives au sens du mouvement dans l'image par analyse des variations de luminosité au cours du temps. Dans le cas d'une caméra mobile, il s'agit de détecter les objets animés d'un mouvement différent de celui de la caméra. Nous ne considérons dans notre étude que le cas de la caméra fixe.

  Généralement ce problème de détection est abordé par une approche de classification en deux classes où les zones d'inactivité sont opposées à celles en mouvement. Or une distinction plus subtile pourrait être émise : elle consisterait à résoudre ce problème non pas en deux classes -"objet/fond"- mais plutôt en trois classes -"objet/parasite/fond"-... Ceci dit, quelle que soit l'approche retenue, le problème est généralement formulé comme un problème de classification en terme soit de régions soit de pixels divisé en deux parties : apprentissage et classification.

  Pour que ces variations reflètent effectivement les situations de mouvement certaines hypothèses doivent être respectées :

  - l'illumination globale de la scène ne varie pas ou peu sur l'intervalle de temps considéré -ce qui est une contrainte forte pour les scènes filmées à l'extérieur,
  - la caméra est statique,
  - les changements ponctuels de luminance ne sont dus qu'au déplacement d'éléments de la scène.

- La deuxième classe est celle de la **détection d'un objet d'intérêt** connu et de la définition de son **espace de représentation**.

  Dans cette étude nous nous sommes particulièrement intéressés au problème de la détection de visages humains dans la vidéo. L'approche à retenir dans ce travail devait permettre néanmoins d'apprendre n'importe quelle forme pourvu qu'un espace de représentation soit connu. Cet espace doit être suffisamment discriminant pour pouvoir caractériser cet objet dans l'espace retenu. Les problèmes de détection de manière générale ne peuvent se faire sans avoir, au préalable, retenu un espace de représentation dans lequel définir notre objet d'intérêt : valeur d'intensité de pixels (YUV [CN98], RGB [DCBP05], ...), structure morphologique particulière (blob [BID+00], étoile [FL98], ...), vecteur de traits caractéristiques[OFG97] sont des exemples d'espace à retenir. Il convient alors de choisir l'espace le plus discriminant possible afin de rendre le processus de détection robuste.

  Le problème de détection de visages dans la vidéo est un des plus vieux problèmes de vision par ordinateur. Il s'agit de la première pierre pour beaucoup d'autres traitements sur le visage comme par exemple la reconnaissance de visages, d'expressions, etc. Bien qu'il existe de nombreuses déclinaisons du traitement de détection de visages, nous soulignerons tout de suite la différence entre détection et localisation de visages. Dans le premier cas un visage humain peut être présent ou non à l'image alors que dans le second cas au moins un visage humain est présent. Nous distinguerons également la détection d'éléments du visage (comme par exemple les yeux, le nez, la bouche) qui suppose que l'image représente seulement un visage. Les problèmes

de reconnaissance et d'identification de visages seront également mis à l'écart car il s'agit de trouver une correspondance entre l'image de visage et une image dans une collection de visages à reconnaître.

La détection de visage est un problème classique de reconnaissance de formes qui peut s'énoncer de la façon suivante : considérant une image d'entrée quelconque, qu'elle soit statique ou extraite d'un flux vidéo, il s'agit de déterminer la présence ou non d'un ou plusieurs visages humains dans cette image. Dans le cas de la détection, il sera nécessaire par la suite de retourner par un codage la position spatiale de chaque visage dans l'image avec un rectangle par exemple. La difficulté de ce problème vient du nombre de paramètres à prendre en considération :

- La pose du visage : l'image du visage dépend de la position du visage par rapport à la caméra : visages pris de face, à 45 degrés, de profil, en plongée, ...
- La présence ou non d'éléments structurels comme par exemple la moustache, les lunettes, ... sont autant de facteur qui peuvent même varier en fonction de leur forme, de leur couleur, de leur taille, ...
- L'expression du visage qui elle même peut varier d'un individu à l'autre.
- Le visage peut être partiellement occulté par un autre visage ou un élément du décor : la détection doit alors se priver d'une partie de l'information de la scène.
- L'orientation du visage varie également suivant les différents axes de la caméra.
- De nombreux facteurs intrinsèques à l'image ou à la caméra sont également à prendre en considération tels que l'éclairage, son intensité, ... et les caractéristiques de la caméra en elle-même peuvent altérer l'apparence d'un visage sur l'image.

Pour résumer nous dirons donc qu'un visage est un objet 3D dans un environnement lui-même 3D qui se présente comme une image 2D et à détecter par des méthodes 2D.

Quel que soit le type d'objets d'intérêt à détecter par un système de surveillance, il s'agit de choisir de façon appropriée un espace de représentation et ensuite de proposer une méthode de détection.

- Enfin la dernière classe d'étude est celle du **suivi de primitive ou de formes** dans une séquence d'images. Ce suivi d'un instant t vers un instant $t+1$ se base généralement sur une prédiction de la position de la forme recherchée à l'instant $t+1$ via l'ensemble de ses positions précédentes [Gel98]. Cette prédiction peut guider l'extraction de la forme à l'instant $t+1$ bien qu'elle puisse être également extraite indépendamment de cette prédiction. La principale difficulté réside dans le fait d'avoir non seulement à proposer un suivi le plus précis possible mais également de résoudre les éventuels problèmes d'occultations. Ces dernières peuvent se découper suivant deux classes dépendant de la nature de l'objet intercalé entre l'objectif de la caméra et l'objet à suivre : on distingue ainsi les occultations dites dynamiques et celles statiques. Une occultation dynamique intervient lorsqu'au moins deux objets bougent

et croisent leur chemin respectif dans la scène filmée. Une occultation statique quant à elle se réfère à un objet mobile disparaissant derrière un objet immobile. L'exemple typique de ce genre de problème consiste à suivre un piéton passant derrière un arbre.

1.1.2 Modèles et outils mathématiques

Espace de représentation

Le choix de l'espace de représentation joue un rôle primordial dans les problèmes de détection et de décision car les problèmes d'assignation d'une donnée -image, rétine,...- à une classe -visages, piétons,..- peut se poser de différentes manières.

A titre d'exemple si le problème de détection de visages est abordé suivant une approche type "couleur de peau du visage" le choix d'un espace couleur approprié peut se poser. Mais lequel choisir ? Landry [Lan99] et Liévin [Lié00] résument parfaitement bien les différentes possibilités offertes : un espace couleur n'est ni plus ni moins qu'une représentation dans laquelle les attributs d'une image sont transposés de façon la plus appropriée vis-à-vis de l'application considérée. Si le système de couleur CIE XYZ est l'espace de référence, alors tout autre espace sera inclu dans celui-ci. Ainsi, l'espace couleur RGB plus réduit ne comprend pas toutes les couleurs perceptibles par l'oeil humain et peut être représenté par un cube d'axes rouge, vert et bleu. L'espace CMY (cyan, magenta, yellow) sera lui aussi représenté par un cube à une rotation et une translation près. Les espaces couleurs utiles en traitement d'image sont alors : RGB linéaire, RGB non linéaire (HSV pour Hue, Saturation et Value), CMY(K) (cyan, magenta, yellow et black) et les combinaisons linéaires de RGB telles que YCrCb (luminance, chrominance rouge, chrominance bleu) [Lié00].

Relativement au problème plus général de la détection d'objet dans des images extraites de flux vidéo, le choix de la couleur peau n'est pas, bien évidemment, la seule méthode de représentation envisageable et d'autres espaces plus complexes peuvent être envisagés comme par exemple l'espace des gabarits [JBA00], un ensemble de vecteurs caractéristiques [LI04], etc. Les choix possibles sont nombreux et très diversifiés et dépendent de la nature du problème envisagé.

Dans la suite de ce document nous allons fonder le choix de l'espace de représentation suivant le problème de détection considéré : celui de la détection de mouvement et des objets d'intérêt. Nous présentons également un tour d'horizon des espaces de représentation possibles pour la problématique de détection de visages.

Problématique de décision

Dans la variété des problèmes de décision, on peut distinguer deux cas : dans le premier cas les données d'apprentissage sont étiquetées par un superviseur. Cet étiquetage peut être utilisé par la suite pour affecter l'échantillon inconnu à une des entités pré-étiquetées. On parle dans ce cas précis d'*apprentissage supervisé*.

Dans le deuxième cas il n'existe aucun moyen de connaître les classes d'appartenance des données. Le traitement automatique doit alors faire apparaître des groupements de données. Il faut spécifier les représentations ainsi qu'une mesure de distance dans l'*espace de représentation* qui soient les mieux appropriées pour mettre ces groupements en évidence.

La décision supervisée prend deux formes : la *discrimination* ou la *caractérisation*. Pour la discrimination, il s'agit d'assigner une forme inconnue à une des classes possibles (exemples de la reconnaissance des caractères ou de phonèmes). Il s'agit alors à partir d'un ensemble d'échantillons d'apprentissage de distinguer une frontière de séparation entre ces échantillons pour ensuite distinguer les échantillons à venir en fonction de leur position vis à vis de cette frontière. Dans le cas de la caractérisation, il s'agit de décider si une forme possède les caractéristiques d'une classe ou non (exemple : décider si un électrocardiogramme appartient à la classe "normal", rechercher un mot clé dans une séquence parlée). Dans ce cas précis il s'agit au contraire de regrouper les échantillons d'apprentissage entre eux en groupes -ou clusters- pour ensuite caractériser les échantillons à venir en fonction de la proximité vis-à-vis des groupes créés. Dans les deux cas, l'apprentissage des classes ne se fait pas de la même façon. Pour la discrimination des caractères discriminants sont recherchés entre les classes permettant de faire la différence entre les classes. Dans le cas de la caractérisation, l'approche consiste à chercher des invariants de la classe à caractériser, c'est à dire des exemples stables appartenant à la classe.

Les données brutes issues des caméras ou de n'importe quels capteurs sont les représentations initiales des données à partir desquelles des traitements permettent de construire celles qui seront ensuite utilisées dans le cadre de la décision. Les données brutes peuvent soit être bruitées soit contenir des informations parasites, elles n'explicitent donc pas uniquement les informations utiles pour notre problème de classification.

Un certain nombre de pré-traitements doit être mis en place grâce aux connaissances de ces capteurs, des types données, du problème posé et des méthodes d'apprentissage et de reconnaissance qui seront utilisées. Ces pré-traitements nous seront utiles afin de réduire le plus possible les bruits dus au capteur ou à des interférences avec d'autres sources de signaux.

Une des premières capacités des systèmes de surveillance vidéo consiste à pouvoir travailler aussi bien "*en ligne*" que "*hors ligne*". Seront donc associées à chacune des contraintes temps-réel plus ou moins forte suivant que l'on désire travailler directement en même temps que le flux vidéo est capturé et sans retard perceptible -"en ligne"- ou non -"hors ligne"- loin des considérations temps réel, à des fins d'indexation par exemple.

1.1.3 Objectifs de l'étude

L'objectif principal de cette étude consiste donc à créer un système de télésurveillance générique, qui permette tout aussi bien de traiter des scènes filmées d'intérieur -bureau, hangar, banque,..- que d'extérieur. Comme nous le verrons par la suite, ces deux environnements ont des contraintes sous jacentes totalement différentes qu'il convient de gérer. Or

la plupart des applications de télésurveillance décrites dans la littérature scientifique sont dédiées soit à l'un [IL98, BID+00] soit à l'autre [SEG00, GSRL98] mais peu d'entre elles gèrent cette diversité. La seule contrainte qui est finalement imposée dans cette étude est que *les scènes soient filmées par des caméras fixes.*

Le système généré doit donc permettre de traiter de bout en bout tous les maillons nécessaires de la détection de mouvement, de la détection d'un ou plusieurs objets d'intérêt (un visage), et de le suivre au cours du temps. Une interprétation de ces résultats doit permettre ensuite d'aboutir au déclenchement d'éventuelles alarmes.

Ainsi le premier problème à traiter est celui de la détection d'un objet. Par rapport à ce problème deux choix s'offrent à nous : d'un côté il s'agit de choisir un espace de représentation, de l'autre de proposer des algorithmes de décision les plus génériques possibles et indépendants de l'objet à discriminer et de l'espace de représentation retenu.

1.2 Éléments de l'état de l'art des systèmes existants

Les problématiques de détection du mouvement, d'estimation du mouvement apparent 2D, de segmentation, de la détection d'objet d'intérêt et du suivi temporel de ces objets font partie des dénominateurs communs de nombreux systèmes de surveillance aux objectifs pourtant très divers. Il s'agit, comme nous l'avons vu de tâches difficiles à résoudre, mais elles constituent toutes un maillon crucial. Dans ce chapitre nous allons donc tenter de présenter les différents travaux répondant à tout ou partie des problèmes que nous venons d'aborder.

Nous nous attachons simplement à citer ces applications sans forcément proposer pour le moment d'études exhaustives des méthodes de détection et de suivi temporel qui sont à la base de ces systèmes. Une étude sera toutefois proposée par la suite.

Du fait de la diversité de la nature des applications de surveillance envisagées, toutes n'ont pas les mêmes contraintes. Nous les avons donc regroupées suivant l'utilisation des caméras qui en était faite. On distingue *grosso modo* deux types d'applications : les applications mono-vues et multi-vues.

1.2.1 Applications mono-vues

Le premier système n'est pas à proprement parler un système de vidéo surveillance mais les méthodes employées sont proches de notre application : le système KidsRoom [BID+99, BID+00]. Développé par le MIT Media Laboratory, ce système se propose de créer un espace de jeu pour les enfants. Ce dernier utilise entre autre des algorithmes de suivi temps-réel basés sur l'information contextuelle. Il permet donc de suivre et d'analyser le mouvement, les actions et les interactions entre les enfants et les objets. L'information contextuelle inclut donc la connaissance de l'objet à suivre et ses relations directes avec l'environnement qui l'entoure. Chaque objet est détecté par l'élimination du fond [SIB97]

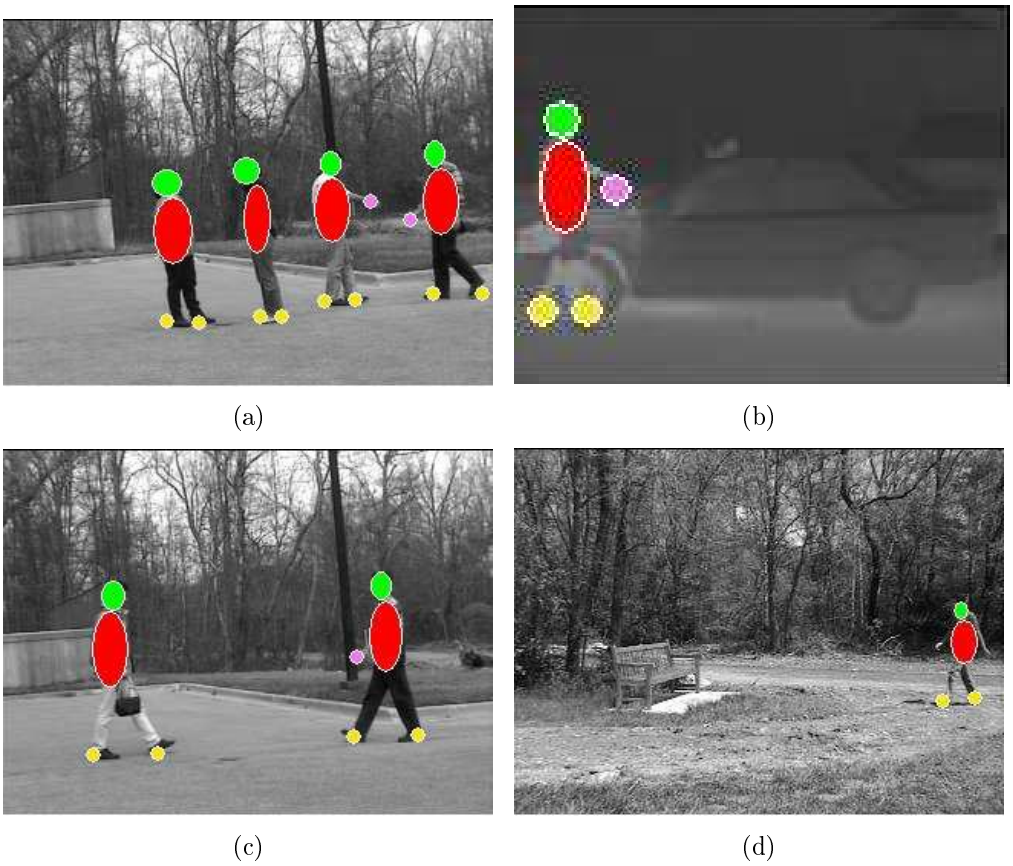
projetant les images dans un espace de blobs dans lequel leur localisation et l'attribut couleur sont mis en correspondance pour définir l'interaction réelle.

W4 [Har98, IL98, LDB98] est un système de suivi temps réel de personnes dans les images monochromes. Le but consiste à répondre aux questions suivantes : que font les personnes filmées ? où vont elles ? quand le font elles ? A partir de modèles dynamiques d'apparence, le système suit un individu, en répondant par ailleurs à une autre question qui ?, même si des occultations interviennent. W4 a été particulièrement étudié pour des scènes filmées à l'extérieur. Un ensemble d'actions prédéfinies -personne échangeant des objets, laissant un objet, prenant un objet,... sont étudiées à partir des mouvements de la scène. Ceci dit le système W4, aux dires des auteurs [IL98], ne pourrait s'appliquer qu'à des systèmes à caméra fixe du fait des outils d'analyse vidéo employés. La détection de mouvement est ainsi effectuée par modélisation du fond suivant les valeurs d'intensité lumineuse minimum et maximum et un seuillage sur l'image de différence. Le fond est ensuite mis à jour de façon adaptative et les blobs -ensembles de pixels en mouvement formant un groupe de pixels connexes- obtenus ne sont pas suffisamment précis pour permettre une extraction de silhouette par exemple. Toujours selon les auteurs leur application de suivi traite 25 images par seconde pour des images de résolution 320×240 sur une architecture de type PIII 300MHz. La figure 1 illustre les résultats de détection et de suivi obtenus sur des vidéos prises à l'extérieur.

Pfinder [WADP97], pour "Person Finder", est un système temps réel également développé dans le but de suivre et d'interpréter le mouvement humain. Initié par le MIT, il a été également déployé pour diverses applications qui vont des jeux vidéos, à la réalité virtuelle en passant par la reconnaissance de la langue des signes. Pfinder détecte le mouvement suivant la traditionnelle méthode de soustraction du fond. La distribution des valeurs de luminosité des pixels de fond est mise à jour de façon récursive par un simple filtre adaptatif. Le corps humain est quant à lui modélisé par un ensemble de blobs connexes obtenu par soustraction. Le modèle représente une combinaison d'attributs de formes, de couleurs et de localisations de ces blobs. Ce système se base donc sur une combinaison lieu/temps/espace des blobs pour détecter des humains à l'image et permet par la même occasion de les suivre au cours du temps. La figure 2 illustre brièvement une partie des résultats obtenus par cette équipe. L'image 2(b) est l'image originale telle qu'elle est acquise par la caméra, l'image 2(c) l'extraction de l'objet et l'image 2(a) la modélisation qui en est faite à partir du résultat précédent.

En France citons les travaux de l'équipe INRIA/ORION [1], Sophia Antipolis. Cette équipe collabore avec de grands noms de l'informatique -Bull, Keeneo, ...- à l'élaboration, à la conception et au développement de logiciels d'interprétation automatique de séquences vidéos pour la Vidéo Surveillance Intelligente (VSI). Ce système de détection automatique des intrusions permet également l'élimination des fausses alarmes liées aux systèmes traditionnels de détection de mouvement (animaux, rondes des vigiles), le tout dans une solution Plug & Play installable sur un système et du matériel déjà existant. Dans [Bre97b], l'auteur

[1] http ://www-sop.inria.fr/orion/



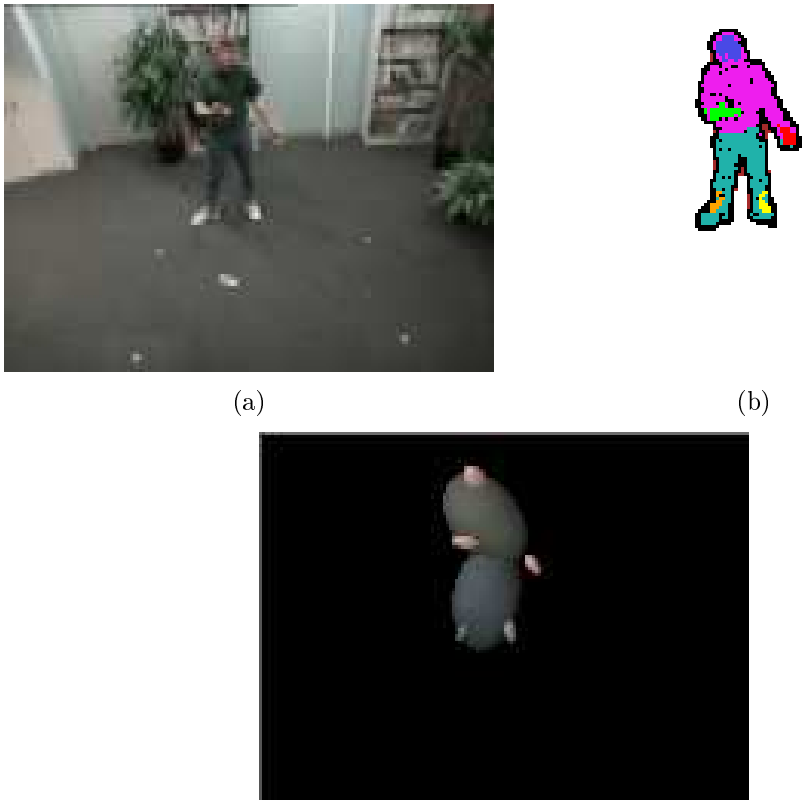
(a) (b)

(c) (d)

Fig. 1. Exemples de résultats obtenus pour la détection et le suivi d'individus via le système W4 -source : http ://www.umiacs.umd.edu/users/lsd/vsam.html

traite notamment de l'interprétation dynamique de scènes à partir de séquences d'images afin d'automatiser la compréhension des activités se déroulant dans une scène donnée. Le capteur utilisé est une caméra fixe et monoculaire filmant à l'intérieur ou à l'extérieur. Les objets mobiles sont principalement des êtres humains et des véhicules. L'objectif de ses travaux consiste d'une part en la modélisation du processus d'interprétation de séquences d'images, et d'autre part en la validation de ce modèle à travers le développement d'un système générique d'interprétation appliqué à la vidéo surveillance, telles que la surveillance de métros et de parkings. L'auteur propose donc un modèle de processus d'interprétation de séquences d'images, basé sur la coopération de trois tâches principales : détection des régions mobiles à l'aide de programmes de traitement d'images, suivi des régions détectées et identification des objets mobiles à partir des régions suivies et la reconnaissance des scénarios relatifs à leurs comportements.

Enfin citons à une moindre mesure le système de surveillance Poséidon [sys05] qui se distingue par son originalité et par son positionnement. Ce dernier est en effet un outil d'aide à la surveillance conçu pour assister les Maîtres-Nageurs-Sauveteurs dans la détection de



(a) (b)

(c)

Fig. 2. Exemples de résultats obtenus pour l'extraction et la modélisation d'un humain à partir de son mouvement grâce au système PFinder -source : http ://vismod.media.mit.edu/vismod/demos/pfinder/

possibles accidents par noyade. Le Système Poséidon est composé d'un ensemble de caméras hautes performances indépendantes les unes des autres qui scrutent en permanence le volume d'eau des bassins, et d'un système expert qui analyse en temps réel les trajectoires des nageurs. Poséidon est ainsi capable de transmettre une alerte aux surveillants dès les premières secondes d'une possible noyade.

1.2.2 Applications multi-vues

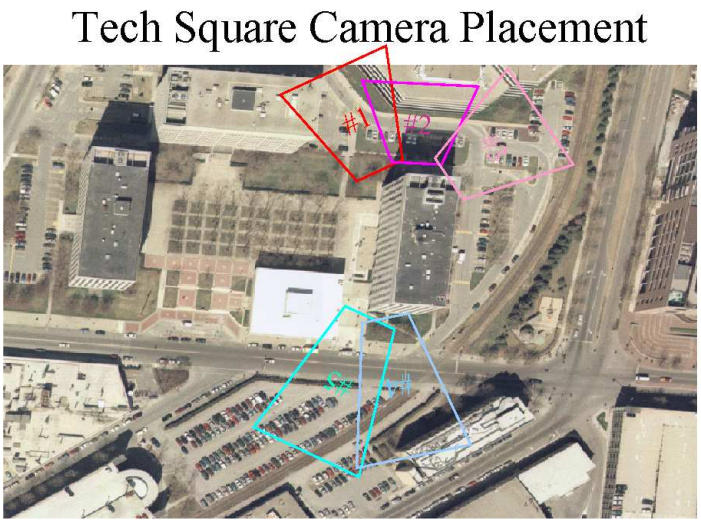
Afin de connaître l'environnement à surveiller certains systèmes ont recours à un ensemble de caméras inter-connectées dont les champs de visées peuvent éventuellement s'entrecouper. Ces méthodes apportent généralement plus de facilités pour les problèmes de suivi et d'occultation par exemple.

Cai et Aggarwal [AC99, CMA95, CA96] ont proposé un cadre de travail pour le suivi de modèles humains pour des environnements d'intérieurs par le biais de multiples caméras synchronisées entre elles. Pour chaque image issue des caméras, un ensemble de traitements

est opéré : détection des objets en mouvement, détection d'humains en mouvement et extraction de caractéristiques clefs pour chaque humain ainsi détecté. Le modèle 2D humain s'appuie principalement sur un modèle du tronc de l'objet à suivre. L'utilisation des moments invariants est requise pour décrire les formes des blobs obtenus ce qui permet ainsi de classer ces blobs en humains/non humains. La mise en correspondance de ces formes est ensuite effectuée suivant un critère de distance -Mahalanobis- pour chaque blob. Les auteurs affirment que leur système opère en temps-réel à 3 images par seconde suivant 3 caméras et pour une résolution de 512×480 sur un processeur RISC.

Le projet VSAM [LCDTH00, LCK99, TKABW97] pour "Video Surveillance and Monitoring project" a été initié au Carnegie Mellon University dans le but de développer un système de vidéo surveillance à partir de multiples caméras actives. Ce système est prévu uniquement pour les environnements extérieurs où la variation de luminosité peut être plus intense que pour des vidéos dites d'intérieur du fait des variations climatiques, des ombres. De plus le vent peut également engendrer des "fausses" détections de mouvement inintéressantes comme par exemple des feuilles en mouvement. Afin de résoudre ce genre de problèmes les auteurs utilisent un modèle de fond adaptatif. Chaque pixel est modélisé par un mélange de lois Gaussiennes (Gaussian Mixture). En supposant que les pixels de fond sont ceux qui d'un point de vue temporel restent affichés le plus longtemps dans la vidéo, le modèle converge vers le modèle fond parfait. Ce modèle permet donc de représenter de façon adaptative le fond tout en tenant compte des petites variations de luminosité. L'extraction des objets en mouvement s'effectue dès lors par soustraction par rapport à l'image de référence modélisée par la mixture. Le problème de ce genre de méthodes, et nous y reviendrons par la suite, est le *phénomène d'absorption* d'un objet qui était en mouvement et qui s'arrête. Le modèle adaptatif va absorber les valeurs de pixels de l'objet arrêté pour les incorporer au modèle de fond de référence. Afin d'éviter ce genre de disconvenue, les auteurs proposent donc une soustraction par "couches", c'est à dire à différents niveaux de complexité dans l'image. Suivant une pré-analyse de la couche au niveau pixel et de la région à laquelle ce pixel appartient, la méthode détermine si le pixel actuel appartient à la classe "fond" ou est dû à un phénomène d'absorption. Pour ce qui concerne le suivi d'objet les auteurs déterminent la meilleure correspondance de l'objet en mouvement entre l'objet à l'instant t et celui à $t-1$ suivant la corrélation des valeurs d'intensité de l'objet en mouvement et des positions spatiales de l'objet entre les images successives. Un réseau de neurones est ensuite utilisé pour classer chaque nouvel objet détecté dans une des classes suivantes : personne, groupe de personnes, véhicule, ... Une analyse discriminante linéaire est appliquée pour fournir plus d'informations sur le type de voiture. La figure 3 illustre le positionnement utilisé lors de la validation sur site du projet VSAM. D'autres résultats sont également disponibles sur le site internet...

Un autre projet VSAM a également vu le jour grâce au MIT pour la détection et le suivi d'objets pour des environnements extérieurs [GSRL98, SG98, SEG00]. Ces travaux précurseurs ont été les premiers à démontrer l'intérêt des modèles adaptatifs pour la modélisation du fond. Ces modèles par mélange de lois Gaussiennes sont par ailleurs devenus une référence. Cette approche a permis, entre autres, de prouver l'extrême flexibilité de ces



(a)

Fig. 3. Exemples de positionnement de caméras retenus pour le projet VSAM du MIT -source : http ://www.ai.mit.edu/projects/vsam/

modèles vis-à-vis du problème d'absorption notamment. L'idée consiste donc à modéliser chaque pixel de l'image de référence par un mélange de lois Gaussiennes et de proposer un schéma de mise à jour le plus rapide possible aussi bien en termes de temps de calcul qu'en terme de rapidité de ré-apprentissage du modèle. Un autre intérêt provient du fait de pouvoir modéliser efficacement des lois bi-modales. Cette remarque est particulièrement intéressante quand la scène filmée est fortement bruitée par de fausses détections. Ce genre d'environnement provoquerait de fortes erreurs avec une seule loi normale par exemple. La construction de l'image de référence du fond permet ensuite d'effectuer sa soustraction avec l'image acquise par la caméra et ainsi d'extraire les éventuels objets en mouvement. Ce projet est par ailleurs intéressant pour une dernière raison : il apporte une solution efficace pour la mise en correspondance des caméras afin de recréer un environnement virtuel global. A partir des trajectoires 2D acquises par un couple de caméras, une correspondance homographique est réalisée afin de projeter les scènes filmées par les 2 caméras dans un même espace.

Chang et Gong [CG01, CG00] ont développé un système de surveillance du trafic autoroutier. Il est constitué de caméras asservies et mobiles connectées à un réseau Gigabit Ethernet où sont directement transférées les images en pleine résolution. Le système proposé permet de suivre des voitures et de déclencher des alarmes suivant des comportements prédéfinis. Le système tel qu'il a été développé tend à prouver qu'il est nécessaire d'avoir une infrastructure réseau à large bande passante pour permettre de réaliser ces suivis en temps réel. Évidemment, cela se traduit par la netteté et la qualité des images transférées sur le réseau qui se traduit par des résultats plus ou moins bons suivant la bande passante accordée.

Khan, Javed et Shah [OJS03, JS02, SKS01a, SKS01b] ont récemment présenté un système multi-vues qui s'adapte tout aussi bien aux scènes d'intérieur que d'extérieur via des caméras qui ne sont pas forcément calibrées. Leur méthode permet de reconnaître les scènes entrecoupées de l'environnement et ainsi permet de conserver l'identité de l'objet dans l'environnement. Un ensemble de scénarii est proposé pour la détection grâce aux données spatio-temporelles de la vidéo et des contenus couleur de la forme en mouvement détectée. Un ensemble d'apprentissage est supposé connu pour chacune des scènes filmées par les caméras.

1.3 Évaluation des algorithmes de vidéo surveillance

Une des questions les plus ardues de la surveillance vidéo automatisée est finalement la validation des outils et des méthodes employés. Une validation de chaque composante (détecteur de mouvement, détecteur d'objet d'intérêt, suivi) individuellement permet certes de déterminer la pertinence d'un traitement mais comme nous l'avons décrit précédemment un système de surveillance vidéo reste, avant tout, une série de composants de traitement inter-connectés les uns aux autres et des résultats de l'un dépendent des résultats des autres.

Il existe de nombreuses façons d'évaluer les performances d'un ou plusieurs algorithmes : de la façon la plus simple c'est à dire en analysant chaque pixel individuellement (on parlera alors de méthodes d'évaluation bas niveau), soit au contraire des méthodes plus complexes qui considèrent la globalité des résultats de détection pour en déterminer leur efficacité (méthodes d'évaluation haut niveau). Il est indispensable que la méthode soit indépendante de la spécificité de l'algorithme de classification employé.

Les évaluations proposées ici sont présentées sous le couvert du problème de la détection de mouvement dans la vidéo mais elles trouvent également leur justification dans les autres problèmes de classification.

1.3.1 Évaluation bas niveau

Les performances des algorithmes de détection de mouvement par exemple peuvent être évaluées quantitativement et visuellement suivant les besoins de l'application. L'approche finalement la plus sûre pour une évaluation visuelle qualitative consiste à afficher une animation où n'apparaissent que les pixels/objets en mouvement sur un fond noir.

D'un point de vue quantitatif cette fois-ci le problème est plus délicat dans le sens où soit il est difficile d'avoir soit il n'existe pas de vérité terrain -"ground truth" en anglais- des cartes de détection. On entend par le terme "vérité terrain", la réponse normalement attendue du processus de détection. Un deuxième challenge est également d'évaluer la relative importance des différents types d'erreurs rencontrés...

La génération d'une vérité terrain est lourde en temps et extrêmement difficile à réaliser.

Pour le moment la vérité la plus appropriée est celle fournie par un opérateur humain. Ceci dit les algorithmes proposés doivent répondre à des problèmes complexes où l'expérience humaine peut elle même être source d'erreurs. Un même opérateur peut générer, pour une même expérience, des vérités terrains complètement différentes.

La méthode la plus "conservatrice" se devrait alors de confronter ces résultats à l'intersection de toutes les vérités terrains proposées par les opérateurs. En d'autres termes, les détections retenues par l'algorithme seraient considérées valides si tous les opérateurs humains ont considéré ce changement comme étant valide. A contrario une autre approche consisterait non pas à employer l'intersection mais l'union des résultats proposés par les opérateurs. Finalement on peut toutefois affirmer qu'il ne faut pas forcément être rigide pendant le processus de détection et qu'un consensus peut et doit être trouvé.

Une fois une vérité terrain établie, il existe un certain nombre de méthodes qui permettent de comparer la vérité terrain au masque binaire de mouvement candidat. Pour cela les termes suivants sont généralement employés et méritent d'être éclaircis. Pour ce faire nous emprunterons à la terminologie du problème de détection de mouvement où la classification au niveau d'un pixel s'effectue pour rappel en deux classes "fond" et "objet en mouvement" :

- Vrais positifs (TP) : Nombre de pixels appartenant à la classe "objet en mouvement" dans la vérité terrain correctement détectés ;
- Faux positifs (FP) : Nombre de pixels appartenant à la classe fond dans la vérité terrain et détectés comme étant des pixels d'objets en mouvement ;
- Vrai négatifs (TN) : Nombre de pixels de fond correctement détectés, et
- Faux négatifs (FN) : Nombre de pixels appartenant à la classe "objet en mouvement" et détecté comme appartenant à la classe "fond".

Remarquons tout de suite que la valeur $TP+FN$ correspond à la dimension de la vérité terrain. Rosin [RI03] résume trois méthodes permettant de quantifier les performances d'un classifieur

- **Le pourcentage de classification correcte PCC** définit par

$$PCC = \frac{TP + TN}{TP + FP + TN + FN}$$

- **Le coefficient de Jaccard** est défini comme étant

$$JC = \frac{TP}{TP + FP + FN}$$

- et enfin **le coefficient de Yule** par

$$YC = |\frac{TP}{TP + FP} + \frac{TN}{TN + FN} - 1|$$

Par rapport au problème de détection des pixels appartenant aux objets, ces coefficients ont finalement pour objectif de quantifier le degré de similarité entre deux images binaires. La façon la plus intuitive et systématique consiste à combiner les 4 valeurs (TP,FP,TN et FN) pour calculer le coefficient PCC. Il s'agit d'ailleurs de la méthode la plus répandue de vérification des performances d'un algorithme. Cependant ce coefficient a tendance a induire en erreur quand le nombre de changements à l'image est très petit vis à vis de la dimension de l'image [RK99, DC96]. Il a été dès lors proposé d'autres métriques palliatives qui permettent justement de prendre en compte cette spécificité. Ainsi les coefficients de Jaccard et de Yule finalement permettent de minimiser voir d'éliminer le problème obtenu avec le coefficient PCC quand le nombre de vrais négatifs est anormalement important. Notons néanmoins que le coefficient de Yule ne peut être appliqué quand l'algorithme ne détecte correctement aucun changement à l'image puisque l'un des dénominateurs devient alors égal à zéro.

La mesure dite de *rappel -recall* en anglais- est définie comme étant le ratio entre le nombre de formes assignées correctement à une classe sur le nombre total d'éléments assignés à cette même classe dans la classification de référence ou vérité terrain.

$$\text{Rappel} = \frac{\mathbf{TP}}{\mathbf{TP} + \mathbf{FN}}$$

Les erreurs de classification sont caractérisées par la précision : la mesure d'*erreur -precision* en anglais- quant à elle désigne le rapport entre le nombre de formes pertinentes trouvées sur le nombre total de formes trouvées.

$$\text{Précision} = \frac{\mathbf{TP}}{\mathbf{TP} + \mathbf{FP}}$$

Par souci de comparaison de nos outils et puisque ces dernières mesures sont fréquemment utilisées dans le domaine multimédia, nous allons donc les adopter dans la présente étude.

Pour un classifieur paramétrable il est également envisageable de dessiner la courbe ROC (pour "Receiver Operating Characteristics") qui trace la probabilité de détection sur la probabilité de fausse détection dans le but de déterminer le niveau désiré de performance à obtenir. On peut aussi avoir dans certains cas un taux de rejet qui correspond à la décision de ne pas classer la forme avec un taux de rejet variable on peut tracer des courbes rappel×précision.

Il est également envisageable d'avoir recours à des indicateurs pour évaluer les algorithmes de classification à partir des données de rappel et de précision. Ces indicateurs peuvent être utilisés indifféremment pour l'évaluation d'algorithme de classification ou de catégorisation. Afin de faciliter la lecture de nos résultats, nous avons recours notamment à l'indicateur F-mesure défini par :

$$\text{F-mesure} = \frac{(\mathbf{1} + \beta^{\mathbf{2}}) \times \text{Précision} \times \text{Rappel}}{(\beta^{\mathbf{2}} \times \text{Précision}) + \text{Rappel})}$$

avec généralement $\beta^2 = 1$ ce qui fait de cette mesure une moyenne harmonique. Dans ce contexte de prédiction, il est également utile d'être plus fin dans l'évaluation des performances et de prendre en compte non seulement un taux d'erreur mais également un taux de faux positifs et de faux négatifs. Trop souvent le coût de mauvaise classification n'est pas symétrique et on peut préférer avoir un taux d'erreur un peu moins bon si cela permet de réduire le type d'erreur le plus coûteux : à titre d'exemple il vaut mieux détecter à tort un visage (faux positifs) plutôt que de ne pas détecter un visage (faux négatifs). La courbe ROC présentée à la figure 1.3.1 permet ainsi de régler ce compromis.

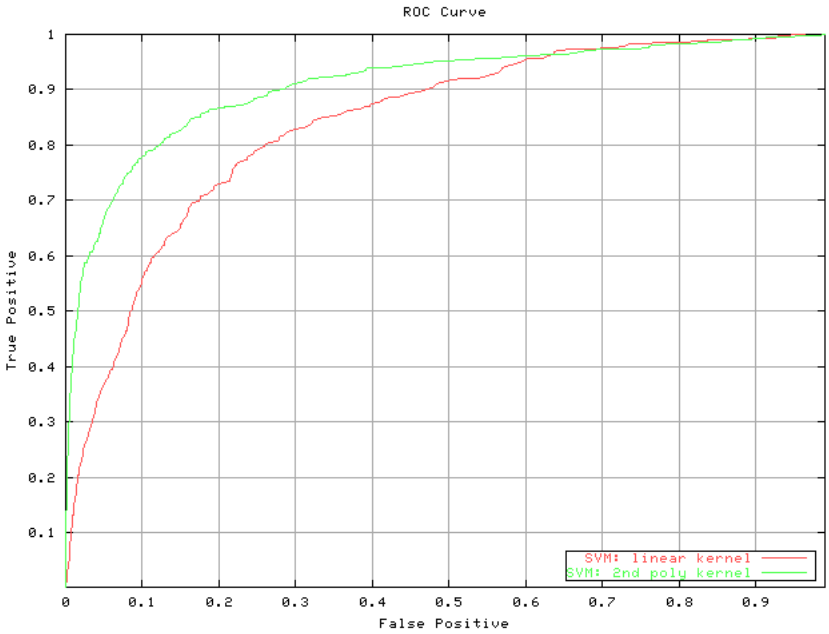


Fig. 4. Exemple de courbe ROC pour un classifieur SVM opérant sur une base de visages. Source : http ://vision.ai.uiuc.edu/mhyang/face-detection-survey.html

Plus la courbe ROC s'éloigne de la diagonale pour rejoindre l'angle supérieur gauche, plus le processus de classification est globalement "puissant". Comme nous le disions auparavant un processus de classification n'est pas obligé d'être "le meilleur dans toutes les catégories comprises" pour être retenu et considéré comme valide. Notons que la diagonale passant par l'origine correspondrait aux performances d'un système aléatoire, l'augmentation du taux de vrais positifs s'accompagnant d'une dégradation équivalente du taux de faux positifs...

Dans le domaine de la recherche d'information, les mesures de qualité adoptées sont les taux de rappel et de précision et MAP. Ces mesures ont été de plus en plus utilisées pour l'évaluation des tâches d'indexation multimédia (TREC VIDEO notamment

[PBPKD04, QMS+04, KBP05]) où elles sont appliquées pour caractériser aussi bien la qualité de recherche d'information que la détection de différents phénomènes dans les médias numériques.

1.3.2 Évaluation haut niveau

Si l'on reprend le problème précédent de détection de mouvement, l'évaluation n'était effectuée qu'au niveau du pixel. Cette notion est totalement arbitraire et finalement ne répond pas à la question de savoir si le pixel en question appartient à un objet en mouvement ou non. De l'étude précédente on supposait une détection parfaite de l'objet par exemple. Qu'en est il si l'objet en mouvement est entièrement détecté, n'est que partiellement détecté ou si un seul pixel de l'objet est détecté. De façon identique qu'en est il si le processus de détection segmente parfaitement l'objet mais laisse une tache d'erreur à l'intérieur de cet objet ? Quel est le degré de validité d'une telle approche par rapport à un autre processus offrant une détection sans tache par exemple ? La notion d'objet était jusqu'alors totalement exclue de notre discours et doit par conséquent être soumise à discussion.

Il n'existe malheureusement aucune métrique permettant justement de quantifier la détection obtenue en terme d'objet. Néanmoins certaines études tentent d'y répondre suivant une approche par blob [Zha96]. Suivant la zone couverte par le blob détecté vis-à-vis d'une couverture de référence et de la compacité de ce blob, Zhang introduit une méthode de seuillage pour évaluer la segmentation obtenue. Les résultats présentés ne restent toutefois applicables que sur des imagettes simples...

L'évaluation des performances des algorithmes de suivi suscite également un très vif intérêt de la part de la communauté scientifique internationale avec notamment l'ouverture du groupe de travail "Performance Evaluation for Tracking and Surveillance" -PETS- sponsorisé par le "IEEE Computer Society Technical Committee" [2]. Un des problèmes qui a permis d'être résolu par ce groupe de travail, fut notamment la mise en place d'un corpus commun afin de permettre la comparaison et l'évaluation des systèmes sur un même pied d'égalité. Couplées à une vérité terrain annotée manuellement par un opérateur où chaque nouvel objet apparaissant dans le champ de vision est clairement identifié et inscrit, ces bases de test constituent un cadre de travail commun à l'ensemble des équipes désirant participer. A titre d'exemple, cette vérité terrain comprend entre autre la forme de la trajectoire de l'objet suivi et une boîte englobante de l'objet en mouvement. L'algorithme de suivi peut ensuite être exécuté sur la séquence vidéo pré-enregistrée et la sortie de l'algorithme comparée à la vérité terrain établie pour obtenir les performances globales du système de suivi.

Dans [Ell02], Ellis propose un ensemble de méthodes d'évaluation des algorithmes de suivi en fonction des conditions climatiques, des changements de luminosité et des mouvements dits non pertinents -feuilles, ...-. Sans rentrer dans le détail du schéma retenu, des considérations sur la complexité de l'ensemble des données à traiter sont à prendre

[2] http ://www.computer.org/,http ://www.ieee.org/

en compte également. Ellis considère ainsi le nombre d'objets présents dans la vidéo ainsi que le nombre d'occultations rencontrées dans ce jeu de données. Il suggère finalement que l'évaluation d'un algorithme de suivi doit couvrir un large spectre d'environnements, de conditions et de natures différentes afin de fournir un test le plus pertinent possible.

Needham et Boyle [NB03] ont défini un ensemble de méthodes pour l'évaluation du positionnement d'un algorithme de suivi. Leurs métriques s'appliquent notamment lors de la comparaison de la trajectoire des objets à suivre. La notion de trajectoire est considérée comme un ensemble de positions ordonnées chronologiquement de l'objet à suivre. Une vérité terrain est établie par un même opérateur afin de garder une certaine stabilité de mesure et les résultats des algorithmes de suivi comparés suivant leurs décalages temporels et spatiaux. Cet ensemble de décalages évalués peut ensuite permettre une évaluation quantitative des algorithmes de suivi.

Pingal et Segen [PS99] ont introduit à leur tour un ensemble d'outils d'évaluation. En l'absence de vérité terrain parfaitement établie qui permettrait de comparer les algorithmes de suivi, les résultats sont souvent alors comparés visuellement ce qui est bien entendu totalement subjectif. Les auteurs ont donc proposé trois catégories de métriques d'évaluation : la première évalue le taux d'erreur de suivi enregistré et le nombre d'images où le suivi a "décroché". La deuxième métrique estime la durée du suivi de l'objet par le système ce qui ne peut être calculé précédemment du fait que la métrique précédente n'indique pas si l'objet est suivi tout le temps ou simplement une fraction de seconde. Enfin la dernière mesure porte sur le positionnement des résultats du suivi dans les images et sur sa proximité avec la vérité terrain. Conscients que finalement ces métriques ne sont valables que pour une vérité terrain déjà bien établie, les auteurs ont également proposé un ensemble de métriques alternatives, plus grossières, basées sur la notion d'événements déterminés par le passage d'un objet dans une zone restreinte et délimitée de l'image à l'instant t. Ces métriques ont finalement elles-mêmes leur propre limite puisque la définition des points d'intérêt que l'objet doit traverser reste manuelle et donc subjective ...

D'autres travaux, plus récents, proposent également des métriques pour la mesure des performances des algorithmes de suivi. Ainsi, dans [CZ05], la qualité du suivi à chaque instant de temps t est caractérisée par l'intersection entre le masque de la vérité terrain et le masque de l'objet obtenu par l'algorithme de suivi. Une telle approche est avantageuse dans le cadre de la validation incrémentale des modifications des algorithmes, pour comparer, entre autres, les performances d'autres propositions alternatives.

Après l'étude de ces quelques travaux, il est important de rédiger un protocole d'évaluation dans lequel doivent être associés un ensemble de données étiquetées -vérité terrain- et un ensemble de métriques prédéfinies. En ce qui concerne la vérité terrain pour le domaine de la vidéo surveillance, nous pouvons citer les données PETS [3], FGNET [4], CAVIAR [5],

[3] http ://www.visualsurveillance.org
[4] http ://www.fg-net.org
[5] http ://homepages.inf.ed.ac.uk/rbf/CAVIAR/

ARGOS [6], ... Ces immenses bases de données vidéo contiennent des centaines de milliers d'images mais sont généralement appliquées à certains types de traitements particuliers dus au contexte environnant le projet.

1.4 Présentation des corpus de validation

L'ensemble des outils de détection et de suivi que nous avons proposé a été validé sur un certain nombre de corpus vidéo mis à disposition pour une évaluation nationale ou internationale.

Il peut s'agir de corpus vidéo dédié spécifiquement à la vidéo surveillance comme c'est le cas pour la campagne Argos, mais il est également intéressant de tester nos outils de détection de visages par exemple sur des corpus plus vastes en considérant notamment les vidéos de journaux télévisés d'ABC et CNN, corpus diffusé lors de notre participation à la campagne TREC Video 2004 [QMS+04]. Nous allons donc brièvement présenter ces deux campagnes et leurs corpus de validation associés.

1.4.1 Campagne Argos Technovision

Le corpus Argos comprend un ensemble de trente heures de vidéo de surveillance enregistré avec du matériel de nature différente et de qualité différente : caméra analogique reliée à une carte d'acquisition, caméra IP relié à un serveur HTTP. La première fournissant un flux vidéo brut de type YUV 420P, la seconde une succession d'images JPEG plus ou moins compressées suivant la qualité et le débit du réseau proposé. Les flux vidéo de surveillance proposés peuvent également avoir des cadences temporelles différentes (de 3 à 4 images par seconde pour la caméra IP par exemple et de 25 ou 30 images par seconde pour l'autre) ajoutant ainsi une contrainte supplémentaire sur les algorithmes de détection de mouvement entre autres.

Au niveau national (et maintenant international avec la diffusion du corpus sous le couvert de l'action COST292 et du Queen Mary University of London), la campagne Argos propose de délivrer à l'ensemble de ses participants une vérité terrain sur des vidéos de télésurveillance à débit et à qualité variable. A partir d'une division en segments, un opérateur a annoté manuellement un certain nombre d'événements dans le but de l'évaluation des outils d'analyse et d'indexation vidéo. Appliquée à notre problématique de surveillance vidéo, la campagne Argos a retenu les tâches suivantes :

L'intérêt de cette campagne vient non seulement de la diversité de son corpus mais également des scénarios d'évaluation proposés. Dans le cadre de la surveillance vidéo, les tâches à considérer sont :

– l'identification du lieu de tournage (intérieur/extérieur)

[6] http ://www.irit.fr/recherches/SAMOVA/MEMBERS/JOLY/argos/

- l'identification de la présence de personnes à l'image,
- l'identification des différentes apparitions d'une même personne et enfin
- l'identification de comportements humains

Cette dernière revêt un caractère particulier et se décompose en trois scénarios :

- personne qui court,
- personne s'arrêtant et, enfin,
- personne déposant un objet à terre.

Les images de la figure 5 et 6 donnent un aperçu du contenu des séquences vidéos. Les séquences enregistrées et présentées à la figure 5 sont issues d'une caméra analogique reliée à une carte d'acquisition. Les données filmées sont donc soumises principalement aux bruits intrinsèques de la caméra. Ces vidéos contiennent comme nous pouvons le constater aussi bien des scènes d'intérieur que d'extérieur avec la présence ou non d'objets en mouvement, éventuellement de toute petite taille. Les séquences de vidéos de surveillance présentées à la figure 6 ont été enregistrées par l'IRIT via une caméra IP. Un flux d'images JPEG est encapsulé dans des messages HTTP à leur tour encapsulés dans des paquets IP qui sont ensuite émis sur le réseau : dès lors le débit peut être variable, les images peuvent même manquer dans le flux de données, bref rien n'assure que le flux ne soit pas corrompu. Il est important de noter qu'ici le bruit intrinsèque de la caméra n'est plus la source majeure de bruit du fait de l'élimination des hautes fréquences lors de la compression JPEG. Apparaissent néanmoins des artefacts typiques au JPEG tels que "les effets de blocs" observables sur les images du flux.

La base Argos de vidéo surveillance comprend 1427 minutes de vidéo. Dans ce travail de thèse nous n'avons utilisé qu'une partie du corpus pour l'évaluation interne de notre outil. L'autre partie sera par la suite évaluée lors de la campagne proprement dite. Il s'agit ici de tester nos méthodes sur des extraits de la base -1 heure de vidéo- mis à disposition pour la phase dite d'"apprentissage".

1.4.2 Corpus TREC Vidéo 2003 NITS/ARPA

Le but principal de la campagne TREC Video Retrieval Evaluation (TRECVID) est de promouvoir les outils d'analyse et d'indexation basés contenu à partir de vidéos numériques enregistrées grâce à un système ouvert d'évaluation. La campagne TRECVID suggère un découpage en "tâches" correspondant à des situations du monde réel. TRECVID distingue trois tâches principales auxquelles tout participant doit au moins répondre à une d'entre elles. Les tâches proposées en 2003 étaient les suivantes :

- La première tâche consiste à détecter les frontières de plan -shot boundary determination- [PBPKD04] ce qui consiste à identifier les frontières de plans suivant leur type -coupure ou fondu- pour un clip vidéo donné.

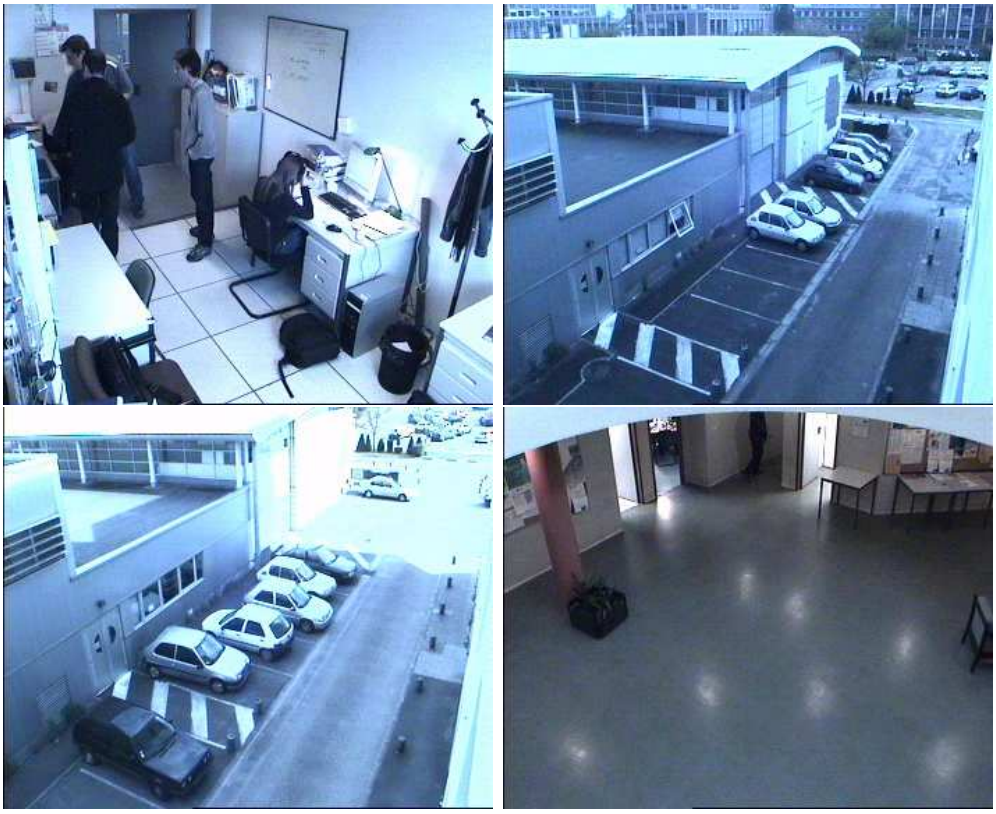


Fig. 5. Échantillons d'images (intérieur/extérieur) extraites du corpus Argos de surveillance vidéo filmé à partir d'une caméra analogique -25 img/s.

- La deuxième tâche détermine la délimitation des frontières de scènes sémantiques -story segmentation-. Pour une collection de frontière et de contenu sémantique donné, cette tâche consiste à relever pour un clip, les frontières temporelles et sémantiques regroupant une même histoire. Une histoire est un ensemble de plans regroupant par exemple une même personne présente à l'image ou un même thème sémantique.
- Enfin la dernière tache permet l'extraction de contenu haut niveau défini au préalable par les organisateurs. Ces contenus sont généralement considérés comme des objets d'intérêt susceptibles d'apporter un contenu sémantique à la vidéo. Ces objets peuvent d'une campagne à l'autre être tout aussi un visage, une voiture, une porte de prison. Cette tâche est appelé "high-level feature extraction task".

Lors de notre participation nous avons proposé une évaluation en commun avec les laboratoires CLIPS, LIS et LSR en vue de la détection et de la reconnaissance des plans contenant les visages Bill Clinton et Madeleine Albright [QMS+04].

Une fois l'évaluation terminée, nous nous sommes munis de la base d'images offerte par cette campagne pour l'apprentissage et l'évaluation de nos outils à des fins futures. Ceci nous a permis notamment d'obtenir une base d'apprentissage de visages la plus importante et pertinente possible. La figure 7 donne quelques exemples d'images clefs extraites de ce

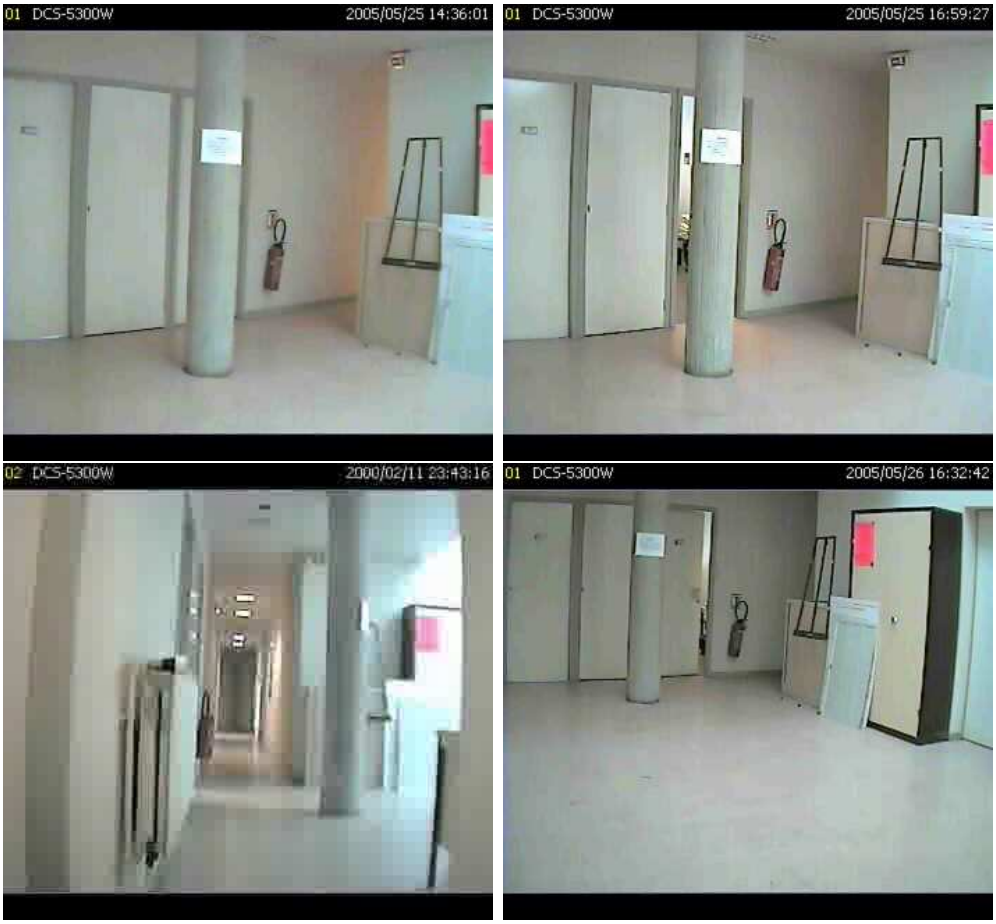


Fig. 6. Échantillons d'images extraites du corpus Argos de surveillance vidéo filmées à partir d'une caméra IP -débit variable-.

corpus.

Cette base a été utilisée pour une partie de ce travail de thèse et notamment pour l'entraînement et l'évaluation des performances des processus de détection et de suivi de visage humain dans le flux vidéo. La diversité de la base TREC s'est avérée intéressante pour constituer la base d'apprentissage de visages.

1.4.3 Corpus propre de vidéo surveillance

La campagne Argos n'ayant réellement commencé qu'à partir de Janvier 2005 et la soumission de nos résultats se faisant courant février 2006, il nous était impossible d'évaluer réellement nos outils avant le début de la campagne. Dans le souci d'évaluer nos outils, nous avons donc créer notre propre corpus de surveillance vidéo. Parmi l'ensemble des sources de vidéos possibles, la vidéo surveillance est certainement la plus riche. Nous avons créer des environnements de complexité différente afin de répondre aux scénarios envisagés.

Fig. 7. Échantillons d'images extraites du corpus TREC Vidéo 2003

Dans notre approche nous avons essayé de ne pas nous restreindre à une scène particulière ni à un environnement particulier. Les images de la figure 8 sont extraites du corpus que nous avons créé spécialement pour la validation. Elles donnent quelques exemples de la diversité des situations choisies : extérieur, intérieur, rond point, bureau,... et sont autant d'éléments de réponse à la validation de nos méthodes de surveillance. Les vidéos d'intérieur présentées sur cette figure ont notamment la particularité de proposer trois sources de lumière différentes : une source lumineuse offerte par l'éclairage au néon de la scène pour l'éclairage général de la pièce couplé à deux autres sources de lumière issues de la lumière extérieure offerte par deux portes vitrées. La scène filmée a donc un comportement extrême du fait des ombres portées des objets sur les murs qui sont d'autant plus nombreuses que le nombre de sources lumineuses est important. Comportement extrême également du fait des fortes variations potentielles des intensités lumineuses dans un laps de temps parfois très court. La durée globale de ce corpus est de 15 heures à la cadence temporelle de 25 images par seconde.

1.5 Dispositif de surveillance retenu

1.5.1 Scénarios de surveillance

Le dispositif proposé est orienté objet ce qui veut dire qu'à partir d'une connaissance *a priori* de l'objet d'intérêt, le système doit détecter et suivre cet objet le plus longtemps possible en se basant uniquement sur cette connaissance. Il est alors important de dégager un certain nombre de scénarios de surveillance, impliquant un ou plusieurs objets.

Les scénarios retenus sont les suivants : le premier scénario consiste à détecter la présence d'individus entrant dans la scène filmée et à le suivre le plus longtemps possible tant qu'il reste dans la scène filmée. Ce scénario, bien que classique, fait intervenir un deuxième scénario dans le cas où le suivi "décrocherait". Si ce dernier cas intervient, on relance le processus de détection non pas sur l'intégralité de l'image mais sur un voisinage proche de la position précédente de l'objet à suivre.

Ce genre de situation est tout à fait typique de la surveillance de locaux : à titre d'exemple, une société souhaiterait connaître le nombre exact d'employés actuellement présents dans ses locaux à partir de la différence entre le nombre de personnes qui sont entrées et celles qui sont sorties.

1.5.2 Système de surveillance proposé

Un système automatisé de surveillance vidéo doit comprendre un ensemble de composants performants. Chacun de ces composants se classe suivant la complexité des opérations qu'il réalise. On distingue deux types d'opérations : les premières forment l'ensemble des opérations de bas-niveau pour l'acquisition, la détection de mouvement par exemple qui, dans notre système, doit demander peu de ressources en calcul. Les opérations plus complexes relèvent d'un autre niveau d'interprétation, comme par exemple la détection d'un objet d'intérêt -un visage, un piéton,..- et son suivi au cours du temps. Le schéma global du système de surveillance retenu dans notre travail est présenté à la Figure 9. Entre chaque opération (détection de mouvement, visage, ...) est associé un canal de transmission de données où circule un flux d'informations. A titre d'exemple sur la Figure 9, le processus "détection de visages" accepte deux entrées : soit une image directement acquise par le processus de "capture", soit un masque de mouvement et une image en vue de la détection de visages dans une zone d'intérêt délimitée par le mouvement apparent dans la scène.

Le premier processus de traitement consiste naturellement à effectuer l'acquisition d'un flux d'entrée (via une caméra -Video For Linux2-, ou un ensemble de fichiers image -jpg,pnm,...-, etc) et de construire une structure d'image appropriée pour la transmettre à l'ensemble des autres processus de ce système. Cette structure doit être suffisamment flexible pour permettre la manipulation des images suivant différents espaces de représentation comme ceux cités précédemment.

Le premier processus qui fait intervenir les outils de décision dans cette étude est celui de la détection de mouvement. Cette segmentation au sens du mouvement apparent à l'image définit les zones d'intérêt pour l'analyse et les décisions suivantes. Cette première segmentation offre un compromis entre rapidité de calcul et précision des masques afin de fournir aux autres traitements des cartes de mouvement les plus fines possibles. Cette première étape est donc une étape primordiale. Elle doit être suffisamment robuste non seulement aux changements lents mais également rapides des conditions de luminosité de la scène tout en restant la plus précise possible quant aux objets en mouvement. A cela s'ajoutent les conditions climatiques ou les variations de température qui modifient le bruit intrinsèque de la caméra. Tous ces paramètres sont à prendre en compte par ce premier processus de détection de mouvement.

Au sein de cette même étape nous avons également intégré un traitement définissant une consistance spatio-temporelle des masques de mouvement obtenus. Certains mouvements présents sur la scène ne sont pas tous pertinents. Des petits mouvements peuvent apparaître comme par exemple lorsque le vent bouge les feuilles, où lorsque la lumière réfléchit d'une vitre, d'un pare-brise vers le capteur de la caméra. Toutes ces configurations sont sources d'erreurs potentielles de détection lors de l'étape de segmentation du mouvement. Nous avons donc intégré une régularisation spatio-temporelle de ces masques en vue d'éliminer ces fausses détections et de compléter les masques de mouvement parfois creux.

La segmentation régularisée ainsi obtenue permet ensuite d'effectuer des opérations de plus haut-niveau non plus sur la totalité de l'image, mais sur une zone réduite de l'image : la zone de mouvement propre. Ce découpage de l'image en zones d'intérêt permettra un gain de temps conséquent pour des applications déjà gourmandes en temps de calcul comme c'est le cas pour la détection de visages. L'opération de suivi profitera également de cette optimisation pour raffiner ces résultats et améliorer son pouvoir prédictif. Le processus de détection de mouvement est véritablement au coeur de ce système de surveillance, et sert de point d'entrée pour l'ensemble des autres processus mis en jeu.

Suite à cette segmentation intervient un deuxième traitement de détection d'un objet d'intérêt : un visage dans notre étude. Outre l'intérêt scientifique, le problème de détection de visages dans la vidéo a été retenu du fait des scénarios de surveillance considérés : chacun de ces scénarios a pour point commun la présence d'un individu à l'image. Cette présence peut être déterminée de plusieurs façons soit par une analyse de silhouettes par exemple, où l'on va chercher à catégoriser la silhouette de mouvement d'un piéton par rapport à celles d'autres objets présents à la vidéo [HFK98], soit par l'étude des textures en vue de la détection de costumes [Jaf06] par exemple. Finalement le problème de détection de visages, s'il n'est pas utilisé en tant que processus "majeur" pour la détection des individus, est souvent une méthode additionnelle permettant d'améliorer les taux de détection des processus cités précédemment.

Un des points les plus importants pour les systèmes de télésurveillance automatisés reste finalement la capacité à préserver l'identité de l'objet jusqu'à ce que l'objet disparaisse de la scène. Ceci présente donc une capacité additionnelle au problème de segmentation de

mouvement puisqu'il faut à chaque image trouver la correspondance de l'objet sur l'image suivante.

Finalement tous les résultats des processus de traitement sont ensuite directement injectés vers un processus d'affichage. Il s'agit dans cette partie de fournir une interface visuelle d'aide à la prise de décision pour un opérateur.

Un système de télésurveillance relève bien comme on peut le constater de sujets de recherche relativement diversifiées (détection de mouvement, détection d'objet d'intérêt, suivi, éventuellement étalonnage de la caméra, ...). Dans le contexte de ce travail, nous avons cherché à proposer des solutions statistiques à base d'apprentissage pour les trois problèmes énoncés, détection du mouvement, détection de l'objet d'intérêt et suivi d'objet. Dans le chapitre suivant nous allons donc considérer, dans un premier temps, le problème de détection des objets en mouvement par apprentissage non-supervisé.

Fig. 8. Échantillon d'images extraites de notre corpus de surveillance vidéo filmée à partir d'une caméra analogique à 25 images par seconde.

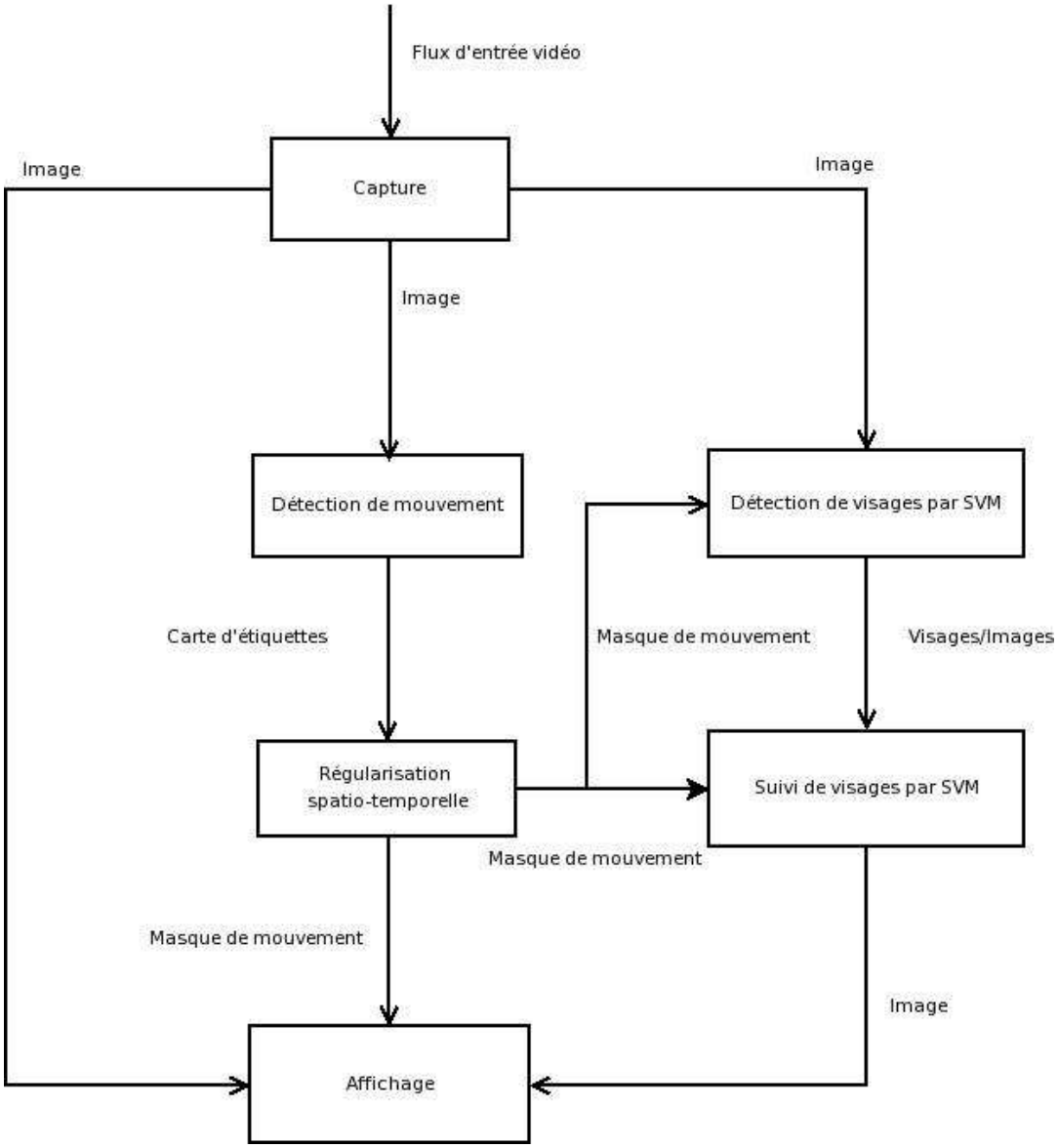


Fig. 9. Diagramme global de l'enchaînement des processus

Chapitre 2

Détection d'objets en mouvement par apprentissage non-supervisé

Comme nous l'avons vu lors de la présentation du schéma global de notre approche, la détection des zones en mouvement revêt un caractère essentiel dans l'ensemble des traitements de la vidéo. Dans ce chapitre, nous allons répondre à la question de la détection du mouvement dans des scènes filmées avec une caméra fixe par des méthodes d'apprentissage non-supervisé. Nous dressons un tour d'horizon des méthodes et des algorithmes existants et nous introduisons les modèles à mélange de lois Gaussiennes.

Nous présentons par la suite la méthode de détection que nous avons mise en œuvre pour les zones mobiles ainsi que les résultats obtenus en fin de ce chapitre.

2.1 Généralités

La détection des régions animées de mouvement dans les images prises d'une scène à différents instants de temps est indispensable pour un nombre important d'applications de disciplines diverses. On peut citer à titre d'exemple la surveillance vidéo [SEG00, LCK99, WADP97], la télédétection [BP02, CW96], le diagnostic médical [MBNGR03, TC99], les systèmes d'aide au conducteur [FCF03, WKD96] mais également la compression vidéo [PQJP05, PBP04]. Du fait de l'extrême diversité des applications nécessitant ce traitement, il existe pléthore de méthodes de détection.

A partir d'un ensemble d'images extraites d'une même scène à différents instants de temps, la détection de mouvement consiste à identifier dans la dernière image de la séquence, l'ensemble des pixels qui sont suffisamment différents entre la dernière image de la séquence -l'image actuelle- et les précédentes. L'image comprenant tous les pixels en mouvement est appelée "masque de mouvement". Le masque de mouvement ainsi obtenu dépend de l'apparition ou la disparition d'objets, du mouvement d'objet par rapport au fond, ou encore de la déformation du contour d'objets dans la scène. De plus les objets

stationnaires peuvent également changer de couleur, de texture, etc. Le masque de mouvement ne doit pas cependant contenir d'objets en mouvement inintéressant -"parasites"- ou de bruit -caméra, variation de luminosité, absorption atmosphérique,...-.

Appliquée au problème de surveillance, la détection de mouvement peut simplement permettre d'éliminer les zones non intéressantes au sens du mouvement mais également de détecter des intrusions à certains endroits bien spécifiques de la scène vidéo observée. Cette analyse doit se faire à deux niveaux : quantitativement, c'est à dire pouvoir détecter tous les objets en mouvement et qualitativement en proposant une détection la plus fine possible.

La détection de mouvement doit résoudre également les problèmes de recouvrement. Il existe plusieurs types de recouvrement de zones [WBPB95] :

- zone de recouvrement : zone où le fond a été recouvert par un ou des objet(s) en mouvement
- zone de découvrement : zone où le fond ou un objet a été découvert par un ou des objet(s) en mouvement
- zone de chevauchement : zone où plusieurs objets se recouvrent.

2.2 Éléments d'état de l'art

Au travers de cet état de l'art nous présenterons les méthodes de détection non-supervisée. Par conséquent nous ne décrirons pas les méthodes de pré-traitement qui peuvent également fournir des informations sur les changements de luminosité. Le lecteur toutefois intéressé par cet aspect pourra se référer à [RAAKR05] concernant les méthodes d'ajustement de l'intensité lumineuse, les filtrages homomorphiques...

On distingue dans la littérature deux grands types d'approches de résolution du problème de détection du mouvement :

- les méthodes basées sur les différences d'images, dites de "détection de changement",
- les méthodes basées sur les variations du signal de luminance ou de couleur au cours du temps.

2.2.1 Détection par différence inter-images

Le principe de ces méthodes consiste à effectuer une soustraction entre l'image à traiter et une image de référence, créée statistiquement ou non, sensée représenter le fond.

Différence entre deux images

En supposant que les régions qui différent entre deux instants de temps sont celles affectées d'un mouvement, le premier résultat simple à implémenter est celui du seuillage entre deux images successives aux instant $t - dt$ et t. En notant $I(x, y, t)$ l'intensité lumineuse à l'instant t du pixel de coordonnées (x, y) dans le plan image, la différence temporelle absolue $|FD|$ s'exprime par :

$$|FD(x, y, t)| = |\delta I_{t,t-dt}| = |I(x, y, t) - I(x, y, t - dt)| \quad (1)$$

L'image $|FD|$ (Frame Difference en anglais) ainsi obtenue est nulle en tout point identique entre les deux images, les points où un changement a pu être observé peut être récupéré par simple seuillage [JMA79, Jai81, YAT81]. Une telle méthode est malheureusement fortement dépendante du bruit présent à l'image du fait de la caméra par exemple. Le seuillage semble être la solution la plus communément adoptée afin que les observations non nulles ne soient que des objets en mouvement. Ainsi la décision sur la présence de mouvement dans le pixel donné est prise si $|FD(x, y, t)| >$ Seuil. Du coup, de la valeur du seuil dépend la qualité de la détection et s'il est facile de déterminer un seuil pour des scènes filmées où il n'y a que peu de variations des valeurs de luminosité, scène d'intérieur par exemple, il en va tout autrement pour les scènes à fortes variations.

Bien que cette méthode soit facilement implémentable et puisse répondre aisément aux contraintes temps-réel, la règle de décision par seuillage est fortement tributaire de la qualité originale des images. Un rehaussement du seuil éliminerait les fausses détections engendrées par le suite mais ceci au détriment de la qualité et de la finesse des zones détectées et inversement une réduction du seuil augmenterait les fausses détections tout en augmentant la qualité de la détection.

Certaines solutions existent néanmoins pour gérer ce problème non pas en travaillant directement sur chaque pixel de l'image mais par moyennage sur des blocs de pixels [PBP04]. Une autre solution consiste à choisir un seuil moins "abrupte" avec un test de Hinkley par exemple [Pla85], ou encore un test du χ^2 [AK93].

Différences cumulées

L'idée de l'approche par différence cumulée n'est plus d'utiliser deux images pour faire la segmentation mais de s'appuyer sur les n précédentes images afin de fournir le meilleur résultat possible.

Dans [IL98], les auteurs proposent d'analyser pour chaque pixel x de l'image I sa valeur minimale m et maximale M de l'intensité lumineuse ainsi que la plus grande différence inter images D. Ces valeurs sont initialisées pendant une période d'entraînement et sont ensuite mises à jour périodiquement lors du processus de détection à condition qu'un pixel donné n'ait pas été classé comme appartenant à la classe "objet en mouvement" pendant un certain

temps. Les pixels sont classés suivant un processus en 4 étapes : seuillage, élimination du bruit, filtrage morphologique et enfin détection des objets. Chaque pixel est classé comme "objet" suivant la règle de décision suivante

$$|m(x) - I(x)| > D(x) \text{ ou } |M(x) - I(x)| > D(x) \tag{2}$$

Les auteurs appliquent ensuite un filtre morphologique afin d'éliminer les détections isolées sources majoritaires de fausses détections.

D'autres techniques peuvent également se situer dans cette catégorie. Bien qu'elles soient majoritairement employées pour la télédétection, l'analyse des changements vectoriels -Change Vector Analysis- [Hal98, BP02] est utilisée pour la détection de changements d'états -forêt, corail, ...- sur des images multi-spectrales. Un vecteur caractéristique est généré pour chaque pixel de l'image considérant un nombre limité de spectres. Après accumulation de l'ensemble de ces vecteurs, on récupère la différence entre l'image actuelle et l'image cumulée afin d'obtenir l'image soustraite et ainsi déterminer l'évolution de champs, de forêts, etc.

D'autres travaux utilisent plutôt une image moyenne et effectuent une différence entre cette image et l'image actuelle. Cette méthode, à cheval entre méthode par différence et par image de référence, fut notamment proposée dans [Mec89, Mak96] puis repris par [FR97]. Cette méthode suppose que les pixels de fond sont ceux qui sont affichés le plus longtemps à l'image. Friedman propose donc de construire l'image de fond B de l'instant t en faisant la moyenne pour chaque pixel de la façon suivante

$$B(x, y, t) = \frac{1}{t} \sum_{t^*=1}^{t} I(x, y, t^*) \tag{3}$$

avec $I(x, y, t^*)$ la valeur de luminosité du pixel (x, y) de l'image de temps t^*. Cette image peut se calculer de façon incrémentale avec l'équation suivante

$$B(x, y, t) = \frac{t-1}{t} B(x, y, t-1) + \frac{1}{t} I(x, y, t) \tag{4}$$

La variance est également calculée de façon incrémentale et les objets en mouvement sont identifiés par seuillage sur la distance de Mahalanobis entre les pixels $I(x, y, t)$ et $B(x, y, t)$.

Comme le fait remarquer Friedman les problèmes de cette méthode viennent du fait des variations brutales ou répétitives auxquelles sont soumises les scènes d'extérieur par exemple. Pour cela, la solution suggérée consiste à utiliser une fenêtre temporelle et de sélectionner la contribution des images de fond suivant un poids de façon à converger réellement vers l'image de fond. L'équation suivante est alors adoptée

$$B(x, y, t) = (1 - \alpha) B(x, y, t-1) + \alpha I(x, y, t) \tag{5}$$

où le rapport α est une constante de temps indiquant la rapidité du processus. Les résultats obtenus sur des vidéos de télésurveillance déidés à l'analyse du trafic routier sont tout à fait satisfaisants. Il est intéressant de noter que cette méthode permet également de supprimer les ombres de l'image de référence. Le choix du paramètre est critique car un compromis entre l'occupation mémoire et la rapidité d'intégration de nouvelles valeurs n'appartenant pas au modèle doit être trouvé. La nécessité de régler correctement ce paramètre limite l'utilisation de ce type de méthodes.

Toutefois le problème majeur de cette approche vient des objets avec faible mouvement c'est à dire quand le temps de déplacement de l'objet est inférieur à α : les pixels de fond étant occultés pendant que l'objet bouge faiblement, cette méthode aura donc tendance à apprendre l'objet en mouvement comme appartenant au fond. Il apparaît selon la terminologie employée par Friedman des "fantômes"...

Détection par test de vraisemblance

Une alternative à la différence d'images pixel par pixel consiste à étudier localement des propriétés statistiques du signal d'intensité lumineuse. Certaines méthodes proposent ainsi non pas de travailler sur un pixel mais sur une fenêtre autour de ce point et d'utiliser un test de vraisemblance initialement exploité pour la segmentation d'images fixes [Yak76] puis étendu pour la détection des changements entre images [Nag78, HNR84]. Le principe est le suivant : pour un pixel donné, on suppose que la distribution des intensités lumineuses dans un voisinage local proche est la combinaison d'un bruit blanc Gaussien et d'un modèle a priori de la fonction d'intensité. On considère alors deux fenêtres de référence, A_1 et A_2, centrées toutes les deux sur le pixel de coordonnées (x_0, y_0) respectivement aux instants de temps t et $t + dt$. Deux hypothèses sont ensuite confrontées : la première, H_0 suppose que les distributions d'intensité lumineuse possèdent les mêmes caractéristiques, c'est à dire qu'il n'y a finalement aucune variation temporelle. La deuxième, H_1, suppose que ces mêmes distributions ont des caractéristiques différentes entre A_1 et A_2 ce qui permet de déterminer un changement temporel entre ces deux fenêtres.

A chacune de ces hypothèses est associée une fonction de vraisemblance qu'il convient de seuiller de façon appropriée afin de trancher entre l'une ou l'autre des hypothèses. De nombreux modèles de répartition des niveaux de gris ont été étudiés.

Dans [BL93] et [SJ89], l'intensité lumineuse de chaque fenêtre carrée de taille n est censée suivre le modèle suivant

$$I_f = \mu + \eta \tag{6}$$

En supposant donc un bruit, η, blanc, Gaussien et centré, la densité de probabilité pour chaque pixel p_i suit une loi normale $\mathcal{N}(\mu, \sigma^2)$. Les variables étant supposées indépendantes la fonction de vraisemblance L est en fait le produit des densités de probabilités :

$$L = \prod_{i=1}^{n^2} \left(\frac{1}{\sqrt{2\pi\sigma^2}} \right) e^{-\frac{(I(p_i)-\mu)^2}{2\sigma^2}} \tag{7}$$

Suivant les fenêtres étudiées les hypothèses retenues sont :

- H_0 : les distributions dans les fenêtres A_1 et A_2 suivent la même fonction de répartition $\mathcal{N}(\mu_0, \sigma_0^2)$,
- H_1 : les distributions dans A_1 et A_2 n'ont pas les mêmes caractéristiques,$\mathcal{N}(\mu_1, \sigma_1^2)$ pour la première fenêtre, $\mathcal{N}(\mu_2, \sigma_2^2)$ pour la seconde avec bien évidemment $\mu_1 \neq \mu_2$

Le rapport de vraisemblance est défini par :

$$R = \frac{\left(\frac{1}{\sqrt{2\pi\sigma_1^2}} \right)^{n^2} \prod_{p \in A_1} e^{-\frac{(I(p)-\mu_1)^2}{2\sigma_1^2}} \left(\frac{1}{\sqrt{2\pi\sigma_2^2}} \right)^{n^2} \prod_{p \in A_2} e^{-\frac{(I(p)-\mu_2)^2}{2\sigma_2^2}}}{\left(\frac{1}{\sqrt{2\pi\sigma_0^2}} \right)^{2n^2} \prod_{p \in A_1 \cup A_2} e^{-\frac{(I(p)-\mu_0)^2}{2\sigma_0^2}}} \tag{8}$$

Le critère ξ est obtenu après passage au logarithme de R. En supposant que les variances sont identiques et représentent celle du bruit, $\sigma_0 = \sigma_1 = \sigma_2 = \sigma$, on obtient

$$\xi = lnR = \frac{1}{2\sigma^2} \left(\sum_{p \in A_1 \cup A_2} (I(p) - \mu_0)^2 - \sum_{p \in A_1} (I(p) - \mu_1)^2 - \sum_{p \in A_2} (I(p) - \mu_2)^2 \right) \tag{9}$$

Il s'agit donc de maximiser cette expression. Dès lors il nous est possible de déterminer les paramètres des lois normales en récupérant les valeurs qui annulent les dérivées de ξ suivant μ_i, $\forall i \in \{0, 1, 2\}$. Ainsi on obtient les paramètres :

$$\begin{cases} \hat{\mu_0} &= \sum_{p \in A_1 \cup A_2} I(p)/2n^2 \\ \hat{\mu_1} &= \sum_{p \in A_1} I(p)/n^2 \\ \hat{\mu_2} &= \sum_{p \in A_2} I(p)/n^2 \end{cases} \tag{10}$$

Après développement et en considérant la racine carrée de ce critère, la décision au point $p_0(x_0, y_0)$ devient donc finalement [Lal90] :

$$\sqrt{\xi} = \frac{n}{2\sigma} |\hat{\mu_1} - \hat{\mu_2}| = \frac{1}{2n\sigma} \left| \sum_{i=1}^{n^2} \left(\overbrace{I_{A_1}(p_i) - I_{A_2}(p_i)}^{FD(p_i)} \right) \right| \begin{array}{c} H_1 \\ > \\ < \\ H_0 \end{array} \lambda \tag{11}$$

avec $I_{A_1}(p_i)$ et $I_{A_2}(p_i)$ l'intensité du point de coordonnées (x_i, y_i), des fenêtres A_1 et A_2 aux instant t et $t + dt$ respectivement.

Ce rapport peut être majoré pour une détection plus robuste (moins de fausses détections) [PBP04] par

$$\sqrt{\xi} = \frac{1}{2n\sigma} \left| \sum_{i=1}^{n^2} (I_{A_1}(p_i) - I_{A_2}(p_i)) \right| < \frac{1}{2n\sigma} \sum_{i=1}^{n^2} |I_{A_1}(p_i) - I_{A_2}(p_i)| \underset{<}{\overset{H_1}{>}} \lambda \quad H_0 \tag{12}$$

D'un point de vue empirique, il s'avère que cette méthode est beaucoup moins tributaire du bruit de l'image tout en offrant de bons taux de détection. Suivant le même principe, des modélisations plus complexes (linéaire, quadratique,...) ont donc été étudiées en vue d'accroître la sensibilité de la détection. Comme précédemment quel que soit le modèle adopté, les rapports de vraisemblance doivent être déterminés puis seuillés afin de conclure en faveur d'une des deux hypothèses H_0 ou H_1 [HNR84].

Dans le cas d'une modélisation linéaire, l'intensité lumineuse de l'image est représentée par un modèle polynomial de 1er ordre en x et y :

$$I_f = \beta_1 + \beta_2 \Delta x + \beta_3 \Delta y + bruit \tag{13}$$

Cette modélisation linéaire semble plus réaliste car elle autorise une variation continue de la luminance dans la fenêtre considérée mais peut s'avérer toutefois insuffisante s'il existe un fort gradient de luminance. Dans ce cas le modèle quadratique, plus complexe, s'avère également plus avantageux. Ce dernier s'exprime de la façon suivante [Lal90] :

$$I_f = \beta_1 + \beta_2 \Delta x + \beta_3 \Delta y + \beta_4 \Delta x^2 + \beta_5 \Delta y^2 + \beta_6 \Delta x \Delta y + bruit \tag{14}$$

Ceci dit l'augmentation sensible des résultats de détection obtenus avec des modèles plus complexes se paie au détriment d'un taux de précision car ces modèles sont également beaucoup plus sensibles aux parasites. A cela s'ajoute une complexité en temps de calcul nécessaire à la fois pour la détection en elle même et pour la suppression des détections "parasites". Un bon compromis semble par conséquent être finalement le modèle linéaire [Lal90, Let93].

2.2.2 Détection avec prise en compte d'un contexte spatial

Dans l'optique de proposer des masques de mouvement toujours plus propres, c'est à dire privés le plus possible des détections parasites rendant souvent leur interprétation délicate voire impossible, de nombreuses études se sont concentrées sur des méthodes de traitement spatial des masques de mouvement en vue de les rendre plus homogènes.

La première méthode envisagée réalise des opérations de dilatation/érosion ou bien encore de proposer un certain nombre d'heuristiques. Dans [Die91], les auteurs proposent notamment d'effectuer un filtrage médian sur une fenêtre carrée de taille $5*5$ centrée sur le pixel considéré. Cette méthode permet d'effacer une grande partie des éléments parasites dans les zones de mouvement détectées. Il existe cependant des méthodes plus "élégantes" consistant, à partir de la connaissance a priori pertinente d'une solution recherchée (homogénéité spatiale et temporelle des masques par exemple), à présenter le problème de la détection comme un problème statistique d'étiquetage suivant un modèle en classes spatialement homogènes.

Des modèles complexes ont ainsi été proposés par Lalande et Bouthémy [BL93], Sivan et Malah [SM94], Odobez et Bouthémy [OB94], Caplier et Luthon [CLD98], ... Toutes ces méthodes emploient des champs aléatoires de Markov spatio-temporels et construisent une fonction de coût qu'il convient de minimiser par des algorithmes déterministes de relaxation. Les principales différences entre ces travaux proviennent des choix de la fonction de coût et de l'algorithme de relaxation. Dans [SM94], par exemple, les auteurs posent le problème de détection comme appartenant à l'un des quatre états -configurations- possibles : $\{\text{Texturé}, \text{Non texturé}\} \times \{\text{Statique}, \text{Mobile}\}$. Considérant une approche multi-résolution à deux niveaux, l'algorithme ICM (Iterated Conditional Modes) [Bes74] est utilisé au dépens du traditionnel algorithme de recuit simulé -moins rapide- afin d'obtenir un minimum de la fonction de coût à chaque niveau. Dans [OB94], la détection de mouvement est réalisée par une approche de régularisation statistique où une attention particulière est accordée à la définition de la fonction d'énergie rendant, par ailleurs, cette modélisation complexe et quelque peu coûteuse en calcul. Le schéma de multi-résolution propose de régler les problèmes des caméras mobiles.

Dans [LB90], les auteurs proposent un cadre de travail original de détection de mouvement dans une séquence d'images consécutives. Suivant un contexte statistique de type champs de Markov spatio-temporels, les auteurs modélisent le problème suivant une formulation bayésienne. Sachant que cette modélisation, après avoir été employée pour des problèmes de segmentation des images texturées [MDZ97] a connu un très vif succès dans le cadre des images animées, nous allons l'exposer plus en détail.

Un champ de Markov est un ensemble de variables aléatoires, défini sur un graphe où la probabilité de chaque variable ne dépend que de l'état de ses voisins. Formulons le problème comme suit :

- Soit $E = \{e_s, s \in \mathcal{S}\}$ le champ de variables aléatoires ou autrement dit étiquettes, défini sur le domaine S.
- Soit $\mathcal{S} = s_1, ..., s_N$ une grille de N éléments, autrement appelés sites, constituant le support de E,
- Soit $\mathcal{L}$ l'ensemble des valeurs possibles pour chaque étiquette e_{s_i} et Ω l'ensemble de toutes les configurations possibles pour les champ des étiquettes : $\Omega = \{e = (e_{s_1}, ..., e_{s_N}) | e_{s_i} \in \mathcal{L}\}$

- Notons à partir de maintenant la réalisation particulière du champ d'étiquette $\{E_{s_1} = e_{s_1}, ..., E_{s_N} = e_{s_N}\}$ par $\{E = e\}$.
- Soit le système de voisinage $\mathcal{V} = \{v_s, s \in \mathcal{S}\}$ défini sur l'ensemble des sites, avec v_s le voisinage du site s. Ce système doit vérifier les conditions suivantes,
  - $s \notin v_s$
  - $r \in v_s \iff s \in v_r$
- Soit $\mathcal{C}$ l'ensemble des cliques de $\mathcal{S}$ associés au système de voisinage $\mathcal{V}$. Une clique est un sous ensemble de sites (pixels dans notre cas) qui sont mutuellement voisins. Généralement nous ne considérons que des systèmes de voisinage d'ordre 1 ou 2.

Le champ E associé au système de voisinage $\mathcal{V}$ et à la fonction de probabilité p est un champ de Markov s'il vérifie :

$$\forall e \in \Omega, \quad p(E = e) > 0, \tag{15}$$

$$\forall s \in \mathcal{S}, \quad p(E_s = e_s | E_r = e_r, r \in \mathcal{S} \text{ et } r \neq s) = p(E_s = e_s | E_r = e_r, r \in v_s) \tag{16}$$

L'équation (15), dite "condition de positivité", indique que toute configuration du champ d'étiquettes est réalisable. L'équation (16) exprime quant à elle le fait que l'étiquette en chaque site, connaissant toutes les autres, ne dépend finalement que du contexte du voisinage local en ce site.

Le résultat fondamental a été établi par Hammersley et Clifford [HC71]. Il établit l'équivalence entre les champs de Markov et les distributions de Gibbs en définissant la probabilité p du champ E. Elle s'écrit de la façon suivante :

$$p(E = e) = \frac{1}{Z_c} exp(-U_1(e)) \tag{17}$$

avec Z_c une constante de normalisation définie de la façon suivante :

$$Z_c = \sum_{e \in \Omega} exp(-U_1(e)) \tag{18}$$

$U_1(e)$ est appelée fonction d'énergie qui se décompose comme la somme de potentiels locaux

$$U_1(e) = \sum_{c \in \mathcal{C}} V_c(e) \tag{19}$$

L'objectif consiste finalement à estimer, à partir d'un ensemble d'observations O, un ensemble d'étiquettes E ayant des propriétés attendues. Bien que les étiquettes et les observations soient différentes d'un problème à l'autre, l'approche bayésienne reste souvent une approche élégante pour la formulation des relations entre observations et étiquettes

à estimer [Bes74, GG84]. Une réalisation $\hat{e}$ des étiquettes doit être dès lors estimée de probabilité maximale conditionnellement aux observations. Cette relation se réécrit suivant le théorème de Bayes :

$$p(E = e|O = o) = \frac{p(O = o|E = e) \times p(E = e)}{p(O = o)} \tag{20}$$

De l'équation (20) on remarque que le dénominateur ne dépend pas des étiquettes, ce qui revient à dire que maximiser le critère revient à maximiser la loi conjointe

$$\hat{e} = \arg\max_{e} p(O = o|E = e) \times p(E = e) \tag{21}$$

Cet estimateur, dit Maximum A Posteriori -MAP-, est le critère le plus souvent exprimé mais il n'est pas le seul. Une présentation des autres critères est détaillée dans [Ric87]. Finalement, en supposant une définition au sens du MAP, il est impératif connaissant la probabilité du champ E, $p(E = e)$, de définir la probabilité conditionnelle a posteriori des observations $p(O = o|E = e)$. Il est donc pour cela nécessaire de modéliser la relation qui lie les étiquettes du champ E et les mesures données par les observations du champ O. Cette relation se modélise différemment suivant la problématique abordé.

Dans notre étude nous allons nous attacher à la maximisation de la relation suivante

$$p(O = o|E = e) \times p(E = e) = \frac{1}{Z} exp^{-U(e,o)} \tag{22}$$

avec

- Z une constante de normalisation,
- $U(e, o) = U_1(e) + U_2(e, o)$ une fonction d'énergie totale,
- $U_1(e) = \sum_{c \in \mathcal{C}} V_c(e)$ la connaissance a priori supposée sur la répartition des étiquettes du champ, et enfin
- $U_2(e, o) = \sum_{s \in \mathcal{S}} V_o(e, o)$ le terme d'énergie lié à la vraisemblance des observations.

Au vu de la relation (22), la configuration qui maximise, au sens du MAP, la configuration des étiquettes la plus probable est celle qui minimise la fonction d'énergie totale :

$$\hat{e} = \arg\min_{e \in \Omega} \left(U_1(e) + U_2(e, o)\right) \tag{23}$$

De cette façon, et grâce à $V_c(e)$ et à $V_o(e, o)$ respectivement modèle a priori et terme d'attache aux données, ce critère global ne s'obtient finalement qu'en effectuant des calculs locaux. Le choix de ces potentiels locaux est donc d'une importance capitale car il conditionne les propriétés du champ E et la formulation de la relation entre E et les observations O.

Étant donné un ensemble de mesures, il s'agit de trouver la configuration des étiquettes traduisant au mieux l'adéquation aux données et aux propriétés du champ de Markov. En reprenant la formulation markovienne, les auteurs de [LB90] choisissent comme "observation" la dérivée temporelle de la fonction d'intensité d'un pixel donné.

Soit o_t l'observation au temps t, et $p = p(x, y)$ un pixel, alors $o_t(p) = \frac{\partial f(p)}{\partial t}$. En fait, ces dérivées temporelles peuvent être approximées par une différence finie entre les temps t et $t - dt$ telle que

$$o_t(p) = \tilde{f}_t(p) = FD(x, y, t) = f(x, y, t) - f(x, y, t - dt) \tag{24}$$

Les auteurs suggèrent d'ajouter à cette observation une autre information concernant le changement temporel effectif du pixel p, $\overline{o}_t(p)$. Ainsi $\overline{o}_t(p)$ prend la valeur 1 si un changement est réellement observé, 0 sinon.

A partir de ces observations, il est ensuite nécessaire de créer une information contextuelle entre la carte d'étiquettes d'instant t, $e_t \in \{a, b\}$ (où a correspond à l'objet en mouvement et b pour le fond), et celle d'instant $t - dt$. La solution est formulée suivant la maximisation de la probabilité conjointe ξ entre étiquettes et observations, elle même explicitée par l'équivalence entre les champs de Markov et les distributions de Gibbs :

$$\xi(e_t - dt, e_t, o_t, \overline{o}_t) = \frac{1}{Z} exp\left[-W(e_t - dt, e_t, o_t, \overline{o}_t)\right] \tag{25}$$

avec ξ la probabilité conjointe entre étiquettes et observations, Z une constante de normalisation et W une fonction d'énergie définie par $W = \sum_{c \in C} V_c$. Dans l'équation précédente C est l'ensemble des cliques suivant un système de voisinage décrivant les interactions entre les différentes variables et V_c une fonction de potentiels définie localement sur la clique. Maximiser la probabilité (25) revient donc à minimiser l'énergie W dont l'expression est donnée par :

$$W(e_t - dt, e_t, o_t, \overline{o}_t) = W_s(e_{t-dt}) + W_\gamma(e_{t-dt}, e_t, \overline{o}_t) + W_e(e_{t-dt}, e_t, o_t) + W_s(e_t) \tag{26}$$

avec

- W_s est la fonction d'énergie définissant les cartes spatialement attendues. W_s vaut β_s si $e_t = e_s$ avec $< s, t >$ une clique binaire, $-\beta_s$ sinon. Ce terme répond de la régularité spatiale des cartes.
- W_γ définit quelles configurations sont encouragées parmi les 2^3 possibles : $(a, a, 0), (a, a, 1), (a, b, 0), ...$ [LB90] donne l'ensemble des configurations favorisées. Ce terme défavorise les changements intempestifs des étiquettes au fil du temps et donc répond de la régularisation temporelle.

– W_e définit la consistance entre les observations et les estimations courantes des étiquettes. W_e s'écrit de la façon suivante $W_e = \frac{1}{2\sigma^2} \sum_p \left[\tilde{f}_t(p) - \psi(e_{t-dt}(p), e_t(p)) \right]$ avec ψ prenant les valeurs 0 pour (b, b), m_1 pour (a, a) et m_2 dans tous les autres cas possibles.

Ainsi $U_1(e) = W_s(e_{t-dt}) + W_s(e_t)$ et $U_2(e, o) = W_\gamma(e_{t-dt}, e_t, \overline{o}_t) + W_e(e_{t-dt}, e_t, o_t)$

Dans [BL93] aussi bien que dans [LB90], la modélisation markovienne est utilisée pour la détection d'objets en mouvement. A partir de trois images successives, la solution est dérivée par minimisation d'une fonction d'énergie, grâce à un schéma déterministe et itératif de relaxation, indépendemment de la taille, de la distribution du mouvement et de la direction des objets. L'idée principale de cette approche est de considérer trois images successives à des instants de temps différents t_1, t_2 et t_3 afin de déterminer la localisation des objets en mouvement au temps t_2. Deux cartes binaires de détection, entre t_1 et t_2, et t_2 et t_3 sont déterminées et un ET logique est opéré sur les deux cartes de mouvement. Cette méthode a le mérite de traiter les zones de mouvement à faible amplitude entre deux images mais ne gère pas le phénomène d'occultations entre ces images.

Dans [CLD98], les auteurs présentent un algorithme de détection de mouvement dans les séquences d'images acquises avec une caméra fixe (étiquetage binaire de l'image en pixels fixes ou mobiles). L'approche est basée sur une modélisation multi-échelle des interactions spatio-temporelles entre étiquettes par un champ de Markov également faisant intervenir trois images successives. Ce modèle permet d'intégrer la prise en compte du contexte spatial de chaque point pour produire une carte non bruitée. Les liens contextuels temporels avec les cartes à des instants de temps différents permettent de localiser des objets mobiles et de diminuer les zones d'écho. Le modèle proposé s'étend au cas du multi-étiquetage moyennant une modification de la stratégie d'initialisation des champs d'étiquettes et de la stratégie de relaxation. Ceci permet de faire non plus simplement une détection binaire des zones mobiles mais une détection multi-étiquette non supervisée (discrimination des divers objets mobiles et estimation de leur nombre) en plus d'un suivi court-terme des objets mobiles grâce à un filtrage de Kalman de la trajectoire du centre de gravité des régions détectées. Ce travail a ensuite été complété par le développement d'une architecture matérielle pour un traitement temps-réel.

Ces méthodes sont, d'un point de vue qualitatif, très intéressantes mais la formulation retenue semble pour le moins coûteuse pour une application temps-réel sans optimisation matérielle. L'étude proposée ne permet pas non plus de connaître les valeurs de la relation entre étiquettes et observations. Cette valeur conditionne bien évidemment les résultats obtenus.

2.2.3 Méthodes basées sur les variations du signal de luminance

Recours à une image de référence

L'utilisation d'une image de référence fait appel à la représentation de la scène privée de toute activité de mouvement ce qui rend la localisation des objets en mouvement immédiate. Cette technique a le bénéfice d'éliminer le phénomène d'écho comme c'est le cas souvent avec les méthodes par différence inter-images.

Ces méthodes ont pour avantage que le choix du seuil entre l'image actuelle et l'image de référence ne soit pas aussi critique que pour les méthodes par différences. Ceci dit ces méthodes sont tributaires du faible changement de luminosité de la scène. Si les conditions d'éclairage viennent à changer brutalement ces méthodes seraient moins efficaces : il est donc indispensable que l'image de référence soit réactualisée spécifiquement.

Filtrage de Kalman

L'extraction de l'image de référence et donc des objets en mouvement peut se faire automatiquement par filtrage. Dans [KB89, Kil92, KWH+94, DKM94], l'image de fond est extraite par un formalisme proche du filtrage de Kalman [Kal60] ce qui permet de régler le problème des variations de conditions climatiques et des variations de luminosité suivant l'heure de la journée.

L'image de fond B est ensuite mise à jour suivant l'équation suivante :

$$B(x,y,t) = B(x,y,t-1) + (\alpha_1(1 - M(x,y,t)) + \alpha_2 M(x,y,t-1))D(x,y,t) \qquad (27)$$

où $B(x,y,t)$ est la valeur d'intensité de l'image de référence aux coordonnées (x,y) et à l'instant t, $D(x,y,t)$ correspond à différence entre les pixels de coordonnées (x,y) de l'image actuelle et de l'image de référence et $M(x,y,t)$ le pixel appartenant au masque binaire de mouvement obtenu par seuillage de l'image des différences entre B et l'image actuelle. Ainsi $M(x,y,t) = 1$ si (x,y) est un pixel de l'objet et 0 sinon. Les gains α_1 et α_2 sont des constantes décrivant les taux d'apprentissage. Ces valeurs ne sont pas fixées a priori, mais estimées par le filtre de Kalman considéré. La détection à proprement parler est réalisée par simple différence de chaque valeur d'intensité correspondant à un pixel entre la nouvelle image à segmenter et l'image de fond B obtenue. Comme le fait remarquer Kaewtrakulpong dans [KB01], cette méthode est incapable de gérer le problème d'objets entrant ou sortant dans la scène.

Approche paramétrique et non paramétrique

Une autre catégorie de méthodes possibles pour résoudre le problème de détection de mouvement consiste à approximer la fonction de densité de probabilité des valeurs RGB,

YUV, ou plus simplement de luminance d'un pixel de fond par un modèle paramétrique ou non-paramétrique à un instant t puis de confronter chaque nouveau pixel, à $t+1$, au modèle existant en vue de sa classification.

En supposant que les données observées soient des réalisations d'un vecteur aléatoire X de loi inconnue p, l'objectif consiste à reconstituer p à partir de ses réalisations. Pour cela, on définit un ensemble de lois possibles, $\{p_\theta, \theta \in \Theta\}$ de forme fixée a priori -ex. lois Gaussiennes- tout en supposant qu'il existe un $\theta^* \in \Theta$ tel que $p = p_{\theta^*}$. Il s'agit donc d'estimer $p_{\hat{\Theta}(X)}$ tel que $p = p_{\hat{\Theta}(X)}$.

On distingue dès lors deux cas de figure : soit Θ est inclus dans un espace euclidien, on parle alors d'approche "paramétrique", soit elle est qualifiée de "non paramétrique". Dans les parties suivantes, nous allons tout d'abord présenter les approches non-paramétriques dites par noyau puis nous reviendrons sur celles dites paramétriques pour étudier plus en détail l'algorithme dit d'Expectation Maximization.

Approche non-paramétrique : méthode par noyaux

Les approches d'estimation non-paramétrées ont connu récemment un essor important notamment grâce aux méthodes d'estimation par noyaux -Kernel Density Estimation [DHS00, WJ95]. Ces méthodes sont connues pour leur capacité à estimer de façon "lisse", continue et dérivable n'importe quelle distribution d'autant que le nombre de modes est déterminé automatiquement.

Supposons l'existence d'un échantillon de n données, $x_i \, \forall i \in \{1, .., n\}$, suivant une fonction de densité de probabilité multi-variée $P(x)$, une estimation de cette densité pour x peut se formuler de la façon suivante :

$$\hat{p}(x_t) = \frac{1}{N} \sum_{i=1}^{N} K(x_t - x_i) \,, \tag{28}$$

où K es une fonction noyau. Dans [EHD99], les auteurs choisissent un noyau Gaussien pour un contexte de vidéo surveillance, $\mathcal{N}(0, \Sigma)$ avec Σ la matrice de covariance des canaux couleurs utilisés. La densité à estimer se réécrit dès lors

$$\hat{p}(x) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{2\pi^{\frac{d}{2}} |\Sigma|^{\frac{1}{2}}} e^{-\frac{d}{2}(x_t - x_i)^T \Sigma^{-1} (x_t - x_i)} \tag{29}$$

avec d le nombre de composantes couleur retenues.

Si l'on suppose que les valeurs des composantes de couleur sont statistiquement indépendants du noyau choisi et en dénotant σ_j^2 la variance de la $j$ème couleur alors on a

$$\Sigma = \begin{pmatrix} \sigma_1^2 & 0 & . & . & . & 0 \\ 0 & \sigma_2^2 & 0 & . & . & . \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ . & . & . & . & . & 0 \\ 0 & . & . & . & 0 & \sigma_d^2 \end{pmatrix} \tag{30}$$

et la fonction de densité peut se réduire à

$$\hat{p}(x_t) = \frac{1}{N} \sum_{i=1}^{N} \prod_{j=1}^{d} \frac{1}{\sqrt{2\pi\sigma_j^2}} e^{\frac{-(x_{tj}-x_{ij})^2}{2\sigma_j^2}} \tag{31}$$

En général, une valeur $d = 3$ est retenue pour modéliser les différents composantes couleurs RGB, YUV,... La fonction $\hat{P}(x)$ est ensuite seuillée suivant une constante a priori th permettant d'estimer ainsi le modèle de fond, la classification s'opérant comme d'habitude par différence entre le pixel courant à estimer et le pixel de fond. Ces travaux ont été ensuite repris dans [YDGD03] et dans [AED03].

Approche paramétrique : Modèle de Mélange

La densité de probabilité $p(x)$ de la variable x étant inconnue, le principe d'un modèle de mélange consiste à décomposer cette densité en une somme de K composantes $p(\bullet|\theta_k)$ correspondant aux K classes desquelles on cherche à estimer les paramètres $\theta_k, k \in \{1, ..., K\}$ à partir de l'échantillon $X = \{x_1, ..., x_n\}$.

La nature des densités de probabilité $p(\bullet|\theta_k)$ en terme de modalité peut être tout à fait diverse. Les proportions w_k, également appelées *poids* entre les différentes composantes, représentent les probabilités a priori des différentes classes. Inconnues ces valeurs de proportions sont à estimer tout en tenant compte des contraintes suivantes :

$$w_k \in]0; 1[\text{ et } \sum_{k=1}^{K} w_k = 1 \tag{32}$$

Notons $\Theta = [w_1, ..., w_k, \theta_1, ..., \theta_k]$ le vecteur de paramètres à estimer. La densité de probabilité $p(x)$ de x est approximée conditionnellement aux paramètres Θ par :

$$p(x|\Theta) = \sum_{k=1}^{K} w_k p(x|\theta_k) \tag{33}$$

Dans le cas de la classification supervisée, l'information supplémentaire indiquée par l'appartenance des points aux classes, permet l'estimation directe des paramètres Θ du modèle à savoir les poids des classes ainsi que les paramètres θ_k des classes.

En supposant que le modèle Θ soit identifiable (c'est à dire que pour deux valeurs distinctes θ_1 et θ_2 du paramètre Θ, les densités $p(x|\theta_1)$ et $p(x|\theta_2)$ soient également distinctes) nous allons nous pencher sur l'étude de l'algorithme dit "Expectation-Maximization".

Algorithme Expectation-Maximization (EM)

L'algorithme Expectation-Maximization (EM) est une technique itérative de maximisation de la fonction de vraisemblance en présence de données incomplètes [DLR77]. Il est en particulier applicable pour les problèmes où les observations ne fournissent qu'une information partielle ou encore quand des données sont manquantes. Cet algorithme est surtout retenu dans la littérature (ce pour quoi nous lui consacrons une description détaillée) du fait qu'il est applicable pour l'estimation de paramètres pour de nombreux modèles : mélange d'experts -Mixture of experts M.O.E-, les mélanges de lois Gaussiennes -Gaussian Mixture Model G.M.M- ou encore la quantification vectorielle -Vector Quantization V.Q.-.

L'algorithme dit "classique" présenté ici et proposé pour la première fois par [DLR77] en 1977 peut être assimilé à une sorte d'algorithme de Newton du fait de la recherche de la direction ayant une projection positive sur le gradient de la log-vraisemblance. Il est découpé en deux étapes : "Expectation"-E- et "Maximization" -M-. L'étape dite "M" maximise la fonction de vraisemblance qui est raffinée à chaque itération par l'étape "E".

"EM" a démontré des propriétés de convergence très favorables : satisfaction automatique aux contraintes et rapidité de convergence sont autant d'atouts pour cet algorithme. Notons néanmoins que cet algorithme n'est pas l'unique méthode d'estimation des paramètres, et diverses approches ont été proposées : citons à titre d'exemple la méthode des moments de Pearson [Pea94].

Retenons la formulation suivante soient :

- $X = \{x_t \in \mathbb{R}^m; t = 1, ..., T\}$ une séquence d'observations avec T le nombre d'observations réellement présentes et m la dimension de x_t,
- $C = \{C_1, ..., C_K\}$ l'ensemble des étiquettes possibles pour les groupes formés par le mélange avec K le nombre de composantes du mélange,
- $Z = \{z_t \in C; t = 1, ..., T\}$ l'ensemble des données manquantes,
- $\Theta = \{w_1, ..., w_k, \theta_1, ..., \theta_k\}$ l'ensemble des paramètres inconnus défini pour approximer la vraie densité de probabilité de X avec w_i les poids du ième composant du mélange et θ_i l'ensemble de ses paramètres à estimer. Dans le cas du mélange de lois Gaussiennes, ces paramètres sont la moyenne, l'écart type par exemple.

Notons tout d'abord que la combinaison des observations X et des données cachées Z constitue l'ensemble des données complètes des observations. Afin de faciliter la dérivation définissons

$$L(X|\Theta_n) \equiv \log p(X|\Theta_n) \tag{34}$$

la log-vraisemblance des données incomplètes suivant l'estimation courante Θ_n, où n représente le nombre d'itération. Ainsi défini, notons $p(Z, X|\Theta_n)$ la vraisemblance des données complétées. Suivant le théorème de Bayes et en reprenant la notation usuelle en probabilité [1] nous avons :

$$p(X|\Theta_n) = \frac{p(Z, X|\Theta_n)}{P(Z|X, \Theta_n)} \tag{35}$$

En utilisant les équations (34) et (35), la vraisemblance des données incomplètes peut être formulée de la façon suivante :

$$\begin{aligned} L(X|\Theta_n) &\equiv \log p(X|\Theta_n) \\ &= \log p(X|\Theta_n) \sum_Z P(Z|X, \Theta_n) \quad \text{puisque } \sum_Z P(Z|X, \Theta_n) = 1 \\ &= \sum_Z P(Z|X, \Theta_n) \log p(X|\Theta_n) \\ &= \sum_Z P(Z|X, \Theta_n) \log \frac{p(Z, X|\Theta_n)}{P(Z|X, \Theta_n)} \quad \text{du fait de l'équation (35)} \\ &= \sum_Z P(Z|X, \Theta_n) \log p(Z, X|\Theta_n) - \sum_Z P(Z|X, \Theta_n) \log P(Z|X, \Theta_n) \end{aligned} \tag{36}$$

Ce qui, en faisant introduire la notion d'espérance E_Z conditionnellement à Z, permet d'écrire l'équation précédente de la façon suivante :

$$\begin{aligned} L(X|\Theta_n) &= E_Z\{\log p(Z, X|\Theta_n)|X, \Theta_n\} - E_Z\{logP(Z|X, \Theta_n)|X, \Theta_n\} && (37) \\ &= Q(\Theta|\Theta_n) + R(\Theta|\Theta_n) && (38) \\ & && (39) \end{aligned}$$

avec $Q(\Theta|\Theta_n) \equiv E_Z\{\log p(Z, X|\Theta)|X, \Theta_n\}$ et $R(\Theta|\Theta_n) \equiv -E_Z\{logP(Z|X, \Theta)|X, \Theta_n\}$. Le terme $R(\Theta|\Theta_n)$ est appelé entropie et représente la différence entre la vraisemblance des données incomplètes et l'espérance de la vraisemblance des données complétées.

Nous ne rentrerons pas dans le développement de la convergence d'algorithme, elle pourra être étudié plus en détail dans [DLR77].

A l'étape "M" de la $n^{\text{iéme}}$ itération l'algorithme sélectionne un Θ^* par la relation suivante :

[1] ci après P désigne la probabilité, p la fonction de densité

$$\Theta^* = \arg\max_{\Theta} Q(\Theta|\Theta_n)$$

ce qui signifie qu'à chaque itération n on doit pouvoir trouver un Θ^* tel que

$$Q(\Theta^*|\Theta_n) \geq Q(\Theta_n|\Theta_n)$$

L'équation précédente est une condition suffisante pour assurer la convergence de l'algorithme EM car

$$\begin{aligned} L(X|\Theta^*) &\geq Q(\Theta^*|\Theta_n) + R(\Theta_n|\Theta_n) \\ &\geq Q(\Theta_n|\Theta_n) + R(\Theta_n|\Theta_n) \\ &= L(X|\Theta_n) \end{aligned}$$

par définition Θ^* nous avons $R(\Theta^*|\Theta_n) \geq R(\Theta_n|\Theta_n)$. Plutôt que de maximiser directement $L(X|\Theta)$, l'algorithme EM divise les opérations en deux sous problèmes : "Expectation" et "Maximization". Pour chaque itération l'étape "Expectation" calcule $Q(\Theta|\Theta_n)$ grâce à un ensemble de paramètres présumés Θ_n. L'étape "Maximization" quant à elle détermine la valeur Θ^* qui maximise $Q(\Theta|\Theta^n)$ via

$$\Theta^* = \max_{\Theta} \sum_{Z} P(Z|X, \Theta_n) \log p(Z, X|\Theta)$$

ce qui se détermine par

$$p(Z, X|\Theta^*) = \frac{P(Z|X, \Theta_n)}{\sum_Z P(Z|X, \Theta_n)} \tag{40}$$

Diviser le problème en deux sous-problèmes interdépendants est beaucoup plus pratique : maximiser $Q(\Theta|\Theta_n)$ est plus facile que $L(X|\Theta_n)$.

Dans notre explication nous n'avons proposé ici que la méthode traditionnelle de dérivation permettant d'aboutir à la solution, mais nous avons omis volontairement de présenter comment calculer réellement la valeur de $Q(\Theta|\Theta_n)$ à l'étape "E" et comment la maximiser à l'étape "M".

Notons que dans le cas où Θ^* est difficile à obtenir l'approche EM reste valide si l'on arrive tout de même à "améliorer" $Q(\Theta|\Theta_n)$ pour chaque étape "M", ce qui revient à une descente de gradient classique. L'algorithme est alors appelé EM généralisé et aboutit, malgré une vitesse de convergence beaucoup plus lente, à la solution.

L'algorithme EM commence par une optimisation de la fonction de vraisemblance qui peut être considérablement simplifiée si un ensemble de données "cachées" ou "manquantes"

est supposé connu. Si $X = \{x_t; t = 1, ..., T\}$ contient T vecteurs statistiquement indépendants et $Z = \{z_t \in C; t = 1, ..., T\}$, où $z_t = C_j$ signifie que la j-ième mixture génère x_t, alors on peut écrire

$$p(Z, X|\Theta) = \prod_{t=1}^{T} p(z_t, x_t|\Theta) \tag{41}$$

Introduisons un ensemble de variables d'indications qui indique le statut des états cachés

$$\Delta = \{\delta_t^k; k = 1, ..., K \ \ et \ \ t = 1, ..., T\}$$

tel que

$$\delta_t^k \equiv \delta(z_t, C_k)$$

autrement dit que δ_t^k vaut 1 si x_t est généré par le mélange étiqueté par C_k, 0 sinon. En remarquant que :

$$\forall t \in \{1, ..., T\}, \sum_{k=1}^{K} \delta_t^k = 1$$

nous avons donc

$$\begin{aligned} p(Z, X|\Theta) &= \prod_{t=1}^{T} \sum_{k=1}^{K} \delta_t^k p(x_t, z_t|\Theta) \\ &= \prod_{t=1}^{T} \sum_{k=1}^{K} \delta_t^k p(x_t, z_t = C_k|\Theta) \\ &= \prod_{t=1}^{T} \sum_{k=1}^{K} \delta_t^k p(x_t, \delta_t^k = 1|\Theta) \end{aligned}$$

ce qui permet de calculer la vraisemblance des données complétées par

$$\begin{aligned}
\log p(Z, X|\Theta) &= \sum_{t=1}^{T} \log \left\{ \sum_{k=1}^{K} \delta_t^k p(x_t, \delta_t^k = 1|\Theta) \right\} \\
&= \sum_{t=1}^{T} \log \left\{ \sum_{k=1}^{K} \delta_t^k p(x_t, \delta_t^k = 1, \Theta) P(\delta_t^k = 1|\Theta) \right\} \\
&= \sum_{t=1}^{T} \log \left\{ \sum_{k=1}^{K} \delta_t^k p(x_t, \delta_t^k = 1, \theta_k) P(\delta_t^k = 1) \right\} \\
&= \sum_{t=1}^{T} \sum_{k=1}^{K} \delta_t^k \log \left[p(x_t, \delta_t^k = 1, \theta_k) w_k \right] \\
&= \sum_{t=1}^{T} \sum_{k=1}^{K} \delta_t^k \log \left[p(x_t|z_t = C_k, \theta_k) w_k \right]
\end{aligned} \tag{42}$$

L'équation (42) profite du fait que $p(x_t, \delta_t^k = 1, \Theta) = p(x_t, \delta_t^k = 1, \theta_k)$ et que $P(\delta_t^k = 1|\Theta) = w_k$.

En reprenant l'équation (42) et la définition de $Q(\Theta|\Theta_n)$ de l'équation (38), nous avons pour l'étape "E"

$$\begin{aligned}
Q(\Theta|\Theta_n) &= E_Z\{\log p(Z, X|\Theta)|X, \Theta_n\} && (43) \\
&= \sum_{t=1}^{T} \sum_{k=1}^{K} E\{\delta_t^k|x_t, \Theta_n\} \log \left[p(x_t|\delta_t^k = 1, \theta_k) w_k \right] && (44)
\end{aligned}$$

Définissons alors

$$h_n^k(x_t) \equiv E\{\delta_t^k|x_t, \Theta_n\} = P(\delta_t^k = 1|x_t, \Theta_n) \tag{45}$$

et dénotons w_k^n comme étant le poids de la $k^{\text{éme}}$ composante du mélange à l'itération n. Grâce au théorème de Bayès nous avons le résultat suivant

$$\begin{aligned} h_n^k(x_t) &= P(\delta_t^k = 1|x_t|\Theta_n) \\ &= \frac{p(x_t|\delta_t^k = 1, \Theta_n)P(\delta_t^k = 1|x_t|\Theta_n)}{p(x_t|\Theta_n)} \\ &= \frac{p(x_t|\delta_t^k = 1, \theta_k^n)P(\delta_t^k = 1|x_t|\Theta_n)}{p(x_t|\Theta_n)} \\ &= \frac{p(x_t|\delta_t^k = 1, \theta_k^n)w_k^n}{\sum_{l=1}^K p(x_t|\delta_t^l = 1, \theta_l^n)w_l^n} \end{aligned}$$

L'étape "E" détermine le meilleur choix pour la fonction $h_n^k(x_t)$. Une fois les probabilités $h_n^k(x_t)$ calculées pour chaque t et k , $Q(\Theta, \Theta_n)$ peut être considéré comme une fonction de Θ. Pour chaque itération de l'étape "M" de la fonction est maximisée dans le but d'obtenir la meilleure valeur de Θ notée Θ^*. Dans la plupart des cas, l'étape "M" peut être simplifiée puisque la fonction h est connue. Ceci dit l'étape "E" peut être vue comme une étape de préparation pour l'étape "M".

Approche paramétrique : Mélange de lois Gaussiennes

Dans [SG98, GSRL98, SEG00, KB01], les auteurs proposent une méthode adaptative de modélisation de la scène animée par une approche en deux classes, "fond"/"objet en mouvement", par un mélange de lois normales. Ce modèle à base de mélange est devenue très populaire du fait que l'intensité d'un pixel peut être modélisée de façon efficace par un mélange de lois Gaussiennes à condition de supposer un bruit d'acquisition non corrélé et de faibles changements de luminosité. Les auteurs de [GSRL98, KB01] proposent ainsi de modéliser le fond de la scène -par opposition aux objets en mouvement- par un mélange de lois Gaussiennes des valeurs RGB d'un pixel. Grimson [GSRL98] propose également un schéma de mise à jour dans lequel les pixels sont confrontés au modèle existant afin de permettre leur classification dans l'une des deux classes.

Considérant les valeurs de niveaux de gris de chaque pixel de l'image comme un processus stochastique indépendant, les auteurs supposent que la distribution suivie par les valeurs de luminance d'un pixel peut être modélisée par un mélange de lois Gaussiennes si la probabilité de la luminance x_t à l'instant t est définie comme suit :

$$p(x_t) = \sum_{i=1}^{K} w_{i,t}\eta(x_t|\ \mu_{i,t}, \sigma_{i,t}^2) \tag{46}$$

oú $w_{i,t}$ est le poids , $\mu_{i,t}$ et $\sigma_{i,t}^2$ sont respectivement la moyenne et la variance de la $i$ème Gaussienne à l'instant t ainsi que les poids $w_{i,t}$. Le but de leurs travaux consiste donc à estimer la densité de probabilité $p(x_t)$ pour chaque pixel et pour chaque instant t. Afin

de déterminer l'image de référence, chaque pixel est considéré de façon indépendante, les K distributions étant triées suivant le rapport w_i/σ_i^2. Ne sont retenues alors que les B distributions avec $B = argmin_b \left(\frac{\sum_{i=1}^{b} w_i}{\sum_{i=1}^{K} w_i} \right) > T$ où T est un paramètre défini a priori. La difficulté consiste ensuite à mettre à jour l'ensemble des paramètres du modèle pour correspondre à partir de la connaissance passée, à l'expérience actuelle. Grimson propose un schéma de mise à jour où les poids sont ajustés en considérant l'équation suivante :

$$w_{i,t} = (1 - \alpha) w_{i,t-1} + \alpha v(w_k | x_t) \tag{47}$$

oú $\alpha \in [0, 1]$ est le taux d'apprentissage fixé, $v(w_k|x_t)$ est égal à 1 si $k = i^*$, i.e. $\eta_k(\mu_k, \sigma_k^2)$ est la meilleure correspondance pour x_t, et 0 sinon. Moyenne et écart-type sont mises à jour en considérant

$$\mu_{i,t} = (1 - \alpha) \mu_{i,t-1} + \rho x_t \quad , \tag{48}$$

$$\sigma_{i,t}^2 = (1 - \alpha) \sigma_{i,t-1}^2 + \rho (x_t - \mu_{i,t})^2 \tag{49}$$

avec $\rho = \alpha \eta(x_t | \mu_{i,t}, \sigma_{i,t}^2)$. En pratique, et suivant les travaux de Grimson, une valeur de α aux alentours de 0.1 donne des résultats convaincants au cours du temps. La classification s'effectue ensuite en vérifiant que chaque nouveau pixel appartienne à $2,5$ fois l'écart type de chacune des B Gaussiennes restantes. Si une correspondance est trouvée le pixel fait partie du fond sinon il s'agit d'un pixel appartenant à un objet en mouvement. L'intérêt d'une telle approche réside dans le fait que les pixels de fond texturé où de façon plus générale dont la distribution est multi-modale sont parfaitement modélisés.

Kaewtrakulpong et Bowden, dans [KB01], proposent une optimisation de la méthode précédente pour des environnements très texturés. Les auteurs remplacent les équations de mise à jour de Grimson et al. pour répondre à des contraintes temps-réel plus drastiques. Un schéma de suppression des ombres y est également introduit et la phase d'initialisation est améliorée pour ne plus à avoir d'entraînement. Les auteurs font par ailleurs une remarque judicieuse sur la rapidité de réentrainement et l'incapacité du système proposé par Grimson et Stauffer à se remettre à jour rapidement dans le cas où un pixel de fond reste "fond" pendant trop longtemps.

S'appuyant sur le modèle de l'algorithme d'Expectation Maximization -EM- [Now91, NH98], les auteurs mettent à jour le modèle à mélange de lois Gaussiennes en estimant la fonction de densité de probabilité pour chaque pixel en conservant en mémoire une fenêtre temporelle de T pixels. Les équations de mise à jour deviennent dès lors :

$$w_{i,t} = w_{i,t-1} + \frac{1}{T} \left(v(w_k | x_t) - w_{i,t-1} \right) \tag{50}$$

avec $v(w_k|x_t)$ égale à 1 si la Gaussienne $\eta_k(\mu_k, \sigma_k^2)$ est la meilleure correspondance pour la donnée x_t, 0 sinon. Moyenne et variance sont également mises à jour comme suit :

$$\mu_{i,t} = \mu_{i,t-1} + \frac{1}{T}\left(\frac{v(w_k|x_t)x_t}{w_{i,t}} - \mu_{i,t-1}\right) \tag{51}$$

$$\sigma^2_{i,t} = \sigma^2_{i,t-1} + \frac{1}{T}\left(\frac{v(w_k|x_t)(x_t - \mu_{i,t-1})}{w_{i,t}} - \sigma^2_{i,t-1}\right) \tag{52}$$

2.3 Méthode proposée

Dans notre étude, le processus de détection des objets en mouvement à l'image a été découpé en deux étapes :

- détection des zones en mouvement,
- régularisation spatio-temporelle des masques de mouvement.

La détection de mouvement s'effectue par analyse de la valeur d'intensité lumineuse d'un pixel par rapport à la fonction de densité de probabilité correspondante. La deuxième étape consiste à affiner les masques de détection tout en réduisant les fausses détections présentes. Chacune de ces opérations fait donc l'objet d'une section particulière.

2.3.1 Détection de mouvement basée sur le modèle à mélanges de lois Gaussiennes

Du fait de la nature stochastique du signal vidéo, principalement bruité pendant l'acquisition, la majorité des méthodes procède par modélisation des distributions d'intensité lumineuse ou de la couleur par une loi choisie *a priori*. Dans la plupart des travaux, le choix d'une seule distribution Gaussienne est très souvent retenu du fait de sa faible complexité due au nombre réduit de paramètres à estimer. La détection des objets en mouvement revient dès lors à proposer un schéma de classification des nouvelles valeurs de pixel suivant une connaissance *a priori* de leur distribution. Comme nous l'avons vu précédemment, de nombreuses méthodes ont été suggérées afin de déterminer si la distribution des pixels d'une zone déterminée est préservée ou non au cours du temps. Or il s'avère que la modélisation par une seule loi normale n'est plus suffisante dès lors que l'on souhaite proposer un modèle sensible aux variations progressives de luminosité aussi bien pour les scènes filmées à l'intérieur qu'à l'extérieur. A cela s'ajoute le fait que généralement les caméras de surveillance vidéo sont soumises à de nombreuses perturbations et le signal transmis est fortement bruité au cours du temps. Pour toutes ces raisons, plutôt que de modéliser les valeurs d'un pixel suivant une distribution uni-modale, il lui est préféré [GSRL98] une modélisation multi-modale par un mélange fini de lois Gaussiennes (*Finite Mixture of Gaussians*).

Ces modèles de lois Gaussiennes ont été employés avec succès dans de nombreuses applications telles que la segmentation vidéo [HM00], la détection de visages humains [MGR98],

les problèmes de classification [McQ67] et la détection de mouvement [GSRL98] entre autres. Cette modélisation a également été employée dans différents espaces de représentation : espace objet spécifique [HM00], espace RGB [GSRL98], ...

Modèle de mélanges

Considérons l'ensemble des valeurs de luminance d'un pixel donné de l'image comme un processus stochastique au cours du temps. Le problème consiste tout d'abord à estimer la fonction de densité de probabilité de cet ensemble de valeurs le plus "finement" possible. Dans notre étude, nous avons considéré une approche paramétrique où les composantes sont représentées par un modèle de mélanges fini [LP00]. Ces modèles fournissent un cadre extrêmement flexible ainsi qu'un cadre de travail générique aussi bien pour la classification que pour l'estimation de la fonction de densité de probabilité.

Les modèles paramétriques et non-paramétriques, nous l'avons remarqué dans la section précédente, ont chacun leurs avantages et leurs inconvénients. Les méthodes paramétriques supposent une forme spécifique de la fonction de densité de probabilité (une loi normale par exemple) qui peut toutefois être totalement différente de la fonction idéale. D'un autre côté, les approches non paramétriques peuvent approximer n'importe quelle fonction de densité mais le nombre de variables du modèle adopté peut croître de façon drastiques suivant le nombre d'échantillons considéré [TSM85]. Les modèles à base de mélange représente donc un compromis appréciable.

Reprenons la formulation de la densité de probabilité telle que présentée à l'équation (46). Soit $X = \{x_1, ...x_n\}$, l'ensemble des valeurs d'intensité d'un pixel dans les n dernières images, nous supposons que ces valeurs peuvent être modélisées par un mélange de K distributions Gaussiennes -(46)-. Le but de notre travail consiste donc à estimer la densité de probabilité $p(x_t)$ pour chaque pixel et pour chaque instant t. L'ensemble des paramètres du mélange doit également être estimé depuis l'ensemble d'apprentissage X. Nous appelons cette estimation "initialisation du modèle à mélange" -*mixture model initialization*- Nous proposons une estimation qui comporte deux phases : initialisation du modèle et évolution du modèle au cours du temps.

Initialisation du modèle

Afin de proposer l'approximation de fonction de densité de probabilité la plus précise possible, le nombre optimal de composantes -ou modes-, K, doit être déterminé pour chaque pixel indépendamment les uns des autres. L'ensemble des paramètres de chacune des K composantes devra également être initialisé.

Ceci dit, nous devons résoudre le problème du choix, sans connaissance *a priori*, de ce nombre optimal. Pour ce faire nous avons supposé que pour chaque pixel, nous possédons un historique des valeurs d'intensité de ce pixel, et nous allons considérer une partition optimale de cet ensemble en groupes homogènes. Dans chacun de ces groupes, la distribution

de probabilité peut être modélisée par une seule Gaussienne. Afin de construire cette partition nous avons recours à l'algorithme de regroupement, non supervisé, appelé ISODATA [BH67], acronyme de "Iterative Self-Organizing Data Analysis Technique". Basé sur le algorithme K-means [McQ67], l'algorithme ISODATA est un algorithme non-hiérarchique de regroupement procédant à la fois par minimisation d'un critère d'inertie *intra* classe et par maximisation d'un critère *inter* classe.

Soit une partition complète de l'espace de données X défini tel que $X = \{X_1, ..., X_i, ..., X_K\}$ avec K le nombre maximal des éléments de la partition, c'est à dire $X = X_1 \cup ... \cup X_i \cup ... \cup X_K$ et $X_i \cap X_j = \emptyset$ pour $i \neq j$. Notons $g(X_i)$ le barycentre de X_i, et considérons la distance, d, de Mahalanobis entre les exemples $x_{ij} \in X_i$ et le barycentre de X_i, nous définissons l'inertie de X_i par $I(X_i) = \sum_{i=j}^{n_i} d(x_{ij}, g(X_i))$

Dès lors l'inertie inter et intra classes sont définies comme suit :

$$\underset{intra}{I(X)} = \sum_{i=1}^{K} I(X_i) \tag{53}$$

$$\underset{inter}{I(X)} = \sum_{j=1}^{K} \sum_{\substack{i=1 \\ i \neq j}}^{K} card(X_j).card(X_i)\ d(g(X_i), g(X_j)) \tag{54}$$

À chaque groupe X_i correspond une fonction de densité de probabilité. A la partition complète d'un pixel donné correspond le modèle de mélange.

En vue de définir un nombre "optimal" de groupes -clusters- avec les meilleurs critères d'inertie, l'algorithme ISODATA s'appuie sur un ensemble d'étapes de fusion et de découpage jusqu'à ce que le critère d'inertie inter reste stable. A la fin de l'algorithme, nous avons donc pour chaque pixel g un nombre total $K(g)$, $K(g) \leq K$, de composantes et nous pouvons dès lors estimer les paramètres associés au modèle. Afin de respecter les contraintes temps réel, la phase d'apprentissage ne doit pas être trop lourde. Le nombre de groupes pour chaque pixel g est également volontairement réduit. Ainsi l'algorithme ISODATA est arrêté volontairement suivant un seuil de décroissement de (53) et d'accroissement de (54). Dans le cadre de ce travail, nous nous sommes limités à un maximum de 3 classes.

Schéma de mise à jour du modèle

Comme les conditions d'éclairage ne sont pas fixes au cours du temps, le modèle nécessite d'être mis à jour. Nous avons divisé ce processus en deux étapes [CBP05a, CBP05b]. La première détermine et identifie pour chaque valeur d'intensité de pixel ou groupe de valeurs, celles qui ne suivent pas le modèle de distribution existant. Si un pixel est considéré comme faisant partie de la classe "objet en mouvement" pendant une période fixée *a priori* alors la deuxième étape intervient. Elle consiste à mettre à jour le modèle de distribution de ce pixel. Si le pixel suit le modèle (ce qui arrive la plupart du temps) nous avons choisi de

considérer non pas l'algorithme EM trop coûteux, ni même ses optimisations [MB03, IU03] [FR97], ni le filtre de Kalman [KWH+94] trop sensible, mais plutôt de suivre l'ensemble des équations itératives de mise à jour proposées dans [SG98, GSRL98, SEG00].

Pour chaque pixel à l'instant t, la valeur de luminance de chaque pixel est confrontée au modèle de mélange précédemment construit. Si la valeur de luminance suit le modèle alors cette nouvelle valeur sera utilisée pour mettre à jour le modèle. Dans le cas contraire, le pixel est considéré comme étant un "outlier", donc appartenant à la classe "objet". Un nouveau modèle doit être alors entraîné grâce à cette nouvelle valeur et les valeurs d'intensité cumulées de ce pixel.

Règle de décision statistique

Afin de prendre la décision entre la mise à jour ou le réentrainement complet du modèle, nous devons déterminer à quelle Gaussienne du mélange cette nouvelle valeur correspond le mieux. Pour ce faire nous allons introduire une règle de décision basée sur le principe de maximum de vraisemblance [CBP05b].

La vérification est réalisée de la façon suivante : soit x_t un échantillon donné, nous devons maximiser la vraisemblance de la composante η_i dans le mélange $p(x_t) = \sum_{i=1}^{K} w_{i,t}.\eta(x_t|\mu_{i,t}, \sigma_{i,t}^2)$ par rapport à x_t. Dans ce cas, nous allons considérer la vraisemblance de la composante η_i conditionnellement à l'ensemble du mélange. Plus formellement, soit une partition complète de l'espace des hypothèses H définie par $B = \{H_1, ..., H_i, ..., H_K\}$. Associons chaque loi Gaussienne η_i avec H_i. En considérant le Théorème de Bayes, nous avons $P(H_i/B) = P(H_i.B)/P(B)$. En supposant que H_i, $i = 1, .., K$ forme une partition complète de B, c'est à dire que $B = H_1 \cup ... \cup H_i \cup ... \cup H_k$, avec $P(B) = 1$ et $P(H_i.H_j) = 0$. Ainsi $P(H_i.B) = P(H_i.(H_1 + H_2 + \cdots + H_i + \cdots + H_K)) = P(H_i)$. Cela se déduit du fait de l'indépendance des hypothèses $\forall i \neq j \quad P(H_i.H_j) = 0$ et du fait que $P(H_i.H_i) = P(H_i)$. Comme $P(B) = 1$ on a donc $P(H_i/B) = P(H_i)$.

Associons à cela la densité de probabilité $p(x_t \in H_i/B)$ telle que

$$p(x_t \in H_i/B) = \frac{w_i \eta_i}{\sum_{k=1}^{K} w_k \eta_k} \tag{55}$$

alors la vraisemblance conditionnelle de la i^{me} Gaussienne dans le mélange pour un échantillon x_t est

$$l(x_t) = \frac{w_i \eta_i}{\sum_{k=1}^{K} w_k \eta_k} \tag{56}$$

Suivant le processus de décision usuel, il s'agit dès lors de maximiser la log-vraisemblance $L_i = log(l(x))$. Nous obtenons ainsi

$$L_i = log(l(x)) = log\ \frac{w_i.\eta_i}{\sum_{k=1}^{K} w_k \eta_k} = log\ (w_i \eta_i) - log\ (\sum_{k=1}^{K} w_k \eta_k) \tag{57}$$

or comme $log\ (\sum_{k=1}^{K} w_k \eta_k)$ est le même pour tous les L_i, cela revient à maximiser $log\ (w_i \eta_i)$ exprimé de la façon suivante :

$$log\ (w_i \eta_i)\ =\ log\ w_i + log\ \frac{1}{\sqrt{2\pi\sigma_i^2}} exp\left(-\frac{(x-\mu_i)^2}{2\sigma_i^2}\right) \tag{58}$$

ce qui finalement équivaut à rechercher η_i^* et $\Theta_i^* = (w_i^*, \mu_i^*, \sigma_i^{2*})$ tels que

$$\Theta_i^* = \underset{\Theta_i=(w_i,\mu_i,\sigma_i^2)}{argmax} \left(log\ w_i - \left(\frac{log\ \sigma_i^2}{2} + \frac{(x-\mu_i)^2}{2\sigma_i^2}\right)\right) \tag{59}$$

Nous définissons ainsi une règle de décision qui permet la sélection de la "meilleure" loi Gaussienne dans le mélange pour un échantillon donné x_t. Si $\Theta_i^* = (w_i^*, \mu_i^*, \sigma_i^{2*})$ obtenu par la fonction de décision est au dessus d'un seuil fixé, la valeur x_t sera utilisée pour mettre à jour le modèle. Pour le choix du seuil nous utilisons les bornes de l'équation (59) soit $0 \leq l(x) \leq 1$.

A partir des tests réalisés sur différents corpus, l'équation (59) donne des résultats tout à fait satisfaisant comme il est démontré dans la section suivante.

Néanmoins le calcul du premier terme logarithmique dans l'équation (59) peut être pénalisant pour des applications de temps réel ou pseudo temps-réel, ce qui est notre objectif ici. Pour régler ce problème on peut toutefois effectuer une supposition forte sur les poids en les supposant tous égaux. La maximisation définie par (59) devient donc

$$\Theta_i^* = \underset{\Theta_i=(\mu_i,\sigma_i^2)}{argmin} \left(\frac{log\ \sigma_i^2}{2} + \frac{(x-\mu_i)^2}{2\sigma_i^2}\right) \tag{60}$$

Mise à jour du modèle

Du fait des contraintes de temps, la modélisation de chaque pixel de l'image nécessite un algorithme de mise à jour efficace et rapide. En supposant que les valeurs de luminance de fond sont celles dont les variations sont les plus faibles au cours du temps, les travaux de Grimson et Stauffer [GSRL98] ou encore les travaux Kaewtrakulpong et Bowden's [KB01] ont proposé une série d'équations de mise à jour basée sur l'algorithme d'Expectation Maximization.

Dans nos travaux nous avons suivi les équations proposées dans [GSRL98] du fait des contraintes temps réel imposées. A partir des équations décrites ci-dessous les poids du mélange sont mis à jour, au temps t par

$$w_{i,t} = (1-\alpha)w_{i,t-1} + \alpha v(w_k|x_t) \tag{61}$$

où $\alpha \in [0,1]$ est le taux d'apprentissage, $v(w_k|x_t)$ est égale à 1 si $k = i^*$, c'est à dire si $\eta_k(\mu_k, \sigma_k^2)$ est la meilleure correspondance pour x_t, et 0 sinon.

De plus moyenne et variance sont mises à jour avec

$$\mu_{i,t} = (1-\alpha)\mu_{i,t-1} + \rho x_t \quad , \tag{62}$$

$$\sigma_{i,t}^2 = (1-\alpha)\sigma_{i,t-1}^2 + \rho(x_t - \mu_{i,t})^2 \tag{63}$$

avec $\rho = \alpha\eta(x_t|\mu_{i,t}, \sigma_{i,t}^2)$. Une valeur α proche de 1 aura tendance à écraser le modèle de mélange existant par une seule Gaussienne centrée sur une valeur définie comme étant la nouvelle valeur d'intensité. Si $\alpha = 0$, le processus de mise à jour n'effectuera aucune modification sur le mélange. En pratique, une valeur de 0.1 donne des résultats satisfaisants.

Dans les travaux de Grimson et Stauffer[GSRL98, SG98, SEG00], un seuil minimum de probabilité est fixé a priori. Ce seuil indique si la valeur d'intensité d'un pixel suit le modèle de fond défini ou non. Ce seuil signifie que si le poids d'une des Gaussiennes dans le mélange est en dessous, la Gaussienne correspondante sera supprimée du modèle. Pendant le processus de mise à jour, dans les travaux de Grimson, les auteurs peuvent supprimer les Gaussiennes de poids faible en supposant qu'elles modélisent éventuellement la distribution des valeurs d'intensité d'un pixel qui serait un pixel de mouvement.

Dans nos travaux, nous n'avons pas choisi de retenir cette approche pour la raison suivante : comme le fait remarquer Kaewtrakulpong [KB01], le schéma présenté par Grimson est satisfaisant pour de faibles variations de luminosité. Le paramètre α -cf. équations (61), (62) et (63)- permet certes de modifier la "rapidité d'apprentissage" de leur algorithme mais une fois ce paramètre figé, la rapidité d'apprentissage ne peut plus être modifiée pendant le processus de classification. Si la valeur de α est choisie proche de 1, le processus s'adaptera rapidement aux nouvelles valeurs d'intensité au détriment du modèle antérieur. Si des changements rapides de luminosité sont constatés, cette valeur de α peut être intéressante mais s'il s'agit de variation ponctuelle de la luminosité, le modèle est alors détruit au détriment de toutes les précédentes valeurs d'intensité. Au contraire une valeur de α proche de 0 a tendance à proposer le schéma inverse : les fortes variations sont détectées et le modèle ne se met pas à jour suffisamment rapidement. Il s'agit dès lors de trouver un compromis entre rapidité d'apprentissage et taux de détection...

Dans la méthode que nous proposons, nous nous efforçons de toujours conserver le même nombre de distributions Gaussiennes qu'initialement proposé. Cette affirmation suppose que l'échantillon d'apprentissage soit suffisamment conséquent pour être représentatif du modèle réel à estimer. Il est donc important de proposer un schéma de décision/mise à jour qui puisse *conserver le nombre de composantes du modèle initial.*

Tout en conservant cette optique, nous proposons le schéma de mise à jour suivant : à chaque itération de temps, les valeurs d'intensité pour chaque pixel d'une nouvelle image sont confrontées au modèle de mélange suivant les équations (59) ou (60). En fonction de la log-vraisemblance obtenue, une décision de mise à jour est prise. Ainsi si la log-vraisemblance (59) ou (60) est inférieure à un seuil fixé expérimentalement, cette nouvelle valeur de pixel est jugée comme hors modèle. On étudie également combien de temps ce pixel reste "en dehors" du modèle. Si cette période de temps Δ_t est trop importante, cela veut dire que le modèle par mélange ne convient plus pour modéliser le fond.

Deux situations peuvent alors se présenter dans ce cas précis : soit il s'agit d'un pixel appartenant à un objet en mouvement et, pendant la période Δ_t, il est normalement considéré comme étant en mouvement. Dans ce cas, la période Δ_t doit être suffisamment grande pour éviter d'incorporer l'objet dans le modèle de fond. Soit il s'agit d'une variation du modèle de luminance de fond pour ce pixel dans ce cas, comme par exemple le changement d'éclairage, Δ_t doit être petit pour mettre à jour le modèle le plus rapidement possible. Un compromis doit clairement être trouvé entre rapidité de mise à jour et précision de la détection. Dans le cadre de ce travail, la valeur Δ_t est égale à 10.

Si un pixel est considéré pendant toute la longueur de la période de temps, Δ_t, comme étant un "outlier" au modèle, une fenêtre de taille Δ_t des valeurs précédentes d'intensité est considérée. Le modèle de mélanges est réinitialisé en ré-exécutant sur cette fenêtre d'intensité l'algorithme ISODATA comme lors de la première phase d'initialisation. Autrement dit le modèle de fond est réappris à chaque variation brutale d'éclairage.

2.3.2 Régularisation par approche markovienne

Afin de prendre en considération les informations contextuelles locales en vue d'une amélioration de la qualité et de la robustesse du processus de détection, nous avons défini un modèle markovien d'interaction locale des cartes de détection. La segmentation du mouvement obtenue avec le processus précédent fournit, certes, des résultats déjà tout à fait satisfaisants mais malheureusement la segmentation est bruitée et insuffisamment précise pour pouvoir fournir une interprétation correcte des zones de la scène à analyser.

Dans le but d'affiner la détection et en supposant une continuité à la fois spatiale et temporelle des cartes de détection, nous avons intégré une information contextuelle, basée sur une connaissance a priori du modèle, dans la décision prenant en compte les relations statistiques entre un point et son voisinage temporel et/ou spatial. Appliqué aux problèmes de détection de contour [CR87], d'analyse de mouvement [Hei88, Pér93], la modélisation par champ aléatoire de Markov s'est finalement révélée également adaptée pour notre problématique de détection de mouvement [Lal90].

Ainsi le choix d'une étiquette, "fond" ou "objet en mouvement" en un point ne dépend plus uniquement du processus de détection mais plutôt des liens directs entre l'observation en ce point et des valeurs de son voisinage.

Besag [Bes74] a finalement rendu célèbre la relation entre champs de Markov et les distributions de Gibbs, théorème déjà prouvé bien longtemps avant par Hammersley et Clifford [HC71].

Formulation

Nous reprenons ici la modélisation des cartes d'étiquettes proposée au chapitre précédent et montrons les particularité de cette formulation dans notre cas. Nous rappelons qu'un champ de Markov est un ensemble de variables aléatoires, défini sur un graphe où la probabilité de chaque variable ne dépend que de l'état de ses voisins. Afin d'éviter toute redondance nous reprenons les définitions présentés dans la section 2.2.2.

Le Théorème d'Hammersley et Clifford établit la relation entre les Champs de Markov et la distribution de Gibbs (cf Chapitre précédent). Elle se résume de la façon suivante :

$$p(E = e)\frac{1}{Z_c}\exp(-U_1(e)) \tag{64}$$

avec Z_c une constante de normalisation. $U_1(e)$ est appelée fonction d'énergie qui se décompose comme la somme de potentiels locaux

$$U_1(e) = \sum_{c \in \mathcal{C}} V_c(e) \tag{65}$$

avec $\mathcal{C}$ l'ensemble des cliques de $\mathcal{S}$ associées au système de voisinage $\mathcal{V}$. Généralement on ne considère que des systèmes de voisinage d'ordre 1 ou 2.

Dans notre travail nous avons adopté l'estimation au sens du Maximum A Posteriori -MAP- comme présenté précédemment. Il s'agit donc de maximiser la probabilité conditionnelle a posteriori des observations $p(O = o|E = e)$. Il est donc pour cela nécessaire de modéliser la relation qui lie les étiquettes du champ E et les mesures données par les observations du champ O. Cette relation se modélise différemment suivant la problématique abordée.

En suivant les travaux de Ricquebourg [Ric87], nous allons nous attacher à la maximisation de la relation suivante :

$$p(O = o|E = e) \times P(E = e) = \frac{1}{Z}\exp^{-U(e,o)} \tag{66}$$

avec

- Z une constante de normalisation,
- $U(e, o) = U_1(e) + U_2(e, o)$ une fonction d'énergie totale,
- $U_1(e) = \sum_{c \in \mathcal{C}} V_c(e)$ la connaissance a priori supposée sur la répartition des étiquettes du champ, et enfin

– $U_2(e,o) = \sum_{s\in\mathcal{S}} V_o(e,o)$ le terme d'énergie lié à la vraisemblance des observations.

Au vue de la relation (66), la configuration des étiquettes la plus probable qui maximise, au sens MAP, est celle finalement qui minimise la fonction d'énergie totale :

$$\hat{e} = \arg\min_{e\in\Omega} U_1(e) + U_2(e,o) \tag{67}$$

De cette façon, et grâce à $V_c(e)$ et à $V_o(e,o)$ respectivement modèle a priori et terme d'attache aux données, ce critère global s'obtient finalement en n'effectuant que des calculs locaux. Le choix de ces potentiels locaux est donc d'une importance capitale car il conditionne les propriétés du champ E et la formulation de la relation entre E et les observations O.

Une modélisation d'adéquation aux observations couramment employée consiste à choisir la relation suivante :

$$O = \Phi(E) + \eta \tag{68}$$

où η est le bruit du modèle que l'on considère généralement blanc, Gaussien, centré et de variance σ^2.

Nous avons considéré dans cette étude un contexte à la fois spatial et temporel dans une séquence d'images [Lal90]. Dans ces mêmes travaux, le problème de détection de mouvement est analysé suivant les variations d'intensité lumineuse entre deux images successives. L'observation retenue est donc la différence inter-images $\Delta I_{t+dt,t}(s) = |I_{t+dt}(s) - I_t(s)|$ et le champ de primitives est un étiquetage binaire a ou b de chaque site. La signification de l'étiquetage retenu est la suivante : a pour l'appartenance du site au masque d'un objet en mouvement, b pour un fond fixe.

A la différence de [Lal90], nous n'avons pas considéré la différence inter-images mais nous avons plutôt retenu directement l'étiquetage obtenu par le processus de détection de mouvement par mélanges de lois Gaussiennes.

La relation entre observations et étiquettes est donc définie d'après l'équation (68) :

$$\Phi(e_{t-dt}(s), e_t(s)) = \begin{cases} 0 & si \quad (e_{t-dt}(s), e_t(s)) = (b,b) \\ m_1 & si \quad (e_{t-dt}(s), e_t(s)) = (a,a) \\ m_2 & sinon \end{cases} \tag{69}$$

avec m_1 et m_2 des constantes définies a priori de telle façon que $0 < m_1 << m_2$.

En ce qui concerne le modèle d'interaction, le choix du potentiel $V_c(e)$ dépend du modèle a priori souhaité et doit, par conséquent, favoriser certaines répartitions de cartes d'étiquettes désirées.

Bien que le choix de ce potentiel dépende de la problématique abordée, nous cherchons généralement à définir des champs les plus homogènes possibles.

Dans notre étude nous avons préféré une partition en régions compactes et aux frontières bien régulières, lissant et débruitant le plus possible les cartes de mouvement obtenues précédemment. Considérant les étiquettes e_s et e_t de deux sites voisins au sein d'une clique binaire $< s, t >$ nous avons retenu le modèle d'interaction suivant [DE87, MB87, CBP05b, CBP05a] :

$$V(e_s, e_t) = \begin{cases} 0 & si \quad e_s = e_t \\ \beta & sinon \end{cases} = \beta(1 - \delta_{e_s,e_t}) \tag{70}$$

En reprenant les équations (69) et (70), la formulation finale devient dès lors

$$U(e) = \sum_{c \in \mathcal{C}} V(e) = \sum_{c \in \mathcal{C}} (\beta(1 - \delta_{e_s,e_t}) + \Phi(e_{t-dt}(s), e_t(s))) \tag{71}$$

Dans la section suivante, nous donnons quelques résultats de détection de mouvement et de régularisation sur des séquences vidéos typiques de surveillance. Dans notre étude, nous avons considéré le voisinage de premier ordre 4 connexe d'un site S. Les cliques sont donc constituées de pixels adjacents verticalement ou horizontalement.

En ce qui concerne la relation entre observations et étiquettes, nous avons considéré l'image courante et l'image précédente. Suivant notre approche à deux classes, nous calculons l'équation (71) pour une étiquette "objet en mouvement" et pour la classe "fond" et nous retenons celle qui maximise l'énergie définie par la fonction potentielle.

2.4 Résultats Expérimentaux

La méthode de détection de mouvement que nous proposons dans ce travail a été testée sur notre corpus de vidéo surveillance comprenant de nombreuses séquences vidéo sujettes à différentes prises de vues. Ce corpus comprend des vidéos d'intérieur, d'extérieur, avec ou non de fortes variations de luminosité, etc.

Expérimentations

Nous avons validé notre méthode à la fois sur des vidéos de surveillance prises à l'intérieur d'un local (cf. figure 6) et sur des vidéos prise à l'extérieur (cf. figure 8). Dans le premier cas, du fait de la position de la source lumineuse et des variations rapides des conditions de luminosité dans la scène, ce genre d'environnement est un défi très intéressant pour de nombreux systèmes de détection. La deuxième validation intervient ensuite sur les séquences du corpus comprenant des scènes d'extérieur où les conditions de luminosité ne sont pas aussi critiques ici puisqu'il y réside une certaine stabilité de la source lumineuse. Ceci dit dans ce genre de scène, le système doit être suffisamment robuste pendant de longues périodes de temps.

Nous avons tout d'abord testé notre implémentation sur deux séquences vidéos "noir et blanc" de surveillance d'extérieur contenant chacune environ 5,000 images au format CIF. Ensuite nous avons testé nos algorithmes sur 3 séquences couleur contenant chacune environ 2,500 images toujours au format CIF. Toutes ces séquences ont été acquises à une cadence temporelle de 25 images par seconde.

Un compromis entre complexité en temps de calcul et qualité de l'approximation de la fonction de densité de probabilité peut être obtenu avec 3 Gaussiennes au maximum dans le mélange comme il est illustré sur les figures 10(a) et 10(b). La figure 10(a) illustre l'histogramme normalisé d'un pixel donné appartenant toujours à la classe "fond". La loi normale de poids le plus élevé correspond à l'approximation de la distribution de probabilité par une seule loi normale. Les trois lois de poids inférieur modélisent la distribution suivant trois lois. La figure 10(b) présente l'histogramme normalisé d'un pixel appartenant alternativement aux classes "fond" et "objet en mouvement", c'est à dire qu'un objet a traversé ce pixel à un instant donné. Ces figures permettent de comparer l'approximation qui est faite avec une seule loi normale et avec trois Gaussiennes. De ce schéma il apparaît qu'effectivement la modélisation par mélange est bien plus précise que pour une seule Gaussienne.

La figure 11 donne pour un même pixel l'erreur quadratique entre l'approximation effectuée par une Gaussienne ou par un mélange de Gaussiennes par rapport à l'histogramme des intensités de ce même pixel. Comme nous pouvons le voir sur la figure 11, la différence en terme de qualité d'approximation n'est pas si importante dans les cas de 3 ou 4 comparé à 1 ou 2 Gaussiennes. Cette situation est typique pour l'ensemble des pixels que nous avons pu observer. Il n'est pas moins intéressant de remarquer également qu'à partir de 3 Gaussiennes dans le mélange, l'erreur d'approximation ne semble plus réellement diminuer suivant l'augmentation du nombre de composantes du modèle.

Tous les traitements présentés dans cette étude ont été réalisés par un processeur de type Pentium IV 1,8Ghz, 1GoRAM. La séquence réalisée est extraite du corpus "Lionel&Laurent" grâce à la règle de décision par vraisemblance donnée par l'équation (59) et son optimisation donnée par l'équation (60). Les résultats de la figure 10(a) correspondent à l'approximation stochastique d'un pixel de fond de coordonnées $(x = 120, y = 140)$ pour la séquence "Lionel&Laurent" des trames $t = 100...250$. Les résultats présentés sur la figure 10(b) illustrent l'approximation d'un pixel de fond de coordonnées $(x = 10, y = 30)$ pour la séquence "RondPoint" des images $t = 1150...1400$. Cette figure montre également les distributions expérimentales et les mélanges Gaussiens obtenus.

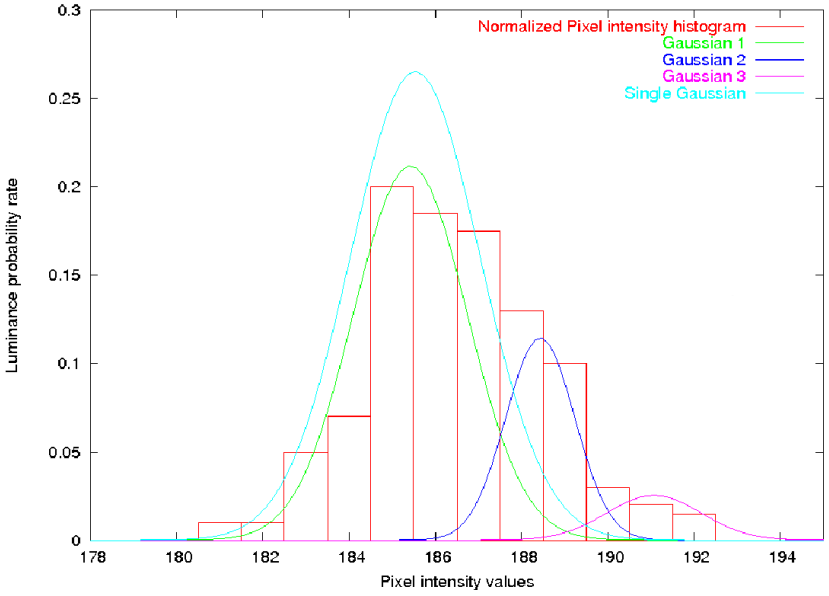
La figure 13 présente les résultats de la détection de mouvement. Les images de la colonne du milieu correspondent à la règle de décision complète -Eq. (59)-, la colonne de droite les images résultant de la prise de décision simplifiée -Eq. (60). Notons la détection des mouvements des petits objets, feuilles notamment, sur les images 13(h)13(i) 13(k)13(l).

La figure 14 présente des résultats de la détection de mouvement régularisée par champs de Markov telle que nous l'avons décrite dans la section 2.2.2. La colonne de gauche correspond à une régularisation purement spatiale où le terme d'énergie $V(e_s, e_t)$ (Eq. (70))

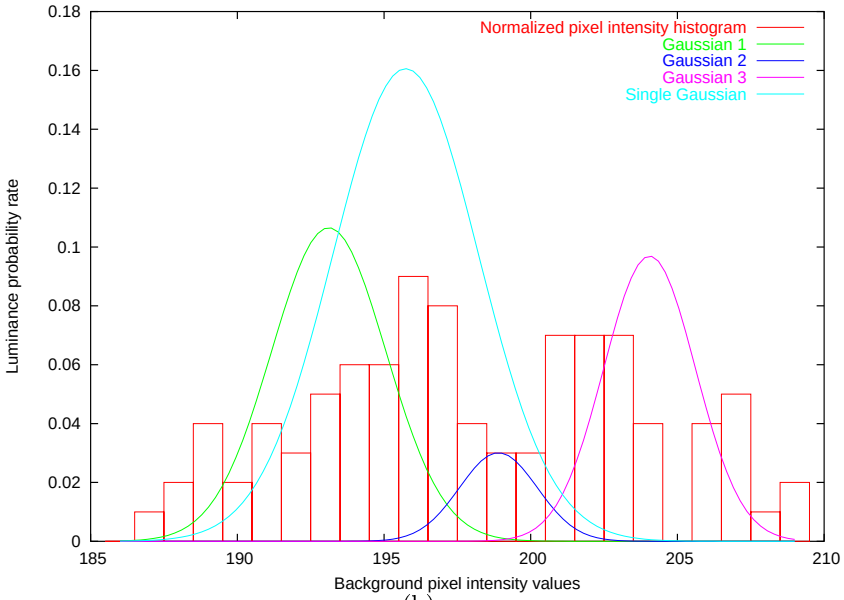
est supposé constant pour toutes les configurations. Nous pouvons constater dans ce cas la non-détection de la personne à droite sur les images 14(a) et 14(d). L'activité des petits objets en mouvement tels que les feuilles des arbres est détectée sur les images 13(h),13(i), 13(k) et 13(l) mais ensuite filtrée par le processus de régularisation (figures 14(g), 14(h), 14(i),14(j), 14(k), 14(l)). La colonne du milieu de la figure 14 présente les résultats pour la méthode de décision optimisée (Eq. (60)) avec la même régularisation. Enfin la colonne de droite donne des illustrations de masques obtenus suivant la règle de décision optimisée (Eq. (60)) avec notre méthode de régularisation spatio-temporelle. Notons que les activités de mouvement des petits objets ne sont plus dorénavant détectées. On peut dénoter également une augmentation qualitative de la précision des masques de mouvement obtenus. Ces exemples de traitement valident donc l'approche de régularisation proposée.

La figure 12 présente les résultats de consommation en temps processeur (CPU Time) requis pour faire le traitement d'une image complète au format CIF de la séquence "Lionel&Laurent". Ce temps comprend le temps total requis pour une image, c'est à dire pour le chargement de l'image en mémoire, sa classification et son affichage. Du fait de diverses accélérations mises en place pour l'accès direct à la mémoire vidéo et disque, nous pouvons estimer que les temps de chargement et d'affichage sont quasiment nuls. De ce schéma il est possible de constater, que même sans optimisation particulière du code C/C++ développé, une implémentation temps-réel est déjà possible grâce à la méthode de classification simplifiée que nous avons proposé à l'équation (60) -la courbe la plus basse- ainsi que pour la classification simplifiée combinée à une régularisation markovienne -courbe au dessus de la précédente-. Nous restons dans le cadre temps-réel proche de 25 images par seconde si le temps CPU est inférieur à $0.04s$ pour une image.

Dans le souci de réduire les coûts calculatoires (temps, occupation) des processus de régularisation de type Markov, le traitement peut être arrêté après plus de 5 itérations ou quand un état stable de l'énergie définie par l'équation (71) est atteint. Comme nous pouvons le constater sur la figure 13, le temps de détection est réduit par un facteur de 2 à 4 en moyenne avec l'utilisation de la règle de décision optimisée donnée par l'équation (60) au lieu de l'équation (59). Comme nous pouvons le constater la détection obtenue ne décroît pas en terme de qualité de détection mais augmente légèrement le taux de fausses détections -zones blanches sur le fond-.



(a)



(b)

Fig. 10. Approximations stochastiques des histogrammes d'intensité lumineuse pour un pixel de fond 10(a) et pour un pixel traversé par un objet 10(b) pour la séquence "Lionel&Laurent".

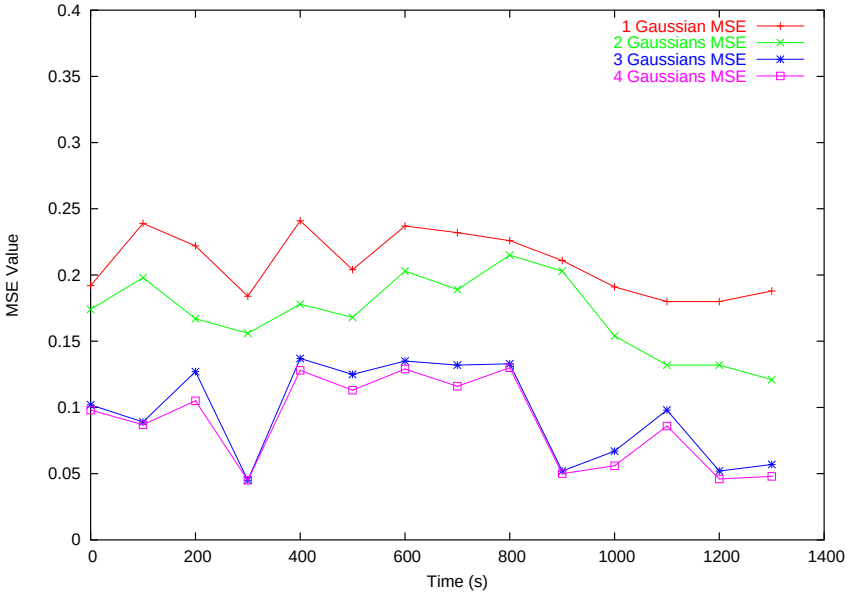


Fig. 11. Comparaison de l'erreur quadratique suivant 1 et n Gaussiennes pour un pixel de fond de coordonnées $x = 120, y = 140$ de la séquence "Lionel&Laurent".

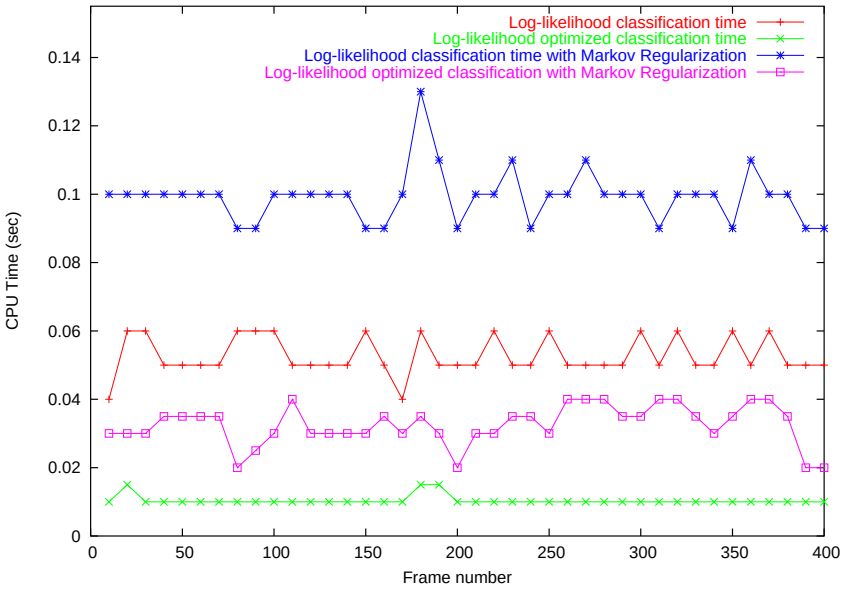


Fig. 12. Temps processeur consommé pour le processus de détection en considérant 3 Gaussiennes pour la séquence "Lionel&Laurent".

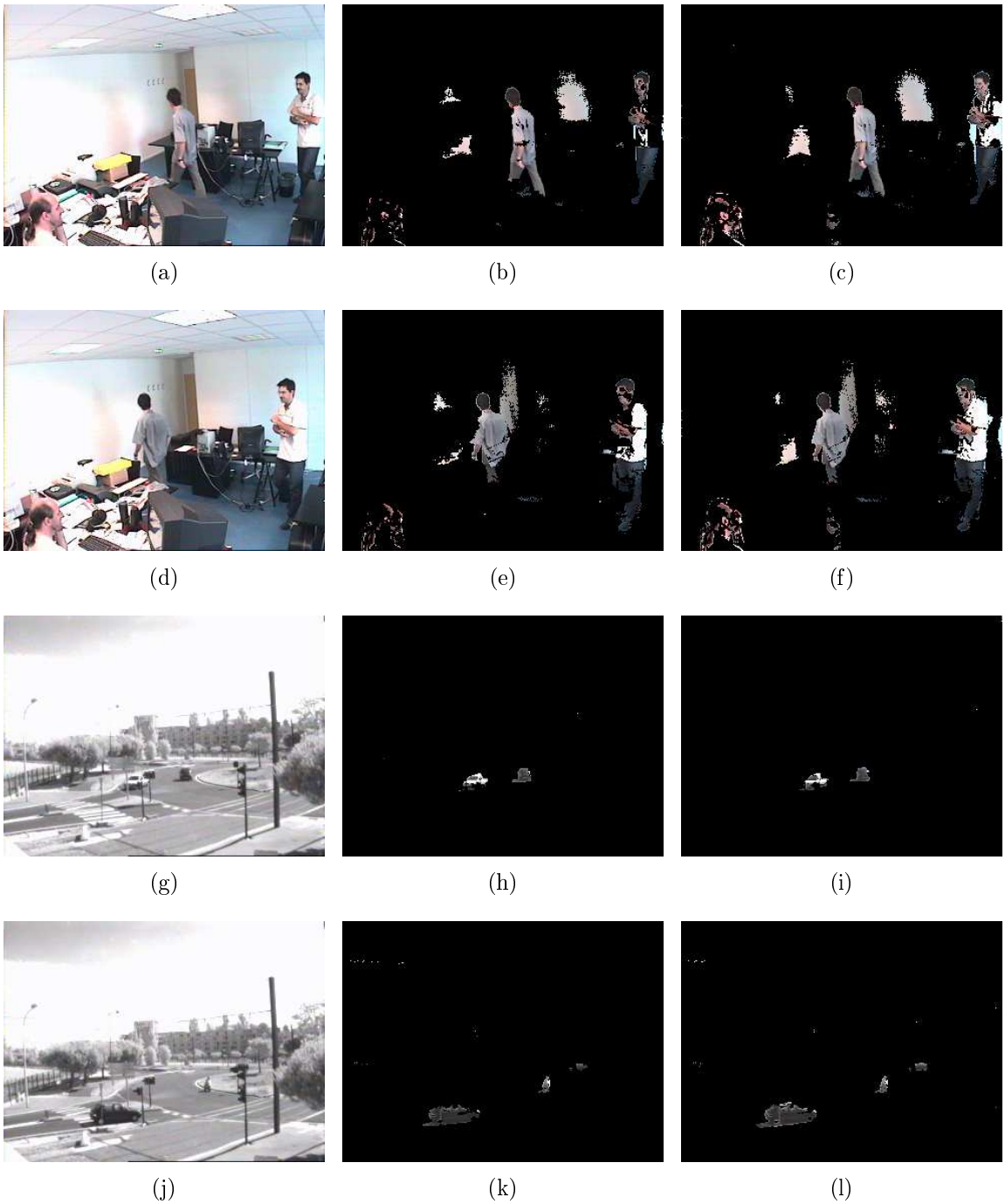


Fig. 13. Exemples de masques de mouvement obtenus sans régularisation Markovienne sur les séquences "Lionel&Laurent" (images 260 and 300) et "RondPoint" (images 450 and 600). La colonne du milieu présente la détection de mouvement suivant la méthode de décision complète (Eq. (59)), la colonne de droite suivant la version simplifiée (Eq. (60)).

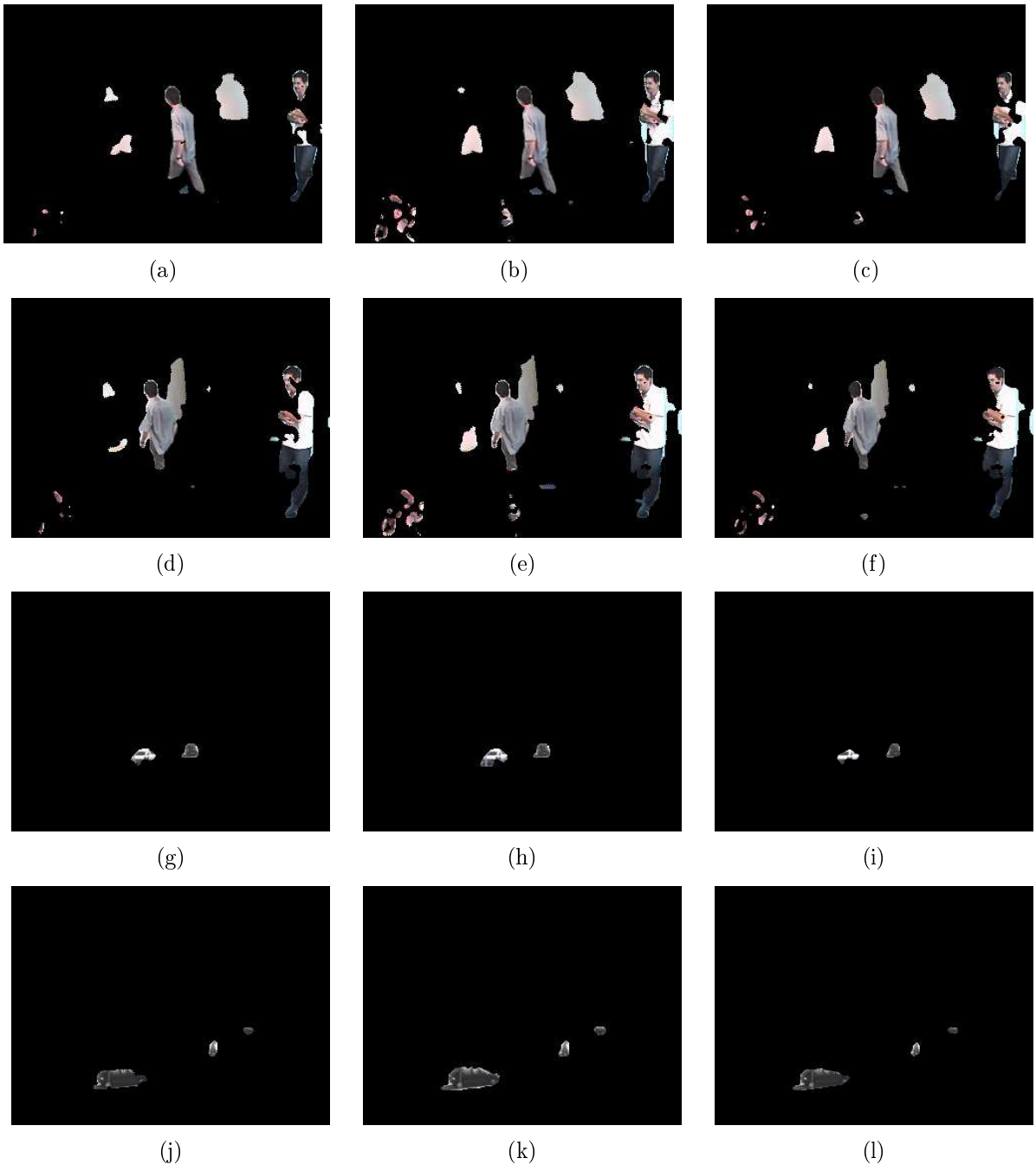


Fig. 14. Échantillon de masques de mouvement obtenus à partir de notre corpus de surveillance vidéo.

2.5 Conclusion

Après un tour d'horizon de l'état de l'art en matière de détection de mouvement, nous avons ensuite présenté notre contribution à l'extraction des masques des objets mobiles basée sur un modèle de mélange de lois Gaussiens associée à une régularisation Markovienne. Le modèle proposé est une alternative à celui proposé par Grimson, notamment dans le schéma de mise à jour retenu. Ainsi ce schéma dit par réentrainement permet de répondre au mieux, selon nous, à l'aspect générique que doit avoir une solution de surveillance vidéo en répondant à la fois à la problématique des faibles et des fortes variations de luminance entre deux images.

De plus, dans notre étude nous n'avons considéré que les valeurs de luminance ce qui permet, au contraire des travaux de Grimson, de s'affranchir des deux autres composantes, U et V, et d'obtenir un gain non négligeable sur les temps de calcul tout en offrant un modèle de détection stable et robuste. Cette méthode permet ainsi d'être attractive pour des systèmes basés sur des équipements peu coûteux. A partir d'une règle de décision basée sur le maximum de vraisemblance, nous l'avons optimisé pour la rendre temps-réel sans perte significative en terme de détection.

Nous avons basé cette technique sur une coopération entre détection et régularisation dans le but de réduire les fausses détections présentes à l'image tout en offrant des solutions également plus compactes au sens du mouvement. Ce procédé intègre plus correctement les informations de détection de mouvement observables que les autres approches que nous avons pu détaillé auparavant et donne des résultats bien meilleurs sur l'intégrité de l'objet et dans l'élimination des fausses détections.

Il est intéressant de noter que le schéma de régularisation markovienne que nous proposons a d'ores et déjà fait l'objet d'une implémentation temps-réel dans le cadre de la campagne Technovision-Argos. L'évaluation restant, lors de la rédaction de ce mémoire, en cours nous n'avons pas pour le moment souhaité proposer de résultats de détection. Les résultats obtenus sur notre corpus de surveillance vidéo sont toutefois très encourageants sur du matériel de surveillance vidéo non spécifique c'est à dire en utilisant simplement des ordinateurs personnels.

En perspective de ce détecteur de mouvement, il reste toutefois à répondre à un certain nombre de points : le premier point d'étude serait d'éliminer les ombres projetées sur le sol par les objets et les ombres présentes également sur les objets. Cette étape reviendrait à sophistiquer notre étape de détection en lui apportant les améliorations suivantes : il est possible de considérer le modèle par mélange de lois, travailler non plus sur la seule composante Y mais gérer explicitement les autres composantes mettant en défaut le processus de détection sur les ombres. L'autre possibilité consisterait sinon à intégrer un module extérieur, précédent le module de détection, en vue de la seule détection des zones d'ombre comme il est fait dans [PN03] par exemple.

La deuxième optimisation interviendrait au niveau de la finesse des masques de mouvement : en effet, dans un souci de conservation du temps de calcul, le processus de régularisation est stoppé soit après que l'énergie des potentiels se stabilise soit après un nombre fini d'itérations. Or il s'avère que de tels arrêts provoquent des trous de "non détection". Des approches morphologiques -comme proposé dans [PBP04] avec un approche par pixel et non par bloc- permettraient en complément de ces travaux d'obtenir des résultats encore plus exploitables tout en conservant suffisamment de processeur pour d'autres ressources...

Chapitre 3

Détection d'objets par apprentissage supervisé

Dans ce chapitre, nous allons exposer les outils mathématiques utilisés dans la deuxième partie de ces travaux, à savoir la détection et le suivi d'objets appris. Dans un premier temps, nous allons nous intéresser aux notions générales de la théorie de l'apprentissage en exposant notamment les réseaux de neurones. Nous nous concentrons, par la suite, sur un outil mathématique qui a servi de base à nos travaux : les Machines à Vecteurs de Support et nous détaillons notre contribution quant à leur application pour les problèmes de détection et de suivi des visages dans le contexte de surveillance vidéo.

3.1 Notions de théorie d'apprentissage

3.1.1 Généralités

Si l'homme est naturellement doué d'intelligence, il le doit principalement à sa capacité d'apprentissage. Depuis son enfance il apprend à lire, à écrire, à composer à partir ce qu'il a appris afin de lui même créer, développer et intégrer les différents éléments de l'univers qui le composent. A titre d'exemple, un enfant apprend à distinguer et même à reconnaître la voix de ses parents par rapport aux autres voix. Depuis que les ordinateurs sont apparus, l'homme n'a de cesse d'essayer de leur injecter cette capacité "d'apprentissage" et les termes d'apprentissage automatique et d'apprentissage machine -Machine Learning (M.L.) en anglais- ont été employés depuis.

La capacité à apprendre étant reconnue comme une disposition essentielle dans de très nombreuses fonctions cognitives, il n'est pas étonnant que l'apprentissage devienne un sujet central de l'intelligence artificielle [Mic84, MST94, Mit97, DeJ86] dès l'origine de cette discipline. La mise à plat d'une définition propre à l'apprentissage est à relier à celle de la notion de calcul dans les années 30 [Val84] : dans les deux cas il s'agit de définir

clairement une notion déjà usuelle mais dont le contenu reste souvent flou. La notion de calcul a été bien définie par la notion de machine de Turing. Une définition consensuelle de la notion d'apprentissage émerge toutefois. Généralement une telle définition doit faire intervenir un apprenant doté d'une capacité de calcul, un objet à apprendre, un protocole qui précise de quelle manière l'objet est présenté à l'apprenant ainsi qu'un critère de succès qui indique à quel moment le dit objet est appris. A partir de là on peut tenter de répondre à la question : qu'est ce qu'apprendre ? une réponse serait alors ; l'apprentissage est le processus d'adaptation des paramètres pour donner une réponse désirée à une entrée ou une stimulation quelconque.

3.1.2 Principe d'inférence inductive

Les méthodes d'apprentissage ont pour but d'identifier les classes auxquelles appartiennent des objets à partir de certains traits descriptifs et caractéristiques suivant le protocole choisi. Elles s'appliquent à un grand nombre d'activités humaines et conviennent en particulier au problème de prise de décision automatisée [CM03]. La première approche, intuitive dirons nous, consiste à employer des "systèmes experts". La connaissance de l'expert est décrite sous forme de règles qui seront ensuite utilisées pour classer d'autres nouveaux cas. Bien évidemment, de la capacité à extraire et à décrire correctement ces règles dépend les performances du système d'apprentissage. Nous considérons dans ce chapitre l'approche par laquelle la procédure d'apprentissage/classification est extraite automatiquement à partir d'un ensemble d'exemples. Un exemple consiste en la description d'un cas avec la classification correspondante. Un système d'apprentissage doit alors, à partir de cet ensemble d'exemples, extraire une procédure de classification qui, d'un échantillon de test, devra déterminer sa classe d'appartenance. Il s'agit donc d'induire une procédure de classification générale à partir d'exemples. Le problème est donc un problème d'**inférence inductive**.

La classification doit être capable d'interpoler l'entrée donnée, c'est à dire que ces méthodes doivent attribuer à une entrée qu'elles n'ont pas apprises la bonne valeur de sortie, pourvu que des exemples suffisamment proches de cette entrée soient appris lors de l'entraînement -phase d'apprentissage- et qu'il n'y ait pas de phénomène de sur-apprentissage. Par opposition au terme de sous-apprentissage qui désigne le manque d'exemples à apprendre d'où un entraînement insuffisant, le sur-apprentissage désigne le fait de sur-paramétrer le modèle adopté. Cette sur-paramétrisation entraîne un apprentissage "par cœur" des exemples et peut donner une interpolation incorrecte pour des exemples même voisins de ceux appris.

La tâche d'apprentissage est définie par rapport à ce que l'apprenant doit réaliser pour appréhender son expérience. Cela peut inclure d'apprendre à reconnaître l'expérience E -formes, parties, ...-. Lorsque l'apprentissage se fait par un professeur ou un superviseur, qui fourni un étiquetage des expériences on parle d'apprentissage supervisé. Il consiste à chercher une fonction $f : \mathcal{X} \to \{0, 1\}$, autrement dit pour chaque forme $x \in \mathcal{X}$ correspond

un étiquetage arbitraire 0 ou 1 décrivant l'appartenance ou non à la classe de concept visé. La fonction f est apprise à partir d'un échantillon d'apprentissage supposé sûr $S = (x_1, y_1), (x_2, y_2), \cdots (x_n, y_n)$, c'est à dire que les n exemples de l'échantillon d'apprentissage sont correctement étiquetés. Dans le cas où l'apprentissage s'effectue sans ce professeur on parle d'apprentissage non supervisé. Ceci dit l'apprentissage peut également consister à apprendre à prédire ce qui implique nécessairement une dépendance de causalité, voire même temporelle, de l'expérience E.

3.1.3 Formalisation du problème d'apprentissage supervisé

Suivant la littérature et la nature des problèmes considérés, la définition du problème d'apprentissage supervisé diffère. Il est intéressant de mettre en avant les différentes notations et propositions de définition afin de les confronter entre elles. Ceci dit toutes ces définitions font appel à certaines suppositions. Supposons l'existence d'une base d'apprentissage, ou de façon identique d'un échantillon d'apprentissage, alors

- Les méthodes d'apprentissage par inférence font appel au fait que les échantillons à venir soient proches -sans pour autant être identiques- des échantillons d'apprentissage.
- Puisque le cœur du problème est défini comme un modèle probabiliste, la relation entre données passées et futures suppose que tous les échantillons soient distribués de façon identique et indépendamment les uns des autres -i.i.d.-.
- On suppose que le processus d'apprentissage est cohérent, c'est à dire que plus le processus d'apprentissage apprend d'échantillons, plus il converge vers les sorties optimales. En d'autres termes plus "le processus d'apprentissage apprend, meilleur il doit être".

Le problème d'apprentissage peut se décrire de la façon suivante. Un programme apprend l'expérience E en fonction de certaines classes de tâches T et réalise une mesure de performance P, si cette performance s'améliore au fur et à mesure de l'expérience E [Mit97]. La tâche T dépend du problème que l'on considère : reconnaissance et classification de formes dans une image par exemple. La mesure de performance correspond, pour ces exemples, au taux de classification correcte. L'expérience E, quand à elle, correspond à une base de données de formes à reconnaître.

D'un point de vue statistique, l'apprentissage supervisé considère trois variables aléatoires $X|T$, et Z de dimensions respectives n, K et N et un système d'entrée - sorties S qui à partir des entrées X et Z produit la sortie T. Une illustration est donnée sur la figure (15) [Fri95, STC97]

Le système réalise l'application

$$t_k = h_k(x_1, \cdots, x_n; z_1, ..., z_K) \, \forall k = 1, \cdots, N$$

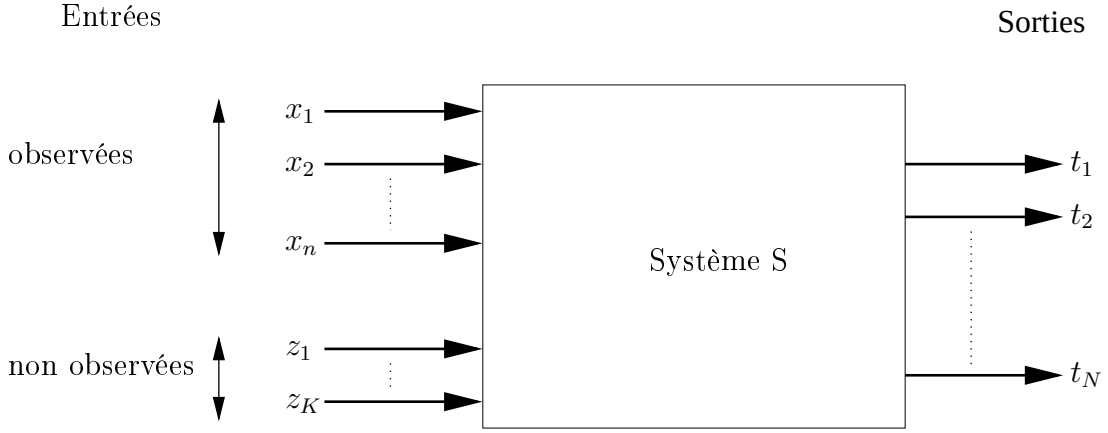


Fig. 15. Système d'entrées sorties selon Friedman [Fri95]

Ici x correspondent aux données observées, z les données inconnues et t les classes de ces données. Or comme seules les entrées x_i sont observables, on modélise le système S par un modèle statistique de la forme

$$t_k = g_k(x_1, \cdots, x_n) + \epsilon_k \quad \forall k = 1, \cdots, N$$

où g_k est la fonction de sortie du système d'apprentissage, ϵ_k est le bruit qui décrit le degré "d'ignorance" de la variable Z suivant la distribution de probabilité P_ϵ inconnue.

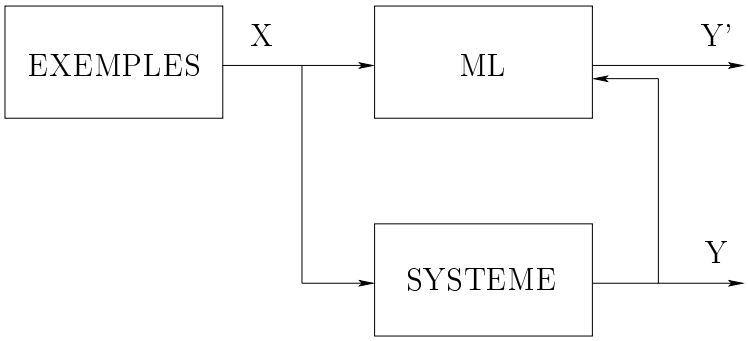


Fig. 16. Schéma d'apprentissage selon Vapnik [Vap95]

Vapnik [Vap95] donne la définition suivante de l'apprentissage supervisé pour des problèmes de classification à 2 classes. Soit un espace d'entrée $\mathcal{X}$ et un espace de sortie $\mathcal{Y}$. Considérons le cas de la classification binaire, nous choisissons alors $\mathcal{Y} \in \{-1, 1\}$. Formellement nous supposons que les couples $(X, Y) \in \mathcal{X} \times \mathcal{Y}$ sont des variables aléatoires distribuées suivant une loi de probabilité P. Observons une séquence de couples (X_i, Y_i) indépendamment distribués et de façon identique -i.i.d.- suivant P, le but de l'apprentissage consiste à établir une fonction $g : \mathcal{X} \to \mathcal{Y}$ qui puisse prédire Y à partir de X. A partir de

là il s'agit de définir un critère qui établit comment choisir cette fonction g. Ce critère minimise l'erreur ou risque R défini par $R(g) = P(g(X) \neq Y)$. Une définition plus générique sera donnée dans la section suivante. La figure (16) présente le schéma d'apprentissage selon Vapnik. A partir d'une base d'exemples X il s'agit pour la machine d'apprentissage ML, de fournir une réponse Y' la plus proche possible de celle fournie par l'étiquetage Y du système.

A partir d'un ensemble d'échantillons, il s'agit donc d'inférer un processus de classification dont l'erreur de classification serait minimale, c'est à dire où la probabilité qu'un échantillon -exemple- tiré au hasard soit mal classé par le processus soit également minimale. On s'intéressera donc à extraire une règle générale à partir de données observées. La procédure générée devra classifier correctement l'ensemble d'échantillons d'apprentissage mais surtout avoir un bon pouvoir prédictif de classification pour d'autres échantillons n'appartenant à l'ensemble initial. Or l'apprenant n'a qu'un ensemble d'échantillons fixé comme données d'apprentissage. Ceci implique donc que le langage de description soit suffisamment riche pour permettre une prédiction. On dit alors que le langage de représentation a un "pouvoir prédictif".

Définition 3.1.1. *Soit S un échantillon d'apprentissage et C un processus de classification, le taux d'erreur apparent -ou risque empirique- sur S est le rapport $E_{app}(C) = \frac{err}{card(S)}$ où err est le nombre d'exemples de S mal classés.*

D'un point de vue historique, la théorie de l'apprentissage peut se découper en quatre étapes majeures résumées dans le livre de Vapnik [Vap95] :

- construction des premières machines d'apprentissage (Machine de Turing par exemple),
- construction des fondements de la théorie en elle même,
- élaboration des réseaux de neurones,
- et enfin, élaboration d'alternatives aux réseaux de neurones

En 1943 McCulloch et Pitts [MP43] proposent pour la première fois une définition d'un neurone formel. Les percepts ou concepts ont été physiquement représentés dans le cerveau par l'entrée en activité de façon simultanée d'une assemblée de neurones [Heb49] en 1949 : La *loi de Hebb* indique que si deux neurones entrent en activité simultanément alors ils doivent être associés, autrement dit leur contact synaptique va être renforcé.

Dans les années 60, Rosenblatt [Ros62, Ros59] a suggéré le premier modèle de machine d'apprentissage supervisé appelé perceptron. D'un point de vue conceptuel, l'idée du perceptron n'était pas nouvelle en soi et faisait partie de la littérature des neurophysiologistes depuis des années. Rosenblatt a décrit, dans [Ros62], le modèle du perceptron comme un programme et démontra que son modèle pouvait être généralisé. L'algorithme du perceptron fut développé afin de résoudre des problèmes de reconnaissance de formes, c'est à dire dans le but d'établir une règle de décision qui sépare les données en deux catégories différentes à partir d'exemples donnés.

La même année, Bernard Widrow développa un algorithme qui devait ensuite devenir célèbre sous le nom de règle de Widrow-Hoff, ou règle *Least Mean Square* [WH60].

3.1.4 Apprentissage automatique et réseaux de neurones

L'enjeu de cette partie consiste à présenter un des outils majeurs utilisés pour la classification supervisée : les réseaux de neurones [Fau94]. Largement étudiés dans la littérature, les réseaux de neurones sont aussi employés pour les problèmes de détection/reconnaissance de formes [Bis95, RBT99, GD02], de parole [Ben93], d'analyse financière [Har95, MBF91], d'extraction de connaissances et datamining [Big96], de détection d'intrusion réseau - Network Intrusion Detection System NIDS- [Ric99, LD01], etc.

Suivant la définition donnée par le DARPA [Per87] un réseau de neurones est un système composé de plusieurs unités de calcul simples fonctionnant en parallèle, dont la fonction est déterminée par la structure du réseau, la solidité des connexions, et l'opération effectuée par les éléments ou nœuds.

L'approche de la présentation adoptée ici est certes classique mais permet de décrire les réseaux de neurones suivant un ordre chronologique et laisse entrevoir les percepts des séparateurs à marges.

Le neurone formel de McCulloch et Pitts

Le neurone formel fut introduit dès les années 40 par McCulloch et Pitts [MP43] dans lequel ils modélisaient de façon simple le neurone biologique. Le comportement de ce dernier était décrit en trois points :

- le neurone reçoit l'influx d'autres cellules nerveuses via les "dentrites"
- l'intensité de l'influx est pondérée par des "poids" dits synaptiques
- la cellule s'active et envoie à son tour un influx sur "l'axone" pour atteindre d'autres cellules.

L'influx parcourant les dentrites ainsi que les poids synaptiques valuant les connexions sont ainsi modélisés par des variables. Soit $x = (x_1, \cdots, x_n) \in X \subset \mathbb{R}^n$ l'influx arrivant à la dentrite i et w_i son poids associé. Les vecteurs x_i composent le "vecteur d'entrée" et $w = \{(w_1, \cdots, w_n) \in W \subset \mathbb{R}^n$ est le "vecteur de poids" de la cellule. Un neurone est ensuite caractérisé par le type d'entrées fournies et de sortie à fournir, des poids, d'un seuil b, et enfin d'une fonction d'activation f. L'algorithme ensuite se décompose de la façon suivante :

- calcul de la somme pondérée des entrées : $\sum_{i=1}^{n} w_i x_i = w^t x$,
- calcul de la fonction de sortie de la cellule : $s = f(\sum_{i=1}^{n} w_i x_i)$,
- seuillage de la précédente fonction suivant b

Le modèle de Rosenblatt

Pour construire une règle de décision binaire, le perceptron s'appuie sur les propriétés du modèle de neurone le plus simple. Chaque neurone est décrit par le modèle de McCulloch-Pitts [MP43] dans lequel un neurone a n entrées $x = (x_1, \cdots, x_n) \in X \subset \mathbb{R}^n$ et une seule sortie binaire cette fois ci $y \in \{-1, 1\}$.

La sortie est connectée aux entrées suivant une fonction de dépendance de la forme

$$y = sign\left[(w.x) - b\right],$$

où $(u.v)$ est le produit scalaire entre deux vecteurs, b est un seuil et la fonction $sign(x)$ retourne 1 si $x > 0$, -1 sinon. Schématiquement on peut représenter le neurone formel de McCulloh-Pitts par la figure (17).

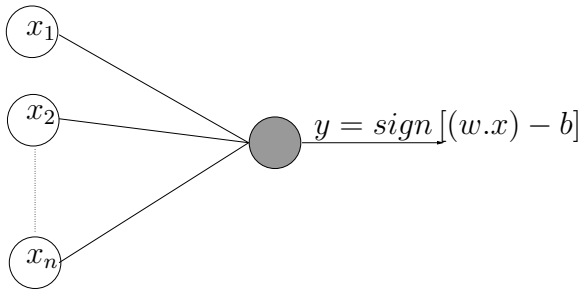


Fig. 17. Perceptron de Rosenblatt avec seuil

D'un point de vue géométrique les neurones, selon Rosenblatt, divisent l'espace d'entrée X en deux régions : une première région où la sortie y prend la valeur 1 et une région où y vaut -1. Ces deux régions sont ainsi découpées par l'hyperplan $(w.x) - b = 0$. Les vecteurs w et b déterminent la position de l'hyperplan séparateur grâce à la phase d'apprentissage pendant laquelle le Perceptron détermine les poids appropriés pour le neurone. Un exemple de séparation est proposé à la figure (18).

Rosenblatt est allé plus loin en proposant un modèle conceptuel dans lequel on ne travaille plus avec un seul neurone mais avec plusieurs neurones constituant ainsi un modèle en couche de neurones où l'entrée d'une couche d'un certain niveau reçoit la sortie de la couche précédente et ainsi de suite. Le dernier niveau est quant à lui constitué d'un seul et unique neurone. Dans ce cas le Perceptron divise l'espace d'entrée X en deux par un ensemble de surfaces linéaires. L'apprentissage consiste donc à trouver des poids appropriés pour tous les neurones à partir de l'ensemble d'apprentissage.

La façon de choisir ces poids simultanément n'est pas aisée et Rosenblatt suggéra le schéma suivant : fixer les poids de tous les neurones excepté celui du dernier, et pendant la phase d'apprentissage essayer de trouver le coefficient du dernier neurone manquant. Cette méthode suggère en fait de transformer l'espace d'entrée X en un nouvel espace Z, grâce au choix de ces coefficients, et enfin de construire l'hyperplan solution via les

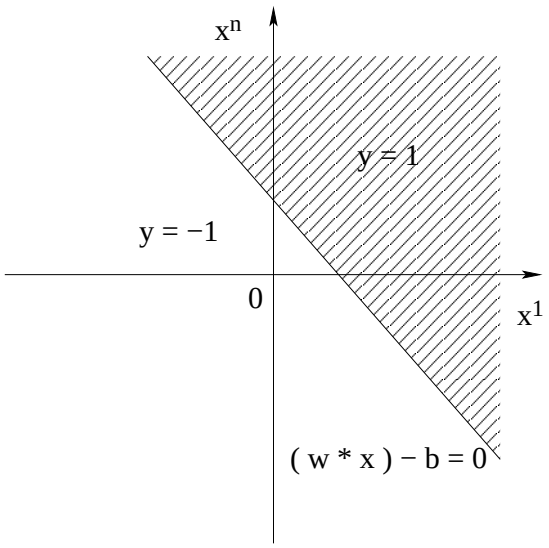


Fig. 18. Séparation linaire de l'espace d'entrée par un Perceptron

données d'entraînement dans l'espace Z. Il a ensuite étendu ce principe pour déterminer ces coefficients : soit $(x_1, y_1), \cdots (x_l, y_l)$ un ensemble d'apprentissage donné en entrée et soit $(z_1, y_1), \cdots, (z_l, y_l)$ l'ensemble d'apprentissage transformé dans Z, c'est à dire où les coefficients de x_i sont modifiés en z_i. A chaque étape de temps t, on considère un élément de l'ensemble d'apprentissage injecté dans le perceptron. Notons $w(k)$ le vecteur de coefficient du dernier neurone à l'instant t. L'algorithme proposé est le suivant :

- le vecteur w est initialisé à zéro c'est-à-dire $w(1) = 0$.
- Si le prochain exemple des données d'apprentissage z_{k+1}, y_{k+1} est correctement classifié, $y_{k+1}(w(k) \cdots z_{k+1}) > 0$, alors le vecteur de coefficients reste inchangé, $w(k+1) = w(k)$.
- Si le prochain élément n'est pas classifié correctement, $y_{k+1}(w(k) \cdots z_{k+1}) < 0$, alors le vecteur est changé de la façon suivante : $w(k+1) = w(k) + y_{k+1} z_{k+1}$,

Si les deux classes sont linéairement séparables, l'algorithme de Rosenblatt converge vers un vecteur w assurant une séparation parfaite, c'est à dire sans erreur. De plus il existe en général une infinité de vecteurs de coefficients possibles. Dans le cas où les données ne seraient pas linéairement séparables, cet algorithme oscille sans pour autant se stabiliser. Grâce à ces règles le Perceptron a démontré ses capacités de généralisation sur des exemples simples.

Comme on peut le constater le fait d'employer la fonction *sign* peut s'avérer gênant car elle présente un phénomène de discontinuité. On fait appel dans ce cas problématique à des fonctions de forme plus "lisse" .

Les prémisses de l'analyse des processus d'apprentissage

En 1962 Novikoff [Nov62] a été le premier à démontrer -ce qui marquera par la suite comme le début de la théorie de l'apprentissage- le théorème qui établit les points suivants :

- la norme des vecteurs d'apprentissage z est majorée par une constante R, c'est à dire donc que $|z| \leq R$,
- les données d'apprentissage peuvent être séparées par un hyperplan de marge ρ : $\sup_w \min_i y_i(z_i.w) > \rho$,
- L'algorithme de Rosenblatt initialisé par $w_0 = 0$ et $b_0 = 0$ converge en un temps fini après au plus $\left[\frac{R^2}{\rho^2}\right]$ corrections.

Ce théorème joue un rôle fondamental car il démontre la capacité en généralisation par principe de minimisation du nombre d'erreurs sur les exemples de la base d'apprentissage.

Novikoff prouva également que le perceptron pouvait séparer les données d'apprentissage. Il prouva entre autre que si les données étaient linéairement séparables, le perceptron pouvait, après un nombre fini de corrections, séparer n'importe quelles données d'apprentissage.

Les Réseaux de Neurones

La puissance du connexionnisme vient de l'intersection de plusieurs neurones en un réseau afin de permettre d'approximer, d'aussi près que possible n'importe quelle fonction continue et ses dérivées [HSW92]. Les réseaux de type MLP, pour Multi Layer Perceptron, ont rencontré et rencontrent encore aujourd'hui un très vif intérêt dans de nombreux domaines.

Il existe de nombreuses façons d'inter-connecter des cellules. La capacité d'un réseau de neurones à résoudre un problème dépend de la taille et de l'architecture du réseau adopté :

- les réseaux à couches,
- les réseaux à connexions latérales,
- et enfin les réseaux récurrents.

Le réseau à couches est très certainement l'architecture la plus commune. Le réseau à couches est défini comme un arbre acyclique de cellules et ces connexions sont orientées. On définit une ou plusieurs "cellules d'entrée", ainsi qu'une ou plusieurs cellules de sortie. Le nombre maximal de cellules entre les deux définit le nombre de couches du réseau.

Historiquement, en 1986 de nombreux auteurs [RHW86] ont proposé une méthode qui détermine de façon simultanée les vecteurs de coefficients pour tous les neurones grâce à une technique appelée rétropropagation - "back-propagation" en anglais- [LeC86, LeC88] basée sur la méthode dite de Descente de Gradient (Cauchy, 1847). L'idée de la méthode

consiste à propager l'erreur obtenue à une unité de sortie d'un réseau à couches comportant une ou plusieurs couches cachées à travers le réseau par descente du gradient dans le sens inverse de la propagation des activations. Pour pouvoir effectuer la méthode de descente de gradient sur l'erreur par rapport aux coefficients du réseau, la fonction de sortie d'un neurone doit être dérivable et non linéaire. Généralement on utilise la fonction sigmoïde comme présentée sur la figure 19 :

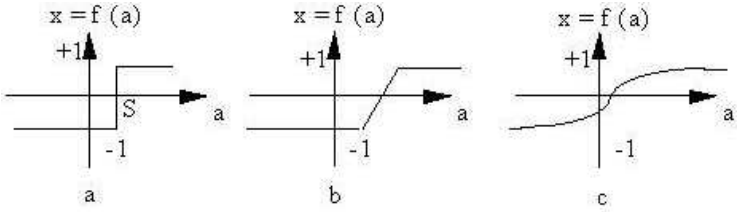


Fig. 19. Exemple de fonctions sigmoïdes retenues

$$y = S\{(w.x) - b\}$$

où $S(x)$ est une fonction monotone avec les propriétés suivantes : $S(-\infty) = -1$ et $S(+\infty) = 1$

Le perceptron multi-couche est un exemple très utilisé de réseau de neurones à architecture "feedforward" à couches, c'est-à-dire où la forme à apprendre ou à reconnaître est injectée au réseau et les premières cellules sont forcées afin de représenter la forme. L'influx se propage de couche en couche par activation des cellules. La réponse du réseau est ensuite lue sur la dernière couche. Une application au problème de détection et reconnaissance de formes a été proposé par [RBT99] par superposition en cascade de réseaux de neurones.

Le deuxième type d'architecture, proche du réseau de type "feedforward", est le réseau à connexions latérales [RMS92, Koh90]. En effet le fonctionnement reste identique mais l'activation d'une cellule est suivie d'une deuxième activation qui est envoyée vers une éventuelle couche suivante. En général le nombre de couches est faible comparativement aux autres types de réseaux de neurones. De plus les connexions latérales n'interviennent pas forcément sur toutes les couches.

Les "cartes de Kohonen" notamment [Koh82b, Koh82a, Koh90] font appel au principe suivant : Soit un ensemble de données x_i de dimension n et un réseau de N neurones disposés de la façon suivante :

- à 1, 2, ou 3 dimensions,
- selon un maillage défini : carré, hexagonal, ...,
- avec une forme définie : ligne, carré, rectangle, cube,...

Ces neurones sont dotés :

- d'un vecteur d'entrée w_i qui pointe dans l'espace des données et susceptible d'être adapté,
- d'un voisinage sur l'espace du réseau,
- de relations de compétition entre les neurones vis-à-vis des données,
- de relations de coopération entre les neurones voisins pour permettre l'adaptation.

L'objectif consiste à trouver des positions pour les vecteurs w_i dans l'espace des données telles que la topologie locale soit respectée : c'est à dire que deux neurones voisins dans l'espace de sortie aient des vecteurs voisins dans l'espace d'entrée. L'algorithme proposé par Kohonen part d'une distribution au hasard des vecteurs w_i puis effectue des tirages aléatoires de données. A chaque tirage à un instant t :

- on cherche un neurone i^* dont le vecteur d'entrée w_i est le plus proche de x, au sens euclidien par exemple, que l'on appelle "neurone gagnant". On appelle cette première étape : phase de compétition
- on adapte le vecteur w_i^* suivant la direction de la donnée x par l'expression :

$$w_{i^*}(t) = w_{i^*}(t-1) + \mu(t)[x - w_{i^*}(t-1)]$$

  avec $\mu(t)$ défini comme étant le gain d'adaptation variant de $1/t$,
- finalement, on adapte les vecteurs des neurones i dans le voisinage V_i^* du neurone i^* avec le terme :

$$w_i(t) = w_i(t-1) + \mu(t)[x - w_i(t-1)].g(\frac{d_{ii^*}}{\lambda(t)}), i \in V_{i^*}$$

  avec d_{ii^*} la distance euclidienne dans l'espace des neurones entre le neurone i et le neurone gagnant i^*, $g(.)$ une fonction de pondération en cloche, $\lambda(t)$ une distance de référence qui décroît de la même manière que $\mu(t)$.

La principale caractéristique de ces cartes et de leur processus d'organisation est la possibilité d'adaptation des poids seulement sur la région de la carte la plus active. L'utilisation du voisinage du poids le plus grand introduit des contraintes topologiques dans la représentation finale d'où le nom de connexions latérales. A la fin de l'apprentissage le poids de chaque neurone va converger pour atteindre une valeur telle qu'un neurone ne sera plus actif que pour un sous-ensemble bien précis d'éléments de la base d'apprentissage. Un des critères de sélection du neurone le plus actif, utilisé par les "cartes de Kohonen", consiste à chercher le neurone dont le vecteur de poids est le plus proche au sens de la distance euclidienne de la forme présentée [Koh82b, Koh82a]. Une application pratique de ce type de réseau à des problèmes de détection de visages a été proposé dans [BC94].

Enfin le troisième et dernier type d'architecture de réseaux de neurones est le réseau dit récurrent. Le graphe qui représente ce type de réseaux comporte une ou plusieurs boucles dans une structure qui peut tout aussi bien être partiellement ou entièrement interconnectée. Ces connexions peuvent être orientées lorsque deux cellules sont mutuellement

connectées. La politique de propagation de l'influx donné à ces réseaux est appelée "feedback neural networks" :

- la forme à apprendre est présentée au réseau, certaines cellules sont forcées,
- les cellules s'activent formant une configuration d'états d'activation

En pratique, le choix du type de réseau dépend de l'objectif que l'on souhaite atteindre. Ainsi si l'on désire faire des prédictions ou des modélisations de comportement d'un système, les réseaux de type Perceptron suffisent. Par contre, si l'on veut extraire de l'information de données ou réaliser des classifications, alors les réseaux compétitifs sont indiqués. La plupart des grands logiciels statistiques proposent les différents types de réseaux et permettent parfois de construire des réseaux hybrides. Dans tous les cas, ces logiciels proposent un réseau standard, après le choix fait par l'utilisateur du type de réseau. Il convient ensuite de remanier le réseau en modifiant les fonctions seuils, le nombre de couches et donc la configuration globale du réseau. Ensuite, il faut sélectionner un algorithme d'apprentissage et choisir le nombre d'itérations du réseau. Pour toutes ces opérations, il faut procéder par essais / erreurs et donc de façon empirique. On sait que l'on a obtenu un bon réseau capable de traiter les données d'entrée et de réaliser des prédictions de façon optimale lorsqu'on atteint le phénomène de sur-apprentissage. A un moment donné, les taux obtenus par le réseau chutent brusquement après un dernier ajustement. La configuration précédente du réseau est alors généralement la configuration optimale.

3.2 Théorie de l'apprentissage selon Vapnik

Tout au long de cette partie, nous allons considérer le problème d'apprentissage supervisé de façon plus approfondie et suivant le point de vue de Vapnik. Ensuite nous introduirons les classifieurs SVM appliqués, dans ce travail de thèse, aux problèmes de détection et de suivi de visages.

3.2.1 Formalisation du problème d'apprentissage

Considérons la formalisation suivante du problème d'apprentissage dans le cas de l'apprentissage supervisé. Supposons l'existence d'une base d'apprentissage S de taille finie. On considère l'existence de vecteurs x_i de S générés dans $\mathbb{R}^n$, indépendamment les uns des autres, et suivant une densité de probabilité fixée mais inconnue $p(x)$. Pour chaque vecteur d'entrée x, le superviseur attribue une étiquette y selon la loi $p(x, y)$ fixée mais elle-même inconnue. La base d'apprentissage S de taille m est donc constituée d'un ensemble de m couples d'observations

$$S = (x_1, y_1), (x_2, y_2) \cdots, (x_m, y_m) \in \mathbb{R}^n \times \{\pm 1\} \tag{72}$$

indépendantes et identiquement distribuées (i.i.d.) selon la loi jointe $p(x,y) = p(x)p(y|x)$. Le choix des étiquettes $\{\pm 1\}$ est purement arbitraire mais il correspond au cas de la classification binaire.

L'objectif du processus d'apprentissage est alors de rechercher une fonction f, appartenant à l'ensemble de fonctions $f(x,\alpha), \alpha \in \Lambda$ avec Λ un ensemble de paramètres abstraits, qui classifie correctement les exemples inconnus (x,y) tel que $f(x) = y$ pour des exemples (x,y) suivant la même distribution de probabilité fixée précédemment $p(x,y)$. Ceci dit si l'on ne pose aucune restriction sur l'ensemble de fonctions à partir duquel on choisit la fonction f, il en reste pas moins qu'une fonction qui réalise un "bon apprentissage" -et nous reviendrons sur ce terme dans la section suivante-, ne permet pas d'affirmer une "bonne généralisation" sur les exemples de test à venir. Afin de s'en convaincre, notons que pour chaque fonction f ainsi que pour n'importe quel ensemble de test $(\overline{x_1},\overline{y_1}),(\overline{x_2},\overline{y_2})\cdots,(\overline{x_m},\overline{y_m}) \in \mathbb{R}^n \times \{\pm 1\}$ satisfaisant $\{x_1,x_2,\cdots,x_m\} \bigcap \{\overline{x_1},\overline{x_2},\cdots,\overline{x_m}\} = \emptyset$, il existe une fonction f^* telle que $f^*(x_i) = f(x_i), \forall i = 1,\cdots,m$ mais $f^*(x_i) \neq f(x_i)$ pour $\forall i = 1,\cdots,\overline{m}$. Comme nous n'avons à notre disposition que la base d'apprentissage, il est impossible de choisir entre ces deux fonctions.

Fonction de coût et de risque

Afin de déterminer la meilleure fonction f d'approximation des réponses fournies par le superviseur, il est nécessaire d'introduire une fonctionnelle de coût $L(y,f(x,\alpha))$ entre la réponse y du superviseur pour un échantillon d'apprentissage x et la réponse $f(x,\alpha)$ fournie par la machine d'apprentissage. Dans ce cas, on définit le risque fonctionnel $R(\alpha)$ par

$$R(\alpha) = \int L(y,f(x,\alpha))dP(x,y) \tag{73}$$

Le problème revient à rechercher une fonction $f(x,\alpha_0)$ qui minimise le risque fonctionnel $R(\alpha)$. Or comme la formulation du problème d'apprentissage est vaste, la formulation de $L(y,f(x,\alpha))$ peut varier d'un problème à l'autre. Dans le cas du problème de la reconnaissance de forme ou de la classification binaire, la réponse donnée par le superviseur prend seulement deux valeurs soit $y = \{\pm 1\}$ soit $y = \{0,1\}$ et $\{f(x,\alpha), \alpha \in \Lambda\}$ est alors un ensemble de fonctions indicatrices (fonctions ne prenant que deux valeurs). La fonction de coût est dès lors définie par

$$L(y,f(x,\alpha) = \begin{cases} 0 & \text{si } y = f(x,\alpha), \\ 1 & \text{sinon} \end{cases} \tag{74}$$

La fonctionnelle (73) détermine la probabilité des différentes réponses fournies par le superviseur et par la fonction indicatrice $f(x,\alpha)$. La fonctionnelle (73) ainsi définie s'appelle erreur de classification.

La fonctionnelle (73) doit être minimisée en ne connaissant que la base d'apprentissage alors que la probabilité $P(x, y)$ est quant à elle toujours inconnue.

Le problème général d'apprentissage se résume donc à minimiser le risque fonctionnel défini par

$$R(\alpha) = \int L(y, f(x, \alpha)) dP(x, y), \alpha \in \Lambda, \tag{75}$$

Or comme le risque fonctionnel est exprimé en fonction d'une probabilité de distribution inconnue P(x,y) le principe inductif suivant doit être appliqué :

- Le risque fonctionnel $R(\alpha)$ est remplacé par le risque empirique fonctionnel défini par

$$R_{emp}(\alpha) = \frac{1}{m} \sum_{i=1}^{m} L(y, f(x_i, \alpha)) \tag{76}$$

  calculable à partir des données de l'ensemble d'apprentissage.
- Pour estimer la fonction $f(x, \alpha_0)$ qui minimise le risque (76) on met en œuvre le "principe de minimisation du risque empirique". On va ainsi chercher à minimiser le nombre d'erreurs d'apprentissage.

Dès lors se pose la question de la validité, comme méthode d'apprentissage, du principe d'induction pertinent qui minimise le risque empirique (76) et son lien avec la minimisation du risque $R(\alpha)$.

Dans sa théorie sur l'apprentissage Vapnik [Vap95] décrit quatre étapes fondamentales : étudier la consistance des processus d'apprentissage, établir les bornes sur la vitesse de convergence, contrôler la capacité en généralisation et enfin établir une théorie pour construire des algorithmes d'apprentissage.

Étude de la consistance des processus d'apprentissage

Le but de cette partie de la théorie consiste à décrire le modèle conceptuel des processus d'apprentissage qui sont basés sur le principe de minimisation structurelle. Ce point de la théorie explique comment et à quelle condition une machine d'apprentissage qui minimise le risque empirique peut également minimiser le risque de généralisation, autrement dit de déterminer les conditions nécessaires et suffisantes pour qu'un processus d'apprentissage soit "pertinent".

Définition 3.2.1. *Soit $f(x, \alpha_m)$ une fonction qui minimise le risque empirique (Eq. 76) pour un ensemble donné i.i.d d'observations $x_1, \cdots, x_m$. Un principe de minimisation du risque empirique (ERM) est dit pertinent pour un ensemble de fonctions $f(x, \alpha), \alpha \in \Lambda$ et pour la distribution de probabilité $P(x, y)$ si les suites suivantes convergent en terme de probabilité vers la même limite*

$$R(\alpha_m) \xrightarrow[m\to\infty]{P} \inf_{\alpha\in\Lambda} R(\alpha), \tag{77}$$

$$R_{emp}(\alpha_m) \xrightarrow[m\to\infty]{P} \inf_{\alpha\in\Lambda} R(\alpha). \tag{78}$$

Les deux suites, si elles convergent, convergeront vers la plus petite valeur possible du risque $R(\alpha)$. Vapnik a par ailleurs énoncé le théorème clef de la théorie de l'apprentissage par

Théorème 3.2.1. *Soit $\{f(x,\alpha), \alpha \in \Lambda\}$ un ensemble de fonctions vérifiant la condition :*

$$A \leq \int f(x,\alpha) dP(x,y) \leq B \qquad (A \leq R(\alpha) \leq B)$$

Alors une condition nécessaire et suffisante de cohérence du principe de minimisation du risque empirique est que le risque empirique $R_{emp}(\alpha)$ converge uniformément vers le risque $R(\alpha)$ sur l'ensemble de fonctions $\{f(x,\alpha), \alpha \in \Lambda\}$ c'est à dire

$$\lim_{m\to\infty} P\{\sup_{\alpha\in\Lambda}(R(\alpha) - R_{emp}(\alpha)) > \epsilon\} = 0, \forall\epsilon > 0 \tag{79}$$

La présence de la borne supérieure dans l'expression (79) indique que la cohérence du principe de minimisation du risque empirique est déterminée par la fonction la "pire" parmi l'ensemble des fonctions $\{f(x,\alpha), \alpha \in \Lambda\}$. Vapnik intitule ce pire des cas par le nom "worst case analysis"[Vap95], analyse au pire des cas. La corrélation entre le risque empirique $R_{emp}(\alpha)$ et le risque $R(\alpha)$ dépend de la taille de l'échantillon d'apprentissage, S, et, puisque celui-ci est tiré aléatoirement de façon i.i.d, de sa taille m.

Le théorème cité précédemment fournit un cadre théorique fondamental mais le risque $R(\alpha)$ reste en pratique incalculable, la distribution $P(x,y)$ restant inconnue. Pour pallier à ce problème, Vapnik introduit d'autres conditions suffisantes qui permettront à la fois d'avoir un résultat exploitable et de déduire la cohérence du principe de minimisation du risque empirique grâce à la notion de VC-Entropie.

Bornes sur la vitesse de convergence des processus d'apprentissage

Vapnik a ainsi introduit les notions de VC-entropie, de VC-entropie recuite et de fonction de croissance [Vap95, Shi93] pour aboutir à d'autres conditions pour la cohérence du principe de minimisation du risque empirique. Considérons $N = N^{\Lambda}((x_1,y_1),(x_2,y_2),\cdots,(x_m,y_m))$ le nombre minimal d'éléments distincts suffisants pour représenter l'ensemble des fonctions de coût $L(y,f(x,\alpha)), \alpha \in \Lambda$ sur l'ensemble $S = \{(x_1,y_1),(x_2,y_2),\cdots,(x_m,y_m)\}$. Considérant les fonctions indicatrices, nous avons les définitions suivantes

Définition 3.2.2. *La VC-Entropie d'un modèle notée H^{Λ} consiste en l'espérance du logarithme de la diversité de l'ensemble des fonctions $A \leq f(x, \alpha) \leq B$*

$$H^{\Lambda} = E\left[\ln N^{\Lambda}((x_1, y_1), (x_2, y_2), \cdots, (x_m, y_m))\right] \tag{80}$$

Définition 3.2.3. *La VC-Entropie recuite d'un modèle notée H_r^{Λ} est le logarithme de l'espérance de la diversité de l'ensemble des fonctions $A \leq f(x, \alpha) \leq B$*

$$H_r^{\Lambda} = \ln E\left[N^{\Lambda}((x_1, y_1), (x_2, y_2), \cdots, (x_m, y_m))\right] \tag{81}$$

Définition 3.2.4. *La fonction de croissance notée $G^{\Lambda}(m)$ est la fonction supérieure sur l'ensemble des échantillons S de la diversité de l'ensemble des fonctions que le modèle peut réaliser sur un échantillon de taille donnée*

$$G^{\Lambda}(m) = \sup_{S} \quad N^{\Lambda}((x_1, y_1), (x_2, y_2), \cdots, (x_m, y_m)) \tag{82}$$

Vapnik a introduit, à partir de ces définitions, une condition suffisante de la cohérence grâce à la VC-Entropie de l'ensemble des fonctions $\{f(x, \alpha), \alpha \in \Lambda\}$.

$$\lim_{m \to \infty} \frac{H^{\Lambda}(m)}{m} \leq \epsilon, \forall \epsilon > 0 \tag{83}$$

Cette condition signifie également que la diversité de l'espace de fonctions sur les exemples doit croître moins vite que la taille de la base. Ceci dit nous n'avons pas plus d'information sur la vitesse de convergence de la suite de risque $R(\alpha_m)$ vers le plus petit risque possible. La notion de VC-entropie recuite et de fonction de croissance intervient dès lors. Ces trois quantités, (VC-entropie, VC-entropie recuite et fonction de croissance) peuvent être rangées de façon croissante :

$$H^{\Lambda}(m) \leq H_r^{\Lambda}(m) \leq G^{\Lambda}(m) \tag{84}$$

On dit que la convergence des risques $R(\alpha_m)$ vers le plus petit risque possible, quand m tend vers l'infini, est une convergence rapide s'il existe une constante c et un entier m_0 tels que, pour tout $m > m_0$ on ait la borne exponentielle suivante

$$P\{R(\alpha_m) - \inf_{\alpha \in \Lambda}(\alpha) > \epsilon\} < e^{-c\epsilon^2 m} \tag{85}$$

Une borne peut être trouvée dès lors que les conditions suivantes sont respectées

$$\lim_{m \to \infty} = \frac{H_r^{\Lambda}(m)}{m} = 0 \tag{86}$$

Dans ce cas en effet il est justifié d'utiliser le principe inductif ERM pour choisir une hypothèse à partir d'un échantillon de données. Mais comme précédemment cette condition est suffisante pour la convergence rapide du principe de minimisation du risque empirique mais H_r^Λ n'est pas calculable en pratique. Comme la fonction de croissance ne fait pas intervenir de terme d'espérance et qu'elle n'est plus dépendante de la distribution de probabilité $P(x, y)$, on peut progresser vers une condition de convergence rapide

$$\lim_{m\to\infty} = \frac{G^\Lambda(m)}{m} = 0 \tag{87}$$

Concrètement le calcul de la fonction de croissance suppose une connaissance parfaite de l'ensemble des échantillons ce qui n'est possible que lorsque la taille de l'échantillon est réduite.

Vapnik et Chervonenkis ont ensuite étendu leurs travaux aux fonctions à valeurs réelles et ont finalement abouti au fait que les bornes de convergence obtenues sont valables, dans les conditions de convergence citées précédemment, lorsque certaines caractéristiques de la diversité des fonctions de coût croissent moins vite que la taille de la base d'exemples. Ils ont par ailleurs démontré le théorème principal s'appuyant sur la notion de VC-dimension du modèle d'apprentissage

Définition 3.2.5. *La VC-dimension d'un ensemble de fonctions $\{f(x, \alpha), \alpha \in \Lambda\}$, notée $VCdim(f)$ est égale au cardinal d du plus grand ensemble $D \in \{f(x, \alpha)\}$ capable de "pulvériser" l'espace de fonctions.*

Définition 3.2.6. *Un ensemble $\{f(x, \alpha), \alpha \in \Lambda\}$ est "pulvérisé" par D si tout étiquetage de D peut être décrit par au moins une fonction f de l'ensemble de fonctions $\{f(x, \alpha)\}$.*

Notons par ailleurs qu'on peut démontrer que la VC-dimension d'un espace R^n est égale à $n + 1$.

Théorème 3.2.2. *Pour que le principe de minimisation du risque empirique soit pertinent, indépendamment de la distribution de probabilité des exemples, il suffit que l'ensemble des fonctions $\{f(x, \alpha), \alpha \in \Lambda\}$ que le système d'apprentissage est capable d'implémenter possède une VC-dimension finie.*

La figure (20) donne une illustration du principe de minimisation structurelle.

Considérant les fonctions de coût bornées $A \leq f(x, \alpha) \leq B$, où A et B sont des constantes réelles, on peut affirmer alors que la relation suivante est vérifiée avec une probabilité au moins égale à $1 - \eta$ avec $\eta \in]0; 1]$ pour toutes les fonctions $f(x, \alpha)$ et en particulier pour celle qui minimise le risque empirique

$$|R(\alpha) - R_{emp}(\alpha)| \leq (B - A)\sqrt{\frac{h(ln(2m/h) + 1 - ln(\eta/4)}{m}} \tag{88}$$

où h est la VC-dimension finie de l'ensemble de fonctions $\{f(x, \alpha), \alpha \in \Lambda\}$.

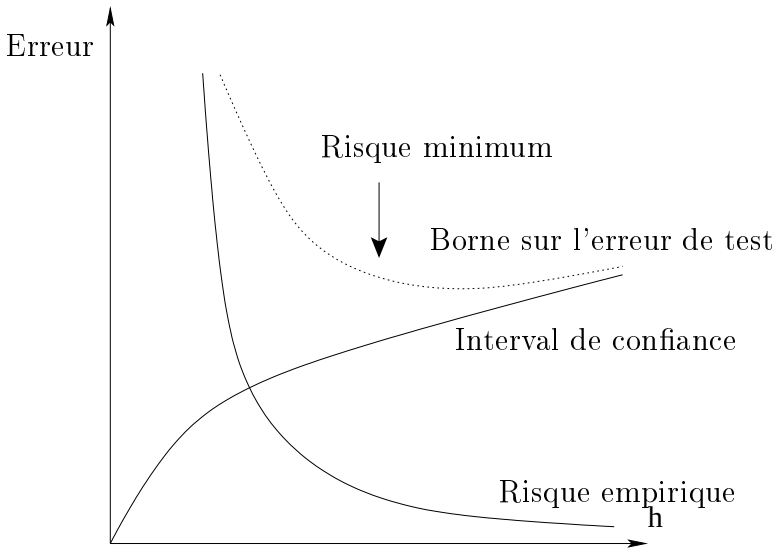


Fig. 20. Principe de minimisation structurelle

Contrôle de la capacité de généralisation

De la relation (88) précédente, on en déduit que pour tout m, l'inégalité suivante sera vérifiée avec une probabilité au moins égale à $1 - \eta$

$$R(\alpha) \leq R_{emp}(\alpha) + (B - A)\sqrt{\frac{h(ln(2m/h) + 1) - ln(\eta/4)}{m}} \tag{89}$$

Cette formulation particulière fait clairement apparaître deux termes dans la partie de droite de l'inégalité appelée **risque garanti**. Le premier terme $R_{emp}(\alpha)$ est le risque empirique nous l'avons déjà vu, alors qu'une autre entité apparaît sous la racine carrée dépendante de m/h. Elle est appelée **intervalle de confiance**. On peut remarquer que si le rapport m/h est suffisamment grand, le terme prédominant du second membre est le risque empirique. On en déduit donc que la minimisation du risque empirique suffit à contrôler la capacité de généralisation du modèle d'apprentissage : une faible valeur de risque empirique suffira à garantir une faible valeur du risque réel. Toutefois, si la taille m de l'échantillon est petite, le rapport m/h l'est également et l'intervalle de confiance prend alors une valeur importante.

Afin de minimiser le second membre de l'inégalité (89), il faut donc minimiser à la fois les deux termes qui le composent. Ceci implique donc de contrôler la VC-dimension, h du processus d'apprentissage.

Vapnik propose une méthode qui consiste à définir une *structure* particulière de l'ensemble H des fonctions $\{f(x, \alpha), \alpha \in \Lambda\}$

Définition 3.2.7. *Soit $F = \{f(x, \alpha), \alpha \in \Lambda\}$ une famille de fonctions. Une structure sur*

F est une suite de sous-ensembles emboîtés $F_i = \{f^i(x, \alpha), \alpha \in \Lambda_i\}$ tel que

$$F_1 \subset F_2 \subset \cdots \subset F_i \subset \cdots$$

et donc les VC-dimensions h_i sont finies. Ces VC-dimensions forment une suite croissante

$$h_1 \leq h_2 \leq \cdots \leq h_i \leq \cdots$$

De fait, le risque empirique décroît lorsque la VC-dimension du modèle augmente, tandis que l'intervalle de confiance est une fonction croissante de h. La borne sur le risque, c'est à dire la somme des deux termes, admet donc un minimum lorsque la valeur de h est optimale par rapport au h_i [MMR+01].

L'objectif du principe de minimisation du risque structurel [Vap82, Vap95, L. 96] consiste à chercher un compromis entre qualité de l'approximation sur l'échantillon considéré et complexité de la fonction qui réalise l'approximation.

Afin de minimiser conjointement les deux termes de la borne de l'inégalité (89), on peut soit fixer l'intervalle de confiance, soit fixer la valeur du risque empirique, à la valeur 0 par exemple, et ainsi minimiser l'intervalle de confiance.

3.2.2 Support Vector Machines

Maintenant le formalisme de l'apprentissage fixé, nous allons à travers cette section détailler comment apprendre un concept sous la forme géométrique la plus simple : l'hyperplan. A partir d'un ensemble d'exemples/contre exemples dans l'espace de représentation $\mathcal{H}$, on cherche un hyperplan qui généralise ces éléments d'apprentissage.

Plus particulièrement nous focaliserons notre attention sur un système d'apprentissage particulier : les "Support Vector Machines"[Vap95], SVM en abrégé, en vue de la recherche du meilleur hyperplan permettant de discriminer deux classes. Ses performances, robustes et remarquables, obtenues sur des bases dispersées et bruitées, ont fait des SVMs un système de choix pour de nombreuses applications. Les premières apparitions des SVMs ont été notamment proposées pour les problèmes de vision : reconnaissance de caractères [Vap95, Bur98], détection de visages [OFG97, CBPG03], identification d'objets [BSB+96], identification du locuteur [KPC01], catégorisation de texte [Joa98], suivi d'objets d'intérêt [Avi04, CBP05c].

Pour la reconnaissance de formes, que l'on peut modéliser par deux classes d'échantillons différentes, les Support Vector Machines cherchent une surface de décision optimale, déterminée par un certain nombre d'éléments de l'ensemble d'apprentissage, appelé Support Vector ou en français vecteurs de support. En projetant les données d'entrée non-linéairement séparables dans un espace de plus grande dimension appelé espace de caractéristiques (features space), les SVM peuvent résoudre des problèmes qui ne sont pas

linéairement séparables. Dans cet espace de plus grande dimension, la surface de séparation, considérée comme un hyperplan optimal de décision, est déterminée par la résolution d'un problème quadratique. Les SVMs font appel aux trois concepts mathématiques suivants :

- **Le principe de Fermat**
  Les points qui minimisent ou maximisent une fonction dérivable annulent sa dérivée. Ils sont appelés points stationnaires.
- **Le principe de Lagrange**
  Pour résoudre un système d'optimisation sous contraintes, il suffit de rechercher un point stationnaire x_0 du Lagrangien L de la fonction f à maximiser

$$L(x, \lambda) = f(x) + \sum_{i=0}^{K} \lambda_i f_i(x)$$

  où les λ_i sont des constantes appelées multiplicateurs de Lagrange fixant les contraintes.
- **Le Principe de Kuhn-Tucker** [KT51, LS01, Man69]
  Soit un problème d'optimisation d'une fonction $f : \mathbb{R}^m \to \mathbb{R}$ sous les contraintes $f_i(x) \leq 0$ pour $i = 1, ..., K$ et L son Lagrangien associé de la forme

$$L(x, \lambda) = f(x) + \sum_{i=0}^{K} \lambda_i f_i(x) \quad \text{où} \quad \lambda_i \geq 0$$

  Si un couple $(\overline{x}, \overline{\lambda})$ avec $\overline{x} \in \mathbb{R}^n$ et $\forall i \in [0, .., K], \overline{\lambda_i} \geq 0$ existe tel que $\forall x \in \mathbb{R}^m$ et $\lambda \in [0, \infty[^n$,

$$L(\overline{x}, \lambda) \leq L(\overline{x}, \overline{\lambda}) \leq L(x, \overline{\lambda}) \text{ (Point Selle)}$$

  alors $\overline{x}$ est une solution du problème d'optimisation.
  Dans le cas où f et f_i sont convexes, il est même toujours possible de trouver un point-selle $(\overline{x}, \overline{\lambda})$ qui vérifie :

$$\min_x(L(x, \overline{\lambda})) = L(\overline{x}, \overline{\lambda}) = \max_{\lambda \geq 0} L(\overline{x}, \lambda)$$

Cas de données linéairement séparables

Considérons une base d'exemples $S = (x_1, y_1), (x_2, y_2) \cdots, (x_m, y_m) \in \mathbb{R}^n \times \{\pm 1\}$ linéairement séparable. Un hyperplan séparateur est l'un des hyperplans vérifiant les conditions suivantes :

$$w.x_i + b \geq 1 \quad si \quad y_i = 1 \tag{90}$$
$$w.x_i + b \leq -1 \quad si \quad y_i = -1 \tag{91}$$

Les deux expressions ci-dessus peuvent se résumer par une description unifiée

$$y_i((w.x_i) + b) \geq 1, \quad \forall i \in \{1, \cdots, m\} \tag{92}$$

La distance $d(w, b, x)$ entre un point x et l'hyperplan (w, b) est donnée par :

$$d(w, b, x) = \frac{|w.x + b|}{||w||}$$

Définition 3.2.8. *On appelle hyperplan optimal l'hyperplan séparateur qui est situé à la distance maximale des vecteurs x_i les plus proches parmi l'ensemble des exemples appartenant à deux sous-espaces différents. On dit que cet hyperplan maximise la marge.*

L'hyperplan optimal est donné par maximisation de la marge $M(w, b)$ sous les contraintes de l'équation (92). La marge $M(w, b)$ s'écrit sous la forme suivante :

$$\begin{aligned} M(w,b) &= \min_{x_i, y_i=-1} d(w, b, x_i) + \min_{x_i, y_i=1} d(w, b, x_i) \\ &= \min_{x_i, y_i=-1} \frac{|w.x_i + b|}{||w||} + \min_{x_i, y_i=1} \frac{|w.x_i + b|}{||w||} \\ &= \frac{\min_{x_i, y_i=-1} |w.x_i + b| + \min_{x_i, y_i=1} |w.x_i + b|}{||w||} \\ &= \frac{2}{||w||} \end{aligned}$$

Plus la marge est importante, plus l'erreur attendue sera faible. Maximiser la marge $M(w, b)$ revient donc à minimiser le carré de la norme $||w||^2 =< w, w >$ du vecteur w sous les contraintes de l'équation (92). Ce qui revient à écrire

$$\text{Minimiser} \qquad \Phi(w) \qquad = \frac{1}{2}||w||^2 \tag{93}$$

$$\text{en respectant} \quad y_i(w.x + b) \geq 1 \quad \text{avec} \quad i = 1, \cdots, m \tag{94}$$

Afin de répondre à la formulation du théorème de Kuhn-Tucker, remarquons que la contrainte peut être reformulée par $-y_i(w.x + b) + 1 \leq 0$. En appliquant donc le principe de Kuhn-Tucker, la solution de ce problème est le point-selle $\overline{w}$, $\overline{b}$, $\overline{\lambda}$ du Lagrangien :

$$L(w, b, \lambda) = \frac{1}{2}||w||^2 - \sum_{i=1}^{m} \lambda_i (y_i(w.x_i + b) - 1) \tag{95}$$

avec $\lambda_i \geq 0$ les coefficients de Lagrange. Le Lagrangien L doit être maximisé suivant λ_i, et minimisé suivant w et b. Par conséquent au niveau du point-selle, les dérivées de L par rapport à chacune des variables doivent s'annuler. Dénotant $\lambda = \{\lambda_1, \cdots, \lambda_m\}$, on a

$$\begin{aligned} \frac{\partial L(w,b,\lambda)}{\partial b} &= 0 \leftrightarrow \quad \sum_{i=1}^{m} \overline{\lambda_i} y_i = 0 \\ \frac{\partial L(w,b,\lambda)}{\partial w} &= 0 \leftrightarrow \quad \overline{w} = \sum_{i=1}^{m} \overline{\lambda_i} y_i x_i \end{aligned}$$

avec $\overline{\lambda_i}$ les λ_i qui annulent les dérivées partielles. Notons que bien que la solution $\overline{w}$ soit unique du fait de la stricte convexité des 2 expressions de (93), les coefficients $\overline{\lambda_i}$, eux, ne le sont pas forcément. Suivant le Théorème de Kuhn-Tucker, seuls les coefficients de Lagrange λ_i qui sont supérieurs ou égaux à 0 au point selle répondent aux contraintes fixées par (94) [SS02].

Formellement le point selle vérifie donc la condition suivante

$$\overline{\lambda_i}[y_i(\overline{w}.x_i + \overline{b}) - 1] = 0, \quad \forall i \in \{1, \cdots, m\} \tag{96}$$

Les exemples pour lesquels $\lambda_i > 0$ sont appelés Support Vectors, Vecteurs de Support en français.

Définition 3.2.9. *Les "vecteurs de support" -support vectors- sont les vecteurs x_i de S pour lesquels l'égalité $y_i(\overline{w}.x_i + \overline{b}) = 1$ est vérifiée. Plus concrètement les vecteurs de support sont les points les plus proches de l'hyperplan optimal. Pour tous les autres exemples on a donc $\overline{\lambda_i} = 0$.*

Cette terminologie fait référence à la théorie des ensembles convexes relevant des problèmes d'optimisation convexe. Tous les autres exemples de la base sont donc inutiles dans l'expression de la solution, leurs contraintes (93) automatiquement satisfaites puisque leurs coefficients satisfont $\lambda_i = 0$. Ceci implique directement une majoration sur la capacité de généralisation des hyperplans optimaux.

Ces contraintes d'égalité permettent d'exprimer la forme duale du Lagrangien [Bur98] :

$$\text{Maximiser} \quad W(\lambda) = \textstyle\sum_{i=1}^{m} \lambda_i - \frac{1}{2} \sum_{i,j=1} \lambda_i \lambda_j y_i y_j (x_i.x_j) \tag{97}$$

$$\text{en respectant} \quad \lambda_i \geq 0 \qquad \text{avec} \quad i = 1, \cdots, m \tag{98}$$

$$\text{et} \quad \textstyle\sum_{i=1}^{m} \lambda_i y_i = 0 \tag{99}$$

Pour trouver le point-selle solution, il ne suffit plus de minimiser mais de maximiser $W(\lambda)$ en considérant $\lambda_i \geq 0$ pour tout $i \in \{1, \cdots, m\}$. La résolution de ce problème d'optimisation quadratique permettra de connaître l'équation de l'hyperplan optimale :

$$\sum_{i=0}^{m} \overline{\lambda_i} y_i (\overline{w}.x_i) + \overline{b} \quad avec \quad \overline{b} = -\frac{1}{2}\left[(\overline{w}.sv_{class+1} + \overline{w}.sv_{class-1})\right]$$

avec $sv_{class+1}$ et $sv_{class-1}$ les vecteurs de support à "gauche" (-1) et à droite (+1) de l'hyperplan.

Cas de données non séparables

A partir des conditions (90) et (91), il est possible de reformuler les conditions précédentes en introduisant des variables positives ξ_i, pour $i = 1, \cdots, m$ lorsque les données ne sont pas linéairement séparables :

$$w.x_i + b \geq 1 - \xi_i \quad si \quad y_i = 1 \tag{100}$$
$$w.x_i + b \leq -1 + \xi_i \quad si \quad y_i = -1 \tag{101}$$
$$\xi_i \geq 0 \quad \forall i \in \{1, \cdots, m\} \tag{102}$$

La machine affecte une valeur fausse si à un vecteur x_i , le ξ_i correspondant est supérieur à 1. La somme des ξ_i représente une borne sur le nombre d'erreurs à commettre. Au lieu de chercher le vecteur poids $\overline{w}$ minimisant la norme au carré (w.w), on va, suivant les conditions (100), (101) et (102), minimiser la quantité suivante :

$$\text{Minimiser} \quad \Phi(w) = \tfrac{1}{2}||w||^2 + C\sum_{i=1}^{m} \xi_i \tag{103}$$
$$\text{en respectant} \quad y_i(w.x + b) \geq 1 - \xi_i \quad \text{avec} \quad i = 1, \cdots, m \tag{104}$$
$$\text{et} \quad \xi_i \geq 0 \quad \text{avec} \quad i = 1, \cdots, m \tag{105}$$

La solution $\overline{\lambda_i}$ s'obtient de façon identique par maximisation du Lagrangien dual admettant la même expression que dans l'équation (97). Seules les contraintes changent car $0 \leq \lambda_i \leq C$, pour tout $i \in \{1, \cdots, m\}$ avec C un paramètre défini par l'utilisateur qui permet de régler la granularité de la fonction de décision. En réitérant le même processus que précédemment, on cherche la solution en considérant le point selle du lagrangien $W(\lambda)$.

$$\text{Maximiser} \quad W(\lambda) = \sum_{i=1}^{m} \lambda_i - \tfrac{1}{2}\sum_{i,j=1} \lambda_i \lambda_j y_i y_j (x_i.x_j) \tag{106}$$
$$\text{en respectant} \quad 0 \leq \lambda_i \leq C \quad \text{avec} \quad i = 1, \cdots, m \tag{107}$$
$$\text{et} \quad \sum_{i=1}^{m} \lambda_i y_i = 0 \tag{108}$$

on obtient ainsi l'équation de l'hyperplan optimal au sens des nouvelles contraintes tenant compte des ξ_i. Les vecteurs de support sont toujours les vecteurs exemples les plus proches de cet hyperplan.

Hyperplans canoniques et capacité de généralisation

La question phare finalement des Machines à Vecteurs de support est de comprendre en quoi la recherche de l'hyperplan optimal permet de répondre au problème de la "capacité" de généralisation que nous avons introduit précédemment.

Vapnik introduit la notion de normalisation des hyperplans séparateurs repris ensuite par Burges [Bur98]. Considérons l'ensemble S^* des vecteurs d'apprentissage de $\mathbb{R}^n$ contenus dans une sphère de rayon R.

$$S^* = \{x_1, \cdots, x_r\} \quad \text{avec} \quad |x_i - a| \leq R \quad \forall x_i \in S^*$$

Définition 3.2.10. *Les hyperplans canoniques pour les vecteurs $x_i \in S^*$ sont ceux qui vérifient la condition $min_{x_i \in S^*} = |(w.x_i) + b| = 1$*

Vapnik [Vap95] a démontré le théorème suivant

Théorème 3.2.3. *La VC-dimension h d'un ensemble d'hyperplans canoniques définis sur S^* et satisfaisant la contrainte $||w|| \leq A$ admet la borne suivante*

$$h \leq min(d, R^2 A^2) \tag{109}$$

La recherche de l'hyperlan qui minimise la norme de w permettrait selon toute vraisemblance de fournir un cadre où la VC-dimension serait minimale suivant le théorème précédent. Ceci dit ce théorème impose trop de conditions sur la nature de la base d'entraînement et n'est pas en pratique directement exploitable.

Une autre façon d'aborder le problème a donc été introduite -une fois de plus- par Vapnik [Vap95] afin de permettre de faire le lien entre performances en terme de généralisation et la recherche des hyperplans optimaux. De cette étude, deux théorèmes ont ainsi émergé [Vap95] :

Théorème 3.2.4. *Soit $sv(m)$ le nombre de vecteurs de support de l'hyperplan optimal séparant les m données de la base d'apprentissage. L'espérance de la probabilité d'erreur sur un exemple de test est bornée par*

$$E[P(erreur] \leq \frac{E[sv(m)]}{m-1}$$

où E dénote l'espérance mesurée sur un échantillon de taille m. Cette borne est également indépendante de la dimension de l'espace : ainsi si l'on trouve un hyperplan optimal séparant nos deux classes défini par un petit nombre d'exemples critiques, alors il devrait y avoir de bonne performance en terme de généralisation, quelle que soit la dimension de l'espace.

Théorème 3.2.5. *[Vap98, SBS99, Bur98] Si les exemples vérifient la condition* $||x_i|| \leq D$, *pour tout* $i = 1, \cdots, m$ *et s'ils admettent un hyperplan séparateur optimal, passant par l'origine, de marge* ρ, *alors l'espérance de la probabilité d'erreur admet comme borne :*

$$E[P(erreur)] \leq \frac{E[\frac{D^2}{\rho^2}]}{m} \tag{110}$$

On trouvera une démonstration de ce théorème notamment dans [Vap95]. De ce résultat on peut en déduire que si l'on sait calculer le diamètre minimal de la sphère D contenant toutes les données, alors on peut parvenir à obtenir une bonne capacité en généralisation quel que soit le nombre d'exemples. De l'équation (110) du théorème précédent on peut déduire trois raisons qui permettent d'affirmer que l'hyperplan optimal permet une bonne généralisation :

- les données attendues sont suffisamment compactes [Vap95],
- la marge attendue est grande,
- l'espace d'entrée est très petit.

Les méthodes traditionnelles ignorent, comme le fait remarquer Vapnik, les deux premières raisons pour le processus de généralisation et s'en remettent principalement au dernier aspect. Les Machines à Vecteurs de Support, elles par contre, ignorent la dimension et s'appuient sur les deux premières raisons.

Passage par un espace de redescription : les fonctions noyau

Les Support Vector Machines donnent de bons résultats lorsque les données sont linéairement séparables et leur principal intérêt réside dans le fait de contrôler facilement la capacité et donc leur pouvoir en généralisation. Dans la vie pratique, il est cependant peu probable d'avoir un jeu de données linéairement séparable et il s'agirait donc de déterminer une hypersurface de séparation non linéaire qui déterminerait la frontière entre les exemples positifs et négatifs. Or le problème de ce genre de méthode vient du fait de la paramétrisation du modèle qui devient vite trop élevé.

Par souci de généralité, les SVMs plongent les vecteurs d'entrée (input space) dans un espace de représentation (features space) interne de dimension suffisamment grande pour que la surface de séparation dans cet espace reste linéaire et que l'hyperplan optimal construit sur cet espace généralise le "mieux possible" indépendamment de la dimension de l'espace.

Les termes de "mieux possible" se réfèrent au fait que soit la dimension VC de la famille est faible, soit que cet hyperplan est défini par un petit nombre de vecteurs de support. La capacité qu'un problème puisse devenir linéairement séparable dans un espace de plus grande dimension voire même infinie trouve sa justification dans certains travaux plus particulièrement appliqués sur les réseaux de neurones [GBPM97, Gib96]. Nous noterons

Φ la fonction de transfert définie de telle façon $\Phi : \mathbb{R}^n \mapsto \mathcal{H}$ où $\mathcal{H}$ est un espace de dimension plus grande.

Dans [BBV92] une application pratique de cette proposition est rendue possible par le fait que l'on puisse construire l'hyperplan optimal dans l'espace de représentation interne sans même considérer cet espace explicitement. Il suffit pour cela d'estimer le produit scalaire entre les vecteurs de support et les vecteurs de l'espace de représentation interne. Un produit scalaire s'exprime de la façon suivante :

$$(u.v) = \sum_{i=1}^{\infty} \Phi(x)\Phi(x') = K(x, x') \tag{111}$$

où $K(x, x')$ est une fonction symétrique vérifiant les conditions de Mercer, c'est à dire

$$\int K(x, x')\psi(x)\psi(x')dxdx' \geq 0$$

pour toute fonction ψ telle que $\int \psi^2(x)dx < \infty$.

K est appelé également noyau. Pour résumer dans le cas de données non linéairement séparables, la recherche de l'hyperplan optimal consiste à remplacer les produits scalaires des vecteurs d'entrée $(x_i.x_j)$ par leur correspondance dans l'espace de représentation interne $K(x_i, x_j)$. La forme quadratique à maximiser (99) devient

$$W(\lambda) = \sum_{i=1}^{m} \lambda_i - \frac{1}{2} \sum_{i,j=1} \lambda_i \lambda_j y_i y_j K(x_i.x_j) \tag{112}$$

sous les contraintes $0 \leq \lambda_i \leq C$ pour tout $1 \leq i \leq m$. Le résultat λ^0 obtenu après maximisation permet de déterminer l'expression de la règle de décision suivante

$$\sum_{i=1}^{m} \lambda_i^0 y_i K(x, x_i) + b^0 \tag{113}$$

Finalement la fonction de décision utilise la convolution du produit scalaire pour construire les fonctions de décision non linéaires dans l'espace d'entrée. La règle de décision s'exprime sous sa forme générale par

$$f(x) = sign \left[\sum_{i=1}^{N} \lambda_i y_i K(x, x_i) + b \right] \tag{114}$$

avec N le nombre de vecteurs de support retenus pendant l'apprentissage, x_i le $i$ème vecteur de support, y_i son étiquette associée et enfin λ_i une constante déterminée pendant la phase d'apprentissage.

Il existe donc divers noyaux réalisant les condition de Mercer bien que leur nombre soit finalement limité. Nous retiendrons les trois principaux :

- **Polynomial :** Pour construire des règles de décisions modélisées par des polynômes de degré p , on met en œuvre un SVM grâce au noyau

$$K_{P^d}(x, x') = [(x.x') + 1]^p \quad p \geq 2 \tag{115}$$

- **Gaussien :** Pour obtenir une machine à base radiale on utilise le noyau suivant

$$K_G(x, x') = e^{-\frac{||x-x'||}{2\sigma^2}} \tag{116}$$

- **Sigmoïde :** Dans ce cas, l'expression du noyau s'écrit alors

$$K_{Th}(x, x') = tanh(Cx.x' - \sigma) \tag{117}$$

  avec C une constante choisie de façon à vérifier les conditions de Mercer.

- **Linéaire :** Ces noyaux correspondent au cas où l'espace d'entrée et l'espace de plongement sont les mêmes s'exprime comme

$$K_L(x, x') = (x.x') \tag{118}$$

Dans [Bur98], [SBS99] et [SS02] sont proposés les démonstrations aux problèmes d'unicité et de globalité de la solution envisagée.

3.3 Détection de visages par apprentissage supervisé

Le problème que nous allons aborder dans cette partie est celui de la détection d'un objet prédéfini dans les flux vidéo. Suite à notre étude des méthodes d'apprentissage, nous allons dans ce chapitre étudier de façon "pratique" comment les appliquer à notre contexte de surveillance vidéo et notamment à la détection de visages humains.

Après avoir dressé un tour d'horizon des espaces et méthodes retenues dans la littérature, nous présentons les méthodes que nous avons proposées ou expérimentées, à savoir la détection de visages en mono-résolution par les SVM, un schéma multi-résolution élaboré ainsi qu'une méthode dite de "boot-strapping" faisant coopérer détecteurs SVM et probabilistes à base de mélanges de lois Gaussiennes. Enfin nous présentons les aspects temps-réel par la réduction du nombre de vecteurs de support et par la combinaison détecteur de mouvement/détection de visages.

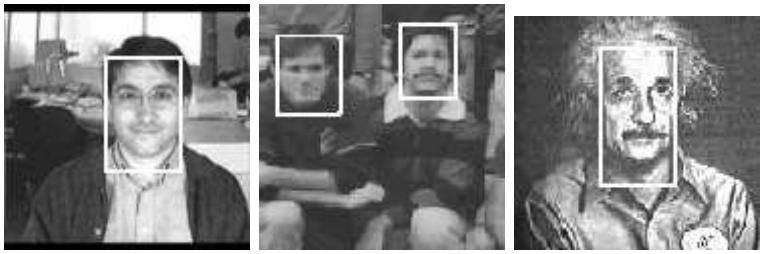


Fig. 21. Quelques exemples de détection de visages extraits du corpus MIT.

3.3.1 Problématique de la détection de visage

Définition du problème de détection

Au cours des dix dernières années, les problèmes de reconnaissance de visages et d'expression du visage ont suscité un très vif intérêt de la part de la communauté de la vision par ordinateur bien qu'ils aient été largement étudiés depuis plus de 20 ans par les neurophysiciens. Or il s'avère que le processus de détection de visages humain dans une image est un traitement majeur pour de nombreuses applications malgré la difficulté de ce problème due à la variabilité d'un visage : échelle, lieu, orientation, expressions, occultation éventuelle et les conditions d'éclairage changent l'apparence physique d'un visage.

Commençons avant tout par définir les termes de "détection de visages". Ce problème classique de reconnaissance de formes peut s'énoncer de la façon suivante : *considérant une image d'entrée quelconque, qu'elle soit scannée ou extraite d'un flux vidéo, il s'agit de déterminer la présence ou non d'un ou plusieurs visages humains dans cette image.* Dans le cas d'une détection il sera nécessaire par la suite de retourner par un codage la position spatiale de chaque visage dans l'image avec un rectangle par exemple. La Figure (21) donne quelques exemples de détection de visages sur des images prises en noir et blanc. Si ce problème de détection est déjà largement utile par lui-même, la détection de visages est, de plus, une étape préalable à la reconnaissance de visages (identification de personnes).

Dans cette étude, nous parlons de détection de visage et nous n'abordons pas le problème de la localisation du visage, le second étant un problème simplifié du premier dans lequel l'image d'entrée contient forcément un visage. En conséquence, nous nous focaliserons sur les méthodes de détection plutôt que sur celles de localisation [CTB92, MP97].

Cependant la détection de visages reste un problème difficile à traiter et ce pour deux raisons principales qui rendent le champ de détection très vaste :

– La première raison est due aux caractéristiques morphologiques propres à l'individu. Bien que la plupart des visages soient structurellement semblables avec des caractères communs (bouche, nez, oeil) placés selon une certaine configuration spatiale, il existe tout de même de grandes différences entre deux visages (forme du nez, couleur des yeux, couleur de peau). De plus certains caractères morphologiques peuvent être présents ou non selon les visages comme par exemple la moustache, la barbe,

etc. Enfin certains caractères "extérieurs" peuvent déformer des caractères morphologiques comme par exemple la teinte des lunettes modifiant partiellement la couleur de la peau. Tous ces paramètres rendent le champ de détection très variable pour un système performant.

– La deuxième raison n'implique pas les caractères propres à l'individu mais plutôt ceux de la prise de vue de la scène. Le point de vue choisi par celui qui tient l'objectif implique des échelles, des angles et des distances qui ne sont pas forcément communs à toutes les images. Ces différences peuvent complexifier la détection et impliquer de mettre en œuvre des techniques de transformations géométriques pour se ramener à un modèle proche de l'existant. De plus certaines zones du visage peuvent être visibles et d'autres cachées soit par un objet soit par un autre visage. Enfin parce que les visages sont avant tout des structures 3D dans un espace 3D, de nombreux paramètres s'ajoutent encore au problème original : des variations de luminosité (du fait de la position de la tête ou du type d'éclairage choisi), de couleur, d'ombres et de rotations éventuelles de la tête (profil, face, 3/4, etc).

Parmi les méthodes de détection, certaines sont basées sur des algorithmes d'apprentissage qui ont démontrés récemment leur efficacité par d'excellents résultats en termes de "rappel" et "précision" -cf chapitre précédent-. Force est de constater que de nombreuses métriques ont été adoptées afin d'évaluer les algorithmes tels que le temps d'apprentissage, le temps d'exécution, le ratio entre le taux de détection et le taux de fausse détection. L'évaluation devient alors plus difficile si l'on ne convient pas, à l'avance, d'utiliser un ensemble de métriques communes pour la comparaison. Dans cette thèse nous utilisons comme métriques celles de "rappel" et de "précision" telles que nous les avons présentées dans le chapitre précédent par rapport à la vérité terrain. Dans notre cas la vérité terrain représente le nombre de visages détectés par un humain dans l'ensemble des vidéos du corpus de test. Le rappel se mesure donc par le ratio entre le nombre de visages détectés par le système et la vérité terrain. La précision représente le taux de visages correctement classifiés par rapport au nombre total de visages détectés par le système.

Une région de l'image identifiée comme un visage par un classifieur est considérée comme correctement détectée si la région de l'image couvre plus qu'un certain pourcentage -environ 50% dans cette étude- du visage dans l'image. En général, les détecteurs font deux types d'erreurs : fausses détections négatives, les visages sont omis à cause des faibles taux de détection et fausses détections positives, où une région de l'image est déclarée être un visage alors qu'elle ne l'est pas.

Après avoir défini le problème de détection nous allons ensuite brièvement résumer les différents outils méthodologiques utilisés dans les différents éléments de la littérature. La description de ces méthodes doit être abordée de deux façons : premièrement décrire les espaces de représentation (modèle -template-, vecteur caractéristique -feature vector-, modèle d'apparence -appareance model-, ...) retenus puis décrire les algorithmes employés sur ces espaces de représentation.

Espaces de représentation

Espace couleur

Lorsque les images d'entrées sont en couleur, il peut être intéressant d'utiliser cette information supplémentaire afin d'isoler les régions susceptibles de contenir des visages. En effet, plusieurs auteurs ont développé et utilisé ce que l'on pourrait qualifier de détecteur de couleur de peau. Dans la majorité des cas, la peau est représentée par une portion d'un espace de couleur particulier. En utilisant les frontières de cette région comme valeur de seuillage sur une image, il est possible d'extraire les pixels dont la couleur peut s'apparenter à celle de la peau.

A partir de ce constat la majorité des espaces couleurs a été choisie pour ce problème. L'un des espaces le plus couramment utilisé pour effectuer la détection est l'espace de couleur HSV ou HSI [SP96b, SP96a]. L'avantage de cet espace HSV pour la détection de la couleur peau réside dans le fait qu'un des canaux représente la luminance (Value V). Cette particularité permet d'exprimer de façon pertinente les couleurs sans se soucier des variations de luminosité. Ainsi, les pixels "peau" sont extraits en observant seulement la teinte (Hue H) et la saturation (S) des pixels.

Dans les résultats proposés par les auteurs [SP96b, SP96a], on peut remarquer des fausses détections pour les cheveux ainsi que les régions non détectées comme par exemple le front. Cela s'explique par le fait que certaines zones reflètent davantage la lumière et semblent ainsi plus éclairées, ce qui modifie la couleur de la région dans l'espace HSV. Lorsque la saturation est basse et que la valeur V est élevée, la couleur tend vers le blanc, échappant donc aux seuils de détection. Pour ce qui est des cheveux, ils peuvent correspondre dans certains cas aux couleurs de la peau.

Ceci dit l'espace HSV n'est pas le seul a avoir été considéré dans la littérature scientifique : citons à titre d'exemple l'espace RGB [SNK96], RGB normalisé [CBBS94, SP97, OPBC97, YSMW98, QSM98], YCrCb [CN98, HS97], YIQ [DN96, DN95], YES [ST98], CIE XYZ [QCY95] et CIE LUV [YA98], ...

Arêtes du visage

La deuxième classe d'espace de représentation fréquemment utilisée est celle des arêtes/silhouettes d'un visage [KP97, AGJ98, SI95, KdVL94, CL92, YC96]. Une arête est décrite comme étant formée de points de discontinuité dans la fonction d'intensité de l image. Le principe de base consiste à reconnaître les visages à l'image à partir du modèle de contour connu au préalable grâce à différents algorithmes d'extraction de contour. Pour réaliser la détection d'un visage à partir des contours de l'image, deux méthodes sont généralement admises : la transformée de Hough et la distance de Hausdorff.

La transformée de Hough [XO93, LXK90, ZC01] est une méthode qui extrait et localise des groupes de points respectant certaines caractéristiques. La transformée de Hough est un outil permettant une représentation paramétrique de l'image à partir de sa représentation

spatiale habituelle. Le principe qui sous-tend la transformée de Hough est qu'il existe un nombre infini de lignes qui passent par un point, dont la seule différence est l'orientation (l'angle). Le but de la transformée est de déterminer lesquelles de ces lignes passent au plus près du schéma attendu.

Afin de déterminer que deux points se trouvent sur une même ligne potentielle, on doit créer une représentation de la ligne qui permette une comparaison dans ce contexte. Par exemple, les particularités recherchées peuvent être des droites, des arcs de cercle, des formes quelconques, etc. Dans un contexte de détection de visage, ce dernier est représenté par une ellipse dans la carte d'arêtes. L'application de la transformée de Hough circulaire produirait donc une liste de tous les candidats étant des cercles ou des dérivés [3, 39].

L'algorithme de base a également été modifié pour faire apparaître plusieurs variantes, dont la Randomized Hough Transform, qui peut être appliqué à la recherche de visages [ZC01]. Finalement, la transformée de Hough peut être utilisé pour détecter les yeux et les iris. Par contre, cette méthode échouera lorsque l'image est trop petite ou lorsque les yeux ne sont pas clairement visibles.

L'autre méthode couramment utilisée est la distance de Hausdorff [JKF01, HKR93]. Elle considère les arêtes comme données de base et les compare suivant un algorithme dit de "template matching". En effet, la distance de Hausdor mesure la distance entre deux ensembles de points, qui sont la plupart du temps une carte d'arêtes (image de recherche) et un modèle. L'algorithme de base effectue la recherche des meilleurs endroits de correspondance partout dans l'image (translation) et ce, pour différentes rotations. Cette recherche peut également inclure un facteur d'échelle afin de détecter des variations du modèle. A noter que cette distance est également utilisée pour comparer les régions où la couleur peau serait présente.

Forme visage

La troisième classe d'espaces de représentation possible est basée sur la structure morphologique du visage. Cette structure doit être ensuite repérée dans l'image via une méthode de correspondance et en ne retenant que celle qui se rapproche le plus de celle d'un visage. On parle de forme -shape- d'un visage. Comme un visage apparaît dans une image comme deux yeux qui sont symétriques par rapport à la ligne constituée par le nez et la bouche, il est normal de considérer des relations entre ces caractéristiques qui peuvent être représentées sous la forme de distances relatives et de positions.

On associe généralement cet espace avec celui des **rétines**. Dans plusieurs travaux -[APS94] notamment- l'espace de représentation est formé par des vecteurs constitués des valeurs de luminance, extraites séquentiellement dans des zones rectangulaires du plan image, comportant des visages. L'avantage d'un tel espace est qu'il conserve les relations spatiales-contrastes et qu'il fonctionne pour des résolutions élevées.

Principales méthodes employées

Classes de méthodes

Dans [YKA02], les auteurs résument parfaitement les différentes catégories de classes de méthodes possibles pour ce problème. Ils en distinguent quatre principales :

- Premièrement, les approches s'appuyant sur les **caractères invariants** du visage humain -feature invariant methods- utilisées principalement pour la localisation. Ces algorithmes ont pour but de déterminer les caractéristiques structurelles qui existent quels que soient la pose, le point de vue, les variations d'éclairage afin d'utiliser tous ces éléments dans la localisation de visages. On compte parmi cette catégorie les méthodes basées sur la couleur de peau, par regroupement d'arêtes, par texture, et les différentes méthodes mixant ces outils. Ces méthodes ayant déjà fait l'objet d'une étude auparavant, nous n'y reviendrons pas plus en détail.
- Deuxièmement, les méthodes dites d' **appariement de gabarits** (Template Matching) sont utilisées à la fois pour la localisation et pour la détection. Ces méthodes consistent à détecter un visage grâce à d'autres visages que l'on connaît et à établir une correspondance entre les deux. Cette approche, qui considère donc le visage dans sa globalité, utilise une image de visage standard définie manuellement ou paramétrée par une fonction. Ainsi, on donne en entrée une image aléatoire et sa valeur de corrélation avec l'image standard est calculée pour les contours, les yeux, le nez et la bouche de façon totalement indépendante. L'existence d'un visage est alors entièrement déterminée par cette valeur de corrélation. Cette méthode a, certes, le bénéfice d'être simple à implémenter, cependant elle pose problème car elle gère mal les changements d'échelle, de position, etc. D'autres modèles multi-résolutions, multi-échelles et déformables ont été proposés pour résoudre ces problèmes d'invariance.
  Les modèles ont été utilisés de nombreuses fois avec succès. La première approche que nous étudierons est celle dite des modèles prédéfinis. Ainsi si l'on utilise une collection de sous-modèles tels que les yeux, le nez, la bouche, le contour du visage définis sous forme de segments de droites. La détection se fait en deux phases : la première détermine les régions où il est probable qu'un visage apparaisse. La deuxième examine cette zone, l'analyse avec des correspondances entre lignes et enfin détermine la présence d'un visage. Govindaraju [Gov96] modélise le visage grâce à un ensemble de trois courbes, appelées : "haut", "gauche" et "droit". La détection de visages est réalisée par détection de ces trois courbes et par combinaison de ces dernières suivant leur position spatiale. Govindaraju a ensuite étendu son principe grâce à celui des forces élastiques entre les courbes du contour du visage pour améliorer l'arrangement de ces courbes. Les silhouettes [SI95] ont aussi été sources d'inspiration de modèle. Un ensemble de silhouettes de visages est obtenu en utilisant l'analyse en composantes principales (A.C.P.) sur une base d'exemples de visages dans laquelle la silhouette est représentée par un tableau de bits. Cette base de silhouettes est alors mise à contribution pour la détection.

– les méthodes basées sur la **connaissance**, utilisées principalement pour la localisation. Ces méthodes sont basées sur des règles codées via la connaissance du visage humain avec ce qui constitue un visage typique.

  De ce point de vue, la détection ne se fait plus via un modèle, mais on cherche plutôt à déterminer des règles de description entre caractères du visage. Par exemple, un visage se décompose de la façon suivante : deux yeux symétriques l'un par rapport à l'autre, un nez entre ces deux yeux et une bouche située en dessous. Ces relations sont exprimées grâce aux distances relatives et à la position des traits. Pour détecter un visage dans une image d'entrée, on extrait tout d'abord les caractères que l'on souhaite garder et les visages candidats sont extraits à partir des caractères selon des règles spécifiques. Un processus de vérification est ensuite enclenché pour éliminer les fausses détections.

  Un des problèmes de ce type d'approche réside dans la difficulté de traduire les caractères humains en règles bien définies. Si les règles sont trop strictes, il est probable que le système de détection ne puisse pas trouver un visage dans une scène. Par contre, si les règles sont trop générales, un nombre trop important de fausses détections -notion de "faux positifs". De plus, il est très difficile d'étendre ces règles aux différentes poses que peut prendre un visage, et donc de prévoir tous les cas possibles. Notons cependant qu'il existe un bon nombre d'heuristiques qui fournissent d'excellents résultats pour des images de visages pris de face.

  Yang et Huang [YH94] ont utilisé un système de connaissances hiérarchiques de détection. Leur système consiste en un ensemble de règles étalées sur trois couches. Au niveau le plus haut, tous les visages candidats possibles sont cherchés à travers l'image par une rétine en appliquant des règles relativement simples de description générale. Plus on descend dans la hiérarchie, plus les règles de description sont complexes et précises. A la fin du processus, on récupère tous les visages détectés.

  Schmid [SV99] a étudié une méthode de détection basée à la fois sur des méthodes de correspondance de modèles et sur des méthodes de détection locale et spatiale. Cette dernière utilise la notion de descripteurs locaux génériques tels les yeux, le nez et la bouche. Un visage est alors perçu comme un ensemble de descripteurs génériques et de relations spatiales entre ces descripteurs. L'algorithme de détection est alors le suivant : on sélectionne pour chaque image d'entrée une localisation commune (par exemple le milieu de l'œil gauche). Pour chaque localisation, on calcule un descripteur local, qui est un vecteur local de l'image. Schmid utilise les dérivées Gaussiennes pour décrire localement l'image. L'ensemble des descripteurs d'un caractère est utilisé pour calculer la distribution d'une variable statistique qui représente un descripteur local générique et ainsi l'on peut sauvegarder ces différents aspects. Cette distribution est estimée grâce à des modèles Gaussiens qui permettent d'appréhender les différences entre personnes d'origines différentes (asiatique, européenne, ...). Les configurations spatiales sont utilisées dans cet algorithme comme des contraintes améliorant les performances de détection. Ces contraintes sont représentées par des angles et des distances entre des descripteurs locaux génériques. En représentant ces

mêmes contraintes par des variables Gaussiennes, on peut ainsi facilement traiter les écarts dus aux différences morphologiques.

Néanmoins ces méthodes supposent une résolution des images assez importante qui permette une détection de ces traits -yeux, pupilles,...-. Dans le cas de la vidéo surveillance où la résolution des images peut être faible, une telle détection locale n'est pas très fiable car les gabarits des traits caractéristiques souffrent de la discrétisation et ne représentent que quelques pixels.

– les méthodes basées sur l'**apparence**, utilisées principalement pour la détection. A l'opposé de la méthode précédente, les modèles sont appris à partir d'une sélection d'images qui exprime la variabilité représentative de l'apparence des visages. Ces modèles ainsi créés sont ensuite utilisés pour la détection.

  Par opposition avec la correspondance de modèles, développée ci-dessus, dans laquelle ces modèles sont prédéfinis par un expert, les attributs employés dans les méthodes basées sur l'apparence sont appris depuis un ensemble d'apprentissage. La plupart de ces méthodes peuvent être comprises d'un point de vue probabiliste. Une image ou un vecteur de caractéristiques extraits d'une image est perçu comme une variable aléatoire x et cette variable détermine si nous sommes en face d'un visage ou non par la fonction de densité $p(x|visage)$ et $p(x|non-visage)$. Une autre approche fréquemment employée est de trouver une fonction de discrimination, c'est à dire soit un hyperplan de séparation comme pour les machines à vecteurs de support, soit une hypersurface pour les réseaux de neurones multi-couches ou encore une fonction de seuillage entre nos deux classes "visages" et "non-visages". Généralement, si l'image est projetée dans un espace de dimension plus petite, alors on trouve facilement cette fonction basée le plus souvent sur une distance métrique.

  Ces méthodes basées sur l'apparence opèrent généralement dans des espaces de couleur. Dans le cas de la vidéo dite naturelle -vidéo broadcast par exemple- provenant de sources hétérogènes d'acquisition, ces méthodes sont moins performantes du fait de la grande variation des couleurs dans l'espace en fonction du matériel d'éclairage, d'acquisition calibrée plus ou moins correctement, etc. De plus, dans le contexte de vidéo surveillance où les caméras sont souvent en noir et blanc, une telle approche n'est pas possible ! Elles supposent donc une certaine adaptation/normalisation.

  Une autre méthode consiste à employer des vecteurs propres pour effectuer la localisation et la reconnaissance de visages. Grâce à un réseau de neurones par exemple, comme pour Turk et Pentland [APS94], on calcule une description du visage à partir des vecteurs propres de la matrice d'auto-corrélation de l'image. Ces vecteurs propres sont communément appelés eigenfaces. Grâce à la transformation de *Karhunen-Loève*, on peut démontrer que les images contenant des visages peuvent être codées linéairement grâce à un nombre modeste d'images de bases. Considérons une collection d'images d'entraînement de $n \times m$ pixels représentées par un vecteur de taille $m \times n$, les vecteurs de base couvrent un sous-espace optimal de telle façon que la moyenne de l'erreur au carré entre la projection de l'ensemble d'apprentissage sur ce sous-espace et l'image d'origine soit minimale. On appelle l'ensemble de base des vecteurs

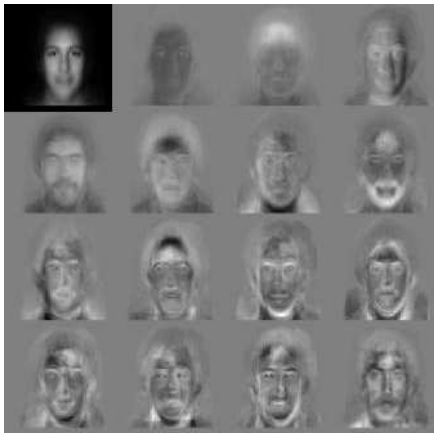


Fig. 22. Exemple d'une base d'eigenfaces.

optimaux, les "eigenpictures", puisqu'ils sont les vecteurs propres de la matrice de covariance calculée à partir des images vectorisées de l'ensemble d'apprentissage.

Turk et Pentland [APS94] ont appliqué l'analyse en composantes principales (A.C.P.) à la détection et la reconnaissance de visages. L'A.C.P. est utilisée sur l'ensemble d'apprentissage pour créer les "eigenpictures", ici appelés "eigenfaces", qui recouvrent l'espace de visages de l'espace d'images. Les images de visages sont projetées sur le sous-espace et sont ensuite rassemblées. De façon analogue, l'ensemble d'apprentissage des non-visages subit les mêmes projections et fusions que précédemment. De ce fait, la projection d'une image de test contenant un visage sur cet espace ne fait pas apparaître de réelles différences alors que la projection d'une image non-visage montre beaucoup de dissimilarités. Pour détecter la présence d'un visage dans une scène, on calcule pour chaque point de l'image, la distance entre ce point et sa projection dans le sous-espace. Un visage est détecté en évaluant le minimum local de l'espace d'image.

Ajoutons que cette technique a rapidement été étendue à la description de caractères locaux, les "eigenfeatures", comme par exemple les "eigeneyes", les "eigenmouths", les "eigennoses",.... , mais leur champ d'application est plutôt réservé à celui de la reconnaissance de visages. La figure 22 donne un exemple de base eigenfaces obtenues dans [APS94].

Méthodes d'apprentissage supervisées de détection

Comme nous l'avons vu tout au long de la description des méthodes de détection, la plupart d'entre elles s'appuient sur des méthodes par apprentissage supervisé. Ici nous allons plus spécifiquement citer les méthodes employant les principes et modèles de l'apprentissage supervisé que nous avons exposés dans le chapitre précédent.

Réseaux de Neurones

La première architecture neuronale fut introduite, pour la problématique de la détection de visages, par Agui [AKNN92] et utilise des réseaux de neurones hiérarchiques. Le sommet consiste en deux sous-réseaux parallèles dans lesquels les entrées sont les valeurs d'intensité de l'image d'origine et celle de l'image filtrée par un filtre 3*3 de Sobel, réhausseur de contours. L'entrée du second niveau correspond à la sortie des deux sous-réseaux précédents après extraction des éléments caractéristiques. La sortie de ce niveau indique la présence d'un visage dans la région indiquée. Les résultats expérimentaux tendent à prouver que ce type de méthode est capable de détecter tous les visages pourvu que toutes les images de test aient la même dimension.

Une application intéressante des réseaux de neurones à la détection de visages est proposée par Rowley [RBT99]. Sa méthode se découpe en deux phases. On applique tout d'abord un ensemble de détecteurs à base de réseaux de neurones sur une image et ensuite on utilise un superviseur pour combiner les sorties de chaque réseaux. En fait un détecteur examine l'image par blocs de différentes tailles, à la recherche d'un éventuel bloc contenant un visage. Le superviseur fusionne les résultats de chaque détecteur et élimine les détections qui se chevauchent. Le premier composant de ce système, outre le système de pré-traitement, est un réseau de neurones qui reçoit en entrée un bloc de 20×20 pixels, et génère un code de sortie de -1 à 1 qui indique respectivement l'absence ou la présence d'un visage. Pour détecter un visage contenu dans une image, on parcourt l'image avec une rétine qui découpe et retourne ces blocs de pixels. Pour détecter des visages plus larges que la rétine, l'image d'entrée est réduite par sous-échantillonage, et le détecteur est appliqué pour chaque taille.

Approche par arbre de décision

Les travaux de Fleuret et Vézien [FV99] occupent une place à part dans la détection de visages. En effet ils proposent une méthode originale de détection par arbre de décision. Les arbres de classification, plus communément appelés arbres de décision, constituent un algorithme non paramétrique standard de reconnaissance avec apprentissage.

Un arbre de décision est un arbre au sens informatique du terme. Les nœuds internes d'un tel arbre sont appelés *nœuds de décision*. Un tel nœud est étiqueté par un test qui peut être appliqué à toute description d'un exemple de la base. Les réponses possibles correspondent aux labels des arcs. Dans le cas de nœuds de décision binaires, les labels des arcs sont omis et, par convention, l'arc gauche correspond à une réponse positive au test. Les feuilles sont étiquetées par une classe appelée classe par défaut. A tout arbre de décision, on associe de façon naturelle une procédure de classification. En effet, à toute description est associée une seule feuille de l'arbre de décision. Cette association est définie en commençant à la racine de l'arbre et en descendant dans l'arbre selon les réponses aux tests qui étiquettent les nœuds internes. La classe associée est alors la classe par défaut associée à la feuille qui correspond à la description. La procédure de classification obtenue a une traduction immédiate en terme de règles de décision. Les systèmes de règles obtenus

sont particuliers car l'ordre dans lequel on examine les attributs est fixé et les règles de décision sont mutuellement exclusives. Dans notre cas, l'idée principale de l'apprentissage par arbres de décision est de diviser récursivement les exemples de l'ensemble d'apprentissage par des tests définis à l'aide d'attributs jusqu'à ce que l'on obtienne des sous-ensembles d'exemples ne contenant -presque- que des exemples appartenant tous à une même classe.

Les nombreuses applications d'arbres de décision pour la reconnaissance de caractères manuscrits ou la reconnaissance d'objets rigides, ont montré qu'ils constituaient aussi un algorithme adapté à la reconnaissance. Les deux auteurs proposent une application de cette technique à une tâche différente, la détection de visages dans une séquence vidéo. La démarche consiste à opérer une détection en ne considérant que deux classes : visage et non visage. La détection se ramène à la recherche sur l'image d'un graphe dont les sommets se positionnent sur certains points caractéristiques et dont les arêtes vérifient des contraintes géométriques. Ce graphe est recalé en plusieurs positions sur l'image, et ces différentes positions, appelées "instances", sont utilisées pour localiser le visage. La robustesse de cette technique peut être accrue si l'on combine les résultats de plusieurs arbres de décision et en tenant compte de la cohérence temporelle entre deux images successives.

La principale difficulté de cette méthode est la détection des contours. Les auteurs utilisent un recodage adaptatif des images, qui au cours de l'apprentissage construit un arbre de classification qui permet de recoder une imagette 4×4 au voisinage de chaque point en 16 codes différents. Ce recodage est ensuite utilisé par des arbres binaires de décision pour réaliser la classification proprement dite. Cette méthode a déjà été utilisée pour la reconnaissance de caractères car elle permet de déterminer automatiquement quelles sont les configurations locales les plus intéressantes (courbes, points de rebroussement,...) dans des images binaires à très faible résolution. Fleuret et Vézien ont ensuite étendu ce principe, par un algorithme de détection qui repère les bords et quantifie en 16 classes les directions des tangentes. Cet algorithme ne fait intervenir que des comparaisons entre pixels et est donc stable pour n'importe quelle transformation croissante des niveaux de gris. Ainsi les variations de l'éclairage ne modifient que faiblement l'image des contours.

Après avoir décrit les différents travaux existants, nous allons présenter tout au long des sections suivantes les méthodes que nous avons proposées. Nous avons mis en place pour cela trois façons différentes de détecter des visages. La première consiste à détecter des visages humains grâce à un classifieur SVM sur des images de taille fixe aussi bien pour l'apprentissage que pour le test. Cette méthode a fait l'objet d'une extension pour permettre de détecter des visages de tailles différentes. Nous avons donc proposé une deuxième méthode de détection de visages, suivant un schéma multi-résolution et profitant d'optimisations dans le souci de répondre aux contraintes de temps réel. Ainsi le processus de détection ne s'effectue plus cette fois sur la globabilité de l'image mais seulement sur une zone réduite de l'image : la zone de mouvement. Enfin la dernière méthode développée dans cette étude consiste en un schéma itératif d'affinement des résultats de deux détecteurs travaillant en collaboration : un détecteur par Support Vector Machine pour l'apprentissage des visages pris de face, et un détecteur à base de mélange de lois Gaussiennes en vue de l'apprentissage de la couleur de peau du visage.

3.3.2 Étude du détecteur de référence : Viola & Jones

Une des contributions majeures de ces dernières années est celle proposée par Viola et Jones [VJ04]. Cette dernière est devenue une référence en la matière aussi bien pour ses qualités scientifiques que pour ses résultats pratiques. Cette méthode de détection, temps-réel, permet la détection de visages sur des images de résolution à une cadence temporelle de 15 images/s avec des processeurs type Intel Pentium III à 700MHz. Cette prouesse est rendue réalisable grâce à 3 étapes principales que nous allons détailler par la suite.

Inspiré des travaux de Papageorgiou [CPP98], le système de détection travaille non pas suivant les niveaux d'intensité de l'image mais plutôt sur un ensemble de vecteurs caractéristiques ("features") qui ressemblent assez aux fonctions de base de Haar. Plus précisément les auteurs ont recours à trois types de vecteurs caractéristiques à base de zone rectangulaire (horizontale ou verticale) de l'image : le "two-rectangle feature" définit comme étant deux rectangles dont l'un contient la différence pixel à pixel entre la somme des pixels de l'autre région rectangulaire, le "three-rectangle feature" définit le rectangle central comme étant la différence de la somme des pixels des deux rectangles de l'extrémité, le "four-rectangle feature" énoncé comme étant la différence entre les paires de rectangles diagonaux. Un exemple est présenté dans [VJ04] qui reprend plus précisément les subtilités de chacune des topologies. L'intérêt de l'algorithme de Viola et Jones provient du fait qu'il soit indépendant de l'espace dans lequel l'image est enregistrée. Plutôt que de travailler sur un espace couleur ou dans un domaine "pixel" particulier, les auteurs proposent une nouvelle représentation de l'image appelée "Image Intégrale" [VJ04] qui permet de plonger toutes images dans un nouvel espace de dimension plus réduite. Afin de calculer le plus rapidement possible les vecteurs caractéristiques, précédemment décrits, à différentes échelles, la notion d'image intégrale prends toute sa dimension. En effet plutôt que de recalculer chacune des sommes, les auteurs stockent les sommes cumulées de chaque ligne d'une zone rectangulaire. Cette optimisation permet ainsi de rendre temps-réel la détection sur des "petits" processeurs (Intel PIII).

A partir d'un ensemble de vecteurs caractéristiques étiquetés, les auteurs utilise un algorithme de classification supervisé construit grâce à l'algorithme AdaBoost [FS95]. Il permet d'extraire un petit nombre de vecteurs caractéristiques dits critiques à partir de l'ensemble initial de vecteurs caractéristiques. Cette sélection s'opère en assignant des poids à chacun des échantillons d'apprentissage et en augmentant ces poids en fonction de "l'intérêt" que les vecteurs peuvent apporter à la classification. L'algorithme AdaBoost est un mécanisme agressif de sélection d'un petit ensemble de bonnes fonctions de classification qui n'ont néanmoins pas de diversité importante. On trouvera une description de l'algorithme AdaBoost dans [FS95] et une description de l'implémentation retenue dans [VJ04].

Enfin la dernière contribution est la méthode de combinaison des classifieurs dite "en cascade" formant ainsi un arbre de décision proche de celui décrit dans [FV99]. Suivant la même topologie, cette méthode a le bénéfice de complexifier la prise de décision au fur et à mesure que l'on descend dans l'arbre. Ainsi cette méthode permet d'éliminer rapidement les fausses détections que sont les zones de fond notamment. Les zones de

détection potentiellement intéressantes, c'est à dire où un visage peut se situer, prennent plus temps de calcul afin d'affiner les résultats de détection et d'offrir les meilleurs scores.

La cascade complète de détection de visages comprend en tout 38 étages sur quelques 6000 features finalement retenus par l'algorithme de "boosting". Sur un ensemble de test de 507 visages, la détection s'opère sur un ensemble 75 million sous fenêtres. Chaque sous-fenêtre correspond à une portion de l'image d'origine où un feature va être extrait. En moyenne les auteurs ont constaté que pour chaque sous-fenêtre 10 évaluations de vecteurs caractéristique étaient nécessaires. Malgré cela, les auteurs annoncent une détection 15 fois plus rapide que le détecteur de visage proposé par Rowley [RBT99] dans sa version la plus rapide.

L'ensemble d'apprentissage du classifieur en cascade est composé d'images extraites d'Internet et étiquetées une par une à la main. 4916 ont été étiquettées comme comportant un visage et 9500 comme étant des images sans visages. Toutes ces images ont pour résolution commune $24 * 24$ pixels. Rien que pour les images de "non-visages", les auteurs retiennent en tout 350 million sous fenêtres à l'apprentissage. Il est nécessaire, pour une machine équipée de Pentium III, d'une semaine pour réaliser l'apprentissage.

Afin d'avoir une estimation qualitative de leur détection, les auteurs ont utilisés les bases MIT et CMU telles qu'elles ont été proposées dans les travaux de Rowley. La base totale contient en tout 130 images pour 507 visages. Suivant les différentes configurations du détecteur retenu, les résultats oscillent, sur cette base, entre $88,4\%$ et $94,1\%$ en terme de rappel. A titre de comparaison les auteurs démontrent que le classifieur de Rowley -à configuration équivalente- oscilleraient entre 86% et $90,1\%$ en rappel.

3.3.3 Détection de visages de taille fixe par SVM

La détection de visages par SVM sur des images fixes n'est pas en soit une méthode nouvelle. En 1997, Osuna [OFG97] présenta des travaux pionniers dans ce domaine en proposant entre autres une solution élégante d'optimisation pour l'apprentissage et la classification de bases d'apprentissage de grande voire de très grande taille c'est à dire plus de 5000 exemples.

Les travaux que nous avons effectués ont donc naturellement pour point de départ ceux d'Osuna. Le travail réalisé consiste en la détection de visages humains en considérant leurs niveaux de gris, suivant une orientation verticale du visage et sans phénomène d'occultation. Les SVMs sont utilisées dans cette étude pour permettre de gérer le problème de la cardinalité importante de l'ensemble des visages possibles suivant les différentes formes de visages, les variations d'intensité lumineuse et éventuellement la présence d'ombre sur le visage en reprenant finalement l'espace de représentation "rétine" suivant la formulation de Pentland [APS94].

La première approche que nous avons retenue consiste à classifier des images de la même taille que celle apprise. Par conséquent la détection n'apporte aucun aspect multi-résolution

et s'avère uniquement à titre expérimentale et d'évaluation. Le système mono-résolution développé fonctionne de la façon suivante :

- Tout d'abord un classifieur SVM est entraîné sur une base d'apprentissage d'images de visages et de non-visages de taille fixe. Ces images ont pour dimension 19×19 et sont issues de la base MIT [M.I00]. Cette base comprend 6977 images pré-découpées de visages centrées sur le nez, les yeux et la bouche. Parmi ces 6977 images on compte 2429 visages et 4548 non-visages. Quelques échantillons d'apprentissage de cette base sont proposés, à titre d'exemple, à la figure (23). Pour chaque image on considère uniquement les valeurs de luminance auxquelles on associe une étiquette $+1$ si l'image représente un visage, -1 sinon. Ce classifieur est alors entraîné sur cette base avec un noyau polynomial d'ordre 2.
- Malheureusement les images du corpus d'apprentissage et de test n'ont pas toutes des conditions d'éclairage identiques. Certaines sont plus claires, d'autres plus sombres, il est donc nécessaire de toutes les pré-traiter en vue de normaliser et de réduire ces effets d'éclairage sur les valeurs de luminance. Comme tout processus d'analyse de données finalement il est important de proposer des méthodes de traitement de la base en vue de son utilisation future. Dans notre cas, nous avons recours à des méthodes de traitement d'image, mais d'autres méthodes plus "statistiques" auraient pu également être retenues comme par exemple une analyse en composantes principales.

  Pour ce faire, nous avons repris le schéma de pré-traitement proposé par Osuna [OFG97], dans lequel il distingue trois étapes principales :

  - Le **masquage** des quatres coins de l'image : cette opération a pour but de réduire la dimensionalité de l'espace d'apprentissage, de supprimer le bruit contenu à chacun des coins, et donc de n'apprendre que les valeurs de luminance appartenant d'un visage,
  - L'**égalisation d'histogramme**. Cette égalisation est réalisée sur l'ensemble des images de la base pour compenser les différences entre les images de l'intensité lumineuse. Cette méthode permet donc de réduire les effets de luminosité sur le visage tout en rehaussant également les contours.
  - et enfin la **normalisation** des valeurs d'intensité dans l'intervalle $[0; 1]$

  Les figures 23, 24 et 25 donnent quelques résultats des traitements opérés sur des images de la base CBCL [1]. Les deux premières images -de gauche à droite- sont étiquetées par le superviseur comme étant des images de visage, les trois dernières des "non-visages"
- Une fois l'étape d'apprentissage terminée, une surface de séparation est clairement déterminée entre les exemples positifs et négatifs. Le classifieur peut dès lors être utilisé sur des images de test. La base MIT comprend 24045 images de taille $19 * 19$ dont 472 visages et 23573 non-visages.

[1] http ://cbcl.mit.edu/cbcl/

Fig. 23. Images d'origine sans pré-traitement (Base d'images CBL MIT)

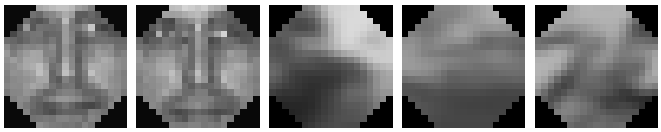


Fig. 24. Images avec luminosité normalisée et masquage des bords bruités.

Afin d'améliorer le processus, une première base de test est considérée et les mauvaises classifications obtenues sont stockées à part, puis ensuite réapprise avec l'étiquetage approprié, il s'agit de la phase dite de **ré-apprentissage**. Osuna [OFG97] l'appelle également étape de "bootstrapping" [SP94]. Les images généralement mal classifiées sont les images en présence de *texture*, comme par exemple les images de paysages, d'arbre, de bâtiment, ... Cette étape de ré-entraînement est nécessaire pour la majorité des processus d'apprentissage à partir d'exemples. Elle permet entre autre d'améliorer les résultats en terme de détection de faux-positifs. Elle permet donc de mieux considérer les exemples négatifs de l'ensemble d'apprentissage dans le sens où il est difficile de les caractériser du fait de leur abondance. Cette étape est également nécessaire pour définir au mieux les frontières entre les deux classes. Ces dernières ne sont pas de complexité égale puisque la classe des non-visages est beaucoup plus riche que celle des visages. Par conséquent, elle nécessite plus d'exemples pour définir la surface la plus précise.

3.3.4 Résultats

Chunking & SMO

Introduit par Vapnik [Vap95] pour la première fois dans le cadre de la formulation des SVM, le terme de "chunk" -tas en anglais- fait appel au fait que la forme quadratique (108) reste inchangée même si l'on élimine, dans la matrice des multiplicateurs de Lagrange, les lignes et les colonnes égales à zéro. Partant de cette remarque, il est possible alors

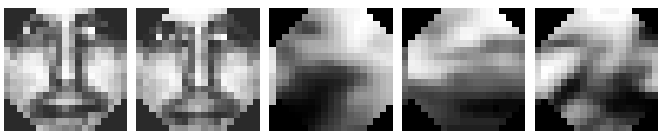


Fig. 25. Images avec luminosité normalisée, masquage des bords bruités et égalisation d'histogramme.

de simplifier le problème quadratique difficile à résoudre en une série de sous-problèmes quadratiques moins complexes. Le but de cette technique dite de "chunking" consiste alors à identifier tous les multiplicateurs de Lagrange différents de zéro et les M exemples associés qui ne respecteraient pas les conditions de KKT. Chaque sous-problème est alors initialisé avec les résultats des sous-problèmes précédents.

Chaque sous-problème créé par le processus de "chunking" à partir du problème initial est résolu en appliquant un algorithme intitulé Sequential Minimal Optimization [SBS99]. Sans forcément rentrer dans les détails de cet algorithme, il consiste à résoudre un problème quadratique sans passer par d'autres matrices de stockage et sans utiliser toute la batterie numérique de la résolution par optimisation des problèmes quadratiques. Un descriptif plus détaillé de cet algorithme est présenté dans [Vap95, SS02].

Malheureusement en contrepartie de ce gain de rapidité pendant la phase d'apprentissage, ces techniques sacrifient une partie de la précision des calculs. Le problème d'optimisation est alors résolu en plusieurs parties ce qui aboutit finalement à un patchwork de solutions. Dans le cadre de cette thèse, nous allons plus particulièrement nous attacher à comparer l'effet de ce paramètre "chunk size" sur le nombre de vecteurs de support en sortie de l'apprentissage. Si, globalement, les métriques de rappel et de précision restent inchangées avec la variation de ce paramètre, il s'avère que le nombre de vecteurs de support retenus peut varier comme le prouve le tableau 6. Le nombre de vecteurs de support pour un même noyau mais avec des paramètres de "chunk" différents peut passer de 2555 à 2134 simplement en modifiant ce paramètre.

Résultats

Lors d'expérimentation avec le classifieur par Machines à Vecteurs de Support, la première question à soulever est celle du choix du meilleur noyau et des paramètres qui lui sont associés. Nous avons retenu dans l'ensemble de nos tests l'implémentation SvmFu [M.I01] premièrement parce qu'elle était facilement portable pour notre application et qu'elle était intégrable pour nos besoins spécifiques. Aucune étude ne permet, pour le moment, de déterminer si une implémentation SVM est meilleure qu'une autre.

Pour répondre à la question "quelles sont les meilleurs paramètres pour la détection de visages à partir de la base MIT ?" nous avons proposé une batterie de tests empiriques en modifiant simplement le type de noyau retenu. Afin de résoudre le problème quadratique par des méthodes de type SMO, un certain nombre de paramètres sont à intégrer pour effectuer une expérimentation la plus correcte possible. Les paramètres de Tolérance -seuil de précision des calculs - et la valeur Epsilon -seuil à partir duquel on considère la valeur comme égale à 0- sont restés inchangés tout au long de cette première phase. Ces paramètres sont directement liées à l'implémentation SVM que nous avons retenue.

Les premiers résultats de classification ont été résumé dans le tableau 1. La colonne de gauche, intitulée configuration indique le nom de la configuration en fonction du nombre de Vecteurs de Support retenu (SVs). La colonne "Taille du chunk" indique la dimension

des sous-problèmes maximaux à résoudre. Plus la taille de ce "chunk" augmente plus le nombre de Vecteurs de Support diminue comme on peut le constater. Ceci trouve une explication logique du fait que la solution finalement proposé en augmentant le "chunk" est plus précise pour un noyau donné. Les colonnes "Rappel" et "Précision" donnent comme leur nom l'indique les taux de rappel et de précision obtenus pour une configuration donnée.

En ce qui concerne les performances en temps du classifieur SVM rappelons que tous les tests, présentés dans le cadre de cette thèse, ont été effectués sur un ordinateur de type Intel P4 1,8Ghz et 1Go de RAM. Nous avons donc considéré la base MIT qui comprend, pour rappel, 24045 images de taille 19×19 dont 472 visages et 23573 non-visages. Le tableau 2 récapitule les performances en temps de traitement total que nous avons obtenues. Le temps de classification comprend donc le temps global de classification des 24045 images de test de la base MIT auxquelles nous avons retiré le temps d'accès aux images. Il s'agit donc réellement du temps mis par le processeur pour effectuer l'opération décrite à l'équation (114) où N correspond au nombre de vecteurs de support (colonne #SVs dans notre tableau). A noyau équivalent le temps requis pour l'opération est, comme on pouvait s'y attendre, proportionnel au nombre de vecteurs de support.

Au vue des tableaux récapitulatifs 2 et 1, nous pouvons faire les constats suivants :

- Les noyaux polynomiaux, K_{P^2} d'ordre 2 -Eq. (115)-, K_{P^3} d'ordre 3 et K_{P^4} d'ordre 4, donnent des résultats bien meilleurs pour le problème de la détection de visages que les noyaux linéaires -Eq. (118). Nous rejoignons en cela la remarque d'Osuna et constatons comme il l'avait fait auparavant que l'augmentation du degré du polynôme modifie très légèrement le taux de rappel mais augmente de façon plus significative le taux de précision. Les noyaux RBF n'ont pas été retenus dans notre étude pour deux raisons : premièrement les résultats en rappel/précision dans cette étude se sont avérés très décevant -bien inférieur au noyaux polynomiaux- et deuxièmement du fait que les temps de calcul aussi bien en classification qu'en apprentissage soient beaucoup trop long -le double voir parfois le triple des noyaux polynomiaux-.
- L'élévation du degré pour les noyaux polynomiaux implique une augmentation considérable des temps d'apprentissage et de classification, qui ne se justifie pas au niveau des taux de rappel et précision. Bien évidemment le temps d'apprentissage peut être aussi long que voulu cela ne pose aucun problème. Dans notre étude les noyaux polynomiaux d'ordre 2 ont finalement les meilleurs rapports temps/performance.
- La taille du chunk influence comme on pouvait s'y attendre le nombre de vecteurs de support. Plus la taille du chunk est importante plus le nombre de vecteurs de support diminue.
- La remarque précédente est à reprendre avec le fait que plus la taille du chunk augmente plus le taux de rappel augmentent au détriment parfois du taux de précision. Certes l'amélioration n'est pas très significative mais peut faire varier de 2 à 4% les taux précédents.
- L'étape de pré-traitement des échantillons à apprendre et à classifier est nécessaire

comme pour tous les processus d'analyse de données : sans ces traitements les indices de rappel et de précision chutent de façon importante.

Configuration	Taille du chunk	Noyau	#SVs	Rappel	Précision	F-mesure
SVM_{510}	1000	K_L	510	67.6%	64.5%	66.01%
SVM_{515}	2000		515	67.4%	62.8%	65.01%
SVM_{489}	4000	⋮	489	68.6%	64.2%	66.32%
SVM_{435}	8000		435	69.1%	61.8%	65.24%
SVM_{440}	1000	K_{P^2}	440	88.8%	84.2%	86.43%
SVM_{424}	2000		424	88.6%	82.1%	88.25%
SVM_{387}	4000	⋮	387	87.8%	86.2%	86.99%
SVM_{357}	8000		357	90.9%	82.3%	86.36%
SVM_{455}	1000	K_{P^3}	455	88.4%	88.8%	88.59%
SVM_{432}	2000		432	87.8%	85.5%	86.63%
SVM_{413}	4000	⋮	413	90.3%	85.6%	87.88%
SVM_{371}	8000		371	92.0%	85.1%	88.41%

Tab. 1. Performances du classifieur SVM en fonction du noyau et des paramètres de "chunk" sur la base de visages de test du MIT.

Ce tableau prouve expérimentalement que les classifieurs SVM sont adéquats pour la détection de visages de taille fixe. Dans la partie suivante nous étudierons la possibilité d'adapter cette détection sur des vidéos naturelles avec de fortes variations dans la taille des visages.

3.3.5 Détection de visages par approche multi-résolution

Le système que nous avons décrit précédemment permet la détection de visages en considérant des images de la même taille que celle apprise [CBPG03]. Malheureusement cette situation n'arrive finalement que très rarement dans les vidéos naturelles y compris celles issues des applications de surveillance vidéo.

Dans le cadre de ce travail, nous avons donc proposé un schéma multi-résolution [CBP05c], illustré par la figure 26. Par rapport au système de détection précédent, le système de détection multi-résolution se décline en deux phases suivantes :

- **Phase 1**

  Dans un premier temps, le classifieur SVM est entraîné en considérant une base d'apprentissage d'images contenant des visages et des non-visages de taille fixe. Le choix de la taille initiale est conditionné par la prise de vue et doit correspondre aux visages de taille minimale à détecter. Afin de déterminer cette taille, nous avons exploré un ensemble de données vidéo diverses à savoir :

Configuration	Taille du chunk	Noyau	#SVs	Temps de classification (sec.)
SVM_{510}	1000	K_L	510	$120s$
SVM_{515}	2000		515	$121s$
SVM_{489}	4000	⋮	489	$115s$
SVM_{435}	8000		435	$102s$
SVM_{440}	1000	K_{P^2}	440	$145s$
SVM_{424}	2000		424	$139s$
SVM_{387}	4000	⋮	387	$127s$
SVM_{357}	8000		357	$117s$
SVM_{455}	1000	K_{P^3}	455	$242s$
SVM_{432}	2000		432	$229s$
SVM_{413}	4000	⋮	413	$219s$
SVM_{371}	8000		371	$197s$

Tab. 2. Temps de calcul global de classification par SVM en fonction du noyau et des paramètres de "chunk" sur la base de test du MIT.

- les vidéos de surveillance dans des locaux correspondant à notre configuration matérielle sur le site expérimental au LaBRI,
- les vidéos de journaux télévisés issues de la base TRECVideo 2003,
- les vidéos documentaires extraites de la base SFRS (Argos Technovision).

Empiriquement, une taille d'imagette de 30×30 pixels semble tout à fait correspondre aussi bien en terme de taille de visage minimale que de nombre d'opérations de classification à réaliser pour notre contexte. Pour chaque image, on considère uniquement les valeurs de luminance auxquelles on associe, de façon supervisée donc, une étiquette $+1$ si l'image représente un visage, -1 sinon. Ce classifieur est entraîné avec un noyau polynomial d'ordre 2, tel qu'il est défini à l'équation (115).

Chaque image du corpus d'apprentissage et de test suit le même schéma de pré-traitement que présenté pour le détecteur mono-résolution à savoir le masquage, l'égalisation d'histogramme et la normalisation des valeurs d'intensité.

- **Phase 2**

Afin de détecter des visages de taille plus importante que celle apprise, les images d'entrée I sont filtrées avec un filtre Gaussien et sous-échantillonnées

$$I_i = (g * I_{i-1}) \downarrow \quad i = 0, ..., J$$

de façon que l'image I_J au sommet de la pyramide ait une taille supérieure ou égale à la taille de la rétine apprise et I_0 soit l'image originale. Pour chaque image $I_i, 0 \leq i \leq J$ on appliquera la même méthode que celle mono-résolution.

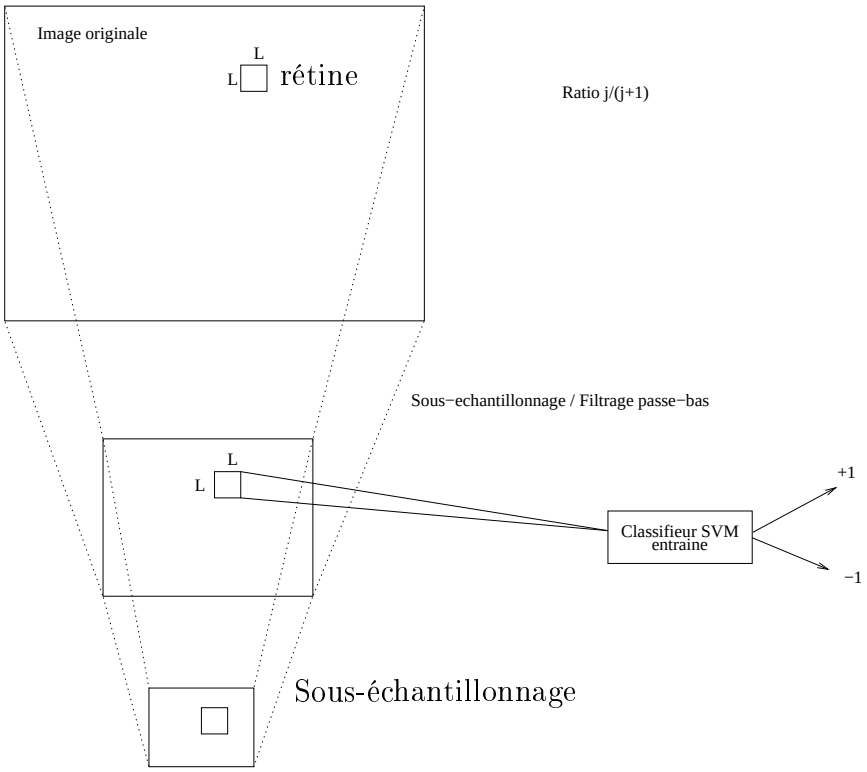


Fig. 26. Schéma multi-résolution de détection de visages.

Le schéma de détection, présenté à la figure 26, est assez similaire à celui proposé dans l'ensemble des systèmes de détections automatisées. On le rencontre notamment dans [RBT99, SP94]. A cela nous avons donc rajouté une validation croisée des résultats de détection à différents niveaux de la pyramide multi-résolution.

Cette validation, introduite dans [CBP05c], a pour but d'améliorer les taux de détection du système et de réduire le temps de classification par image I. La méthode consiste à vérifier sur les différents niveaux de la pyramide la présence d'un visage humain.

Si un visage est détecté à un niveau $j, 0 \leq j \leq J$ de la pyramide, la zone de couverture au niveau $j+1$ est éliminée temporairement de la détection. La rétine parcoura normalement l'image sous-échantillonnée -au niveau j+1- à l'exception de la zone indiquée par la position d'un visage. Cette zone sera parcourue si le temps processeur est suffisant pour effectuer la classification sur cette zone. Si un visage est détecté à plusieurs niveaux, la priorité est donnée alors à la détection de plus bas niveau -cf figure 26- de la pyramide ce qui permet d'assurer la meilleure couverture possible du visage à l'image.

3.3.6 Comparaison avec le détecteur de Viola & Jones

Il est important de signaler également les différences entre notre schéma de détection et un des détecteur de référence : celui proposé par Viola & Jones [VJ04].

Malheureusement, suite à un manque de temps dans ce travail de thèse, nous n'avons pas pu comparer nos résultats avec ceux de Viola & Jones. Dans la perspective de ce travail, notre détecteur de visages devrait être tester sur la base de test proposée par ce détecteur de référence. Ceci dit, bien que n'ayant pu réellement tester sur la même base que celle retenue par Viola et Jones, nous avons pu, par contre, comparer notre classifieur avec celui de Rowley sur une autre base de test. Ce dernier classifieur a, quand à lui, été confronté à celui de Viola et Jones sur la base MIT+CMU.

Contrairement aux travaux de Viola & Jones nous avons proposé un schéma multi-résolution, multi-échelle afin de répondre aux problèmes de scalabilité dans la taille d'un visage. Du fait des vecteurs caractéristiques choisis dans leur travaux, les auteurs n'ont pas besoin, avec une rétine, de parcourir pixel par pixel chaque niveau de la pyramide. Au contraire de leur approche, notre vecteur caractéristique reste de taille inchangée et nous n'avons pas de réel problème d'extraction des vecteurs caractéristiques de l'image comme c'est le cas avec les travaux de Viola & Jones. Même si pendant l'apprentissage, cette tâche d'extraction est faite "offline", l'extraction de ces vecteurs est une tâche complexe difficile pour un système temps-réel.

L'autre différence, évidente, vient de l'espace de représentation choisi : dans les travaux de Viola et Jones, les auteurs ont préféré un espace dit d"'image intégrale" [VJ04]. Contrairement à ces travaux nous avons préféré travailler directement sur l'espace de luminance et apprendre la distribution de ces pixels dans l'espace. Si l'on compare ces deux approches, nous avons le mérite de travailler avec beaucoup moins d'échantillons d'apprentissage pour des résultats qualitatif en terme de rappel/précision identique. Cette affirmation est possible si l'on compare les résultats obtenus par Rowley obtenu sur la base MIT+CMU.

La méthode par cascade proposée par Viola et Jones est par contre une source d'inspiration dans la perspective de ce travail de thèse afin d'améliorer notre système de classification. En effet à l'heure actuelle notre système classifie toutes les "sous-fenêtres" extraites de l'image originale à différente échelle avec la même complexité calculatoire. La méthode proposée par Viola et Jones est plus subtile et permet d'affiner ou de simplifier le processus suivant que le classifieur soit confronté à une "sous-fenêtre" visage ou non respectivement. En moyenne il doit être effectivement plus intéressant d'éliminer les nombreux "non-visages" présents afin de réduire les temps de calcul.

3.3.7 Résultats

Comme nous l'avons dit auparavant, la méthode de détection par approche multi-résolution a été testée sur les vidéos de surveillance, sur le corpus TRECVideo 2003 et sur les contenus artistiques de la base SFRS. Dans ce paragraphe nous allons présenter les

résultats quantitatifs sur la base TRECVideo 2003 et donnons des exemples de détection sur les vidéos de surveillance. Une étude exhaustive des taux de détection sur ces dernières ne peut être raisonnablement menée que lors d'expérimentations à grande échelle dans un contexte réel tout en restant dans les perspectives de ce travail.

Evaluation sur la base TRECVideo 2003

Dans l'ensemble des vidéos d'évaluation de TRECVideo 2003 nous avons extrait toutes les images clefs des plans afin de constituer une base d'apprentissage et une base de test. Pour rappel, la base utilisée comprend pour l'entraînement de notre système 10186 visages pris de face et 20586 non visages. Pour la classification nous avons regroupé 2503 visages humains et 90024 non visages. Il est important d'avoir une quantité conséquente de non visages pour tester la robustesse réelle du classifieur et notamment apprécier son taux de précision.

(a) (b)

Fig. 27. Exemples de résultats de détection de visage par Support Vector Machine extrait de vidéo de surveillance

Considérant toujours cette base pour l'apprentissage et pour la classification, nous avons établi le tableau récapitulatif -Table 3- des résultats de rappel et précision. Comme nous l'avons remarqué précédemment, l'augmentation du degré du noyau polynomial améliore légèrement les taux de rappel et de précision mais au détriment du temps de classification -cf tableau 2. Bien que la fonction de décision soit plus complexe du fait du degré du polynôme, le nombre de vecteurs de support n'est pas pour autant augmenté bien au contraire. Les résultats que nous avons obtenus pour des noyaux linéaires et Gaussiens n'ont pas permis d'aboutir à des résultats aussi pertinents que ceux ci. Nous les avons alors volontairement écartés de nos tableaux de comparaison.

Nous avons voulu ensuite évaluer les performances de notre schéma de validation multi-résolution par propagation des résultats de détection entre les niveaux de la pyramide multi-résolution. Les tableaux 4 et 5 récapitulent les résultats suite à cette étude. Le tableau 4 présente notamment les indices de rappel/précision obtenus par notre schéma de validation.

Nom	*Taille du chunk*	*Noyau*	*Tolérance*	*Epsilon*	*#SVs*	*Rappel*	*Précision*
SVM_{2510}	4000	K_{P^2}	$10E-04$	$10E-20$	2510	97.0%	67.4%
SVM_{2555}	8000				2555	97.0%	67.0%
SVM_{2583}	16000	⋮	⋮	⋮	2583	96.7%	67.2%
SVM_{2134}	32000				2134	97.0%	67.3%
SVM_{2504}	4000	K_{P^3}	$10E-04$	$10E-28$	2504	97.4%	74.4%
SVM_{2492}	8000				2492	97.4%	72.8%
SVM_{2468}	16000	⋮	⋮	⋮	2468	97.6%	73.2%
SVM_{2098}	32000				2098	97.6%	73.4%
SVM_{2472}	4000	K_{P^4}	$10E-12$	$10E-44$	2472	97.7%	75.6%
SVM_{2569}	8000				2569	97.8%	73.9%
SVM_{2512}	16000	⋮	⋮	⋮	2512	97.5%	74.4%
SVM_{2112}	32000				2112	97.9%	71.4%

Tab. 3. *Résultats de rappel et de précision d'un classifieur SVM sans validation multi-résolution avec des noyaux polynomiaux différents sur la base de test TRECVideo2003.*

Ce tableau est à comparer avec le tableau 3. On peut constater que quel que soit le noyau utilisé le taux de rappel diminue légèrement ce qui s'explique par le fait que nous privilégions le niveau le plus haut de la pyramide. Certains visages peuvent malgré tous nos efforts disparaître par notre validation, notamment lorsque deux visages sont sur deux profondeurs de champ différentes mais à des positions spatiales très proches sur l'image. Les temps de calcul seront présentés dans le paragraphe suivant.

Les figures 27, 28 et 29 présentent des résultats de détection obtenus pour un classifieur SVM sur des images extraites de la campagne d'évaluation TREC Video 2003. Le noyau polynomial d'ordre deux a été choisi pour l'entraînement du classifieur. Bien que ces images présentent des résultats de détection -cf images 28(b), 28(d), 29(b), 29(c),28(d) et 28(e)- d'un visage par image, le processus de détection permet réellement de détecter plusieurs images sur la même image. Lors de la construction de notre vérité terrain ne sont considérés comme appartenant à la classe visage que les visages pris de face c'est à dire ou il est tout à fait possible de distinguer les deux yeux, le nez et la bouche. Par exemple le visage présenté à l'image 28(a) fait partie de notre vérité terrain. Tous les éléments du visage sont visibles et pris de face. L'image 28(f) présente deux visages qui font tout deux partis de la vérité terrain. Le visage de droite est bien détecté celui de gauche ne l'est pas alors qu'il devrait l'être. L'image 28(a) présente un exemple de vrai positif -le visage du présentateur- ainsi qu'un "faux positif" -la détection de la tour en arrière plan.

L'approche par validation permet de réduire les temps de classification comme l'indique le tableau 5. Ce dernier compare les temps de classification sur l'intégralité de la base de test avec et sans validation multi-résolution. De manière générale tous les temps de classification sont réduits sans réelle perte des taux de rappel et de précision. Une optimisation de presque 30% des temps de classification est possible sur le base de test TRECVideo 2003. La méthode est donc efficace même si les images ne comportent qu'un

Nom	*Taille du chunk*	*Noyau*	*Tolérance*	*Epsilon*	*#SVs*	*Rappel*	*Précision*
SVM_{2510}	4000	K_{P^2}	$10E-04$	$10E-20$	2510	95.7%	71.2%
SVM_{2555}	8000				2555	95.2%	71.1%
SVM_{2583}	16000	⋮	⋮	⋮	2583	96.2%	71.2%
SVM_{2134}	32000				2134	96.5%	71.3%
SVM_{2504}	4000	K_{P^3}	$10E-04$	$10E-28$	2504	96.4%	75.2%
SVM_{2492}	8000				2492	96.8%	75.8%
SVM_{2468}	16000	⋮	⋮	⋮	2468	96.9%	76.3%
SVM_{2098}	32000				2098	97%	76.6%
SVM_{2472}	4000	K_{P^4}	$10E-12$	$10E-44$	2472	96.8%	76.5%
SVM_{2569}	8000				2569	96.8%	75.6%
SVM_{2512}	16000	⋮	⋮	⋮	2512	96.5%	77.1%
SVM_{2112}	32000				2112	96.9%	77.4%

Tab. 4. *Résultats de rappel et de précision d'un classifieur SVM avec validation multi-résolution avec des noyaux polynomiaux différents sur la base de test TRECVideo2003.*

seul visage de petite taille. Le meilleur des cas étant bien évidemment la présence d'un visage au premier plan occupant la majeure partie de l'image ce qui élimine une plus grande zone de l'image dans la pyramide multi-résolution.

Expériences sur les vidéos de surveillance

Nous avons également voulu tester les performances de notre détecteur sur des visages non appris bien évidemment pour différentes échelles et rotations. Cette approche consiste à tester qualitativement le détecteur de visages afin de connaître les limitations quant à la variation de pose.

Pour cela nous avons donc enregistré plusieurs séquences vidéos où seul le visage bouge à l'écran dans un premier temps. Nous allons présenter les résultats sur les séquences "Lionel", "Laurent" et "Marta" comprenant chacune 500 images. Alors que les visages ont été appris purement de face, il s'avère que le classifieur SVM est tout de même capable de détecter des visages suivant certaines rotations.

Il est important de noter que les visages présentés ici n'ont pas été appris par le classifieur et que même aucun des visages présents à l'image n'a été proposé à l'apprentissage. Ces images présentent des résultats de détection pour les séquences "Lionel" -figure (30)-, "Marta" -figure (31)- et "Laurent" -figure (32).

Les exemples présentés sur la figure (30) donnent des résultats de détection de visages sur la séquence "Lionel" par un carré blanc sur un même visage pris à des angles différents. Au vu de ces résultats, la détection du visage est robuste à une rotation d'environ 30° suivant l'axe X mais n'est pas robuste aux rotation suivant l'axe Y. Les résultats sont nettement meilleurs sur le visage de "Marta" où son visage est détecté malgré une rotation

Nom	*Taille du chunk*	*Noyau*	*#SVs*	*Temps sans validation*	*Temps avec validation*
SVM_{2510}	4000	K_{P2}	2510	$11040s$	$9052s$
SVM_{2555}	8000		2555	$11237s$	$9214s$
SVM_{2583}	16000	⋮	2583	$11361s$	$9316s$
SVM_{2134}	32000		2134	$9386s$	$7696s$
SVM_{2504}	4000	K_{P3}	2504	$15823s$	$12341s$
SVM_{2492}	8000		2492	$15747s$	$12282s$
SVM_{2468}	16000	⋮	2468	$15595s$	$12164s$
SVM_{2098}	32000		2098	$13257s$	$10340s$
SVM_{2472}	4000	K_{P4}	2472	$19712s$	$16952s$
SVM_{2569}	8000		2569	$20485s$	$17617s$
SVM_{2512}	16000	⋮	2512	$20030s$	$17225s$
SVM_{2112}	32000		2112	$16841s$	$14483s$

Tab. 5. *Résultats des temps de classification pour différents type de noyau polynomiaux sur la base de test TRECVideo 2003.*

au delà d'un angle 45°. L'expression de son visage changeant également le détecteur est suffisamment robuste pour pallier à ces variations. Les images présentées à la figure (32) donnent d'autres résultats de détection sur le visage de "Laurent". Cette séquence présente quelques fausses détections comme illustrées à l'image 32(d). Comparativement aux images précédentes, le détecteur reste robuste pour de petites variations sur l'axe Y -inférieures à 10°- tout en conservant sa capacité à détecter le visage suivant une rotation sur l'axe X.

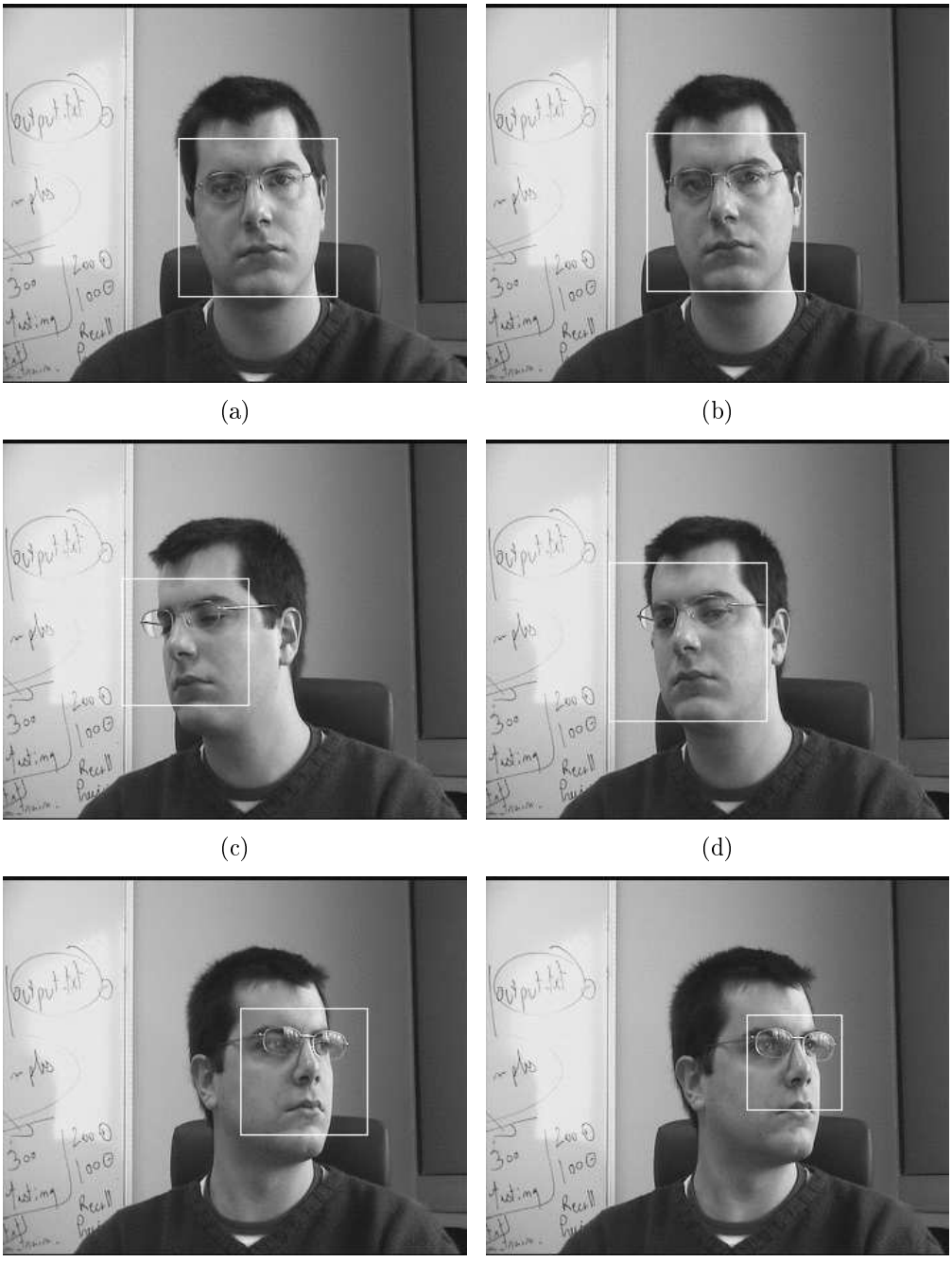
En résumé, nous dirons donc que globalement le système de détection est, comme on pouvait s'y attendre, tributaire de la pose de l'objet vis-à-vis de la caméra. Plus le visage est frontal par rapport à la caméra plus l'objet aura de chance d'être détecté. Ceci dit, une légère rotation sur les axes X et Y reste recevable dès lors qu'elle ne dépasse pas une dizaine de degré.

La figure (33) résume les résultats de détection de visage que nous avons effectués pour tester l'approche multi-résolution sur les séquences "Lionel" et "Laurent". Les images retenues sont les images 9, 141 et 191 pour la séquence "Lionel" et pour les images 195, 208 et 284 pour la séquence "Laurent". Lors de cette expérimentation, nous avons enregistré une séquence vidéo où, à partir de visages pris de face, nous reculons le zoom de la caméra pour tester la taille minimale de détection et connaître si le détecteur décroche à une échelle particulière. Les résultats sont probants pour toutes les échelles, l'approche multi-résolution fournissant un cadre robuste pour la détection de visages à différente échelle. Le détecteur ne permet pas cependant de détecter des visages de taille inférieure à celle apprise.

Fig. 28. Exemples de résultats de détection de visages par SVM sur les images clefs des vidéos fournies pour la campagne d'évaluation internationale TREC Video 2003.

(a) (b) (c) (d) (e) (f)

Fig. 29. Exemples de résultats de détection sur les images clefs des vidéos fournies pour la campagne d'évaluation internationale TREC Video 2003.



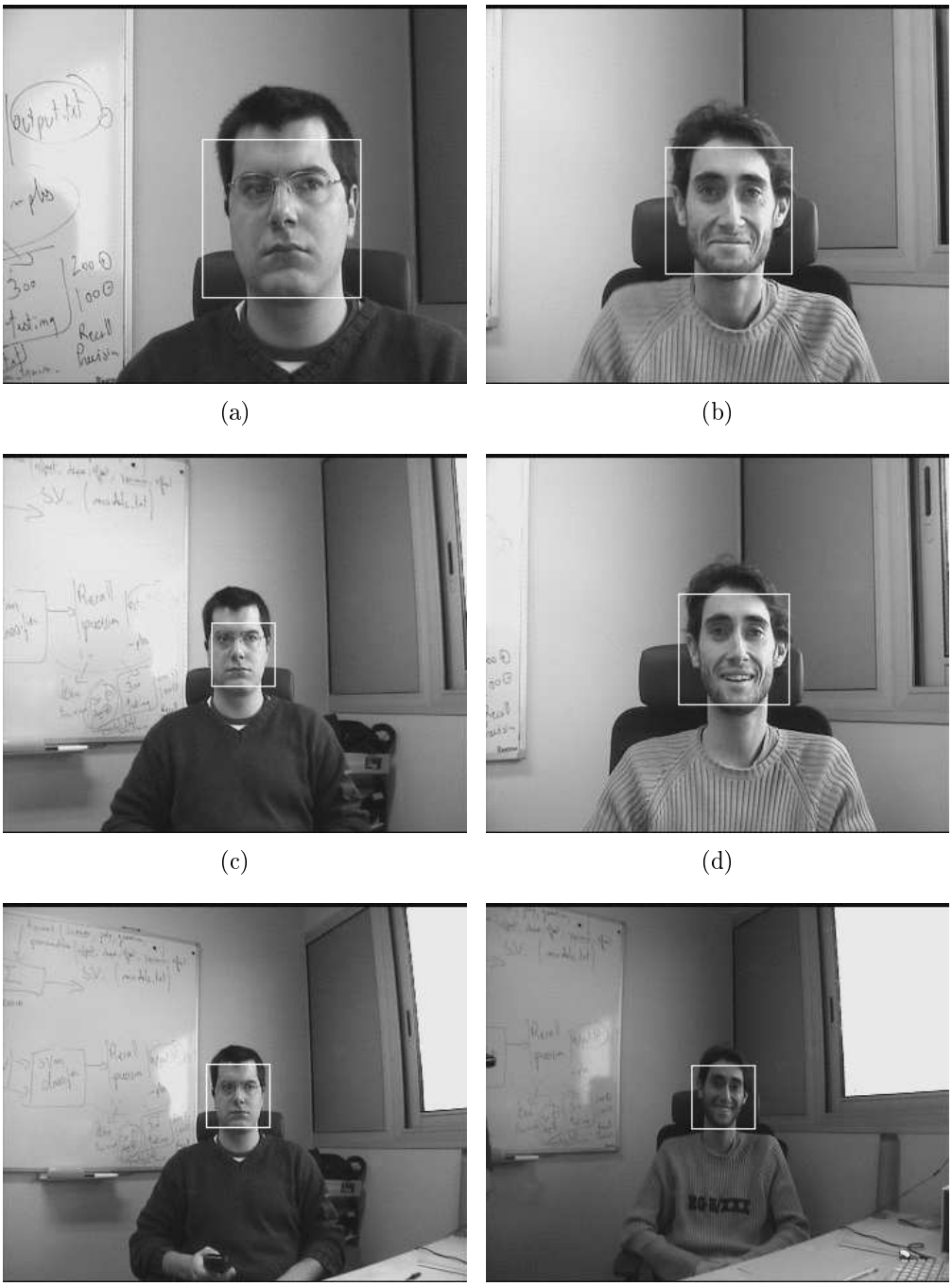
(a) (b) (c) (d) (e) (f)

Fig. 30. Exemples de détection de visages par SVM sur la séquence "Lionel". Les exemples présentés constituent, dans l'ordre, les images 1, 9, 20, 25, 53 et 56 de la séquence au format PAL 720×576 enregistrée à 25 images par seconde.

(a) (b) (c) (d) (e) (f)

Fig. 31. Exemples de détection de visages par SVM sur la séquence "Marta". Les exemples présentés constituent les images 39, 70, 81, 153, 168 et 171 de la séquence au format PAL 720×576 enregistrée à 25 images par seconde.

Fig. 32. Exemples de détection de visages par SVM sur la séquence "Laurent". Les exemples présentés constituent les images 54, 63, 76, 105, 126 et 140 de la séquence au format PAL 720×576 enregistrée à 25 images par seconde.



(a) (b) (c) (d) (e) (f)

Fig. 33. Exemples de détection de visages par SVM sur les séquences "Lionel" -colonne de gauche- et "Laurent" -colonne de droite.

3.3.8 Méthode adaptative par "bootstrapping"

Dans cette section, nous allons décrire notre contribution au problème de la détection de visages par coopération de deux méthodes : un détecteur de visages par SVM multi-résolution, tel qu'il a été décrit dans la section précédente et un détecteur de couleur de peau.

Le travail de cette partie de thèse a été réalisé en collaboration avec Anthony Don de l'équipe Visualisation au sein du LaBri. Notre contribution s'est limitée à l'implémentation et à la coopération entre deux classifieurs différents dans le but de leur amélioration commune. La partie Boostrapping décrite ici fait partie intégrante du travail d'Anthony Don. Elle sera toutefois présentée dans ce document à titre informatif afin de comprendre la coopération possible entre ces deux détecteurs ainsi que les résultats qu'Anthony Don a pu obtenir avec une telle approche.

Le travail d'Anthony Don publié dans [DCBP05] s'attache à la détection des "scènes de dialogue". Bien que de nombreux modèles et définitions des "scènes de dialogue" aient été proposés auparavant [YY96, sun01], le modèle générique retenu dans son travail ne requiert aucune hypothèse a priori sur le type de document vidéo [DCBP05]. L'approche retenue ici considère qu'une scène de dialogue est constituée d'une scène découpée en plans de montage consécutifs plus ou moins rapprochés, suivant le modèle A-B-A-... et contenant au moins un visage humain. La figure 34 illustre le schéma global retenu pour la détection des scènes visuelles de dialogue.

Dans le contexte de surveillance vidéo avec des caméras couleurs professionnelles, la collaboration active de deux détecteurs de visages basée sur deux algorithmes et sur deux espaces de représentation différents permet notamment d'augmenter les performances d'un système de surveillance global. Néanmoins, au vu de la difficulté d'obtention de données de surveillance suffisamment complètes, nous avons illustré cette approche en l'appliquant à la détection des scènes de dialogue visuelles dans les contenus artistiques [DCBP05].

La détection de visages représente donc une étape essentielle dans l'ensemble des traitements. La méthode est appelée "bootstrapping" car elle est conçue suivant la coopération active entre deux détecteurs de visages travaillant sur deux espaces de représentation différents comme présenté à la figure 34 -partie de gauche :

- un détecteur de visages, à base de Support Vector Machine, entraîné sur les valeurs d'intensité des visages humains pris de face comme décrit dans la section précédente,
- et d'un détecteur de visages dont le but est d'apprendre, à partir des résultats donnés par le détecteur par SVM, la couleur peau du visage grâce à une modélisation à base de mélanges de lois Gaussiennes

La coopération entre les deux détecteurs est nécessaire et se justifie pour deux raisons : la première concerne la richesse de l'ensemble des poses possibles dans la vidéo naturelle -série TV, surveillance vidéo, ...- d'un visage humain. Un processus entraîné, comme c'est le cas pour le détecteur SVM, sur une pose particulière, pourrait échouer dans les cas

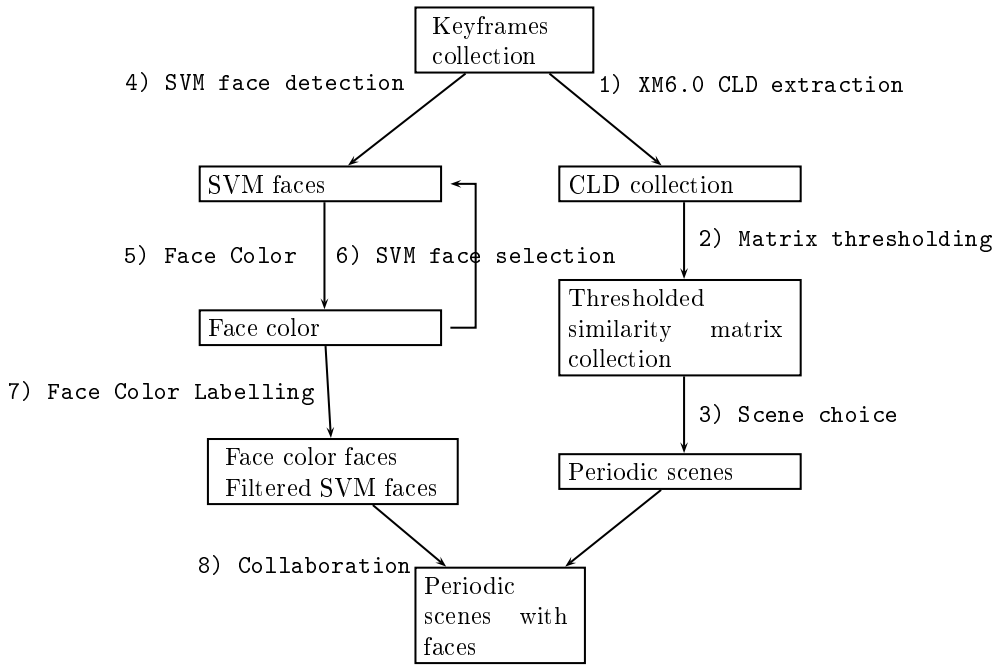


Fig. 34. Schéma de détection des scènes visuelles de dialogue [DCBP05].

de pose différente. La deuxième raison vient du fait de l'extrême diversité des zones de couleurs peau : les parties du corps humains, les zones parasites,... pourraient également être considérées comme des visages si l'on ne retenait que cet espace. L'intérêt de cette méthode réside également dans le fait que le modèle n'est plus tributaire des conditions de prise de vue (calibrage couleur, éclairage, etc).

La méthode développée doit réaliser deux objectifs : raffiner l'ensemble initial de visages détectés par SVM et détecter éventuellement d'autres visages grâce au détecteur basé couleur que nous dénommerons dorénavant détecteur FC -pour Face Color.

Modéliser la couleur peau d'un visage humain requiert de choisir auparavant un espace de représentation approprié, et d'identifier les groupes -clusters- de couleurs associés à la couleur du visage dans cet espace comme nous l'avons vu dans la partie 3.3.

Le modèle de couleur de peau visage retenu est exprimé dans l'espace $YCbCr$, du fait de son uniformité perceptuelle et de l'indépendance de la luminance et de la chrominance. Dans cet espace la distribution de la couleur peau du visage est modélisée par un modèle à mélange de lois Gaussiennes (GMM). Nous reprenons ainsi la modélisation retenue au chapitre 2 pour la détection de mouvement appliquée cette fois ci dans un espace couleur.

Chaque composante du mélange est constituée d'une loi normale avec comme paramètre, un vecteur moyenne $\mu_k = (\mu_k^Y, \mu_k^{Cb}, \mu_k^{Cr})$, une matrice de covariance Σ_k. Un poids est associé à chaque composante du mélange. Soit $w_1, .., w_K$ les poids associés tels que

$\sum_{k=1}^{K} w_k = 1$ et $\forall k \in [1, K], w_k > 0$. Nous avons recours à la modélisation par mélange afin de prendre en compte la diversité des tons de couleurs possibles pour un visage qui se reflète dans la diversité des clusters de couleurs dans notre espace de couleur. Ce point s'avère crucial si l'on travaille sur différentes vidéo avec des effets d'éclairage complexes.

La méthode itérative de "bootstrapping" que nous proposons est décrite ci-dessous :

Algorithme 1 Algorithme de détection de visages par "bootstrapping"

$l \leftarrow 0$

Détection des visages par SVM multi-résolution : L'ensemble $F_{SVM}^{(l)} = \{f_{SVM_i}^{(l)}\}$ représentent toutes les images $f_{SVM_i}^{(l)}$ classifiées comme représentant des visages à l'instant l.

répéter

Phase 1 : Apprentissage de la couleur de peau

Considérer tous les pixels $P_{i,l} \in f_{SVM_i}^{(l)}$ et estimer le modèle de mélange par l'algorithme ISODATA -cf. Section 2.3.1-. L'ensemble des paramètres (μ_i, σ_i^2, w_i) caractérise les Gaussiennes dans le mélange. En supposant que les pixels de visages représentent la majorité des imagettes $f_{SVM_i}^{(l)}$ supprimer les Gaussiennes de poids faibles.

Phase 2 : Étiquetage des pixels "couleur peau" dans les images $f_{SVM_i}^{(l)}$

Phase 3 : Étape de "bootstraping"

Suppression des faux positifs dans l'ensemble $F_{SVM}^{(l)}$. Les imagettes $f_{SVM_i}^{(l)}$ qui contiennent un faible pourcentage de pixels "couleur de peau" sont supprimées de $F_{SVM}^{(l)}$. L'ensemble respecte donc $F_{SVM}^{(l+1)} \subset F_{SVM}^{(l)}$.

$l \leftarrow l + 1$

jusqu'à ce qu'aucun visage $f_{SVM_i}^{(l)}$ ne soit supprimé de l'ensemble $F_{SVM}^{(l)}$

Cette méthode opère sur l'ensemble des images à traiter et donc suppose la disponibilité du contenu vidéo "off-line". Ainsi la phase 1 de l'algorithme présenté ci dessous permet de détecter les visages par SVM et de "filtrer" ou tout du moins d'améliorer la précision par la détection basée sur l'apprentissage de la couleur de peau. Cette phase d'apprentissage est basée sur des regroupements des couleurs de peau et sur l'estimation de l'ensemble des paramètres de chacune des lois du mélange. La segmentation de la couleur du visage s'opère en recentrant les détections données par le détecteur SVM en fixant plus particulièrement le nez, la bouche, les yeux, ... En supposant dès lors que le visage occupe la majeure partie de cette boîte englobante, nous considérons l'ensemble des valeurs $YCbCr$ des pixels de cette boîte comme base d'apprentissage. Les regroupements en couleur peau sont ensuite effectués par l'algorithme ISODATA [RG99] comme décrit dans le chapitre précédent.

ISODATA est utilisé dans ce cas de figure pour déterminer le nombre de modèles et les poids associés de façon automatique. Chaque groupe -cluster- de couleur du visage est alors modélisé par une Gaussienne suivant le modèle de mélange.

En supposant que la matrice de covariance soit diagonale, un pixel $X = (x_Y, x_{Cb}, x_{Cr})$ est considéré comme appartenant à la classe pixel de visage s'il existe une valeur k telle

que pour chaque composante de X on ait :

$$\begin{array}{rcl} \mu_k^Y - \alpha\sigma_k^Y \leq & x_Y & \leq \mu_k^Y + \alpha\sigma_k^Y \\ \mu_k^{Cb} - \alpha\sigma_k^{Cb} \leq & x_{Cb} & \leq \mu_k^{Cb} + \alpha\sigma_k^{Cb} \\ \mu_k^{Cr} - \alpha\sigma_k^{Cr} \leq & x_{Cr} & \leq \mu_k^{Cr} + \alpha\sigma_k^{Cr} \end{array}$$

Empiriquement, et selon nos expériences sur les bases TRECVideo 2003 et 2004, une valeur de α égale à $1,5$ donne des résultats tout à fait satisfaisants.

L'étape de "bootstrapping" de la méthode consiste ensuite à supprimer les faux positifs de l'ensemble de visages détectés par SVM. Partant de l'ensemble initial des visages détectés, F_{SVM}, un modèle de couleur de peau est estimé.

Les taux de précision de l'algorithme SVM impliquent qu'il y ait plus de "vrais" visages que de "faux" visages dans l'ensemble initial F_{SVM}. De plus la majeure partie des fausses détections est due aux effets de texture dont la taille observée est plus petite que celle des visages humains. Suite à l'algorithme ISODATA, les clusters ayant un poids faible représenteront plus probablement la couleur des faux-positifs détectés.

Cette étape de "bootstrapping" non seulement améliore le taux de précision du détecteur de visages par SVM, mais permet également d'apprendre de façon adaptative la couleur de peau des visages à travers le document vidéo. Le modèle de couleur du visage est ainsi mieux adapté aux conditions d'éclairage du document vidéo à analyser que s'il avait été appris simplement sur une base d'apprentissage de visage. Comme le détecteur de visages SVM détecte uniquement les visages pris de face, le but du détecteur basé sur la couleur consiste ensuite à construire un ensemble de visages, F_{FC}, complétant F_{SVM} avec les visages non frontaux.

La détection des visages non-frontaux s'effectue ensuite suivant le principe suivant - phase 2 : dans les images clefs où le détecteur SVM n'a pas localisé de visage humain, les pixels appartenant au modelé de couleur peau, c'est à dire respectant les équations (119), sont étiquetés comme étant des pixels de visages afin de déterminer une première carte de segmentation. Avant l'étape finale de détection, la carte binaire est traitée de la façon suivante :

- un filtrage médian est tout d'abord appliqué afin de supprimer les pixels isolés,
- suivi d'une opération morphologique binaire : la dilatation. Cette opération est réalisée en choisissant un élément structurant carré de 5×5 dans le but de reconnecter les régions de visage sur-segmentées.
- et enfin pour les régions 4 connexes, une boîte englobante est calculée.

La phase 2 de l'algorithme permet donc de compléter les résultats de la détection de la phase 1 pour des prises de vue non-frontales. Néanmoins, étant basée sur la "couleur de peau" elle engendre des fausses détections -autres parties du corps notamment- qu'il

convient de supprimer. Lors de la dernière étape de détection des visages, la phase 3, un certain nombre de règles ont donc été définies afin d'éliminer les boîtes englobantes qui ne contiendraient pas a priori de visages humains.

- La position des centres de visage contenus dans F_{SVM} est modélisée par deux Gaussiennes : une pour la coordonnée X et une pour Y du centre : $\eta_X(\mu_{X_{CENTRE}}, \sigma_{X_{CENTRE}})$ et $\eta_Y(\mu_{Y_{CENTRE}}, \sigma_{Y_{CENTRE}})$. Les images f_{FC_i} des boîtes englobantes dont le centre n'appartient pas à l'intervalle de confiance sont éliminées.
- Les imagettes dont la taille des boîtes englobantes est plus petite que le plus petit visage de F_{SVM} sont également éliminées.
- Les imagettes dont la taille de la boîte englobante est plus grande que le plus grand visage de F_{SVM} sont également éliminées. Cette règle a pour but de supprimer les larges zones de pixels de fond étiquetés comme des pixels de visage.
- Les boîtes englobantes se chevauchant sont fusionnées.

Ainsi l'ensemble F'_{FC} des détections filtrées est obtenu. Ensuite, les deux ensembles filtrés $F^{(l)}_{SVM} + F'_{FC}$ forment l'ensemble des régions contenant des visages. Il est important de noter que toutes les règles citées ci-dessus dépendent des paramètres obtenues par les visages appartenant à F_{SVM} -taille de fenêtre et position par exemple-. Ceci fournit donc un cadre adaptatif de rejet des visages suivant la nature du document vidéo à analyser.

Les résultats des deux algorithmes de détection de visage sont donnés dans la section suivante en terme de précision et de rappel.

Résultats

La méthode de détection de visage par bootstrapping proposée dans [DCBP05] et [QMS+04] a été validée sur la vidéo "Quels temps font-ils ?" fournie par SFRS et sur les vidéos de la campagne TRECVideo 2003. Dans le cadre de [DCBP05], il s'agit de détecter les scènes périodiques contenant au moins un visage dans le but de détecter des scènes de dialogue. Dans [QMS+04] la méthode par "bootstrapping" permet, suite à un processus de détection de visages pas assez précis, de zoomer sur le visage en ne retenant que les yeux le nez et la bouche.

Afin de vérifier les modifications de performance de l'approche proposée dans [DCBP05], nous avons étiqueté à la main les images clefs du document vidéo intitulé "Quel temps font ils ?". Des visages pris de face et de profil ont été ainsi étiquetés pour aboutir à un nombre total de 456 visages pour cette vidéo. Chaque image clef est traitée par le classifieur SVM avec un noyau polynomial de second ordre. Ce classifieur est entraîne sur 6977 visages pris de face et 23478 images de non visages de la base TRECVideo 2003.

Le modèle de mélange contient pour sa part 9 Gaussiennes dans le domaine de couleur $YCbCr$, nombre automatiquement déterminé par l'algorithme ISODATA. Le processus de détection de visages par mélange permet d'obtenir 413 régions de l'image dites de visages,

ce qui représente une précision de 79% (328 régions couvrent réellement un visage pour 85 régions couvrant un non-visage) et un taux de rappel égal à 72%.

Les exemples présentés aux figures 35 et 36 illustrent les résultats de détection de visages issus de [DCBP05] suivant les deux méthodes de détection de visages retenues, l'une basée couleur et l'autre basée apparence. La figure 35 donne quelques exemples de résultats obtenus à la phase 1 par le classifieur SVM avec l'approche multi-résolution comme présentée dans la section précédente. Les résultats obtenus sont, comme on peut le constater, fortement tributaires de la position du visage vis à vis de la caméra : ce résultat n'est certes pas nouveau en soit mais il est toutefois intéressant de constater que de nombreux visages sont détectés suivant une légère rotation sur un axe. Des visages de profil ou de 3/4 ne sont pas détectés. A titre d'exemple les résultats présentés aux images 35(a), 35(c), 35(e) et 35(g) illustrent des résultats sur des visages qui répondraient a priori plus favorablement aux résultats de détection de visage par SVM. Les visages sont positionnés de face vis à vis de la caméra et ressemblent donc plus à ceux de la base d'apprentissage en terme de positionnement. Les images 35(b), 35(d) et 35(f) présentent quant à elles des résultats de détection alors que le visage est légèrement orienté soit sur l'axe X ou Y.

L'intérêt de l'approche retenue par "bootstrapping" réside finalement dans la possibilité qu'un visage, même non détecté par le premier classifieur SVM puisse l'être par un autre classifieur basé sur la couleur celui-là. Ceci permet entre autres de pouvoir détecter des visages non détectés par le premier processus du fait de la pose de l'objet notamment. Le descripteur couleur est défini lors de la première passe suite aux résultats de détection de visage par SVM. Sur la figure 36 apparaissent les diverses étapes de détection. La colonne de gauche montre les résultats de classification pour chaque pixel. Les pixels en noir correspondent aux pixels n'appartenant pas à la classe "couleur de peau", les autres ceux appartenant à cette classe. La colonne du milieu présente les regroupements entre pixels que nous avons opérés. Sont ainsi éliminées les fausses détections et les pixels isolés. La colonne de droite donne finalement les résultats finaux de détection de visage à proprement parler. Comme on peut le constater, de nombreux visages sont ajoutés et la détection est moins tributaire de la pose du visage.

Fig. 35. Exemples de résultats de détection de visage basée sur l'apparence par Support Vector Machine extraits de la vidéo "Quel Temps Font Ils ?" -SFRS-

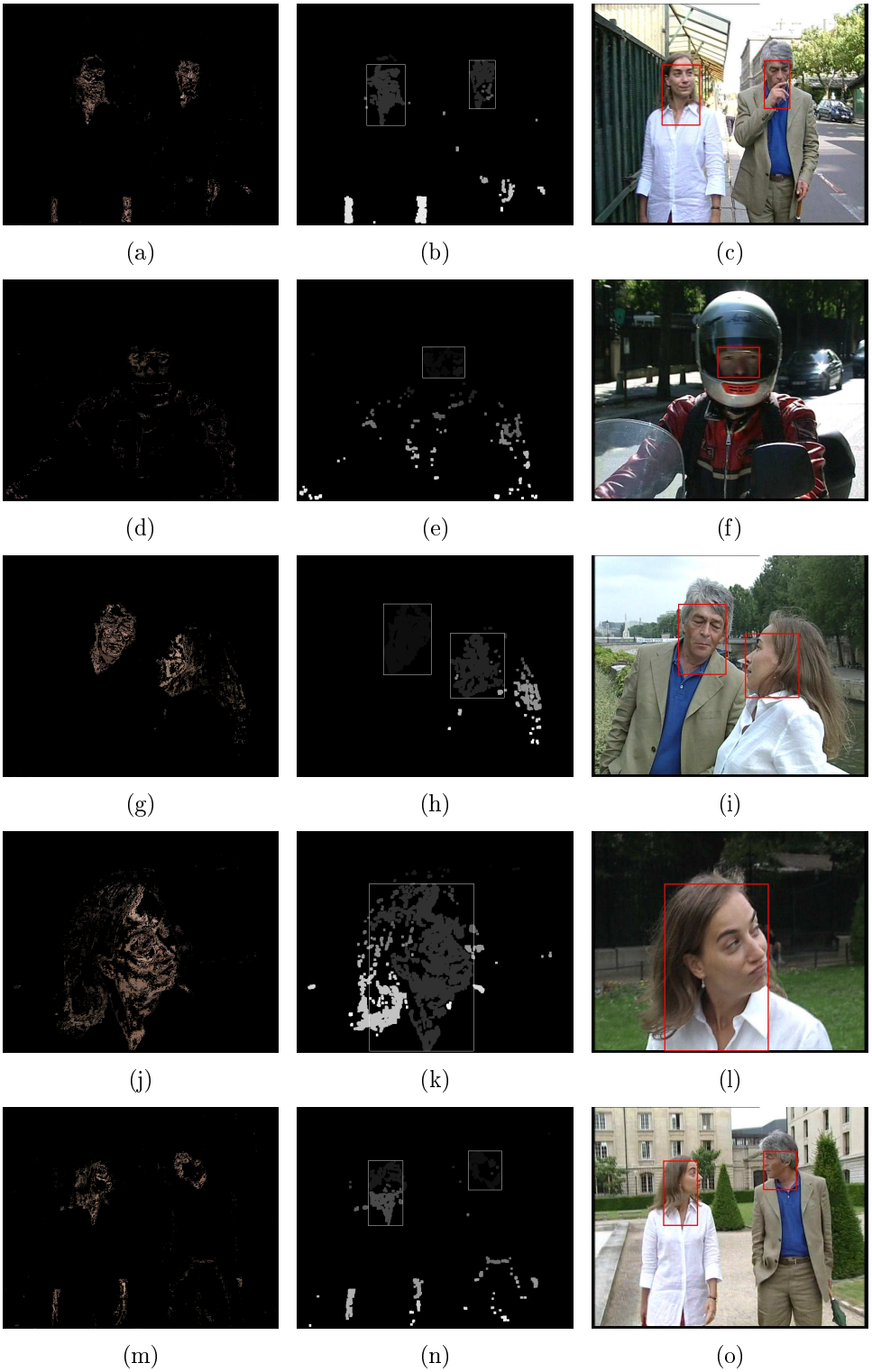


Fig. 36. Résultats de détection de visage basée couleur peau extraits de la vidéo "Quel Temps font Ils?"

3.4 Conclusion

Suite à l'analyse de l'état de l'art des outils mathématiques et des espaces de représentation possibles pour résoudre le problème de détection de visages, nous avons décrit plusieurs méthodes de détection de visages par Machines à Vecteurs de Support, SVM.

En partant du fait que le visage humain et plus particulièrement les valeurs d'intensité d'un visage humain sont suffisamment discriminantes pour discerner les visages des autres objets présents dans une rétine, nous avons poursuivi les travaux d'Osuna, travaux pionners pour la détection de visages par SVM pour des images de taille fixe. Cette première contribution est purement expérimentale. Nous avons, en effet, expérimenté les travaux d'Osuna et avons utilisés puis optimisés les outils SVM développés par le MIT. Néanmoins au vue des limitations d'une telle méthode nous l'avons amélioré et avons proposé une méthode par approche multi-résolution (multi échelle combinée avec un classifieur SVM). Cette nouvelle méthode a démontré de bonnes performances sur des vidéos naturelles de type Broadcast (TREC Video 2003) et de surveillance (Argos Technovision).

Le problème majeur des méthodes d'apprentissage et notamment des SVM vient du fait que le classifieur soit tributaire de la pose de la forme apprise. Ayant appris l'apparence globale des valeurs d'intensité d'un visage humain pris de face, un classifieur perdra tout son pouvoir d'induction sur des images différentes de la pose apprise. Pour répondre à ce type de problème, nous avons mis en place et valider une méthode intelligente de détection par coopération de deux classifieurs : un premier classifieur par SVM détecte les visages humains alors qu'un second classifieur par mélange de lois Gaussiennes apprend la couleur de peau des visages détectés à partir des vraies détections et élimine les autres. La méthode proposée par Anthony Don dite par bootstrapping permet ensuite par itérations successives d'approximer au mieux la couleur et permet de ne plus être tributaire de la pose du visage. Cette collaboration aboutit à une amélioration sensible du taux de rappel et de la précision de la détection des visages.

En perspective de ce travail, nous avons vu que les vecteurs de support comportent des propriétés géométriques intéressantes vis-à-vis de leur proximité par rapport à l'hyperplan dans le "feature space". Ceci dit ces vecteurs de support sont avant tout des éléments de la base d'apprentissage, par conséquent sont avant tout des images. Lorsqu'on entraine un classifieur SVM sur une base de visages/non visages, nous obtenons un ensemble de vecteurs de support de la même taille que les images apprises. si l'on entraîne un autre classifieur SVM sur la même base mais dont la taille serait réduite par la même méthode de filtrage, les vecteurs de support retenus seront exactement les mêmes mais à une résolution inférieure. Par conséquent plutôt que d'échantillonner l'image initiale pour construire la pyramide, il serait intéressant, dans la perspective de ce travail, de comparer cette méthode avec un échantillonage direct de la rétine et des vecteurs de support retenus pendant l'apprentissage. Le schéma de classification s'opérerait alors comme suit : l'image originale est scannée par une rétine, cette fois-ci, de taille variable. Chaque valeur de luminance est ensuite injecté dans un classifieur SVM qui, en fonction de la taille de la rétine utilisée va calculer le

produit scalaire sur la base des SV retenus pour cette même taille. Enfin d'autres espaces de représentation de visages sont envisageables.

Chapitre 4

Suivi d'objets par Support Vector Machines

Quand un objet se déplace vis à vis d'un observateur, les images projetées de l'objet sur la rétine ou sur une caméra changent. Dans la communauté de la vision par ordinateur, suivre un objet signifie maintenir une correspondance de la structure de l'objet à travers les images. Cette structure correspond à une structure 3D, projetée sur un plan 2D ou non suivant le capteur, qu'il est nécessaire de reconnaître avant d'estimer son mouvement et finalement sa position. Le suivi de formes a été étudié de façon pléthorique dans la littérature de la vision par ordinateur à la fois du fait de son intérêt intrinsèque et du nombre sans cesse croissant d'applications nécessitant de suivre une région ou une forme à l'image.

A titre d'exemple, les robots autonomes ont besoin d'appréhender l'environnement qui les entourent et donc de suivre les objets en mouvement présents dans cet environnement [RM96, PUE96]. Un autre cas particulièrement étudié est celui de l'assistance à la conduite dans les voitures, qui doit, entre autres permettre de suivre les marquages sur la route [Dic97, Cri93] ou de suivre les autres véhicules [Smi98]. Ce suivi peut également permettre de glaner des informations sur le trafic des scènes filmées d'une autoroute [FRB94, KWH+94] ou d'un aéroport [Sul94] par exemple. En terme d'applications toujours, citons les applications de suivi pour les interfaces Homme-Machine [Roy97], le suivi de lèvres pour l'aide à la reconnaissance de parole [BO95, SMY97], où encore le suivi de mains ou de doigts pour la problématique des tableaux virtuels dans le contexte de la réalité augmentée [Wel93, BI94]. Dans ce dernier cas les gestes de la main sont alors à développer dans le cadre de la perception des actions c'est à dire qu'on déduit une connaissance de la scène à partir des informations de suivi. Cette hypothèse est également à la base d'applications de suivi d'individus dans le contexte de surveillance vidéo notamment [Hog83, BH94, FCH96].

La capacité de suivre un objet en déplacement est d'une importance capitale pour de nombreuses applications et par conséquent il a été proposé de nombreuses solutions et

entre autres pour la problématique de la surveillance vidéo [GSRL98, KCL+98]. Chacune de ces solutions propose des scénarios de complexité variable suivant le thème abordé. A titre d'exemple il suffit de consulter la variété des scénarios proposés par le projet CAVIAR -Context Aware Vision using Image-based Active Recognition- [uIbAR01] pour se rendre compte de leur complexité : personne marchant, courant, rencontre entre deux individus, combat entre deux personnes, personne à terre sont autant de cas à traiter dans ce projet.

Chercher une correspondance de l'objet détecté et le suivre au cours du temps nécessite de prédire la position de l'objet en supposant un certain nombre de contraintes sur son mouvement. Malheureusement la position de l'objet n'est pas le seul paramètre qui varie au cours du temps, et l'objet lui même peut changer de forme, de luminosité,... et ce pour de multiples raisons. A cela si l'on ajoute la possibilité de gérer les occultations passives -objet passant derrière un objet immobile- ou des occultations actives -objet en mouvement occultant un autre objet en mouvement-, alors le suivi d'un objet serait impossible sans supposer une connaissance a riori soit sur l'objet soit sur l'environnement de cet objet.

De la pertinence et de la précision de ces suppositions dépend naturellement la robustesse du suivi pendant les changements d'apparence de l'objet. Suivant [Toy96], la "robustesse" d'un système de suivi peut se diviser en deux classes : robustesse pré-échec (pre-failure robustness) et robustesse post-échec (post-failure robustness). La première classe a pour but de détecter préventivement que le processus de suivi va décrocher en supposant par exemple différents aspects de changements enclins à l'échec du suivi. La deuxième classe, post-échec, se réfère à la capacité de détecter automatiquement et éventuellement de récupérer l'objet suite à un échec de suivi.

Le suivi d'objet temps réel ou pseudo temps réel est une tâche encore plus difficile si l'on remarque que généralement elle est en fin d'une longue chaîne de processus de traitement comme la détection de mouvement ou d'objets par exemple. Dans l'optique de construire un système global de surveillance vidéo automatisée et temps-réel par ordinateur, le processus de suivi d'objet est alors le dernier processus et doit se contenter des ressources laissées par les autres processus Il convient alors de trouver une solution rapide, via un modèle simple -voire parfois simpliste-, pour déterminer la position d'un objet à un instant de temps $t + dt$.

4.1 État de l'art

Dans de nombreux cas, le suivi d'un objet est accompli en suivant simultanément différentes parties de ce même objet où chacune de ces parties correspond à une région de l'image. Il est normal alors que l'hypothèse la plus commune soit de supposer que les changements des régions entre deux images sont faibles. Cette supposition est fondamentale pour les méthodes basées *corrélation*, appelées également méthodes de "bas niveau" [IB98]. Il s'agit probablement de la méthode de suivi la plus ancienne par correspondance et par analyse des similarités des régions de l'image [SP72]. En supposant une fenêtre de l'image

d'une taille donnée et à un instant t, la fenêtre correspondante à l'instant $t+1$ est celle qui obtient le meilleur score de corrélation avec la précédente fenêtre.

D'autres méthodes de suivi s'appuient plutôt sur la notion de *flot-optique* ou de champ de déplacement dans le domaine image, qui peut être interprété comme étant le résultat simultané du suivi de tous les points de l'image. Dans le cadre du suivi d'objets, le mouvement cohérent d'une région ou d'un objet obtenu à partir de l'estimation du flot optique peut être ensuite réutilisé pour guider le processus de suivi comme il a été démontré dans [MB94, TLS93]. La robustesse du suivi basé flot optique dépend de la façon dont le champ de déplacement est calculé. Pour cela il existe une large variété d'algorithmes d'estimation du flot optique. Un des problèmes majeurs de ces méthodes est qu'elles sont fortement tributaires des changements d'illumination de la scène filmée. Néanmoins ces effets peuvent être atténués avec l'utilisation de techniques de pré-filtrage [BAHH92] par exemple, ou encore de correspondance basée sur des informations mutuelles [Vio95] mais finalement peu de travaux se penchent réellement sur l'estimation du flot optique avec variations de luminosité.

Afin de rendre le processus de suivi plus robuste non seulement aux variations de luminosité mais également aux conditions de prise de vue, une autre solution consiste cette fois ci à extraire des éléments caractéristiques -features- de l'image. On parle de "feature tracking". Elle consiste à extraire un ensemble de points caractéristiques (arêtes, coins, contours, blobs, couleur, ...) suivant une méthode appropriée puis de chercher à suivre non plus l'objet mais ses points caractéristiques grâce à une procédure de corrélation, de filtrage, ... En supposant que l'objet à suivre est rigide, c'est à dire que les distances entre tous les points sont fixes dans l'espace 3D, un certain nombre de contraintes géométriques peuvent être émises entre ces différents points. Ceux qui ne respectent pas ces contraintes -outliers- sont alors rejetés suivant un modèle statistique robuste [BI94, IB98]. L'intérêt d'avoir un modèle rigide réside dans le fait qu'il soit capable, par la suite, de reconstruire une structure 3D de l'objet.

S'il est possible d'avoir une connaissance sur le mouvement de l'objet, cette information s'avère utile pour ensuite prédire la position de l'objet dans la suite des images. Cette connaissance permet par exemple de réduire la zone de recherche de l'objet sur l'image suivante par exemple ou de points caractéristiques comme nous le mentionnions auparavant. Ces "prédicteurs" du mouvement sont soit basés sur des filtres (Kalman [Kal60], Wiener, ...) en supposant que le mouvement est d'amplitude faible [DF90], soit en supposant une vitesse constante ou une accélération constante de l'objet. Des modèles plus complexes permettent toutefois d'améliorer le suivi en augmentant la robustesse vis à vis des changements d'apparence d'un objet. Dans [Zha94] par exemple, un modèle rigide de l'objet est supposé et injecté dans un filtre de Kalman étendu [AMGC02] dans le but de suivre des segments de ligne. Des modèles probabilistes ont ensuite été proposés pour résoudre le problème même dans des conditions difficiles. Dans [IB98] un modèle probabiliste du mouvement est supposé en avance et réutilisé dans le schéma de suivi de contours dans les scènes fortement texturées. L'algorithme développé dans [IB98] calcule la densité de probabilité conditionnelle de différents mouvements à un instant donné. Cet algorithme

appelé "Condensation" pour "Conditional Density Propagration" semble donner des résultats performants. D'autres espaces de représentation ont été ensuite proposés suivant la même technique avec principalement les espaces couleur [NKMG02], ...

Malheureusement supposer que le modèle soit rigide pour des applications telles que la surveillance vidéo est trop contraignant et il devient alors nécessaire de considérer un cadre plus flexible : soit premièrement de fusionner plusieurs résultats de processus de suivi différents [UNME95, IB98], soit de considérer carrément des modèles déformables [CRM00, CR00, DRM03]. Dans le premier cas, le processus de suivi est le résultat de plusieurs modules de suivi travaillant en parallèle sur des espaces de données différents et donc des hypothèses de travail différentes. La robustesse "post-échec" est ainsi améliorée dans le sens ou si l'un des modules de suivi décroche, il peut être détecté suivant les résultats des autres modules. Dans [IB98] sont combinés suivi de couleur et suivi de contours actifs, alors que dans [UNME95], les auteurs fusionnent les résultats d'un "trackeur" basé flot optique et la profondeur obtenue par stéréo pour suivre des objets rigides et semi-rigides. Toutefois ces méthodes sont coûteuses en terme d'occupation processeur et pour une application temps-réel il est souvent préférable de considérer des modèles déformables : ainsi dans [CRM00, CR00, DRM03], les auteurs proposent une méthode de suivi temps réel de régions d'intérêt définies suivant leur couleur et leur texture par la technique dit "Mean-shift". L'algorithme Mean-Shift est une procédure itérative de recherche de maximum local dans un espace R^d, proposée par Fukunaga [Fuk90] en 1975 puis reprise par Comaniciu en 2000 pour le suivi d'objets déformables [CRM00, CR00, DRM03]. Le but consiste à rechercher la position de l'objet à suivre dans les différentes images, à partir du coefficient de Bhattacharyya [Bha43, DSG90], coefficient dérivé de l'erreur bayésienne, pour mesurer la similarité entre les distributions de couleur d'un modèle de l'objet et de certaines régions de l'image.

Établir un état de l'art des méthodes et algorithmes de suivi est un exercice difficile tant il existe pléthore de méthodes. Nous allons alors dresser un panorama des méthodes majeures que nous avons regroupées en trois parties : méthodes basées images, basées flot optique et enfin celles par contours actifs.

4.1.1 Suivi basé image dite "bas niveau"

Nous avons regroupé sous le terme "bas niveau" [IB98] les méthodes de suivi qui supposent peu d'hypothèses sur la nature de l'objet à suivre. Des caractéristiques génériques sont extraites de l'image et sont ensuite regroupées et interprétées suivant une connaissance de plus haut niveau de la scène. Ces méthodes différent des méthodes basées flot optique qui posent de fortes contraintes de modélisation mais nous y reviendrons dans la prochaine partie.

De nombreux modèles, et par voie de conséquence de nombreux systèmes [SSIB97, WADP97, Bre97a, MDBP97, IL98, Roy97, YC96, GC96, MS95] sont apparus et se conforment à cette notion de critères dits de "bas niveaux". Dans les travaux de Intille

[SSIB97] par exemple, les auteurs construisent un algorithme de suivi de blob -zone homogène en terme d'intensité ou de couleur- pour le suivi temps-réel de personnes vues de face dans une pièce. Ainsi, après avoir détecté les zones de mouvement suivant une méthode de soustraction de fond, ces régions en mouvement sont ensuite segmentées en blob de couleur ou d'intensité homogène -via un opérateur Laplacien par exemple [Mar90, BA89]. Ces blobs sont ensuite regroupés suivant leur proximité et leur vitesse en supposant qu'ils appartiennent à une seule et même personne suivant ces critères. Cette technique a l'intérêt d'être extrêmement rapide mais devient problématique lorsque les personnes filmées sont proches les unes des autres.

Dans [WADP97], les auteurs établissent également un algorithme de suivi basé sur les blobs couleurs à partir d'une caméra filmant une seule personne. Les blobs sont également regroupés suivant la topologie de l'individu et on recherchera par exemple à regrouper des blobs aux positions des bras, des jambes, du tronc, etc. Mis à jour de façon dynamique, les blobs obtenus sont moins sujets à perturbation pendant les phases d'occultation, mais le modèle reste toujours tributaire de la pose de l'individu...

Les auteurs de [Bre97a] segmentent les pixels de l'image en un ensemble de blobs, chacun étant représenté par une probabilité de distribution Gaussienne sur la base de la vitesse basée flot optique et de l'information couleur. L'algorithme E.M. est utilisé dans le cadre d'un filtrage hiérarchique et un filtre de Kalman est ensuite utilisé pour suivre chaque blob et pour initialiser l'algorithme EM pour l'étape suivante. Finalement les auteurs récupèrent un ensemble de paramètres des blobs grâce à un modèle continu-discret en combinant des processus auto-régressifs et des modèles de Markov cachés [Bre97a]. D'autres méthodes de suivi qui n'utilisent aucun modèle explicite s'appuient par contre sur le suivi des distances de Hausdorff [HKR93, JKF01], ou le suivi de points caractéristiques suivant un modèle de contrainte semi-rigide ou rigide sur cet ensemble de points d'un même objet [TM94, CK98, Tor97].

4.1.2 Suivi basé flot optique

Les méthodes de flot optique appliquées au contexte de suivi sont des méthodes depuis longtemps éprouvées [KvD75, HS81, BA93, SXJ96]. Ces méthodes ont été retenues à la fois pour l'estimation des champs denses de mouvement sur les régions visibles d'une séquence d'images [BA93, SXJ96] et pour la segmentation des zones de flot optique consistants au sens du mouvement en objet [MBPL04, MLBP05, BJ98]. Dans le but de résoudre l'équation de contrainte du flot optique, il est nécessaire soit d'appliquer une régularisation en supposant que le mouvement des régions de l'image est faible [Gel98], soit de paramétrer le mouvement d'une région en utilisant un modèle de faible dimensionalité, par exemple un modèle affine [KvD91, MZ92].

De nombreuses méthodes ont été proposées dans cette optique [LWB96, MBPB02, BJ98, SXJ96, Avi04, CBP05c]. Dans [MBPB02], les auteurs proposent une méthode de suivi d'objets présents dans un contenu vidéo. Chaque objet est caractérisé par un ensemble

de régions polygonales. Une première étape de segmentation spatiale puis d'estimation de mouvement est tout d'abord appliquée, qui, couplée avec une intervention humaine, permet de construire une carte sémantique sur la première image de la séquence vidéo. Le suivi de ce modèle tout au long de la vidéo est basé sur la détection et l'indexation des nouveaux objets de la scène. Des règles sémantiques sont ensuite utilisées pour l'étiquetage des objets.

L'estimation du modèle affine peut également faire intervenir des méthodes plus complexes comme des algorithmes d'apprentissage [Avi04, CBP05c] par exemple. Dans ces mêmes études, la connaissance apprise des objets d'intérêt, véhicules [Avi04] ou visages humains [CBP05c, CBPJ06, CBPJ05], est réutilisée pour estimer le modèle de mouvement de l'objet appris. Ces méthodes ont le mérite d'offrir un cadre élégant d'un point de vue mathématique à la fois pour la détection et pour le suivi.

4.2 Outils mathématiques

Après avoir présenté les espaces de représentation possibles pour le suivi, nous allons présenter, tout au long de cette section, quelques-une des grandes familles d'outils mathématiques très fréquemment utilisés dans la problématique de suivi de formes à l'image.

4.2.1 Filtrage de Kalman

Le problème de suivi est typiquement celui d'un système évoluant dans le temps tout en étant perturbé par le bruit qu'il génère. Ce type de problème peut être résolu par des estimateurs particuliers : les filtres. Avant de commencer, revenons sur certains termes. Un vecteur d'état contient toute l'information pertinente nécessaire à la bonne description d'un système à étudier. A titre d'exemple, dans notre problème de suivi cette information contient la description cinématique d'un objet. Le vecteur de mesure contient pour sa part toute l'information observée et éventuellement bruitée du système. Il est généralement de dimension plus réduite que le vecteur d'état.

Avant d'analyser et d'inférer sur un système particulier, deux modèles, au moins, sont nécessaires :

- un modèle qui décrit l'évolution du système dans le temps dit modèle du système
- un modèle d'évolution du bruit ou modèle de mesure.

On supposera également que tous deux sont accessibles en terme de probabilité. En considérant l'approche bayésienne, il faut construire la fonction de densité de probabilité a posteriori de l'état grâce aux informations disponibles. Un filtre se divise en deux étapes : la phase de prédiction et la phase de mise à jour. La prédiction se base sur le modèle du système pour prédire un état de la fonction de densité de probabilité d'un instant t vers un instant $t+1$. Comme le modèle est déformable, car perturbé par le bruit, l'état de la fonction de densité de probabilité est lui même perturbé et nécessite forcément une mise

à jour. Afin de définir le problème de suivi, il est nécessaire de considérer l'évolution au cours du temps d'un ensemble d'états :

$$x_k = f_k(x_{k-1}, v_{k-1}) \tag{119}$$

où $f_k : \mathcal{R}^{n_x} \times \mathcal{R}^{n_v} \longrightarrow \mathcal{R}^{n_x}$ est une fonction éventuellement non-linéaire de x_{k-1}, $\{v_k, k \in \mathcal{N}\}$ le bruit, n_x et n_v sont respectivement les dimensions des vecteurs d'état et de bruit du processus.

L'objectif du suivi est d'estimer récursivement x_k à partir des mesures

$$z_k = h_k(x_k, \eta_k) \tag{120}$$

où $h_k : \mathcal{R}^{n_x} \times \mathcal{R}^{n_n} \rightarrow \mathcal{R}^{n_z}$est une fonction éventuellement non-linéaire, $\{\eta_k, k \in \mathcal{N}\}$ l'ensemble de bruit mesuré, n_z et n_n les dimensions respectivement des mesures et du vecteur de bruit mesuré.

Le problème de suivi, en considérant la perspective bayésienne, revient à calculer récursivement la fonction de densité de probabilité, $p(x_k|z_{1:k})$, à partir d'un ensemble de mesures disponibles à un instant k, $z_{1:k} = \{z_i, i = 1, ..., k\}$. En supposant que $p(x_0|z_0) = p(x_0)$ et que la valeur z_0 est disponible, on peut alors, en principe, estimer la valeur $p(x_k|z_{1:k})$ en deux étapes : prédiction et mise à jour.

Si l'on connaît la valeur à un instant $k-1$ de la fonction de densité de probabilité $p(x_{k-1}|z_{1:k-1})$, on peut dès lors inférer sa valeur à un instant k grâce à l'équation de Chapman-Kolmogorov.

$$p(x_k|z_{1:k-1}) = \int p(x_k|x_{k-1})p(x_{k-1}|z_{1:k-1})dx_{k-1} \tag{121}$$

A noter que, dans l'équation précédente, on suppose que $p(x_k|x_{k-1,}z_{1:k-1})$ suit un modèle de Markov d'ordre 1. A l'étape k, on a, à notre disposition, la valeur de z_k de mesure qui sera utilisée pour effectuer la mise à jour. En suivant la règle de Bayès :

$$p(x_k|z_{1:k}) = \frac{p(z_k|x_k)p(x_k|z_{1:k-1})}{p(z_k|z_{1:k-1})} \tag{122}$$

avec

$$p(z_k|z_{1:k-1}) = \int p(z_k|x_k)p(x_k|z_{1:k-1})dx_k \tag{123}$$

Le filtre de Kalman [Kal60] suppose que la densité a posteriori, à chaque étape temps, suit une loi gaussienne et donc paramétrée par sa moyenne et sa covariance. Si $p(x_{k-1}|z_{1:k-1})$ est gaussien il est démontré que $p(x_{k-1}|z_{1:k})$ l'est aussi et ce en supposant certaines conditions :

- v_{k-1} et η_k sont directement issus d'une distribution gaussienne de paramètres connus.
- $f_k(x_{k-1}, v_{k-1})$ est supposée connue de type linéaire pour x_{k-1} et v_{k-1}
- $h_k(x_k, \eta_k)$ est une fonction linéaire connue de x_k et η_k.

Dès lors on obtient des équations précédentes,

$$x_k = F_k x_{k-1} + v_{k-1} \tag{124}$$
$$z_k = H_k x_k + \eta_k \tag{125}$$

avec F_k et H_k des matrices connues définissant ces fonctions. Les covariances de v_{k-1} et η_k sont respectivement Q_{k-1} et R_k. Le filtrage de Kalman considère les équations récursives suivantes pour résoudre le problème :

$$p(x_{k-1}|z_{1:k-1}) = \mathcal{N}(x_{k-1}, m_{k-1|k-1}, \Sigma_{k-1|k-1}) \tag{126}$$
$$p(x_k|z_{1:k-1}) = \mathcal{N}(x_k, m_{k|k-1}, \Sigma_{k|k-1}) \tag{127}$$
$$p(x_k|z_{1:k}) = \mathcal{N}(x_k, m_{k|k}, \Sigma_{k|k}) \tag{128}$$

où

$$m_{k|k-1} = F_k m_{k-1|k-1} \tag{129}$$
$$\Sigma_{k|k-1} = Q_{k-1} + F_k \Sigma_{k-1|k-1} F_k^T \tag{130}$$
$$m_{k|k} = m_{k|k-1} + K_k(z_k - H_k m_{k|k-1}) \tag{131}$$
$$\Sigma_{k|k} = \Sigma_{k|k-1} - K_k H_k \Sigma_{k|k-1} \tag{132}$$

avec $\mathcal{N}(x, m, \Sigma)$ une gaussienne avec pour argument x, moyenne m et covariance Σ.

$$S_k = H_k \Sigma_{k|k-1} H_k^T + R_k \tag{133}$$
$$K_k = \Sigma_{k|k-1} H_k^T S_k^{-1} \tag{134}$$

sont respectivement la covariance du terme $z_k - H_k m_{k|k-1}$, et le gain de Kalman. Ajoutons qu'il s'agit de la solution optimale à notre problème de suivi à condition de respecter les conditions hautement restrictives présentées au dessus. Ceci implique qu'aucun algorithme ne permet d'obtenir de meilleurs résultats dans cet environnement gaussien.

Cependant, dans beaucoup de situations, les conditions imposées ci-dessus ne sont pas respectées et il est impossible d'utiliser le filtrage de Kalman en tant que tel et il nécessite donc quelques approximations : le filtrage de Kalman étendu. Sans rentrer dans les détails,

signalons que ce genre de filtrage utilise une approximation par linéarisation locale d'un problème qui n'est pas linéaire. Le livre de Bar-Shalon et Fortmann [BSF88] décrit un certain nombre d'extensions standards au filtre de Kalman dans le cas notamment non-Gaussien.

Les suivis de formes [GKK05, Dic93, Har93, Gen92, Ric87, KFSW02] et de contours [TS93, BCZ93] par filtrage de Kalman ont été très largement étudiés dans la littérature scientifique. Malheureusement cette approche est limitée et si le mouvement change brusquement le comportement du filtre peut être absurde. Si l'objet, ou point d'intérêt de l'objet a été suivi sur une longue période de temps, la trajectoire filtrée obtenue implique que les prédictions à venir soient également dans la même direction "générale" et ce malgré le fait que le mouvement puisse changer radicalement. Pour cette raison de nombreuses extensions existent dont le filtrage par particules.

4.2.2 Filtrage particulaire

L'approche de Monte Carlo Séquentielle (SMC) [DGA00, ADG01] est aussi connue sous les noms d'algorithme de filtrage particulaire, de Condensation [IB98]... Cette technique implémente un filtrage bayésien récursif dont l'idée principale consiste à représenter la fonction de densité de probabilité désirée par un ensemble d'échantillons aléatoires et de poids associés et de réduire les calculs à ces couples. Soit $\{x^i_{0:k}, w^i_k\}^N_{i=1}$ un ensemble de "mesures aléatoires" associé à la fonction de densité de probabilité a posteriori $p(x_{0:k}|z_{1:k})$, où $\{x^i_{0:k}, i = 0, ..., N_s\}$ est un ensemble de points de support avec leurs poids associés $\{w^i_k, i = 1, ..., N_s\}$ et $x_{0:k} = \{x_j, j = 0, ..., k\}$ l'ensemble d'états du système jusqu'à l'instant k. Les poids sont normalisés de tel façon que $\sum_i w^i_k = 1$. Dès lors, on peut approximer la densité a posteriori, à un instant k par :

$$p(x_{0:k}|z_{1:k}) \approx \sum_{i=1}^{N_s} w^i_k \delta(x_{0:k} - x^i_{0:k}) \tag{135}$$

Le choix des poids n'est pas si aléatoire que cela et fait appel à une technique appelée "l'échantillonage de particules". Ce principe est le suivant : on suppose que $p(x) \propto \pi(x)$ est une densité de probabilité de laquelle il est difficile d'extraire des échantillons, mais qui puisse être évalué. De plus $x^i \sim q(x), i = 1, ..., N_s$ est générée par $q(.)$, dit "densité d'importance". Dès lors, l'approximation des poids est obtenue par :

$$p(x) = \sum_{i=1}^{N_s} w^i \delta(x - x^i) \tag{136}$$

avec

$$w^i \propto \frac{\pi(x^i)}{q(x^i)} \tag{137}$$

correspondant au poids normalisé de la ième particule. Donc, si la particule $x^i_{0:k}$ est générée à partir de la "densité d'importance", $q(x_{0:k}, z_{1:k})$ alors les poids définis ci-dessus sont estimés par

$$w^i_k \propto \frac{p(x^i_{0:k}|z_{1:k})}{q(x^i_{0:k}|z_{1:k})} \tag{138}$$

De plus, si $q(x_k|x_{0:k-1}, z_{1:k}) = q(x_k|x_{k-1}z_k)$, alors la densité d'importance devient dépendante seulement de x_{k-1} et de z_k. Du coup seul x^i_k a besoin d'être enregistré ainsi que l'historique d'observations $z_{1:k-1}$. Le poids modifié devient donc

$$w^i_k \propto w^i_{k-1}\frac{p(z_k|x^i_k)p(x^i_k|x^i_{k-1})}{q(x^i_k|x^i_{k-1}, z_k)} \tag{139}$$

et la densité a posteriori filtrée peut être approximée, par conséquent, par

$$p(x_k|z_{1:k})\sum_{i=1}^{N_s} w^i_k \delta(x_k - x^i_k) \tag{140}$$

Cet algorithme tel qu'il est décrit consiste à propager récursivement les poids et les points de support à chaque mesure. Cependant un problème typique de ce genre d'algorithme vient du fait de la dégénérescence du nuage, qui après quelques itérations, attribue un poids négligeable à toutes les particules sauf à une. Règler ce problème de dégénérescence nécessite de mettre à jour les particules dont l'approximation de $p(x_k|z_{1:k})$ est proche de zéro. De plus une mesure effective doit être créée pour évaluer la dégénérescence d'un nuage [Ber99] :

$$N_{eff} = \frac{N_s}{1 + Var(w^{*i}_k)}$$

Le nuage est considéré comme "décomposé" si $N_{eff} \leq N_s$, avec N_{eff} petit. Le problème de dégénérescence est un des effets indésirables du filtrage particulaire. La solution "brutale" consisterait à choisir une valeur très grande de N_s, cependant cette solution n'est pas pratique et généralement on préférera la solution suivante :

- choisir convenablement la fonction de "densité d'importance" et
- utilisation du rééchantillonnage

Le premier point consiste à choisir $q(x_k|x^i_{k-1}, z_k)$, fonction d'importance de telle façon qu'elle minimise $Var(w^{*i}_k)$ et donc maximise N_{eff}. Cependant ce type de choix n'est pas sans inconvénient majeur car il faut notamment être capable d'échantillonner $p(x_k|x^i_{k-1}, z_k)$.

Le rééchantillonage consiste à réduire le phénomène de dégénérescence en éliminant les particules dont le poids est inférieur à un certain seuil et donc de concentrer les calculs sur les points de poids forts. Bien que les solutions apportées par ce genre d'algorithme réduisent les effets de dégénérescence, cette méthode empêche de paralléliser le problème puisque toutes les particules doivent être traitées de façon combinée.

4.3 Méthode retenue

Nous l'avons vu tout au long de la section précédente, le suivi d'un objet d'intérêt est un problème bien connu et très étudié. Les méthodes généralement retenues vont des algorithmes de suivi basés Filtrage, dont le fameux Filtrage de Kalman [Kal60], aux algorithmes basés modèle [RK95] ou encore des algorithmes particulaires [IB98]. Une nouvelle tendance toutefois s'oriente de plus en plus vers les méthodes basées noyau -kernel-based method- [DRM03] ou encore le tracker SVM proposé par Avidan dans [Avi04] par exemple. Ceci s'explique par le changement de point de vue des auteurs pour ce problème. Il s'agit cette fois ci de suivre un objet en exploitant le modèle appris pour la détection.

4.3.1 Présentation du système

Dans [Avi04], Avidan propose une méthode originale de suivi d'un objet appris défini au préalable : piétons, voiture ou visage. Si l'un objet doit être détecté cela implique l'existence d'un classifieur spécifiquement entraîné pour la reconnaissance de cet objet par rapport au fond. La question sous-jacente est : comment intégrer un détecteur et un "tracker" dans un même système ? Dans un système basé "features", détecteur et suivi sont généralement employés de façon séquentielle. Le classifieur localise un certain nombre de points d'intérêt de l'objet ou l' objet lui même et exécute le suivi ensuite. A l'image suivante le processus de suivi retourne la prédiction de la position au détecteur qui retourne un score. Ce schéma se répète jusqu'à ce que le résultat du score de classification soit inférieur à un seuil prédéfini expérimentalement.

Avidan [Avi04] propose un système "tout en un" où détecteur et suivi sont utilisés pour la détection et le suivi de l'arrière de véhicules par une caméra embarquée. En supposant l'existence d'un classifier SVM entraîné à reconnaître un objet spécifique, il offre un schéma de suivi intitulé SVT -pour Support Vector Tracking- [Avi04] dans lequel il profite de la connaissance apprise par le classifieur pour être utilisé pour le suivi. L'auteur suppose que le mouvement d'un objet appris est faible entre deux images. Après avoir détecté un visage sur une image à l'instant de temps t, Avidan propose de localiser sur l'image de temps $t+1$, la sous-image qui obtient du classifieur le score maximum. D'un point de vue théorique, l'approche est séduisante dans le sens ou détection et suivi utilisent le même modèle mathématique.

L'approche toutefois proposée par Avidan ne gère que des mouvements de type translationnel et ne peut, en tant que telle, être appliquée dans un contexte de surveillance vidéo car l'objet appris peut avoir des mouvements plus complexes -rotation, zoom, ...- vis-à-vis de la caméra. L'approche que nous avons retenue a donc naturellement pour point de départ les travaux d'Avidan. Nous allons plus loin en proposant un modèle de mouvement plus complexe et réaliste. Considérant un classifieur SVM entraîné à reconnaître des visages humains [CBPG03], une méthode de suivi par SVM du visage est proposée par estimation de son modèle de mouvement affine complet. Le système est donc naturellement divisé en deux parties : détection et suivi, les deux par SVM avec le même ensemble de paramètres -noyau, ensemble d'apprentissage, etc-.

Appliqué au problème de surveillance vidéo, nous nous sommes plus particulièrement intéressé au scénario où une personne entre dans une pièce et le problème consiste à localiser son visage en vue de le suivre et de le reconnaître par la suite. Cette même personne peut être animée de mouvement complexe : elle peut s'approcher de la caméra, tourner, ... Dans tous les cas nous supposerons que le mouvement de la caméra est compensé par estimation du modèle global de caméra [DBP01].

4.3.2 Suivi par SVM

Une fois qu'un objet d'intérêt est détecté, le problème du suivi de cet objet intervient. Un classifieur entraîné à reconnaître la classe d'objets particulière peut être utilisé pour le suivi en appliquant ce même classifieur sur le voisinage suivant une tesselation de l'espace de configurations possibles. Nous avons donc développé une méthode basée sur les SVM dont le but consiste à suivre un objet d'intérêt -un visage dans notre cas- grâce à l'estimation du modèle affine complet à 6 paramètres. Nous injectons un classifieur SVM dans un estimateur de flot optique et nous estimons la fonction de classification f par rapport au mouvement affine de l'objet dans l'image.

Comme nous l'avons indiqué dans le chapitre précédent, la classification binaire par SVM [Vap95] est réalisée suivant le signe de la fonction de

$$f(I,\alpha) = \sum_{j=1}^{N} y_j \alpha_j K(I, x_j) + b \tag{141}$$

où x_j sont les N vecteurs de support obtenus après la phase d'apprentissage, y_j leur signe et α_j leur coefficient de Lagrange. $K(I, x_j)$ est le noyau. Ce noyau calcule le produit scalaire entre I, la région de l'image à tester, et les vecteurs de support. Tous les paramètres -noyau, α_j- et les vecteurs de support ont été obtenus lors de l'apprentissage pour le processus de détection (cf Chapitre 3).

Considérons I_{init} la région initiale de l'objet d'intérêt à l'image. En supposant que la position de l'objet cible est proche de la position initiale et suivant le développement usuel de I_{final} en série de Taylor du première ordre, nous avons

$$I_{final} = I_{init} + uI_x + vI_y \tag{142}$$

où I_x et I_y sont les dérivées suivant x et y respectivement. Nous supposerons également que le flot optique (u, v) suit un modèle de mouvement affine complet à 6 paramètres défini par

$$\begin{aligned} u &= (a_0 + a_1 x + a_2 y) \\ v &= (a_3 + a_4 x + a_5 y) \end{aligned} \tag{143}$$

Par définition, le score SVM de I_{final} sera supérieur à celui de I_{init} donc nous avons

$$\sum_{j=1}^{N} y_j \alpha_j K(I_{final}, x_j) = max\{I | \sum_{j=1}^{N} y_j \alpha_j K(I, x_j)\}$$

où I correspond à toutes les imagettes -rétines- de test aux alentours de la position initiale. Si l'on injecte la définition de I_{final} -Eq. (142)- dans l'équation (141) nous avons

$$\sum_{j=1}^{N} y_j \alpha_j K(I_{init} + uI_x + vI_y, x_j) + b$$

que nous devons maximiser. En supposant un noyau polynomial de second ordre donné par la définition suivante $K(x, x_j) = (x \ . \ x_j)^2$, nous introduisons la fonction d'énergie E à maximiser

$$\begin{aligned} E(a_0, .., a_5) &= \sum_{j=1}^{N} y_j \alpha_j K(I_{init} + uI_x + vI_y, x_j) \\ &= \sum_{j=1}^{N} y_j \alpha_j ((I_{init} + uI_x + vI_y) \ . \ x_j)^2 \end{aligned} \tag{144}$$

En dérivant E par rapport à chaque paramètre, $a_i, i = 0, ..., 5$ et en simplifiant les expressions nous avons les équations suivantes

$$
\begin{aligned}
\frac{\partial E}{\partial a_0} &= 2\sum_{j=1}^{N} y_j\alpha_j(I_x \,.\, x_j)((I_{init}+uI_x+vI_y) \,.\, x_j)\\
\frac{\partial E}{\partial a_1} &= 2\sum_{j=1}^{N} y_j\alpha_j(xI_x \,.\, x_j)((I_{init}+uI_x+vI_y) \,.\, x_j)\\
\frac{\partial E}{\partial a_2} &= 2\sum_{j=1}^{N} y_j\alpha_j(yI_x \,.\, x_j)((I_{init}+uI_x+vI_y) \,.\, x_j)\\
\frac{\partial E}{\partial a_3} &= 2\sum_{j=1}^{N} y_j\alpha_j(I_y \,.\, x_j)((I_{init}+uI_x+vI_y) \,.\, x_j)\\
\frac{\partial E}{\partial a_4} &= 2\sum_{j=1}^{N} y_j\alpha_j(xI_y \,.\, x_j)((I_{init}+uI_x+vI_y) \,.\, x_j)\\
\frac{\partial E}{\partial a_5} &= 2\sum_{j=1}^{N} y_j\alpha_j(yI_y \,.\, x_j)((I_{init}+uI_x+vI_y) \,.\, x_j)
\end{aligned}
$$

tel que $\forall i \in \{0,..,5\}\ \frac{\partial E}{\partial a_i} = 0$. Ici u et v sont supposés suivre le modèle décrit par l'équation (143). En vue de la simplification des notations, considérons les matrices $A_{ij,0\leq i,j\leq 5}$, $B_{ij,0\leq i\leq 5,j=1}$ et $C_{ij,0\leq i\leq 5,j=1}$. L'équation précédente devient alors, après développement, de la forme matricielle suivante $CA = B$ avec

$$
\begin{aligned}
C_{00} &= \textstyle\sum_{j=1}^{N} y_j\alpha_j(I_x.x_j)^2(I_x.x_j)^2\\
C_{01} &= \textstyle\sum_{j=1}^{N} y_j\alpha_j(I_x.x_j)^2(xI_x.x_j)^2\\
C_{02} &= \textstyle\sum_{j=1}^{N} y_j\alpha_j(I_x.x_j)^2(yI_x.x_j)^2\\
C_{03} &= \textstyle\sum_{j=1}^{N} y_j\alpha_j(I_x.x_j)^2(I_y.x_j)^2\\
C_{04} &= \textstyle\sum_{j=1}^{N} y_j\alpha_j(I_x.x_j)^2(xI_y.x_j)^2\\
C_{05} &= \textstyle\sum_{j=1}^{N} y_j\alpha_j(I_x.x_j)^2(yI_y.x_j)^2\\
C_{10} &= C_{01}\\
\vdots &\quad \vdots\\
C_{50} &= C_{05}\\
C_{51} &= C_{15}\\
C_{52} &= C_{25}\\
C_{53} &= C_{35}\\
C_{54} &= C_{45}\\
C_{55} &= \textstyle\sum_{j=1}^{N} y_j\alpha_j(yI_y.x_j)^2(yI_y.x_j)^2
\end{aligned}
\tag{145}
$$

,

$$A = (a_0, a_1, a_2, a_3, a_4, a_5)^T$$

et

$$B = \begin{pmatrix} -\sum_{j=1}^{N} y_j \alpha_j (I_x \ . \ x_j)(I \ . \ x_j) \\ -\sum_{j=1}^{N} y_j \alpha_j (xI_x \ . \ x_j)(I \ . \ x_j) \\ -\sum_{j=1}^{N} y_j \alpha_j (yI_x \ . \ x_j)(I \ . \ x_j) \\ -\sum_{j=1}^{N} y_j \alpha_j (I_y \ . \ x_j)(I \ . \ x_j) \\ -\sum_{j=1}^{N} y_j \alpha_j (xI_y \ . \ x_j)(I \ . \ x_j) \\ -\sum_{j=1}^{N} y_j \alpha_j (yI_y \ . \ x_j)(I \ . \ x_j) \end{pmatrix}$$

En vue de l'estimation des paramètres du vecteur A, nous avons développé le processus itératif suivant :

- lors de la première itération, à l'instant t, nous considérons la sous-image I initiale donnée par la position de la détection précédente de visage à l'instant $t-1$.
- Pour chaque pixel (x, y) de l'image I, on estime les coefficients a_i en calculant $A = C^{-1}B$ qui définit ainsi un vecteur de déplacement $(u(x, y), v(x, y))^{(i)}$ selon le modèle (143).
- A partir de là, nous calculons la fonction d'énergie E -Eq. (144)- et nous répétons le procédé jusqu'à ce que la fonction d'énergie devienne stable.
- Afin de calculer les valeurs de tous les pixels I qui n'ont pas forcément des positions entières, une interpolation bilinéaire est réalisée.

4.3.3 Résultats

Le comportement du système de suivi dans le cas où un objet d'intérêt est détecté est illustré à la figure 37. Les résultats présentés sont extraits de deux vidéos de notre corpus de vidéo surveillance enregistrées à 25 images par seconde : "Lionel2" et "Jenny". Ces séquences vidéo contiennent respectivement 500 et 600 images à la résolution CIF.

Les images en haut de la figure 37 de chacune des deux colonnes correspondent aux résultats de détection de visages par SVM suivant le scénario que nous avons défini : le résultat de la détection de visages sur la première images de chaque colonne permet d'initialiser notre algorithme de suivi. Le classifieur SVM utilisé pour la détection conserve le même schéma multi-résolution que nous avons présenté tout au long du chapitre précédent. Les visages détectés sur ces deux exemples sont à la résolution minimale permise par notre système de détection. Insistons sur le fait qu'encore une fois les visages détectés ici ne font pas partie de la base d'apprentissage des visages.

Les images suivantes à gauche correspondent aux résultats de suivi obtenus sur les images 150, 300 et 400 de la séquence "Lionel2". Les images de droite présentent les résultats de suivi obtenus sur les images 150, 300 et 500 de la séquence "Jenny". Notons quelques échecs de suivi sur les images 400 – et 500 – des séquences "Lionel2" et "Jenny" respectivement. Il s'avère que suivant l'amplitude du déplacement de l'objet en mouvement, le

système de suivi peut échouer. Pour pallier à ces échecs, nous avons mis en place un outil coopératif dans ces travaux de thèse : un détecteur de mouvement tourne en parallèle avec le suivi. Ainsi si le suivi décroche, la détection de visages est opérée dans les zones de mouvement uniquement. Le suivi peut alors reprendre normalement.

Les performances globales du système en terme de temps de calcul dépendent fortement de la zone de recherche fixée pour le processus de détection, de la qualité du suivi désirée et du nombre de vecteurs de support obtenus après la phase d'apprentissage. Sans optimisation le processus actuel de détection est capable de classifier toute une image de résolution CIF en 0.15 ms sur un Pentium IV à 1,8Ghz.

La figure 38 présente également des résultats de détection et de suivi sur la séquence vidéo "Michel&Benoit" enregistrée à 25 images par seconde sur une caméra analogique en noir et blanc. Les images, de droite à gauche, et de haut en bas, correspondent aux images 66 -38(a)-, 79 -38(b)-, 109 -38(c)-, 172 -38(d)-, 173 -38(e)-, 189 -38(f)-, 207 -38(g)- et 216 -38(h)- de cette séquence comprenant en tout 250 images. L'image 38(a) est l'image de départ de la séquence vidéo. Un visage frontal est recherché en vain suivant la méthode de détection multi-résolution par SVM telle que nous l'avons présentée dans le chapitre précédent. L'image 38(b) illustre la détection de visages sur un visage pris de face. A cet instant, le détecteur cède la place au suivi comme présenté sur les images 38(c), 38(d) et 38(e). Ce suivi s'opère dans ce cas sur environ 100 images avant de décrocher comme présenté sur l'image 38(b). Le détecteur de visages s'exécute alors sur une dizaine d'images -38(g)- avant de récupérer le visage à l'image numéro 207 -38(h). Le processus d'estimation de l'équation (144) est arrêté au bout de 10 itérations ou quand la fonction d'énergie est devenue stable.

Dans le cas où le suivi décroche deux cas de figure se présentent : soit le suivi atteint une zone immobile c'est à dire sans activité de mouvement, le processus de suivi n'indique alors aucun changement et stagne à la même position à l'image. Dans le cas où, au contraire, la zone est traversée par un objet en mouvement, le suivi est alors incohérent et se déplace de façon incohérente.

Dans le cas d'occultation, notre système de suivi n'est pas pour le moment suffisamment robuste. L'intérêt toutefois de notre approche réside dans la collaboration entre le détecteur et l'algorithme de suivi. En effet lorsque le score du noyau, déterminé par la première localisation d'un visage à l'image, est trop différent d'une image à l'autre lors du processus de suivi, on peut alors conclure que le suivi a décroché. Ce constat permet ensuite de réenclencher le détecteur de visages qui relocalisera le visage perdu... Ce processus a pu être testé sur la base de test présentée à la figure 38.

L'algorithme de suivi pourrait toutefois être optimisé pour les occultations en émettant un modèle a priori sur le mouvement, et notamment sur la continuité de celui-ci. A condition que le mouvement de l'objet ne change pas radicalement pendant la phase d'occultation, une des perspectives de ce travail consisterait à détecter un phénoméne d'occultation et à faire priviligier à l'algorithme de suivi une direction possible de la position de l'objet...

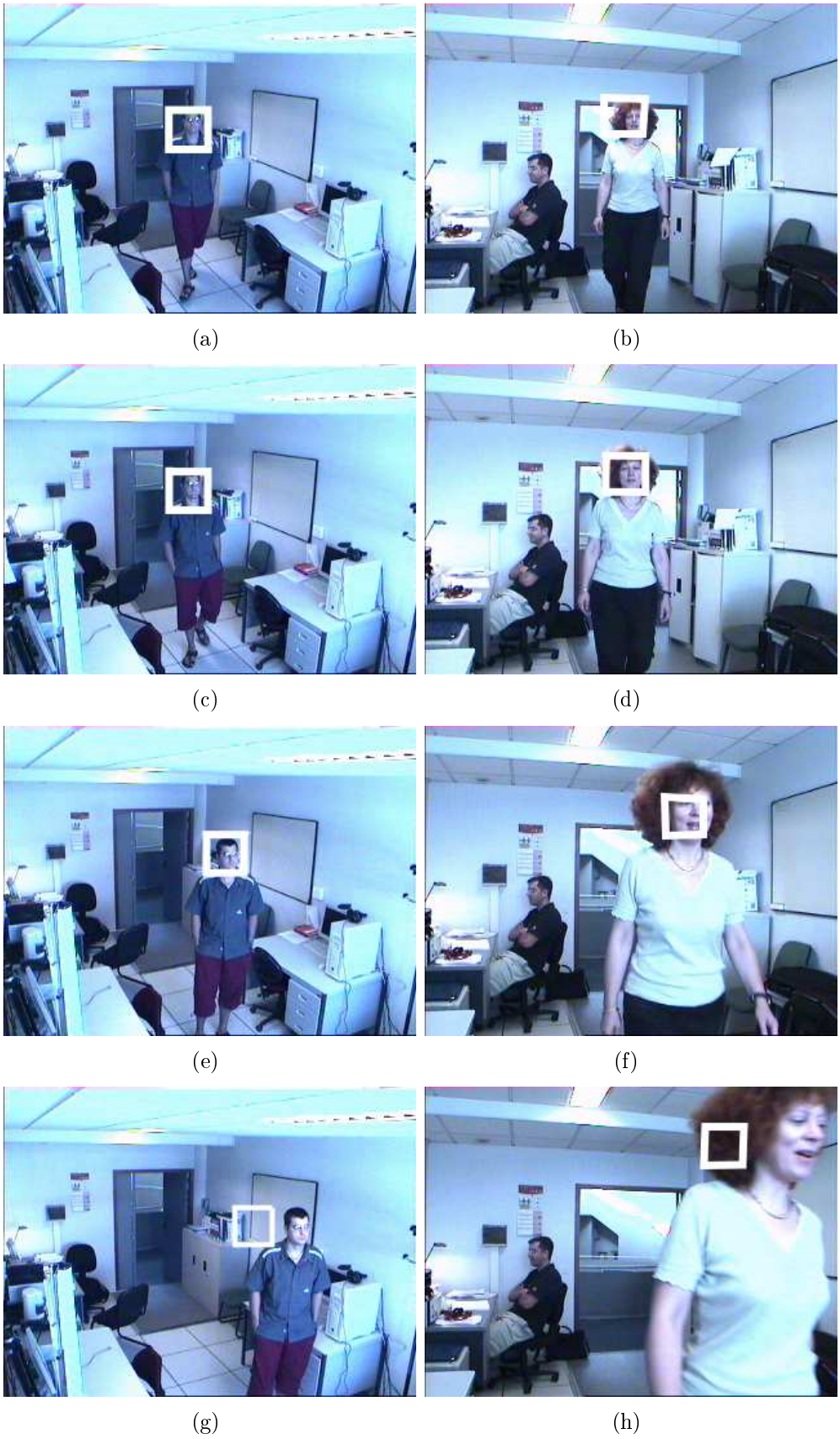
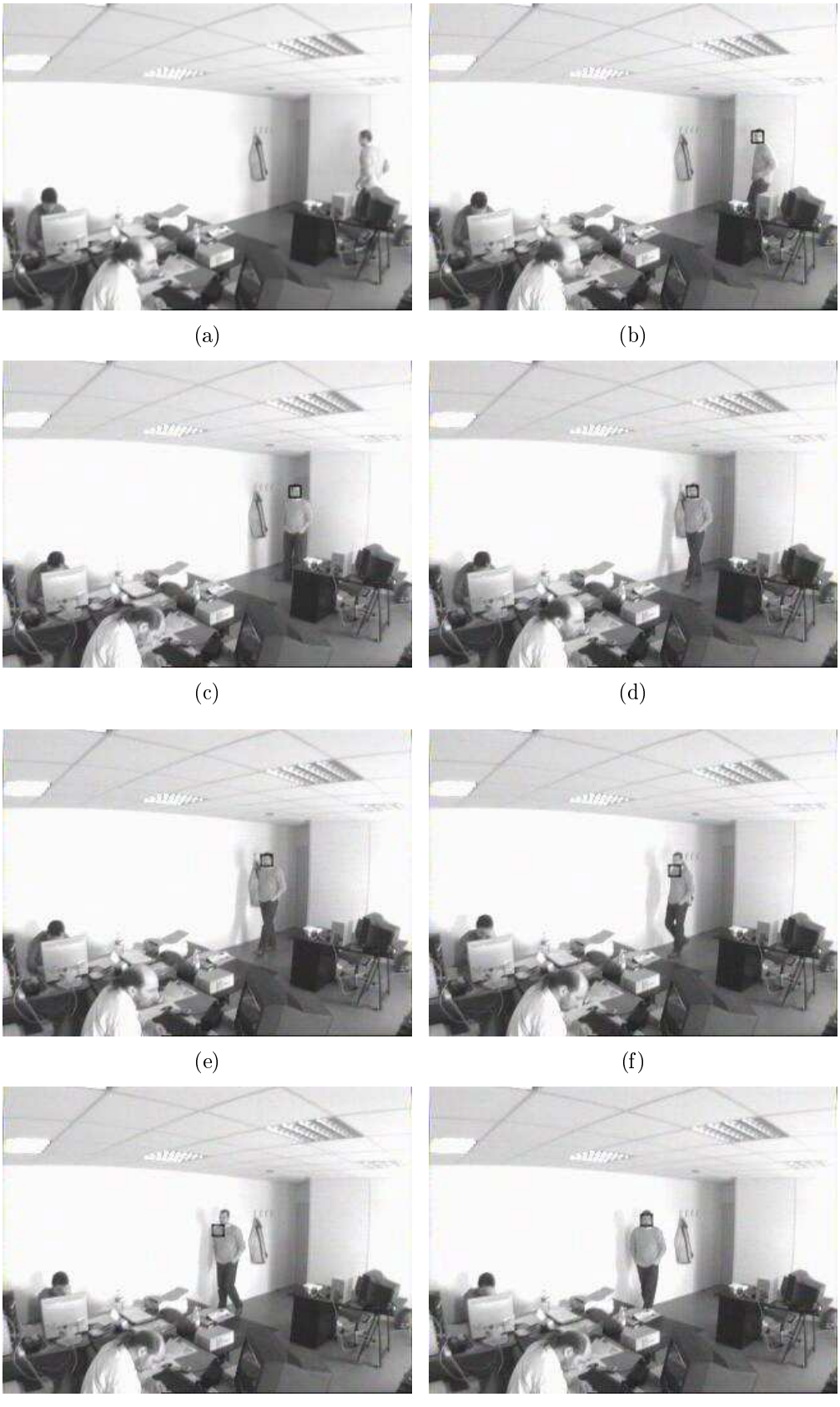


Fig. 37. Résultats de détection et de suivi sur deux vidéos extraites du corpus de surveillance vidéo : "Lionel2" -colonne de gauche- et "Jenny" -colonne de droite.



(a) (b) (c) (d) (e) (f) (g) (h)

Fig. 38. Résultats de détection et de suivi sur la vidéo de surveillance "Michel&Benoit".

4.4 Réduction de l'ensemble des vecteurs de support

Dans cette section, nous allons détailler un autre point de notre étude : la réduction du nombre de vecteurs de support en vue de l'optimisation d'un classifieur SVM. Suite à la présentation de la problématique de ce travail et d'une brêve étude bibliographique, nous présenterons notre méthode d'optimisation basée sur l'élimination, de l'expression de la fonction de décision, des vecteurs de support linéairement dépendants des autres.

4.4.1 Problématique

Dans nos travaux présentés précédemment [CBPG03, CBP05c], nous avons proposé une approche pour la détection de visages à base de Machine à Vecteurs de Support[Vap95] permettant de localiser un visage humain à différentes échelles dans la vidéo. Les résultats de ce système montrent de bonnes performances en terme de rappel/précision pour des vidéos naturelles comme nous avons pu le constater au chapitre précédent. Malheureusement les SVM, du fait de la résolution d'un problème quadratique lors de l'apprentissage, souffrent d'une complexité de calculs qui empêche le déploiement de ce système sur des machines en temps-réel, aussi bien dans la phase d'apprentissage que de classification des échantillons. Des solutions mathématiques complexes existent toutefois, citons notamment les travaux entrepris dans [MBB03]. D'autres solutions peuvent être envisagées s'appuyant cette fois sur le traitement d'image en vue de la simplification du processus de détection et ainsi d'approcher des temps quasi temps-réel (proche de 25 images/s), en ne considérant qu'une zone d'intérêt de l'image : la zone de mouvement [CBPG03] par exemple. Néanmoins, le problème d'optimisation de ces classifieurs en terme d'efficacité de calcul reste ouvert.

Le processus de classification par SVM fait intervenir un sous ensemble d'échantillons d'apprentissage, appelés vecteurs de support (SV), déterminé après la phase d'apprentissage. Comme nous l'avons vu au chapitre précédent, les vecteurs de support sont directement impliqués dans l'expression de la fonction de décision des SVM et donc plus leur nombre est important plus le temps de classification l'est également. Or dans les problèmes de détection/reconnaissance, la machine doit être entraînée avec de grandes bases de données d'apprentissage afin de couvrir au mieux l'espace des formes à apprendre, et le nombre de vecteurs de support peut être directement proportionnel à la taille de la base d'apprentissage.

Le but de notre étude consiste donc ici à proposer une méthode de réduction du nombre de vecteurs de support en vue d'une optimisation de notre détecteur de visages et du suivi. Évidemment ce processus de simplification ne doit pas perdre en efficacité ni en terme de rappel ni en terme de précision.

4.4.2 État de l'art

De nombreux travaux de recherche se sont intéressés à l'optimisation des Machines à Vecteurs de Support en général et à la réduction du nombre de vecteurs de support en particulier. Suite à la phase d'apprentissage, un certain nombre d'échantillons d'apprentissage sont retenus comme vecteurs de support dans la solution de la fonction de décision. Dans le cas de la détection de visages ou pour notre méthode de suivi, le nombre d'images d'apprentissage peut être extrémement grand (> 10000 images) et le nombre de vecteurs de support retenus peut lui aussi être important. A noyau équivalent, les performances de deux systèmes entraînés sur la même base dépendent du nombre de vecteurs de support qu'ils utilisent -cf équation (114).

D'un point de vue temps d'éxécution global du problème de détection, on va chercher à classifier chaque sous-image -patterns- de l'image originale pour repèrer une imagette de visage. Si l'image initiale a une résolution de 320×240 et que la sous-image ou rétine à une taille de 30×30, on doit alors classifier $290 * 210 = 60900$ imagettes. On l'aura compris, réduire le nombre de vecteurs de support pour une application temps réel ou quasi temps réel devient vite une nécessité.

Suivant la littérature, on remarque que les méthodes d'optimisation s'articulent autour de deux grands axes principaux : la première idée consiste à regrouper, en amont de la phase d'apprentissage, les données de l'ensemble d'apprentissage suivant certains critères de proximité ou de vraisemblance afin de réduire la taille des données d'apprentissage et ainsi simplifier le processus d'apprentissage [KH04, SS02]. La deuxième consiste à optimiser, en aval, l'ensemble des vecteurs de support obtenus après apprentissage tout en gardant la "finesse" de l'hyperplan de séparation [Bur96, DGM01].

Les travaux de Wang [WC04] proposent une méthode d'optimisation basée sur le "problème de couverture" [Teg96, Sak84] dans le but d'améliorer à la fois les performances à l'apprentissage et à l'entraînement. Au lieu de résoudre un problème d'optimisation quadratique, les auteurs proposent de le reformuler suivant un problème plus classique de géométrie et de le résoudre. En appliquant cette méthode la distribution statistique de l'ensemble d'apprentissage n'est pas boulversée ce qui permet de conserver les performances en généralisation du système. Les auteurs vont même jusqu'à promettre une amélioration notable pendant la phase d'apprentissage et un nombre de vecteurs de support considérablement réduit.

Koggalage a présenté dans [KH04] une méthode de réduction du nombre de vecteurs de support par pré-regroupement (pre-clustering) de l'ensemble d'apprentissage. Puisque l'hyperplan construit par les SVM n'utilisent qu'une portion -très- réduite de l'ensemble d'apprentissage, l'approche retenue ici consiste à supprimer les échantillons d'apprentissage qui n'auront a priori aucun effet sur la fonction de décision. Koggelage propose alors d'identifier et de supprimer ces échantillons moins importants suivant une technique de clustering proche des k-means. Une fois les regroupements effectués, et les groupes -clusters- déterminés, un rayon variable est identifié à partir du centre des groupes pour reconnaître

les groupes contenant tous les éléments d'une même classe visage ou non-visages -"crisp clusters". Les auteurs retiennent alors les échantillons à la frontière du groupe, éventuellement de possibles vecteurs de support, et retirent des groupes les échantillons du centre. On obtient alors une base d'apprentissage "épurée" des échantillons non pertinents. Cette méthode permet ainsi la réduction du nombre de vecteurs de support sans dégradation des résultats de classification. Cette solution implique également une modification de la fonction de décision pour prendre en compte le rayon déterminé précédemment.

Une autre approche introduite par Ben-Hur dans [ABHV01] consiste à regrouper les échantillons de l'ensemble d'apprentissage suivant un critère de "proximité", d'où découlent des frontières de groupe de forme arbitraire. L'ensemble d'apprentissage est ensuite projeté dans un espace de dimension supérieur -feature space- via un noyau Gaussien, dans le but de déterminer une sphére englobante minimale pour chacun de ses groupes et ne retenir que les échantillons à sa frontière. Les échantillons retenus sont ensuite projetés dans l'espace initial -data space- pour séparer les échantillons d'apprentissage suivant leur classe d'appartenance. Finalement Ben-Hur a étendu ses propres travaux et proposé une démarche algorithmique plus simple en vue d'une implémentation à plus grande échelle...

En terme de rapidité en temps de calcul, les deux méthodes précédentes ont finalement leurs avantages et leurs inconvénients. La méthode de Koggelage dans [KH04] influence principalement la rapidité de l'apprentissage du fait de la quantité réduite d'exemples à apprendre. L'approche de Ben-Hur [ABHV01] implique par exemple d'éxécuter le SVM deux fois, la première pour initialiser les regroupements dans l'ensemble d'apprentissage et la seconde pour entraîner la machine pour la classification.

La méthode proposée par Burges [Bur96] calcule quant à elle, une approximation de la fonction de décision sur un ensemble réduit de vecteurs appartenant à l'ensemble d'apprentissage ou non. Cette ensemble réduit ne comprend donc pas forcément de vecteurs de support, et dans certains cas, ces vecteurs peuvent ne pas appartenir à l'ensemble d'apprentissage et donc sont construits analytiquement. Cet ensemble d'échantillons remplace l'ensemble de vecteurs de support lors de l'étape de classification. L'intérêt de la méthode vient du fait que le nombre des vecteurs soit paramétré et donc soit défini a priori. Un nombre croissant de vecteurs accroit la précision du système de classification, alors que réduire ce nombre augmente la vitesse d'éxécution.

Récemment, les travaux de Downs [DGM01] ont démontré que seuls les vecteurs de support linéairement indépendants pouvaient être retenus dans le choix de l'hyperplan de séparation. Ces travaux ont prouvé que de nombreux vecteurs de support pouvait être éliminés sans pour autant proposer d'exemple concret quant à ce nombre, ni d'application à la réduction. De plus les travaux de Downs ne considèrent que le cas de noyau linéaire et ne traite pas les autres noyaux.

Considérant donc le même cadre d'optimisation du nombre de vecteurs de support que Downs, nous proposons d'étudier le processus de détection de visages et d'étendre les travaux de Downs pour la méthode de réduction du nombre de vecteurs de suppport appliquée à d'autres noyaux plus complexes : les noyaux polynomiaux d'ordre d. Ensuite

nous déterminerons l'impact réel d'une telle réduction sur les performances globales du processus.

4.4.3 Réduction du nombre de vecteurs de support

Les travaux de Downs [DGM01] partent du principe que les algorithmes standards d'entraînement des machines à vecteurs de support produisent en général un nombre plus important de vecteurs de support que réellement nécessaire. Nous avons fait le même constat lors du chapitre précédent en remarquant qu'en fonction d'un paramètre -chunck size- -Cf section 3.3.4- le nombre de vecteurs de support n'était pas identique d'une configuration à l'autre.

Downs a donc proposé de réduire leur nombre par élimination des vecteurs de support linéairement dépendants de la façon suivante : considérons X l'ensemble des vecteurs de support obtenus après apprentissage et X_s l'ensemble des s vecteurs de support de X linéairement indépendants. Si l'un vecteur de support, x_k, est linéairement dépendant des autres vecteurs de support alors la fonction noyau K peut s'écrire de la façon suivante

$$K(x, x_k) = \sum_{\substack{j=1 \\ j \neq k}}^{s} c_j K(x, x_j) \tag{146}$$

où c_i est une constante, x l'échantillon à tester. Dès lors la fonction de décision peut être réécrite comme

$$f(x) = \sum_{\substack{j=1 \\ j \neq k}}^{s} \alpha_j y_j K(x, x_j) + \alpha_k y_k \sum_{\substack{j=1 \\ j \neq k}}^{s} c_j K(x, x_j) + b \tag{147}$$

avec c_j connus on peut poser $\alpha_k y_k c_j = \alpha_j y_j \gamma_k$, ou $\gamma_k = (\alpha_k y_k c_j)/(\alpha_j y_j)$. On a donc

$$f(x) = \sum_{\substack{j=1 \\ j \neq k}}^{s} \alpha_j (1 + \gamma_j) y_j K(x, x_j) + b \tag{148}$$

en dénotant $\overline{\alpha_j} = \alpha_j(1 + \gamma_k)$ on obtient finalement :

$$f(x) = \sum_{\substack{j=1 \\ j \neq k}}^{s} \overline{\alpha_j} y_j K(x, x_j) + b \tag{149}$$

Si l'on compare les équations (141) et (149), on s'aperçoit que le vecteur de support x_k linéairement dépendant n'intervient plus dans l'étape de classification et que seul l'ensemble des vecteurs linéairement indépendants est pris en considération. Cependant les

coefficients multiplicateurs de Lagrange α_j doivent être modifiés afin d'obtenir la fonction de décision simplifiée. La fonction de décision (149) restant inchangée par rapport à (141), les performances de la machine aussi bien en terme de rappel que de précision ne seront pas affectées par cette modification, seules les performances en temps de calcul seront accrues. Considérant la méthode précédemment décrite par Downs, on peut étendre son étude à l vecteurs linéairement dépendant, en définissant les coefficients de la façon suivante : $\overline{\alpha_j} = \alpha_j(1 + \gamma_1 + ... + \gamma_l)$. Donc si $N = s + l$, où on a l vecteurs linéairement dépendants, grâce à cette résolution on économise l calculs de produit scalaire. Dénotons l_{multi} les l coûts de multiplication, s_{add} les s coûts d'addition, etc, l'économie du coût opératoire sera donc :

$$\begin{aligned} \Delta_L = 2l_{multi} + l(N_{multi} &+ (N-1)_{add}) \\ &- (4s_{multi} + s_{add}) \end{aligned} \tag{150}$$

Dans l'équation précédente, le terme de gauche correspond à l'économie du produit scalaire engendré, tandis que la partie de droite traduit le coût rajouté par le calcul des coefficients α_j.

La réduction exacte du nombre de vecteurs de support -Équation (149)- proposée par Downs n'est valide que si le noyau est $K(x, x_j) = x.x_j$. On l'appelle noyau linéaire car l'équation (141) représente une forme linéaire par rapport à x. On le dénotera $K_L(x, x_j)$. Le cas du noyau K_L correspond à la séparabilité linéaire des classes dans l'espace de représentation. Ce cas est assez rare dans la problématique de classification des données bruitées. Relativement au problème de reconnaissance "visage/non visage"[OFG97], les noyaux polynomiaux de degré d se sont avérés efficaces et sont définis par

$$K_{Pd}(x, x_j) = (x.x_j)^d \tag{151}$$

Ainsi dans le cas de la dépendance linéaire du vecteur de support, x_k, par rapport aux autres, on a

$$K_{Pd}(x, x_k) = \left(\sum_{\substack{j=1 \\ j \neq k}}^{s} c_i \; K_L(x, x_j) \right)^d , \tag{152}$$

L'expression ci dessus permet de réécrire la fonction de décision de telle façon

$$\begin{aligned} f(x) &= \sum_{\substack{j=1 \\ j\neq k}}^{s} \alpha_j y_j \; K_{Pd}(x, x_j) \\ &+ \sum_{k=1}^{l} \alpha_k y_k \left(\sum_{\substack{j=1 \\ j\neq k}}^{s} c_{j_k} \; K_L(x, x_j) \right)^d + b. \end{aligned} \tag{153}$$

où s est encore une fois le cardinal de l'ensemble X_s des vecteurs de support linéairement indépendants, l est le cardinal de l'ensemble X_l des vecteurs de support linéairement dépendants. Ici comme dans le cas du noyau linéaire, seuls les vecteurs de base de l'ensemble X_s participent au calcul. De l'équation (153), on constate que l'ensemble des vecteurs de support linéairement indépendants doit être parcouru 2 fois. Ceci dit en rappelant que $K_{Pd}(x, x_j) = (K_L(x, x_j))^d$, on remarque que les noyaux $K_L(x, x_j)$ peuvent être pré-calculés et réutilisés 2 fois dans le calcul du premier et du deuxième terme de la fonction de décision. si l'on compare l'équation précédente (153) avec le calcul direct obtenu sans réduction à savoir

$$f(x) = \sum_{j=1}^{l+s} \alpha_j y_j \; (K_L(x, x_j))^d \tag{154}$$

on remarque que le calcul du produit scalaire $< x, x_j >$ est économisé l fois. Comme dans le cas du noyau linéaire, le coût de la réduction peut se calculer ainsi

$$\begin{aligned} \Delta_{Pd} = l.(N_{multi} &+ (N-1)_{add}) \\ &- l.(s_{multi} + (s-1)_{add}) \end{aligned} \tag{155}$$

Le premier terme de l'équation (155) correspond au coût de calcul des l produits scalaires alors que le deuxième terme décrit le surcoût engendré par le calcul de la somme intérieure dans l'équation (153). De l'équation (155), il est évident que la réduction de coût opératoire n'est possible que si $N > s$. Néanmoins pour N suffisamment grand, le temps d'accès aux données de x et des vecteurs de support x_j est non négligeable. La réduction en terme de temps de calcul est donc possible même si $N = s$ due au pré-calcul des s produits scalaires.

En pratique toutefois, les vecteurs de support ne sont pas "parfaitement" linéairement dépendants mais sont plutôt "presque" linéairement dépendant. La nuance est important pour être tout à fait précis. La démarche que nous avons retenus est la suivante : premièrement nous déterminons par la méthode du pivot de Gauss les coéfficients c_j de l'équation

(146). Sont alors considérés comme linéairement dépendants les vecteurs dont les coefficients déterminés sont inférieur à 0.001 suivant un certains nombre d'axes -moins de 5− et égaux à 0 sur les autres axes. La dépendance n'est pas ici tout à fait stricte en pratique ...

4.4.4 Résultats

Présentation du système de validation

Contrairement à nos travaux précédents [CBPG03, QMS+04] où nous décrivons un processus de détection de visages humains suivant une approche multi-résolution, nous nous plaçons ici dans le cadre de l'approche mono-résolution, c'est à dire qu'à une échelle donnée l'image est scannée par une rétine de taille fixe 30×30 pixels. Les valeurs de luminance de chaque rétine sont ensuite injectées dans notre classifieur entraîné qui nous retourne si oui ou non l'imagette de la rétine contient un visage humain.

Pour effectuer nos tests, nous avons considéré l'implémentation existante SvmFu [M.I01] pour les deux phases : entraînement et classification. Cette implémentation a été entraînée à partir d'une base d'images de visages/non-visages en niveaux de gris extraites de l'exercice 2004 de la campagne d'évaluation internationale TRECVideo [QMS+04]. Cette base de données comprend 10186 images de visages humains extraits manuellement -visages pris de face et centrés sur les yeux, le nez et la bouche- d'une taille de 30×30 pixels et 20586 images de non visages de la même taille.

La machine a ensuite été testée sur une deuxième base, celle de test. Cette dernière comprend 2503 images de visages humains et 90024 images de non visages de la même taille que précédemment.

Toutes les images des bases d'apprentissage et de test subissent un ensemble de prétraitements comme proposé dans [OFG97, CBPG03] afin de corriger les différences de luminosité entre chacune des images et de rehausser les contours des images. Une égalisation d'histogramme suivie d'une normalisation sont ainsi opérées grâce à la librairie ImageMagick [Ima06]. Les implémentations de SVM, comme par exemple SvmFu ou SvmLight [SVM], font appel à différentes optimisations dont celle dite de "chunking". Cette technique consiste à résoudre le problème d'optimisation quadratique localement sur des sous-ensembles -chunks- de l'ensemble total d'apprentissage. L'algorithme commence avec un chunk arbitraire et les vecteurs de support obtenus sont ensuite intégrés lors de la solution finale progressivement [Vap95, Smo98].

Résultats

La première étape de validation consiste à tester différents paramètres pour l'apprentissage (chunk, noyau, ...) et de comparer le nombre de vecteurs de support retenus avec et sans notre méthode de réduction (Tableaux 6 et 7). Ces tests ont été effectués sur un ordinateur de type Intel P4 1,8Ghz, 1Go de RAM pour deux types de noyau polynomiaux

différents : deuxième et troisième ordre.

Nom	Dimension du chunk	Nombre de SVs
Noyau polynomial d'ordre 2		
SVM_{2555}	8000	2555
SVM_{2583}	16000	2583
SVM_{2134}	30000	2134
Noyau polynomial d'ordre 3		
SVM_{2492}	8000	2492
SVM_{2468}	16000	2468
SVM_{2098}	30000	2098

Tab. 6. Variation du nombre de vecteurs de support -SVs- pour un SVM de noyau polynomial de degré 2 et 3, entraîné sur la base TREC Video 2004, en fonction de la dimension du chunk.

Le tableau 6 présente le nombre de vecteurs de support retenus suite à la phase d'apprentissage en fonction de la dimension du "chunk". La colonne de gauche est le nom donné à la configuration d'apprentissage, la colonne du milieu la dimension du chunk et la colonne de droite le nombre total de vecteurs de support obtenus après apprentissage sans optimisations.

Il est ensuite possible de comparer ces résultats avec le tableau 7. Il s'avère que de 40% à presque 70% des vecteurs de support sont linéairement dépendants suivant le type de configuration que l'on choisit. Il est important de noter que, dans tous les cas, la réduction que nous avons obtenue est minimale, car le nombre de vecteurs de support indépendants est égal à la dimension des vecteurs de l'image, c'est à dire 900. A l'heure actuelle les SVMs de noyau polynomial d'ordre 3 donnent les meilleurs taux de rappel/précision comme l'indique le tableau 8.

Nom	Nombre de SVs dépendants	% réduction	% F-Mesure
Noyau polynomial d'ordre 2			
SVM_{2134}	1234	$57,8\%$	79.46%
SVM_{2555}	1655	$64,7\%$	79.26%
SVM_{2583}	1684	$65,2\%$	79.25%
Noyau polynomial d'ordre 3			
SVM_{2098}	1198	$57,8\%$	83.13%
SVM_{2468}	1568	$63,5\%$	83.28%
SVM_{2492}	1592	$63,9\%$	83.28%

Tab. 7. Exemples de réduction du nombre de vecteurs de support en fonctions des configurations d'apprentissage retenues pour les SVM.

Pour ce qui concerne les temps de calcul le tableau 9 résume les résultats des temps de classification des quelques 92000 images de test pour l'ensemble des configurations citées.

Nom	% Rappel	% Précision
Noyau polynomial d'ordre 2		
SVM_{2134}	97%	$67,3\%$
SVM_{2555}	$96,9\%$	$67,2\%$
SVM_{2583}	97%	67%
Noyau polynomial d'ordre 3		
SVM_{2098}	$97,6\%$	$72,4\%$
SVM_{2468}	$97,3\%$	$72,8\%$
SVM_{2492}	$97,3\%$	$72,8\%$

Tab. 8. Taux de rappel et précision obtenus à partir de SVM de noyau polynomial d'ordre 2 et 3 sur la base de test TRECVideo 2004.

Nom	Temps de classification sans/avec simplification (sec.)
Noyaux polynomiaux d'ordre 2	
SVM_{2134}	3101s / 1867s
SVM_{2555}	3892s / 2608s
SVM_{2583}	3986s / 2708s
Noyaux polynomiaux d'ordre 3	
SVM_{2098}	2937s / 1697s
SVM_{2468}	3455s / 2176s
SVM_{2492}	3488s / 2228s

Tab. 9. Résultats des temps de classification obtenus sur l'ensemble de la base de test TREC Video 2004.

Les temps de calcul correspondant à cette réduction, pour la classification de la base de test sont présentés à titre indicatif dans le tableau 9. En fait, comme nous l'avons expliqué précédemment la réduction du temps de calcul dans le cas où $N = s$ (Équation 155) est due à l'économie de l'accès aux données.

Entraînement itératif

Au cours de nos expérimentations, nous nous sommes également posés la question de savoir quel serait le comportement d'un classifieur SVM suite à un apprentissage itératif c'est à dire en entraînant le classifieur sur une base de vecteurs de support d'un entraînement précédent. Au vu des remarques précédentes concernant les méthodes d'optimisation, les résultats fournis pendant l'apprentissage ne sont qu'une approximation de la solution globale. Si l'on regarde les résultats obtenus, que nous allons présenter dans la suite de cette étude, il semblerait que cette approximation s'affine en proposant un schéma itératif d'entraînement.

Dans les résultats que nous allons présenter nous avons eu recours à deux bases d'apprentissage, celle du MIT et celle de TRECVideo 2004. Pour chacune des bases nous entraînons notre classifieur et obtenons, pour des paramètres donnés d'apprentissage un ensemble de vecteurs de support. Cet ensemble de vecteurs est ensuite injecté comme base d'apprentissage avec les mêmes paramètres d'apprentissage et on s'aperçoit alors que le nombre de vecteurs de support diminue tout en offrant de façon générale une amélioration des taux de rappel et de précision. Le schéma itératif d'entraînement tel qu'il est décrit ici, est stoppé lorsque le nombre de vecteurs de support ne diminue plus.

Evaluation sur la base MIT

Les tableaux 10, 11, 12 et 13 résument les résultats obtenus sur la base MIT en effectuant des entraînements itératifs pour des noyaux linéaires -tableaux 10 et 11-, polynomiaux d'ordre 2 -tableau 12- et polynomiaux d'ordre 3 -tableau 13. Tous les tableaux sont structurés de la façon suivante : les deux premières colonnes de ces tableaux indiquent le nombre de vecteurs de support retenu pour l'itération n en fonction de l'étiquette qui leur a été donnée pendant l'apprentissage. La colonne $\#SVs$ indique le nombre de vecteurs de support au total à la fin de l'itération, R et P les taux de rappel et précision obtenus pour cette configuration. La colonne ΔSVs indique le pourcentage de réduction réalisé et les colonnes ΔR et ΔP le pourcentage d'augmentation ou de réduction. Enfin les colonnes t et Δt soulignent respectivement le temps et la réduction en temps pour la classification opérée. Les valeurs de tolérance et d'epsilon sont purement arbitraires et liées au logiciel que nous avons utilisé, SvmFu dans notre cas. Ceci dit, nous retrouvons ces paramètres dans autres implémentations SVM et notamment SVMLight.

	$+1$	-1	$\#SVs$	R	P	ΔSVs	ΔR	ΔP	t	*F-Mesure*
	2429	4548	489	68.6%	64.2%				14s	66.32%
1	199	290	363	66.7%	63.5%	-25.8%	-1.9%	-0.7%	10s	65.06%
2	140	223	331	66.7%	63.2%	-32.3%	-1.9%	-1.0%	9s	64.90%
3	128	203	317	66.3%	63.1%	-35.2%	-2.3%	-1.1%	6s	64.66%
4	122	195	303	67.1%	63.7%	-38.0%	-1.5%	-0.5%	6s	65.35%
5	120	183	292	67.4%	62.9%	-40.3%	-1.2%	-1.3%	6s	65.07%
6	117	175	287	64.8%	62.7%	-41.3%	-3.8%	-1.5%	6s	63.73%
7	116	171	281	66.3%	62.9%	-42.5%	-2.3%	-1.3%	6s	64.55%
8	113	168	274	67.8%	64.4%	-44.0%	-0.8%	$+0.2\%$	6s	66.05%
9	112	162	274	67.4%	64.0%	-44.0%	-1.2%	-0.2%	6s	65.65%

Tab. 10. *Résumé des résultats des dimensions de l'ensemble des vecteurs de support pour* 9 *itérations pour un noyau linéaire, avec une taille de chunk de* 4000*, une tolérance de* $10E-4$ *et une valeur d'epsilon de* $10E-20$*.*

Pour les noyaux linéaires et à la vue des résultats des tableaux 10 et 11, les taux de rappel et de précision sont globalement maintenus pour une réduction de l'ordre de 45% du nombre de vecteurs de support. Dans l'expérience menée et résumée au tableau 10, le nombre de vecteurs de support retenus passe de 489 -suite à la phase d'apprentissage- à 274 avec notre schéma itératif d'entraînement. Dans l'expérience résumée au tableau 11,

le nombre de vecteurs de support passe de 534 à 308 après 10 itérations ce qui permet une réduction de près de 42% du nombre de vecteurs de support !

	$+1$	-1	$\#SVs$	R	P	ΔSVs	ΔR	ΔP	t	*F-Mesure*
	2429	4548	534	65.5%	63.5%				$16s$	64,48%
1	255	309	378	65.9%	63.3%	−29.2%	+0.4%	−0.2%	$10s$	64,57%
2	144	234	349	65.0%	62.7%	−34.6%	−0.5%	−0.8%	$8s$	63,82%
3	134	215	342	64.8%	62.7%	−35.6%	−0.7%	−0.8%	$8s$	63,73%
4	131	211	331	65.0%	62.6%	−38.0%	−0.5%	−0.9%	$8s$	63,77%
5	127	204	322	65.5%	62.8%	−39.7%	±0%	−0.7%	$8s$	64,12%
6	125	197	317	64.6%	62.5%	−40.6%	−0.9%	−1.0%	$7s$	63,53%
7	124	193	313	64.8%	62.6%	−41.4%	−0.7%	−0.9%	$7s$	63,68%
8	122	191	310	65.0%	62.7%	−41.9%	−0.5%	−0.8%	$7s$	63,82%
9	120	190	308	64.8%	62.5%	−42.3%	−0.7%	−1.0%	$7s$	63,62%
10	118	190	308	65.0%	62.7%	−42.3%	−0.5%	−0.8%	$7s$	63,82%

Tab. 11. *Résumé des résultats des dimensions de l'ensemble des vecteurs de support pour* 10 *itérations pour un noyau linéaire, avec une taille de chunk de* 1000*, une tolérance de* $10E-6$ *et une valeur d'epsilon de* $10E-16$*.*

	$+1$	-1	$\#SVs$	R	P	ΔSVs	ΔR	ΔP	t	*F-Mesure*
	2429	4548	424	88.6%	92.1%				$11s$	90.31%
1	188	236	296	92.8%	91.0%	−30.2%	+4.2%	−1.1%	$6s$	91.89%
2	129	167	273	88.4%	86.5%	−35.6%	−0.2%	−5.6%	$5s$	87.43%
3	119	154	268	89.4%	84.9%	−36.8%	+0.8%	−7.2%	$5s$	87.09%
4	116	152	268	89.2%	89.2%	−36.8%	+0.6%	−2.9%	$5s$	89.2%
-										

Tab. 12. *Résumé des résultats de la dimension de l'ensemble des vecteurs de support pour* 4 *itérations pour un noyau polynomial d'ordre 2, avec une taille de chunk de* 2000*, une tolérance de* $10E-4$ *et une valeur d'epsilon de* $10E-20$*.*

	$+1$	-1	$\#SVs$	R	P	ΔSVs	ΔR	ΔP	t	Δt
	2429	4.548	371	92.0%	95.1%				$9s$	
1	164	207	327	92.6%	97.5%	−11.9%	+0.6%	+2.4%	$7s$	−22%
2	151	176	305	92.2%	96.7%	−17.8%	+0.2%	+1.6%	$6s$	−33%
3	140	165	297	92.0%	98.1%	−19.9%	±0%	+3.0%	$6s$	−33%
4	136	161	292	94.1%	99.4%	−21.3%	+2.1%	+4.4%	$6s$	−33%
5	133	159	289	91.4%	96.9%	−22.1%	−0.6%	+1.8%	$6s$	−33%
6	131	158	285	94.3%	90.2%	−23.2%	+2.3%	+5.1%	$6s$	−33%
7	129	156	285	98.6%	95.8%	−23.2%	−3.4%	+0.7%	$6s$	−33%

Tab. 13. *Résumé des résultats des dimensions de l'ensemble des vecteurs de support pour différentes itérations pour un noyau polynomial d'ordre 3, avec une taille de chunk de* 8000*, une tolérance de* $10E-4$ *et une valeur d'epsilon de* $10E-28$*.*

Dans le cas des expériences menées sur des noyaux polynomiaux d'ordre 2 et résumées dans le tableau 12, les taux de rappel et de précision sont même améliorés allant pour le dernier cas jusqu'à une augmentation du taux de précision de l'ordre de 7%. Après 4 itérations le nombre de vecteurs de support se stabilise pour rester de l'ordre de 268 vecteurs pour le premier entraînement. Il en va de même pour notre deuxième expérience, le nombre de vecteurs de support se stabilisant dès la quatrième itération. Dans le cas de noyau polynomial d'ordre supérieur -3 par exemple- la réduction n'est pas aussi prononcée -de l'ordre de 33% tout de même, les indices de rappel et de précision n'étant pas cette fois-ci autant boulversés que précédemment.

Nous avons également testé sur d'autres implémentations des SVM et nous avons pu faire le même constat. Les résultats présentés ici se limitent donc pour l'implémentation SvmFu que nous avons retenue dans notre rapport mais les résultats sont tout à fait comparables avec d'autres implémentations.

Evaluation sur la base TRECVideo 2004

Nous avons ensuite effectué le même processus mais cette fois ci sur la base d'entraînement de TRECVideo 2004 comme décrite précédemment. Les tableaux 14, 15, et 16 reprennent la convention de nommage telle que décrite précédemment.

Les résultats présentés au tableau 14 indiquent que les taux de rappel et de précision varient finalement peu autour des indices initiaux. Après 29 itérations le nombre de vecteurs de support est diminué de 43% avant d'atteindre un état stable. Notons que les meilleurs résultats que nous ayons pu obtenir pour cette expérimentation ont été obtenu dès la première itération où une réduction de 26% a été constatée alors que l'indice de rappel a pour sa part très légérement augmenté.

Pour ce qui concerne le tableau 15 qui proposent les résultats obtenus sur un classifieur SVM entraîné à partir d'un noyau polynomial d'ordre 3, les taux de rappel et de précision sont encore maintenus avec une dégradation maximale de l'ordre de 0.5% pour le rappel et de 3.3% pour la précision. Après 20 itérations, le nombre de vecteurs de support est conservé pour arriver finalement à une réduction de l'ordre de 38%. Encore une fois la première itération permet d'obtenir une réduction significative avec une légère différence des taux de rappel et de précision. On peut constater une augmentation du taux de rappel lors de la première itération...

Enfin le tableau 16 est consacré pour sa part à un noyau polynomial d'ordre 4. vis-à-vis du tableau précédent les taux de rappel sont sensiblement équivalents mais les taux de précision par contre subissent une augmentation non négligeable. Le meilleur choix dans ce cas précis est obtenu lors de la vingtième itération quand le nombre de vecteurs de support est réduit de 22% alors ques l'indice de rappel est diminué de 0.1% et le taux de précision amélioré de 5.2%.

	$+1$	-1	$\#SVs$	R	P	ΔSVs	ΔR	ΔP	t	Δt
	10186	20586	2583	97.0%	67.2%				$720s$	
1	1110	1473	1899	97.1%	66.7%	-26.5%	$+0.1\%$	-0.5%	$526s$	-30%
2	874	1025	1770	96.5%	65.6%	-31.5%	-0.5%	-1.6%	$491s$	-32%
3	823	947	1696	96.7%	65.2%	-34.3%	-0.3%	-2.0%	$472s$	-34%
4	789	907	1649	96.6%	64.3%	-36.2%	-0.4%	-2.9%	$458s$	-36%
5	768	881	1629	96.8%	65.3%	-36.9%	-0.2%	-1.9%	$451s$	-37%
6	763	866	1610	96.8%	64.2%	-36.7%	-0.2%	-3.0%	$447s$	-38%
7	758	852	1595	96.9%	64.7%	-38.3%	-0.1%	-2.5%	$442s$	-39%
8	752	843	1580	96.6%	64.5%	-38.8%	-0.4%	-2.7%	$439s$	-39%
9	749	831	1573	96.5%	63.7%	-39.1%	-0.5%	-3.5%	$436s$	-39%
10	746	827	1564	96.8%	63.9%	-39.5%	-0.2%	-3.3%	$434s$	-40%
11	744	820	1557	96.6%	64.6%	-39.7%	-0.4%	-2.6%	$431s$	-40%
12	744	813	1549	96.6%	64.0%	-40.0%	-0.4%	-3.2%	$430s$	-40%
13	741	808	1540	96.6%	63.7%	-40.4%	-0.4%	-3.5%	$427s$	-41%
14	736	804	1534	96.8%	63.6%	-40.6%	-0.2%	-3.6%	$426s$	-41%
15	734	800	1526	96.7%	64.3%	-40.9%	-0.3%	-2.9%	$423s$	-41%
16	730	796	1521	96.6%	62.5%	-41.1%	-0.4%	-4.7%	$422s$	-41%
17	728	793	1513	96.5%	63.6%	-41.4%	-0.5%	-3.6%	$419s$	-42%
18	724	789	1508	96.7%	64.4%	-41.6%	-0.3%	-2.8%	$418s$	-42%
19	723	785	1501	96.8%	63.0%	-41.9%	-0.2%	-4.2%	$416s$	-42%
20	720	781	1496	96.6%	63.3%	-42.1%	-0.4%	-3.9%	$414s$	-43%
21	720	776	1494	96.5%	64.6%	-42.2%	-0.5%	-2.6%	$414s$	-43%
22	720	774	1492	96.6%	63.4%	-42.2%	-0.4%	-3.8%	$414s$	-43%
23	719	773	1489	96.4%	62.3%	-42.4%	-0.6%	-4.9%	$413s$	-43%
24	718	771	1485	96.7%	62.6%	-42.5%	-0.3%	-4.6%	$413s$	-43%
25	716	769	1482	96.7%	62.9%	-42.6%	-0.3%	-4.3%	$412s$	-43%
26	714	768	1479	96.7%	62.8%	-42.7%	-0.3%	-4.4%	$411s$	-43%
27	714	765	1476	96.4%	62.9%	-42.9%	-0.6%	-4.3%	$409s$	-43%
28	712	764	1474	96.4%	63.5%	-42.9%	-0.6%	-3.7%	$409s$	-43%
29	712	762	1474	96.4%	62.9%	-42.9%	-0.6%	-4.3%	$409s$	-43%

Tab. 14. *Résumé des résultats de dimension de l'ensemble de vecteurs de support obtenus à partir de la base TRECVideo 2004. Noyau polynomial de degré* 2*, une taille de chunk de* 16000*, une tolérance de* $10E-4$ *et une valeur epsilon de* $10E-20$*.*

4.5 Conclusion

Ainsi dans ce chapitre nous avons proposé un schéma complet de détection et de suivi d'objet d'intérêt : un visage humain dans notre cas. A partir d'un même cadre mathématique que la détection de visages, les Machines à Vecteurs de Support, nous avons formulé le problème de suivi et proposé une solution pour l'estimation du modèle affine complet de premier ordre. Le modèle adopté permet ainsi de suivre des visages pris de face sous différents angles, sous différents mouvements ce qui permet de mieux répondre aux variations du mouvement d'un visage dans une scène de vidéo surveillance. Ce suivi nous

	$+1$	-1	$\#SVs$	R	P	ΔSVs	ΔR	ΔP	t	Δt
	10186	20586	2504	97.4%	74.4%				$695s$	
1	1077	1427	1894	97.2%	74.7%	-24.4%	-0.2%	$+0.3\%$	$526s$	-24%
2	878	1016	1764	97.3%	74.1%	-29.6%	-0.1%	-0.3%	$490s$	-29%
3	828	936	1703	97.3%	73.0%	-32.0%	-0.1%	-1.4%	$473s$	-32%
4	809	894	1676	97.3%	73.8%	-33.1%	-0.1%	-0.6%	$465s$	-33%
5	797	879	1646	97.4%	73.8%	-34.3%	$\pm 0\%$	-0.6%	$456s$	-34%
6	788	858	1630	97.4%	73.3%	-34.9%	$\pm 0\%$	-1.1%	$453s$	-35%
7	783	847	1611	97.2%	72.6%	-35.7%	-0.2%	-1.8%	$447s$	-36%
8	776	835	1604	97.4%	73.7%	-35.9%	$\pm 0\%$	-0.7%	$445s$	-36%
9	774	830	1593	97.2%	72.1%	-36.4%	-0.2%	-3.3%	$442s$	-36%
10	771	822	1583	97.1%	72.5%	-36.8%	-0.3%	-2.9%	$440s$	-37%
11	769	814	1580	97.2%	73.8%	-36.9%	-0.2%	-0.6%	$438s$	-37%
12	767	813	1571	97.3%	73.4%	-37.3%	-0.1%	-1.0%	$436s$	-37%
13	760	811	1563	97.2%	72.9%	-37.6%	-0.2%	-1.5%	$434s$	-38%
14	757	806	1558	97.3%	72.8%	-37.8%	-0.1%	-1.6%	$432s$	-38%
15	755	806	1552	97.1%	73.0%	-38.0%	-0.3%	-1.4%	$431s$	-38%
16	753	799	1550	97.4%	73.1%	-38.1%	$\pm 0\%$	-1.3%	$430s$	-38%
17	752	798	1546	97.0%	72.4%	-38.3%	-0.4%	-2.0%	$429s$	-38%
18	751	795	1544	96.9%	72.1%	-38.3%	-0.5%	-2.3%	$428s$	-38%
19	751	793	1542	97.5%	72.6%	-38.3%	$+0.1\%$	-1.8%	$428s$	-38%
20	751	791	1542	97.1%	72.2%	-38.3%	-0.3%	-2.2%	$428s$	-38%

Tab. 15. *Résumé des résultats de dimension de l'ensemble de vecteurs de support obtenus à partir de la base TRECVideo 2004. Noyau polynomial de degré* 3*, une taille de chunk de* 4000*, une tolérance de* $10E-4$ *et une valeur epsilon de* $10E-28$*.*

permet ensuite d'indexer automatiquement la vidéo en fonction de la présence ou non d'un individu à l'écran.

Prolongeant les travaux de Downs nous avons développé une méthode de réduction du nombre de vecteurs de support par élimination des vecteurs linéairement dépendants pour des classifieurs de noyaux polynomiaux de degré d. Appliqué au problème de la détection puis de suivi de visages humains, cette simplification permet de réduire les temps de calcul tout en laissant intact les taux de rappel/précision obtenus sans cette optimisation. Cette propriété est vérifiée car il s'agit d'une reformulation de la fonction de décision. Une autre optimisation est également possible en sauvegardant temporairement et de façon judicieuse certains calculs.

Une évaluation plus complète devrait permettre d'étudier la sensibilité de ce détecteur aux occultations de visages par d'autres visages. La coopération entre le détecteur et l'algorithme de suivi permet en tout cas de récupérer un visage même si l'algorithme de suivi décroche.

En complément de ces travaux, nous avons également proposé un schéma de réentraînement itératif dans le but de conserver le nombre "minimal" de vecteurs de support. Ce

	+1	−1	$\#SVs$	R	P	ΔSVs	ΔR	ΔP	t	Δt
	10186	20586	2112	97.9%	71.4%				586s	
1	961	1151	1890	97.8%	74.6%	−10.5%	−0.1%	+3.2%	526s	−10%
2	886	1004	1813	97.8%	74.4%	−14.2%	−0.1%	+3.0%	503s	−14%
3	860	953	1779	97.9%	74.4%	−15.8%	±0%	+3.0%	494s	−16%
4	844	935	1753	98.0%	74.7%	−17.0%	+0.1%	+3.3%	487s	−17%
5	832	921	1736	97.8%	74.5%	−17.8%	−0.1%	+3.1%	482s	−18%
6	826	910	1722	97.9%	75.0%	−18.5%	±0%	+3.6%	478s	−18%
7	822	900	1714	97.8%	74.5%	−18.8%	−0.1%	+3.1%	475s	−19%
8	818	896	1706	97.8%	74.5%	−19.2%	−0.1%	+3.1%	473s	−19%
9	815	891	1698	97.8%	74.8%	−19.6%	−0.1%	+3.4%	472s	−19%
10	810	888	1692	97.8%	74.7%	−19.9%	−0.1%	+3.3%	470s	−20%
11	807	885	1686	97.8%	75.1%	−20.2%	−0.1%	+3.7%	468s	−20%
12	804	882	1681	97.8%	75.2%	−20.4%	−0.1%	+3.8%	467s	−20%
13	802	879	1675	97.7%	75.7%	−20.7%	−0.2%	+4.3%	465s	−21%
14	802	873	1672	97.6%	75.2%	−20.8%	−0.3%	+3.8%	464s	−21%
15	800	872	1668	97.9%	74.9%	−21.0%	±0%	+3.5%	463s	−21%
16	796	872	1665	97.7%	75.2%	−21.2%	−0.2%	+3.8%	463s	−21%
17	794	871	1660	97.7%	74.8%	−21.4%	−0.2%	+3.4%	461s	−21%
18	792	868	1656	97.8%	75.0%	−21.6%	−0.1%	+3.6%	460s	−22%
19	790	866	1652	97.8%	75.6%	−21.8%	−0.1%	+4.2%	459s	−22%
20	789	863	1651	97.8%	76.6%	−21.8%	−0.1%	+5.2%	458s	−22%
21	788	863	1642	97.8%	75.9%	−22.3%	−0.1%	+4.5%	456s	−22%
22	786	856	1640	97.7%	75.4%	−22.3%	−0.2%	+4.0%	456s	−22%
23	785	855	1633	97.7%	75.6%	−22.7%	−0.2%	+4.2%	454s	−23%
24	779	854	1628	97.7%	75.3%	−22.9%	−0.2%	+3.9%	457s	−22%
25	778	850	1625	97.8%	75.4%	−23.1%	−0.1%	+4.0%	455s	−22%
26	777	848	1624	97.8%	75.5%	−23.1%	−0.1%	+4.1%	453s	−23%
27	776	848	1623	97.7%	75.4%	−23.2%	−0.1%	+4.0%	459s	−22%
28	776	847	1620	97.7%	75.4%	−23.3%	−0.1%	+4.0%	451s	−23%
29	775	845	1620	97.6%	75.6%	−23.3%	−0.2%	+4.2%	452s	−23%

Tab. 16. *Résumé des résultats de dimension de l'ensemble de vecteurs de support obtenus à partir de la base TRECVideo 2004. Noyau polynomial de degré* 4*, une taille de chunk de* 32000*, une tolérance de* $10E-12$ *et une valeur epsilon de* $10E-44$*.*

schéma basé sur un réapprentissage, à l'instant t, des vecteurs de support obtenus à $t-1$, permet principalement de réduire les effets des optimisations prévues lors du développement des implémentations SVM et notamment de la résolution du problème quadratique.

Les résultats que nous avons obtenus, à la fois sur la méthode par dépendance linéaire et sur le schéma itératif de réentraînement, sont tout à fait convaincants aussi bien pour la détection que pour le suivi. Le surcoût opératoire induit par l'élimination des vecteurs de support linéairement dépendants pouvant être optimisé par l'accès à la relecture des données notamment. Dans le cas de la réduction par réapprentissage itératif, il semble, au vue des expérimentations que nous avons menées, que nous ayons obtenu l'ensemble des

vecteurs de support le plus petit possible ! En confrontant les deux méthodes que nous avons proposé il semble que la deuxième offre des résultats meilleurs tout en offrant une stabilité pour les indices de rappel et de précision.

En perspective de ce travail mené sur la réduction des vecteurs de support, il pourrait s'avérer intéressant d'utiliser ces deux méthodes non plus en paralléle comme nous venons de le faire mais plutôt d'étudier si la réduction de l'ensemble des vecteurs de support pourrait être encore plus importante si l'on combinait les deux méthodes présentées précédemment.

Par manque de temps nous n'avons pas pu intégrer l'outil de suivi et la réduction de l'ensemble des vecteurs de support dans le système complet temps-réel. Cette intégration reste dans la perspective de ce travail.

Conclusion

Un système de vidéo surveillance intelligent relève d'un ensemble de problématiques différentes à résoudre : détection de mouvement, détection d'un ou plusieurs objets d'intérêt et suivi de cet objet. Proposer un système de vidéo surveillance intelligent c'est répondre à l'ensemble de ces points. Tout au long de ce travail, nous avons donc cherché à proposer des solutions à base d'apprentissage statistique comme cadre de travail pour résoudre chacun de ces problèmes.

Avant toute chose nous nous sommes intéressés au contexte applicatif de ce travail de thèse et nous avons du prendre en compte les points suivants : caméras fixes de faible coût, possibilité de réutiliser un système de vidéo surveillance déjà existant avec des caméras de nature différente et de qualité inégale, temps de calculs rapides pour les traitements de base. Cet ensemble de caractéristiques nous a ainsi permis de constituer un cahier des charges applicatif et théorique. Avant de réellement commencer nos travaux de recherche nous avons également établi une analyse de l'existant en matière de système de vidéo-surveillance dit "intelligent". Cette étude a par ailleurs été reproduite au tout premier chapitre de ce mémoire. Nous avons ensuite proposé notre propre architecture logiciel et théorique du système mono-caméra que nous avons retenu.

Dans un second temps, nous avons étudié un des problèmes majeurs pour la vidéo surveillance : la **détection de mouvement**. Après avoir défini les enjeux de ce problème et les principales méthodes existantes dans la littérature scientifique nous avons présenté, dans le Chapitre 2, notre technique d'extraction des masques d'objets mobiles basée sur un modèle de mélange de lois Gaussiennes associé à une régularisation Markovienne. Un apprentissage non-supervisé a donc finalement été retenu pour cette problèmatique afin de constituer une modélisation en deux classes : "objet en mouvement" et "fond".

Dans le modèle de détection de mouvement, nous avons préconisé un schéma par ré-entraînement qui permet de répondre au mieux, selon nous, à l'aspect générique auquel doit répondre une solution de vidéo surveillance. Puisque nous sommes limités en terme de puissance de calcul, nous avons cherché à proposer une solution à faible coût calculatoire. Contrairement aux travaux antérieurs de Grimson et Stauffer, nous proposons un schéma de ré-entraînement conservant le même nombre de Gaussiennes dans le mélange que lors de la phase d'apprentissage initial. Cette solution offre un cadre mathématique pertinent qui répond à la fois à la problématique des faibles et des fortes variations de luminance entre deux images enregistrées à des fréquences temporelles variées. Le choix de l'espace

de représentation retenu est capital ici : ainsi notre système de détection de mouvement considère les valeurs de luminance ce qui permet, premièrement, de s'affranchir des deux autres composantes, U et V, et deuxièmement d'obtenir un gain non négligeable sur les temps de calcul. Les résultats s'avèrent néammoins stables.

La coopération entre détecteur de mouvement et régularisation Markovienne permet ensuite de réduire le nombre de fausses détections. Les résultats des masques de mouvement obtenus apparaissent clairement plus compacts aussi bien spatialement que temporellement que les résultats obtenus par le détecteur de mouvement seul. Notre méthode coopérative intègre donc de façon plus correcte les informations de détection de mouvement observables que l'approche de détection seule. De plus, la régularisation donne des résultats de bien meilleur qualité sur l'intégrité de l'objet et dans l'élimination des fausses détections.

A la différence des approches connues dans la littérature, nous avons déterminé une **fonctionnelle d'énergie à minimiser à partir des résultats de la détection**. L'affranchissement du terme d'attache aux données par les mesures de luminance permet d'économiser en temps de calcul et assure une régularisation simplifiée, plus "géométrique".

L'implémentation temps-réel que nous avons réalisée de ce schéma de régularisation permet d'entrevoir une production à plus grande échelle pour de futurs systèmes de vidéo surveillance. Les résultats de la détection de mouvement sur notre corpus sont très encourageants d'autant que nous n'opérons pas sur du matériel informatique spécifique.

En perspective de ce travail sur la détection de mouvement, un modèle de détection et de suppression des ombres détectés serait à envisager. En effet comme nous avons pu le constater le système tel qu'il est actuellement, reste sensible aux fausses détections que sont les ombres portées par exemple. Un module externe, couplé à notre schéma de détection pourra faire l'objet d'une étude plus approfondie.

Une fois l'ensemble des zones de mouvement détectées à l'image, nous avons ensuite posé le problème de la **détection d'un objet d'intérêt** dans une zone de mouvement dans le chapitre 3.

Afin de conserver l'aspect générique du problème de détection, nous avons proposé un système de détection d'un objet appris : un visage humain dans notre cas. Le terme de généricité se réfère au fait de fournir un cadre mathématique le plus général possible afin qu'il puisse être réutilisé dans d'autre contexte que celui de la détection de visages. Détecter un visage suppose donc dans notre cas de choisir un espace de représentation "adéquat", dans lequel puisse intervenir un classifieur dit supervisé. Cet espace doit comporter suffisamment de propriétés discrimantes pour permettre la séparation de la classe à apprendre par rapport à l'ensemble des objets à lui confronter. Nous nous sommes plus particulièrement intéressés à l'étude des classifieurs à base de Support Vector Machines, afin de répondre au problème de la classification en deux classes, "visages" et "non-visages" dans notre cas.

Partant du fait que le visage humain et plus particulièrement l'agencement des valeurs d'intensité d'un visage humain dans l'espace soit suffisamment discriminant pour discerner un visage d'un autre objet présent, nous avons abouti à **un schéma de détection**

multi-résolution. Cette détection s'effectue en parcourant la pyramide multi-résolution multi-échelle d'une image d'origine à l'aide d'une rétine de taille fixe. L'ensemble des valeurs d'intensité contenues dans cette rétine est ensuite étiqueté par un classifieur SVM entraîné suivant les deux classes énoncées précédemment. Cette méthode, couplée avec une validation croisée à différents niveaux de la pyramide de multi-résolution, donne des résultats probants aussi bien sur des images de résolution différente que sur des images contenant des visages de taille différente. Ces résultats ont été obtenus à la fois sur des vidéos de type "broadcast" que sur des vidéos de surveillance.

Toutefois le problème majeur des méthodes par apprentissage -et notamment des SVM- provient du fait que le classifieur soit tributaire de la pose de la forme apprise. Ayant appris l'apparence globale des valeurs d'intensité d'un visage humain pris de face, le classifieur SVM perd tout son pouvoir inductif sur des images où la pose est différente de celle apprise. Plus le visage est semblable aux visages appris plus il a de "chance" d'être reconnu.

Cependant comme nous avons pu le constater, lorsque le classifieur SVM est entraîné sur de grandes bases de données de visages pris de face par exemple, les tests tendent à prouver que le classifieur SVM reste tout de même robuste suivant une certaine variation du visage ne concernant qu'un axe à la fois. Malheureusement dans le cadre de la vidéo surveillance, il est rare que le visage se présente véritablement de face vis-à-vis de la caméra. Pour répondre à ce type de problème, nous avons donc mis en place une méthode intelligente de détection par coopération de deux classifieurs : un premier classifieur par SVM détecte les visages humains alors qu'un second classifieur par mélange de lois Gaussiennes apprend au fur et à mesure la couleur de peau des visages détectés. La méthode proposée par Anthony Don, dite par "bootstrapping", permet ainsi de supprimer les fausses détections des deux classifieurs tout en conservant les vraies. Cette méthode permet par itérations successives d'approximer au mieux la couleur de peau et de pallier au problème de la pose du visage. Cette collaboration aboutit finalement à une augmentation sensible des taux de rappel et de précision des visages à détecter.

Un des points que nous avons abordé en toute fin de ce Chapitre 3 et qui reste dans la perspective de ce travail est l'utilisation de vecteurs de support dits "scalables". En effet, plutôt que d'échantillonner l'image originale afin de créer une pyramide multi-résolution multi-échelle de détection, il serait intéressant d'étudier la possibilité de classifier les valeurs d'intensité retournées par une rétine de taille différente. Ce système nécessiterait donc la mise en place d'un classifieur SVM "scalable" qui permettrait la classification avec des vecteurs de support à des échelles différentes.

Un autre point à étudier serait la possibilité de proposer un classifieur SVM en cascade inspiré de celui proposé par Viola et Jones. En effet, dans une image, la très grande majorité des informations à classifier appartienne à la classe des "non-visages". Le modèle de Viola et Jones propose donc une optimisation en utilisant une cascade de classifieurs de complexité croissante. Ainsi des classifieurs de faible complexité élimineraient rapidement les rétines de non-visages par exemple alors que celles contenant potentiellement un visage sont étudiées plus finement grâce à une classification plus entraînée. Cette approche permet ainsi de

sensiblement réduire les temps de classification. Cette optimisation pourrait être rendu possible, dans la perspective de ce travail, par différents classifieurs SVM entraînés avec plus ou moins d'échantillons par exemple.

Une fois l'objet d'intérêt détecté, se pose naturellement la question du **suivi de cet objet** au cours du temps. Plutôt que de suivre cet objet détecté uniquement sur le principe de conservation de l'intensité lumineuse, nous avons préféré une méthode dans laquelle l'objet d'intérêt est suivi parce qu'il s'agit réellement de l'objet que l'on désire suivre. Cette dernière remarque implique donc de connaître la nature de l'objet et de profiter de la connaissance apprise de cet objet pendant l'apprentissage du détecteur par exemple pour en estimer son mouvement. A partir des mêmes outils mathématiques, les Machines à Vecteurs de Support, nous avons formulé le problème de suivi et proposé une solution pour l'**estimation du modèle affine complet de premier ordre avec les SVM**. Nos travaux se dissocient des travaux précédents notamment développés par Avidan, qui ne considère qu'un modèle translationnel de mouvement limité.

Le modèle adopté permet ainsi de suivre des visages pris de face sous différentes contraintes pour l'ensemble des vidéos de surveillance. Les résultats que nous avons obtenus sont tout à fait probants et démontrent la pertinence d'une telle approche. Néanmoins si l'objet change d'apparence (forte variation de la pose dans notre cas) le détecteur de mouvement permet de maintenir la localisation jusqu'à la détection suivante.

Du point de vue des temps de calcul, le système est trop complexe pour être pleinement exploitable sur un système embarqué. Le code du calcul de la fonction d'énergie à maximiser peut être simplifié en sauvegardant notamment l'ensemble des calculs intermédiaires susceptibles d'être réutilisés. Le code de l'interpolation bi-linéaire reste encore à optimiser. Une solution en passant par des instructions processeur MMX pourrait être envisagée.

Enfin dans un dernier temps nous nous sommes intéressés à l'**optimisation du classifieur SVM par réduction de l'ensemble des vecteurs de support** retenus après la phase apprentissage. A partir des travaux de Downs, nous avons développé une méthode de réduction du nombre de vecteurs de support par élimination des vecteurs linéairement dépendants pour des classifieurs entraînés avec des noyaux polynomiaux de degré d. Appliquée au problème de la détection puis de suivi de visages humains, cette simplification permet de réduire sensiblement les temps de calcul tout en conservant les taux de rappel et de précision obtenus sans cette optimisation. En complément de ces travaux, nous avons proposé un schéma de ré-entraînement itératif dans le but de conserver le nombre "minimal" de vecteurs de support. Ce schéma basé sur un ré-apprentissage, à l'instant t, des vecteurs de support obtenus à l'instant $t-1$, permet de réduire les effets des optimisations prévues lors du développement des implémentations SVM et notamment de la résolution du problème quadratique.

En perspective de ce travail sur la réduction de l'ensemble des vecteurs, un étude sur la comparaison entre la réduction que nous proposons avec l'analyse en composantes principales pourrait être mené. L'idée ici est alors différente : simplifier le processus de classification non pas en réduisant le nombre mais la dimension de l'espace de représentation.

Ainsi dans le cadre de ce travail de thèse, nous avons apporté une solution complète au problème de la vidéo surveillance intelligente avec comme objectif la détection d'objets connus. Tout le système tel qu'il a été développé tout au long de ce mémoire peut être déployé dans un contexte applicatif et industriel réel.

Bibliographie

[ABHV01] H. Siegelmann A. Ben-Hur, D. Horn and V. Vapnik. Support vector clustering. *Journal of Machine Learning Research*, 2(1) :125–137(13), 2001.

[AC99] J.K. Aggarwal and Q. Cai. Human motion analysis : a review. *Computer Vision and Image Understanding*, 73(3) :364–356, March 1999.

[ADG01] N. Freitas A. Doucet and N. Gordon. *An Introduction to Sequential Monte Carlo Methods. In Sequential Monte Carlo Methods in Practice.* 2001.

[AED03] Ramani Duraiswami A.M. Elgammal and Larry S. Davis. Efficient kernel density estimation using the fast gauss transform with applications to color modeling and tracking. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 25(11) :1499–1504, 2003.

[AGJ98] Y. Amit, D. Geman, and B. Jedynak. Efficient focusing and face detection. *In H. Wechsler and J. Phillips, Face Recognition : From Theory to Applications, NATO ASI Series F. Springer-Verlag, Berlin, 1998.*, 1998.

[AK93] T. Aach and A. Kaup. Statistical model-based change detection in moving video. *Signal Processing*, 31 :165–180, 1993.

[AKNN92] T. Agui, Y. Kokubo, H. Nagashashi, and T. Nagao. Extraction of facerecognition from monochromatic photographs using neural networks. *Second International Conference on Automation, Robotics, and Computer Vision*, 1 :1–18, 1992.

[AMGC02] S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for on-line non-linear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2) :174–188, February 2002.

[APS94] B. Moghaddam A. Pentland and T. Starner. View based and modular eigenspaces for face recognition. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 34–58, 1994.

[Avi04] S. Avidan. Support vector tracking. *IEEE Transaction on Pattern Analysis Machine Intelligence*, 26(8) :1064–1072, 2004.

[BA89] D. Blostein and N. Ahuja. A multiscale region detector. *Computer Vision Graphics and Image Processing*, 45(1) :22–41, 1989.

[BA93] M. J. Black and P. Anandan. A framework for the robust estimation of optical flow. In *International Conference on Computer Vision*, May 1993.

[BAHH92] J.R. Bergen, P. Anandan, Keith J. Hanna, and R. Hingorani. Hierarchical model-based motion estimation. In *ECCV '92 : Proceedings of the Second European Conference on Computer Vision*, pages 237–252, London, UK, 1992. Springer-Verlag.

[BBV92] I.M. Guyon B.E. Boser and V.N. Vapnik. A training algorithm for optimal margin classifier. *Proc. Fifth Annual Workshp on Computational Learning Theory*, pages 144–152, 1992.

[BC94] G. Burel and D. Carel. Detection and localization of faces on digital images. *Pattern Recognition Letters*, 15(10) :963–967, October 1994.

[BCZ93] A. Blake, R. Curwen, and A. Zisserman. Affine-invariant contour tracking with automatic control of spatiotemporal scale. In *Proccedings of the International Conference on Computer Vision*, pages 66–75, 1993.

[Ben93] Y. Bengio. A connectionist approach to speech recognition. *International Journal on Pattern Recognition and Artificial Intelligenec*, 7, 1993.

[Ber99] N. Bergman. *Recursive Bayesian Estimation Navigation and Tracking Applications*. PhD thesis, Linkping University, 1999.

[Bes74] J. Besag. Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society*, pages 36 :192–236 Series B, 1974.

[BH67] G.H. Ball and D.J. Hall. A clustering technique for summarizing multivariate data. In *Behavioral Science, Volume 12*, pages 153–155, 1967.

[BH94] A. Baumberg and D. Hogg. Learning flexible models from image sequences. In *ECCV '94 : Proceedings of the third European conference on Computer vision (vol. 1)*, pages 299–308, Secaucus, NJ, USA, 1994. Springer-Verlag New York, Inc.

[Bha43] A. Bhattacharyya. On a measure of divergence between two statistical populations defined by their probability distributions. *Bulletin of the Calcutta Mathematics Society*, 35 :99–110, 1943.

[BI94] A. Blake and M. Isard. 3d position, attitude and shape input using video tracking of hands and lips. In *SIGGRAPH '94 : Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pages 185–192, New York, USA, 1994. ACM Press.

[BID+99] A.F. Bobick, S. Intille, J.W. Davis, Freedom Baird, Claudio S. Pinhanez, Lee W. Campbell, Yuri A. Ivanov, Arjan Schütte, and Andrew Wilson. The KidsRoom : A Perceptually-Based Interactive and Immersive Story Environment. *Presence : Teleoperators and Virtual Environments*, 8(4) :367–391, August 1999.

[BID+00] A. F. Bobick, S.S. Intille, J. W. Davis, F. Baird, C.S. Pinhanez, L.W. Campbell, Y.A. Ivanov, A. Schütte, and A. Wilson. Perceptual user interfaces : the KidsRoom. *Communications of the Association for Computing Machinery*, 43(3) :60–61, March 2000.

[Big96] J.P. Bigus. *Data Mining with Neural Networks : Solving Business Problems from Application Development to Decision Support.* McGraw-Hill, ISBN 0-07-005779-6, 1996.

[Bis95] C.M. Bishop. *Neural Networks for Pattern Recognition.* Oxford University Press, ISBN 0-19-853849-9, 1995.

[BJ98] M. J. Black and A.D. Jepson. A probabilistic framework for matching temporal trajectories : Condensation-based recognition of gestures and expressions. In *ECCV '98 : Proceedings of the 5th European Conference on Computer Vision-Volume I*, pages 909–924, London, UK, 1998. Springer-Verlag.

[BL93] P. Bouthémy and P. Lalande. Recovery of moving object masks in an image sequence using local spatiotemporal contextual information. *Optical Engineering*, 32(6) :1205–1212, 1993.

[BO95] C. Bregler and S. M. Omohundro. Nonlinear manifold learning for visual speech recognition. In *ICCV '95 : Proceedings of the Fifth International Conference on Computer Vision*, page 494, Washington, DC, USA, 1995. IEEE Computer Society.

[BP02] L. Bruzzone and D. F. Prieto. An adaptive semiparametric and context-based approach to unsupervised change detection in multitemporal remote-sensing images. *IEEE Transactions on Image Processing*, 11(4) :452–466, April 2002.

[Bre97a] C. Bregler. Learning and recognising human dynamics in video sequences. *In Computer Vision and Pattern Recognition*, 1997.

[Bre97b] F. Bremond. *Environnement de résolution de problèmes pour l'interprétation de séquences d'images.* PhD thesis, Université de Nice-Sophia Antipolis, 1997.

[BSB+96] V. Blanz, B. Scholkopf, H.H. Bulthoff, C. Burges, V. Vapnik, and T. Vetter. Comparison of view-based object recognition algorithms using realistic 3d models. In *ICANN*, pages 251–256, 1996.

[BSF88] Y. Bar-Shalom and T. E. Fortmann. *Tracking and Data Association*, volume 179 of *Mathematics in Science and Engineering.* Academic Press, 1988.

[Bur96] C. Burges. Simplified support vector decision rules. *Proceedings of the 13th International Conference on Machine Learning*, 2(1) :71–77(7), 1996.

[Bur98] C.J.C. Burges. A tutorial on support vector machine for pattern recognition. *Data-Mining and Knowledge Discovery*, 2 :121–167, 1998.

[CA96] Q. Cai and J. K. Aggarwal. Tracking human motion using multiple cameras. In *International Conference on pattern Recognition*, volume 1, pages 68–73, Vienna, Autria, 1996.

[CBBS94] J.L. Crowley, J. Bedrune, M. Bekker, and M. Schneider. Integration and control of reactive visual processes. In *ECCV '94 : Proceedings of the Third European Conference-Volume II on Computer Vision*, pages 47–58, London, UK, 1994. Springer-Verlag.

[CBP05a] L. Carminati and J. Benois-Pineau. Détection des zones de mouvement et régularisation : Application à la télésurveillance. In *GRETSI'05*, Louvain la Neuve (Belgium), September 2005.

[CBP05b] L. Carminati and J. Benois-Pineau. Gaussian mixture classification for moving object detection in video surveillance environment. In *IEEE International Conference on Image Processing (ICIP) 2005*, Italie (Gènes), 2005.

[CBP05c] L. Carminati and J. Benois-Pineau. Support vector tracking of human faces with affine motion models. In *Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS) 2005*, Suisse (Montreux), 2005.

[CBPG03] L. Carminati, J. Benois-Pineau, and M. Gelgon. Human detection and tracking fro video surveillance applications in low density environment. *SPIE VCIP'2003 SPIE 0277 -786X*, 5150 :51–60, 2003.

[CBPJ05] L. Carminati, J. Benois-Pineau, and C. Jennewein. Optimisation des performances des svm appliqués au problème de la détection de visages. In *CORESA 2005*, Rennes (France), 2005.

[CBPJ06] L. Carminati, J. Benois-Pineau, and C. Jennewein. Knowledge-based supervised learning methods in a classical problem of video object tracking. In *IEEE International Conference on Image Processing (ICIP) 2006*, Atlanta (USA), 2006.

[CG00] T.H. Chang and S. Gong. Tracking multiple people under occlusion using multiple cameras. *British Machine Vision Conference*, September 2000.

[CG01] T.H. Chang and S. Gong. Tracking multiple people with a multi-camera system. *IEEE Workshop on Multi-Object Tracking*, pages 19–28, July 2001.

[CK98] J.P. Costeira and T. Kanade. A multibody factorization method for independently moving objects. *International Journal on Computer Vision*, 29(3) :159–179, 1998.

[CL92] D. Chetverikov and A. Lerch. Multiresolution face detection. pages 130–140, Berlin, 1992. Buckow (Eds. R. Klette, W. G. Kropatsch.).

[CLD98] A. Caplier, F. Luthon, and C. Dumontier. Real-time implementations of an mrf-based motion detection algorithm. *Journal of Real Time Imaging, Special Issue on Real-Time Motion Analysis*, 4(1) :41–54, February 1998.

[CM03] A. Cornuéjols and L. Miclet. *Apprentissage artificiel, Concepts et algorithmes*. Eyrolles, ISBN :2-212-11162-2, France, 2003.

[CMA95] Q. Cai, A. Mitiche, and J. K. Aggarwal. Tracking human motion in an indoor environment. In *Proceedings of the International Conference on Image Processing*, volume 1, pages 215–218, Washington, D.C., 1995.

[CN98] D. Chai and K. N. Ngan. Locating facial region of a head-and-shoulders color image. In *FG '98 : Proceedings of the 3rd. International Conference on Face & Gesture Recognition*, page 124, Washington, DC, USA, 1998. IEEE Computer Society.

[CPP98] M. Oren C. Papageorgiou and T. Poggio. A general framework for object detection. *International Conference on Computer Vision*, 1998.

[CR87] P. Chou and R. Raman. On relaxation algorithms based on markov random fields. *Technical Report N.212 Computer Science Dpt, University of Rochester*, 1987.

[CR00] D. Comaniciu and V. Ramesh. Robust detection and tracking of human faces with an active camera. In *VS '00 : Proceedings of the Third IEEE International Workshop on Visual Surveillance (VS'2000)*, page 11, Washington, DC, USA, 2000. IEEE Computer Society.

[Cri93] J. Crisman. Color region tracking for vehicle guidance. pages 107–120, 1993.

[CRM00] D. Comaniciu, V. Ramesh, and P. Meer. Real-time tracking of non-rigid objects using mean shift. In *Proceedings on the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 142–149, 2000.

[CTB92] I. Craw, D. Tock, and A. Bennett. Finding face features. In *ECCV '92 : Proceedings of the Second European Conference on Computer Vision*, pages 92–96, London, UK, 1992. Springer-Verlag.

[CW96] J.B. Collins and C.E. Woodcock. An assessment of several linear change detection techniques for mapping forest mortality using multitemporal landsat tm data. *Remote Sensing of Environment*, 56 :66–77, 1996.

[CZ05] A. Cavallaro and F. Ziliani. Characterisation of tracking performance. *Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS)*, 2005.

[DBP01] M. Durik and J. Benois-Pineau. Robust motion characterisation for video indexing based on MPEG2 optical flow. *Content Based Multimedia Indexing*, September 2001.

[DC96] D. Yuan D. and C.D. Elvidge C.D. Comparison of relative radiometric normalization techniques. *ISPRS J. Remote Sens*, 51 :117–126, 1996.

[DCBP05] A. Don, L. Carminati, and J. Benois-Pineau. Detection of visual dialog scenes in video content based on structural and semantic features. In *International Workshop on Content-based Multimedia Indexing (CBMI) 2005*, Létonie (Tampere), 2005.

[DE87] H. Derin and H. Elliott. Modeling and segmentation of noisy and textured images using Gibbs random fields. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 9 :39–55, 1987.

[DeJ86] G. DeJong. *An Approach to Learning from Observation*, pages 571 – 590. Morgan Kaufmann, Los Altos, CA, 1986.

[DF90] R. Deriche and O. Faugeras. Tracking line segments. *Image Vision Computer*, 8(4) :261–270, 1990.

[DGA00] A. Doucet, S. Godsill, and C. Andrieu. On sequential monte carlo sampling methods for bayesian filtering. *Statistics and Computing*, 10(3) :197–208, 2000.

[DGM01] T. Downs, K. Gates, and A. Masters. Exact simplification of support vector solutions. *Journal of Machine Learning Research*, 2(1) :293–297(5), 2001.

[DHS00] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley-Interscience Publication, 2000.

[Dic93] E. D. Dickmanns. Expectation-based dynamic scene understanding. pages 303–335, 1993.

[Dic97] E. D. Dickmanns. Vehicles capable of dynamic vision. In *15th International Joint Confernece on Artificial Intelligence (IJCAI-97)*, Nagoya, Japan, 1997.

[Die91] N. Diehl. Object-oriented motion estimation and segmentation in image sequences. *Signal Processing : Image Communication*, 3(1) :23–56, 1991.

[DKM94] J. Weber D. Koller and J. Malik. Robust multiple car tracking with occlusion reasoning. *Third European Conference on Computer Vision, Stockholm*, pages 189–196, 1994.

[DLR77] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1) :1–38, 1977.

[DN95] V. Dai and Y. Nakano. Extraction for facial images from complex background using color information and sgld matrices. *In Proceedings of First International Workshop Automatic Face and Gesture Recognition*, pages 238–242, 1995.

[DN96] Y. Dai and Y. Nakano. Face-texture model-based on SGLD and its application in face detection in a color scene. *Pattern Recognition*, 29 :1007–1017, June 1996.

[DRM03] D.Comaniciu, V. Ramesh, and P. Meer. Kernel-based object tracking. *IEEE Transaction on Pattern Analysis Machine Intelligence*, 25(5) :564–575, 2003.

[DSG90] A. Djouadi, J. Snorrason, and F. D. Garber. The quality of training sample estimates of the bhattacharyya coefficient. *IEEE Transaction on Pattern Analysis and Machince. Intelligence*, 12(1) :92–97, 1990.

[EHD99] A. Elgammal, D. Harwood, and L. Davis. Non-parametric model for background substraction. *IEEE ICCV 1999 Frame Rate WorkShop*, 1999.

[Ell02] T. Ellis. Performance metrics and methods for tracking in surveillance. In *3rd IEEE International Workshop on Performance Evaluation of Tracking and Surveillance PETS'2002*, Copenhagen, Denmark, June 2002.

[Fau94] L. Fausett. *Fundamentals of Neural Networks : Architectures, Algorithms, and Applications*. Englewood Cliffs, NJ : Prentice Hall, 1994. ISBN 0-13-334186-0.

[FCF03] C.Y. Fang, S.W. Chen, and C.S. Fuh. Automatic change detection of driving of driving environments in a vision-based driver assistance system. *IEEE Transaction on Neural Networkd*, 14 :646–657, May 2003.

[FCH96] J. H. Fernyhough, A. G. Cohn, and D. Hogg. Generation of semantic regions from image sequences. In *ECCV '96 : Proceedings of the 4th European Conference on Computer Vision-Volume II*, pages 475–484, London, UK, 1996. Springer-Verlag.

[FL98] H. Fujiyoshi and A. Lipton. Real-time human motion analysis by image skeletonization. *Proceedings of the IEEE Workshop on Applications of Computer Vison*, 1998.

[FR97] N. Friedman and S. Russel. *Image Segmentation in Video Sequences : A Probabilistic Approach.* Morgan Kaufman Publishers Inc., 1997.

[FRB94] N.J. Ferrier, S.M. Rowe, and A. Blake. Real-time traffic monitoring. In *Workshop on Applications of Computer Vision (WACV94)*, pages 81–88, 1994.

[Fri95] J.H. Friedman. Introduction to computational learning and statistical prediction, tutorial. *ICANN'95*, 1995.

[FS95] Y. Freund and R.E Schapire. *A decision-theoric generalization of on-line learning and an application to boosting.* Springer-Verlag, 1995.

[Fuk90] Keinosuke Fukunaga. *Introduction to statistical pattern recognition (2nd ed.).* Academic Press Professional, Inc., San Diego, CA, USA, 1990.

[FV99] F. Fleuret and J.M. Vézien. Face detection from video stream using decision trees. *INRIA Technical Report*, pages 1–10, 1999.

[GBPM97] C. Kenyon G. Brightwell and H. Paugam-Moisy. Multilayer neural network : one or two hidden layers ? *Advances in Neural Information Processing Systems 9, Proc. NIPS 96*, pages 148–154, 1997.

[GC96] A. Gee and R. Cipolla. Fast visual tracking by temporal consensus. *Image and Vision Computing, 14(2)*, pages 105–114, February 1996.

[GD02] C. Garcia and M. Delakis. Convolutional face finder : A neural architecture for fast and robust face detection. *IEEE-IAPR International Conference on Pattern Recognition*, 26 :1408–1423, 2002.

[Gel98] M. Gelgon. *Segmentation spatio-temporelle et suivi dans une séquence d'images : application à la structuration et à l'indexation de vidéo.* PhD thesis, Université de Rennes 1/IRISA, France, 1998.

[Gen92] Donald B. Gennery. Visual tracking of known three-dimensional objects. *International Journal on Computer Vision*, 7(3) :243–270, 1992.

[GG84] S. Geman and D. Geman. Stochastic relaxation, gibbs distributions and the bayesian restoration of images. *IEEE Pattern Analysis and Machine Intelligence*, 6, 1984.

[Gib96] G.J. Gibson. Exact classification with two layer neural nets. *J. Computer and System science*, 52(2) :349–356, 1996.

[GKK05] J. Gao, A. Kosaka, and A. C. Kak. A multi-kalman filtering approach for video tracking of human-delineated objects in cluttered environments. *Computer Vision and Image Understanding*, 99(1) :1–57, 2005.

[Gov96] V. Govindaraju. Locating humans faces in photographs. *International Journal Of Computer Vision*, pages 129–146, 1996.

[GSRL98] W.E.L. Grimson, C. Stauffer, R. Romano, and L.Lee. Using adaptive tracking to classify and monitor activities in a site. *IEEE Conference on Computer Vision and Pattern Recognition 1998*, pages 22–29, 1998.

[Hal98] P. H. Halpin. Multi-period landscape change vector analysis of the north carolina piedmont. In *International Association of Landscape Ecologists (IALE) Proceedings*, March 1998.

[Har93] C. Harris. Tracking with rigid models. *MIT Press, Cambridge, MA, USA*, pages 59–73, 1993.

[Har95] Elizabeth L. Hartshorn. *Mapping the Market's Future with Neural Networks.* Research, September 1995.

[Har98] I. Haritaoglu. *A Real Time System for Detection and Tracking of People and Recognizing Their Activities.* PhD thesis, University of Maryland at College Park, 1998.

[HC71] J. M. Hammersley and P. Clifford. Markov fields on finite graphs and lattices. *Unpublished*, 1971.

[Heb49] Donald O. Hebb. *The Organization of Behavior.* John Wiley, New York, 1949.

[Hei88] Fabrice Heitz. *Restauration de la radiographie d'un tableau dissimulé par une composition postérieure.* PhD thesis, Ecole Nationale Supérieure des Télécommunications, Thèse de Doctorat, Juillet 1988.

[HFK98] A. Lipton H. Fujiyoshi and T. Kanade. Real-time human motion analysis by image skeletonization. In *Proc. of the Workshop on Application of Computer Vision*, October 1998.

[HKR93] D. P. Huttenlocher, G. A. Klanderman, and W. A. Rucklidge. Comparing images using the hausdorff distance. *IEEE Transactions on Pattern Analysis Machine Intelligence*, 15(9) :850–863, 1993.

[HM00] R. Hammoud and R. Mohr. Gaussian mixture densities for indexing of localized objects in a video sequence. In *INRIA Rapport de Recherche ISSN 0249-6399*, volume 3905, 2000.

[HNR84] Y.Z. Hsu, H.-H. Nagel, and G. Rekers. New likelihood test methods for change detection in image sequences. *Computer Vision, Graphics and Image Processing*, 26 :73–106, 1984.

[Hog83] D. Hogg. Model-based vision : A program to see a walking person. *Image and Vision Computing*, 1(1) :5–20, 1983.

[HS81] B. K. P. Horn and B. G. Schunk. Determining optical flow. *Artificial Intelligence*, 17 :185–203, 1981.

[HS97] Wang H. and Chang S.F. A highly efficient system for automatic face region detection in mpeg video. *IEEE Transactions on Circuits and Systems for Video Technology*, 7(4) :615–628, 1997.

[HSW92] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. In Halber White, editor, *Artificial Neural Networks : Approximation and Learning Theory*, pages 12–28. Blackwell, Oxford, UK, 1992.

[IB98] M. Isard and A. Blake. Condensation conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29(1) :5–28, 1998.

[IL98] D. Harwood I.Haritaoglu and L.Davis. W4 : Who, when, where, what : A real time system for detecting and tracking people. *Third International Conference on Automatic Face and Gesture*, April 1998.

[Ima06] Image Magick. ImageMagick, version 6.2.6, est une suite logicielle libre permettant la manipulation, l'édition et la composition d'images, 2006.

[IU03] M. Inoue and N. Ueda. Exploitation of unlabeled sequences in hidden markov models. *IEE Transaction on on Pattern Analysis and Machine Intelligence*, 25 :1570–1581, 2003.

[Jaf06] G. Jaffré. *Costume : A New Feature for Automatic Video Content Indexing.* PhD thesis, IRIT Université Paul Sabatier Toulouse, 2006.

[Jai81] R Jain. Dynamic scene analysis using pixel–based processes. *Computer*, 14(8) :12–18, 1981.

[JBA00] S. Jehan-Besson, M. Barlaud, and G. Aubert. Detection and tracking of moving objects using a new level set based method. *Proceedings of the International Conference on Pattern Recognition*, pages 1112–1117, 2000.

[JKF01] Oliver Jesorsky, Klaus J. Kirchberg, and Robert Frischholz. Robust face detection using the hausdorff distance. In *AVBPA '01 : Proceedings of the Third International Conference on Audio- and Video-Based Biometric Person Authentication*, pages 90–95, London, UK, 2001. Springer-Verlag.

[JMA79] R Jain, W N Martin, and J K Aggarwal. Segmentation through the detection of changes due to motion. *Computer Graphics and Image Processing*, 11 :13–34, 1979.

[Joa98] T. Joachims. Text categorization with support vector machines : learning with many relevant features. In *Proceedings of ECML-98, 10th European Conference on Machine Learning*, pages 137–142. Springer Verlag, Heidelberg, DE, 1998.

[JS02] O. Javed and M. Shah. Tracking and object classification for automated surveillance. *The Seventh European Conference on Computer Vision*, May 2002.

[Kal60] R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the AMSE, Part D, Journal of Basic Engineering*, 82 :35–45, 1960.

[KB89] K.P. Karmann and A. Brandt. Detection and tracking of moving objects by adaptative background extraction. *Proceedings of the 6th Scandinavian Conference on Image Analysis*, pages 1051–1058, 1989.

[KB01] P. Kaewtrakulpong and R. Bowden. An improved adaptative background mixture model for real-time tracking with shadow detection. *Proc. 2nd European Workshop on Advanced Video Based Surveillance Systems, AVBS01*, 2001.

[KBP05] P. Kraemer and J. Benois-Pineau. Camera motion detection in the rough indexing paradigm. In *TREC Video Retrieval Evaluation Online Proceedings, TRECVID'05*, November 2005.

[KCL+98] T. Kanade, R. Collins, A. Lipton, P. Burt, and L. Wixson. Advances in cooperatives multi-sensor video surveillance. In *Proceedings of DARPA Image Understanding Workshop*, volume 1, pages 3–24, 1998.

[KdVL94] Y.H. Kwon and N. da Vitoria Lobo. Face detection using templates. In *Proceedings of the International Conference on Pattern Recogintion (ICPR)*, pages A :764–767, 1994.

[KFSW02] C. Komninakis, C. Fragouli, A. H. Sayed, and R. D. Wesel. Multi-input multi-output fading channel tracking and equalization using kalman estimation. *IEEE Transactions on Signal Processing,*, 50, :1065–1076, 2002.

[KH04] R. Koggalage and S. Halgamuge. Reducing the number of training samples for fast support vector machine classification. *Neural Information Processing – Letters and Reviews*, 2(3) :57–65(9), 2004.

[Kil92] M. Kilger. A shadow handler in a video-based real-time traffic monitoring system. *IEEE Workshop Applications of Computer Vision*, pages 11–18, 1992.

[Koh82a] Teuvo Kohonen. Analysis of a simple self-organizing process. *Biological Cybernetics*, 44(2) :135–140, 1982.

[Koh82b] Teuvo Kohonen. Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43(1) :59–69, 1982.

[Koh90] Teuvo Kohonen. The self-organizing map. *Proceedings of the IEEE*, 78(9) :1464–1480, 1990.

[KP97] C. Kotropoulos and I. Pitas. Rule-based face detection in frontal views. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'97)*, volume IV, pages 2537–2540, Munich, Germany, 20-24 April 1997.

[KPC01] J. Kharroubi, D. Petrovska, and G. Chollet. Text-independent speaker verification using support vector machines. In *2001 : A Speaker Odyssey*, Crete, Greece, 2001.

[KT51] H. W. Kuhn and A. W. Tucker. Proceedings of the 2nd berkely symposium on mathematical statistics and probabilitics. In *Non Linear Programming, Berkeley, University of California Press*, pages 481–492, 1951.

[KvD75] J. J. Koenderink and A. J van Doorn. Invariant properties of the motion parallax field due to the movement of rigid bodies relative to an observer. *Optica Acta*, 22(9) :773–791, 1975.

[KvD91] Jan J. Koenderink and Andrea J. van Doorn. Affine structure from motion. *Journal of the Optical Society of America*, 8 :377–385, 1991.

[KWH+94] D. Koller, J. Weber, T. Huang, J. Malik, G. Ogasawara, and B. Rao S. Russel. Towards robust automatic traffic scene analysis in real time. In *IEEE CDC 1994*, pages Cat. No.94CH3460–3 Part vol.4, 1994.

[L. 96] L. Devroye and L. Györfi and G. Lugosi. A probabilistic theory of pattern recognition. *Applications of Mathematic*, 31, 1996.

[Lal90] Patrick Lalande. *Détection du mouvement dans une séquence d'image selon une approche markovienne. Application à la robotique sous-marine.* PhD thesis, Université de Rennes I, Thèse de Doctorat, 1990.

[Lan99] Christiane Landry. Correction interactive de couleur par association. M.sc. thesis, Département d'Informatique et Recherche Opérationnelle, Université de Montréal, April 1999.

[LB90] P. Lalande and P. Bouthémy. A statistical approach to the detection and tracking of moving objects in an image sequence. *5th European Signal Processing Conference EUSIPCO 90*, 1990.

[LCDTH00] F. Kanade L. Collins, T. Duggins, E. Tolliver, and Hasegawa. A system for video surveillance and monitoring : VSAM final report. Technical report, Technical report CMU-RI-TR-00-12, Robotics Institute, Carnegie Mellon University, May 2000.

[LCK99] A.J. Lipton L. Collins and T. Kanade. A system for video surveillance and monitoring. *American Nuclear Society (ANS) Eighth International Topical Meeting on Robotics and Remote Systems*, April 1999.

[LD01] S. C. Lee and D.V.Heinbuch. Training a neural network-based intrusion detector to recognize novel attacks. *IEEE Transactions on Systems, Man, and Cybernetics - Part A : Systems and Humans*, 31(4) :294–299, July 2001.

[LDB98] S. Fejes D. Harwood Y. Yacoob I. Haritaoglu L. Davis and M. Black. Visual surveillance of human activity. *Asian Conference on Computer Vision*, January 1998.

[LeC86] Y. LeCun. Learning processes in an asymmetric threshold network. In E. Bienenstock, F. Fogelman-Soulié, and G. Weisbuch, editors, *Disordered systems and biological organization*, pages 233–240, Les Houches, France, 1986. Springer-Verlag.

[LeC88] Y. LeCun. A theoretical framework for back-propagation. In D. Touretzky, G. Hinton, and T. Sejnowski, editors, *Proceedings of the 1988 Connectionist Models Summer School*, pages 21–28, CMU, Pittsburgh, Pa, 1988. Morgan Kaufmann.

[Let93] J.M. Letang. *Intégration temporelle et régularisation statistique appliqués à la détection d'objets mobiles dans une séquence d'images.* PhD thesis, Institut National Polytechniques de Grenoble, Thèse de Doctorat, 1993.

[LI04] T.Y. Lui and E. Izquierdo. Automatic detection of human faces in natural scene images by use of skin colour and edge distribution. In *Proceedings of the 5th European workshop on Image Analysis for Multimedia Interactive Services*, page 32, Portugal, April 2004.

[Lié00] Marc Liévin. Analyse entropico-logarithmique de séquences vidéo couleur. application à la segmentation markovienne et au suivi de visages parlants. Thèse de doctorat, Institut National Polytechnique de Grenoble (INPG), 2000.

[LP00] G.J. Mc Lachlan and D. Peel. *Finite Mixture Models.* Wiley Series In Probability And Statistics, 2000.

[LS01] N. Laurence and B. Schölkopf. Estimating a kernel fisher discriminant in the presence of label noise. In *Proceeding of the 18th International Conference on Machine Learning.* Morgan Kaufman, 2001.

[LWB96] P. Delagnes L. Wu, J. Benois-Pineau and D. Barba. Spatiotemporal segmentation of image sequences for object-oriented low bit-rate image coding. *Signal Processing : Image Communication*, 8(6) :513–544, 1996.

[LXK90] E. Oja L. Xu and P. Kultanen. A new curve detection method : Randomized hough transform. *Pattern Recognition Letters*, 11(5), 1990.

[Mak96] A. Makarov. Comparison of background extraction based intrusion detection algorithms. In *ICIP96*, page 16P3, 1996.

[Man69] O. L. Mangasarian. *Nonlinear Programming.* McGraw-Hill, New York, 1969.

[Mar90] D. Marr. *Vision.* Freeman and Co., New York, 1990.

[MB87] D.W Murray and B.F. Buxton. Scene segmentation from visual motion using global optimization. *IEEE Transaction on on Pattern Analysis and Machine Intelligence*, 9, 1987.

[MB94] Fernand G. Meyer and Patrick Bouthémy. Region based tracking using affine motion models in long image sequences. *Computer Vision, Graphics and Image Processing : Image Understanding*, 60(2) :119–140, September 1994.

[MB03] D. Miller and J. Browning. A mixture model and EM-based algorithm for class discovery, robust classification, and outlier rejection in mixed labeled/unlabeled data sets. *IEEE Transaction on on Pattern Analysis and Machine Intelligence*, 25 :1468–1483, 2003.

[MBB03] J. Miteran, S. Bouillant, and E. Bourennane. Svm approximation for real-time image segmentation by using an improved hyperrectangles-based method. *Real-Time Imaging*, 9(3) :179–188, 2003.

[MBF91] Walter J. Loick Mark B. Fishman, Dean S. Barr. *Using Neural Nets In Market Analysis.* Technical Analysis of Stocks And Commodities, April 1991.

[MBNGR03] J.P. Armspach M. Bosc, F. Heitz, I. Namer, D. Gounot, and L. Rumbach. Automatic change detection in multimodal serial mri : application to multiple sclerosis lesion evolution. *Neuroimages*, 20 :643–656, 2003.

[MBPB02] A. Mahboubi, J. Benois-Pineau, and D. Barba. Tracking of objects in video scenes with time varying content. *EURASIP Journal of Applied Signal Processing (JASP)*, 2002(6) :582–594, June 2002.

[MBPL04] F. Manerba, J. Benois-Pineau, and R. Leonardi. Extraction of foreground objects from MPEG2 video stream in rough indexing framework. In *Proc. EI'2004, Storage and Retrieval Methods and Applications for Multimeida 2004*, San Jose, CA, USA, 2004. SPIE.

[McQ67] J. McQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathmatical Statistics and Probability*, pages 291–287, 1967.

[MDBP97] Pattie Maes, Trevor Darrell, Bruce Blumberg, and Alex Pentland. The alive system : wireless, full-body interaction with autonomous agents. *Multimedia Syst.*, 5(2) :105–112, 1997.

[MDZ97] R. Morris, X. Descombes, and J. Zerubia. Fully bayesian image segmentation : An engineering perspective. pages 54–57, 1997.

[Mec89] A. Mecocci. Moving object recognition and classification in external environments. *Signal Processing*, 18 :183–194, 1989.

[MGR98] S.J. McKenna, S. Gong, and Y. Raja. Modelling facial colour and identity with gaussian mixtures. *Pattern Recognition Letter Vol. 31(12)*, 1998.

[M.I00] M.I.T. Cbcl face database #1. Base de données de visages créée par le M.I.T. Disponible à l'adresse http ://cbcl.mit.edu/cbcl/software-datasets/FaceData2.html, 2000.

[M.I01] M.I.T. SvmFu Version 3. Software developed by the Center for Biological and Computational Learning du M.I.T. http ://five-percent-nation.mit.edu/SvmFu, 2001.

[Mic84] R. S. Michalski. A theory and methodology for inductive learning. *Artificial Intelligence*, 20(2), February 1984.

[Mit97] Tom T. Mitchell. *Machine Learning*. McGraw-Hill ISBN 0-07-042807-7, 1997.

[MLBP05] F. Manerba, R. Leonardi, and J. Benois-Pineau. Real-time extraction of foreground objects in a rough indexing framework. In *WIAMIS'2005*, Montreux, SW, 2005.

[MMR+01] K.-R. Müller, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf. An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks*, 12(2) :181–201, 2001.

[MP43] W.S. McCulloch and W. Pitts. A logical calculus of ideas immanent in neural activity. *Bulletin of Mathematical Biophysics*, 5 :115–133, 1943.

[MP97] Baback Moghaddam and Alex P. Pentland. Probabilistic visual learning for object recognition. *IEEE Transactions on Patterns Analysis and Machine Intelligence*, 19(7) :696–710, July 1997.

[MS95] N.J.B. McFarlane and C.P. Schofield. Segmentation and tracking of piglets in images. *Machine Vision and Applications*, 8 :187–193, 1995.

[MST94] D. Michie, D. J. Spiegelhalter, and C. C. Taylor, editors. *Machine Learning, Neural and Statistical Classification.* Ellis Horwood, 1994.

[MZ92] J. L. Mundy and A. Zisserman, editors. *Geometric invariance in computer vision.* MIT Press, Cambridge, MA, USA, 1992.

[Nag78] H. H. Nagel. Formation of an object concept by analysis of systematic time variations in the optically perceptible environment. *Computer Graphics and Image Processing*, 7(2) :149–194, April 1978.

[NB03] C. J Needham and R. D. Boyle. Performance evaluation metrics and statistics for positional tracker evaluation. In *Computer Vision Systems : Third International Conference, ICVS 2003, Graz, Austria*, pages 278–289. Springer Verlag, April 2003.

[NH98] R. M. Neal and G. E. Hinton. *A New View of the EM Algorithm that Justifies Incremental, Sparse, and Other Variants.* NATO Science Series. Kluwer Academic Publishers, M. I. Jordan, Dordrecht, 1998.

[NKMG02] K. Nummiaro, E. Koller-Meier, and L. Van Gool. A color-based particle filter. In A.E.C. Pece, editor, *First International Workshop on Generative-Model-Based Vision*, volume 2002/01, pages 53–60. Datalogistik Institut, Kobenhavns Universitet, 2002.

[Nov62] A. B. J. Novikoff. On convergence proofs on perceptrons. *Proceedings of the Symposium on the Mathematical Theory of Automata*, 12 :615–622, 1962.

[Now91] Steven J. Nowlan. *Soft Competitive Adaptation : Neural Network Learning Algorithms based on Fitting Statistical Mixtures.* PhD thesis, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, 1991.

[OB94] J.M. Odobez and P. Bouthémy. Detection of multiple moving objects using multiscale markov random fields, with camera compensation. *International Conference on Image Processing (ICIP)*, 2 :257–26, 1994.

[OFG97] E. Osuna, R. Freund, and F. Girosi. Training support vector machines : an application to face detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 130–136, 1997.

[OJS03] K. Shafique O. Javed, Z. Rasheed and M. Shah. Tracking in multiple cameras with disjoint views. *The Ninth IEEE International Conference on Computer Vision (ICCV)*, pages 13–16, October 2003.

[OPBC97] Nuria Oliver, Alex Pentland, François Bérard, and James Crowley. LAFTER : Lips and face real time tracker. Technical Report 396, MIT Media Lab, 1997.

[PBP04] L. Primaux and J. Benois-Pineau. Optimized real time h.264 encoder for videosurveillance applications. In *WIAMIS 2004*, Lisbonne, Portugal, avril 2004.

[PBPKD04] L. Primaux, J. Benois-Pineau, P. Kraemer, and J-P. Domenger. Shot boundary detection in the framework of rough indexing paradigm. In *TRECVID Workshop*, NIST in Gaithersburg, MD, November 2004.

[Pea94] Karl Pearson. Contribution to the mathematical theory of evolution. *Philosophical Transactions of the Royal Society of London*, 185 :71–110, 1894.

[Per87] L.I. Perlovsky. Multiple sensor fusion and neural networks. In *DARPA Neural Network Study*. MIT/Lincoln Laboratory, Lexington, MA, USA, 1987.

[Pla85] T. Plassard. *Extraction des paramètres caractéristiques dans une séquence d'images de particules passant de l'état de brouillard à l'état d'agglomérats. Application à la génération automatique d'alarme en milieu industriel.* PhD thesis, Université de Rennes I, Thèse de Doctorat, 1985.

[PN03] J. Pinel and H. Nicolas. Cast-shadows detection on lambertian surfaces in video sequences. *Proceedings of Visual Communication and Image Processing*, pages 378–384, 2003.

[PQJP05] L. Primaux, J.C. Quillet, and J.Benois-Pineau. Méthodologie et outils logiciel en vue de la compression spatio-temporelle d'images vidéo. Technical report, LaBRI, June 2005.

[PS99] S. Pingal and J. Segen. Performance evaluation of people tracking systems. *IEEE Workshop on Applications of Computer Vision*, pages 33–39, December 1999.

[PUE96] K. Pahlavan, T. Uhlin, and J.O. Eklundh. Dynamic fixation and active perception. *International Journal of Computer Vision*, 17(2) :113–135, February 1996.

[Pér93] Patrick Pérez. *Champs markoviens et analyse multirésolution de l'image : application à l'analyse de mouvemenent.* PhD thesis, Université de Rennes I, Thèse de Doctorat, Juillet 1993.

[QCY95] W. Haiyuan Q. Chen and M. Yachida. Face detection by fuzzy pattern matching. *Proceedings of the Fifth International Conference on Computer Vision*, pages 20–23, June 1995.

[QMS+04] G. M. Quenot, D. Mararu, S.Ayache, M. Charhad, L. Besacier, M. Guironnet, D. Pellerin, J. Gensel, and L. Carminati. Clips lis lsr labri experiments in trec video retrieval 2004. In *TREC Video Retrieval 2004*, 2004.

[QSM98] R. Qian, M. Sezan, and K. Matthews. A robust Real-Time face tracking algorithm. pages 131–135, 1998.

[RAAKR05] R. Radke, S. Andra, O. Al-Kofahi, and B. Roysam. Image change detection algorithms : A systematic survey. *IEEE Transactions on Image Processing*, 14(3) :294–307, 2005.

[RBT99] H.A. Rowley, S. Baluja, and K. Takeo. Neural network-based face detection. *IEEE on Pattern Analysis and Machine Intelligence*, pages 20 :23–38, 1999.

[RG99] T. V. Ravi and K. C. Gowda. An isodata clustering procedure for symbolic objects using a distributed genetic algorithm. *Pattern Recognition Letter Vol. 20*, 1999.

[RHW86] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. In D. E. Rumelhart and J. L. McClelland, editors, *Parallel Distributed Processing*, volume 2, pages 318–362. MIT Press, 1986.

[RI03] P. Rosin and E. Ioannadis. Evaluation of global image thresholding for change detection. *Pattern Recognition Letters*, 24(14) :2345–2356, 2003.

[Ric87] Yann Ricquebourg. *Analyse de mouvements articulés : mesure et suivi 2D ; application à la télésurveillance.* PhD thesis, Université de Rennes 1, Thèse de Doctorat, Janvier 1987.

[Ric99] Kevin Richards. Network based intrusion detection : A review of technology. *Computers & Security*, 18 :671–682, 1999.

[RK95] J. M. Rehg and Takeo Kanade. Model-based tracking of self-occluding articulated objects. In *ICCV*, pages 612–617, 1995.

[RK99] C. Russell and G. Kass. *Assessing the Accuracy of Remotely Sensed Data : Principles and Practices.* Lewis Publishers Inc, Boca Raton, 1999.

[RM96] I. D. Reid and David W. Murray. Active tracking of foveated feature clusters using affine structure. *International Journal of Computer Vision*, 18 :41–60, 1996.

[RMS92] Helge Ritter, Thomas Martinez, and Klaus Schulten. *Neural Computation and Self-Organizing Maps.* Addison-Wesley Publishing Company, New York, 1992.

[Ros59] F. Rosenblatt. Two theorems of statistical separability in the perceptron. In *Mechanization of Thought Processes : Proceedings of a Symposium held at the National Physical Laboratory, November 1958*, volume 1, pages 421–456. HM Stationery Office, London, 1959.

[Ros62] F. Rosenblatt. *Principles of Neurodynamics : Perceptrons and the Theory of Brain Mechanisms.* Spartan Books, Washington, DC, 1962.

[Roy97] Deb Roy. Toco the toucan : a synthetic character guided by perception, emotion, and story. In *ACM SIGGRAPH 97 Visual Proceedings*, page 66, New York, USA, 1997. ACM Press.

[Sak84] M. Sakarovitch. *Optimisation combinatoire : méthodes mathématiques et algorithmiques.* Hermann, 1984.

[SBS99] B. Schölkopf, C. J. C. Burges, and A. J. Smola, editors. *Advances in Kernel Methods—Support Vector Learning.* MIT Press, Cambridge, MA, 1999.

[SEG00] Chris Stauffer, W. Eric, and W. Grimson. Learning patterns of activity using real-time tracking. *IEEE on Pattern Analysis and Machine Intelligence*, 22 :747–757, 2000.

[SG98] C. Stauffer and W.E.L. Grimson. Adaptive background mixture models for real-time tracking. The Artificial Intelligence Laboratory, Massachussets Institute of Technology Cambridge, MA02139, 1998.

[Shi93] A. Shinohara. Complexity of computing vapnik-chervonenkis dimension. In K. P. Jantke, S. Kobayashi, E. Tomita, and T. Yokomori, editors, *Algorithmic Learning Theory : Proc. of the 4th Workshop, ALT'93*, pages 279–287. Springer, 1993.

[SI95] A. Samal and P.Y. Iyengar. Human face detection using silhoulletes. *International Journal Of Pattern Recognition and Artificial Intelligence*, pages 845–867, 1995.

[SIB97] J. Davis S.S. Intille and A. Bobick. Real-time closed-world tracking. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 697–703, 1997.

[SJ89] K. Skifstad and R. Jain. Illumination independent change detection for real world image sequences. *Computer Vision, Graphics and Image Processing*, 46 :387–399, 1989.

[SKS01a] O. Javed S. Khan and M. Shah. Tracking in uncalibrated cameras with overlapping field of view. *Performance Evaluation of Tracking and Surveillance (PETS)*, December 9th 2001.

[SKS01b] Z. Rasheed S. Khan, O. Javed and M. Shah. Human tracking in multiple cameras. *The Eighth IEEE International Conference on Computer Vision (ICCV)*, pages 9–12, July 2001.

[SM94] Z. Sivan and D. Malah. Change detection and texture analysis for image sequence coding. *Signal Processing : Image Communication*, 6(4) :357–376, August 1994.

[Smi98] Stephen M. Smith. ASSET-2 : Real-time motion segmentation and object tracking. *Real-Time Imaging*, 4(1) :21–40, February 1998.

[Smo98] A. J. Smola. *Learning with Kernels*. PhD thesis, Technische Universität Berlin, 1998. GMD Research Series No. 25.

[SMY97] Rainer Stiefelhagen, Uwe Meier, and Jie Yang. Real-time lip-tracking for lipreading. In *Proceedings of Eurospeech*, Rhodes, Greece, 1997. lipreading, lip tracking.

[SNK96] S. Satoh, Yasuaki Nakamura, and Takeo Kanade. Name-it : Naming and detecting faces in news videos. *Network-Centric Computing (NCC) Special Issue*, 1996.

[SP72] E. A. Smith and D. R. Phillips. Automated cloud tracking using precisely aligned digital ATS pictures. *IEEE Transactions on Computers*, 21(7) :715–729, 1972.

[SP94] K. Sung and T. Poggio. Example-based learning for view based human face detection. Technical report, Massachussets Institute of Technology artificial intelligence laboratory and center for biological computation learning, 1994. A.I. Memo 1521.

[SP96a] K. Sobottka and I. Pitas. Extraction of facial regions and features using color and shape information. *Proceedings of the International Conference on Image Processing (ICIP)*, August 1996.

[SP96b] Karin Sobottka and Ioannis Pitas. Segmentation and tracking of faces in color images. In *Proceedings of the 2nd International Conference on Automatic Face and Gesture Recognition*, page 236, Washington, DC, USA, 1996. IEEE Computer Society.

[SP97] T. Starner and A. Pentland. Real-time american sign language recognition from video using hmms. In *Motion Based Recognition*, pages 227–243, 1997.

[SS02] B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, 2002.

[SSIB97] J. W. Davis S. S. Intille and A. F. Bobick. Real-time closed-world tracking. In *Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition*, page 697, Washington, DC, USA, 1997. IEEE Computer Society.

[ST98] Eli Saber and A. Murat Tekalp. Frontal-view face detection and facial feature extraction using color, shape, and symmetry based cost functions. *Pattern Recognition Letter*, 19(8) :669–680, 1998.

[STC97] O. Gascuel S. Thiria, Y. Lechevallier and S. Canu. *Statistique et méthodes neuronales*. Dunod, ISBN :2 10 003544 4, Paris, France, 1997.

[Sul94] G. D. Sullivan. Model-based vision for traffic scenes using the ground plane constraint. *In D.Terzopoulos and C.Brown (Eds) : Real-time Computer Vision*, 1994.

[sun01] Condensing computable scenes using visual complexity and film syntax analysis. In *2nd IEEE International Conf. on Multimedia and Expo. (ICME-2001)*, 2001.

[SV99] C. Schmid and V. Vogelhuber. Face dedection based on generic local descriptor and spatial constraints. *International Conference on Pattern Recognition*, 1 :1084–1087, 1999.

[SVM] Le logiciel SVM-light peut être téléchargé à l'adresse : http ://svm-light.joachims.org/.

[SXJ96] Allan D. Jepson Shanon X. Ju, Michael J. Black. Skin and bones : Multi-layer, locally affine, optical flow and regularization with transparency. In *Proceedings of the 1996 Conference on Computer Vision and Pattern Recognition*, page 307, Washington, DC, USA, 1996. IEEE Computer Society.

[sys05] Poséidon système. Système de Surveillance Assistée par Ordinateur pour l'aide à la prévention des noyades. http ://www.poseidon-tech.com/fr/, 2005.

[TC99] J.P. Thirion and G. Calmon. Deformation analysis to detection and quantify active lesions in three dimensional medical image sequences : Application to multiple sclerosis. *Medical Image Analysis*, 6 :429–441, 1999.

[Teg96] J. Teghem. *Programmation linéaire*. Ellipses-Marketing, 1996.

[TKABW97] A.J. Lipton T. Kanade, R.T. Collins, P. Anandan, P. Burt, and L. Wixson. Cooperative multi-sensor video surveillance. *DARPA Image Understanding Workshop*, pages 3–10, May 1997.

[TLS93] William B. Thompson, Pamela Lechleider, and Elizabeth R. Stuck. Detecting moving objects using the rigidity constraint. *IEEE Transaction on Pattern Analysis Machine Intelligence*, 15(2) :162–166, 1993.

[TM94] P. H. S. Torr and D. W. Murray. Stochastic motion clustering. In *ECCV '94 : Proceedings of the third European conference on Computer Vision (Vol. II)*, pages 328–337, Secaucus, NJ, USA, 1994. Springer-Verlag New York, Inc.

[Tor97] P. H. S. Torr. An assessment of information criteria for motion model selection. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, page 47, Washington, DC, USA, 1997. IEEE Computer Society.

[Toy96] Kentaro Toyama. Handling tradeoffs between precision and robustness with Incremental Focus of Attention for visual tracking. In *Working Notes AAAI Symp. on Flexible Computation in Intelligent Systems*, pages 142–147, Cambridge, MA, 1996.

[TS93] Demetri Terzopoulos and Richard Szeliski. Tracking with kalman snakes. pages 3–20, 1993.

[TSM85] D.M. Titterington, A.F. Smith, and U.E. Makov. *Statistical Analysis of Finite Mixture Distributions*. Wiley, 1985.

[uIbAR01] CAVIAR : Context Aware Vision using Image-based Active Recognition. http ://homepages.inf.ed.ac.uk/rbf/CAVIAR/, 2001.

[UNME95] T. Uhlin, P. Nordlund, A. Maki, and J.-O. Eklundh. Towards an active visual observer. In *ICCV '95 : Proceedings of the Fifth International Conference on Computer Vision*, page 679, Washington, DC, USA, 1995. IEEE Computer Society.

[Val84] L. G. Valiant. A theory of the learnable. *Communications of the ACM*, pages 1134–1142, 1984.

[Vap82] V.N. Vapnik. *Estimation of Dependences Based on Empirical Data [In Russian]*. English Translation : Springer-Verlag, New York, 1982.

[Vap95] V.N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, New York, ISBN 0-387-94559-8, 1995.

[Vap98] V Vapnik. *Statistical Learning Theory*. Wiley, New York, 1998.

[Vio95] P. Viola. *Alignment by Maximisation of Mutual Information*. PhD thesis, M.I.T., 1995.

[VJ04] P. Viola and M. Jones. Robust real-time face detection. *Internal Journal of Computer Vision*, 57(2) :137–154, 2004.

[WADP97] C.R. Wren, A. Azarbayejani, T. Darrell, and A.P. Pentland. Pfinder : real-time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7) :780–785, Jully 1997.

[WBPB95] L. Wu, J. Benois-Pineau, and D. Barba. Spatio-temporal segmentation of image sequences for object-oriented low bit-rate image coding. In *International Conference on Image Processing (ICIP)*, pages 2406–2409, 1995.

[WC04] J. Wang and Z. Chengqi. Support vector machines based on set covering. *Proceedings of the 2nd International Conference on Information Technology for Application*, 1(1) :181–184(4), 2004.

[Wel93] Pierre Wellner. Interacting with paper on the digitaldesk. *Commun. ACM*, 36(7) :87–96, 1993.

[WH60] B. Widrow and M.E. Hoff. Adaptive switching circuits. In *1960 IRE WESCON Convention Record*, volume 4, pages 96–104. IRE, New York, 1960.

[WJ95] M. P. Wand and M. C. Jones. *Kernel Smoothing*. Chapman & Hall, London, 1995.

[WKD96] J.V. Lrogmeier W.Y. Kan and P.C. Doerschuk. Model-based vehicle tracking from images sequences with an application to raod surveillance. *Opt. Eng.*, 35 :1723–1729, 1996.

[XO93] L. Xu and E. Oja. Randomized hough transform : Basic mechanisms, algorithms, and computational complexities. *CVGIP : Image Understanding*, 57(2) :131–154, 1993.

[YA98] Ming-Hsuan Yang and Narendra Ahuja. Detecting human faces in color images. volume 1, pages 127–130, 1998.

[Yak76] Y. Yakimovsky. Boundary and object detection in real world images. *Journal of ACM*, 23(4) :599–618, 1976.

[YAT81] Masahiko Yachida, Minoru Asada, and Saburo Tsuji. Automatic analysis of moving images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 3(1) :12–20, 1981.

[YC96] Kin Choong Yow and R. Cipolla. A probabilistic framework for perceptual grouping of features for human face detection. In *Proceedings of the 2nd International Conference on Automatic Face and Gesture Recognition (FG '96)*, page 16, Washington, DC, USA, 1996. IEEE Computer Society.

[YDGD03] Changjiang Yang, Ramani Duraiswami, Nail A. Gumerov, and Larry Davis. Improved fast gauss transform and efficient kernel density estimation. *International Conference on Computer Vision*, pages 464–471, 2003.

[YH94] G. Yang and T. S. Huang. Human face detection in complex background. *Pattern Recognition*, pages 53–63, 1994.

[YKA02] M.H. Yang, D. Kriegman, and N. Ahuja. Detecting faces in images : a survey. *IEEE on Pattern Analysis and Machine Intelligence*, 24, 2002.

[YSMW98] Jie Yang, Rainer Stiefelhagen, Uwe Meier, and Alex Waibel. Visual tracking for multimodal human computer interaction. In Clare-Marie Karat, John Karat, and Ian Horrocks, editors, *Human Factors in Computing Systems : CHI 98*, pages 140–147, Los Angeles, CA, USA, April 1998. ACM SIGCHI, Addison-Wesley Pub. Co.

[YY96] M. M. Yeung and B-L. Yeo. Time-constrained clustering for segmentation of video into story unites. In *ICPR '96 : Proceedings of the International Conference on Pattern Recognition (ICPR '96) Volume III-Volume 7276*, page 375, Washington, DC, USA, 1996. IEEE Computer Society.

[ZC01] Qiang Zhu and Jiashi Chen. A new approach for rotated face detection. *Proceedings of the ninth ACM international conference on Multimedia*, pages 537–539, 2001.

[Zha94] Zhengyou Zhang. Token tracking in a cluttered scene. *Image and Vision Computing*, 12(2) :110–120, 1994.

[Zha96] Y. Zhang. A survey of evaluation methods for image segmentation. *Pattern Recognition*, 29 :1335–1346, 1996.

Détection et suivi d'objets dans les scènes animées : Application à la vidéo surveillance

Résumé :

Dans le cadre de cette thèse nous nous sommes intéressés à l'application des outils d'apprentissage statistique aux problèmes d'extraction et de suivi d'objets dans le contexte de la surveillance vidéo par des caméras statiques. Notre étude se déroule en trois phases : la première consiste à détecter l'ensemble des objets en mouvement par l'analyse des variations de luminosité au cours du temps.

Nous avons proposé pour cela une méthode de détection de tels objets dans la continuité des travaux de Grimson et Stauffer à la base de la modélisation de la luminance des pixels par des mélanges des lois Gaussiennes. Nous proposons une règle de décision statistique au sens du MAP simplifiée pour satisfaire les contraintes temps-réel. Une régularisation des résultats de la détection par modélisation des champs des étiquettes en tant que champs de Markov nous a permis d'obtenir des masques de mouvement les plus complets aussi bien au niveau spatial que temporel.

Suite à cette première analyse, un deuxième processus de détection est exécuté en vue de l'identification d'un ou plusieurs objets d'intérêt. L'ensemble de ces objets est prédéfini au préalable par un opérateur. Dans le cadre de ce travail, nous avons envisagé une application concrète, celle de la détection de visages, tout en considérant des solutions statistiques de classification suffisamment généralistes ainsi qu'une étude des espaces de représentation adéquats. Nous avons étudié et déployé un classifieur à base de Support Vector Machine -SVM-.

Afin d'optimiser les temps de calcul, nous avons proposé une méthode d'élimination des vecteurs de support linéairement dépendants ainsi qu'un schéma de détection de l'objet d'intérêt uniquement dans les zones de mouvement précédemment détectées.

Enfin, la troisième contribution de cette thèse consiste en un outil de suivi d'objet appris avec un modèle de mouvement affine complet de premier ordre et ceci à la base du même formalisme SVM. Ici les paramètres de classifieur appris à l'étape de la détection sont réutilisés pour le suivi. Ce modèle permet ainsi de gérer les mouvements complexes des objets filmés dans des environnements naturels.

Discipline : Informatique

Mots-clés : Détection de mouvement, régularisation Markovienne, mélange de lois Gaussiennes, détection de visages, Support Vector Machine, Suivi d'objets, estimation du modèle affine complet.

Object detection and tracking in the context of video surveillance

Abstract:

In this work, we are focused on applications of statistical learning methods as a global framework for an intelligent video surveillance system with fixed cameras. The solution deals with such problems as motion detection, detection of an object of interest and tracking.

The process we choose is naturally divided into three parts : first we extract all moving objects in a video scene. Here, our contribution is a follow-up of Grimson and Stauffer research work. It is based on modelling luminance signal in each pixel by a mixture of Gaussians. We proposed a simplified MAP - based statistical decision rule in order to satisfy real-time constraints. A regularization of detection results is then proposed based on modelling of label fields by Markov fields. It allows for smoothing both spatially and temporally the detection results. As a follow-up of this first analysis step, we designed a second process to detect specific objects of interest, human faces in our case, using a Support Vector Machine classifier. In order to optimize computational time, we propose the reduction of the set of Support vectors set to those which are linearly independent and detection of objects of interest only in moving areas previously detected.

Finally, the third contribution of this work consists in a method of tracking of learned objects with a complete first order affine motion model. The method is based on the same SVM formulation and allows for the re-use of trained classifier. This model allows for capturing of complex motions of objects in natural environments

Discipline: Computer-Science

Keywords: Motion detection, markov regularization, Gaussian mixture, face detection, Support Vector Machine, object tracking, complet affine model estimation.

LaBRI,
Université Bordeaux 1,
351 cours de la Libération,
33405 Talence Cedex (FRANCE).