Détection automatique de chutes de personnes basée sur des descripteurs spatio-temporels : définition de la méthode, évaluation des performances et implantation temps-réel

Imen Charfi

SPIM

Thèse de Doctorat

école doctorale **sciences pour l'ingénieur et microtechniques**

U N I V E R S I T É D E B O U R G O G N E

N° | X | X | X

THÈSE présentée par

Imen CHARFI

pour obtenir le

Grade de Docteur de

l'Université de Bourgogne

Spécialité : **Informatique et Instrumentation de l'Image**

Thèse en cotutelle avec l'Université de Monastir

Détection automatique de chutes de personnes basée sur des descripteurs spatio-temporels : définition de la méthode, évaluation des performances et implantation temps-réel.

Soutenue le 21 Octobre 2013 devant le Jury :

Ali GHARSALLAH	Président	Professeur à la Faculté des Sciences de Tunis
Christophe DUCOTTET	Rapporteur	Professeur à l'Université de Saint-Etienne
Chokri BEN AMAR	Rapporteur	Professeur à l'ENIS de Sfax
Johel MITÉRAN	Directeur de thèse	Professeur à l'Université de Bourgogne
Rached TOURKI	co-directeur de thèse	Professeur à l'Université de Monastir
Julien DUBOIS	co-encadrant de thèse	Maître de conférence à l'Université de Bourgogne

SOMMAIRE

Remerciements

Ce travail est le fruit d'une thèse en cotutelle effectuée au sein du laboratoire Le2i, à l'Université de Bourgogne, et le laboratoire LEME, à la Faculté des Sciences de Monastir, et qui n'aurait jamais pu voir le jour sans le soutien moral et intellectuel de nombreuses personnes auxquelles je voudrais exprimer ma profonde reconnaissance.

Tout d'abord, je remercie très respectueusement Monsieur ALi Gharsallah, Professeur à la Faculté des Sciences de Tunis, pour l'honneur qu'il me fait en acceptant de présider le jury de cette thèse.

Je remercie chaleureusement Monsieur Christophe Ducottet, Professeur à l'Université de Saint-Étienne, et Monsieur Chokri Ben Amar, Professeur à l'École Nationale d'Ingénieurs de Sfax, pour l'intérêt qu'ils ont bien voulu porter à ce travail en acceptant de le juger et pour les précieuses remarques qu'ils ont apportées.

Je tiens à exprimer ma profonde reconnaissance à mon directeur de thèse Monsieur Johel MITERAN, Professeur à l'Université de Bourgogne, pour l'enrichissement formidable que j'ai pu en bénéficier en travaillant à ses côtés. Je le remercie pour ses réponses infaillibles à toutes les questions scientifiques que je pouvais me poser et même celles que je ne me posais pas. Sa bonne humeur, sa patience et ses encouragements ont été une source intarissable qui me poussait à avancer.

J'adresse toute ma gratitude à Monsieur Rached TOURKI, Professeur à l'Université de Monastir, pour ses conseils et encouragements et pour la confiance qu'il m'a accordée en acceptant de diriger cette thèse.

Je remercie également mon co-encadrant de thèse Monsieur Julien DUBOIS, Maître de Conférences à l'Université de Bourgogne, pour son aide efficace avec ses précieux conseils. Ses encouragements et sa bonne humeur contagieuse m'ont beaucoup aidée aux moments de doute.

J'ai toujours apprécié les remarques et les conseils de mon co-encadrant de thèse Monsieur Mohamed Atri, Maître de Conférences à la Faculté des Sciences de Monastir. Je le remercie pour son écoute, ses encouragements et la confiance qu'il m'a accordée.

Je tiens à remercier toutes les personnes que j'ai pu côtoyer dans le laboratoire ainsi que tous mes collègues et amis. A ce titre je remercie également ceux qui ont joué les cascadeurs en simulant des chutes dans nos séquences vidéos.

Je suis particulièrement reconnaissante envers mon collègue Wajdi pour son constant

soutien.

J'adresse un grand merci à mes chers parents, mes frères et sœurs, Houda et ma chère Raja qui ensoleillent ma vie. Ils ont accepté ma mauvaise humeur sans forcément comprendre les raisons et ont toujours été présents quand j'avais besoin.

J'adresse aussi un grand merci à Lazhar, Skander, Naceur et Hosni pour leur encouragement et leur disponibilité.

Je remercie également Mey, Omar, Yessine, Ahmed, Sarra, Iyed, Khalil, Zeineb, Khaled et Yesmine pour la joie qu'ils m'offrent en les voyant grandir.

INTRODUCTION

CONTEXTE GÉNÉRAL

Dans l'Union Européenne, selon le "Center for Research and Prevention of Injuries", les blessures causées par les chutes chez les personnes âgées sont cinq fois plus fréquentes que les autres ([42]). En France, ceci est confirmé par les projections démographiques de l'INSEE4 de 2005 qui compte 450 000 chutes par an [19]. En effet, on prévoit entre 2000 et 2050 un quasi-doublement du nombre des personnes de plus de 65 ans. Quant aux plus de 75 ans, leur nombre serait triplé voire même quadruplé pour les plus de 85 ans. Autrement dit, en 2030, la France comptera 7 millions supplémentaires de personnes âgées de plus de 60 ans par rapport à 2005. Ce vieillissement démographique provoque l'augmentation des risques dans l'habitat quotidien des personnes en question. Les accidents de la vie courante chez les personnes de plus de 65 ans sont constitués pour 80% par des chutes dont 62% ont lieu à domicile.

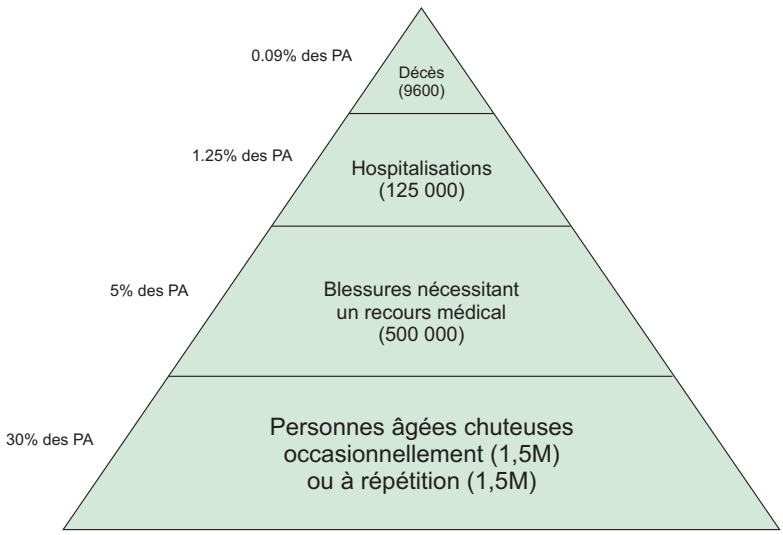


FIGURE 1 – Importance des chutes et de leurs conséquences annuelles chez les personnes âgées (PA) - Données françaises, adapté de Bourdessol et Ermanel [18].

Les conséquences des chutes (hospitalisations, décès), sont schématisées par la pyramide reportée figure1, extraite du rapport de M. Boulmier [18]. Souvent incapable de se relever voire même inconsciente après une chute, une personne âgée ne peut pas appe-

ler les urgences ou un centre médical pour lui venir en aide. Ceci a incité de nombreux laboratoires de recherche à tenter de mettre en place des solutions afin de limiter les conséquences de ce problème de santé publique, qui engendre problèmes humains et financiers importants.

Au moment où nous avons commencé ce travail, la plupart des solutions proposées aux problèmes de chutes sont basées soit sur des capteurs portatifs tels que les accéléromètres et/ou gyroscopes pour détecter le mouvement de la personne, soit basées sur des capteurs (présence, pression) équipant l'environnement où se trouve la personne. Ainsi, Lord [88] en 1991 et Williams [149] en 1998, ont mis en place un dispositif qui détecte l'impact d'un choc sur le sol, et un capteur d'inclinaison pour détecter si la personne est couchée. La personne doit porter ce dispositif sur elle. Noury et al. [105] ont conçu un capteur à partir de trois accéléromètres disposés orthogonalement et d'un microcontrôleur qui détermine l'inclinaison du porteur du dispositif. Lindeman [85] a proposé un appareil conçu à base d'un accéléromètre 3D. Une chute est détectée si trois seuils appliqués à la vitesse et l'accélération 3D sont dépassés. D'autres méthodes [106], [20] se basent sur les gyroscopes afin de de déterminer la vitesse angulaire et distinguer les chutes des activités de la vie quotidienne.

Bien qu'ils puissent assurer une détection prometteuse de chutes, les capteurs portatifs présentent des limites d'utilisation. D'une part, ils représentent une solution peu pratique, car pas toujours acceptée par les personnes âgées ou trop encombrantes pour une utilisation quotidienne. D'autre part, le risque que la personne oublie de se servir de ce genre de dispositifs est important. De ce fait, des systèmes "passifs" ont été envisagés : installer des capteurs à l'intérieur de l'environnement où réside l'utilisateur représente une solution intéressante. A titre d'exemple, le système GAITRite proposé par Alwan [2] a été utilisé par Besser [15] pour prévoir les chutes en se servant de paramètres temporels et spatiaux extraits lors du passage de la personne sur ce dispositif présenté sous forme d'un tapis actimétrique (permettant de capter et d'enregistrer les mouvements).

D'un autre côté, la vision par ordinateur offre des solutions prometteuses pour différentes applications. La détection d'objets tels que les visages ou les piétons et la reconnaissance des événements ont connu d'énormes progrès dans ce domaine suite à l'intérêt que les chercheurs leurs ont porté depuis quelques années. Ceci a été ensuite étendu aux applications spécialisées comme la détection d'événements inhabituels comme les chutes. Ainsi, ces dernières années, des dizaines d'équipes de recherche à travers le monde se sont intéressées à la vision artificielle pour la détection automatique des chutes des personnes âgées, qui doit permettre le déclenchement rapide des secours.

C'est exactement dans ce cadre que se situe ce travail de thèse, c'est à dire la vidéo-assistance des personnes âgées à travers la détection automatique de chutes en temps réel.

Objectifs et motivations

Notre objectif premier est donc d'implanter un système autonome qui détecte en temps réel la chute d'une personne dans une scène vidéo. A sa détection, l'information de "chute" ainsi que les images de l'événement lui-même peuvent être extraits de la scène vidéo puis éventuellement envoyés à un centre d'assistance pour lever le doute et appeler des secours. Dans ce contexte, la qualité de l'information visuelle qu'on souhaite transmettre est d'une grande importance, puisqu'elle permettra de prendre la "bonne" décision. Par conséquent, si la compression de la vidéo est incluse dans la chaîne de traitement afin de réduire la bande-passante, il est intéressant d'adapter la qualité du flux transmis à l'aide de l'information de chute détectée. La caméra peut donc en temps normal envoyer un flux vidéo continu avec un très haut taux de compression. Si une chute est détectée, le taux de compression est momentanément diminué dans le but de transmettre une meilleure qualité d'image, combiné avec l'information qu'une chute a été détectée.

Ce travail de thèse s'inscrit dans le cadre d'une coopération franco-tunisienne, projet CMCU intitulé « **Plateforme de compression vidéo pour des applications basse résolution incorporant des mesures temps réel de reconnaissance de formes** » au sein de deux laboratoires : Laboratoire Électronique, Informatique et Image (Le2i) de l'Université de Bourgogne et le Laboratoire d'Électronique et Microélectronique (Leme) de l'Université de Monastir. Il est à noter que si le contexte général du travail de recherche était bien la compression adaptative, cette compression ne sera pas abordée dans ce mémoire, qui est exclusivement consacré à la méthode de détection de chute en elle-même.

D'autre part, le contexte général étant également lié au projet « Smart Camera » du département Électronique du laboratoire Le2i, notre second objectif était une implantation du système de détection dans une caméra intelligente. Cette implantation devait nous permettre de répondre aux contraintes de compacité et de traitement temps réel de ce type d'applications.

Contributions

Les travaux réalisés au cours de cette thèse s'articulent autour de deux phases de recherches ayant pour but de proposer une nouvelle solution de détection de chutes en temps réel basée sur des descripteurs spatio-temporels adaptée à une implantation dans une caméra intelligente.

Une première contribution sera la constitution d'une base de vidéos large et réaliste qui permettra d'évaluer la méthode. Elle se compose d'un nombre important de scénarios de

chutes, d'événements de la vie quotidienne ainsi que d'événements qui pourraient facilement être confondus avec des chutes. La métrique d'évaluation que nous proposerons à ce niveau peut également être considérée comme une contribution de ce travail.

La deuxième contribution vient des descripteurs nommés par la suite "Spatio-Temporal Human Fall descriptors" ($STHF$), qui seront construits à partir d'attributs géométriques extraits de l'image de mouvement, et leurs variations temporelles. Dans le but de réduire la complexité des descripteurs d'une part et d'améliorer les résultats de classification d'autre part, nous avons effectué une sélection automatique d'attributs et de transformations les plus pertinentes.

Enfin, la dernière contribution à la détection de chute temps réel sera l'implantation de l'ensemble de l'algorithme sur une plate-forme de prototypage rapide pour caméra intelligente.

Organisation du mémoire

Le premier chapitre, après une présentation qui permet de mieux situer la détection de chutes et par conséquent notre contribution dans le contexte scientifique que nous abordons, passe en revue les principales méthodes utilisées dans l'état de l'art pour couvrir les différentes étapes d'un système générique de détection de chutes. Nous y mettrons en évidence deux méthodes de classification les plus robustes et communément utilisées dans la littérature : nous détaillerons ainsi les bases des Support Vector Machine (SVM) qui sera utilisé comme classifieur pour évaluer les performances de notre système, et le Boosting qui lui sera comparé.

Le deuxième chapitre est consacré à un état de l'art des bases de vidéos utilisées en reconnaissance d'actions et détection de chutes, ainsi que des méthodologies traditionnellement utilisées pour l'évaluation des performances de détection en terme d'erreur de classification. Nous en arriverons à la présentation de notre propre base de vidéos et des protocoles qui seront utilisés lors de l'évaluation. Nous terminerons par une présentation de la métrique permettant d'évaluer les erreurs de classification tout en tolérant une erreur sur la précision de la localisation temporelle de cette détection.

Le troisième chapitre présente notre méthode dans son ensemble, depuis la détection de la personne en mouvement, jusqu'à la détection finale, en passant par l'extraction d'attributs géométriques, leurs analyses temporelles, puis la définition des descripteurs spatio-temporels $STHF$ qui seront optimisés en terme d'erreur de classification, grâce à des méthodes de sélection d'attributs.

L'évaluation des performances sera abordée dans le quatrième chapitre. Nous présenterons notamment l'évaluation de notre méthode de détection automatique de la personne en mouvement, et l'influence de divers paramètres de la méthode. Nous en viendrons

alors à une comparaison avec la méthode basée sur le Boosting, et terminerons par une analyse des confusions les plus fréquentes.

Dans le dernier chapitre, nous aborderons l'étude de l'implantation matérielle de la phase de détection de chute, dans le contexte des caméras intelligentes, sur plate-forme matérielle basée sur le composant Zynq de type System on Chip (SOC). En suivant une démarche de type Adéquation Algorithme Architecture, nous adapterons l'algorithme afin d'obtenir un bon compromis performances de classification/temps de traitement.

1

LA DÉTECTION DE CHUTES : UN CAS PARTICULIER DE RECONNAISSANCE D'ACTIVITÉS HUMAINES

1.1/ GÉNÉRALITÉS

La reconnaissance d'activités humaines par vision artificielle est un domaine de recherche très actif. Cette reconnaissance consiste, grâce à un système de vision constitué d'une ou plusieurs caméras, à analyser une scène et à donner un nom à chaque phase particulière de l'activité des personnes présentes dans cette scène. Les activités ciblées sont généralement celles qui sont considérées par la société comme anormales, suspectes, c'est à dire à l'origine d'un danger potentiel, comme l'abandon d'un bagage dans une gare [23], le franchissement de lignes de sécurité sur les quais de métro [99] ou la chute accidentelle d'une personne [83] [119] [148].

En reconnaissance de forme et pour les images fixes, "reconnaissance" signifie généralement que le nom d'un objet ou d'une personne (reconnaissance de visages, reconnaissance de caractères, etc.) est produit en sortie du système de mesure, alors que pour la "détection", la sortie du système est "présence" ou "absence" d'un élément recherché. C'est le cas par exemple de la détection de visages ou de piétons dans une scène. La plupart du temps, la détection est accompagnée de la localisation de l'élément recherché dans la scène, c'est pourquoi la détection en image fixe est presque toujours associée à un balayage au sens large de la scène explorée.

En ce qui concerne les images animées ou vidéos, la distinction est à peu près similaire : la reconnaissance d'un événement A parmi un ensemble de k événements généralement prédéfinis, consiste à donner un nom à l'action qui vient de se dérouler. Il peut s'agir d'actions comme marcher, sauter, courir, applaudir, sourire, etc. La détection quant à elle consiste à repérer dans le temps la réalisation d'une activité précise. Cette détection est alors suivie du déclenchement d'un signal de type alarme si l'événement recherché a eu lieu.

Dans le cadre de cette étude, nous nous plaçons dans le cas de la détection de chute, qui

est donc un cas particulier de la reconnaissance d'activités humaines à deux classes : "chute" et "non-chute". La localisation précise de l'événement dans la scène ne sera pas abordée, étant donné le domaine d'application envisagé. Toutefois, la chute n'étant qu'un cas particulier d'action, nous nous sommes intéressés lors de l'étude bibliographique liée à ce sujet à des méthodes de détections plus générales afin d'en extraire les éléments les plus pertinents et d'en réaliser une synthèse.

Quels que soient les domaines d'applications, ces reconnaissances d'activités demandent trois grandes étapes :

– une analyse spatiale préliminaire de la scène appelée aussi analyse instantanée comme par exemple la détection de piétons,
– une analyse temporelle de la scène, comme l'analyse de trajectoire, le filtrage particulaire ou le filtrage de Kalman, permettant au système de prendre une décision prenant en compte plusieurs images successives,
– la prise de décision en tant que tel.

De nombreuses méthodes ont fait l'objet de recherches et ont été décrites dans la littérature pour réaliser ces étapes, que ce soit pour la reconnaissance de gestes simples (tels que le mouvement des bras, la marche, etc.) ou pour le cas particulier de la chute humaine.

Toutes les méthodes qui nous concernent sont basées sur la vision artificielle et peuvent être établies avec une ou plusieurs caméras selon les contraintes initialement imposées. Par conséquent, l'analyse spatiale ou temporelle de la scène vidéo peut exploiter des informations 2D ou 3D.

D'autre part, il existe une grande diversité de méthodes permettant la prise de décision finale à partir de l'analyse des données. Certaines font appels à des méthodes de classification supervisée, d'autres non supervisée, comme nous le verrons en détails par la suite.

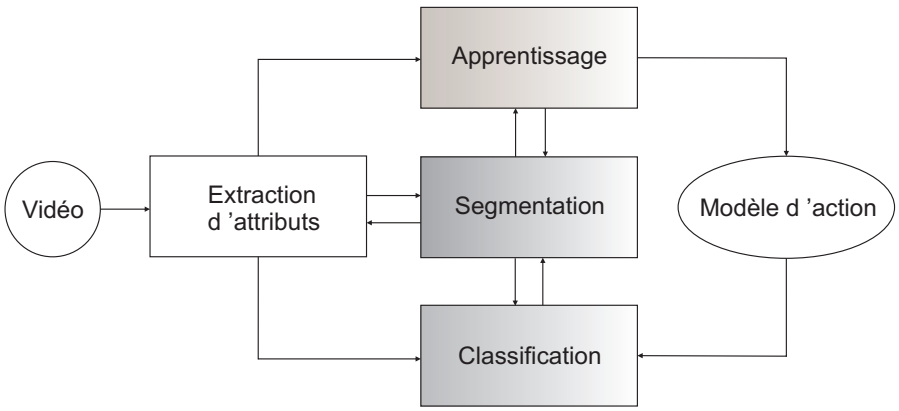


FIGURE 1.1 – Système générique de reconnaissance d'actions humaines [146].

Ainsi, nous présentons dans ce premier chapitre l'état de l'art des méthodes principales

développées pour réaliser chacune des étapes qui constituent le système générique typique de reconnaissance d'actions représenté Figure 1.1.

L'extraction des attributs est la tâche initiale de tout système de reconnaissance. Dans le cas de la reconnaissance d'actions, elle consiste à extraire de la vidéo les informations discriminantes de posture et de mouvements en fonction des actions humaines à détecter. Plusieurs représentations sont possibles, d'une représentation de modèles complexes jusqu'à une représentation simple d'images de type silhouettes. Des adaptations algorithmiques aux contraintes liées à l'environnement doivent être prises en compte, comme la robustesse aux occlusions. Nous détaillerons dans les sections 1.2 et 1.3 les différents types d'attributs habituellement employés pour la détection d'actions.

Les méthodes de classification, très nombreuses, ont pour but de déterminer de manière automatique la classe d'un élément inconnu. Les méthodes auxquelles nous nous sommes restreints (des ouvrages entiers leur sont consacrés, une des références les plus connues étant les travaux de Duda et Hart ([35, 36]) consistent pour la plupart à apprendre des modèles à partir des attributs extraits d'un ensemble d'apprentissage. Ces modèles sont ensuite utilisés pour classer un élément à partir de nouveaux attributs observés dans d'autres séquences d'images. Ces méthodes nécessitent donc un ensemble de données d'apprentissage et de test critiques pour l'évaluation des performances du système. Nous aborderons ce point particulier dans le chapitre 2. Les méthodes que nous avons abordées seront décrites section 1.4.

La segmentation d'actions consiste à découper chaque vidéo d'apprentissage afin de distinguer dans une même séquence les actions les une des autres.

Nous allons maintenant décrire l'état de l'art concernant les attributs spatiaux habituellement utilisés dans les différentes méthodes de détection de chutes de personnes de la littérature. De même, nous décrirons les méthodes principales permettant la prise en compte de l'aspect temporel de cette détection, puis les deux grandes méthodes de classification qui ont été abordées au cours de cette étude. Nous terminerons par un état de l'art des méthodes complètes de détection de chute.

1.2/ L'ANALYSE SPATIALE DE LA SCÈNE - LES ATTRIBUTS SPATIAUX

1.2.1/ PREMIÈRE ÉTAPE : LA DÉTECTION DE PERSONNES

Le choix des attributs spatiaux dépend fortement de l'action à détecter. En ce qui concerne la chute de personne, la première étape à réaliser consiste à détecter et localiser la personne dans la scène. Cette opération peut être réalisée grâce à une analyse purement statique, basée sur l'apparence, ou combinée à une approche analysant le mouvement. La qualité de cette détection aura une influence considérable sur les perfor-

mances finales du système, puisque la non détection d'un acteur engendrera obligatoirement la non détection ou reconnaissance de l'action.

Ce type de détection représente donc un thème largement étudié dans la littérature, car comme illustré figure 1.2, il est difficile d'établir un modèle définitif d'une personne. En effet, celui-ci peut apparaître, selon l'âge ou la position dans la scène, dans différentes postures et présente une large variabilité de taille. D'autre part, les tenues vestimentaires varient largement selon les goûts et les saisons, ce qui induit de grandes variations d'apparences.

Étant donné les contraintes de notre application, nous nous limitons dans cette étude aux méthodes de détection de personnes dans un flux vidéo provenant d'une seule caméra fixe et en 2D.

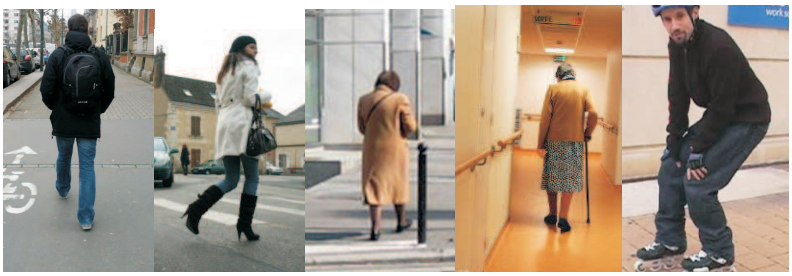


FIGURE 1.2 – Exemples d'images de personnes.

Le choix des caractéristiques qui décrivent les personnes est par conséquent très important. En fonction de ce choix, nous proposons de classer les méthodes en deux catégories :
– les approches basées sur l'apparence qui peuvent être globales ou locales,
– les approches basées sur l'analyse du mouvement.

1.2.1.1/ APPROCHES BASÉES SUR L'APPARENCE

Cette famille d'approches considère la détection de personnes comme un problème de reconnaissance de formes à deux classes (piétons et non-piétons). Ces méthodes s'appuient sur des modèles d'apprentissages statistiques construits à partir d'une base d'exemples. On distingue en général les approches globales et les approches locales. Les approches globales consistant à prendre une décision unique pour toute l'image (par exemple donnant le nom de la personne ou de l'objet unique présent dans la scène) sont nombreuses (on peut citer les approches basées sur l'Analyse en Composantes Principales [91, 142] ou encore sur les descripteurs de Fourier généralisés [126, 93]), mais ne correspondent pas à notre besoin qui est ici de localiser la personne en mouvement dans la scène.

Parmi les approches locales les plus utilisées et les plus efficaces, nous pouvons citer

deux grandes familles de méthodes :

– Les méthodes basées sur l'extraction de points ou régions d'intérêts possédant des propriétés d'invariance spécifiques (coins de Harris [96, 84], EBR [1] [135], MSER [2] [92], suivis de l'extraction de descripteurs locaux (SIFT [3] [89], HOG [4][29], etc.), qui servent à établir le modèle permettant la reconnaissance finale. Nous pouvons citer en particulier, pour ses très bonnes performances en détection de piétons, le travail de Dalal [29] qui a utilisé l'approche basée sur les descripteurs HOG, qui eux-mêmes alimentent un classifieur de type SVM.
– Les méthodes basées sur un parcours régulier de l'image, et où la détection est accélérée grâce une approche de type de cascade de classifieurs faibles. L'origine en est la méthode bien connue de Viola et Jones, qui ont proposé un système très efficace de détection en temps réel de visages [138] puis de piétons [137]. En ce qui concerne les visages, leur détection repose sur la combinaison d'un ensemble de filtres spatiaux (horizontaux, verticaux et diagonaux) avec une représentation originale de l'image, l'image intégrale, et la construction d'un classifieur puissant, basé sur la technique du Boosting. Cette technique sera détaillée dans la section 1.4.2. Elle est bâtie sur le principe d'une combinaison de classifieurs très simples. En ce qui concerne les piétons, le classifieur utilise l'apparence, avec une méthode proche de celle utilisée pour les visages, combinée avec le mouvement, en prenant en compte deux images successives.

La popularité et l'efficacité des méthodes basées sur une cascade de type Viola-Jones nous ont conduits à réaliser des expériences préliminaires basées sur ces principes, à l'aide des descripteurs de type HOG et de la cascade de type Haar implantée dans la bibliothèque OpenCV [21]. Toutefois, les performances de détection n'étant pas à la hauteur de nos espérances, notamment en terme de temps pour les HOG et en terme de détection pour la cascade de Haar, nous nous sommes orientés vers une approche basée mouvement, qui, nous le verrons, donnera des performances satisfaisantes étant donné nos contraintes à la fois de temps et d'unicité de la personne dans la scène.

1.2.1.2/ APPROCHES BASÉES SUR LE MOUVEMENT

La chute étant caractérisée par un mouvement important, il est plus important pour nous de détecter ce mouvement que de localiser la personne quand elle est immobile par exemple. Les approches de détection de personnes basée sur le mouvement consistent la plupart du temps à déterminer quels sont les pixels en mouvement, par la différence entre images successives ou par soustraction d'une image de fond. Certaines vont jusqu'à l'estimation des vecteurs mouvement en tout point. Cette évaluation conduit à une

1. Edge-Based Regions
2. Maximally Stable Extremal Regions
3. Scale-Invariant Feature Transform
4. Histogram of Oriented gradients

segmentation de l'image, généralement en deux régions (pixels en mouvement et pixels immobiles). De cette image binaire, on peut extraire un certain nombre de caractéristiques géométriques permettant la reconnaissance de la forme ou de l'action.

Dans cette section, nous allons présenter quelques exemples de détections de personnes. La localisation des différentes parties du corps humain sur une image est une tâche très difficile surtout lorsque la personne effectue des actions complexes. Ces méthodes restent donc limitées aux actions simples.

Segmentation zones fixes/zones en mouvement Les méthodes de soustraction de fond (considéré comme étant l'ensemble des zones fixes de l'image) sont des approches très populaires et largement utilisées comme première étape d'un système de vidéosurveillance basé sur une caméra statique. Pour cette raison, cette étape doit être la plus simple et la plus rapide possible. Les systèmes conçus par Haritaoglu dans [56], par Wren [151] représentent des exemples de cette thématique. Leur principal avantage est de permettre d'envisager un système de détection peu gourmand en ressources matérielles, adapté à une implantation temps réel, en restreignant l'analyse à une zone réduite de l'image. Ces méthodes ont été largement utilisées et ont fait leurs preuves dans de nombreux cas pratiques notamment en détection de chute [148, 86, 118, 83, 6]. Une étude comparative complète a été établie dans [14] présentant les principales méthodes de soustraction de fond. Nous ne présenterons dans ce manuscrit que les deux principales méthodes évaluées au cours de ce travail. La première méthode, ne fait aucune hypothèse sur le type de fond, et peut donc présenter quelques défauts lors de similitudes entre la personne et ce même fond. La seconde méthode est basée sur l'établissement d'un modèle statistique du fond, et prend donc mieux en compte les variations locales de luminance. Dans les deux cas, la caméra doit être statique, puisque ces méthodes supposent que le fond évolue de manière très lente par rapport au mouvement de la personne. Elles consistent à seuiller la différence entre image courante et fond de la manière suivante :

$$F_t(p) = \begin{cases} 1 & si \quad d(I_t(p), B_t(p)) > \tau \\ 0 & sinon, \end{cases} \tag{1.1}$$

où $F_t(p)$ est l'ensemble des pixels $p = (i, j)$ en mouvement (on l'appelle aussi l'image silhouette) à l'instant t, d est la distance entre les luminances des pixels de l'image courante, à l'instant t, c'est à dire $I_t(p)$ et leurs correspondants dans le modèle de l'arrière-plan estimé à l'instant t, c'est à dire $B_t(p)$. Ces méthodes varient essentiellement dans leur manière de modéliser l'arrière-plan et dans le choix de la distance d utilisée.

Une des variantes les plus connues et simples à implanter est la différence entre l'image courante et l'image de fond qu'on actualise régulièrement pour faire face aux changements d'éclairage et déplacement temporaires de meubles, petits objets, etc, selon

l'équation 1.2.

$$B_{t+1}(p) = (1-\alpha)B_t(p) + \alpha I_t(p) \tag{1.2}$$

où la valeur de α varie entre 0 et 1. Les pixels formant l'objet en mouvement sont ensuite détectés par l'intermédiaire d'un seuillage de l'une des distances d_0, d_1, d_2 et d_∞ qu'on choisit en fonction des images qui peuvent être en niveau de gris (équation 1.3) ou couleurs (voir les équations 1.4, 1.5 et 1.6). Notons qu'on désigne par R, V et B respectivement les composantes *Rouge, Vert et Bleu*.

$$d_0 = |I_t(p) - B_t(p)| \tag{1.3}$$

$$d_1 = \left|I_t^R(p) - B_t^R(p)\right| + \left|I_t^V(p) - B_t^V(p)\right| + \left|I_t^B(p) - B_t^B(p)\right| \tag{1.4}$$

$$d_2 = (I_t^R(p) - B_t^R(p))^2 + (I_t^V(p) - B_t^V(p))^2 + (I_t^B(p) - B_t^B(p))^2 \tag{1.5}$$

$$d_\infty = max\left\{\left|I_t^R(p) - B_t^R(p)\right|, \left|I_t^V(p) - B_t^V(p)\right|, \left|I_t^B(p) - B_t^B(p)\right| \right\} \tag{1.6}$$

Les variantes de cette méthode ont été utilisées dans plusieurs travaux de détection de chutes et ont donné des résultats satisfaisants surtout en temps de traitement [86].

La variante proposée par Li dans [81] consiste à réaliser une estimation probabiliste du fond, puis à appliquer une décision basée sur la règle de Bayes pour finaliser la segmentation. Ce système vise les scènes où le fond peut être stationnaire ou complexe et représenter des changement brusques ou progressifs. Le principe est illustré dans le schéma de la Figure 1.3. L'algorithme est divisé en quatre étapes principales.

Lors de la première étape, dite "détection de changement" *change detection*, la différence temporelle entre deux images successives et la différence par rapport au fond sont effectuées afin, lors de la deuxième étape dite "classification du changement", de dissocier ceux stationnaires aux ceux pixels en mouvement. Pour un pixel jugé stationnaire, un vecteur d'attributs couleurs est généré avec $L = 64$ niveaux pour chaque composante couleur. De plus, un vecteur d'attributs de co-occurence de $L = 32$ niveaux est généré pour chaque pixel en mouvement. Ce vecteur est ensuite comparé aux vecteurs de la même classe provenant de la table des attributs statistiques apprise selon le critère de classification choisi. La troisième étape consiste à réaliser des opérations de morphologie mathématique afin d'éliminer la plupart des erreurs de classification. Le fond est alors mis à jour (les étapes correspondantes sont présentées en gris dans la Figure 1.3). Elle consiste à adapter le modèle de fond construit aux changements possibles au cours du

temps. Une adaptation des vecteurs attributs statistiques et une adaptation du modèle de fond de référence sont donc réalisées.

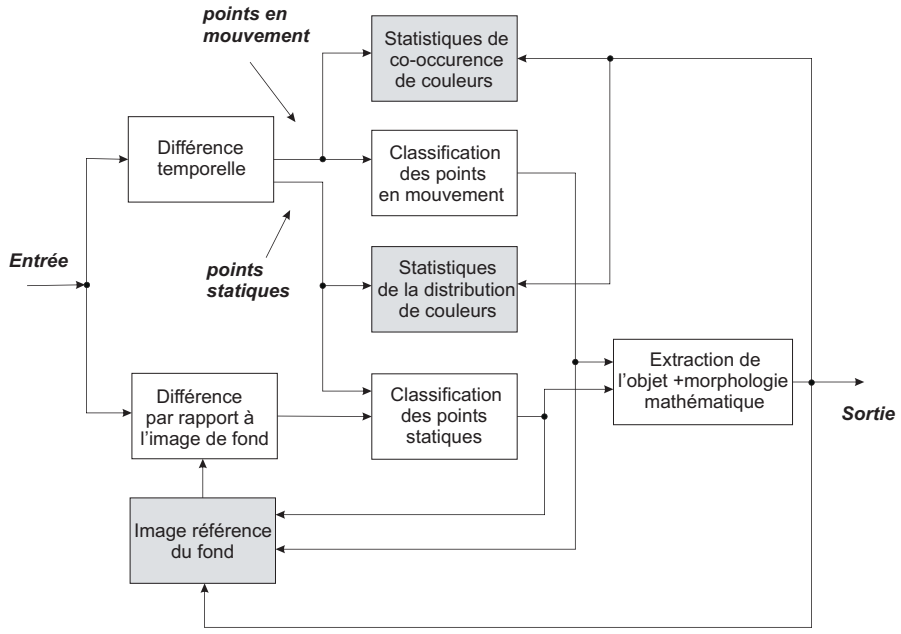


FIGURE 1.3 – Schéma bloc de l'algorithme de soustraction de fond proposé dans [81].

Cette méthode donne de bons résultats et a été reprise dans divers travaux, comme [78],[124],[128]. Elle est, de plus, librement disponible dans la librairie OpenCV, et malgré sa complexité, relativement rapide. Ceci nous a conduits à l'utiliser dans une partie des travaux que nous avons menés.

Le flux optique Toujours dans les approches liées à la segmentation des régions en mouvement, plusieurs méthodes proposent d'évaluer le flux optique [60],[10] pour permettre la détection de personnes. L'idée est de représenter le mouvement réel 3D grâce à une projection dans le plan image du mouvement 2D observé, ce qui donne un champs de vecteurs à deux dimensions.

Parmi les algorithmes les plus populaires et les plus rapides, celui de Lucas et Kanade [90] consiste à calculer le déplacement d au point $\mathrm{p} = (i, j)$ à l'instant t selon l'hypothèse de conservation de l'intensité lumineuse I traduite par l'équation suivante :

$$\forall \mathrm{p}, \forall t > 0, I(\mathrm{p} + d, t + 1) - I(\mathrm{p}, t) = 0 \tag{1.7}$$

Ainsi, le vecteur qui minimise la fonction d'erreur quadratique ε, sur un voisinage $\vee(\mathrm{p})$ de p donnée par l'équation 1.8, représente l'estimation du flux optique.

$$\varepsilon(d) = \sum_{\mathsf{y} \in \vee(\mathsf{p})} [I(\mathsf{y} + d, t + 1) - I(\mathsf{y}, t)]^2 \tag{1.8}$$

Le calcul du flux optique s'est montré utile lorsque la caméra est en mouvement. Elzein et al. [39] traitent ainsi les vidéos qui proviennent d'une caméra installée dans un véhicule.

Dans le cadre de la reconnaissance d'action, le flux optique est parfois utilisé afin de restreindre la zone de recherche dans l'image, autrement dit, définir les régions d'intérêts dans l'image contenant le mouvement. Gao [49] utilise par exemple le flux optique pour suivre les avant-bras et la tête d'une personne âgée prenant son repas. Dans [66], il est employé dans le but de suivre uniquement les pieds d'une personne.

Bien qu'il existe de nombreux algorithmes permettant d'estimer ce flux (Barron et al. [10] ont notamment proposé neuf algorithmes précurseurs de Horn et Schunck [60]), Liu et al. [87] ont montré que le flux optique reste une méthode très coûteuse en temps de calcul quelle que soit l'implantation choisie.

Au sein du laboratoire Le2i, plusieurs études ont été menées pour implanter l'estimation du mouvement dans le cadre de la compression MPEG. Plusieurs architectures optimisées ont été proposées [38]. Toutefois la complexité importante de ces architectures est dûe à la nécessité d'obtenir le vecteur mouvement en tout point, ce qui n'est pas une nécessité absolue pour la détection de chute. Nous verrons un peu plus loin que les deux méthodes finalement retenues ont été la soustraction simple du fond et la méthode de Li [81].

1.2.2/ DEUXIÈME ÉTAPE : L'EXTRACTION D'ATTRIBUTS GÉOMÉTRIQUES

A partir du moment où l'on sait où se trouve la personne dans la scène, et ce à tout instant (sauf si elle est parfaitement immobile), il reste à extraire des attributs caractéristiques de la forme, de la position ou même de la posture, avant de les transmettre à la phase de classification. Nous allons maintenant présenter les différents types d'attributs géométriques utilisés par les chercheurs en vue de mettre en place un système de reconnaissance d'actions. Nous nous focalisons particulièrement aux attributs choisis pour les méthodes de détection de chutes.

1.2.2.1/ LA BOITE ENGLOBANTE

L'image de mouvement est donc binaire, et les pixels en mouvement sont blancs (voir figure 1.4). L'image est constituée de R régions potentiellement indépendantes. Chacune de ces régions est constituée d'un ensemble de pixels blancs connexes. On peut alors définir la boite englobante comme étant le rectangle de surface minimum contenant tous les pixels blancs d'une même région. Cette boite englobante peut servir de base à la

localisation et l'estimation de la position d'une personne. Les informations de sa posture peuvent être déduites à partir de la hauteur et la largeur de cette boite englobante. Par exemple, lorsque la personne est allongée ou lorsqu'elle est tombée, la hauteur devient inférieure à la largeur contrairement aux situations dans lesquelles la personne est en position assise ou debout. Le rapport de ces deux grandeurs est donc caractéristique de la posture ; il a d'ailleurs été utilisé au cours de plusieurs études de détection de chutes [134, 3, 139, 86, 73] .

Toutefois, on peut noter que les algorithmes de segmentation précédents ne garantissent pas que la personne est bien définie par une seule région au sens mentionné ci-dessus. Il sera par conséquent parfois nécessaire d'introduire une phase de fusion des régions.

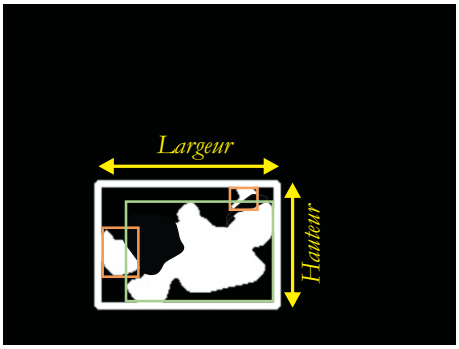


FIGURE 1.4 – Exemple de boite englobante : fusion de $R = 3$ régions indépendantes des pixels blancs représentant le mouvement.

1.2.2.2/ LES MOMENTS GÉOMÉTRIQUES

Parmi les autres grandeurs souvent utilisées en reconnaissance de forme et notamment de chutes, on trouve les moments géométriques [102]. Ceux-ci sont calculés pour chacune des R régions définies ci-dessus, dans la boite englobante, avant ou après fusion, suivant les méthodes. Les moments d'ordre p, q sont donnés par l'équation suivante :

$$M_{pq} = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} x^p y^p I(x, y) dx dy \tag{1.9}$$

Ces moments, comme ceux de Hu[61] ou de Zernike[131], font partie des attributs traditionnellement utilisés en reconnaissance de formes.

1.2.2.3/ ELLIPSE

En complément ou à la place de la boite englobante, il est commun dans la littérature de modéliser grossièrement le piéton à l'aide d'une ellipse englobante. Cette ellipse présente l'avantage d'avoir des propriétés géométriques relativement simples à exploiter, comme

l'orientation de ses axes principaux. Les caractéristiques géométriques de l'ellipse ont été utilisées avec succès dans le cadre de la détection de chute par Foroughi dans [44], Rougier dans [117] et Liao [83]. Nous les inclurons dans la liste des attributs candidats de notre méthode.

Les coordonnées du centre de l'ellipse d'une région donnée sont (E_x, E_y) et définis par :

$$E_x = M_{10}/M_{00} \qquad \text{et} \qquad E_y = M_{01}/M_{00} \tag{1.10}$$

A partir de ces coordonnées, il est possible de calculer les moments centraux m_{pq}, qui eux-mêmes servent de base à la détermination de l'orientation E_o de l'ellipse, ainsi qu'à la matrice de covariance J :

$$m_{pq} = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} (x - E_x)^p (y - E_y)^p I(x, y) dx dy \tag{1.11}$$

$$E_o = \frac{1}{2} arctan(\frac{2m_{11}}{m_{20} - m_{02}}) \tag{1.12}$$

$$J = \begin{bmatrix} m_{20} & m_{11} \\ m_{11} & m_{02} \end{bmatrix} \tag{1.13}$$

Les valeurs propres de la matrice de covariance J sont les moments d'inertie I_{min} et I_{max} de l'ellipse. Il permettent de déterminer les axes principaux de l'ellipse E_a et E_b :

$$E_a = (\frac{4}{\pi})^{1/4} \left[\frac{(I_{max})^3}{I_{min}} \right]^{1/8} \tag{1.14}$$

$$E_b = (\frac{4}{\pi})^{1/4} \left[\frac{(I_{min})^3}{I_{max}} \right]^{1/8} \tag{1.15}$$

1.2.2.4/ LES PROJECTIONS DES HISTOGRAMMES

La répartition statistique des pixels en mouvement, dans la boite englobante, suivant les lignes et les colonnes, représente une information qui peut être caractéristique de la posture, donc de la chute. On détermine donc l'histogramme correspondant vertical Vp (resp. horizontal Hp) suivant les lignes (resp. colonnes), à partir du nombre des pixels blancs sur chacune des lignes (resp. colonnes) de l'image (voir figure 1.5). Ces grandeurs sont utilisées pour les méthodes de détections de chutes par Khan [73], Chen [24], Foroughi [44] et Cucchiara [28]. Certains auteurs utilisent l'intégralité des informations des histogrammes, d'autres n'utilisent que la valeur maximum, normalisée par rapport à la taille

de la boite englobante.

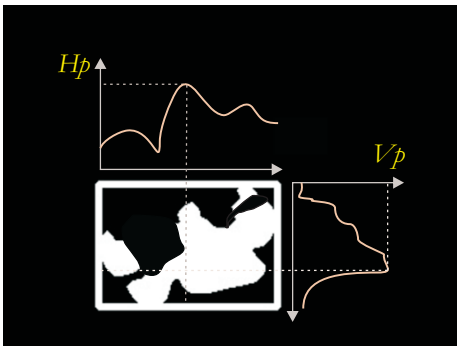


FIGURE 1.5 – Projections verticale et horizontale des histogrammes.

1.3/ ANALYSE TEMPORELLE POUR LA DÉTECTION D'ACTIONS

Nous venons de voir que le temps est déjà pris en compte lors de l'analyse du mouvement pour obtenir la détection de la personne. Il est toutefois parfois nécessaire d'en tenir compte à un niveau supérieur, c'est à dire d'exploiter la variation temporelle des informations spatiales extraites précédemment. L'analyse temporelle pour la reconnaissance d'actions s'effectue alors à partir d'un groupe d'images successives. Foroughi [45] a proposé d'analyser la variation temporelle des attributs pour détecter les chutes.

L'analyse des trajectoires est une méthode qui assure la reconnaissance d'actions abordée dans plusieurs travaux de recherche dans lesquels le problème de reconnaissance devient un problème de classification de trajectoires [68, 107]. On définit une trajectoire par les coordonnées spatiales au cours du temps, obtenues souvent grâce à un algorithme de suivi. Cette analyse temporelle semble la plus directe mais n'est pas forcément la plus simple puisque les performances de reconnaissance dépendent aussi de l'algorithme de suivi. En effet, la segmentation des pixels en mouvement donnant souvent plusieurs régions, il est nécessaire parfois de les fusionner, et surtout d'établir une correspondance entre la région d'une image à l'instant t et la même région, qui s'est déplacée dans l'image, à l'instant $t + 1$.

Parmi les méthodes de suivi les plus utilisées en traitement d'image, se trouvent les méthodes basées sur le filtre de Kalman et filtre particulaire, dont nous rappellerons brièvement les principes et applications en détection de chute ci-dessous.

Une fois la trajectoire obtenue grâce au suivi ou à la détection de la personne à chaque image, il est possible de l'analyser pour par exemple reconnaître les actions : nous présenterons un exemple de ce type d'application.

Enfin, les variations temporelles des signaux peuvent encore être mises en évidence à l'aide d'outils de base telles que la dérivée première (analyse de la vitesse) et la dérivée

seconde (accélération), et surtout d'autres outils fondamentaux d'analyse du signal que sont les transformées de Fourier et transformées en ondelettes. Largement utilisées en reconnaissance d'objet, elles le sont aussi en analyse vocale (reconnaissance du locuteur), analyse de signaux médicaux, etc [123, 110, 111, 122, 13]. Il nous a paru donc important de rappeler les bases de ces méthodes temps-fréquence, que nous utiliserons d'ailleurs dans la mise en place de notre propre méthode.

1.3.1/ LE SUIVI DE PERSONNE : MÉTHODES COURAMMENT EMPLOYÉES

1.3.1.1/ LE FILTRE DE KALMAN

Soit un système dynamique composé de deux vecteurs aléatoire X et Z, où X est le processus d'état, représentant ce que l'on cherche et Z le vecteur d'observation. Dans le cadre du suivi de personnes X peut représenter l'étiquette du rectangle en mouvement, et Z les différentes mesures de position, surface de boite englobantes, etc. Le but de méthodes dites directes est de trouver la transformation $Z = f(X)$ qui permet de faire correspondre une observation avec une prédiction de l'état X. Les méthodes probabilistes cherchent à estimer la loi de probabilité *a posteriori* de cet état.

Le filtre de Kalman [71] est un des moyens permettant d'arriver à cette estimation, par le biais d'un filtre récursif basé d'abord d'une phase de prédiction, puis d'une phase de correction.

La phase de prédiction consiste à estimer un état X_t à un instant courant t sachant l'estimation X_{t-1} de l'état précédent. La phase de correction permet ensuite de corriger l'état prédit à l'aide de la mesure actuelle.

Le modèle de Kalman est basé sur l'hypothèse que l'état réel X au temps t dépend de l'état précédent $t-1$ suivant l'équation :

$$X_t = F_t X_{t-1} + B_t u_t + w_t \tag{1.16}$$

où
– la matrice F_t est la matrice de transition appliquée à l'état précédent,
– u_t sont des entrées de commande,
– B_t : modèle de contrôle appliqué à l'entrée u_t ,
– w_t est le bruit d'état, de covariances Q_t.
A l'instant t, une observation est réalisée, selon l'équation :

$$Z_t = H_t X_t + D_t u_t + v_t \tag{1.17}$$

où

– la matrice H_t est le modèle qui permet de passer de l'état réel à l'espace d'observation,
– D_t : entrée du système connue ,
– v_t est le bruit de mesure pour de covariances R_t.
Le problème de filtrage se ramène à la résolution d'un système linéaire récursif formé par les équations 1.16 et 1.17.

si X_0, l'état initial, est gaussien, d'espérance $\hat{X_0}$ et de covariance Σ_0, le système peut alors s'écrire de la façon suivante :

$$\begin{cases} X_0 \sim N(X_0; \hat{X}_0, \Sigma_0) \\ X_t|X_{t-1} \sim N(X_t; F_t X_{t-1} + B_t u_t, Q_t) \\ Z_t|X_t \sim N(Z_t; H_t X_t + D_t u_t, R_t) \end{cases} \tag{1.18}$$

$N(X; \hat{X}, \Sigma)$ désigne la loi normale de la variable X, d'espérance $\hat{X}$ et de covariance Σ. Le processus $\{X_t, Z_t\}$ est gaussien et la loi de filtrage recherchée $p(X_t|Z_{1:t})$ est gaussienne. Cette loi est donc décrite par son espérance $\hat{X_{t|t}} = \mathbb{E}[X_t|Z_{1:t}]$ et sa covariance $\hat{\Sigma}_{t|t} = \mathbb{E}[(X_t - \hat{X}_{t|t})(X_t - \hat{X}_{t|t})^t]$.

Le cycle de Kalman se déroule donc en deux étapes récursives :

– une étape de prédiction qui consiste à estimer l'état courant à partir de l'estimation précédente en calculant $\hat{X}_{t|t-1}$ et la matrice de covariance de l'erreur d'estimation $\hat{\Sigma}_{t|t-1}$.
– une étape de correction dite mise à jour qui consiste à corriger la prédiction en se servant de l'observation de l'état courant ; ceci revient à calculer la matrice de covariance de l'innovation S_t qui indique l'écart entre la prédiction et la mesure observée, calculer le gain du filtre de Kalman G_t qui doit tenir compte des incertitudes relatives à l'estimation courante et enfin l'estimation de la mise à jour de l'état $\hat{X}_{t|t}$ et de la matrice de covariance $\hat{\Sigma}_{t|t}$.

Des extensions de ce filtre ont été conçues pour les modèles non linéaires [70][4].

Appliqué pour le suivi de personnes [82, 125, 65, 25] et de véhicules [69], le filtre de Kalman a montré de bonnes performances. Néanmoins, ce filtre n'est pas aussi efficace lorsqu'il s'agit de trajectoires aléatoires de personnes dans le cas d'occlusions ou de changement brusque de direction, c'est à dire, lorsqu'il s'agit d'un système dont le modèle d'évolution est non linéaire [116].

1.3.1.2/ LE FILTRE PARTICULAIRE

Le filtrage particulaire est une technique permettant d'estimer et prédire des systèmes dynamiques, utilisés notamment lorsque ce système est non linéaire et/ou non gaussien. Il est basé sur le principe de simulation de Monte Carlo, et a pour but d'estimer récursivement la densité de probabilité *à posteriori* $p(X_t|Z_{1:t})$ du vecteur d'état X_t à l'instant t conditionné sur l'ensemble de mesures $Z_{1:t} = Z_1, ..., Z_t$. A chaque instant t, on ap-

proxime la densité $p(X_t|Z_{1:t})$, grâce à un ensemble de "particules" X_t^i qui se déplacent selon des réalisations indépendantes à partir de l'équation d'état. Ces particules seront corrigées en fonction de leur cohérence avec les observations, grâce à un poids qui leur est affecté, $w_t^i, i = 1, ..., N$. Grâce à cette correction, les particules évoluent de manière stochastique dans le temps et sont échantillonnées selon une fonction d'importance afin d'explorer les zones pertinentes de l'espace d'état. Une particule d'indice i est représentée par le couplet $\{X_{0\to t}^i, w_t^i\}$.

En ce qui concerne le suivi de personne, le vecteur d'état estimé que propose Brèthes [22] est constitué des coordonnées de la position de la personne, son facteur d'échelle et son orientation.

L'initialisation du filtrage particulaire consiste à définir un ensemble de particules pondérées décrivant la distribution à priori $p(X_0)$ où X_0 est un vecteur d'état initial, par exemple, en affectant des poids identiques $w_0^i = 1/N$ aux échantillons $X_0, ..., X_0^N$ indépendants identiquement distribués selon $p(X_0)$. A chaque instant t, la détermination de l'ensemble de particules pondérées $\{X_t^i, w_t^i\}$ associée à $p(X_t|Z_{1:t})$ se fait en deux étapes (connaissant $\{X_{t-1}^i, w_{t-1}^i\}$ associée à $p(Xt-1|Z_{1:t-1})$) : une étape de prédiction et une étape de correction. La loi de densité prédite est approximée par :

$$p(X_t|Z_{1:t-1}) \approx \sum_{i=1}^{N} w_{t|t-1}^i \delta(X_t = X_{t|t-1}^i) \tag{1.19}$$

où $X_{t|t-1}^i$ sont obtenues par des réalisations indépendantes de la loi de transition $p(X_t|X_{t-1}^i)$ et $w_{t|t-1}^i = w_{t-1}^i$. Pour la correction, on passe à la loi de densité conditionnelle grâce à la vraisemblance $g(Z_t - H_t(X_{t|t-1}^i))$:

$$p(X_t|Z_t) \approx \sum_{i=1}^{N} w_t^i \delta(X_t = X_t^i) \tag{1.20}$$

avec

$$w_t^i = \frac{w_{t|t-1}^i g(Z_t - H_t(X_{t|t-1}^i))}{\sum_{j=1}^{N} w_{t|t-1}^j g(Z_t - H_t(X_{t|t-1}^j)} \tag{1.21}$$

Il existe de nombreuses variantes permettant l'implantation de ce filtre sous forme récursive, dont certaines sont finalement assez proches de l'algorithme d'Adaboost utilisé en classification (voir section 1.4.2).

Le filtrage particulaire est bien adapté pour le cas des trajectoires complexes (lors d'un changement brusque de direction, par exemple) ou bien le cas de multiples personnes dans la scène. Son efficacité est corrélée principalement au nombre de particules utilisées qui doit être suffisamment grand. Ceci augmente la complexité temporelle de la méthode et le coût calculatoire important qui est inadapté pour les ressources matérielles que nous envisage utiliser lors de l'implantation de notre méthode. Rougier [119]

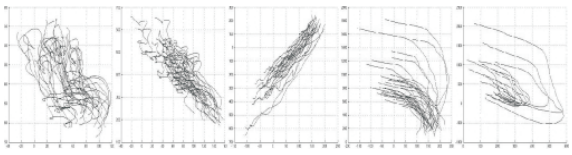
et Nait-Charif [103] se sont contentés de suivre la tête de la personne afin de diminuer le nombre de particules analysées pour mettre en place un système de détection de chute temps réel.

1.3.2/ ANALYSE DES TRAJECTOIRES

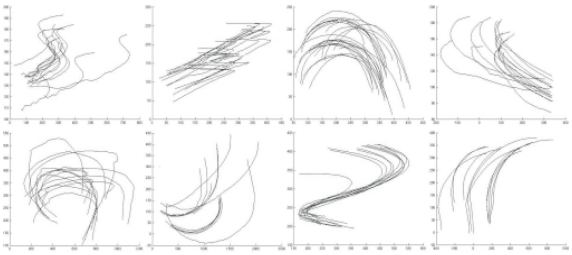
Très utilisée dans les systèmes de vidéosurveillance dans le but de décrire l'activité d'une personne, la trajectoire de toute ou d'une partie partie de la personne dans la scène est une riche source d'informations à la fois spatiales et temporelles qui aide à la discrimination des actions. Par exemple, Nait-Charif [103] et Rougier [119] ont analysé la trajectoire de la tête obtenue par filtrage particulaire pour distinguer les chutes des autres activités.

Elle peut être extraite par l'intermédiaire d'une ou plusieurs caméras. Souvent représentée par les coordonnées spatiales de la personne en mouvement dans l'image, une trajectoire peut aussi être représentée en termes de vitesse et de forme, constituant alors un vecteur caractéristique de l'action [59].

Pour classer la trajectoire, acquise en général à partir des algorithmes de détection et de suivis décrits précédemment, la plupart des auteurs utilisent une mesure de distance ou ressemblance entre les vecteurs descripteurs de trajectoires [153]. Ainsi, Kim [74] propose une méthode de détection d'anomalies pour la sécurité routière basée sur ce principe. Jiang [68] a utilisé la trajectoire du mouvement pour la reconnaissance d'actions humaines. Luisier et Hervieu [33, 58], ont quand à eux appliqué cette méthode à la reconnaissance de scènes de sport. Un exemple de telles trajectoires acquises par Hervieu est représenté figure 1.6.



(a) Tracé de 5 classes de 134 trajectoires de vidéo de ski : slalom et descente.



(b) Tracé de 8 classes de 125 trajectoires de Formule 1.

Chaque classe est composée de trajectoires extraites de vidéos d'une même caméra.

FIGURE 1.6 – Exemples de classes de trajectoires des scène de sport [58]

D'autres auteurs [112, 154, 133] basent l'analyse et la classification de la trajectoire sur le principe des modèles de Markov cachés. Ces modèles, largement utilisés par exemple pour la reconnaissance d'écriture, s'appliquent parfaitement au cas plus général de l'analyse des trajectoires relatives à des actions humaines. Cette analyse, qui se traduit par l'étude de l'évolution de la position et des propriétés dynamiques du système, est souvent réalisée à partir d'une matrice de transitions d'états, donc de MMC. Nous allons donc en rappeler succinctement le principe.

Principe des MMC Connus aussi par HMM [5] dans la littérature anglophone, les modèles de Markov cachés sont nés de la théorie probabiliste à états finis à travers l'hypothèse Markovienne qui énonce qu'un état futur (à l'instant $t+1$) ne dépend que de l'état actuel (à l'instant t) et une observation courante O_t ne dépend que de l'état courant.

Un exemple à trois états est présenté figure 1.7 sachant qu'un modèle de Markov λ se compose de :

- N états cachés :
  $\mathsf{S} = \{\mathsf{S}_1, \mathsf{S}_2, ..., \mathsf{S}_N\}$,
- M observations distinctes dites états visibles :
  $V = \{v_1, v_2, ..., v_M\}$,
  où chaque variable aléatoire x_k associée à l'état S_k prend ses valeurs dans V,
- des probabilités de transitions entre les états $\mathsf{S}_i(t)$ et $\mathsf{S}_j(t+1)$:
  $\mathbf{A} = [a_{i,j}]$ où $a_{i,j} = P(\mathsf{S}_j(t+1)|\mathsf{S}_i(t))$, $1 \leqslant i \leqslant N, 1 \leqslant j \leqslant N$,
- Des probabilités d'émission des observations pour chaque état :
  $b_{jk} = P(v_k(t)|\mathsf{S}_j(t))$, $1 \leqslant k \leqslant M$.

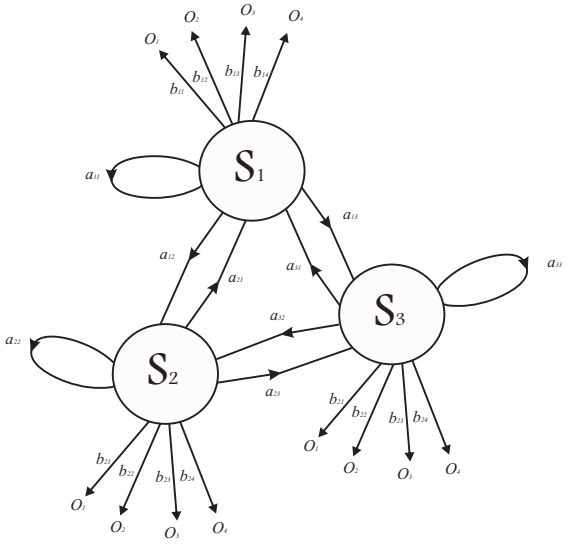


FIGURE 1.7 – Exemple de HMM à 3 états.

5. Hidden Markov Models

Une étape d'apprentissage permet ensuite de déterminer les paramètres du modèle λ qui dépend de la matrice de transition A et des probabilités d'émission à partir de la séquence d'observations. Les algorithmes les plus connus permettant de remplir cet objectif sont : Viterbi [43] et *Forward-Backward* connu aussi sous le nom de Baum-Welch [12].

Les MMC sont parfois utilisés pour analyser directement la suite de positions de la personne, donc la trajectoire [112, 154, 133, 59], et parfois pour analyser la succession des actions en tant que tel. En effet, une chute représente un état qui est souvent la suite logique d'un autre état comme un début de transfert ou une position assise. Ainsi, dans [49], les MMCs sont utilisés pour le contrôle de la prise de repas d'une personne âgée et Toreyin [134], Cucchiara [28], Anderson [3] et Thome [132] proposent des solutions pour la détection de chutes.

La principale difficulté rencontrée par les méthodes d'analyse de trajectoires concerne les fortes occultations où lorsque la personne quitte le champs de la caméra pour une longue période. Les processus de suivi et analyse de trajectoires sont alors complètement interrompus.

1.3.3/ LES MÉTHODES TEMPS-FRÉQUENCE

Parmi les nombreux outils mathématiques d'analyse du signal, nous avons étudié deux transformations largement utilisées en reconnaissance de formes : la transformée de Fourier (ici à fenêtre glissante) et la transformée en ondelettes. Ces transformées peuvent être utilisées pour analyser des variations spatiales, dans le cas par exemple de la reconnaissance d'objets basée sur les Descripteurs de Fourier Généralisés [126], ou temporelles, en analyse de la parole ou analyse de signaux médicaux [109]. En ce qui concerne la détection de chutes, ces transformations peuvent être appliquées à l'analyse des trajectoires ou plus généralement à l'analyse de n'importe quel signal provenant de l'extraction des attributs géométriques mentionnés précédemment.

La transformée de Fourier est la plus connue des méthodes fréquentielles à partir de laquelle plusieurs méthodologies de reconnaissance ou de détection ont été développées. Elle fournit une représentation fréquentielle unique et simple à interpréter. Ces informations de fréquences étant, dans de nombreux cas pratiques, des caractéristiques pertinentes pour classer des parties de signaux, la transformée de Fourier et ses variantes sont fréquemment utilisées en reconnaissance de formes. Heeger [57] et Speini [130] ont par exemple estimé le flux optique, et donc étudié les vitesses d'objets en mouvement, par l'intermédiaire de filtres spatio-temporels basés sur des variantes de cette transformée. L'usage de la transformée de Fourier a fait encore ses preuves dans différentes applications telles que la reconnaissance d'objets 2D [127] ou encore la détection de cancer dans le travail de Parfait [109].

Afin de pallier à certains inconvénients de la transformée de Fourier (perte de d'informa-

tions de localisation notamment), la transformée en ondelette dont l'un des fondateurs est Meyer [94] a été introduite au XX^e siècle. Une transformée en ondelettes diffère d'une transformation de Fourier par le fait qu'elle est capable de fournir une représentation à la fois fréquentielle et temporelle d'un signal.

Ces deux transformées ont fait l'objet de nombreux ouvrages : nous n'en rappellerons donc ici que les principes de base.

– La transformée de Fourier est un cas particulier de la transformée de Laplace, qui est plus utilisée par les automaticiens, en particulier pour caractériser la stabilité des systèmes linéaires.
Considérons une fonction $f(x)$ définie et à valeurs dans $\mathbb{R}$, la transformée de Fourier (notée $\hat{f}$) de $f(x)$ est définie par :

$$\forall \alpha \in \mathbb{R}, \hat{f}(\alpha) = \int_{-\infty}^{+\infty} f(x) e^{-i\alpha x} dx \tag{1.22}$$

L'un des algorithmes de calcul de la transformée de Fourier discrète les plus utilisés est celui de Cooley Tukey [26] appelée Fast Fourier Transform (FFT). Il est souvent utilisé pour les implantations temps réel des applications de traitement d'image. L'application de cette transformée est aussi applicable aux signaux 2D, notamment à une image. Il suffit d'appliquer deux transformées successives dans les deux directions. Toutefois, nous ne nous intéresserons qu'aux signaux mono-dimensionnels puisque nous utiliserons pour notre méthode des attributs temporels 1D.

– La transformée en ondelettes : d'un point de vue général et sur les signaux continus, on définit une famille de fonctions basées sur l'ondelette dite mère :

$$\forall t \in \mathbb{R}, \psi_{s,\tau}(t) = \frac{1}{\sqrt{s}} \Psi(\frac{t-\tau}{s}) \tag{1.23}$$

La transformée en ondelettes elle même est alors définie par :

$$g(s,\tau) = \int_{-\infty}^{\infty} f(t) \psi_{s,\tau}(t)^* dt \tag{1.24}$$

Où $*$ désigne le complexe conjugué, τ le facteur de translation et s est un facteur de dilatation. Cette transformée permet d'accomplir une analyse à la fois locale et globale d'un signal et donc d'une forme représentée par ce signal. De nombreuses formes d'ondelettes mères ont été proposées dans la littérature, conduisant chacune à des propriétés spécifiques d'analyse du signal.
Par la suite, Daubechies [31] a mis au point les ondelettes orthogonales qui portent son nom. Elles ont l'avantage d'être aisément implantables et rapides à calculer. Elles sont largement utilisée dans la littérature pour l'analyse des signaux, ainsi que pour la compression dans la norme JPEG2000 [150, 54]. Nous avons donc utilisé par la suite les ondelettes orthogonales de Daubechies dites D_4, c'est à dire à 4 coefficients.

On trouvera une description complète de cette famille et du calcul des coefficients par exemple dans [113].

1.4/ LES MÉTHODES DE CLASSIFICATION

La dernière étape de la méthode de détection de chute (ou de reconnaissance d'action) est donc la prise de décision finale. Les classes sont en général définies selon l'application (nom de l'action pour la reconnaissance d'actions, nom d'objet dans le cas de reconnaissance d'objet, etc.). Dans notre cas, l'action en cours doit être classée en "chute" ou en "non-chute".

Il existe de très nombreuses méthodes permettant de réaliser cette phase, aux performances et complexité d'implantation variables. Le choix de la méthode de classification qui consiste à attribuer une étiquette adéquate (classe) aux attributs précédemment extraits est donc important.

Dans le cas idéal, deux descripteurs qui représentent deux échantillons d'une même classe devraient être identiques. Or, ceci n'est pas vérifié dans la réalité et on a souvent affaire, pour ces deux échantillons, à des descripteurs différents. Un bon classifieur doit être capable de distinguer des classes différentes alors que les descripteurs correspondant ne sont pas parfaitement discriminants : il doit donc minimiser l'erreur de classification tout en possédant un bon pouvoir de généralisation. En ce qui concerne les méthodes supervisées, ce pouvoir est caractérisé par l'aptitude du classifieur à bien classer des échantillons très différents des échantillons utilisés pendant la phase d'apprentissage.

Des ouvrages entiers sont consacrés à ces méthodes, les références les plus connues étant par exemple Duda et Hart [35], Bishop [16]. Nous ne présenterons ici que les principes généraux, suivis des bases de deux méthodes particulièrement employées actuellement, à savoir les "Support Vector Machine" (SVM) et les méthodes de type "Boosting".

Ces méthodes possèdent un point commun : la nécessité de réaliser un apprentissage et d'évaluer leurs performances. Elles nécessitent donc deux ensembles d'échantillons :

1. Base d'apprentissage : notée L doit contenir un nombre suffisant d'échantillons $\mathbf{x_i}$ de chaque classe. Cet ensemble doit être le plus représentatif possible d'un point de vue qualitatif (en vue de l'application visée) et quantitatif (un nombre important d'échantillons tout en respectant les probabilités à priori de chacune des classes).
2. Base de test : notée T doit aussi être choisie de manière rigoureuse puisqu'elle permet d'évaluer l'algorithme.

Classiquement, le processus se déroule en deux phase principales : apprentissage (hors-ligne) et décision (en-ligne) comme illustré dans la Figure 1.8. Le module d'apprentissage permet, lorsqu'il est supervisé, c'est à dire à partir de la connaissance à priori de la classe

des échantillons fournie par un expert, de construire un modèle (ensemble de règles ou critères de décision). Ce modèle est ensuite utilisé par le classifieur pour fournir la décision concernant l'appartenance d'un échantillon inconnu, représenté par ses attributs, à l'une des classes prédéfinies.

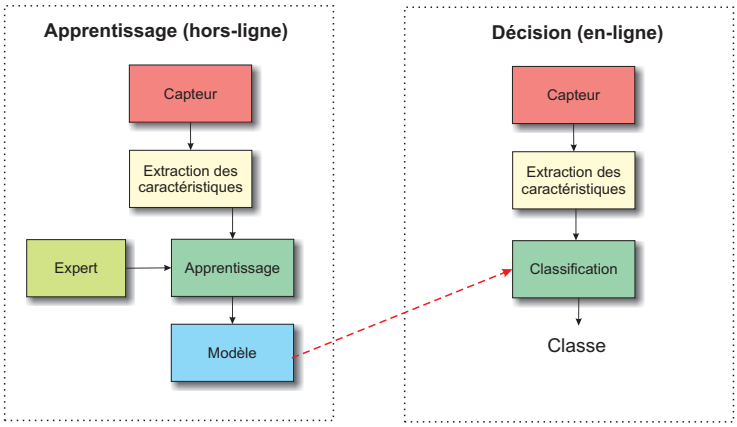


FIGURE 1.8 – Processus classique de reconnaissance par apprentissage.

D'une manière générale, la classification consiste à affecter une classe à un vecteur description de l'objet $\mathbf{x}$ de l'espace de description $\mathbb{R}^d$ (d est le nombre d'attributs de $\mathbf{x}$) tout en minimisant l'erreur de classification appelée "risque attendu" ou tout simplement "risque" définie à partir de la mesure de divergence $L(y, f(\mathbf{x}, \alpha))$ entre la réponse y qu'on souhaite obtenir et l'étiquette $f(\mathbf{x}, \alpha)$ attribuée par la machine d'apprentissage pour $\mathbf{x}$. Donc l'apprentissage permet de trouver, lorsque la fonction de distribution de probabilité jointe $P(\mathbf{x}, y)$ est inconnue, la fonction $f(\mathbf{x}, \alpha)$ qui minimise la fonction de risque suivante :

$$R(\alpha) = \int L(y, f(\mathbf{x}, \alpha)) dP(\mathbf{x}, y) \tag{1.25}$$

où α représente un vecteur de paramètres de la méthode de classification.

Suivant que la classe des échantillons d'apprentissage est connue ou non, on obtient deux grandes familles de méthodes de classification et par conséquent deux principales méthodes d'apprentissage :

– **Apprentissage supervisé :** Dans ce cas, un expert intervient pour attribuer au préalable et sans ambiguïté la classe d'appartenance à chaque objet de l'ensemble d'apprentissage L. Le rôle du classifieur par la suite est de prédire la classe ω_i de chaque prototype $\mathbf{x_i}$ grâce à l'estimation de la probabilité d'apparition de la classe ω_i pour $\mathbf{x_i}$ notée $P(\omega_i \mid \mathbf{x_i})$. Autrement dit, pour une machine devant classer un certain nombre d'échantillons d'un ensemble D où chaque échantillon est composé d'un doublet $\{\mathbf{x_i}, y_i\}$ avec $\mathbf{x_i}$ un vecteur de l'espace de description et y_i l'étiquette de la classe correspondante (i.e. $\mathbf{x_i} \in \omega_\mathbf{i}$ avec $j = y_i$), on cherche à définir la loi de probabilité $P(\mathbf{x}, y)$ à qui pour tout $\mathbf{x}$ donne la probabilité d'appartenance à la classe y. Parmis les méthodes

les plus couramment utilisées dans le cadre des systèmes de reconnaissances, on trouve les SVM, le réseaux de neurones et le Boosting.

– **Apprentissage non-supervisé :** Dans ce cas, il n'existe pas d'expert pour affecter une étiquette à chaque échantillon ce qui rend l'estimation de probabilité inapplicable du moment où aucune information concernant l'appartenance d'un objet à une classe n'est accessible. Le principe de ce type d'apprentissage est de découper l'espace de représentation des échantillons en parties homogènes dites clusters par le biais d'agrégations. La principale difficulté réside dans la détermination de ce critère d'homogénéité. Une des méthodes les plus couramment utilisées est l'algorithme des K-means [35].

A moins de disposer d'attributs très discriminants, les approches supervisées donnent en général de meilleurs résultats que les approches non supervisées. La plupart des auteurs, souhaitant optimiser les performances de détection et pouvant accéder à des vidéos d'apprentissages annotées par un expert, choisissent donc des méthodes supervisées. Ces méthodes sont très connues et ont permis de résoudre de nombreux problèmes dans des applications de reconnaissance de formes. Le Boosting a été largement utilisé en biométrie, pour la localisation et la détection de visages, la détection de défauts sur des objets manufacturés en vision industrielle [97] et l'imagerie médicale [109]. Les SVM sont souvent plus performants en termes d'erreur de classification, alors que les algorithmes dérivés du Bossting conviennent mieux à l'implantation matérielle [98], qui représente un critère crucial pour une application temps réel embarquée de détection de chute.

Nous détaillerons dans ce qui suit les deux méthodes de classification supervisées que nous avons utilisées pour notre application : SVM et Adaboost.

1.4.1/ Les Support Vector Machine

Les SVM ont été introduits par Vladimir Vapnik en 1995 [136]. Ils ont été largement utilisés depuis, pour résoudre les problèmes de reconnaissance de formes, de régression et d'estimation de densité de probabilité. Nous avons vu que pour résoudre un problème par apprentissage, nous cherchons à classer correctement les observations grâces aux connaissances apprises en se basant sur la minimisation du "risque empirique" (réalisation du risque défini équation 1.25). Si n est le nombre d'échantillons, ce "risque empirique" est défini par :

$$R_{emp}(\alpha) = \frac{1}{2p} \sum_{i=1}^{n} |y_i - f(x_i, \alpha)| \tag{1.26}$$

Le principe des SVM est de déterminer, afin de minimiser le risque empirique, un hyperplan séparateur entre des échantillons appartenant à deux classes distinctes en maximi-

sant la marge inter-classes, d'où le terme "Séparateur à Vastes Marges" que l'on trouve parfois dans la littérature francophone. Les échantillons qui servent à déterminer l'hyperplan séparateur sont appelées les vecteurs support.

Cette méthode de classification puissante est largement utilisée dans les applications de reconnaissance de formes, notamment en reconnaissance d'actions [141, 62, 30, 34] et en reconnaissance de gestes faciaux comme dans les travaux de Bartlett [11] et Jiang [67]. En détection de chutes, Foroughi [44], Qian [114] et Jiang [68] ont utilisé avec succès les SVM.

Cas linéairement séparable Pour deux classes d'échantillons linéairement séparables, chaque échantillons de l'ensemble d'apprentissage L étant représenté par le couplet $\{x_i, y_i\}, x_i \in \mathbb{R}^d, y_i \in \{-1, 1\}$. A chaque échantillon x_i est donc associée une étiquette y_i comme illustré dans la Figure 1.9. Notons w la normale à l'hyperplan qui sépare les deux classes et qui a pour équation :

$$\langle w, x \rangle + b = 0 \tag{1.27}$$

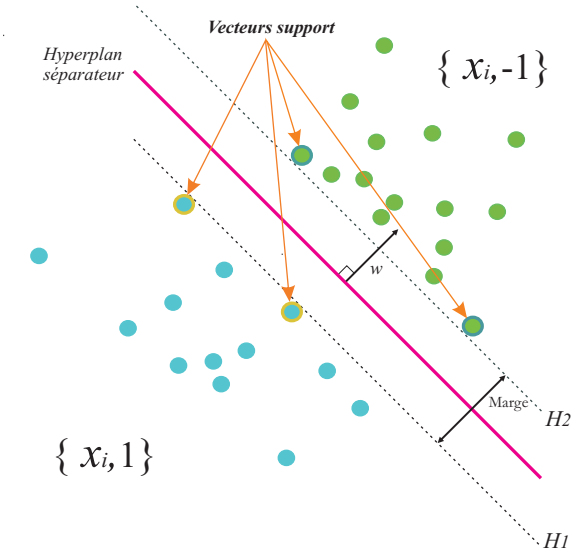


FIGURE 1.9 – Principe des SVM.

Il existe une infinité d'hyperplans qui peuvent assurer la séparation entre les deux classes. La méthode d'apprentissage des SVM va donc consister à sélectionner un seul hyperplan "optimum". La figure 1.10 illustre 3 cas possibles d'hyperplans séparateur. L'hyperplan du cas (c) est optimal comparé aux cas (a) et (b), dans le sens où il maximise la marge entre lui-même et les hyperplans parallèles, H_1 et H_2, situés au bord des classes, représentés en gris sur la figure. Il s'agit donc de celui qui maximise la distance minimale entre les échantillons des deux classes et lui même, ce qui, du même coup, minimise le "risque

empirique" donné par l'équation 1.26.

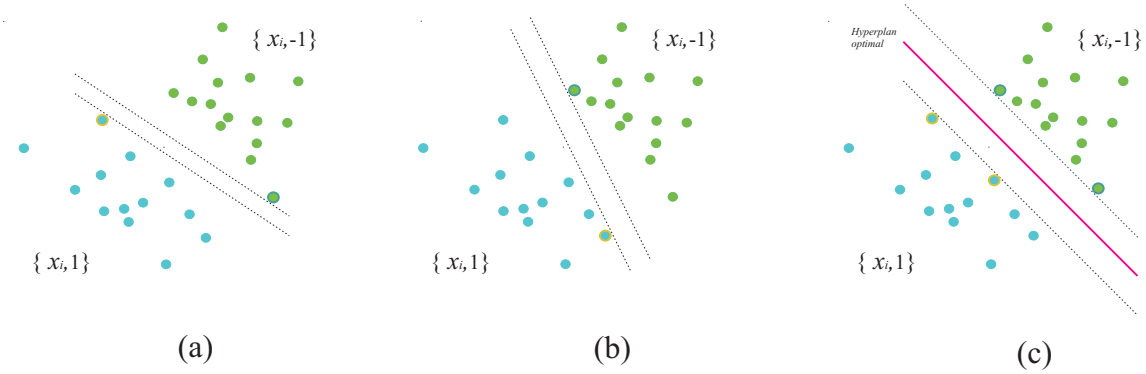


FIGURE 1.10 – Recherche de l'hyperplan qui maximise la marge.

Les deux hyperplans H_1 et H_2 ont pour équations : $H_1 : \langle w, x\rangle + b = 1$ et $H_2 : \langle w, x\rangle + b = -1$. Ils vérifient l'équation 1.28 qui signifie que tous les points d'une même classe sont du même côté de l'hyperplan.

$$\forall i \in [1..n]\,, y_i(\langle w, x_i\rangle + b) \geq 1 \tag{1.28}$$

Maximiser la distance qui sépare ces deux hyperplans (la "Marge") revient à maximiser le quotient $\frac{2}{\|w\|}$ où $\|...\|$ est la norme euclidienne. Trouver l'hyperplan optimum devient donc la minimisation de $\|w\|$ qui est un problème d'optimisation quadratique. Ce type d'optimisation peut être résolu grâce à la méthode de Lagrange. La quantité à minimiser sous contraintes devient :

$$L(w, b, \alpha) = \frac{1}{2}\|w\|^2 - \sum_1^n \alpha_i y_i(\langle w_i, x_i\rangle + b) + \sum_1^n \alpha_i,\ \alpha_i \geqslant 0 \tag{1.29}$$

où α_i sont les multiplicateurs de Lagrange. En annulant les dérivées partielles de $L(w, b, \alpha)$ par rapport à chacune des composantes w_k du vecteur w (pour $k = 1, ..., d$) et par rapport b, on obtient le système d'équations suivant :

$$\begin{cases} \sum_1^n \alpha_i y_i x_{ik} = w_k \qquad pour \qquad k = 1, ..., d \\ \sum_1^n \alpha_i y_i = 0 \end{cases} \tag{1.30}$$

Il est possible de résoudre le problème à partir du dual de Wolf suivant :

$$Ł(w, b, \alpha) = \sum_{i=1}^n \alpha_i - \frac{1}{2}\sum_{i,j=1}^j \alpha_i \alpha_j y_i y_j \langle x_i, x_j\rangle \tag{1.31}$$

$Ł(w, b, \alpha)$ est à maximiser en respectant les contraintes de l'équation 1.30 et par la suite b est facile à en déduire. La décision de la classification y est donnée alors par seuillage de la fonction de décision $f(x)$:

$$y = signe(f(x)) = signe(\langle w, x\rangle + b) = signe(\sum_{i=1}^{n} \alpha_i y_i \langle x_i, x\rangle + b) \tag{1.32}$$

Les échantillons de l'apprentissage pour lesquels la valeur de α_i est non nulle sont appelés vecteur support. Les autres échantillons ne participent pas à la décision finale (voir la Figure 1.9). Les vecteurs support sont notés s_i et sont au nombre de N_v. La fonction de décision devient :

$$f(x) = \sum_{i=1}^{N_v} \alpha_i y_i \langle s_i, x\rangle + b \tag{1.33}$$

Cette équation permet de classer tout échantillon selon sa position, dans l'espace de description, par rapport à ce plan.

Cas non linéairement séparable Le cas linéairement séparable est un cas très optimiste, la séparation linéaire s'avère très difficile dans la plupart des cas puisqu'elle exige l'existence d'une frontière linéaire pour séparer les données sans la moindre erreur. La figure 1.11 montre des exemples de classes non linéairement séparables. La notion de la marge souple donc a été introduite, pour permettre de tolérer des erreurs de classification.

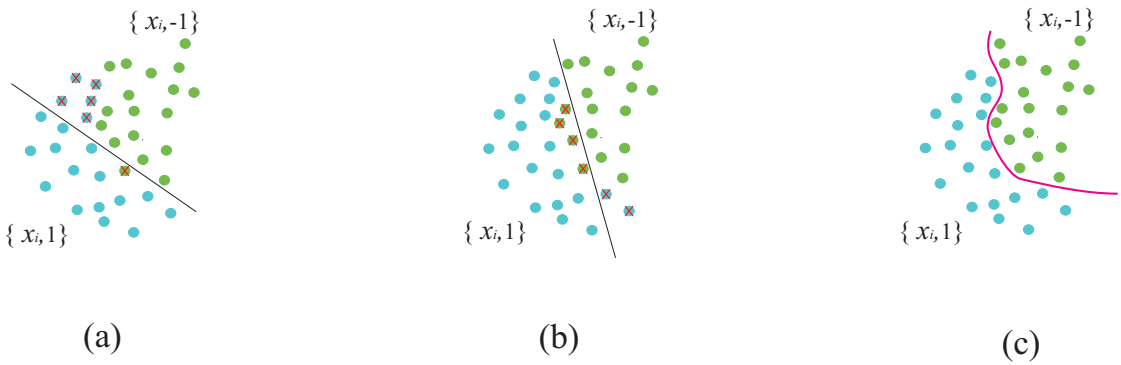


FIGURE 1.11 – Exemple de deux classes non séparables linéairement.

La marge d'erreur est notée ς_i et l'équation 1.28 devient :

$$\forall i \in [1..n], y_i(\langle w, x_i\rangle + b) \geq 1 - \varsigma_i \tag{1.34}$$

La quantité à minimiser devient :

$$\frac{1}{2}\|w\|^2 + C\sum_{1}^{n}\varsigma_i \tag{1.35}$$

où C est le coût affecté à un échantillon mal classé. La Figure 1.12 présente un exemple d'une séparation linéaire mais décalée en fonction de ce coût C à cause d'un élément qu'on appelle "outlier" se trouvant du mauvais côté de la frontière. Ici, la fonction de décision est toujours celle donnée par l'équation 1.33. Elle dépend donc toujours d'une frontière linéaire (hyperplan). Or ce type de frontière s'avère insuffisant pour séparer les classes de manière efficace dans un grand nombre de cas. Une transformation de l'espace de description permet d'obtenir des frontières non linéaires tout en conservant le reste de la méthodologie de résolution identique.

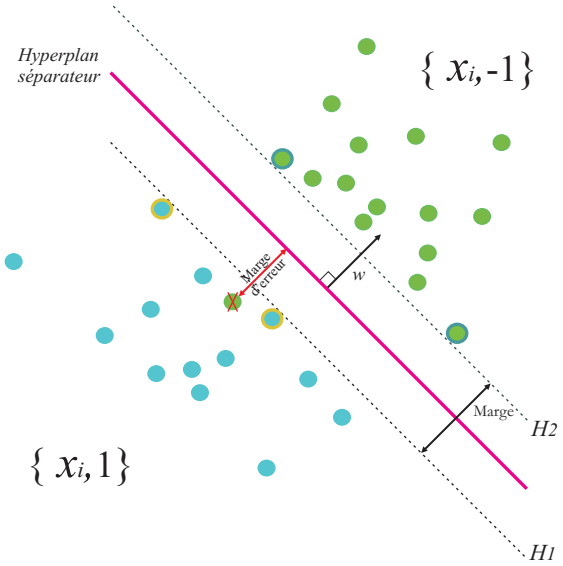


FIGURE 1.12 – Introduction de la tolérance aux erreurs de classification.

Frontière de décision non linéaire Le principe est donc de séparer linéairement les données dans un espace $\mathbb{R}^e$ transformé à partir de l'espace de description d'origine $\mathbb{R}^d$. La dimension e de l'espace d'arrivée est généralement supérieure à la dimension de l'espace de départ. La fonction de passage de $\mathbb{R}^d$ dans $\mathbb{R}^e$ est réalisée à l'aide d'une fonction Φ :

$$\begin{aligned}\Phi : \mathbb{R}^d &\to \mathbb{R}^e \\ x &\mapsto \tilde{x}\end{aligned} \tag{1.36}$$

D'après le théorème de Mercer, le produit scalaire présent dans les équations permettant de résoudre par optimisation quadratique le problème de séparation à l'aide d'un hyper-

plan peut être remplacé par une fonction noyau K, équivalente à un produit scalaire. Cette fonction noyau est donnée par :

$$\begin{aligned} K : \mathbb{R}^d \times \mathbb{R}^e &\to \mathbb{R}^d \times \mathbb{R}^e \\ (x, y) &\mapsto \langle \Phi(x), \Phi(y) \rangle = \langle \tilde{x}, \tilde{y} \rangle \end{aligned} \tag{1.37}$$

En appliquant cette fonction noyau dans l'équation 1.31 la quantité à maximiser sous contraintes devient :

$$Ł(w, b, \alpha) = \sum_{1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j K(x_i, x_j) \tag{1.38}$$

Et par conséquent la fonction de décision non linéaire se traduit par :

$$y = Signe \left(\sum_{i=1}^{N_v} \alpha_i y_i \cdot K(s_i, x) + b \right) \tag{1.39}$$

Où α_i représentent les coefficients de Lagrange obtenus pendant l'apprentissage et s_i sont les N_v vecteurs support pour lesquels $\alpha_i \neq 0$.

Le choix de K s'effectue généralement de manière empirique en fonction de l'application. Les fonctions les plus employées sont :

1. Le noyau polynomial (d'ordre q) : $K(x, y) = (\gamma \langle x, y \rangle + \theta)^q$,
2. Le noyau RBF : $K(x, y) = exp(-\gamma \|x - y\|^2)$,
3. Le noyau Tangente hyperbolique : $K(x, y) = \tanh(\gamma \langle x, y \rangle - \theta)$.

Les différents paramètres des fonctions noyaux (γ, θ, q) sont également à optimiser en fonction des applications.

1.4.2/ Méthodes basées sur le Boosting

Les méthodes de classification basées sur le principe du Boosting sont très populaires étant donnée la relative simplicité de mise en œuvre. L'idée principale consiste à combiner linéairement plusieurs classifieurs dits "faibles" pour donner naissance à un classifieur dit "Fort". Cette famille de méthode s'est montré efficace pour diverses applications de vision par ordinateur comme la détection de visages [138].

1.4.2.1/ Le Boosting

Le principe du Boosting a été mis en place par Freund et Schapire [47] en 1995. Les auteurs indiquent que tout algorithme prédictif, permettant d'apprendre avec une certaine

confiance et une erreur de classification inférieure à 0.5 peut être transformé en un algorithme d'apprentissage avec une confiance aussi grande et une erreur de classification aussi petite que désirée. Autrement dit, il est possible de parvenir à obtenir un classifieur performant en partant d'un classifieur faible donnant de meilleurs résultats que le hasard. A partir d'un modèle prédictif faible linéaire, la fonction de décision finale peut être non linéaire après plusieurs itérations. A partir d'un ensemble de L de n échantillons, le Boosting, dans sa version initiale, se déroule en plusieurs étapes :

1. L'apprentissage est réalisé dans un premier temps à partir d'un ensemble sélectionné L_1 de n_1 exemples ($n_1 < n$) et fournit la règle de décision h_1 : c'est le classifieur faible C_1.
2. La seconde étape consiste à sélectionner un nombre n_2 d'exemples de l'ensemble $L \setminus L_1$ de manière à ce que la moitié de ces exemples soit correctement classée par C_1. Un nouvel apprentissage est réalisé à partir de ce sous ensemble d'échantillon, ce qui permet d'obtenir la règle de décision h_2 du second classifieur C_2.
3. Enfin l'apprentissage est réalisé sur $L \setminus \{L_1 \cup L_2\}$ pour fournir la règle de décision h_3 du classifieur C_3 pour laquelle C_1 et C_2 ne donnent pas la même classe.
4. La règle de décision du classifieur final C est par la suite $h = \textit{vote majoritaire}(h_1, h_2, h_3)$.

Les trois sous-ensemble L_1, L_2 et L_3 doivent couvrir tout l'ensemble d'apprentissage afin de recueillir toutes informations liées à L. Ce processus peut être itéré pour chaque classifieur faible ou encore amélioré en ajoutant une pondération des échantillons d'apprentissage. Ces améliorations ont conduit à une des variantes du Boosting les plus utilisée : Adaboost.

1.4.2.2/ LE BOOSTING ADAPTATIF : ADABOOST

En 1996, Freund et Schapire ont proposé la variante AdaBoost [46]. Son principe est de concentrer progressivement le problème de classification vers les échantillons équivoques, grâce à la prise en compte d'un "poids" pour chaque échantillon. Au début du processus, tous les échantillons ont le même poids. Un apprentissage est réalisé et fournit une première règle de décision h_1. On mesure alors l'erreur apparente de classification e_t, c'est à dire la somme des poids des échantillons qui sont mal classés par le premier classifieur. Si le classifieur faible s'est montré efficace, il se verra affecté un coefficient λ important dans la décision finale. Les poids des échantillons sont alors mis à jour avant d'itérer le processus : le poids des échantillons mal classés est augmenté, alors que celui des échantillons bien classés est diminué. Le processus est itéré tant que l'erreur apparente est inférieure à 0.5, et que le nombre d'itération est inférieur à un seuil fixé au préalable. Les auteurs ont montré que l'erreur de classification globale obtenue est bornée.

L'algorithme Adaboost est décrit en détails ci-dessous. Il consiste donc à construire le classifieur final qui est la somme pondérée des classifieurs faibles h_t obtenus lors de l'apprentissage de l'ensemble D_L de p exemples. Le poids de chaque exemple $\mathbf{x}_i$ est noté g.

Algorithme 1 : Algorithme d'apprentissage par AdaBoost

-Soit une base d'apprentissage $\mathrm{D_L} = \{\mathbf{x}_i, y_i\}_{i=1}^{p}$, nombre maximal d'itération T_{max}

-Initialiser les poids de chaque échantillon $g_i^{(t)} = 1/p, \forall i = \{1, ..., p\}$

pour $t = 1, ..., T_{max}$ **faire**

1. Entraîner le classifieur faible à partir des échantillons pondérés $\left\{\mathrm{D_L}, g^{(t)}\right\}$ et obtenir la règle de classification $h_t : \mathbf{x} \to \{-\mathbf{1}, +\mathbf{1}\}$.

2. Calculer l'erreur pondérée e_t de h_t : $e_t = \sum_{i=1}^{p} g_i^{(t)} \mathrm{I}\left(y_i \neq h_t\left(\mathbf{x}_i\right)\right)$

3. Calculer les coefficients λ_t

$\lambda_t = \frac{1}{2} \log\left(\frac{1-e_t}{e_t}\right)$

4. Mettre à jour les poids

$g_i^{(t+1)} = \frac{g_i^{(t)}}{Z_t} \exp\left\{-\lambda_t y_i h_t\left(\mathbf{x}_i\right)\right\}$

Où Z_t est une constante de normalisation $Z_t = 2\sqrt{e_t(1-e_t)}$

si $e_t = 0$ *ou* $e_t \geq \frac{1}{2}$ **alors**

Arrêt

$T = t - 1$

fin si

fin pour

-Résultat : le classifieur final est alors défini par

$$y\left(\mathbf{x}\right) = Signe\left(\sum_{t=1}^{T} \lambda_t h_t\left(\mathbf{x}\right)\right)$$

Une des principales difficultés d'Adaboost est le choix du classifieur faible.

Dans le cas d'un classifieur faible basé sur un hyperplan, la frontière dans un espace à deux dimensions donné figure 1.13 à titre d'illustration peut être représentée par des lignes obliques. La région définie par le classifieur fort, qui utilise la combinaison linéaire des classifieurs faibles, peut donc être complexe, séparer les classes non linéairement séparables, et donc permettre d'obtenir de bonnes performances de classification.

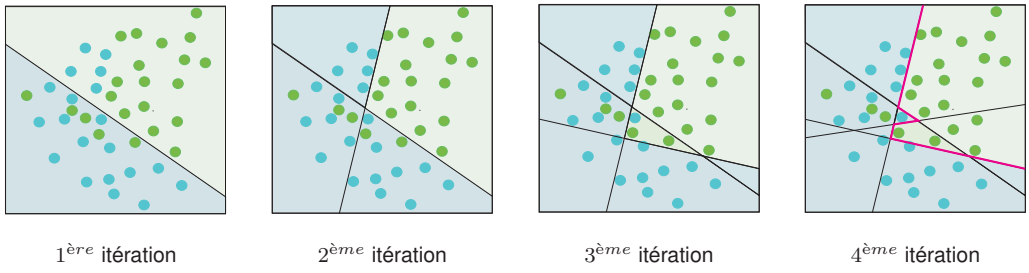


FIGURE 1.13 – Exemple d'apprentissage par AdaBoost - Cas des lignes obliques. (à chaque itération, un classifieur faible est ajouté).

En pratique, les auteurs qui utilisent cet algorithme choisissent souvent, comme classifieur faible, un simple seuillage appliqué sur un seul des attributs parmi les d attributs disponibles (voir figure 1.14). Le classifieur faible est alors défini comme suit : $h_t(\mathbf{x}) = 1 \Leftrightarrow x_k < \rho_k$ et $h_t(\mathbf{x}) = -1$ sinon, où ρ_k est un seuil appliqué à la composante x_k du vecteur d'attributs.

Dans ce cas, l'algorithme Adaboost permet simultanément l'apprentissage de la frontière de décision et la sélection des attributs les plus discriminants. En effet, le classifieur faible ne dépendant alors que d'un seul attribut, chaque itération de la fonction de décision ne porte elle même que sur un attribut. Si T est inférieur à d, alors il y a bien eu sélection d'attributs. Ce peut être également le cas si T est supérieur ou égal à d, car un même attribut peut être utilisé à différentes itérations. Cette astuce a l'avantage de permettre de juger la robustesse des attributs et surtout de diminuer leur nombre (et ainsi éventuellement d'économiser de la puissance de calcul).

Cet algorithme est notamment populaire pour son utilisation en détection de visages et de personnes proposé par Viola et Jones [138, 137]. Il permet d'obtenir un système très efficace et très rapide, grâce à la simplicité de la fonction de décision.

Notons encore qu'un outil logiciel qui permet d'évaluer l'algorithme d'Adaboost et surtout l'implantation matérielle automatique de la fonction de décision, a été développé au sein du laboratoire Le2i. Cet outil génère un code VHDL qui peut être embarqué sur une cible programmable comme les FPGA [98]. Il pourrait être utilisé dans une future évolution de notre travail actuel, c'est à dire, l'implantation matérielle du processus complet de la détection de chutes.

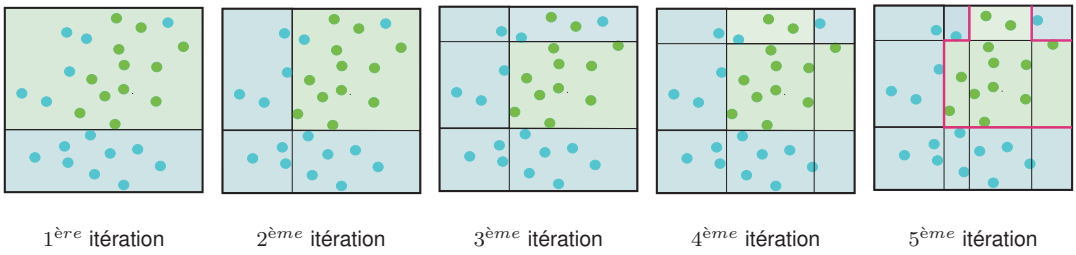


FIGURE 1.14 – Exemple d'apprentissage par AdaBoost (à chaque itération, un classifieur faible est ajouté).

1.5/ CONCLUSION

Nous avons présenté, au cours de ce chapitre, les différentes étapes qui composent généralement un système de détection de chute. Nous avons notamment pu voir, au travers de l'état de l'art, qu'il existe de nombreux attributs spatiaux et temporels pouvant être extraits afin de caractériser la chute. Leur utilisation a permis à plusieurs auteurs d'obtenir de bonnes performances de détection, mais il reste possible d'apporter une contribution en utilisant toute la puissance des méthodes de classification automatiques telles que les SVM ou le Boosting pour sélectionner automatiquement les attributs les meilleurs ainsi que les combinaisons les plus efficaces de ces attributs. C'est une des pistes que nous suivrons pour définir notre propre méthode de détection présentée dans le chapitre 3, tout en en tenant compte de nos contraintes d'implantation temps réel, et de système monoculaire.

Avant d'en arriver à la présentation de la méthode en elle-même, il nous reste à analyser les données et protocoles d'évaluation qui participeront à son élaboration.

2

DONNÉES, PROTOCOLES ET MÉTRIQUE POUR L'ÉVALUATION DE ROBUSTESSE

La mise en place de protocoles d'évaluation est une phase critique de la mise au point de tout système de reconnaissance de forme ou de détection. Ce travail nécessite un ensemble de données fiables, réalistes dans la mesure du possible, afin d'une part de concevoir les méthodes et d'autre part de confronter les résultats de différentes approches. Les protocoles doivent également être clairement établis, ainsi que les métriques d'évaluation. Tous les détails concernant ces éléments ne sont pas toujours donnés dans les articles présentant les méthodes, ce qui ne facilite pas les comparaisons.

Nous allons donc dans ce chapitre étudier les bases de vidéos standards disponibles pour l'évaluation des méthodes, ainsi que les approches adoptées par les chercheurs pour évaluer la robustesse de leurs algorithmes de détection de chutes. Nous verrons qu'il n'existait, lorsque nous avons débuté ce travail, pas suffisamment de bases de vidéos disponibles dédiées à la détection de chutes. Nous avons donc été amenés à construire notre base de vidéos, annotée, intégrant des scènes de chutes et des activités de la vie quotidienne.

L'accent sera mis sur les mesures que nous avons prises pour évaluer la robustesse de notre système contre le changement d'environnement qui pénalise considérablement la méthode en terme de performance puisqu'on exige de l'algorithme d'être capable de distinguer les chutes des autres activités dans une configuration difficile lorsque l'environnement où la scène se déroule n'a pas été pris en compte dans le modèle d'apprentissage. Cette contribution est présentée dans la section 2.4.3 dans laquelle nous définissons trois protocoles utilisés dans nos expériences pour l'évaluation de notre détecteur de chutes. Une deuxième contribution consiste à introduire une métrique qui permet de tolérer, avec une marge précise, un décalage temporel entre la chute réelle et sa détection. Cette métrique, présentée dans la section 2.5, permet d'analyser les séquences d'images par créneaux et respecte la nature continue d'un flux vidéo en temps réel.

2.1/ LES BASES DE VIDÉOS STANDARDS

Ces ensembles de vidéos, comme nous l'avons précisé, permettent la mise au point et l'évaluation des méthodes de reconnaissance d'activités humaines. Il existe différentes bases de vidéos standards utilisées dans plusieurs travaux de recherches qui varient selon le type de l'application et de l'activité. En effet, les activités humaines se différencient essentiellement par leur complexité. Un mouvement partiel de la personne dit "geste" ne se détecte pas de la même manière qu'une "action" qui représente un ensemble de gestes. Les actions elles-mêmes peuvent être très différentes les unes des autres. Nous allons présenter succinctement les ensembles de vidéos standards les plus utilisés pour la reconnaissance d'actions humaines. Ensuite nous présentons plus en détails les bases de vidéos disponibles permettant d'évaluer les méthodes de détection de chutes.

2.1.1/ LES BASES DE VIDÉOS POUR DIFFÉRENTS TYPES D'ACTIONS

Les premières méthodes de reconnaissance d'actions ont été consacrées à des gestes simples. Dans ce contexte, les bases KTH [1] et Weizmann [2] ont été les plus utilisées dans la littérature :

KTH [121] : Cette base se compose de 6 classes d'actions différentes répétées à plusieurs reprises par 25 personnes. Le fond est homogène et statique dans la plupart des séquences prises dans quatre configurations et environnements différents. Elle se compose d'un total de 2391 exemples de vidéos, et contient les actions suivantes : marche, course, applaudissement, jogging, boxe, battement des bras. Les performances en termes de précision de détection obtenues à partir de cette base sont de 94% environ pour des méthodes de reconnaissance d'actions telles que [51], [55] ou [147].

FIGURE 2.1 – Exemples d'images de la base KTH.

Weizmann [52] : Elle contient 90 séquences vidéo de dix classes d'actions différentes (*bending downwards, running, walking, skipping, jumping-jack, jumping forward, jumping*

1. http ://www.nada.kth.se/cvap/actions/
2. http ://www.wisdom.weizmann.ac.il/ vision/SpaceTimeActions.html

in place, galloping sideways, waving with two hands, and waving with one hand) et chacune est réalisée par 9 sujets. Cette base a été utilisée par différents auteurs et leur a permis d'atteindre d'excellents taux de reconnaissance (Précision de 100%) lorsque la méthode est basée sur la soustraction de fond comme dans [143] [40] [120] et 90% lorsqu'il s'agit des méthodes basées sur les sacs de mots (BoF) [104] [63]. La Figure 2.2 montre quelques images de cette base pour laquelle l'arrière-plan est homogène et statique, ce qui est l'une des raisons pour lesquelles les performances obtenues dans l'état de l'art sont très bonnes.

FIGURE 2.2 – Exemples d'images de la base Weizmann.

D'autres bases ont été mises en place par des laboratoires de recherches dans le but d'estimer les poses 3D de la personne. C'est le cas des références IXMAS [3] et HumanEva [4]. Nous présentons des exemples figures 2.3 et 2.4.

IXMAS : L'INRIA a enregistré la base XMAS avec plusieurs acteurs et caméras, et contient 11 actions. La Figure 2.3 montre des exemples d'images acquises des cinq points de vues choisis. Elle a permis dans [145] et [144] d'évaluer la reconnaissance d'actions en 3D.

HumanEva : Cette bibliothèque a été proposée dans le but de valider les algorithmes d'estimation 3D de pose et de mouvement d'une personne. Un système multi-caméras calibrées est associé à un système de Motion Capture (MoCap). Les vidéos acquises contiennent 6 actions courantes (marcher, courir, etc.) de 4 personnes (1 femme et 3 hommes). Des ensembles d'apprentissage et de test (intégrant la vérité-terrain) sont disponibles. Dans [119], la base HumanEva a été choisie afin d'évaluer la robustesse du suivi 3D adopté pour le système de détection de chute.

La majorité des bases existantes dont KTH ou Weizmann présentent des actions artificielles et non-réalistes dans des scènes où le fond est statique et homogène. Reconnaître les actions dans un environnement non-contrôlé est une tâche plus ardue. D'autres types d'actions plus réalistes ont été étudiées. Par exemple, Rodriguez et al. [115] ont mis

3. http ://4drepository.inrialpes.fr/public/viewgroup/6
4. http ://vision.cs.brown.edu/humaneva/

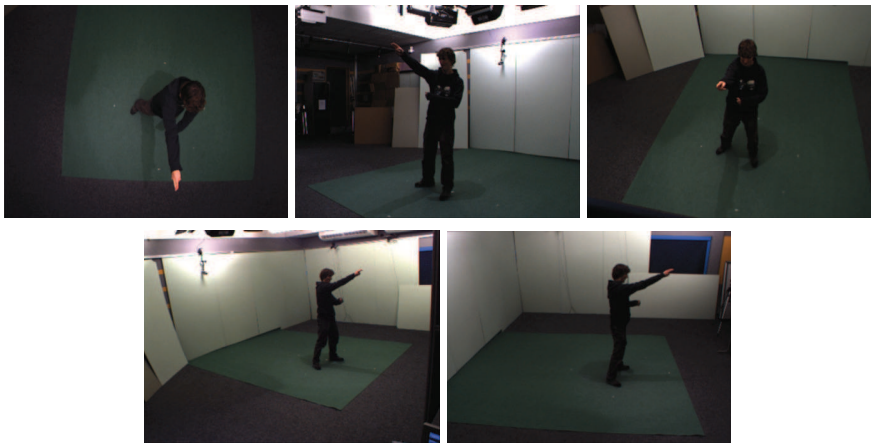


FIGURE 2.3 – Exemples d'images de la base IXMAS.

FIGURE 2.4 – Exemples d'images de la base HumanEva.

en place la base UCF sport contenant 150 vidéos d'actions sportives collectées à partir des scènes de TV et de films présentant des fond d'images dynamiques dans certains exemples. Récemment, le CRCV [5] a proposé la base UCF101 [6] une extension de UCF50 contenant 101 classes d'actions collectionnées à partir de Youtube. Il s'agit d'une large base composée d'un ensemble de 13000 séquences vidéos d'une durée totale de 27 heures [129]. Des images de chacune des 101 classes sont présentées dans la Figure 2.5.

5. Center for Research in Computer Vision
6. http ://crcv.ucf.edu/data/UCF101.php

FIGURE 2.5 – Exemples d'images de la base UCF101.

Les bases de vidéos dédiées à la reconnaissance d'actions humaines sont donc nombreuses, et la base UCF101 mise en ligne récemment contient un large nombre de classes réalistes. Toutefois, il n'existait quand nous avons commencé ce travail que très peu d'ensembles de vidéos standard permettant d'évaluer spécifiquement les méthodes de détection de chutes. Nous allons présenter ces bases dans la partie suivante.

2.1.2/ LES BASES DE VIDÉOS DÉDIÉES AUX CHUTES DE PERSONNES

Comme nous l'avons mentionné auparavant, le problème de détection de chutes et un problème de reconnaissance d'actions à deux classes "chute" et "non-chute". Chaque base de vidéos doit donc contenir des exemples de chacune de ces deux classes d'actions. Toutes les actions de la vie quotidienne seront appelées non-chutes ou ADL [7].

La base DIRO [8] proposée par l'université de Montréal est basée sur un système composé de 8 caméras IP (Gadspot gs-4600) à bas coût et équipée d'un objectif grand angle pour couvrir toute la salle. Cette base de 24 scénarios contient des difficultés qui peuvent

7. Activities of Daily-Life
8. Département d'Informatique et de Recherche Opérationnelle

engendrer des erreurs de segmentation telles que :

– Des artefacts dus à une haute compression (MPEG4),
– des réflexions de lumière ou ombres qui peuvent être détectés comme objets en mouvement durant la phase de segmentation,
– des occlusions, etc.

Des chutes et des activités ADL ont été simulées par un seul sujet pour chaque séquence. Des exemples de chutes vers l'avant et vers l'arrière sont données. Ces chutes sont parfois dues à une perte d'équilibre depuis une position debout ou dues à une assise instable. Les activités ordinaires comme marcher, s'asseoir ou se lever complètent cette base de vidéos. Les auteurs ont annoté cette base en repérant le début et la fin de chaque action de chacune de ses vidéos. Plus de détails sont donnés dans un rapport technique [37].

FIGURE 2.6 – Exemples de 8 points de vue issus de la base utilisée dans [118].

Miao et al. [95] ont également construit un ensemble de vidéos pour évaluer leur algorithme de détection de chutes. Les activités sont effectuées par un cascadeur dans une pièce équipée de 4 caméras placées dans les 4 coins. La figure 2.7 illustre les 4 prises de vues correspondant à la même scène. Cet ensemble se compose de 3 séquences vidéos. La première, destinée à l'apprentissage, contient 30 chutes et 30 non-chutes. La deuxième séquence comporte 29 scènes de chutes et 29 de non-chutes utilisées pour une étape de validation qui sert à ajuster les paramètres du classifieur. La dernière séquence qui présente 29 chutes et 29 non-chutes est utilisée pour le test permettant d'évaluer les performances du classifieur construit. Au moment où nous avons réalisé ce travail, cette base n'était malheureusement pas disponible en ligne.

Olivieri et al. [108] ont construit une base de données qui contient un total de 394 séquences vidéo de 6 actions différentes enregistrées par 12 sujets différents. La fréquence d'acquisition d'images est de 25 images par seconde, et toutes les actions ont été acquises pour une même distance focale de la caméra, sans préparation particulière des conditions d'éclairage. Le lieu de l'acquisition des vidéos représente un fond très simple ne présentant aucune difficulté notamment, présence de meubles pour l'occlusion ou la

FIGURE 2.7 – Exemples de 4 points de vue d'une scène, utilisés dans [95].

texture.

D'autres bases de vidéos de chutes plus réalistes ont été acquises.

Liao et al. [83] ont conçu une base de vidéos pour la détection de chutes contenant des images prises dans différents endroits (des scènes intérieures et extérieures). Ceci permet d'évaluer la méthode de manière approfondie, en comparaison avec les méthodes qui utilisent le même environnement pour toute la base. La résolution des images pour toutes les expériences est 320×240. La méthode d'évaluation retenue est celle du "Leave One Out", avec 23 vidéos pour l'apprentissage et une pour le test. La mise en place de cette base semble être intéressante de point de vue évaluation dans des conditions réalistes. Cette base n'était pas disponible en ligne au moment où de nos travaux.

Leone [79] a mis en place une base de vidéos où 13 cascadeurs professionnels simulent des chutes, guidés par des gériatres afin d'obtenir des séquences vidéos réalistes. Un total de 260 chutes dans toutes les directions parmi 460 séquences vidéo a été enregistré. Ces vidéos contiennent des chutes vers l'arrière, des chutes latérales, et des chutes vers l'avant avec des occlusions partielles pour la moitié de ces exemples, occlusion générées par des meubles. Des scénarios de non-chutes qui ressemblent à des chutes ont été aussi élaborés avec des activités simples de la vie quotidienne. Ces vidéos sont acquises grâce à une caméra 3D de type Time-of-flight donnant des informations de profondeur utilisées par la méthode de détection. Il s'agit d'une base de vidéo complète qui traitent différents types de difficultés et permet une évaluation intéressante de la méthode. Toutefois, l'acquisition a été effectuée en utilisant une caméra 3D ce qui ne permet pas d'établir un système de détection de chute qui intègre à la fois le traitement et la transmission de l'information.

2.1.3/ CONCLUSION

Nous avons présenté l'historique des bases de vidéos utilisées en vue de la reconnaissance d'activités humaines. Les premières bases mises en place contenaient des actions simples dans des environnements peu complexes. Elles se caractérisent par des fonds unis et sont enregistrées toutes dans les mêmes lieux. Ceci ne permet pas d'évaluer

les algorithmes lorsque les environnements de test sont différents de ceux de l'apprentissage. Au moment où nous avons réalisé ce travail, aucune de ces bases n'était malheureusement disponible en ligne. L'ensemble de ces considérations nous a conduits à construire notre propre base et à la mettre à disposition des chercheurs. Pour parfaire le choix des vidéos, des types d'actions, nous avons donc étudié les méthodes d'évaluation des performances et de robustesse communément utilisées dans ce type d'approche.

2.2/ LES MÉTHODES D'ÉVALUATION DES PERFORMANCES

Nous nous intéressons aux méthodes de validation des systèmes de reconnaissance d'actions basés sur la classification supervisée, pour lesquelles deux ensembles d'apprentissage et de test disjoints sont utilisés pour éviter l'apprentissage "par cœur". D'une part, le nombre d'échantillons utilisés pour l'apprentissage est un facteur très important pour la fiabilité de la règle de décision. D'autre part, plus le nombre et la représentativité des échantillons de test sont significatifs, plus le résultat de l'évaluation de la classification aura un sens. Une bonne évaluation des performances de la classification requiert donc si possible un nombre élevé d'échantillons des deux classes. Il existe principalement deux méthodes communément utilisées pour l'évaluation des méthodes de reconnaissance d'activités : la validation simple et la validation croisée. La première est utilisée quand on dispose d'un nombre suffisant d'échantillons et pour le test et pour l'apprentissage. La validation croisée a pour but de maximiser la taille de l'ensemble d'apprentissage, pour un petit nombre de données total disponible.

2.2.1/ LA VALIDATION CROISÉE

La technique de la validation croisée est largement utilisée. Elle consiste à diviser l'ensemble de départ en $r \geq 2$ sous-ensembles disjoints de même taille dont un certain nombre de sous-ensembles est utilisé pour le test alors que le reste sert pour l'apprentissage. C'est le principe de la validation croisée d'ordre r (r folds cross validation $r - CV$), représenté figure 2.8.

Soit S l'ensemble de n échantillons de classes connues (puisqu'on parle d'apprentissage supervisé). S est divisé en r sous-ensemble tous de même taille $E(\frac{S}{r} \pm 1)$. A chaque itération, on détermine le nombre d'erreurs de classification. Connaissant la classe de chaque échantillon, il est possible de construire la matrice de confusion définie de la manière suivante : étant donné c le nombre de classes de S, la matrice de confusion $(C_{i,j})\forall\{i,j\} \in \{1,2,..,c\}$ est de taille $c \times c$ et le coefficient $C_{i,j}$ représente le nombre d'éléments de la classe réelle i classés par le système dans la classe j.

Un cas particulier de la validation croisée appelé "leave one out" est souvent utilisé pour la validation des systèmes de détection de chutes. Le nombre de sous-ensembles est

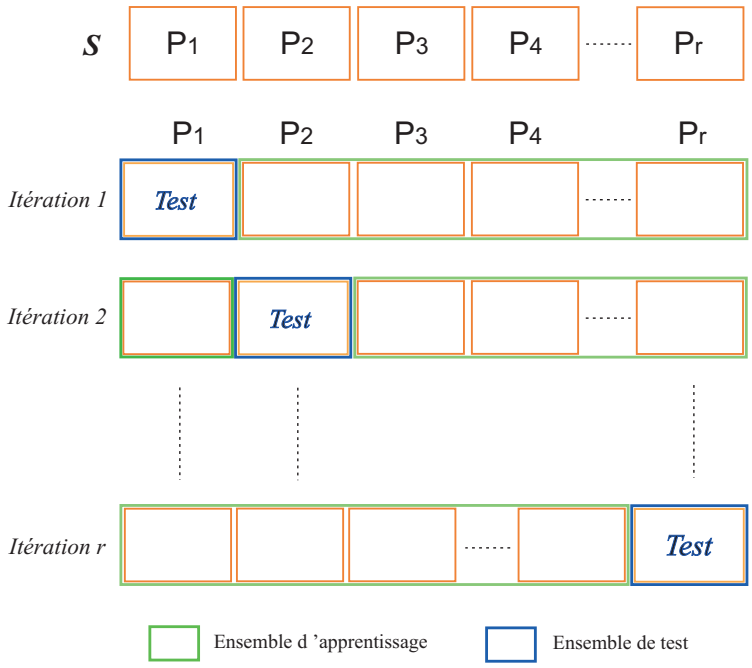


FIGURE 2.8 – Schéma du principe de la validation croisée d'ordre c.

égal au nombre de mesures de l'ensemble ($r = n$). L'apprentissage est donc réalisé sur toute la base excepté un échantillon qui reste pour le test. Souvent utilisé pour les ensembles contenant très peu de données afin de fournir le maximum de données pour l'apprentissage. Ceci permet de maximiser la taille de l'ensemble d'apprentissage mais présente l'inconvénient de se rapprocher de l'apprentissage "par cœur". Rougier et Liao [118, 83] ont par exemple utilisé cette technique pour l'évaluation de leur méthode de détection de chute.

2.2.2/ LA VALIDATION SIMPLE

Dans ce cas les deux ensembles disjoints d'apprentissage L et de test T issus de l'ensemble total d'échantillons S sont fournis. Grâce à cette configuration, "l'apprentissage par cœur" est évité. Mais elle n'est utilisée que lorsqu'on dispose d'une large base de données, c'est à dire un nombre d'échantillons suffisant pour le test et pour l'apprentissage.

2.2.3/ LES GRANDEURS CARACTÉRISANT LES PERFORMANCES

A partir des méthodes de validations précédemment décrites, il est possible de caractériser la méthode de reconnaissance à l'aide de différentes grandeurs. En effet, même si l'erreur globale de classification donne une information importante, elle est rarement suffisamment pertinente à elle seule pour caractériser le système. Les plus couramment

utilisée sont : la sensibilité, la spécificité, la précision, le rappel et l'erreur finale de classification définis communément de la manière suivante :

- vrais positifs (TP) : nombre de chutes correctement détectées,
- faux négatifs (FN) : nombre de chutes non détectées,
- faux positifs (FP) : nombre d'images ou de suite d'images détectées comme chutes bien qu'il s'agisse de non-chtes,
- vrais négatifs (TN) : nombre d'images ou de suites d'images d'activités normales, détectées correctement comme non-chute,
- precision : $Pr = \frac{TP}{TP+FP} \times 100\%$,
- spécificité : $Sp = \frac{TN}{TN+FP} \times 100\%$,
- incertitude (accuracy) : $Ac = \frac{TP+TN}{TP+TN+FP+FN} \times 100\%$,
- rappel : $Re = \frac{TP}{TP+FN} \times 100\%$,
- erreur globale de classification : $E = \frac{FN+FP}{TP+TN+FP+FN} \times 100\%$.

Suivant les besoins, les paramètres des méthodes sont ajustés pour optimiser l'un ou l'autre de ces critères, parfois plusieurs simultanément, par exemple via la mesure de l'aire sous la courbe Receiver Operating Characteristic (ROC), qui représente par exemple le taux de TP en fonction du taux de FP.

2.3/ ÉVALUATION DE LA ROBUSTESSE DANS LA LITTÉRATURE

2.3.1/ LES DIFFÉRENTS TYPES DE ROBUSTESSES ÉVALUÉES

Outre la méthode de validation et les grandeurs mesurées, les méthodes de détection ou reconnaissance sont en général évaluées selon leur robustesse à un certain nombre de perturbations. Dans le cas de la reconnaissance d'objets dans des images fixes, les robustesses les plus fréquemment évaluées sont celles aux occlusions partielles, aux changements de point de vue, d'éclairage, d'échelle, de perspective, d'orientation, etc. Dans le cas de la détection de chutes, assez peu d'évaluations de ce type ont été menées. Souvent, apprentissage et test de détection sont réalisés à partir de prises de vues réalisées par la même caméra, dans le même lieu. Nous avons donc assez peu d'informations sur les stratégies d'évaluations des méthodes de détection de chutes. Seules les conditions d'acquisition des données, les types de scénarios ainsi que les caractéristiques du matériel sont fournis dans la plupart des cas [118, 83, 108]. En ce qui concerne l'évaluation de la séparabilité inter-classe, plusieurs auteurs ont évalué la robustesse de leur méthode à la présence de différents types d'actions faciles à confondre avec des chutes. En effet, certains mouvements brusques naturels, vers le bas, peuvent être confondus avec une chute (se baisser pour ramasser un objet, s'allonger sur un lit, etc.). Il est donc important de les prendre en compte afin de ne pas générer trop souvent de fausse détection. Toutefois, nous nous intéressons ici plutôt à l'évaluation de capa-

cité de généralisation des méthodes, impliquant une robustesse à différents types de variations soit d'environnement, soit de paramètres d'acquisition, c'est à dire non liés au type d'action en cours. Les difficultés abordées principalement dans la littérature sont les suivantes :

– Robustesse au changement de point de vue : nous n'avons trouvé aucune référence de méthodes de détection de chutes utilisant un point de vue différent pour l'apprentissage et pour le test, tels qu'on peut en trouver en reconnaissance d'objets. Quelques auteurs ont utilisé plusieurs points de vue pour améliorer les performances de détection, comme Rougier dans [118], qui utilise 4 points de vue enregistrés simultanément, le résultat final de détection étant obtenu par un vote à la majorité des décisions indépendante effectuées pour chacun des points de vue. Thome et Auvinet [132, 5, 7] ont proposé le même type de structure. Leurs études montrent que l'usage de plusieurs points de vue améliore les performances, mais il ne s'agit pas à proprement parler d'une étude de robustesse au changement de lieu. Même si Liao et al. [83] ont enregistré leurs vidéos dans plusieurs lieux, les auteurs ayant évalué l'efficacité de leur algorithme à l'aide de la méthode du "Leave-one-out", on peut considérer que le lieu de test a été pris en compte dans l'apprentissage. Dans ce cas la robustesse au changement de lieu est obtenue grâce à l'apprentissage.
– La variation des conditions d'éclairage : il s'agit d'évaluer le système de détection avec des vidéos représentant des conditions d'éclairage différentes. Dans [83], cette condition a été traitée par le fait que les vidéos sont enregistrées dans des scènes intérieures et extérieures. Miao [95] a utilisé l'algorithme de soustraction de fond robuste aux variation d'éclairage. Or chez une personne âgée, il est assez fréquent de se trouver dans une pièce où la lumière est variable voire même très faible. Cette contrainte a été traitée partiellement dans plusieurs travaux [132][28] où les auteurs ont introduit des exemples acquis dans différentes conditions de luminosité.
– La robustesse aux occlusions : Rougier et al. ont traité ce problème en mettant la caméra à une haute position dans la salle afin d'éviter les objets occultants [118], ils ont mis en place dans [119] un système de détection basé sur une méthode de suivi robuste permettant de résoudre le problème d'occlusion. Cucchiara [28] a construit la silhouette de la personne à partir de deux points de vue différents ce qui a rendu le système robuste aux occlusions et Thome [132] a construit un système à deux points de vue pour couvrir la totalité de la salle pour traiter les occlusions partielles.

Il est à noter que certaines méthodes sont à la fois robustes et performantes en termes de classification mais ne permettent pas à l'heure où nous écrivons ce mémoire, le traitement en temps réel. C'est le cas de la plupart des systèmes monoculaires basés sur le suivi 3D [140, 32, 53].

Nous avons donc constaté que l'évaluation des systèmes de reconnaissance d'actions sur des vidéos prises dans des lieux différents des lieux d'apprentissage a été souvent omise. C'est un des points dont nous tiendrons compte lors de l'établissement de notre

propre base de vidéos et lors de notre étude de performances.

2.3.2/ Robustesse : à quelles étapes des algorithmes ?

Comme nous l'avons indiqué dans le schéma générique de détection de chute, la détection de la personne, son suivi et la classification finale sont typiquement les étapes principales d'un système de reconnaissance d'actions. Un système fiable doit donc assurer de bonnes performances à chaque étape pour atteindre des taux de reconnaissance acceptables pour des applications industrielles. La plupart des articles précédemment publiés se concentrent sur les performances finales : ce sont effectivement celles qui auront le plus d'importance au bout du compte. Toutefois, afin de justifier les choix des méthodes intermédiaires, il est important de pouvoir les qualifier individuellement. En effet, plus la détection de personnes ou de mouvements et le suivi seront performants, plus les mesures (attributs) seront précises et meilleure sera la détection finale. Lors de la préparation de notre base de vidéos, nous avons donc introduit des informations de segmentation manuelle de référence, permettant notamment l'évaluation de l'annotation automatique (détection de la personne en mouvement) que nous avons effectuée, et d'optimiser certains paramètres de l'algorithme. Cette annotation manuelle permet l'évaluation de la robustesse des attributs choisis pour la classification, indépendamment de l'algorithme initial de détection de la personne en mouvement.

2.3.3/ Tolérance sur la détection de la chute

Les méthodes de détection d'objets dans une image font parfois intervenir une notion de tolérance dans la précision de la détection. C'est le cas par exemple de la détection et localisation de visages de Viola-Jones : les détections multiples (un même visage est détecté plusieurs fois approximativement au même endroit ou à des échelles proches) sont fusionnées pour n'obtenir qu'une seule réponse. Les réponses multiples ne sont donc pas considérées comme des mauvaises détections. La précision de la localisation elle-même n'est pas étudiée, car il est assez difficile de définir précisément où commence un visage dans l'image. En ce qui concerne la reconnaissance d'action ou la détection de chute, il n'y a généralement pas d'information dans les articles sur la manière dont les auteurs ont pris en compte la notion de précision temporelle de la détection. En effet, la précision sur la notion de début et de fin de chute est relative à l'expert qui analysera la scène. De plus, pour un système fonctionnant en temps réel, c'est à dire à une cadence de l'ordre de 10 à 25 images/s, il n'est pas indispensable que la détection de la chute soit donnée à l'image près pour que la détection soit considérée comme valide.

Dans la plupart des méthodes de détection de chute que nous avons étudiées, les résultats sont donnés par action ou par vidéo et aucune information concernant la position temporelle des détections n'est donnée.

Une des contributions de ce travail sera donc d'introduire dans la section 2.5 une métrique permettant de prendre en compte de manière précise et répétable cette tolérance.

2.4/ PRÉSENTATION DE LA BASE DE VIDÉO $DSFD$

Les considérations précédentes nous ont donc conduits à construire notre propre base de vidéos dédiées à la détection de chute, base intitulée "Data Set for Fall Detection" ($DSFD$) et aujourd'hui disponible en ligne à l'adresse suivante : http ://le2i.cnrs.fr/Fall-detection-Dataset.

2.4.1/ CARACTÉRISTIQUES DE L'ACQUISITION

Nous nous sommes servis d'une caméra équipée d'un grand angle (SONY Handycam) pour l'acquisition de la base de vidéos. La fréquence d'acquisition est de 25 images/s, d'abord enregistrées en HD, puis sous-échantillonnée à 320x240 pixels pour toutes les expériences de cette étude. La caméra a été positionnée la plupart du temps proche du plafond, à une hauteur de 2m20. Suivant les méthodes employées par les autres auteurs, nous avons cherché à varier les situations permettant d'évaluer les différentes robustesses mentionnées précédemment. Nous avons donc pris des vidéos à partir de différents points de vue, et contenant des actions réalisées dans de nombreuses directions.

Cette base, qui représente une contribution de ce travail, contient un nombre total de 222 séquences vidéos dans lesquelles on compte 143 chutes. Au total 79 vidéos ne représentent que des activités de la vie quotidienne. Il a été nécessaire d'enregistrer un nombre important de scènes de chutes puisque la durée moyenne d'une chute est seulement 14 images et que la décision du classifieur est donnée dans un premier temps au niveau image.

Ces images illustrent des scènes réelles, contenant des difficultés auxquelles on peut être confronté chez une personne âgée et/ou dans un simple bureau, notamment :

– Un éclairage variable,
– des reflets importants,
– des ombres marquées,
– des occultations partielles, des meubles,
– des arrière-plans simples ou texturés.
– des mouvements de porte.

Neuf sujets vêtus de couleurs différentes ont participé à la réalisation de cette base. Ils ont effectué des actions de la vie quotidienne (marcher dans différentes directions, s'asseoir, se relever, faire le ménage, se pencher, déplacer une chaise ou de petits objets,

ouvrir une porte, etc.) et des chutes (chutes vers l'avant, sur le côté, vers l'arrière, chute due à une mauvaise position assise). Toutes ces activités ont été prises dans différentes directions sans tenir compte particulièrement du point de vue de la caméra comme illustré dans les figures 2.9, 2.10, 2.11 et 2.12.

2.4.2/ ANNOTATIONS MANUELLES DE LA BASE $DSFD$

Les vidéos v_i de cette base sont annotées avec des informations permettant d'établir la vérité terrain de la chute dans la séquence d'images. Cette annotation sera utilisée lors de l'évaluation de la méthode de détection de chute. La position de la chute dans le temps est définie pour chaque vidéo par les indices des trames qui correspondent à son début b_i et sa fin e_i. Ces éléments sont stockés dans des fichiers textes accompagnant les vidéos. Dans le cas d'une vidéo sans chute, $b_i = e_i = 0$.

Pour la moitié des vidéos, chaque image est annotée dans le même fichier texte : nous avons localisé manuellement le sujet, en définissant sa position par une boite englobante β. La position de cette boite permet d'évaluer la phase de classification indépendamment de la détection automatisée de la personne. De plus, une étiquette y_t est associée à chaque image I_t, telle que $y_t = 1$ si $b_i \leqslant t \leqslant e_i$ et $y_t = 0$ ailleurs. Par conséquent, une vidéo v_i n'est d'autre qu'un ensemble d'images étiquetées et annotées que l'on peut noter comme : $v_i = \{I_t, y_t, \beta_t\}_{t=1}^{\gamma}$ où γ est l'indice de l'image de la séquence vidéo. Le nombre total d'images est d'environ 106000 parmi lesquelles on trouve 2574 images de chute.

2.4.3/ LES DIFFÉRENTS LIEUX – LES DIFFÉRENTS PROTOCOLES

2.4.3.1/ LES DIFFÉRENTS LIEUX

Les vidéos de la base $DSFD$ ont été acquises dans différents lieux listés dans la Table 2.1, définissant 4 sous-ensembles, dans le but d'évaluer la robustesse de la méthode de détection de chute contre le changement d'environnement. Dans les quatre figures qui suivent, seule une image sur 4 a été présentée dans les extraits de séquences, pour une question de lisibilité.

TABLE 2.1 – La base de vidéos $DSFD$.

"Coffee room" (c)	$S_c = \{v_i, b_i, e_i\}_{i=1}^{n_c}, n_c = 70$
"Home" (h)	$S_h = \{v_i, b_i, e_i\}_{i=1}^{n_h}, n_h = 60$
"Lecture room" (l)	$S_l = \{v_i, b_i, e_i\}_{i=1}^{n_l}, n_l = 28$
"Office" (o)	$S_o = \{v_i, b_i, e_i\}_{i=1}^{n_o}, n_o = 64$
$DSFD$	$S = S_h \cup S_c \cup S_o \cup S_l$

(a) Exemples de trois séquences de chutes : La séquence 3 (chute vers l'avant) est prise d'un point de vue différent des séquences 1 (chute latérale) et 2 (chute vers l'arrière).

(b) Exemples de non chutes : présence d'un mouvement de chaise, occlusion, effet d'ombre.

FIGURE 2.9 – Quelques exemples du sous-ensemble "Home" de la base de données $DSFD$.

Le premier sous-ensemble a été enregistré au domicile d'un particulier. On y trouve donc des meubles, canapés, chaises, etc. L'éclairage est assuré par des lampes de faible puissance. Certaines chutes ont été simulées à la fin d'une descente d'escalier.

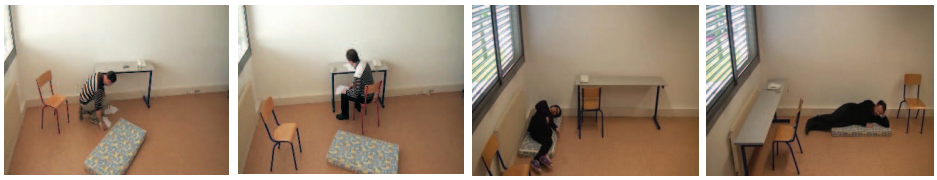
(a) Exemples de trois séquences de chutes.

(b) Exemples de non chutes avec diverses actions de la vie quotidiennes.

FIGURE 2.10 – Quelques exemples du sous-ensemble "Coffee room" de la base de données $DSFD$.

Le deuxième sous-ensemble a été acquis dans une cafétéria, dans une situation de contre-jour léger. Les transferts (passage d'une position assise à une position debout et réciproquement) y sont fréquents. Les acteurs y ont simulés des actions diverses de type prise de boisson, passage d'un balais, etc.

(a) Exemples de trois séquences de chutes.



(b) Exemples de non chutes avec diverses actions de la vie quotidiennes.

FIGURE 2.11 – Quelques exemples du sous-ensemble "Office" de la base de données $DSFD$.

Le troisième sous-ensemble simule un bureau ou une chambre simple. C'est dans cette pièce, pour des raisons techniques, que les séquences simulant le fait de se coucher ou de se lever ont été acquises.

(a) Exemples de trois séquences de chutes.

(b) Exemples de non chutes : balayer, mouvement intrus, réflexion de lumière (contre-jour).

FIGURE 2.12 – Quelques exemples du sous-ensemble "Lecture room" de la base de données $DSFD$.

Enfin le dernier sous-ensemble est une salle de conférence, dans laquelle différents mouvements, entrées, sorties, et bien sûr quelques chutes, sont simulées. C'est dans cette série que l'on trouve le plus de mouvements de portes pouvant induire de fausses détections.

2.4.3.2/ LES PROTOCOLES EXPÉRIMENTAUX

Dans le but de construire les descripteurs de chutes et d'évaluer les performances de la méthode, nous avons défini trois protocoles P_1, P_2 et P_3 dont les ensembles d'apprentissage L et les ensembles de test T proviennent des sous-ensembles définis précédemment en supposant que $\forall s = \{h, o, c, l\}$, nous avons $L_s \cap T_s = \emptyset$ and $S_s = L_s \cup T_s$.

– Pour P_1, les échantillons d'apprentissage et de test ont été construits à partir de vidéos acquises dans les environnements"Home" et "Coffee room" $i.e.$ à partir des sous-ensembles S_h et S_c : $L_{P1} = L_h \cup L_c$ et $T_{P1} = T_h \cup T_c$.
Ce protocole a été utilisé d'une part pour la sélection des transformations (dérivées,

transformées de Fourier et ondelettes) et des attributs de bas-niveau (attributs géométriques) ce qui permettra ensuite de définir expérimentalement le descripteur de chute final, et d'autre part pour évaluer les performances de la détection automatique de la personne en mouvement comparée à la localisation manuelle que nous avons introduite dans l'annotation des vidéos.

– Pour P_2, les échantillons d'apprentissage sont construits à partir des vidéos de "Coffee room" alors que ceux du test proviennent de "Office" et "Lecture room" : $L_{P2} = L_c$ and $T_{P2} = T_o \cup T_l$.
Cette configuration est établie dans le but d'évaluer la robustesse de la méthode au changement d'environnement.

– Pour P_3, l'ensemble d'échantillons utilisés pour l'apprentissage a été sélectionné à partir des vidéos enregistrées dans "Coffee room" et complété avec quelques vidéos de "Office" qui ne contiennent pas de chutes (nf)
($L_o^{nf} = \{v_i, b_i = e_i = 0\}_{i=1}^{\eta_1}$) et "Lecture room" ($L_l^{nf} = \{v_i, b_i = e_i = 0\}_{i=1}^{\eta_2}$).
L'ensemble des échantillons de test provient de "Office" et "Lecture room" : $L_{P3} = L_h \cup L_o^{nf} \cup L_l^{nf}$ et $T_{P3} = T_o \cup T_l$.
Nous avons défini cette configuration dans le but de montrer qu'il est possible d'améliorer les performances grâce à la mise à jour de l'apprentissage, par la simple intégration d'enregistrements d'activités de la vie quotidienne, sans qu'il soit nécessaire d'ajouter des vidéos avec des chutes.

La répartition des différents protocoles est résumée dans le tableau 2.2.

TABLE 2.2 – Construction des protocoles d'évaluation de robustesse.

		"Home"		"Coffee"		"Office"		"Lecture room"	
		Chute	Non chute	Chute	Non chute	Chute	Non chute	Chute	Non chute
$P1$	Apprentissage	Oui	Oui	Oui	Oui	-	-	-	-
	Test	Oui	Oui	Oui	Oui	-	-	-	-
$P2$	Apprentissage	-	-	Oui	Oui	Non	Non	Non	Non
	Test	-	-	Non	Non	Oui	Oui	Oui	Oui
$P3$	Apprentissage	-	-	Oui	Oui	Non	Oui	Non	Oui
	Test	-	-	Non	Non	Oui	Oui	Oui	Oui

2.5/ DÉFINITION D'UNE MÉTRIQUE D'ÉVALUATION

Dans un premier temps, la classe, de type chute ou non chute, sera donnée par le classifieur à une cadence si possible égale à la cadence image. Le flux de sortie du système de classification est donc un flux binaire. Par la suite, nous utiliserons la valeur 1 pour la classe non chute et 0 pour la classe chute. Nous avons représenté un exemple de sortie attendue, basée sur les annotations de l'expert, et de sortie possible d'un classifieur figure 2.13. La chute théorique ou attendue est donc représentée par un créneau d'un

certain nombre de 0 successifs. La sortie réelle du classifieur peut donc présenter de fausses détections à tout instant, mais dans le meilleur des cas, présentera également un créneau de 0 d'une certaine largeur, plus ou moins décalée dans le temps par rapport au créneau théorique.

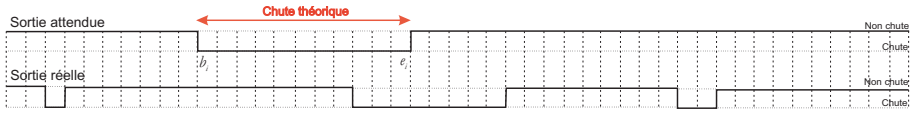


FIGURE 2.13 – Exemple de sortie attendue, basée sur les annotations de l'expert, et de sortie possible d'un classifieur.

Nous pouvons donc définir une méthode d'analyse du signal formé par la sortie du classifieur et qui prend en compte d'une part la largeur du créneau obtenu, et d'autre part le décalage temporel éventuel, afin de donner une fonction de décision finale au niveau du créneau lui-même. Cette définition doit prendre en compte les contraintes temps-réel de l'application, et notamment permettre de fournir une décision avec une latence acceptable (que nous fixerons à 1s par la suite).

Nous introduisons donc un paramètre de tolérance D qui représente le nombre d'images entre le début de la chute théorique et la chute détectée. Afin de déterminer les différentes situations à prendre en compte, nous avons représenté les cas les plus importants en parties pour une meilleure visibilité figures 2.14 2.15.

D'une manière générale, un vrai positif, c'est à dire qu'une chute est bien détectée, (resp. vrai négatif) est généré lorsque Z décisions successives "Chute" (resp. "Non chute") du classifieur dépassent un seuil Z_{max}, et si cette décision est proche au niveau temporel de la vérité de terrain de la chute théorique, ce qui se traduit par les conditions suivantes : Si $D < D_{max}$ et $Z > Z_{max}$ où D_{max} est un paramètre du système, un Vrai Positif est généré. Cette situation est illustrée dans la figure 2.14, cas numéro 1.

Il est évident qu'une très grande valeur du paramètre D_{max} peut amener à une erreur insignifiante de classification (toujours égale à zéro). Il apparaît donc réaliste que cette tolérance sur la position dans le temps de la chute détectée soit inférieure à une seconde (25 images).

Si une chute complète (un créneau) est détecté, le système attend w images avant de continuer le traitement, où w est la taille de la fenêtre d'analyse, qui sera de l'ordre de grandeur de la durée moyenne d'une chute. En effet il peut se produire après cette détection un certain nombre de mauvaises détection de la part du classifieur qui n'ont pas d'importance d'un point de vue pratique, l'important étant que la chute ait été détectée. Il est également peu probable que deux chutes se produisent dans un délai de moins de deux secondes.

Les principaux cas sont résumés figures 2.14 et 2.15 :

– Cas 1 : à l'image numéro 24, un vrai positif (TP) est généré ($Z1 > Z_{max}$ et $D1 < D_{max}$). Notons qu'une bonne détection génère un seul Vrai Positif pour l'ensemble des images d'une chute.
– Cas 2 : pour l'image numéro 14, un faux positif (FP) est généré lorsque $Z2 \geq Z_{max}$. C'est logique puisque le classifieur a donné des fausses détections pour un nombre important d'images successives et le début de cette détection est loin de la chute réelle ($D2 > D_{max}$).
– Cas 3, à l'image numéro 35, un vrai négatif (TN) est généré bien qu'on ait un 1 en sortie de classifieur et un 0 attendu, puisqu'on se trouve dans l'intervalle de la chute réelle. Cependant, puisque $Z3 \geq Z_{max}$, un faux négatif (FN) est généré pour l'image 36. Si la chute n'est pas détectée, le processus doit continuer à analyser l'image suivante. Suivant ce principe, une chute non détectée pourrait engendrer plusieurs faux négatifs (FN) en fonction de la durée de la chute réelle. Le nombre de FN (comme le nombre de TP d'ailleurs) sera donc borné à 1 par vidéo, qui par convention ne contient qu'une seule chute.
– Cas 4 : la détection de la chute apparaît après la chute réelle (entre les images 56 et 59) mais la durée de cette détection est beaucoup plus courte que la durée d'une chute moyenne. Aucun faux positif n'est généré en 59, puisque la condition $Z > Z_{max}$ n'est pas vérifiée. Un vrai négatif (TN) est logiquement généré à l'image numéro 66 ($Z4 < Z_{max}$).
– Cas 5 : la chute est détectée après la chute réelle mais puisque $Z5 > Z_{max}$ et $D5 < D_{max}$, un vrai positif (TP) est généré à l'image numéro 50. Le système peut attendre w images avant de continuer l'analyse.

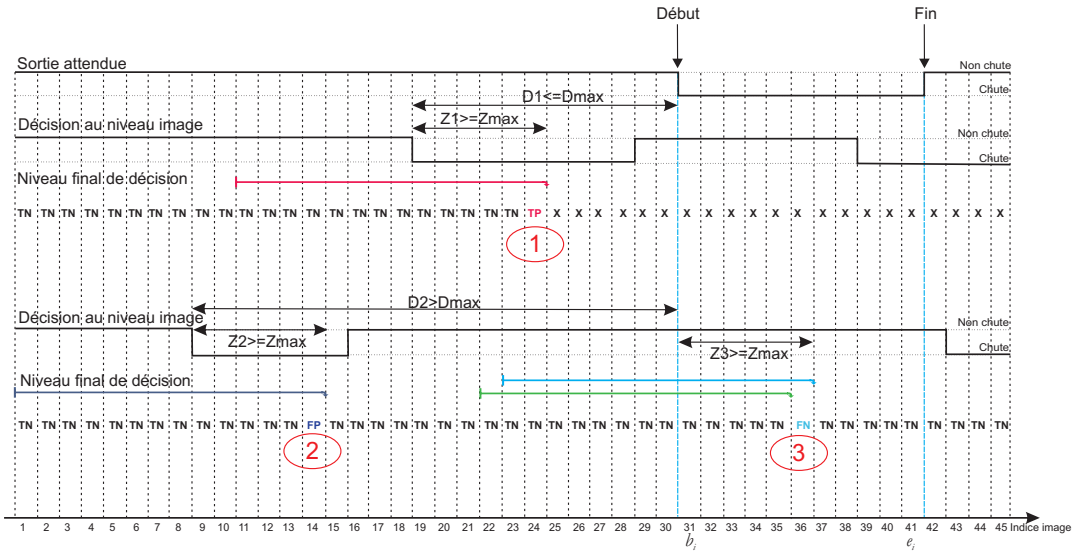


FIGURE 2.14 – Décision finale : cas 1 à 3.

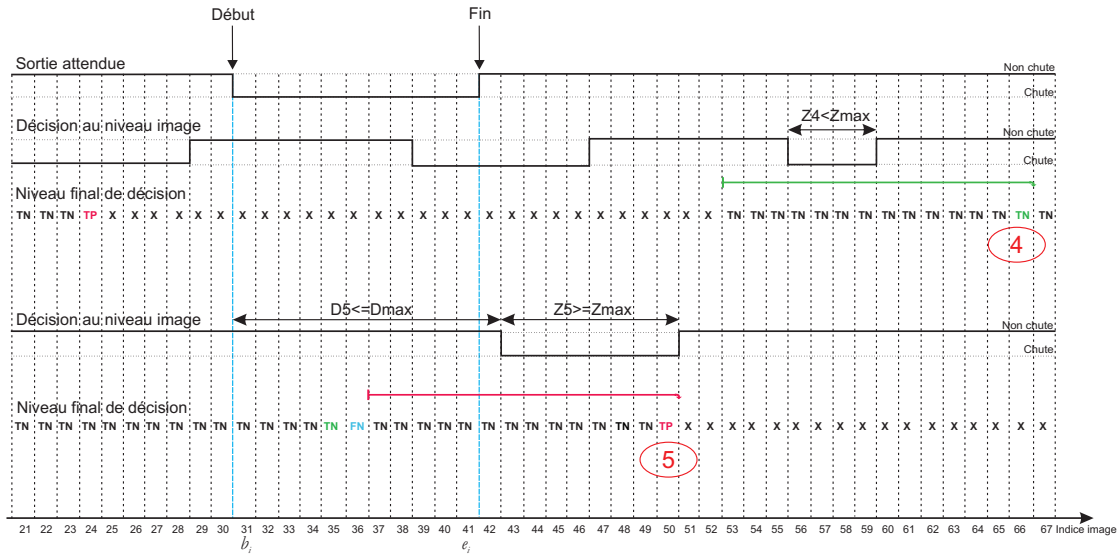


FIGURE 2.15 – Décision finale : cas 4 à 5.

La métrique d'évaluation en elle-même sera constituée de l'ensemble des grandeurs définies en section 2.2.3 et calculées selon la méthode que nous venons d'exposer.

Les valeurs des paramètres de cette métrique (Z_{max},D_{max} et w) peuvent être optimisées en fonction des paramètres d'acquisition (notamment la fréquence d'acquisition d'image) et du nombre moyen d'images caractérisant une chute. Nous présenterons les valeurs retenues dans notre cas, dans la section consacrée à l'évaluation des performances 3.4.2.

2.6/ CONCLUSION

Nous avons présenté dans ce chapitre les caractéristiques des bases de vidéos habituellement utilisées pour la détection de chutes, et les méthodes employées pour évaluer la robustesse de ces méthodes. Nous avons alors présenté la base $DSFD$ que nous avons construite et défini protocoles et métrique d'évaluation. Nous allons maintenant passer à la définition de notre propre méthode, avant de pouvoir l'évaluer grâce à ces définitions.

3

MÉTHODE DE DÉTECTION PROPOSÉE : CONSTRUCTION DES DESCRIPTEURS SPATIO-TEMPORELS

Dans ce chapitre nous allons donc détailler la méthode de détection de chute que nous avons mise en place, basée sur le schéma générique présenté dans le premier chapitre et rappelé figure 3.1.

Nous allons, étape par étape, construire la méthode de détection, parfois en utilisant directement des algorithmes existants tels que ceux cités dans le premier chapitre, parfois en proposant des contributions permettant d'améliorer les performances de détection par rapport à l'état de l'art. Nous serons guidés tout au long de nos choix par les contraintes d'implantations temps réel et embarquées que nous souhaitions *in fine*. Notre contribution principale à ce niveau sera la définition d'un descripteur spatio-temporel nommé $STHF$ formé à partir d'attributs géométriques (informations spatiales) et leurs évolutions temporelles. Ce descripteur a été optimisé en vue de la détection de chute grâce à une phase de sélection automatique d'attributs basée sur l'erreur de classification. Nous présenterons tout d'abord une vue d'ensemble de la méthode, suivie des choix réalisés pour la détection de la personne, l'algorithme de suivi retenu, les attributs géométriques et les outils d'analyse temporels choisis, conduisant à l'optimisation du descripteur appliqué comme vecteur d'entrée de la méthode de classification.

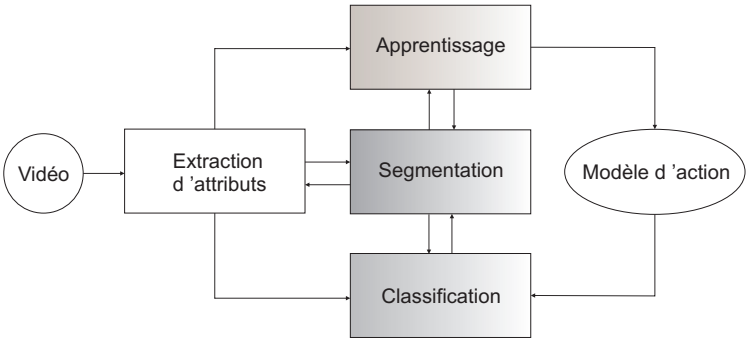


FIGURE 3.1 – Système générique de reconnaissance d'actions humaines.

3.1/ UNE VUE D'ENSEMBLE

Les étapes principales du système de détection de chute que nous avons mis au point, reprennent les phases de calcul décrites au chapitre 1 et illustrées dans la figure 3.2 sont les suivantes :

- Détection et suivi de la plus grande région se déplaçant, en utilisant une méthode de segmentation des zones en mouvement par rapport à l'image de fond. Cette étape est suivie par des opérations de morphologie mathématique (filtrage median, érosion et dilatation) permettant de réduire les artefacts dus à la segmentation. Le résultat est une image binaire et les coordonnées de la boite qui englobe la personne en mouvement.
- Extraction des attributs de bas niveau (attributs géométriques) à partir de l'image binaire, tels que le rapport hauteur/largeur de la boite englobante, l'orientation de l'ellipse qui englobe la personne en mouvement, les moments, etc.
- Transformations des attributs de bas niveau afin de prendre en compte leurs variations temporelles. La combinaison de toutes les transformations appliquées à ces attributs forme le descripteur $STHF$.
- Détection de chute au niveau image basée sur des méthodes de classification supervisée : nous comparerons les Support Vector Machine et Adaboost. Une étape de filtrage par un vote à la majorité sur une fenêtre glissante de taille $m_w = 5$ est appliquée au résultat de décision du classifieur permettant de filtrer les décisions isolées dans le temps. La décision finale est donnée à la cadence image.
- Détection de chute au niveau créneau d'images en utilisant la méthode sur laquelle nous avons basé la métrique qui permet d'évaluer les performances en introduisant une certaine tolérance pour la classification finale.

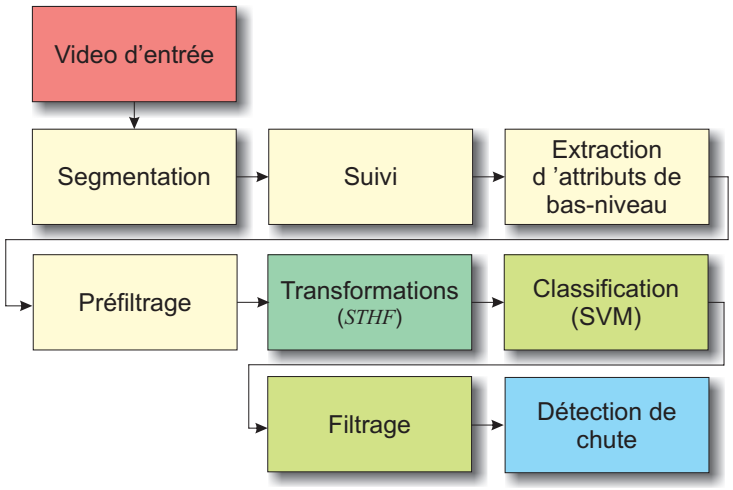


FIGURE 3.2 – Synoptique de la méthode de détection de chute.

3.2/ ANALYSE SPATIALE : CONSTRUCTION DES ATTRIBUTS BAS-NIVEAU

3.2.1/ MÉTHODE DE DÉTECTION DES PARTIES DE L'IMAGE EN MOUVEMENT

Nous avons donc choisi, comme la plupart des auteurs et selon les méthodes décrites dans le premier chapitre, de commencer la détection par une étape de soustraction de la partie statique dans la scène afin de construire une image binaire qui représente la personne en mouvement. Lors des études préliminaires menées, nous avons repris la méthode de segmentation zones fixes/zones en mouvement proposée par Li [81] et rappelée dans la section 1.2.1.2. Les auteurs, dans leur contexte applicatif, ont en effet montré que cette méthode était généralement plus efficace que les simples soustractions de fond. Toutefois, les temps de calculs nécessaires à cette méthode sont très largement supérieurs à ceux des soustractions simples.

Ayant pour objectif une implantation matérielle temps-réel du système de détection de chute complet, nous avons également implanté une méthode classique, rappelée en annexe A, de calcul de la valeur absolue de la différence entre un fond mis à jour périodiquement et l'image courante. L'algorithme de détection que nous avons utilisé est entièrement non-supervisé, et basé sur des images au niveau de gris afin de réduire au maximum le temps de traitement. Étant donné que nous sommes partis de l'hypothèse de la présence d'une seule personne dans la scène, détecter la personne consistera ensuite simplement à construire la boite englobante de plus grande surface à partir des régions détectées, éventuellement fusionnées. Pour que cette détection soit efficace malgré la simplicité de la soustraction de base, nous avons appliqué une succession de filtres visant à éliminer les petites régions issues de la segmentation, et une partie des manques (pixels classés comme fixes alors qu'ils appartiennent à la personne). Cette étape est constituée d'un filtre médian, suivi d'opérations classiques de morphologique mathématique (typiquement une dilatation, suivie d'une érosion, elle même suivie d'une dernière étape de dilatation. La pertinence de ce choix sera discutée en section 3.2.3).

Fusion des zones détectées Cette étape consiste à regrouper les zones représentant le mouvement d'une même personne obtenues par soustraction de fond, afin de combler les manques importants, ce que les opérations de morphologie mathématiques n'ont pu réaliser. Le critère de fusion retenu est une mesure de proximité des boites candidates à la fusion (les boites de trop petite surface ayant été éliminées au préalable). Ce critère $D(A,S)$ est défini de la manière suivante : Soient deux rectangles A et S définis par leurs sommets respectifs A_i et S_j, $i,j = 1,2,3,4$ et $d(A,S)$ la distance euclidienne entre deux points du plan A et S : $D(A,S) = min[d(A_i,S_j)], \forall i,j$.

Les boites englobantes sont alors fusionnées deux à deux si leur proximité D est infé-

rieure à un seuil noté $SeuilDistance$ déterminé au préalable, ou si elles se chevauchent partiellement, suivant l'algorithme 2. La fusion de deux rectangles proches consiste à construire le plus petit rectangle qui les contient.

Soit χ un ensemble de $N \geq 2$ boites englobantes qui définissent les régions en mouvement détecté sur une même image. Les N tentatives de fusion seront effectuées $N-1$ fois, jusqu'à la fin du parcours de tout l'ensemble χ. A chaque itération, le processus de la fusion traite deux boites englobantes. On note par χ' l'ensemble de N' éléments contenant l'ensemble final des boites englobant les zones en mouvement dans l'image après fusion.

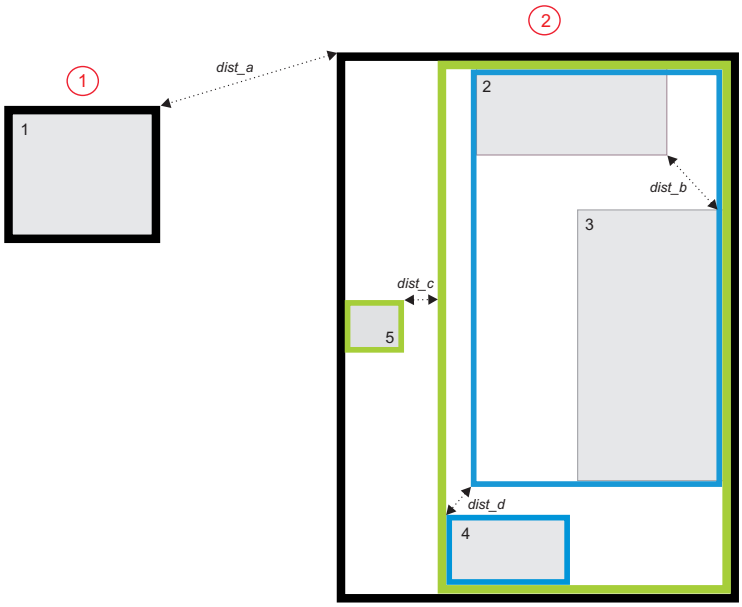


FIGURE 3.3 – Principe de fusion des boites englobantes.

Nous présentons figure 3.3 un exemple de fusion. Dans ce cas χ contient initialement $N = 5$ rectangles. Les rectangles numérotés 2 et 3 ont été fusionnés lors de la première itération étant donné $dist_a = d(B_2, B_3) < SeuilDistance$. L'ensemble χ est ensuite mis à jour et $N = 4$. Étant donné $dist_c$, $dist_d$ sont aussi inférieures au seuil, le processus continue et le nombre de rectangles à traiter est décrémenté à chaque itération. L'algorithme s'arrête lorsqu'il ne reste plus de rectangles proches (la région 1 vérifiant $dist_a > SeuilDistance$). Le résultat final de la fusion est ici un ensemble de $N = 2$ rectangles (contour en noir sur la figure). On a conservé le rectangle 1 puisqu'il représente un mouvement important dans l'image (estimé grâce à sa taille) et le rectangle 2 qui est le résultat de la fusion de 4 régions. Le résultat final de la détection donne deux régions indépendantes représentant le mouvement dans l'image en cours de traitement. La méthode de suivi nous permettra de n'en retenir qu'une seule, qui servira de base à

l'extraction des attributs géométriques.

Algorithme 2 : $\chi' =$ Liste des boites englobantes fusionnées(χ)

Données : i, j : entier
N : nombre de boites initial
N' : nombre de boites final $\forall i,\ B_i$: Boite englobante
Entrées : $\chi = \bigcup_{i=1}^{N}(B_i)$
Sorties : $\chi' = \bigcup_{i=1}^{N'}(B'_i)$

$i \leftarrow 1$;
tant que *($i < N$)* **faire**
  $j \leftarrow 2$;
  tant que *($j < N$)* **faire**
    si *($D(B_i, B_j) \leqslant SeuilDistance$) ou si B_i et B_j se chevauchent*
    alors
      $B'_i \leftarrow Fusion(B_i, B_j)$;
      Actualiser χ par suppression de B_j et remplacement de B_i par B'_i ;
      $N \leftarrow N - 1$;
      $j \leftarrow N$;
      $i \leftarrow 0$;
    sinon
      $j \leftarrow j + 1$;
      fin si
    fin tant que
  $i \leftarrow i + 1$;
  fin tant que
$\chi' \leftarrow \chi$;
$N' \leftarrow N$;

3.2.2/ ALGORITHME DE SUIVI RETENU

Comme nous l'avons vu dans le premier chapitre, l'étape préliminaire de détection de la personne est très souvent implémentée grâce à un algorithme de suivi, permettant notamment de mieux prendre en compte les occlusions partielles temporaires. Les contraintes concernant le temps de calcul (et l'espace mémoire) sont très importantes pour notre application.

Pour proposer un algorithme de suivi efficace et peu gourmand en termes de temps de calcul, nous nous sommes inspiré des méthodes d'association non paramétriques comme celle du plus proche voisin. Cette méthode, décrite par Bar-shalom [9] est une des plus simples et utilisées pour répondre au problème de l'association des données [50]. Elle est séquentielle, et permet de trouver l'association la plus probable entre une observation (rectangle détecté) et une piste (rectangle précédemment détecté), en recherchant celle qui minimise un critère de distance.

Il s'agit d'une méthode par approche régions ne nécessitant pas de prédiction puisque

son principe est de faire correspondre les régions détectées dans l'image en cours de traitement (à l'instant t) et les régions détectées auparavant. Seule une bonne mise en correspondance pourra assurer une extraction temporelle cohérente des attributs géométriques qui assureront la classification.

Le principe de cet algorithme est détaillé ci-dessous :

Soit l'ensemble χ'_t des N' boites englobantes obtenu après l'étape de détection à l'instant t et Γ_t l'ensemble des rectangles qui correspondent aux régions en mouvement ayant été observées et mis en correspondance depuis l'initialisation de l'acquisition.

On associe à chaque rectangle B_i de χ'_t une étiquette $ID(B_i)$ afin de l'identifier. La mise en correspondance consistera à modifier éventuellement l'étiquette d'une région en cours pour lui donner la valeur de l'étiquette de la région correspondante de l'ensemble Γ_t et de mettre à jour ses caractéristiques (coordonnées du rectangle) dans cet ensemble Γ_t.

On définit par δ le critère de correspondance qui dépend de la distance euclidienne entre les centres des deux rectangles, leurs aires (on fera correspondre deux rectangles de surfaces proches) et du délai ρ, égal à la différence entre l'indice de l'image en cours de traitement et l'indice de la dernière image dans laquelle le rectangle considéré est apparu. La région en cours sera mise en correspondance avec celle de Γ_t qui minimise le critère δ. Nous pouvons résumer la méthode par les étapes suivantes :

– Le suivi commence dès la première apparition d'une zone en mouvement dans la vidéo. Au temps initial $t = t_1$ les premiers rectangles B_i sont détectés sur l'image d'indice t_1 ; on leur affecte les étiquettes $ID(B_i)$. Ceci permet d'initialiser χ'_{t_1} et construire l'ensemble Γ_t qui est initialisé tel Γ_t= χ'_{t_1} (étape 1),
– Pour chaque rectangle B_j de χ'_t, donc détecté dans l'image en cours, on détermine le rectangle B_i appartenant à Γ_t qui minimise le critère δ, c'est-à-dire, celui qui lui correspond dans l'historique des rectangles ayant été détectés auparavant (étape 2),
– on affecte à B_i l'identifiant $ID(B_j)$ (étape 3.),
– on met à jour Γ_t en remplaçant le rectangle B_j par B_i et en incluant les nouveaux rectangles s'ils apparaissent pour la première fois (étape 4),
– On actualise l'ensemble χ'_t avec les nouveaux B_i mis à jour en ne retenant que le rectangle de plus grande aire. C'est celui qui correspond au mouvement le plus important dans l'image. Ceci est logique et applicable pour notre application puisque nous traitons des vidéos dans lesquelles une seule personne apparaît dans l'image. Il s'agit donc d'une étape de correction (étape 5).

Un exemple de suivi est présenté figure 3.4 ; à l'instant t, χ'_t contient $N = 2$ rectangles (en bleu) et Γ_t contient les rectangles 1, 2 et 3 provenant des images précédentes (en pointillé). Dans ce cas, la correspondance est réalisée entre le rectangle B_1 de l'image courante et le rectangle B_2 de Γ_t. La boite englobante finale retenue est celle qui est de plus grande aire entre ces deux rectangles. Le rectangle B_1 de l'image courante n'est pas

mis en correspondance avec le rectangle B_1 de Γ_t car son critère δ est trop important. Il est donc ajouté à Γ_t comme nouveau rectangle potentiel. Parmi les quatre rectangles candidats de Γ_t, seul le plus grand, étiqueté 2 sur la figure, sera retenu pour l'extraction des attributs géométriques.

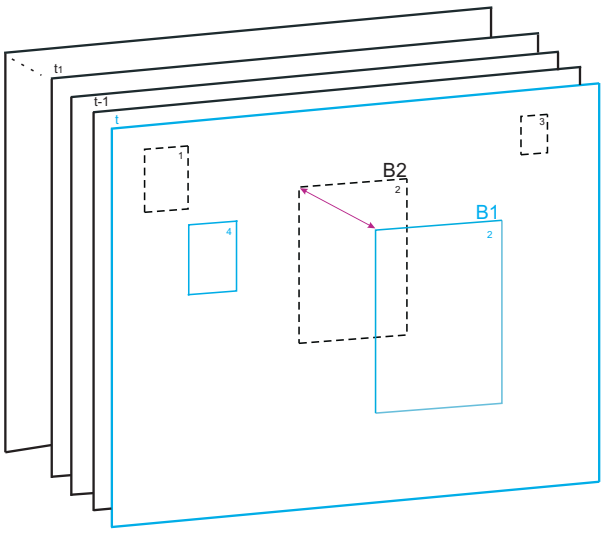


FIGURE 3.4 – Suivi : mise en correspondance des régions.

Grâce à cet algorithme, nous avons construit l'image binaire qui représente le mouvement de la personne dans la scène. Avant de définir les informations spatiales importantes que nous avons pu tirer de cette image binaire, nous allons présenter quelques éléments d'évaluation de cette première étape de détection.

3.2.3/ ÉVALUATION DE LA DÉTECTION

Nous avons fait varier certains paramètres de cette première phase afin de choisir les valeurs optimums qui seront utilisées dans la suite des expérimentations. Nous avons notamment fait varier la taille du masque utilisé pour le filtre médian, pour des tailles de fenêtres 7x7, 5x5 et 3x3. D'autre part, nous avons évalué les performances de segmentation avec et sans les opérations de morphologie mathématique. Pour réaliser cette évaluation, nous avons segmenté manuellement, à l'aide d'un logiciel de retouche d'image, 12 images issues de la base $DSFD$, représentant des situations variées, de chute et de non-chutes. Ces segmentations manuelles représentent la vérité de terrain. Pour chacune de ces images, nous avons comptabilisé les erreurs de classification de chaque type (pixels fixes classés comme pixels en mouvement, et inversement), en comparant les images binaires après fusion et les images binaires segmentées manuellement. Les résultats sont reportés tableau 3.1. Nous avons donc constaté expérimentalement que l'erreur est minimum pour une taille de fenêtre 7x7 du filtre médian, et que la présence

des opérations de morphologie mathématique minimise également l'erreur. Ces opérations ont donc été incluses dans les expériences qui suivront. Nous verrons toutefois dans la section 5.1.2 consacrée à l'adéquation algorithme architecture que leur influence est mineure en ce qui concerne les performances finales de classification. Nous pouvons remarquer au passage que cette évaluation donne des erreurs très faibles, ce qui valide du même coup, pour ces exemples, l'algorithme de fusion des boites englobantes. Une évaluation beaucoup plus complète sera menée concernant cette phase, c'est à dire l'annotation automatique, dans le cadre de l'évaluation globale présentée section 4.2.1.

(a) Segmentation automatique de l'arrière-plan

(b) Vérité-terrain de la segmentation

(c) Résultats de l'annotation automatique sur les images originales

FIGURE 3.5 – Évaluation de l'annotation automatique sur des images de la base $DSFD$.

TABLE 3.1 – Sélection des paramètres de la phase de détection de la personne en mouvement

	Sans filtrage	Filtrage Median			Median et Mopho. Math.
		3×3	5×5	7×7	
E (%)	3.916	3.396	3.318	3.258	**3.175**

Nous avons présenté dans cette partie les étapes qui constituent la phase de détection d'une personne dans une image. Nous avons présenté l'approche basée sur le mouvement que nous avons mise en place pour la segmentation zones en mouvement/zones fixes. Afin d'améliorer cette détection, nous avons proposé d'appliquer des opérations de filtrage. Nous avons ensuite proposé une méthode de fusion puis de suivi des boites qui englobent ces zones en mouvement. Cette dernière étape nous a permis d'avoir une bonne détection de la personne en mouvement dans l'image.

3.2.4/ LES ATTRIBUTS DE BAS NIVEAU

Cette phase consiste à extraire les informations géométriques de l'image binaire, résultat des étapes précédentes. Nous avons défini un ensemble F de $\varphi = 14$ attributs géométriques largement utilisés dans la littérature et détaillés au premier chapitre. Ces valeurs sont ne sont extraites que pour la région sélectionnée en fin de suivi. Elles sont calculées pour chaque image, afin de prendre en compte ensuite leurs variations au cours du temps :

- La hauteur B_h, la largeur B_w et le rapport $B_r = \frac{B_w}{B_h}$ de la boite englobante : ces 3 grandeurs peuvent être très utiles dans notre cas puisqu'elles comportent des informations sur la posture de la personne.
- Les coordonnées (C_x, C_y) du centre de la boite englobante : ces valeurs servent à estimer la position de la personne dans l'image. Si la position en elle-même de la personne n'est pas obligatoirement une information pertinente (la détection de chute doit pouvoir fonctionner quelle que soit cette position), les variations temporelles de ce centre peuvent s'avérer très discriminantes.
- Les valeurs maximums des projections horizontale et verticale des histogrammes (H_{ph}, V_{ph}). Utilisés par Yong dans [152], ces attributs contiennent des informations concernant l'orientation des mouvements verticaux et horizontaux les plus importants dans l'image.
- Les coordonnées du centre de l'ellipse (E_x, E_y) qui englobe les pixels en mouvement dans l'image, restreints à la région détectée par le suivi. Ces coordonnées sont définies par l'équation 1.10.
- Les moments centraux d'ordre 0, 1 et 2 (m_{00},m_{11},m_{02},m_{20}) : ils sont donnés, dans leur version discrète et dans la boite englobante, par l'équation 3.1. Ils contiennent en eux-mêmes des informations géométriques importantes et permettent aussi de définir l'orientation de la distribution de pixels en mouvement (figure 3.6).

$$m_{p,q} = \sum_{x=x_{min}}^{x_{max}} \sum_{y=y_{min}}^{y_{max}} (x - E_x)^p (y - E_y)^p I(x,y) \qquad p,q = 0,1,2 \tag{3.1}$$

- L'orientation de l'ellipse (E_o) : donne une information concernant l'inclinaison de la personne par rapport à l'horizontale. Une orientation proche de 0 (l'horizontale), ou une variation brutale de cet attribut peuvent logiquement être liée à une chute. Ce paramètre a notamment déjà été utilisé par Foroughi dans [44], Liao dans[83] et Rougier dans [117].

L'ensemble complet des attributs de bas-niveau est donc :

$$F = \{B_h, B_w, B_r, C_x, C_y, E_x, E_y, V_{ph}, H_{ph}, m_{00}, m_{11}, m_{02}, m_{20}, E_o\}\,. \tag{3.2}$$

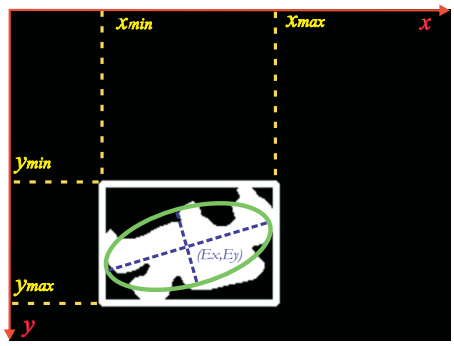


FIGURE 3.6 – Zone délimitée par la boite englobante utilisée pour le calcul des moments centraux.

Il est à noter que certains de ces attributs sont très proches les uns des autres. C'est le cas notamment des coordonnées du centre de l'ellipse qui sont en principe similaires à celles du centre de la boite englobante. Nous avons toutefois choisi de laisser le système de classification déterminer automatiquement quelles seront les informations les plus pertinentes à retenir par la suite.

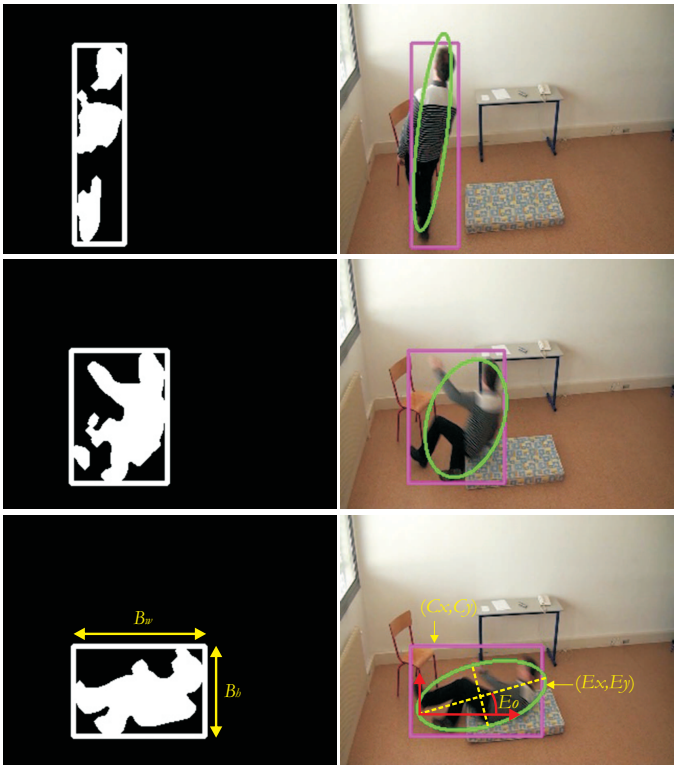


FIGURE 3.7 – Exemples de segmentation initiale et extraction des attributs géométriques de bas-niveaux.

Un exemple de segmentation zone fixe/zone en mouvement ainsi que quelques uns des différents attributs de bas-niveau qui leur correspondent sont illustrés figure 3.7. La figure

montre 3 images : le début, le milieu et fin d'une chute. En début de chute, la personne est encore debout et la hauteur de la boite englobante est supérieure à sa largeur. En milieu de chute ces grandeurs sont devenus presque égales. La fin de la chute se caractérise par une largeur plus grande que la hauteur, lorsque la personne est complètement allongée sur le sol. On peut également constater que l'orientation de l'ellipse varie significativement. Nous pouvons noter au passage l'efficacité de la phase de fusion des rectangles en mouvement qui a permis de définir la boite englobante correctement malgré les manques importants présents au niveau des jambes de la personne dans les deux premières images.

3.3/ ANALYSE DANS LE DOMAINE TEMPOREL

3.3.1/ ANALYSE PRÉLIMINAIRE

Des exemples de la variation de ces attributs au cours du temps sont présentés dans les figures 3.8 et 3.9 pour deux vidéos ne contenant pas de chute (colonnes droites) et pour deux vidéos de chutes (colonnes gauches). Ces séquences vidéo illustrent différentes activités : entrer dans la scène, marcher, chuter, rester immobile, s'asseoir, se relever et s'éloigner de la caméra, marcher vers la caméra, se coucher, se relever d'un matelas. A partir de ces courbes, il est possible de supposer que les attributs qui caractérisent le mieux une chute (la partie numéro 3 dans les deux figures) sont les variations conjointes des projections horizontale des histogrammes H_{ph}, de l'ordonnée du centre de la boite englobante, C_y, et du moment m_{02}. Ces mêmes attributs peuvent caractériser aussi l'action se coucher (numérotée (8) dans la figure 3.9). Donc la caractérisation de la chute n'est pas tout a fait garantie et elle encore plus difficile lorsque les vidéos sont prises dans des lieux différents. Il apparaît toutefois extrêmement délicat, à partir de ces courbes, de définir manuellement un algorithme (par exemple de type arbre de décision) permettant de détecter les chutes de manière fiable. L'avantage de l'utilisation d'une méthode de classification supervisée sera justement d'établir, à partir de l'apprentissage, des règles ou frontières de décision de manière automatisée, y compris dans un espace de dimension élevée. L'utilisation d'une méthode de sélection automatique des attributs présentée dans la section 3.4 nous permettra d'en limiter leur nombre tout en choisissant les plus pertinents.

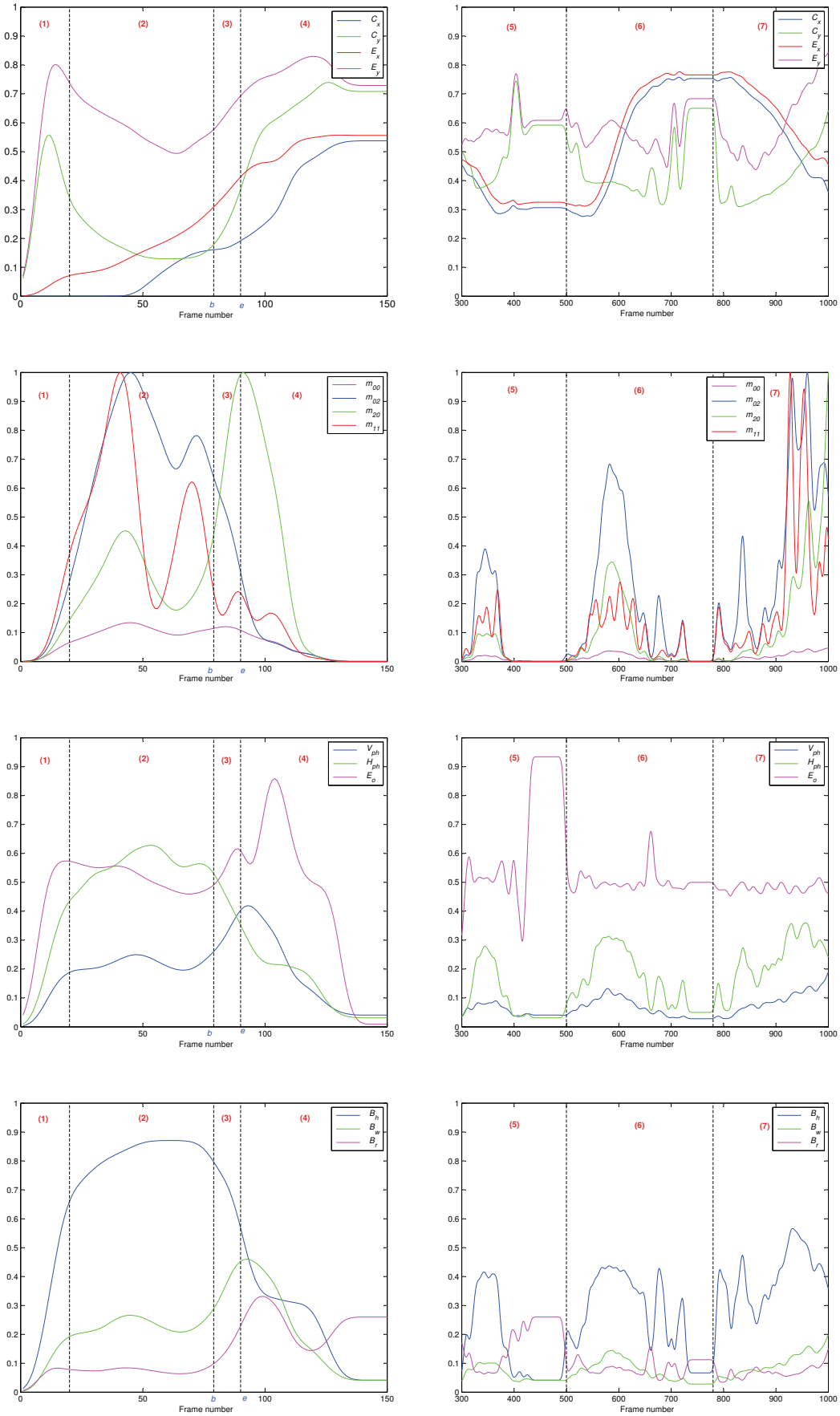


FIGURE 3.8 – Exemples de la variation des attributs de bas-niveau. (colonne gauche) Vidéos avec chute ("Coffee room"), (colonne droite) Vidéos ne contenant pas de chutes ("Lecture room"). Les activités numérotées dans les figures sont (1) entrer dans la scène, (2) marcher, (3) chute, (4) rester immobile, (5) marcher puis s'asseoir, (6) se lever puis s'éloigner de la caméra, (7) marcher en s'approchant de la caméra.

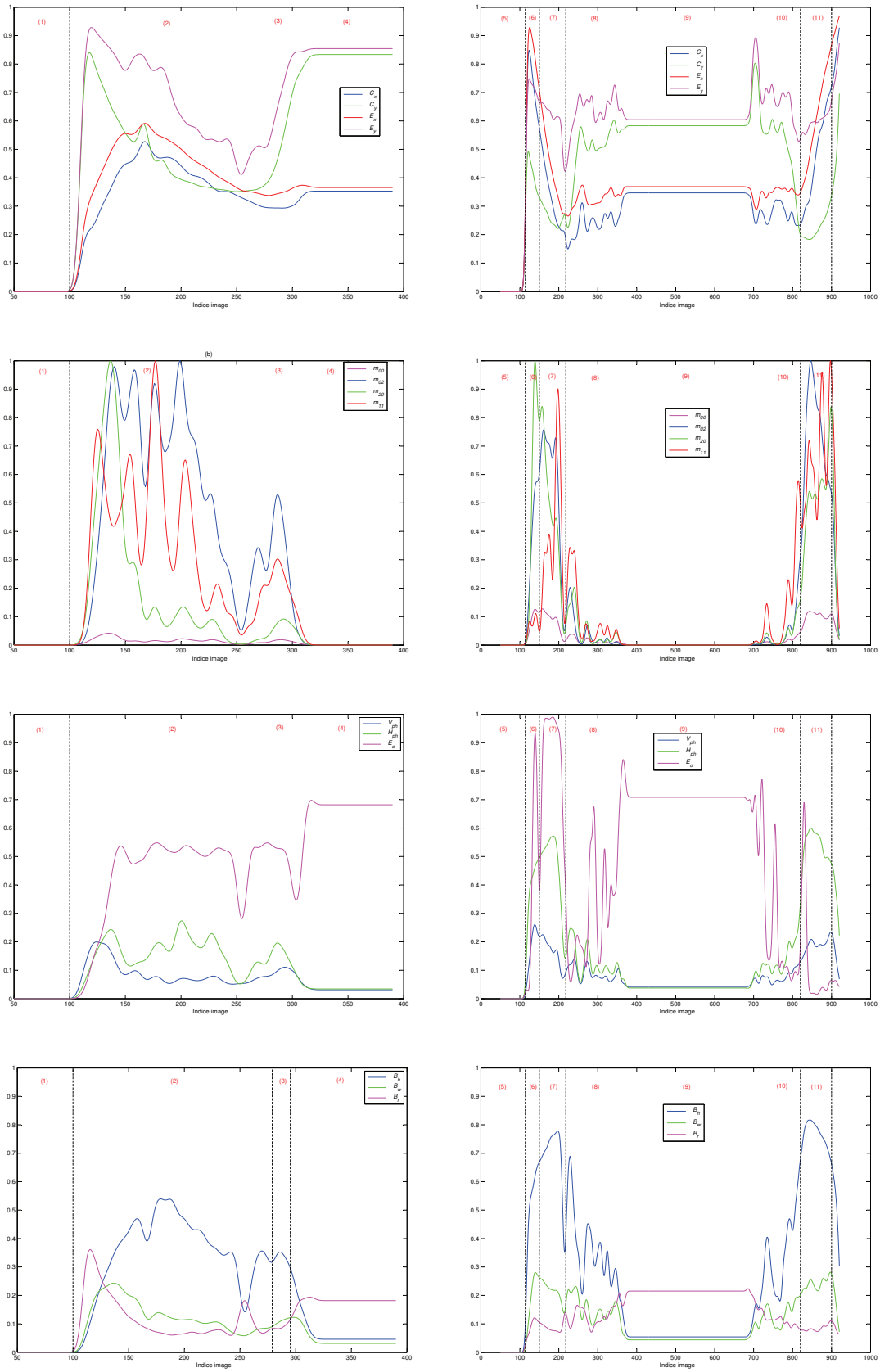


FIGURE 3.9 – Exemples de la variation des attributs de bas-niveau. (colonne gauche) Vidéos avec chute (dans "Lecture room"), (colonne droite) Vidéos ne contenant pas de chutes (dans "Office"). Les activités numérotées dans les figures sont (1) et (5) la scène est vide, (2) entrer dans la scène et marcher loin de la caméra, (3) chute, (4) rester immobile, (6) entrer dans la scène, (7) marcher, (8) se coucher, (9) rester immobile, (10) se relever d'un matelas, (11) marcher.

3.3.2/ DÉFINITION DES DESCRIPTEURS $STHF$

Ces attributs caractérisent à la fois les informations spatiales et temporelles puisqu'ils sont déterminés à partir de la détection des objets en mouvement. Toutefois, une chute comme tout un autre mouvement humain, se caractérise par la variation de position. Il est donc logique d'analyser notamment vitesse et accélération des positions (ou d'autres attributs) pour caractériser la chute. Pour cela, nous avons choisi d'effectuer une analyse temporelle sur une fenêtre glissante des attributs de bas niveau. Cette fenêtre glissante sera de taille w_f qui représentera le nombre d'images successives qui seront étudiées simultanément. Pour chacune de ces fenêtres, nous construisons un descripteur qui sera utilisé comme vecteur d'entrée du classifieur. Le descripteur complet, noté $STHF$, est défini en cinq parties : les attributs de base, leur dérivées première et seconde par rapport au temps, leurs transformées de Fourier et en ondelettes, de la manière suivante :

Soit f_t^i la valeur de l'attribut géométrique de base numéro i déterminé pour l'image numéro t d'une vidéo donnée, pour $i = 1, ..., \varphi$, si l'on suppose disposer de φ attributs de base.

- La première partie du descripteur, notée X est l'ensemble de ces φ valeurs mesurées sur w_f images successives : $X = \left\{ f_i = \{f_t^i\}_{t=1}^{w_f} \right\}_{i=1}^{\varphi}$
- La deuxième partie est le résultat de la dérivée première appliquée à l'ensemble des φ signaux du descripteur X, durant w_f images. C' est un ensemble de $w_f \times \varphi$ valeurs. Cet opérateur est noté FD où :

$$FD(X) = \left\{ \alpha_i = \left\{ (f_t^i - f_{t+1}^i) \right\}_{t=1}^{w_f} \right\}_{i=1}^{\varphi}. \tag{3.3}$$

- La troisième partie est le résultat de la dérivée seconde : le même principe que ci-dessus est appliqué pour cet opérateur notée SD. Le résultat est aussi un ensemble de $w_f \times \varphi$ valeurs.

$$SD(X) = \left\{ \delta_i = \left\{ (\alpha_t^i - \alpha_{t+1}^i) \right\}_{t=1}^{w_f} \right\}_{i=1}^{\varphi}. \tag{3.4}$$

- La quatrième partie est basée sur la transformée de Fourier rapide : nous avons appliqué l'algorithme standard de la FFT aux signaux de X, calculée sur une fenêtre de largeur w_{tf}. Notons ξ_f le module de chacun des coefficients résultats de cette transformée. Nous obtenons au total un ensemble de $w_{tf} \times \varphi$ coefficients :

$$TF(X) = \left\{ \xi_i = \left\{ \xi_f^i \right\}_{f=1}^{w_{tf}} \right\}_{i=1}^{\varphi}. \tag{3.5}$$

- La dernière partie est basée sur la transformée en ondelettes : nous avons appliqué la transformée basée sur les ondelettes D_4 orthogonales de Daubechies [31], afin de déterminer l'ensemble de $w_W \times \varphi$ valeurs qui correspondent aux coefficients d'ondelettes C_k. Cet opérateur est noté W :

$$W(X) = \left\{ C_i = \left\{ C_k^i \right\}_{k=1}^{w_W} \right\}_{i=1}^{\varphi}. \tag{3.6}$$

Le descripteur spatio-temporel $STHF$ (équation 3.7) complet est vu comme un vecteur de dimension $d = \varphi \times (3w_f + w_{tf} + w_W)$ qui vaut par exemple $d = 2240$ lorsque la fenêtre d'analyse est de taille égale $w_f = w_t f = w_W = 32$ et $\varphi = 14$.

$$STHF = \{\{f_i, \alpha_i, \delta_i, \xi_i, C_i\}_{i=1}^{\varphi}\}. \tag{3.7}$$

Le choix des dérivées première et seconde se justifie aisément par le fait que ces grandeurs représentent vitesse et accélération et caractériseront les mouvements dans la scène. Le fait d'utiliser ces grandeurs sur une fenêtre d'une largeur du même ordre de grandeur que le nombre d'images caractérisant une chute permettra au classifieur de classer l'événement au regard de la totalité des informations pendant cette durée. C'est aussi le choix que nous avons fait pour les autres transformées, qui ont été utilisées avec succès dans de nombreux cas de caractérisation de signaux et de reconnaissance de forme, ce qui justifie leur usage ici. En ce qui concerne les ondelettes, nous avons restreint notre application aux ondelettes de Daubechies (elles sont été utilisée avec succès par exemple par Boukhris dans [17]) essentiellement pour des raisons de temps de développement et d'étude : il ne nous a donc malheureusement pas été possible d'explorer d'autres solutions dans le cadre de ce travail.

En outre, une importante redondance entre les parties de ce descripteur peut être générée par les différents opérateurs utilisés. Par conséquent, nous avons consacré une partie importante de ce travail à la sélection des attributs les plus pertinents, sélection qui permet de réduire la dimension d du vecteur d'attributs, et présentée dans la section 4.1. L'étude de l'influence de certains paramètres du descripteur tels que w_f et w_{tf} est également proposée dans cette section.

Il est à noter que nous avons ajouté, avant calcul des différentes parties de $STHF$, une étape de pré-filtrage. En effet, étant donné que nous travaillons sur des vidéos de fréquence relativement basse (25 images/seconde) et que des artefacts de segmentation risquent de se produire lors de l'étape de la détection automatique, nous avons appliqué un filtrage des signaux relatifs à chaque attribut. Un sur-échantillonnage, un filtrage gaussien [89, 27] suivi par un sous-échantillonnage ont été appliqués et nous ont permis de réduire ces artefacts. Le filtrage gaussien permet de lisser les variations trop brutales des attributs bas-niveau, qui pourraient être dues à une mauvaise segmentation de l'image de mouvement, ou un mauvais suivi, c'est à dire une erreur au niveau des fusions des boites englobantes. Le suréchantillonnage qui le précède permet notamment d'améliorer la précision de calcul du filtrage Gaussien réalisé sur le signal discret. Cette étape de pré-filtrage sera évaluée dans la section 4.1.

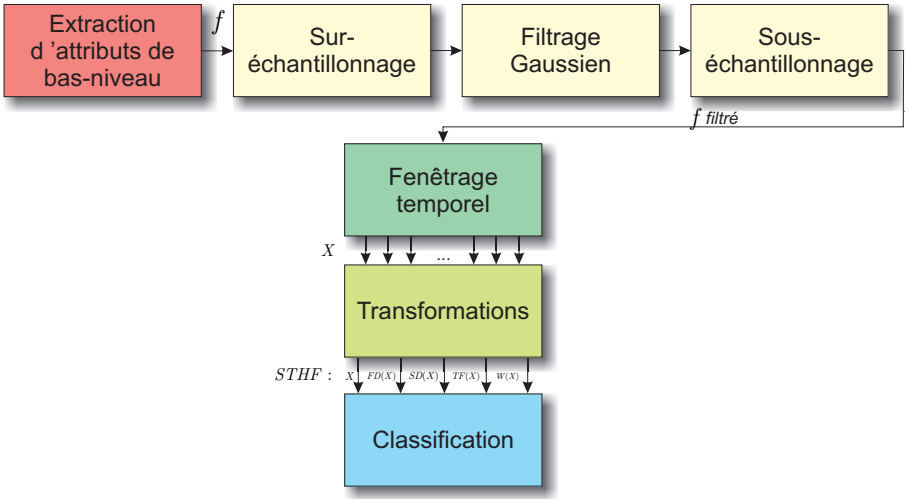


FIGURE 3.10 – Construction du descripteur spatio-temporel sur une fenêtre glissante : fenêtrage temporel.

3.4/ SÉLECTION D'ATTRIBUTS AVEC LES SVM

3.4.1/ PRINCIPE

Nous avons donc construit pour chaque image un descripteur spatio-temporel qui tient compte de la variation temporelle des 14 attributs de bas niveau définis précédemment sur une fenêtre d'analyse de taille w_f. Or, pour tout problème de classification, les attributs pertinents ne sont pas connus à priori. Parmi les attributs choisis, certains peuvent alourdir le traitement sans apporter forcément une information pertinente, voire en dégradant les performances finales.

Nous avons donc choisi de réaliser une sélection automatique d'attributs afin de réduire la dimension du descripteur que nous avons défini et de ne garder que les attributs les plus significatifs tout en maintenant un taux de classification élevé. Dans de nombreux cas, cette sélection, en construisant des descripteurs optimisés, permet d'augmenter la précision du classifieur, de diminuer sa complexité et son temps de traitement et éventuellement de réduire le nombre d'échantillons nécessaires à la phase d'apprentissage.

Pour effectuer une sélection d'attributs, il existe de nombreux algorithmes généralement constitués des quatre éléments suivants [75] :

1. Une procédure de génération : elle permet de générer un nouveau sous-ensemble d'attributs U. Cette procédure débute la recherche soit avec un attribut, soit avec tous les attributs, soit avec un ensemble d'attributs pris aléatoirement.
2. Une fonction d'évaluation : cette fonction généralement notée $J()$ permet d'évaluer l'efficacité d'un sous-ensemble d'attributs. Plus sa valeur est proche de zéro, plus

le sous-ensemble est efficace.

3. Un critère d'arrêt : il permet d'établir la fin de recherche.
4. Une procédure de validation : elle permet de vérifier la validité du sous-ensemble sélectionné. En général, cela consiste à classer un échantillon test et à observer les résultats obtenus.

Étant donné un ensemble Y de d attributs, la sélection d'attributs consiste à déterminer un sous-ensemble U de d' attributs ($d' < d$) qui optimisera le fonctionnement du classifieur. Cette optimalité est évaluée à partir d'une fonction de critère $J(U)$. La plupart du temps, la recherche exhaustive, c'est à dire consistant à évaluer le critère $J(U)$ pour toutes les combinaisons possibles d'attributs, n'est pas envisageable pour des raisons de temps de calcul rédhibitoires. C'est notre cas puisque $d = 2240$. Toutefois, le nombre de combinaisons de transformations est dans notre cas seulement de 31. Nous avons donc choisi une optimisation en deux étapes :

– Sélection de la meilleure combinaison de transformations, en utilisant tous les attributs,
– Sélection sous-optimale du meilleur sous-ensemble d'attributs de bas niveau, (combinés à l'aide de la meilleure combinaison de transformations déterminée précédemment).

En ce qui concerne la fonction d'évaluation, nous avons choisi l'erreur globale de classification au niveau image basée sur les SVM. Cette méthode est en effet réputée pour ses performances, et l'utiliser pour la sélection est homogène avec le choix de l'utiliser en phase de classification finale, c'est à dire sur le système implanté en temps réel.

Il existe toute une famille d'algorithmes sous-optimaux. Nous avons retenu une variante de la méthode dite Sequential Backward Floating Selection (SBFS) [64], résumée dans la figure 3.11, et qui consiste, en partant de l'ensemble des attributs, à retirer celui dont l'absence dans le sous-ensemble de l'itération suivante maximise les performances. Il est à noter qu'à chaque itération, retirer l'attribut f_a signifie pour nous retirer de X un groupe d'attributs : l'ensemble des valeurs de cet attribut dans la fenêtre temporelle w_f , c'est à dire $f_a = \{f_t^a\}_{t=1}^{w_f}$ et les transformations correspondantes.

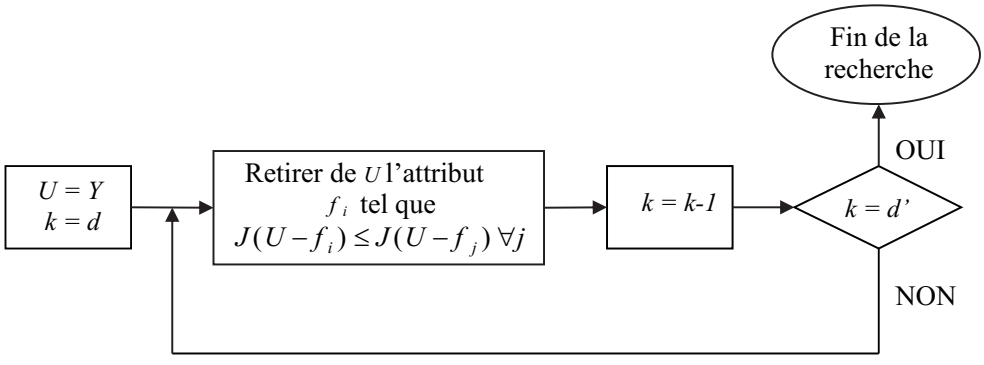


FIGURE 3.11 – Principe de la méthode SBFS.

Pour cette étape, nous avons choisi de réaliser nos expériences en utilisant le protocole $P1$. Nous avons choisi également l'annotation manuelle afin d'évaluer les attributs indépendamment de la méthode de détection de personne utilisée.

3.4.2/ SÉLECTION DE LA MEILLEURE COMBINAISON

Nous avons donc évalué les résultats au niveau classification des SVM (où l'erreur est donnée au niveau image), après un simple vote à la majorité et au niveau de décision final c'est-à-dire en utilisant la métrique que nous avons proposée section 2.5.

Pour chaque étape de ce processus, nous avons ajusté les paramètres de la méthode (la largeur w_f des dérivées première et seconde, la largeur w_{tf} de la fenêtre d'analyse de la FFT et N_W, le nombre de niveaux de la décomposition en ondelettes) et nous avons retenu les valeurs optimales pour chaque combinaison. Des exemples présentés dans les figures 3.12 et 3.13 illustrent l'influence de ces paramètres.

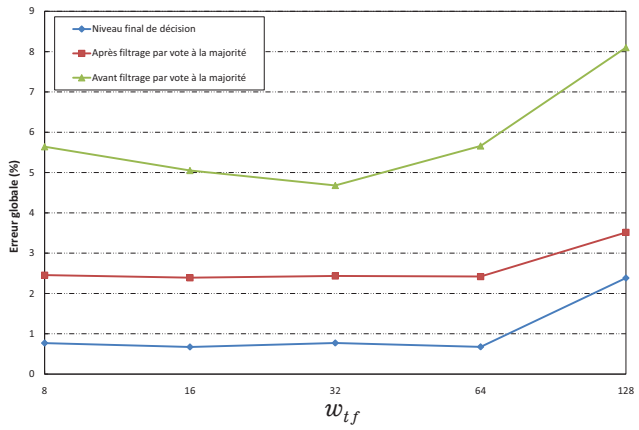


FIGURE 3.12 – Influence de la largeur de la fenêtre d'analyse de la FFT (exemple pour une combinaison d'attributs).

En ce qui concerne largeur de la fenêtre d'analyse de la FFT, nous avons régulièrement trouvé un optimum de 32 ou 64. C'est la valeur 32 qui a été retenue pour l'implantation finale, les différences entre ces deux valeurs n'étant la plupart du temps pas significatives en terme d'erreur de classification. En ce qui concerne le niveau de décomposition en ondelette, nous avons retenu la valeur 4, au delà de laquelle l'erreur avait tendance à augmenter de manière significative.

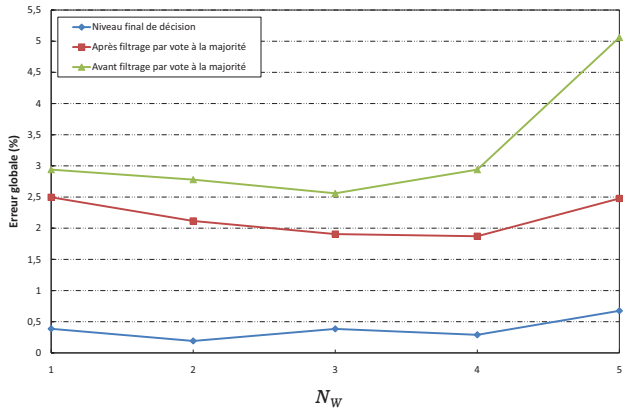


FIGURE 3.13 – Influence du niveau de décomposition des ondelettes (exemple pour une combinaison d'attributs).

Nous avons également étudié l'impact des paramètres de la métrique de tolérance au niveau de décision finale et nous avons fixé pour toutes les expériences $D_{max} = 15$ et $Z_{max} = 9$ qui donnent les meilleurs résultats. Un exemple de ce processus est présenté dans la figure 3.14 au niveau du classifieur SVM, après le filtrage par vote à la majorité et au niveau de décision finale.

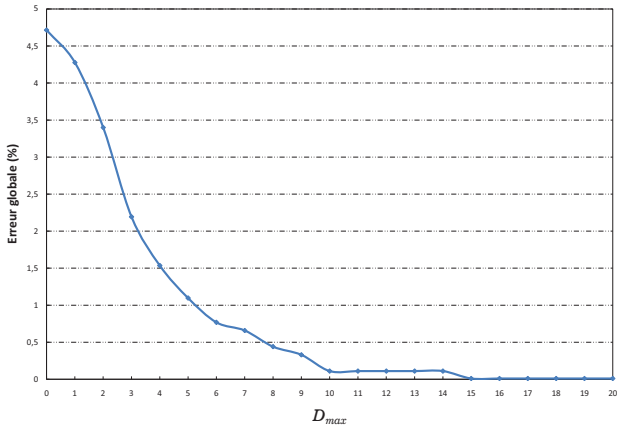


FIGURE 3.14 – Influence du paramètre de la métrique de tolérance D_{max} pour la combinaison $STHF_a = \{X, FD(X)\}$.

Les résultats détaillés sont présentés dans la table 3.2. Quatre combinaisons ont permis d'avoir un taux d'erreur global nul :

$$STHF_a = \{X, FD(X)\} \tag{3.8}$$

$$STHF_b = \{W(X), FD(X)\} \tag{3.9}$$

$$STHF_c = \{X, W(X), FD(X)\} \tag{3.10}$$

$$STHF_d = \{TF(X), W(X), FD(X)\} \tag{3.11}$$

Dans le but d'optimiser le temps global de calcul, nous avons utilisé pour les prochaines expériences ces quatre combinaisons. Pour des contraintes d'implantation temps réel, la plus simple des combinaisons pourra être retenue, c'est à dire les attributs originaux combinés avec leurs dérivées premières par rapport au temps : $STHF_a = \{X, FD(X)\}$.

En effet, au niveau de la décision des SVM, le rappel est meilleur pour les combinaisons $STHF_b$, $STHF_c$ et $STHF_d$ pour lesquelles les transformées de Fourier et en ondelettes sont appliquées, mais cette différence s'avère insignifiante au niveau de décision finale. Pour une implantation matérielle en temps réel de l'ensemble du système, il est donc possible d'éviter le calcul des transformations sans affecter les performances de la classification.

TABLE 3.2 – Performance de détection (%) de toutes les combinaisons en utilisant le protocole $P1$, à la sortie des SVM et au niveau final de décision - annotation manuelle - pré-filtrage activé.

	Perfomances des SVM					Performances au niveau final de décision				
Combinaison	E	Sp	Ac	Pr	Re	E	Sp	Ac	Pr	Re
X	3.51	97.01	96.39	92.21	94.75	0.29	99.79	99.70	96.07	98.01
TF	5.05	94.82	94.77	86.61	94.63	0.67	99.98	99.32	97.87	88.46
W	2.56	98.29	97.40	95.42	95	0.38	99.79	99.61	96.07	96.07
FD	3.1	97.29	96.84	92.59	95.56	0.29	99.79	99.71	96.07	98.03
SD	7.6	94.14	92.85	86.07	89.66	1.37	98.75	98.62	80.05	96.04
X+FD	**2.5**	**98.57**	**97.47**	**96.12**	**94.54**	**0**	**100**	**100**	**100**	**100**
$X+TF$	3.42	97.23	96.53	92.74	94.64	0.39	99.59	99.61	92.59	100
$X+SD$	2.82	98.61	97.16	96.18	93.28	0.29	99.79	99.70	96.07	98.02
$X+W$	3.11	97.50	96.84	93.42	95.10	0.19	99.79	99.80	96.15	100
$FD+SD$	3.75	97.58	96.22	93.48	92.58	0.58	99.48	99.41	90.74	98.01
$TF+FD$	3.8	96.83	96.14	91.74	94.29	0.48	99.69	99.51	94.11	96.08
$TF+SD$	3.61	97.45	96.34	93.20	93.39	0.38	99.79	99.61	96.00	96.03
$TF+W$	3.33	97.18	96.62	92.67	95.10	0.19	99.89	99.80	98.01	98.07
W+FD	**2.91**	**97.24**	**97.08**	**93.40**	**96.67**	**0**	**100**	**100**	**100**	**100**
$W+SD$	3.08	98.59	96.92	96.39	92.76	0.29	99.69	99.70	94.34	100
$X+TF+FD$	2.67	98.08	97.30	94.88	95.20	0.09	99.89	99.90	98.03	100
$X+TF+SD$	2.65	98.43	97.33	95.51	94.20	0.09	99.89	99.90	98.03	100
$X+TF+W$	3.26	96.74	96.66	91.71	96.46	0.29	99.79	99.70	96.07	98.05
X+W+FD	**2.67**	**97.72**	**97.32**	**94.47**	**96.33**	**0**	**100**	**100**	**100**	**100**
$X+W+SD$	3.01	98.01	96.98	95.05	94.42	0.29	99.69	99.70	94.34	100
$FD+SD+X$	3.26	97.94	96.70	94.44	93.39	0.19	99.89	99.80	98.08	98.01
$FD+SD+TF$	3.73	97.41	96.35	93.11	93.52	0.19	99.89	99.80	98.04	98.00
$FD+SD+W$	3.58	97.36	96.44	93.03	93.97	0.29	99.79	99.71	96.07	98.00
TF+W+FD	**3.04**	**96.71**	**96.93**	**92.28**	**97.50**	**0**	**100**	**100**	**100**	**100**
$TF+W+SD$	3.35	97.82	96.64	94.56	93.70	0.29	99.79	99.71	96.07	98.03
$X+TF+W+FD$	3.01	97.05	96.97	92.97	96.80	0.09	99.89	99.90	98.03	100
$X+TF+FD+SD$	3.66	95.93	96.32	90.61	97.27	0.09	100	99.90	100	98.02
$X+TF+W+SD$	3.35	97.82	96.64	94.56	93.70	0.38	99.69	99.61	94.23	98.05
$X+W+FD+SD$	2.97	97.09	97.01	93.08	96.80	0.38	99.79	99.61	96.01	96.08
$TF+W+FD+SD$	3.56	97.34	96.43	93.46	94.20	0.19	99.89	99.80	98.00	98.04
$X+TF+W+FD+SD$	3.29	97.58	96.66	93.59	94.19	0.29	99.79	99.70	96.07	98.03

3.4.3/ SÉLECTION DES MEILLEURES ATTRIBUTS BAS NIVEAU PAR SBFS

Nous avons représenté la variation de l'erreur de classification au cours du processus de SBFS sur la figure 3.15. Nous avons retenu que l'erreur est minimale pour $\varphi = 7$ au niveau de la décision après filtrage à la majorité (voir Figure 3.15).

L'ensemble d'attributs sélectionnés F_s et qui sera utilisé pour les prochaines expériences est donc :

$$F_s = \{C_y, m_{00}, E_x, B_r, E_o, m_{02}, H_{ph}\} \tag{3.12}$$

Le descripteur final, après sélection, est donc formé de ces $\varphi = 7$ attributs de bas-niveau sélectionnés et leurs dérivées première par rapport au temps, soit :

$STHF_{a\_SBFS} = \{F_s, FD(F_s)\}$.

La dimension finale de l'espace est $d' = 32 \times 7 \times 2 = 448$.

Nous pouvons noter au passage que la méthode de sélection automatique a sélectionné aussi bien des informations de la boite englobantes (ordonnée du centre) que de l'ellipse (abscisse du centre). Il paraîtrait logique de remplacer l'ordonnée du centre de la boite par celle de l'ellipse pour éviter le calcul de la boite englobante (L'orientation de l'ellipse est sélectionnée aussi, donc l'ellipse est nécessaire), mais l'ellipse elle-même étant calculée à partir du masque que crée la boite, nous avons besoin de cette dernière et nous ne pouvons pas, de fait, faire l'économie de son calcul. Nous avons donc conservé les attributs tel quel.

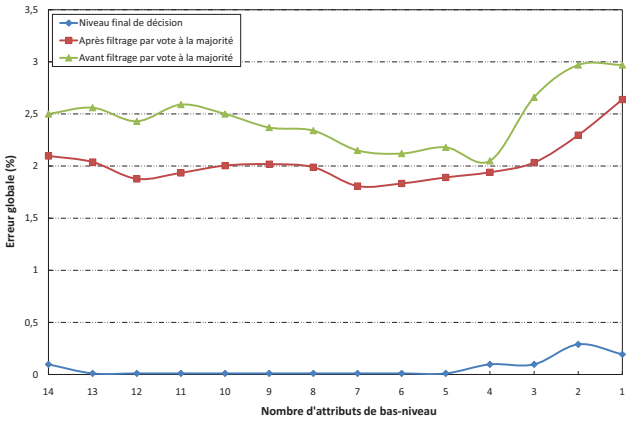


FIGURE 3.15 – Sélection d'attributs de bas-niveau pour la meilleure combinaison $STHF_a = \{X, FD(X)\}$ en utilisant la méthode SBFS .

4

EVALUATION DES PERFORMANCES - ROBUSTESSE ET COMPARAISON AVEC LE BOOSTING

4.1/ INTRODUCTION

La construction des descripteurs $STHF$ décrite dans le chapitre précédent a déjà fait intervenir une forme d'évaluation des performances de la méthode de détection, puisque ces mêmes descripteurs ont été mis au point par l'intermédiaire d'une optimisation des résultats de classification issus des SVM. Il nous reste toutefois à évaluer, grâce aux trois protocoles définis dans la section 2.4.3, la robustesse de la méthode dans son ensemble à certaines perturbations, ainsi qu'à valider expérimentalement nos choix algorithmiques, notamment concernant la partie détection automatique de la personne en mouvement, c'est à dire l'annotation automatique. D'autre part, nous avons également comparé les résultats issus des SVM avec ceux issus du Boosting, et réalisé une comparaison par rapport à certaines méthodes de l'état de l'art. Enfin, nous en viendrons à mettre l'accent sur la robustesse de la méthode au changement d'environnement.

En ce qui concerne les paramètres de la métrique d'évaluation, nous avons fixé la taille de la fenêtre w selon la durée moyenne d'une chute qui est égale à 14 pour toutes les vidéos de la base, avec un écart type de 1,4. Finalement, $w = 18$ images pour toutes les expériences qui suivront. Ainsi, toutes les images utilisées pour le test sont classées en tenant compte des informations des w images précédentes. La latence engendrée dans le cas d'une implantation temps réel est acceptable puisqu'elle serait de deux secondes (il faut ajouter la valeur de $w_f = 32$). D'autre part, la tolérance temporelle sur la détection a été fixée expérimentalement et pour toute la suite à $D_{max} = 15$, ce qui représente également moins d'une seconde de vidéo. Le paramètre Z_{max} a été fixé également grâce à des expériences préliminaires à la valeur $Z_{max} = 9$. Une optimisation de cette valeur est possible suivant les variantes des méthodes employées : il suffit alors d'utiliser l'optimum déterminé, pour la fonction de décision finalement implantée, pour obtenir des résultats cohérents.

4.2/ EVALUATION DE LA MÉTHODE : CLASSIFICATION BASÉES SUR LES SVM

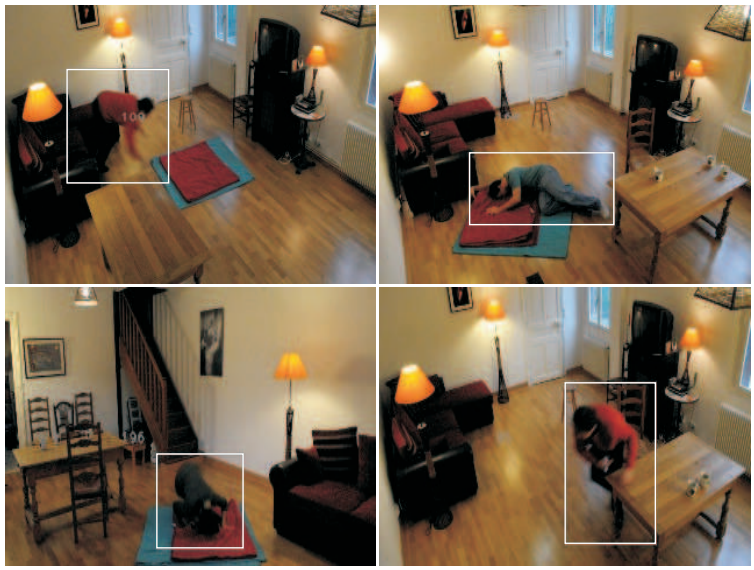
Tout au long de cette section, nous avons évalué les performances de la détection de chute avec les SVM pour le protocole $P1$, c'est à dire avec les images des sous-ensembles "Home" et "Coffee".

4.2.1/ PERFORMANCE DE L'ANNOTATION AUTOMATIQUE

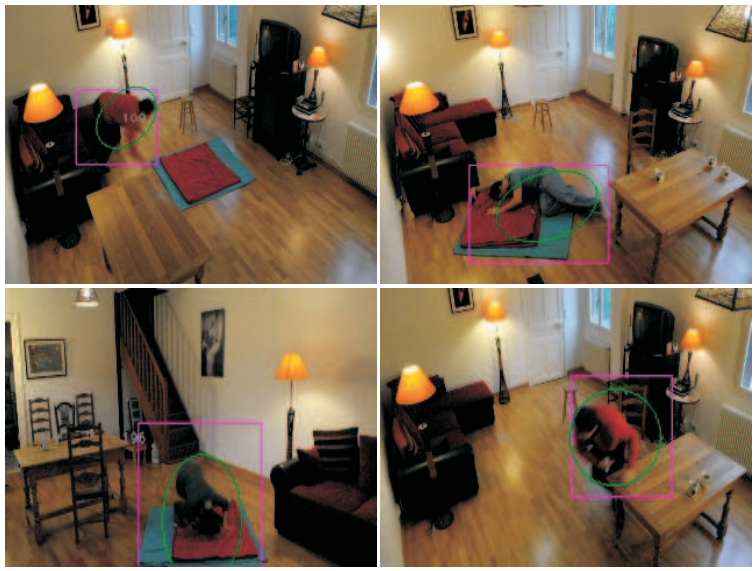
Nous avons évalué notre méthode de détection de chute pour les quatre meilleures combinaisons de transformations sélectionnées dans la section précédente, en annotation automatique, en utilisant les SVM pour la classification, le descripteur optimisé $STHF_{a\_SBFS}$. Nous avons présenté quelques exemples d'annotations automatiques et manuelles (boites englobantes et ellipses) dans la Figure 4.1. Bien qu'il y ait quelques différences visuellement repérables entre les boites englobantes automatiques et manuelles (les ellipses ne sont déterminées qu'automatiquement), ces différences semblent tout à fait acceptables. Il est toutefois bien sûr nécessaire de quantifier cette différence de manière rigoureuse, nous avons aussi reporté dans le tableau 4.1, les erreurs de classification obtenues avec annotation manuelle et celles obtenues avec l'annotation complètement automatique. Les erreurs globales de classification sont donc légèrement plus importantes pour l'annotation automatique que les erreurs en annotations manuelles, ce qui reste logique étant donné la simplicité des algorithmes de détection et de suivi que nous avons retenus. Cependant, au niveau final de décision où la tolérance sur le moment de la détection est introduite, les performances sont proches de celle en annotation manuelle, ce qui tend à valider expérimentalement la phase d'annotation automatique (les erreurs de classification sont inférieures à 1%).

Parmi ces quatre combinaisons, $X + FD$ (attributs géométriques de bas niveau et leurs dérivées premières) appliquée à F_s (les 7 attributs sélectionnés) donne toujours les meilleures performances au niveau de décision final ainsi qu'au niveau de la classification des SVM. Ceci confirme les résultats de la procédure de sélection de combinaisons, y compris en annotation automatique.

Nous avons ensuite évalué notre méthode de détection de chute en utilisant les SVM pour la classification et le descripteur optimisé $STHF_{a\_SBFS}$. Nous avons comparé les performances obtenus en utilisant la méthode de segmentation proposée par Li [81] avec la méthode que nous proposons table 4.2. Nous pouvons constater que les performances de notre algorithme malgré de sa simplicité sont nettement meilleures que celles en utilisant l'algorithme de segmentation proposé par Li[81] aussi bien au niveau de décision des SVM qu'au niveau de décision final. Ceci justifie le choix de notre méthode.



(a)



(b)

FIGURE 4.1 – Exemple d'annotations manuelles (a) et automatiques (b).

TABLE 4.1 – Performance de détection (%) à partir des SVM, du descripteur $STHF_{a\_SBFS}$, en utilisant le protocole $P1$ - Validation de l'annotation automatique.

	Annotation manuelle										Annotation Automatique									
	Décision au niveau des SVM					Niveau final de décision					Décision au niveau des SVM					Niveau final de décision				
Combinaison	E	Sp	Ac	Pr	Re	E	Sp	Ac	Pr	Re	E	Sp	Ac	Pr	Re	E	Sp	Ac	Pr	Re
$X+FD$	**2.50**	**98.57**	**97.47**	**96.12**	**94.54**	**0**	**100**	**100**	**100**	**100**	**4.94**	**87.36**	**95.22**	**87.36**	**96.41**	**0.38**	**99.61**	**94.23**	**98.00**	**99.69**
$W+FD$	2.91	97.24	97.08	93.40	96.67	0	100	100	100	100	5.04	87.51	94.93	87.51	96.08	0.48	99.51	94.34	96.15	99.69
$X+W+FD$	2.67	97.72	97.32	94.47	96.33	0	100	100	100	100	5.11	87.56	94.86	87.56	95.73	0.77	99.22	93.87	90.19	99.69
$TF+W+FD$	3.04	96.71	96.93	92.28	97.50	0	100	100	100	100	5.11	87.56	94.86	87.56	95.73	0.48	94.11	99.51	96.00	99.79

TABLE 4.2 – Performance de détection (%) à partir des SVM, du descripteur $STHF_{a\_SBFS}$. en utilisant le protocole $P1$ - Comparaison de notre méthode avec la méthode de Li[81].

	Décision au niveau des SVM					Niveau final de décision				
	E	Sp	Ac	Pr	Re	E	Sp	Ac	Pr	Re
Méthode choisie	4.94	87.36	95.22	87.36	96.41	0.38	99.61	94.23	98	99.69
Méthode de Li [81]	5.59	93.55	94.45	97.66	95.51	0.7	99.67	99.29	91.83	90

4.2.2/ EVALUATION DE L'ÉTAPE DE PRÉ-FILTRAGE

Les résultats précédents ont été obtenus en insérant l'étape dite de pré-filtrage mentionné dans la vue d'ensemble de la méthode. Il convient donc d'évaluer l'apport de ce traitement, et ce toujours dans le contexte de la validation de l'annotation automatique. Nous avons donc validé expérimentalement l'étape de pré-filtrage qui consiste à appliquer aux vecteurs attributs un suréchantillonnage suivi d'un filtrage Gaussien. Ces filtres sont appliqués sur chacun des signaux temporels des attributs bas niveau. Cette étape est souvent appliquée en reconnaissance de forme lors d'une analyse pyramidale multi-échelle des signaux [89, 27]. Le but du filtrage gaussien, ici, est de lisser les variations trop brutales des attributs bas niveau, qui pourraient être dues à une mauvaise segmentation de l'image de mouvement, ou un mauvais suivi, c'est à dire une erreur au niveau des fusions des boites englobantes. Le suréchantillonnage qui le précède permet notamment d'améliorer la précision de calcul du filtrage Gaussien réalisé sur le signal discret. Les résultats de classification (basé sur le descripteur $STHF_{a\_SBFS}$, les SVM, et la décision niveau créneau) sont reportés dans le tableau 4.3. Les expériences ont été réalisées en annotation manuelle et automatique. Dans les deux cas, les taux de détections sont améliorés par ce pré-filtrage, dont les paramètres (taux de suréchantillonnage, largeur du filtre gaussien) ont été optimisés au préalable (seuls les meilleurs résultats sont reportés ici).

Ces simples opérations ne permettent pas seulement de lisser les erreurs de l'annotation automatique mais aussi ont permis d'améliorer les performances en annotation manuelle (le rappel passe de 98% à 100%).

Dans le meilleur des cas, en annotation automatique, une seule chute n'a pas été détectée sur un total de 65 vidéos, indépendamment de la direction de la chute ou de la position de la personne dans la scène, et seuls 3 faux positifs ont été générés, dont deux concernent des entrées et sorties de la personne dans le champs de la caméra, cas que l'algorithme n'a pas encore pris en compte de manière spécifique.

TABLE 4.3 – Performance de détection (%) à partir des SVM, du descripteur $STHF_{a\_SBFS}$, au niveau final de décision et en utilisant le protocole $P1$ - Apports du pré-filtrage Gaussien

	Annotation manuelle		Annotation Automatique	
	Sans pré-filtrage	Avec pré-filtrage	Sans pré-filtrage	Avec pré-filtrage
Sp	100	**0**	99.69	**99.69**
Ac	99.90	**100**	99.32	**99.61**
Pr	100	**100**	94.00	**94.23**
Re	98.00	**100**	92.15	**98.03**
E	0.097	**0**	0.67	**0.38**

4.2.3/ EVALUATION DE L'INFLUENCE DE LA RÉSOLUTION DES IMAGES

Les résultats précédents ont tous été obtenu pour la résolution d'image choisie au départ, soit 320×240, typiquement utilisée dans les travaux antérieurs de détection de chutes. Toutefois, afin de valider ce choix expérimentalement, nous avons évalué les performances de la méthode appliquée à une résolution inférieure, ce qui pourrait permettre d'accélérer les temps de calcul ou de gagner en terme de ressources matérielles nécessaires. Nous avons donc sous-échantillonné les vidéos à la résolution 160×120 et évalué le système. Les résultats sont comparés tableau 4.4 : nous constatons une légère dégradation des performances par rapport à la résolution d'origine, notamment au niveau de la précision. Ceci est logique étant donné la perte d'information au niveau des images et de précision des attributs bas-niveau. En comparaison avec les résultats obtenus pour la résolution d'origine, cette perte n'a aucune influence sur la détection des chutes proprement dite (une seule chute n'a pas été détectée) mais on compte 5 faux positifs de plus. Étant donné ces erreurs, nous n'avons pas retenu cette résolution dégradée pour les expériences suivantes.

TABLE 4.4 – Robustesse de détection des SVM (%) au changement de résolution, au niveau final de décision et en utilisant le protocole $P1$ - Annotation automatique - Pré-filtrage activé.

	160×120	320×240
Sp	99.04	99.69
Ac	98.98	99.61
Pr	85.71	94.23
Re	97.95	98.03
E	1.01	0.38

4.3/ ÉVALUATION DE LA MÉTHODE : CLASSIFICATION BASÉE SUR ADABOOST

Comme nous l'avons vu dans le premier chapitre, l'algorithme Adaboost, lorsqu'il est basé sur des classifieurs faibles de type simple seuil, effectue conjointement l'apprentissage et la sélection des attributs. Nous avons donc dans un premier temps utilisé comme vecteur d'entrée du classifieur, le descripteur complet $STHF$, c'est à dire un vecteur de dimension $d = 2240$. Comme pour les SVM, nous avons évalué les performances de la méthode en utilisant les annotations manuelle et automatique, avec et sans les opérations de pré-filtrage , en nous basant toujours sur le protocole $P1$. Les résultats (décision au niveau créneau) sont présentés dans la table 4.5.

Nous retrouvons les grandes tendances obtenues avec les SVM. Ainsi, en annotation automatique, les performances sont légèrement inférieures à celle obtenues en annotation manuelle. Toutefois, la différence est encore acceptable, ce qui permet de confirmer la robustesse de l'annotation automatique utilisée pour la méthode de détection de chutes : l'erreur globale est de 0,33% en annotation manuelle alors qu'elle est égale à 0,58% en annotation automatique.

Les résultats reportés dans ce tableau nous permettent encore de valider expérimentalement l'étape de pré-filtrage, puisque les performances sont systématiquement améliorées lorsque ce filtre est appliqué : nous notons un net gain au niveau du rappel (environ 2% en annotations manuelles et automatiques). En annotation manuelle, l'erreur globale passe de 0,48% à 0,33% et de 1,07% à 0,58% en annotation automatique.

TABLE 4.5 – Performance de détection (%) à partir du Boosting, du descripteur $STHF$ au niveau final de décision et en utilisant le protocole $P1$ - Apports du pré-filtrage Gaussien.

	Annotation manuelle		Aannotation automatique	
	Sans pré-filtrage	Avec pré-filtrage	Sans pré-filtrage	Avec pré-filtrage
Sp	99.69	**99.76**	99.38	**99.79**
Ac	99.51	**99.66**	98.92	**99.42**
Pr	94.11	**96.07**	88.46	**95.91**
Re	96.00	**98.00**	90.19	**92.15**
E	0.48	**0.33**	1.07	**0.58**

En ce qui concerne la classification basée sur Adaboost, toujours pour le protocole $P1$ qui présente 65 vidéos au total, parmi les 50 vidéos de chutes, seules deux chutes n'ont pas été détectées et aucune fausse détection n'a été générée.

A titre de synthèse, nous avons regroupé dans le tableau 4.6 les meilleurs résultats obtenus avec les SVM et le Boosting, en annotation automatique puisque c'est évidement celle qui nous intéresse *in fine*. Au niveau de la décision finale et avec pré-filtrage, on note donc une erreur de 0.58% pour le boosting contre 0% pour les SVM. Cette perte

de performances est acceptable si on considère le gain potentiel en terme de temps de traitement ou de ressources matérielles.

Malgré la simplicité de sa fonction de décision, la classification basée sur le Boosting donne des performances encourageantes, qui pourront être adaptée à une implantation temps réel où la décision serait reportée sur une architecture matérielle. Les performances obtenues montrent également l'efficacité de notre descripteur spatio-temporel $STHF$ à distinguer les chutes des autres activités.

Il est à noter que lors de cette expérience (protocole $P1$), seul le descripteur complet $STHF$ a été utilisé à l'entrée du Boosting, alors qu'il existe bien d'autres variantes possibles, notamment l'usage du descripteur après sélection par les SVM ($STHF_{a\_SBFS}$). Ces points seront abordés dans les évaluations de la robustesse au changement d'environnement, section 4.5 .

TABLE 4.6 – Comparaison des performances de détection des SVM et du Boosting (%), au niveau final de décision et en utilisant le protocole $P1$ - Annotation automatique - Pré filtrage activé.

	SVM	Boosting
Sp	99.69	99.79
Ac	99.61	99.42
Pr	94.23	95.41
Re	98.03	92.15
E	0.38	0.58

4.4/ COMPARAISON AVEC LES MÉTHODES DE L'ÉTAT DE L'ART

Nous avons décrit dans le chapitre premier, un certain nombre de bases de vidéos permettant aux auteurs d'évaluer leurs méthodes de détection de chutes. Si le nombre de ces bases parait important au lecteur aujourd'hui, il est toutefois à noter qu'aucune base n'était disponible en ligne au moment où nous avons réalisé cette étude. La seule que nous ayons pu nous procurer était celle de Rougier [118]. De plus, les protocoles d'évaluation des performances n'étant la plupart du temps pas décrit en détail dans les articles, nous n'avons pu les reproduire. Il est donc délicat de comparer directement les valeurs issues de notre métrique à celles des autres auteurs, aussi les comparaisons ci-dessous sont-elles à prendre avec toutes les réserves nécessaires. Parmi les références les plus récentes et donnant les meilleures performances, Rougier a obtenu une erreur globale de 4.6%, alors que notre erreur est de 0.38%. Toutefois, ces nombres ne sont pas directement comparables, puisque nous avons évalué les performances au niveau du créneau, alors que Rougier a réalisé son évaluation au niveau vidéo. Pourtant, à notre avis, une vidéo d'une durée par exemple d'une heure, sans chute ni fausse détection, ne devrait pas compter avec le même poids qu'une vidéo d'une minute. De plus, le choix de l'orien-

tation de la caméra réalisé par Rougier (caméra équipée d'un objectif à très grand angle et fixée au plafond, donc très en hauteur et avec un point de vue en plongée prononcé), n'est pas adapté aux attributs géométriques que nous avons extraits. Nous avons évalué notre méthode à partir de leur base et obtenu dans le meilleur des cas (SVM) une erreur globale de 4%, pour un rappel de 73% et une précision de 97.7%. La différence de performances vient essentiellement de la différence de positionnement de caméra : notre caméra étant située aux alentours de 2m du sol permet de mieux capturer les mouvements verticaux qui caractérisent les chutes. Équipée d'un objectif de plus longue focale, elle est également moins sensible aux déformations des grands angles.

De même, Thome [132] a obtenu, pour des cas de chutes réelles, 82% de vrais positifs, en utilisant un groupe de plusieurs caméras, alors que nous obtenons 92% et 90% de vrais positifs, avec respectivement les SVM et le Boosting, tout en n'utilisant qu'un seul point de vue pour chaque acquisition.

Ces considérations nous ont donc conduits à finalement évaluer la robustesse au changement d'environnement, robustesse importante car dans une application réelle, il n'est pas forcément envisageable de réaliser un apprentissage complet (notamment concernant les chutes) sur le lieu d'installation final de l'équipement de vidéo-assistance.

4.5/ ROBUSTESSE DU SYSTÈME AU CHANGEMENT D'ENVIRONNEMENT

Nous avons donc évalué notre méthode de détection de chute à partir d'images enregistrées dans différents environnements. Les trois protocoles que nous avons défini dans la section 2.4.3 seront utilisés dans les expériences suivantes, dans le but d'évaluer la robustesse de la méthode de détection automatique de chute contre le changement d'environnement. Rappelons que pour le protocole $P1$, les images d'apprentissage et de test ont été acquises dans le même lieu, alors qu'au contraire, pour le protocole $P2$, aucun des lieux utilisé pour le test n'a été utilisé pour l'apprentissage, qu'il s'agisse de vidéos de chutes ou de non-chutes : les vidéos de test proviennent du sous-ensemble "Coffee room" et celles de l'apprentissage proviennent de "office" et "Lecture Room". Enfin le protocole $P3$ reprend les éléments de $P2$, en insérant dans l'apprentissage quelques vidéos ne contenant pas de chute, acquises cette fois dans les lieux de test. La robustesse des méthodes de classifications SVM et Adaboost seront comparées, en utilisant systématiquement l'annotation automatique. Les autres paramètres, notamment ceux de la métrique d'évaluation finale, restent inchangés par rapport aux expériences précédentes.

Il est important de noter que nous avons ajouté à ce niveau d'expérimentation trois variantes d'apprentissage par rapport aux précédentes évaluations. En premier lieu, et afin, pour ces trois protocoles, de valider la sélection d'attributs qui a mené à la construc-

tion des descripteurs, nous avons mesuré les performances des SVM à partir du descripteur complet $STHF$, en plus des SVM déterminés à partir du descripteur sélectionné, $STHF_{a\_SBFS}$. En second lieu, nous avons ajouté l'évaluation de descripteurs $STHF_{boost}$: il s'agit du descripteur construit après sélection d'attributs effectuée pendant la phase d'apprentissage du processus d'Adaboost. Ceci est possible, grâce à l'utilisation de simples seuils comme classifieurs faibles. La dimension du vecteur d'entée, c'est à dire le nombre d'attributs sélectionnés est alors ici de 89. Enfin, nous avons évalué les performances du Boosting appliqué non plus au descripteur complet, mais à celui sélectionné via les SVM et la SBFS, à savoir $STHF_{a\_SBFS}$. Les résultats sont résumés dans le tableau 4.7 en terme de taux de classification par créneau, donc au niveau final.

Pour toutes les expériences, et comme on pouvait s'y attendre, les résultats de classification obtenues pour le protocole $P2$ sont moins bons qu'avec le protocole $P1$, notamment au niveau du rappel et de la précision. Les SVM, utilisé avec des descripteurs sélectionnés $STHF_{a\_SBFS}$ subissent une dégradation toutefois acceptable, de l'ordre de celle que le Boosting avait par rapport aux SVM avec le protocole $P1$. On peut affirmer déjà à ce niveau que le système présente une certaine robustesse au changement complet d'environnement. En ce qui concerne le Boosting, et quelle que soit la variante de descripteurs utilisée, les dégradations sont plus importantes et deviennent inacceptables pour un système réel.

Les comparaisons des résultats obtenus pour le protocole $P3$, dans lequel seules quelques vidéos sans chutes ont été introduites, montrent de bien meilleures performances que pour le protocole $P2$. Les SVM retrouvent un niveau d'erreur et de précision proches (0.16% et 96% contre 0.38% et 94%) du protocole $P1$, tout en subissant une légère perte de rappel (90% contre 98%). Les performances du Boosting utilisé avec les descripteurs complets sont nettement améliorées.

Bien que les résultats obtenus avec les SVM après sélection par le Boosting ne soient pas meilleurs que ceux des SVM après SBFS, le Boosting après SBFS a permis d'améliorer légèrement les performances du Boosting classique (surtout en terme de rappel).

Ces deux variantes sont intéressantes sur le plan du concept, puisque faisant intervenir une sélection d'attribut et combinaison originale de classifieurs, et restent, de notre point de vue, des solutions à explorer, notamment en terme de gain potentiel de temps de calcul. En effet, à titre indicatif, nous avons déjà indiqué dans ce tableau les temps d'exécution obtenus sur une plate-forme de type PC standard. Ces données seront exploitées essentiellement dans la chapitre 5, mais nous pouvons d'ores et déjà constater la rapidité du Boosting après SBFS comparé aux SVM après SBFS (gain d'un facteur 82), et également, pour les SVM l'apport de la sélection par le Boosting comparé à la simple SBFS (gain d'un facteur légèrement supérieur à 5).

Pour conclure cette expérience, les résultats du protocole $P3$ montrent bien qu'il est bénéfique d'intégrer dans l'apprentissage des échantillons de non chutes pris dans l'en-

vironnement final. Ceci est réaliste puisque l'installateur ou l'utilisateur peut facilement enregistrer des séquences de la vie quotidienne. Ceci implique de pouvoir mettre à jour l'apprentissage sur place, grâce à une caméra intelligente embarquant ce processus, ou grâce à une caméra connectée à un réseau permettant le transfert des vidéos et/ou des attributs extraits de ces vidéos vers un serveur chargé de la mise à jour de l'apprentissage.

TABLE 4.7 – Robustesse du système au changement de lieu à la décision finale - Optimisation du descripteur.

	SVM avant SBFS			SVM après SBFS			Boosting			SVM après sélection basée Boosting			Boosting après SBFS		
Descripteur initial	$STHF$			$STHF_{a\_SBFS}$			$STHF$			$STHF_{boost}$			$STHF_{a\_SBFS}$		
Protocole	$P1$	$P2$	$P3$	$P1$	$P2$	$P3$	$P1$	$P2$	$P3$	$P1$	$P2$	$P3$	$P1$	$P2$	$P3$
Sp	99.79	99.05	98.74	99.69	99.70	99.95	99.79	99	99.73	99.88	98.69	99.87	99.69	99.35	99.67
Ac	99.23	98.24	98.44	99.61	99.54	99.83	99.42	98.33	99.32	99.56	97.57	99.58	99.51	98.03	99.36
Pr	95.74	86.84	52.08	94.23	84.84	96.55	95.91	56.41	82.14	97.91	82.92	88.88	94.11	87.87	80.64
Re	88.23	86.84	80.64	98.00	90.32	90.32	92.15	68.75	74.19	94.03	82.92	77.41	96.01	76.31	80.64
E	0.77	1.75	1.55	0.38	0.45	0.16	0.58	1.55	0.67	0.43	2.42	0.41	0.48	1.96	0.63
Temps(μs)	8590.47	7901.39	11932.41	248.61	128.16	216.08	3.63	3.27	3.64	42.31	33.90	40.91	2.63	2.56	2.62

4.6/ ANALYSE DES CONFUSIONS LES PLUS FRÉQUENTES

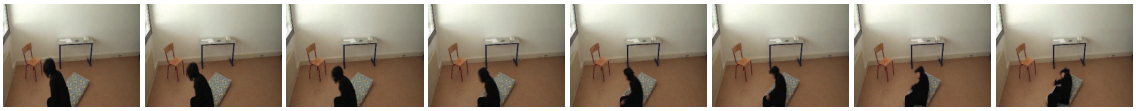
Dans le but de caractériser la méthode, et évaluer dans quelle mesure les activités de la vie quotidienne peuvent être confondues avec des chutes, nous avons classé les actions de l'ensemble des vidéos du protocole $P3$ en dix catégories (chuter, marcher, se coucher sur un matelas, se lever d'un matelas, s'asseoir, ouvrir ou fermer une porte, déplacer une chaise, se pencher, allumer la lumière, éteindre la lumière, autre). Les performances de détection sont reportées dans le tableau 4.8, pour les SVM et le Boosting. On retrouve la supériorité des SVM sur le Boosting.

Une analyse détaillée des faux positifs montre que la plupart sont générés lors d'entrées et sorties dans le champs : c'est une des améliorations à apporter à cette méthode.

Les faux négatifs obtenus (chutes non détectées) sont soit des chutes acquises dans l'axe de la caméra (figure 4.2) soit représentent des situations très difficiles. Dans la première chute, la personne est proche de la caméra et surtout partiellement hors champs, ce qui induit une détection de forme englobante partiellement erronée. La deuxième chute se déroule très loin de la caméra, les attributs sont presque inchangés durant la chute. Dans le dernier cas, la chute de la personne s'est déroulée très lentement (en deux temps), les variations temporelles sont plutôt proches de celles d'une personne qui se couche. Un système à deux caméras pourra donc améliorer les performances et rendre la méthode plus robuste. Nous pouvons également relever un faux positif induit par l'allumage de la lumière. Cette action provoque une variation globale de l'intensité dans toute l'image, que nous pourrons détecter aisément en amont pour éviter ce type de cas particulier qui peut s'avérer fréquent.

TABLE 4.8 – Nombre de faux négatifs et de faux positifs par action. Na est le nombre total d'actions.

	Chute FN/Na	Marche FP/Na	Coucher FP/Na	Lever FP/Na	Asseoir FP/Na	Lever d'une chaise FP/Na	Porte FP/Na	Chaise FP/Na	Pencher FP/Na	Éteindre FP/Na	Allumer FP/Na	Autre FP/Na
SVM	3/31	0/107	1/12	0/14	0/17	0/20	0/12	0/12	0/9	0/4	1/4	0/54
Boosting	6/31	4/107	2/12	0/14	0/17	0/20	0/12	0/12	0/9	0/4	0/4	0/54



(a) Chute dans l'axe de la caméra et partiellement hors champs.

(b) Chute dans l'axe de la caméra et très loin de la caméra, les variations temporelles sont indétectables.

(c) Chute en deux temps dans l'axe de la caméra, le mouvement est quatre fois plus lent que les autres chutes.

FIGURE 4.2 – Exemples de chutes non détectées par le système.

5

ADÉQUATION ALGORITHME ARCHITECTURE : IMPLANTATION SUR CAMÉRA INTELLIGENTE

D'une manière simplifiée, l'implantation d'un algorithme consiste à affecter aux différentes unités de traitement disponibles les différentes tâches à réaliser. Cette répartition s'effectue en tenant compte de la dépendance entre ces tâches. Dans le cas où l'architecture cible est conçue spécifiquement pour exploiter le parallélisme potentiel de l'application, les performances en termes de temps de calcul peuvent alors être grandement améliorées mais cela au détriment du temps de développement. L'architecture ou l'algorithme peuvent être éventuellement modifiés pour optimiser le compromis entre le coût matériel nécessaire et les performances désirées. La phase de définition de l'algorithme de détection de chutes a été réalisée de manière séparée à l'implantation matérielle afin d'obtenir les meilleurs performances en termes de détection possible tout en conservant toujours la perspective d'une intégration sur un système embarqué. En effet, l'objectif final de ces travaux est de concevoir une plate-forme permettant d'une part de reconnaître automatiquement les situations de chutes et d'autre part de transmettre l'information utile à l'extérieur de l'environnement sous surveillance. Cette information utile peut se restreindre uniquement à une notification indiquant une chute potentielle ou inclure la transmission de la vidéo compressée de la scène observée. Suivant la configuration choisie, la vidéo pourrait être diffusée en continu ou encore uniquement lors de la détection de chute afin de limiter l'intrusion dans l'environnement de la personne observée. Dans tous les cas, la plate-forme doit être très performante pour permettre l'intégration de la détection de chutes et éventuellement celle de la compression mais suffisamment compacte pour faciliter son insertion à l'intérieur de l'environnement de la personne sous surveillance. L'utilisation d'un système embarqué de type caméra intelligente prend alors tout son sens. Ce type de caméra associe en effet au sein d'un même système, acquisition vidéo et capacités de traitement. L'implantation sur ce type de plate-forme est pour cela proposée dans ce chapitre. Cependant afin de faciliter l'implantation sur une cible matérielle quelconque, une étude plus générale visant à diminuer la complexité de l'algorithme en dégradant de manière très limitée les performances de détection est tout d'abord proposée.

5.1/ ANALYSE DES TEMPS DE CALCUL ET OPTIMISATIONS

Comme présenté dans les chapitres précédents, les vidéos acquises à l'aide de la caméra ont été décompressées, et sous-échantillonnées en résolution 320×240. Pour chaque image, une image *silhouette* a été créée représentant les pixels en mouvement. Nous avons montré dans le chapitre précédent l'apport en terme de taux de reconnaissance d'une part de la sélection des transformations et des attributs et l'apport des opérateurs de filtrage (filtrage médian et dilatation/érosion) d'autre part. L'analyse des temps de calcul pour chacune des grandes étapes composant l'algorithme de détection est présenté dans le tableau 5.1. Les temps présentés ne prennent pas en compte le temps d'acquisition obtenu à l'aide de la fonction d'OpenCV, qui représente environ 70ms, temps qui a été masqué grâce à une implantation multi-thread.

TABLE 5.1 – Temps de traitement des différentes tâches en ms, hors temps d'acquisition sur PC standard.

	$STHF$	$STHF_{a\_SBFS}$
Soustraction de fond	0.5	0.5
Médian et morpho. math.	4	4
Construction du descripteur	0.9	0.3
Décision des SVM	8.6	0.25
Décision du Boosting	0.003	0.002
Temps total SVM	14	5.05
Temps total Boosting	5.403	4.802

Nous pouvons d'abord constater le bienfait de la sélection d'attribut en terme de temps de calcul, puisque le temps de décision des SVM passe de 8.6ms à seulement 0.25ms avec les attributs sélectionnés. Nous pouvons encore noter que le temps d'exécution de la phase de décision du Boosting est beaucoup plus rapide que celle des SVM. Toutefois, les performances en termes de classification restant parfois significativement en retrait par rapport aux SVM, nous n'avons pas systématiquement mesuré ses temps d'exécution dans les expériences à suivre. Il est aussi à noter que le descripteur $STHF$ utilisé pour la méthode basée Boosting exige un calcul complet de tous les attributs de bas-niveau et surtout de toutes les transformations, notamment FFT et ondelettes, ce qui ralentit le système (le temps de calcul de toutes les transformations est 0.9ms pour le $STHF$ et 0.3ms pour $STHF_{a\_SBFS}$).

D'autre part, il apparaît que l'étape de filtrage médian suivi des opérations morphologiques est la plus gourmande en ressources, puisqu'elle demande environ 80% du temps dans le cas des descripteurs $STHF_{a_SBFS}$. Or cette étape, même si elle a été validée expérimentalement dans la section 3.2.3 en termes de performances de segmentation, peut encore être étudiée en terme d'adéquation algorithme architecture. Enfin, nous avons choisi également de présenter à ce niveau les résultats des expériences menées en utili-

sant pour la soustraction du fond la méthode proposée par L. Li [81] et disponible dans la bibliothèque OpenCV, afin de trouver le bon compromis temps-taux de reconnaissance.

5.1.1/ IMPACT DE LA MÉTHODE DE SOUSTRACTION DU FOND

Nous avons donc comparé la méthode de soustraction de fond proposée par L. Li [81] avec la méthode de segmentation simple présentée en section 3.2.1, basée sur une soustraction entre l'image courante et un fond régulièrement mis à jour à l'aide d'un buffer circulaire. La méthode de Li est basée essentiellement sur la décision Bayésienne. Cet algorithme qui a fait ses preuves pour des vidéos contenant différents types de difficultés comprend quatre parties : détection des pixels en mouvement, classification, segmentation d'objet et apprentissage et maintenance du fond. Les performances de détection de chutes obtenues après segmentation par les deux méthodes sont résumées dans le tableau 5.2.

TABLE 5.2 – Evaluation en termes de performances de détection (%) et temps de traitement (ms) pour deux méthodes de segmentation.

		Méthode de Li [81]	Méthode choisie
Performances de détection	Sp	99,67	**99,69**
	Ac	99,29	**99,61**
	Pr	91,83	**94,23**
	Re	90	**98**
	E	0,70	**0,38**
Temps de traitement (ms)	Soustraction du fond	21,5	**0,5**
	Total	26,70	**5.05**

Nous pouvons constater que la classification basée sur notre algorithme (qui comprend le suivi), sont aussi bonnes que la méthode de Li, voire légèrement meilleures pour la précision, le rappel et l'erreur globale, mais surtout 43 fois plus rapide (0.5ms contre 21.5ms). L'utilisation de notre méthode permet d'avoir un temps global d'exécution de 5.05ms sur un PC standard. Étant donné les différences importantes en terme de performances de classification et de temps d'exécution, nous avons donc conservé la méthode simple de détection/suivi pour le reste des évaluations et optimisations présentées ci-après.

5.1.2/ INFLUENCE DES OPÉRATEURS DE MORPHOLOGIE MATHÉMATIQUES

Le tableau 5.3 présente les performances de classification et les temps de calcul obtenus avec les SVM, sans filtrage d'abord, puis avec le simple filtre médian (dont on a fait varier la taille de 3x3 à 7x7), sans opération de morphologie mathématique, et finalement nous avons reporté les performances et temps avec le filtre complet utilisé jusque-là. Sans filtrage, le nombre de rectangles présents avant fusion et suivi augmente, ce qui donne

un temps de calcul certes court (3.34ms), mais surtout des performances de classification significativement dégradées. L'optimum en termes de performances de classification est clairement le filtre complet que nous avions retenu jusqu'à maintenant. Toutefois, nous pouvons constater que si l'on supprime les opérations de morphologie mathématiques et que l'on diminue la taille du filtre, les performances restent acceptables, alors que les temps de calculs sont très fortement réduits. Ainsi, le filtre médian seul, taille 3x3, semble être un très bon compromis temps/performances.

TABLE 5.3 – Influence des opérations de filtrage (Median et morphologie mathématique) - Compromis Performances de détection/Temps de traitement.

		Sans filtrage	Filtrage Median			Median et Morpho. Math.
			3×3	5×5	7×7	
Performances de détection	Sp	100	**99,76**	99,65	99,64	**99,69**
	Ac	99,3	**99,56**	99,34	99,55	**99,61**
	Pr	100	**96**	94	99,23	**94,23**
	Re	84,21	**96**	94	98	**98**
	E	0,69	**0,44**	0,65	0,447	**0,38**
Temps de trait. (ms)	Filtrage	0	**0,32**	1.40	3.17	**4**
	Total	3,34	**1.40**	2.45	4.22	**5.05**

5.2/ IMPLANTATION SUR UNE PLATE-FORME DE TYPE SMART CAMÉRA

Traditionnellement la vision par ordinateur met en œuvre un système d'acquisition et une plate-forme PC standard. La caméra réalise l'acquisition des images et l'ordinateur joue le rôle d'une unité de traitement mais aussi de plate-forme de développement. Cependant, l'évolution exponentielle des circuits intégrés a contribué à la réalisation, à coûts acceptables, de différents systèmes regroupant les deux fonctionnalités au sein d'un même dispositif. Les interactions entre acquisition et traitement sont bien entendu facilitées (par exemple la sélection de zones d'intérêts dans les images dynamiquement en fonction des résultats de calcul). En outre, des unités de calcul autre que programmables (par exemple de type configurable comme les FPGAs) peuvent être utilisées et permettent alors d'améliorer les performances des systèmes. Ainsi concevoir ces systèmes s'avère très utile pour les applications de vision où des contraintes de temps de traitement doivent être respectées. On parle de systèmes embarqués pour la vision intelligente. Dans ce contexte, étudier les composants matériels et connaître les contraintes des dispositifs utilisés s'avèrent primordiaux afin de réaliser l'implantation des traitements sur ces systèmes.

5.2.1/ INTÉRÊT ET PRINCIPE D'UNE SMART CAMÉRA

Une caméra intelligente est un système de vision, qui en plus des tâches traditionnelles d'acquisition et de stockage des images, est capable d'exécuter des tâches de traitement d'images complexes tels que des opérations d'analyse d'images et de reconnaissance de formes. Une caméra intelligente intègre dans un même boîtier un capteur vidéo un dispositif de communication et un module de traitement. Son architecture matérielle illustrée en figure 5.1 est décomposée en trois modules :

– Acquisition : composé essentiellement d'un capteur d'images (CCD et CMOS). Cependant, afin d'obtenir des informations supplémentaires sur la scène, d'autres types de dispositifs sensoriels peuvent être intégrés.
– Traitement : se charge de l'exécution du traitement de l'information. Les résultats sont ensuite envoyés vers un système hôte, et/ou être utilisés pour contrôler l'acquisition de nouvelles données ;
– Communication : assure la connexion avec le monde extérieur (système hôte ou réseau).

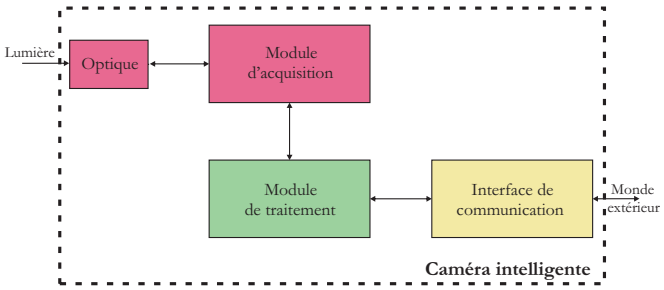


FIGURE 5.1 – Schéma synoptique simplifié d'une caméra intelligente.

5.2.2/ MODULE D'ACQUISITION

Les types des capteurs images les plus couramment utilisés sont CMOS et CCD.

– La technologie CCD [1], dite aussi dispositif à transfert de charge. Elle repose, en effet, sur le principe du transfert des charges accumulées par les photodiodes (zone photosensible du pixel) vers les pixels voisins. Une amplification et un échantillonnage sont appliqués à la dernière photodiode. La synchronisation de ce transfert grâce à un circuit de contrôle qui pilote la lecture des charges ligne par ligne sur l'intégralité de la matrice. Cette technologie présente notamment un rapport signal sur bruit important permettant une haute qualité d'image et une bonne sensibilité à la lumière liée à l'uniformité des images.

1. Charge Coupled Device

– La technologie CMOS [2] possède de nombreux avantages par rapport aux technologies concurrentes qui a permis à cette technologie d'être largement actuellement la plus répandue. Elle propose ainsi une faible consommation d'énergie par rapport à la technologie CCD. De plus, elle permet un accès aléatoire aux pixels du capteur ce qui fondamental pour minimiser les transferts d'information dans le cas de sélection de zones d'intérêt dans l'image.

5.2.3/ Module de traitement embarqué

L'architecture interne des dispositifs de traitement est composée d'opérateurs logiques et arithmétiques, modules de synchronisation et de contrôle des opérations, registres et des mémoires internes. Il existe trois principales catégories de composants utilisés. On trouve les composants de type ASIC [3] possédant une architecture complètement figées et exécutant un algorithme fixe, les modules programmables possédant une architecture, certes figée, mais permettant de modifier l'algorithme à exécuter tels que les processeurs génériques ou les processeurs spécifiques à l'instar des processeurs dédiés au traitement du signal (DSP [4]) et finalement les composants reconfigurables, tels que les circuits CPLD [5] et les FPGA [6], pour lesquels l'architecture doit être conçue en relation avec l'algorithme. Le choix du type de composant est lié principalement aux contraintes de l'application en terme de flexibilité d'utilisation ou de performances de calcul. Les caméras intelligentes peuvent aussi intégrer plusieurs composants, de nature différente, afin de tirer profit des avantages de chacune de ces cibles matérielles. Ce type d'architecture est alors nommée hétérogène.

Un ASIC est une solution particulièrement efficace en termes de performance de calcul et de consommation d'énergie. Cependant, un ASIC n'est capable de réaliser que les tâches pour lesquelles sa structure a été conçue. Son architecture est définie à bas niveau (au niveau silicium) en exploitant l'exploitant les spécificités de l'application de manière à obtenir le compromis recherché en termes de temps de calcul, surface de silicium utilisée (autrement dit les ressources matérielles) et consommation. Le manque de flexibilité représente une principale limite. Finalement le temps de conception ainsi que le coût financier sont très important pour la phase de prototypage. Ainsi en terme financier, la fabrication d'un ASIC ne devient généralement rentable que lorsque, en production, le nombre de composants dépasse plusieurs dizaines ou centaines de milliers.

Au sein des caméras intelligentes, les composants de programmable les plus utilisés sont les processeurs de type RISC (Reduced Instruction Set Computer) ainsi que les DSP. Leur architecture permet d'implanter n'importe quel algorithme (dans le cas la mé-

2. Complementary Metal Oxide Semiconductor
3. Application Specific Integrated Circuit
4. Digital Signal Processor
5. Complex Programmable Logic Device
6. Field Programmable Gate Array

moire externe disponible est suffisante) ce qui offre bien entendu une flexibilité extrême pour s'adapter aux spécificités de l'application. Cependant, l'architecture est figée mais par le fait l'adaptation au parallélisme des tâches ne peut s'effectuer qu'en fonction des ressources matérielles disponibles. Pour autant, certains processeurs de type DSP permettent l'exécution d'opérations (bien souvent arithmétiques) en parallèle ce qui permet d'exploiter, au moins en partie, le parallélisme de tâches de l'application.

L'utilisation des circuits FPGA a connu une évolution très rapide durant cette dernière décennie. Cette technologie est ainsi utilisée dans les applications de la vision intelligente du domaine industriel ou académique, mais aussi dans un grand nombre d'autres domaines [72][80][48] à côté des applications de la vision intelligente . Le composant propose à la fois l'équivalent d'une grande quantité de logique (jusqu'à plusieurs centaines de millions de portes logiques) mais aussi de la mémoire embarquée, un grand nombre de multiplieurs 18bitsx18bits (au maximum plusieurs milliers), des cœurs de processeurs, des liaisons séries à très haute vitesse (limitée à 13 Gbits/s par lien) et finalement un grand nombre d'entrées/sorties utilisateur. L'ensemble de ces ressources est ainsi associé selon les besoins de l'application visée. Le composant est donc reconfiguré en fonction de l'application désirée. Le degré d'utilisation du parallélisme de l'application est à la charge du concepteur. Les performances en termes de temps de calcul vont ainsi dépendre des ressources matérielles mises en jeu. Le temps de développement est par conséquent plus long que sur des cibles de type programmable. Ces circuits bénéficient des dernières technologies disponibles dans le domaine de la conception de circuit puisqu'il existe des familles conçues en 28 nm.

5.2.4/ MODULE DE COMMUNICATION

Le but principal des caméras ordinaires est de fournir le flux vidéo acquis. En raison des résolutions spatiales, de la fréquence image et de la dynamique du pixel couramment utilisés, il est bien souvent nécessaire de transmettre une quantité importante de données. Dans le cas de la caméra intelligente, plusieurs configurations en termes de transfert de données sont envisageables et influent directement sur le choix des interfaces de communication. Le flux sortant peut être constitué soit des images d'origines, des images traitées, ou de paramètres physiques mesurés directement à partir des images acquises. Lorsque le flux se restreint exclusivement aux paramètres physiques alors son volume est particulièrement faible (quelques centaines d'octets par trame). Au contraire, si l'application nécessite l'association des images d'origines et celles d'images traitées, la bande passante nécessaire est au minimum doublée. Ainsi, les caméras intelligentes proposent une grande variété interfaces de communication. Toutes les liaisons série et parallèle standards (RS232, Ethernet, PCI, Camera-Link...) sont couramment implantées sur ces systèmes. Le choix de l'interface dépend de l'application visée. La difficulté majeure rencontrée se situe au niveau de temps de développement nécessaire à l'intégra-

tion d'interfaces de communication complexes au sein de tels systèmes. L'utilisation d'un processeur associé à un système d'exploitation (OS) permet de grandement diminuer ce temps de développement.

5.2.5/ SÉLECTION DE LA PLATE-FORME DE PROTOTYPAGE

Les caméras intelligentes sont très répandues aussi bien dans la recherche académique qu'au niveau industriel. Nous présentons dans ce paragraphe quelques exemples issus de l'état de l'art. Nous présentons exclusivement des caméras intelligentes présentant un avantage ou utilisées dans un cadre proche de notre application. L'objectif étant clairement de proposer un système opérationnel à partir d'une plate-forme de prototypage déjà disponible. Le temps de transfert de notre algorithme depuis la plate-forme PC vers la caméra intelligente est aussi clairement un des critères essentiels de notre sélection. En terme d'interface de communication, une liaison Ethernet sera privilégiée afin de pouvoir éventuellement transmettre les résultats de la détection de chutes à un central d'observation et de décision.

Le laboratoire Le2i possède par lui même une forte expérience dans le domaine des caméras intelligentes. Ainsi, Romuald Mosqueron et al. [101] ont proposé une caméra intelligente et rapide, basée sur l'association d'un capteur CMOS et d'un FPGA. Différents algorithmes ont été implantés au sein du FPGA. Ainsi l'implantation d'un algorithme de compression a permis de réduire un large flux de données (6.55 Gbps) et propose un transfert de données via une simple liaison USB 2.0. Un autre algorithme permet d'extraire des attributs (détection de contours) ou encore de suivre des objets. L'architecture proposée permet de traiter 500 images par seconde pour la résolution 1280×1024. L'utilisation d'une cible de type reconfigurable rend tout à fait possible l'intégration d'algorithmes de traitement d'images tel que la détection de chutes. Cependant le temps de développement sera, selon notre expérience, très certainement de l'ordre de plusieurs mois. En effet, l'algorithme C/C++, faisant appel à la bibliothèque OpenCV, doit alors être traduit en langage VHDL [8], un langage plus proche de l'architecture matérielle.

Une double problématique apparaît : sélectionner la puissance de calcul suffisante mais aussi limiter le temps de portage sur la plate-forme recherchée. L'utilisation de FPGA permet d'accélérer les temps de calcul des parties régulières de l'algorithme, celle des unités programmables, de part leur généricité, permet de limiter le temps de portage.

La INCA+ (Intelligent Camera), conçue par Kleihorst [76], est une caméra autonome qui réalise la tâche de détection des visages en temps réel. La plate-forme proposée est produite par Philips CFT. Un capteur CMOS, un processeur parallèle pour la capture des pixels et un DSP pour la programmation de haut niveau forment ensemble un dispositif idéal pour la détection de visages. Le pré-traitement bas-niveau est pris en charge par un

processeur parallèle Xetal [77] en mode SIMD [7]. Le traitement de niveau plus haut, c'est à dire la détection et la reconnaissance, est établi par un DSPcore de haute performance TriMedia [1]. Des solutions similaires mais plus récente et plus performante existent. Ainsi, la XIMEA CURRERA-G figure 5.2 représente un système de vision complet auquel il est possible d'intégrer tous les périphériques standard (clavier, souris, écran..) et qui est compatible avec la majorité des bibliothèques de traitement d'image et de vision dont OpenCV. Elle offre une très bonne résolution qui peut atteindre 5 megapixels, des processeurs AMD-Fusion qui forment la combinaison d'un CPU x86 et d'un processeur graphique (GPU). Cette famille de caméra intelligentes intègrent Gigabit Ethernet (GigE), des interfaces USB, VGA, RS232 et 8 entrées sorties numériques isolées ainsi qu'un contrôleur logique programmable temps réel (PLC). Cette solution, bien que relativement onéreuse, permettrait d'atteindre notre objectif de portage de l'algorithme de détection avec les contraintes formulées.

FIGURE 5.2 – XIMEA CURRERA-G .

Les caméras intelligentes possédant une architecture hétérogène,associant FPGA et processeur, représentent aussi des solutions pouvant potentiellement répondre à notre problématique. D'une manière simplifiée, les parties régulières des algorithmes sont implantées sur les cibles de type FPGA alors celles irrégulières sur le processeur. On peut citer notamment Fleck [41] qui a proposé une caméra intelligente capable de suivre les objets en temps réel. Elle ne nécessite pas une large bande-passante pour la transmission puisqu'elle envoie uniquement l'information de haut niveau d'abstraction, le résultat de suivi. La caméra mvBlueLYNX 420CX de chez Matrix Vision **www.matrix-vision.com** représente la base de ce travail. Elle est constituée d'un capteur CCD (résolution VGA), un FPGA (Xilinx Spartan-IIE FPGA utilisé pour les traitements de bas niveau), un processeur (200 MHz Motorola PowerPC), avec MMU et FPU, 32 MB de SDRAM, 36 MB de FLASH et une interface Ethernet (100 MBit/s pour la transmission des résultats du suivi). Une solution similaire est proposée par Mosqueron [100] utilisant un processeur Nexperia et FPGA Virtex2 Pro. Ces solutions permettent d'allier la puissance de calcul d'unités configurables pour les parties régulières de l'algorithme à la flexibilité d'un processeur. Ces solutions apparaissent intéressantes cependant l'utilisation immédiate de la librai-

7. Single Instruction Multiple Data

rie OpenCV n'est pas assurée. En effet, cela nécessite alors l'intégration d'un système d'exploitation associée à une distribution logicielle adaptée.

Les progrès des systèmes microélectroniques offrent la possibilité d'intégrer dans un même composant une unité de calcul hétérogène. En effet, des unités de calcul de type programmable peuvent être disponibles au sein de certains FPGA. Cette solution très prometteuse, permet à l'instar des autres solutions hétérogènes d'embarquer des algorithmes de traitement d'images complexes. De plus, le déploiement d'un système d'exploitation est facilité par la mise à disposition par la communauté d'un grand nombre outils et de tutoriels sur ces cibles récentes. L'utilisation du système d'exploitation facilite non seulement la mise en œuvre des interfaces de communications (pour des interfaces standards dont les pilotes sont disponibles) mais permet aussi une utilisation directe de certaines librairies comme OpenCV. Nous avons ainsi sélectionné un FPGA de type Zynq qui associe deux cœurs matériels ARM Cortex-A9 aux ressources habituelles du FPGA. Cette solution utilisant la technologie commerciale la plus récente (28 nm) allie puissance de calcul et la flexibilité requise. Nous allons détailler son architecture et sa mise en œuvre dans le paragraphe suivant.

5.2.6/ ARCHITECTURE ZYNQ ET PERFORMANCES OBTENUES

Ce FPGA peut être considéré comme le premier "System on Chip" (SoC) commercial regroupant un FPGA, plusieurs processeurs ARM Cortex-A9 (dont la fréquence peut atteindre 1GHz) et un grand panel d'interfaces de communication (GigE, USB, SPI, CAN, SDRAM interface, etc...). L'architecture globale est présentée figure 5.3. Des modules de traitement ou des contrôleurs supplémentaires d'interfaces de communication peuvent être inclus à cette structure en utilisant les ressources logiques disponibles du FPGA (représenté en couleur jaune sur la figure 5.3. La communication entre les processeurs et ses composants utilisateur s'effectue via un bus AMBA [8] et des liaisons normalisées de type AXI. Cette communication est transparente pour le concepteur, les outils de développement permettant la génération automatique, lors de l'insertion du composant à la structure, des fonctions de transfert de données classiques de type lecture ou écriture. Finalement, ce composant, de part la technologie utilisée (28 nm), est moins consommateur en énergie que les FPGA de technologie précédente.

Nous avons utilisé la carte d'évaluation ZC702 incluant notamment un composant de type zynq "XC7Z020 CLG484 -1 AP SoC", 1 Go de mémoire SDRAM DDR3 et une interface giga-ethernet (GigE). Un double cœur ARM Cortex-A9 est disponible et fonctionne à la fréquence maximale de 667 MHz. Un noyau Linux est déployé sur ce double cœur. Il s'agit d'un kernel tree 3.6 intégrant les pilotes des composants Xilinx et Analog Device. En terme de distribution logicielle, une version de Linaro Ubuntu ARM a été déployée. Elle

8. Advanced Microcontroller Bus Architecture

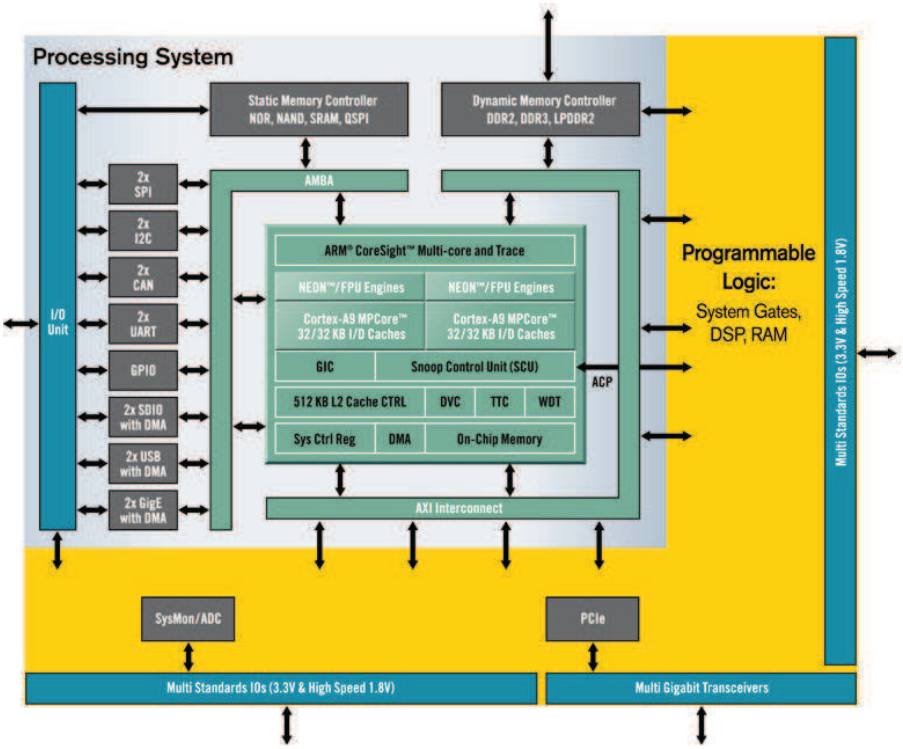


FIGURE 5.3 – Zynq.

permet une utilisation immédiate des interfaces utilisateurs classiques (clavier, souris), l'utilisation des interfaces HDMI, USB et Ethernet mais aussi de la librairie de traitement d'images OpenCV. Une web-cam a pu être connectée via un port USB et permet l'acquisition d'un flux vidéo continu à 25 images/s avec une résolution spatiale de 320×240 pixels. Le couple caméra-SoC permet ainsi de proposer une plate-forme de prototypage à bas coût.

Les performances du système ont été évaluées avec la version modifiée décrite en section 5.1.2.

TABLE 5.4 – Temps de traitement des différentes tâches en ms sur plate-forme Zynq, hors temps d'acquisition.

	$STHF_{a\_SBFS}$
Soustraction de fond	4.45
Médian 3×3	17.16
Construction du descripteur	4.53
Décision des SVM	8.67
Temps total	34.81

Le temps de calcul, obtenu avec cette implantation purement logicielle sur cible Zynq, augmente significativement en comparaison de l'implantation sur plate-forme PC. Ce-

pendant, l'implantation proposée permet de traiter jusqu'à 28 images par seconde. Les temps présentés ne prennent pas en compte le temps d'acquisition obtenu à l'aide de la fonction d'OpenCV, qui représente environ 22ms, temps qui a été masqué grâce à une implantation multi-thread. Nous avons donc démontré la faisabilité du portage de l'algorithme de détection de chutes sur une cible de type SoC. De plus, l'analyse du temps de calcul nécessaire à chacune des étapes de l'algorithme, permet de remarquer que la fonction réalisant le filtrage médian demeure, par rapport aux autres étapes, relativement longue. Des optimisations demeurent ainsi possible avec l'implantation matérielle de certaines fonctions, tel que le filtrage médian ou la décision des SVM, en utilisant les ressources propres au FPGA. La fonction de décision des SVM peut ainsi être remplacée par celle du Boosting comme nous l'avons montré section 5.1. Des développements antérieurs, nous ont permis d'obtenir aussi pour du filtrage médian possédant un noyau de filtrage de taille 3x3 [100] ou encore pour l'implantation de décision basés sur le Boosting [98] que leur temps de calcul pouvait être accélérer de manière très significative à l'aide d'une implantation matérielle basée sur une cible FPGA .

5.3/ CONCLUSION ET PERSPECTIVES

Nous avons démontré la possibilité de porter l'algorithme de détection de chutes sur un système embarqué. Bien que les performances en termes de vitesse de traitement soient réduites à l'heure actuelle de part notre implantation purement logicielle, par rapport à l'implantation sur plateforme PC, la cible choisie permet d'envisager, grâce aux ressources logiques propres du FPGA, d'accélérer de manière significative la fréquence du système. L'utilisation du Boosting comme classifieur à la place des SVM permettrait là encore d'accélérer la fréquence de détection (surtout si le classifieur est implanté sur la partition matérielle c'est à dire avec les ressources logiques). Outre la compacité d'un tel système, l'intégration potentielle de détection comme pré-traitement contrôlant les performances d'encodage avec codec vidéo permet d'envisager une utilisation attractive pour l'utilisateur car très peu intrusive dans son environnement quotidien.

CONCLUSION

Nous avons au cours de ce travail de thèse contribué à la détection de chute de personnes en temps-réel dans trois directions : construction et annotation d'un ensemble de vidéos destinées à l'évaluation de systèmes de détection automatique de chute, définition et évaluation des descripteurs de chutes de personnes, implantation matérielle de l'algorithme de détection sur une plate-forme de prototypage de caméras intelligentes.

De l'état de l'art concernant la reconnaissance d'action de manière générale, nous avons réalisé une synthèse en extrayant des méthodes courantes les grandes étapes les plus communément utilisées. En nous basant sur les points forts de ces différentes étapes, ainsi que sur nos contraintes d'implantation temps réel, nous avons établi les grandes lignes d'une méthode supervisée de détection de chute adaptée à une implantation matérielle, ceci par exemple dans le but de réduire la bande passante finale nécessaire à la transmission des données vers l'extérieur (compression vidéo adaptative). Les différentes contraintes ont notamment orienté notre choix vers les méthodes de détection des silhouettes humaines basées sur le mouvement qui est bien adaptée aux systèmes monoculaires à caméra fixe.

Nous avons alors analysé comment les auteurs, dans la littérature, évaluaient généralement leurs méthodes, à la fois en terme de bases de vidéos existantes, mais aussi en termes de méthodologie d'évaluation de performances. En effet la chute d'une personne âgée est une action complexe en raison du nombre important de scénarios possibles. Elle est souvent réalisée dans un environnement sans contraintes dans lequel on trouve différents types de difficultés. Nous avons mis en évidences les avantages et inconvénients des travaux antérieurs, ce qui nous a conduits à construire notre propre base de vidéos $DSFD$, qui comporte un grand nombre d'annotation manuelles propices aux comparaisons de méthodes, et que nous avons rendue disponible en ligne dans ce but. Afin d'évaluer la robustesse de la méthode au changement de point de vue et d'environnement, les vidéos ont été enregistrées dans quatre locaux différents.

Nous avons défini trois protocoles d'évaluation grâce à cette base, ainsi qu'une métrique d'évaluation que nous avons introduite afin d'évaluer les performance du système s'adaptant à la nature du flux vidéo analysé et la durée d'une chute, et en tenant compte des contraintes temps réel. Nous avons introduit une tolérance d'une seconde environ entre la détection réelle d'une chute et la détection attendue. L'intérêt d'avoir détaillé cette métrique est notamment de permettre maintenant des comparaisons plus aisées entre méthodes de différents chercheurs qu'il ne l'était possible auparavant, sous réserve bien

entendu que d'autres chercheurs utilisent le même algorithme...

Cette méthode est basée sur des descripteurs spatio-temporels que nous avons mis au point au regard de l'état de l'art et en les sélectionnant automatiquement grâce à une méthode de classification supervisée qui a largement prouvé ses bonnes performances dans la littérature, les SVM.

La métrique définie et la base de vidéo établie, il nous a alors été possible de définir complètement la méthode de détection automatique de chute, constitué de la phase de détection de la personne en mouvement, son suivi, l'extraction des attributs géométriques de bas niveau, leur analyse temporelle, la classification au niveau image des descripteurs résultants par une méthode de classification supervisée, puis finalement la décision au niveau créneau.

Une des principales contributions de cette méthode est la construction des descripteurs spatio-temporels $STHF$, qui sont calculés à partir de la géométrie de la forme en mouvement dans la scène, de leurs dérivées premières et secondes par rapport au temps, et leurs transformées de Fourier et en ondelettes. Une recherche exhaustive de la meilleure combinaison de transformations suivie par une méthode de sélection SBFS, permet de construire le descripteur $STHF_{a\_SBFS}$ optimisé en terme d'erreur de classification par les SVM. Les résultats expérimentaux nous permettent de conclure que la meilleure combinaison d'attributs est constituée de la variation temporelle de 7 attributs géométriques de base, combiné avec les variations temporelles de leurs dérivées premières. Finalement, l'analyse de la trajectoire et de la déformation de la forme en mouvement sont réalisées directement par le classifieur, laissant ce dernier prendre en compte l'aspect temporel de la chute, grâce à l'apprentissage.

Tout au long de la mise en place de cette méthode, nous avons tenté, dans la mesure du possible, d'optimiser les différents paramètres qui la caractérisent, comme la taille du filtre médian utilisé en fin de filtrage, ou encore les tailles des différentes fenêtres temporelles utilisées pour les analyses des signaux. Une des faiblesses de notre méthode est en effet de comporter un grand nombre de paramètre qu'il conviendrait d'optimiser de manière encore plus approfondie à chaque étape.

Une autre contribution significative de notre travail est l'évaluation des performances de la méthode, en utilisant d'abord uniquement les SVM puis en effectuant une comparaison par rapport à la classification basée sur le Boosting. Nous avons, entre autre, validé expérimentalement notre approche de détection automatique de la personne en mouvement en la comparant aux résultats obtenus avec une annotation manuelle. Enfin nous avons surtout évalué la robustesse du système au changement d'environnement entre l'apprentissage et la décision. Nous avons ainsi montré expérimentalement q'il suffit de mettre à jour l'apprentissage avec quelques vidéos sans chutes, enregistrées dans l'environnement définitif, pour obtenir des performances de classification acceptables. Une analyse des confusions les plus fréquentes a été menée, montrant que le système peut

encore être amélioré en prenant en compte en amont, c'est à dire au niveau de la détection de la personne en mouvement, des cas particuliers comme l'entrée et la sortie dans le champs de la caméra, ou l'extinction générale de la lumière.

Globalement, le meilleur résultat obtenu grâce aux SVM est un seul faux positif constaté sur l'ensemble des vidéos, alors que toutes les chutes sont détectées. Lorsque le test est réalisé sur des vidéos d'un environnement qui n'a jamais été appris, le nombre de faux positifs passe à 4. Dans le cas du Boosting, les performances finales sont très proches des SVM, (rappel : 86%, précision : 96%) et beaucoup mieux adaptées à une implantation matérielle de type FPGA, puisque la fonction de décision est constituée de simples comparateurs qui peuvent être largement parallélisés.

Enfin, nous avons réalisé l'intégration de la détection de chute sur plate-forme matérielle basée sur le composant Zynq. L'idée essentielle de ce développement réside dans le prototypage rapide de la méthode sur une plate-forme embarquée et autonome. Le composant Zynq possède d'une part une architecture hétérogène qui allie un processeur double-cœur et de la logique configurable et d'autre part propose un grand panel d'interfaces de communication. Ainsi, le déploiement d'un OS de type Linux associé à une distribution logicielle Linaro Ubuntu sur cette cible a permis la réutilisation du code original en C/C++ et des bibliothèques associées (principalement OpenCV) ainsi qu'une gestion simplifiée des interfaces de communications. Le système proposé, assimilable à une caméra intelligente, est composé d'une webcam et d'une carte d'évaluation intégrant un composant Zynq. Suivant une démarche de type Adéquation Algorithme Architecture, nous avons adapté l'algorithme afin d'obtenir un bon compromis performances de classification/temps de traitement, ce qui nous a permis d'obtenir une intégration performante (environ 10 images/s) de la méthode proposée sur ce système. Des perspectives intéressantes demeurent naturellement dans le passage de certaines parties de l'algorithme en implantation matérielle, c'est à dire dans la partie FPGA du Zynq, dans l'objectif de traiter le flux vidéo en temps réel à 25 images/s.

BIBLIOGRAPHIE

[1] Trimedia technologies, 2003. http://www.trimedia.com.

[2] M. Alwan, P. J. Rajendran, S. Kell, D. Mack, S. Dalal, M. Wolfe, and R. Felder. A smart and passive floor-vibration based fall detector for elderly. In *Information and Communication Technologies, 2006. ICTTA'06. 2nd*, volume 1, pages 1003–1007. IEEE, 2006.

[3] D. Anderson, J.M. Keller, M. Skubic, X. Chen, and Z. He. Recognizing falls from silhouettes. In *Engineering in Medicine and Biology Society. EMBS '06. 28th Annual International Conference of the IEEE*, pages 6388–6391, 2006.

[4] M. Athans, R. Wishner, and A. Bertolini. Suboptimal state estimation for continuous-time nonlinear systems from discrete noisy measurements. *Automatic Control, IEEE Transactions on*, 13(5) :504–514, 1968.

[5] E. Auvinet, F. Multon, A. Saint-Arnaud, J. Rousseau, and J. Meunier. Fall detection with multiple cameras : An occlusion-resistant method based on 3D silhouette vertical distribution. *IEEE Transactions on Information Technology in Biomedicine*, 15(2) :290–300, March 2011.

[6] E. Auvinet, L. Reveret, A. St-Arnaud, J. Rousseau, and J. Meunier. Fall detection using multiple cameras. *Conference Proceedings of the International Conference of IEEE Engineering in Medicine and Biology Society*, 2008 :2554–2557.

[7] E. Auvinet, L. Reveret, A. St-Arnaud, J. Rousseau, and J. Meunier. Fall detection using multiple cameras. In *30th IEEE Engineering in Medicine and Biology Conference, EMBC'08*, pages 2554 – 2557, Vancouver, Canada, 2008.

[8] D.G. Bailey. *Design for Embedded Image Processing on FPGAs*. Wiley, 2011.

[9] Bar-Shalom and Fortmann. Alignement and data association. 1988.

[10] J. L. Barron, D. J. Fleet, and S. S. Beauchemin. Performance of optical flow techniques. *International Journal of Computer Vision*, 12 :43–77, 1994.

[11] M.S. Bartlett, G ; Littlewort, M. Frank, C. Lainscsek, I. Fasel, and J. Movellan. Fully automatic facial action recognition in spontaneous behavior. In *Automatic Face and Gesture Recognition, 2006. FGR 2006. 7th International Conference on*, pages 223–230, 2006.

[12] L. E Baum, T. Petrie, G. Soules, and N. Weiss. A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains. *The annals of mathematical statistics*, 41(1) :164–171, 1970.

[13] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(4) :509–522, 2002.

[14] Y. Benezeth, P. Jodoin, B. Emile, H. Laurent, and C. Rosenberger. Comparative study of background subtraction algorithms. *Journal of Electronic Imaging*, 19(3) :033003–033003–12, 2010.

[15] Selby-Silverstin L. Besser MP. Predicting fall risk in the elderly using temporal-spatial parameters of gait. In *Proceedings of the Synopsium of the International Society for Postural and Gait Research, Control of Posture and Gait, Maastricht, The Netherlands*, pages 70–73, 2001.

[16] Christopher M. Bishop. *Neural networks for pattern recognition*. Oxford university press, 1995.

[17] M. Boukhris, A.A. Mohamed, D. D'Souza, M. Beck, N.E. Ben Amara, and R.V. Yampolskiy. Artificial human face recognition via daubechies wavelet transform and svm. In *Computer Games (CGAMES), 2011 16th International Conference on*, pages 18–25, 2011.

[18] M. Boulmier. Bien vieillir à domicile : Enjeux d'habitat, enjeux de territoires. Rapport remis à Monsieur Benoist APPARU Secrétaire d'Etat au Logement et à l'Urbanisme.

[19] M. Boulmier. L'adaptation de l'habitat au défi de l'évolution démographique : un chantier d'avenir. Rapport au secrétaire d'Etat au Logement et à l'Urbanisme.

[20] A.K. Bourke and G.M. Lyons. A threshold-based fall-detection algorithm using a bi-axial gyroscope sensor. *Medical Engineering and Physics*, 30(1) :84 – 90, 2008.

[21] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[22] L Brethes, P Danes, and F Lerasle. Stratégies de filtrage particulaire pour le suivi visuel de personnes : description et évaluation. *Reconnaissance des Formes et Intelligence Artificielle (RFIA'06), Tours*, 2006.

[23] Norbert Buch, Sergio A Velastin, and James Orwell. A review of computer vision techniques for the analysis of urban traffic. *Intelligent Transportation Systems, IEEE Transactions on*, 12(3) :920–939, 2011.

[24] Y. Chen, Y. Lin, and W. Fang. A hybrid human fall detection scheme. In *Image Processing (ICIP), 2010 17th IEEE International Conference on*, pages 3485–3488. IEEE, 2010.

[25] C. Chu, J. Hwang, S. Wang, and Y. Chen. Human tracking by adaptive kalman filtering and multiple kernels tracking with projected gradients. In *Distributed Smart Cameras (ICDSC), 2011 Fifth ACM/IEEE International Conference on*, pages 1–6, 2011.

[26] J. W. Cooley and J. W. Tukey. An algorithm for the machine calculation of complex fourier series. *Mathematics of Computation*, 19(90) :pp. 297–301, 1965.

[27] J. L Crowley. A representation for visual information. Technical Report CMU-RI-TR-82-07, Robotics Institute, Pittsburgh, PA, December 1981.

[28] R. Cucchiara, A. Prati, and R. Vezzani. A multi-camera vision system for fall detection and alarm generation. *Expert Systems*, 24 :334–345, 2007.

[29] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 886–893, 2005.

[30] S. Danafar and N. Gheissari. Action recognition for surveillance applications using optic flow and svm. In *Computer Vision–ACCV 2007*, pages 457–466. Springer, 2007.

[31] I. Daubechies. *Ten lectures on wavelets*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1992.

[32] B. Daubney, D. Gibson, and N. Campbell. Monocular 3d human pose estimation using sparse motion features. In *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, pages 1050–1057. IEEE, 2009.

[33] Y. de Meneses, P. Roduit, F. Luisier, and J. Jacot. Trajectory analysis for sport and video surveillance. *Electronic Letters on Computer Vision and Image Analysis*, 5 :148–156, 2005.

[34] F. D. M. de Souza, G. Camara Chávez, E. Alves do Valle Jr., and A. de Albuquerque Araújo. Violence detection in video using spatio-temporal features. In *SIBGRAPI'10*, pages 224–230, 2010.

[35] R. O. Duda and P. E. Hart. Use of the hough transformation to detect lines and curves in pictures. *Commun. ACM*, 15(1) :11–15, January 1972.

[36] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley, New York, 2 edition, 2001.

[37] J. Meunier E. Auvinet, C. Rougier and J. Rousseau St-Arnaud. Multiple cameras fall dataset. Technical Report 1350, Université de Montréal - DIRO - LISA, 2010.

[38] W. Elhamzi, J. Dubois, J. Miteran, and M. Atri. An efficient low-cost fpga implementation of a configurable motion estimation for h.264 video coding. *Journal of Real-Time Image Processing*, pages 1–12, 2012.

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

[40] A. Fathi and G. Mori. Action recognition by learning mid-level motion features. In *In CVPR*, 2008.

[41] S. Fleck, S. Lanwer, and W. Straßer. A smart camera approach to real-time tracking. In *13th European Signal Processing Conference (EUSIPCO)*, pages 4–8, 2005.

[42] Center for Research and Prevention of Injuries-CEREPRI. *Fact sheet : Prevention of Falls among Elderly*. Europeen Network for Safety among Elderly. http ://www.euroipn.org/eunese/factsheets.htm.

[43] Jr. Forney, G.D. The viterbi algorithm. *Proceedings of the IEEE*, 61(3) :268–278, 1973.

[44] H. Foroughi, A. Rezvanian, and A. Paziraee. Robust fall detection using human shape and multi-class support vector machine. *Sixth Indian Conference on Computer Vision, Graphics and Image Processing*, pages 413–420, 2008.

[45] H. Foroughi, H.S. Yazdi, H. Pourreza, and M. Javidi. An eigenspace-based approach for human fall detection using integrated time motion image and neural network. *International Conference on Signal Processing*, pages 1499–1503, 2008.

[46] Y. Freund and R. E Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. In *Computational learning theory*, pages 23–37. Springer, 1995.

[47] Y. Freund and R. E. Schapire. Experiments with a new boosting algorithm, 1996.

[48] C. Gao, X. Liu, and X. Wang. A model for predicting the obsolescence trend of fpga. In *Reliability, Maintainability and Safety (ICRMS), 2011 9th International Conference on*, pages 1354–1358. IEEE, 2011.

[49] J. Gao, A. G Hauptmann, A. Bharucha, and H. D Wactlar. Dining activity analysis using a hidden markov model. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 2, pages 915–918. IEEE, 2004.

[50] S. Gidel. Methode de detection et de suivi multi-pietons multi capteurs embarquee sur un vehicule routier application a un environnement urbain. *These de doctorat*, 2010.

[51] A. Gilbert, J. Illingworth, and R. Bowden. Fast realistic multi-action recognition using mined dense spatio-temporal features. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 925–931, 2009.

[52] L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri. Actions as space-time shapes. *Transactions on Pattern Analysis and Machine Intelligence*, 29(12) :2247–2253, December 2007.

[53] T. Greif, R. Lienhart, and D. Sengupta. Monocular 3d human pose estimation by classification. In *Multimedia and Expo (ICME), 2011 IEEE International Conference on*, pages 1–6, 2011.

[54] J. Hammerle-Uhl, M. Karnutsch, and A. Uhl. Recognition impact of jpeg2000 part 2 wavelet packet subband structures in polar iris image compression. In *Systems, Signals and Image Processing (IWSSIP), 2012 19th International Conference on*, pages 13–16. IEEE, 2012.

[55] D. Han, L. Bo, and C. Sminchisescu. Selection and context for action recognition. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 1933–1940. IEEE, 2009.

[56] I. Haritaoglu, D. Harwood, and L. S. Davis. W 4 s : A real-time system for detecting and tracking people in 2 1/2d. In *Computer Vision—ECCV'98*, pages 877–892. Springer, 1998.

[57] D. J Heeger. Optical flow using spatiotemporal filters. *International Journal of Computer Vision*, 1(4) :279–302, 1988.

[58] A. Hervieu, P. Bouthemy, and J.-P. Le Cadre. A statistical video content recognition method using invariant features on object trajectories. *Circuits and Systems for Video Technology, IEEE Transactions on*, 18(11) :1533–1543, 2008.

[59] A. Hervieu, P. Bouthemy, and JP. Le Cadre. Reconnaissance d'événements dans des vidéos par l'analyse de trajectoires à l'aide de modèles de markov. In *Actes du Colloque GRETSI*, pages 1–4, Troyes, France, September 2007.

[60] Berthold K. P. Horn and Brian G. Schunck. Determining optical flow. *Artificial Intelligence*, 17 :185–203, 1981.

[61] M. K. Hu. Visual pattern recognition by moment invariants. *IRE Transactions on Information Theory*, pages 179–187, 1962.

[62] Y. Hu, L. Cao, F. Lv, S. Yan, Y. Gong, and T.S. Huang. Action detection in complex scenes with spatial and temporal ambiguities. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 128–135, 2009.

[63] Liu J., Saad A., and M. Shah. Recognizing human actions using multiple features. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8, 2008.

[64] A. Jain and D. Zongker. Feature selection : evaluation, application, and small sample performance. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(2) :153 –158, feb 1997.

[65] D.-S. Jang, S.-W. Jang, and H. Choi. Tracking a partially occluded target with a cluster of kalman filters. *International Journal of Intelligent Systems*, 17(6) :595–607, 2002.

[66] F. Jean, R. Bergevin, and A.B. Albu. Body tracking in human walk from monocular video sequences. In *Computer and Robot Vision, 2005. Proceedings. The 2nd Canadian Conference on*, pages 144–151, 2005.

[67] B. Jiang, M. F Valstar, and M. Pantic. Facial action detection using block-based pyramid appearance descriptors. In *Privacy, Security, Risk and Trust (PASSAT), 2012 International Conference on and 2012 International Confernece on Social Computing (SocialCom)*, pages 429–434. IEEE, 2012.

[68] Y.-G. Jiang, Q. Dai, X. Xue, W. Liu, and C.-W. Ngo. Trajectory-based modeling of human actions with motion reference points. In Andrew Fitzgibbon, Svetlana Lazebnik, Pietro Perona, Yoichi Sato, and Cordelia Schmid, editors, *Computer Vision – ECCV 2012*, volume 7576 of *Lecture Notes in Computer Science*, pages 425–438. Springer Berlin Heidelberg, 2012.

[69] L. Jiao, G. Wu, Y. Wu, E. Y. Chang, and Y.-F. Wang. The anatomy of a multi-camera video surveillance system. *ACM Multimedia System Journal Special Issue*, 10, 2004.

[70] S. J Julier and J. K Uhlmann. New extension of the kalman filter to nonlinear systems. In *AeroSense'97*, pages 182–193. International Society for Optics and Photonics, 1997.

[71] R.E Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82 :35–45, 1960.

[72] T. Kean. Cryptographic rights management of fpga intellectual property cores. In *Proceedings of the 2002 ACM/SIGDA tenth international symposium on Field-programmable gate arrays*, FPGA '02, pages 113–118, New York, NY, USA, 2002. ACM.

[73] M.J. Khan and H.A. Habib. Video analytic for fall detection from shape features and motion gradients. In *Lecture Notes in Engineering and Computer Science*, volume 2179, pages 1311–1316, 2009.

[74] K. Kim, D.I Lee, and Irfan E. Gaussian process regression flow for analysis of motion trajectories. In *Proceedings of the 2011 International Conference on Computer Vision*, ICCV '11, pages 1164–1171, Washington, DC, USA, 2011. IEEE Computer Society.

[75] J. Kittler. Feature selection and extraction. *Handbook of pattern recognition and image processing*, pages 59–83, 1986.

[76] R. Kleihorst, M. Reuvers, B. Krose, and H. Broers. A smart camera for face recognition. In *Image Processing, 2004. ICIP '04. 2004 International Conference on*, volume 5, pages 2849–2852 Vol. 5, 2004.

[77] R.P. Kleihorst, A.A. Abbo, A. Van Der Avoird, M. J R Op De Beeck, L. Sevat, P. Wielage, R. Van Veen, and H. van Herten. Xetal : a low-power high-performance smart camera processor. In *Circuits and Systems, 2001. ISCAS 2001. The 2001 IEEE International Symposium on*, volume 5, pages 215–218 vol. 5, 2001.

[78] H. Lee, L. Tessens, M. Morbee, H. Aghajan, and W. Philips. Sub-optimal camera selection in practical vision networks through shape approximation. In *Proceedings of the 10th International Conference on Advanced Concepts for Intelligent Vision Systems*, ACIVS '08, pages 266–277, Berlin, Heidelberg, 2008. Springer-Verlag.

[79] Al. Leone, G. Diraco, and P. Siciliano. Detecting falls with 3d range camera in ambient assisted living applications : A preliminary study. *Medical engineering & physics*, 33(6) :770–781, 2011.

[80] P.H.-W. Leong. Recent trends in fpga architectures and applications. In *Electronic Design, Test and Applications, 2008. DELTA 2008. 4th IEEE International Symposium on*, pages 137–141, 2008.

[81] L. Li, W. Huang, Gu I., and Q. Tian. Foreground object detection from videos containing complex background. In *In MULTIMEDIA 03 : Proceedings of the eleventh ACM international conference on Multimedia*, pages 2–10. ACM Press, 2003.

[82] X. Li, T. Zhang, X. Shen, and J. Sun. Object tracking using an adaptive kalman filter combined with mean shift. *Optical Engineering*, 49(2) :020503–020503–3, 2010.

[83] Y.T. Liao, C.L. Huang, and S.C. Hsu. Slip and fall event detection using bayesian belief network. *Pattern recognition*, 45 :24 – 32, 2012.

[84] T. Lindeberg. Feature detection with automatic scale selection. *International Journal of Computer Vision*, 30 :79–116, 1998.

[85] U. Lindemann, A. Hock, M. Stuber, W. Keck, and C. Becker. Evaluation of a fall detector based on accelerometers : A pilot study. *Med. Biol. Engineering and Computing*, 43(5) :548–551, 2005.

[86] C.L. Liu, C.H. Lee, and Lin P.M. A fall detection system using k-nearest neighbor classifier. *Expert systems with applications*, 37 :7174 – 7181, 2010.

[87] H. Liu, T.-H. Hong, M. Herman, T. Camus, and R. Chellappa. Accuracy vs efficiency trade-offs in optical flow algorithms. *Computer Vision and Image Understanding*, 72(3) :271 – 286, 1998.

[88] C. J. Lord and D. P. Colvin. Falls in the elderly : Detection and assessment. *Engineering in Medicine and Biology Society, Proceedings of the Annual International Conference of the IEEE*, 13 :1938 – 1939, 1991.

[89] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2) :91–110, November 2004.

[90] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th international joint conference on Artificial intelligence - Volume 2*, IJCAI'81, pages 674–679, San Francisco, CA, USA, 1981. Morgan Kaufmann Publishers Inc.

[91] L. Malagón-Borja and O. Fuentes. Object detection using image reconstruction with {PCA}. *Image and Vision Computing*, 27(1–2) :2 – 9, 2009. Canadian Robotic Vision 2005 and 2006.

[92] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *Proceedings of the British Machine Vision Conference*, pages 36.1–36.10. BMVA Press, 2002. doi :10.5244/C.16.36.

[93] J. Mennesson, C. Saint-Jean, and L. Mascarilla. New geometric fourier descriptors for color image recognition. In *Image Processing (ICIP), 2010 17th IEEE International Conference on*, pages 2685–2688, 2010.

[94] Y. Meyer. Orthonormal wavelets. In Jean-Michel Combes, Alexander Grossmann, and Philippe Tchamitchian, editors, *Wavelets*, Inverse Problems and Theoretical Imaging, pages 21–37. Springer Berlin Heidelberg, 1989.

[95] Y. Miao, A. Rhuma, S.M. Naqvi, and J. Chambers. Fall detection for the elderly in a smart room by using an enhanced one class support vector machine. In *Digital Signal Processing (DSP), 2011 17th International Conference on*, pages 1–6, 2011.

[96] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors, 2005.

[97] J. Miteran, S. Bouillant, M. Paindavoine, F. Meriaudeau, and J. Dubois. Real-time flaw detection on a complex object : comparison of results using classification with a support vector machine, boosting, and hyperrectangle-based method. *Journal of Electronic Imaging*, 15(1) :013018–013018–9, 2006.

[98] J. Miteran, E-B. Matas, J. an Bourennane, M. Paindavoine, and J. Dubois. Automatic hardware implementation tool for a descrete adaboost-based decision algorithm. *EURASIP Journal on Applied Signal Processing*, 7 :1035–1046, 2005.

[99] Andreas Monitzer. Using video surveillance to detect dangerous situations in underground stations by computer vision, 2006.

[100] R. Mosqueron, J. Dubois, M. Mattavelli, and D. Mauvilet. Smart camera based on embedded hw/sw coprocessor. *EURASIP Journal on Embedded Systems*, 2008 :3, 2008.

[101] R. Mosqueron, J. Dubois, and M. Paindavoine. High-speed smart camera with high resolution. *EURASIP J. Embedded Syst.*, 2007(1) :23–23, January 2007.

[102] R. Mukundan and KR Ramakrishnan. *Moment functions in image analysis : theory and applications*, volume 100. World Scientific, 1998.

[103] H. Nait-Charif and S.J. McKenna. Activity summarisation and fall detection in a supportive home environment. In *Proceedings of the Pattern Recognition, 17th International Conference on (ICPR'04)*, ICPR '04, pages 323–326, Washington, DC, USA, 2004. IEEE Computer Society.

[104] J. C. Niebles, H. Wang, and L. Fei-fei. Unsupervised learning of human action categories using spatial-temporal words. In *In Proc. BMVC*, 2008.

[105] N. Noury, P. Rumeau, A.K. Bourke, G. ÓLaighin, and J.E. Lundy. A proposal for the classification and evaluation of fall detectors. *{IRBM}*, 29(6) :340 – 349, 2008.

[106] M.N. Nyan, Francis E.H. Tay, and E. Murugasu. A wearable system for pre-impact fall detection. *Journal of Biomechanics*, 41(16) :3475 – 3481, 2008.

[107] A. Oikonomopoulos, I. Patras, M. Pantic, and N. Paragios. Trajectory-based representation of human actions. In *Proceedings of the ICMI 2006 and IJCAI 2007 international conference on Artifical intelligence for human computing*, ICMI'06/IJCAI'07, pages 133–154, Berlin, Heidelberg, 2007. Springer-Verlag.

[108] D. N. Olivieri, I. Gómez Conde, and X. A. Vila Sobrino. Eigenspace-based fall detection and activity recognition from motion templates and machine learning. *Expert Systems with Applications*, 39(5) :5935 – 5945, 2012.

[109] S. Parfait, P.M. Walker, G. Crehange, X. Tizon, and J. Miteran. Classification of prostate magnetic resonance spectra using support vector machine. *Biomedical Signal Processing and Control*, 2011(0) :1–8, 2011.

[110] R. Patnaik and D. Casasent. Fast fft-based distortion-invariant kernel filters for general object recognition. In *IS&T/SPIE Electronic Imaging*, pages 725202–725202. International Society for Optics and Photonics, 2009.

[111] K. Polat and S. Güneş. Artificial immune recognition system with fuzzy resource allocation mechanism classifier, principal component analysis and fft method based new hybrid automated identification system for classification of eeg signals. *Expert Systems with Applications*, 34(3) :2039–2048, 2008.

[112] Fatih Porikli. Trajectory pattern detection by hmm parameter space features and eigenvector clustering. In *8th European Conference on Computer Vision, Prague, CZ*, 2004.

[113] M. Pouliot. *La détermination des coefficients des ondelettes de Daubechies*. Université Laval, 2009.

[114] H. Qian, Y. Mao, W. Xiang, and Z. Wang. Home environment fall detection system based on a cascaded multi-svm classifier. In *Control, Automation, Robotics and Vision, 2008. ICARCV 2008. 10th International Conference on*, pages 1567–1572, 2008.

[115] M.D. Rodriguez, J. Ahmed, and M. Shah. Action mach a spatio-temporal maximum average correlation height filter for action recognition. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8, 2008.

[116] C. Rougier. Vidéosurveillance intelligente pour la détection de chutes chez les personnes âgées. *These de doctorat*, 2010.

[117] C. Rougier and J. Meunier. Fall detection using 3d head trajectory extracted from a single camera video sequence. In *First International Workshop on Video Processing for Security (VP4S-06), June 7-9, Quebec City, Canada (online at www.computervision.com/4security)*, 2006.

[118] C. Rougier, J. Meunier, A. St-Arnaud, and J. Rousseau. Robust video surveillance for fall detection based on human shape deformation. *Circuits and Systems for Video Technology, IEEE Transactions on*, 21 :611 – 622, 2011.

[119] C. Rougier, J. Meunier, A. St-Arnaud, and J. Rousseau. 3d head tracking for fall detection using a single calibrated camera. *Image and Vision Computing*, 2012.

[120] K. Schindler and L. Van Gool. Action snippets : How many frames does human action recognition require ? In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8, 2008.

[121] C. Schuldt, I. Laptev, and B. Caputo. Recognizing human actions : a local svm approach. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 3, pages 32–36 Vol.3, 2004.

[122] L. Senhadji, G. Carrault, JJ. Bellanger, and G. Passariello. Comparing wavelet transforms for recognizing cardiac patterns. *Engineering in Medicine and Biology Magazine, IEEE*, 14(2) :167–173, 1995.

[123] J.-l. Shen, J. Hung, and L. Lee. Robust entropy-based endpoint detection for speech recognition in noisy environments. In *ICSLP*, volume 98, pages 232–235, 1998.

[124] R. Sicre and H. Nicolas. Improved gaussian mixture model for the task of object tracking. In *Proceedings of the 14th international conference on Computer analysis of images and patterns - Volume Part II*, CAIP'11, pages 389–396, Berlin, Heidelberg, 2011. Springer-Verlag.

[125] N. T. Siebel and S.J. Maybank. The application of colour filtering to real-time person tracking. In *in Proceedings of the 2nd European Work on Advanced Video-Based Surveillance Systems (AVBS'2001*, pages 227–234, 2002.

[126] F. Smach, C. Lemaître, J.-P. Gauthier, J. Miteran, and M. Atri. Generalized fourier descriptors with applications to objects recognition in svm context. *Journal of Mathematical Imaging and Vision*, 30(1) :43–71, 2008.

[127] F. Smach, J. Miteran, M. Atri, J. Dubois, M. Abid, and JP. Gauthier. An fpga-based accelerator for fourier descriptors computing for color object recognition using svm. *Journal of Real-Time Image Processing*, 2(4) :249–258, 2007.

[128] D. Socek, D. Culibrk, O. Marques, H. Kalva, and B. Furht. A hybrid color-based foreground object detection method for automated marine surveillance. In *Proceedings of the 7th international conference on Advanced Concepts for Intelligent Vision Systems*, ACIVS'05, pages 340–347, Berlin, Heidelberg, 2005. Springer-Verlag.

[129] K. Soomro, A. R. Zamir, and M. Shah. Ucf101 : A dataset of 101 human actions classes from videos in the wild. *CoRR*, pages –1–1, 2012.

[130] A. Spinei, D. Pellerin, and J. Hérault. Spatiotemporal energy-based method for velocity estimation. *Signal Processing*, 65(3) :347 – 362, 1998.

[131] M. R. Teague. Image analysis via the general theory of moments∗. *J. Opt. Soc. Am.*, 70(8) :920–930, Aug 1980.

[132] N. Thome, S. Miguet, and S. Ambellouis. A real-time, multiview fall detection system : A lhmm-based approach. *IEEE transactions on circuits and systems for video technology*, 18 :1522–1532, 2008.

[133] K. Tokuda, H. Zen, and T. Kitamura. Trajectory modeling based on hmms with the explicit relationship between static and dynamic features. In *INTERSPEECH*, 2003.

[134] B.U. Toreyin, Y. Dedeoglu, and A.E. Cetin. Hmm based falling person detection using both audio and video. *Signal Processing and Communications Applications, 2006 IEEE 14th*, pages 1–4, 2006.

[135] T. Tuytelaars and L. Gool. Content-based image retrieval based on local affinely invariant regions. In DionysiusP. Huijsmans and ArnoldW.M. Smeulders, editors, *Visual Information and Information Systems*, volume 1614 of *Lecture Notes in Computer Science*, pages 493–500. Springer Berlin Heidelberg, 1999.

[136] V. Vapnik, editor. *The Nature of Statistical Learning Theory*. Springer-Verlag, springer-verlag edition, 1995.

[137] P. Viola, M. J. Jones, and D. Snow. Detecting pedestrians using patterns of motion and appearance. *Int. J. Comput. Vision*, 63(2) :153–161, July 2005.

[138] P. Viola and M.J. Jones. Robust real time face detection. *International Journal of Computer Vision*, 57(2) :137–154, 2004.

[139] V. Vishwakarma, C. Mandal, and S. Sural. Automatic detection of human fall in video. In *Proceedings of the 2nd international conference on Pattern recognition and machine intelligence*, PReMI'07, pages 616–623, Berlin, Heidelberg, 2007. Springer-Verlag.

[140] M. Vondrak, L. Sigal, and O.C. Jenkins. Physical simulation for probabilistic motion tracking. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8, 2008.

[141] H. Wang, M. M. Ullah, A. Klaser, I. Laptev, C. Schmid, et al. Evaluation of local spatio-temporal features for action recognition. In *BMVC 2009-British Machine Vision Conference*, 2009.

[142] J. Wang, A. Barreto, L. Wang, Y. Chen, N. Rishe, J. Andrian, and M. Adjouadi. Multilinear principal component analysis for face recognition with fewer features. *Neurocomputing*, 73(10–12) :1550 – 1555, 2010. Subspace Learning / Selected papers from the European Symposium on Time Series Prediction.

[143] D. Weinland and E. Boyer. Action recognition using exemplar-based embedding. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–7, 2008.

[144] D. Weinland, E. Boyer, and R. Ronfard. Action recognition from arbitrary views using 3d exemplars. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–7, 2007.

[145] D. Weinland, R. Ronfard, and E. Boyer. Free viewpoint action recognition using motion history volumes. *Comput. Vis. Image Underst.*, 104(2) :249–257, November 2006.

[146] D. Weinland, R. Ronfard, and E. Boyer. A survey of vision-based methods for action representation, segmentation and recognition. *Computer Vision and Image Understanding*, 115(2) :224 – 241, 2011.

[147] G. Willems, T. Tuytelaars, and L. Gool. An efficient dense and scale-invariant spatio-temporal interest point detector. In *Proceedings of the 10th European Conference on Computer Vision : Part II*, ECCV '08, pages 650–663, Berlin, Heidelberg, 2008. Springer-Verlag.

[148] J. Willems, G. Debard, B. Vanrumste, and T. Goedeme. A video-based algorithm for elderly fall detection. In *World Congress on Medical Physics and Biomedical Engineering*, volume 25/5, pages 312–315, 2009.

[149] G. Williams, K. Doughty, K. Cameron, and D.A. Bradley. A smart fall and activity monitor for telecare applications. In *Engineering in Medicine and Biology Society, Proceedings of the Annual International Conference of the IEEE*, volume 3, pages 1151 – 1154, 1998.

[150] J. Woodring, S. Mniszewski, C. Brislawn, D. DeMarle, and J. Ahrens. Revisiting wavelet compression for large-scale climate data using jpeg 2000 and ensuring data precision. In *Large Data Analysis and Visualization (LDAV), 2011 IEEE Symposium on*, pages 31–38, 2011.

[151] C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland. Pfinder : real-time tracking of the human body. In *Automatic Face and Gesture Recognition, 1996., Proceedings of the Second International Conference on*, pages 51–56, 1996.

[152] Y.P. Yong and Abu Bakar S.A. Integration of projection histograms and linear prediction for object tracking. *Jurnal Teknologi*, 39(D) :57–68, 2003.

[153] Zhang Z., Kaiqi H., and Tieniu T. Comparison of similarity measures for trajectory clustering in outdoor surveillance scenes. In *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, volume 3, pages 1135–1138, 2006.

[154] H. Zen, K. Tokuda, and T. Kitamura. An introduction of trajectory model into hmm-based speech synthesis. In *Fifth ISCA Workshop on Speech Synthesis*, 2004.

TABLE DES FIGURES

LISTE DES TABLEAUX

A

DIFFÉRENCE TEMPORELLE SIMPLE

Cette étape consiste à faire des différences entre deux images I_t et I_{t-n} de la séquence vidéo pour extraire les changements temporels. Le paramètre n dépend notamment de la nature du mouvement. S'il est rapide, la différence entre deux images consécutives ($n = 1$) devient significative et permet par conséquent de segmenter le mouvement. Dans le cas contraire, il faut tenir compte de la lenteur du mouvement et trouver la valeur n pour laquelle la différence temporelle ait un sens. Pour notre étude, ce paramètre est fixé à $n = 8$ puisqu'une activité humaine est relativement lente. Un pixel p est en mouvement lorsque cette différence temporelle présentée dans l'équation A.1 dépasse un seuil τ et il désigne un pixel du fond sinon. Le résultat (illustré dans la Figure A.1(c)) est une image binaire dite masque de mouvement que nous notons F_t.

$$F_t(p) = \begin{cases} 1 & si \quad |I_t(p) - I_{t-n}(p)| > \tau \\ 0 & sinon. \end{cases} \tag{A.1}$$

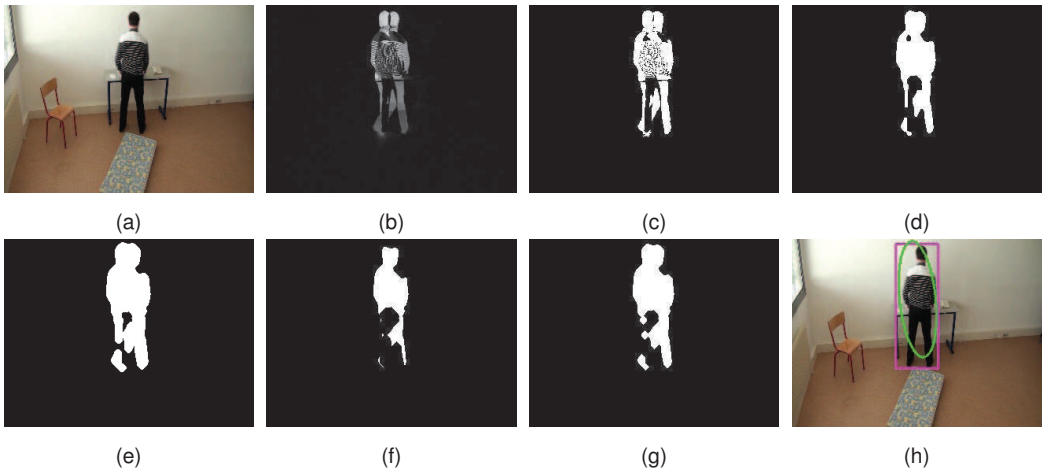


FIGURE A.1 – Résultats de la segmentation de mouvement sur un exemple. (a) image initiale, (b) différence temporelle pour $n = 8$, (c) après seuillage $\tau = 20$, (d) après filtrage Median, (e) après dilatation, (f) après érosion, (g) après dilatation, (h) détection finale (après fusion des régions) sur l'image d'origine.

Résumé :

Nous proposons une méthode supervisée de détection de chutes de personnes en temps réel, robuste aux changements de point de vue et d'environnement. La première partie consiste à rendre disponible en ligne une base de vidéos $DSFD$ enregistrées dans quatre lieux différents et qui comporte un grand nombre d'annotations manuelles propices aux comparaisons de méthodes. Nous avons aussi défini une métrique d'évaluation qui permet d'évaluer la méthode en s'adaptant à la nature du flux vidéo et la durée d'une chute, et en tenant compte des contraintes temps réel. Dans un second temps, nous avons procédé à la construction et l'évaluation des descripteurs spatio-temporels $STHF$, calculés à partir des attributs géométriques de la forme en mouvement dans la scène ainsi que leurs transformations, pour définir le descripteur optimisé de chute après une méthode de sélection d'attributs. La robustesse aux changements d'environnement a été évaluée en utilisant les SVM et le Boosting. On parvient à améliorer les performances par la mise à jour de l'apprentissage par l'intégration des vidéos sans chutes enregistrées dans l'environnement définitif. Enfin, nous avons réalisé, une implantation de ce détecteur sur un système embarqué assimilable à une caméra intelligente basée sur un composant SoC de type Zynq. Une démarche de type Adéquation Algorithme Architecture a permis d'obtenir un bon compromis performance de classification/temps de traitement.
Mots-clés : Détection de chute temps réel, descripteurs spatio-temporels, sélection d'attributs, SVM, Boosting, base de vidéos de chute, robustesse aux changements d'environnement, métrique d'évaluation caméra intelligente, System on Chip (SoC).

Abstract:

We propose a supervised approach to detect falls in home environment adapted to location and point of view changes. First, we maid publicly available a realistic dataset, acquired in four different locations, containing a large number of manual annotation suitable for methods comparison. We also defined a new metric, adapted to real-time tasks, allowing to evaluate fall detection performance in a continuous video stream. Then, we build the initial spatio-temporal descriptor named $STHF$ using several combinations of transformations of geometrical features and an automatically optimised set of spatio-temporal descriptors thanks to an automatic feature selection step. We propose a realistic and pragmatic protocol which enables performance to be improved by updating the training in the current location with normal activities records. Finally, we implemented the fall detection in Zynq-based hardware platform similar to smart camera. An Algorithm-Architecture Adequacy step allows a good trade-off between performance of classification and processing time.

■ École doctorale SPIM - Université de Bourgogne/UFR ST BP 47870 F - 21078 Dijon cedex
■ tél. +33 (0)3 80 39 59 10 ■ ed-spim@univ-fcomte.fr ■ www.ed-spim.univ-fcomte.fr