Estimation de la posture humaine par capteur RGB-D

Lucas Marti

Université Fédérale

Toulouse Midi-Pyrénées

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par :

Université Toulouse 3 Paul Sabatier (UT3 Paul Sabatier)

Présentée et soutenue par :

Lucas MARTI

le lundi 12 octobre 2015

Titre :

Estimation de la posture humaine par capteur RGB-D

École doctorale et discipline ou spécialité :

EDSYS : Informatique 4200018

Unité de recherche :

LAAS-CNRS

Directeur/trice(s) de Thèse :

LERASLE Frédéric

MONIN André

Jury :

CAPLIER Alice, Présidente

NOYER Jean-Charles, Rapporteur

BOVERIE Serge, Rapporteur

ORIAT Luc, Examinateur

Abstract

In a world where the population is increasingly aging, elderly people falling is a public health issue. The use of technology is a major development axis for fall management. We want to design a complete system for detection and estimation of falls. The target market for this system is medicalized retirement homes and individual homes. We want to improve the medical care of people after a fall in order to reduce its consequences. A solution based on an ambient sensor seems to be the most adapted solution. Recent developments in RGB-D (Color+Depth) sensing are a great asset thanks to their relatively low cost, wide availability and good performances.

The first part of the thesis deals with the problem of segmenting people from the surrounding scene in our images. We present an algorithm that determines the silhouettes of each person in the room in which the sensor is installed, thanks to simultaneous use of color and depth. The algorithm is robust to the change of configuration of the room and especially to moving furniture. We use special consideration of depth to reach a performance level sufficient for an industrial application.

The second part of the thesis deals with the estimation of the human posture. Once the silhouettes have been segmented with the algorithm described in the first part, we want to get an estimation of every articulation of the person. We build on existing algorithms that use machine learning and in particular Random Forest by investigating new ideas to improve performances. We found optimal values for some parameters that were not previously investigated. We present a new feature to be computed on depth images. Finally we evaluate the impact of balancing the training database in our context. The algorithm provides a set of predictions for the position of every articulation.

In the third part, we focus on spatio-temporal filtering of the postures. We examine different approaches and in particular we deal with the issue of left/right ambiguity that arises in the algorithm presented in the previous part. The approaches are based on Bayesian filtering.

RESUME

Dans le contexte du vieillissement de la population, le problème de la chute est un problème de santé publique. L'utilisation de la technologie est un axe important pour la prise en compte des enjeux liés à la chute. Nous voulons développer un système complet de détection et d'estimation de la chute à destination des maison de retraites et des particuliers dans un but d'amélioration de la prise en charge médicale et de limitation des conséquences de la chute. La solution de détection à l'aide d'un capteur ambiant déporté nous a semblé être la solution la plus adaptée. Les capteurs RGB-D (couleurs + profondeur) développés récemment sont un atout pour cela car ils sont peu chers, très commun et performants.

La première partie de la thèse traite du problème de segmentation de la silhouette de la personne dans la scène observée. Nous présentons un algorithme qui fourni, grâce à une utilisation conjointe des images de couleurs et de profondeur, les silhouettes des différents personnes dans la pièce dans laquelle est situé le capteur. L'algorithme est robuste au changement de configuration de la pièce et notamment au déplacement d'objets. L'utilisation de caractéristiques propres à la profondeur nous permet d'atteindre des niveaux de performances suffisant pour une utilisation industrielle.

Dans la seconde partie de la thèse nous abordons le problème de l'estimation de la posture humaine. Une fois les silhouettes segmentées grâce à l'algorithme de la première partie, nous voulons avoir les positions des articulations de la personne. Nous incrémentons sur les algorithmes déjà existants utilisant l'apprentissage et notamment les algorithmes des randoms forests en investiguant de nouvelles idées pour augmenter les performances. Nous déterminons les valeurs optimales de certains paramètres qui ne sont pas explorés dans les travaux précedents. Nous présentons un nouveau type de caractéristique à calculer sur les images de profondeur. Nous examinons enfin l'impact de l'équilibrage de la collection d'apprentissage dans notre contexte. L'algorithme fourni alors un ensemble de prédiction pour chaque articulation.

Dans la troisième partie, nous traitons le problème du filtrage spatio-temporel des poses. Nous présentons différentes approches et nous intéressons à l'ambiguïté droite/gauche des membres et cherchons à traiter ce problème. Les approches utilisées se fonde sur le filtrage bayesien.

Table des matières

Chapitre 1

Introduction Générale

Le présent chapitre présente le contexte qui a initié cette thèse CIFRE avec la société ORME : la détection de chute de personnes, les enjeux et applications associés. Nous présentons dans un premier temps l'état de l'art général des techniques existantes de détection de chutes disponibles dans le commerce. Dans une seconde section, nous présentons les enjeux de ces travaux de recherche pour et au-delà de la détection de chute. La troisième section présente le capteur privilégié dans notre application et résume nos travaux puis présente le plan du manuscrit.

1.1 La détection de chute : un enjeu sociétal

Dans cette partie, nous décrivons la problématique de la détection de chute. Avant de détecter une chute, il convient d'en donner une définition précise. La chute correspond à l'action de tomber au sol indépendamment de sa volonté selon Dargent-Molina et al.[20]. A la suite d'une chute, il est d'usage de considérer que la personne aura du mal à se relever et c'est ce qui contribue à en faire potentiellement un évènement grave. Différents types de chutes sont répertoriés : les chutes lourdes correspondent à une perte brutale de verticalité. A l'inverse, lorsqu'une chute est molle, on considère que la personne a tenté de se retenir à un meuble par exemple. La troisième catégorie est la chute syncopale qui est, elle, liée à une perte de connaissance.

Nous évoquerons d'abord le contexte médical et social du problème. Puis nous parlerons des systèmes existants de détection en évoquant leurs forces et leurs faiblesses.

1.1.1 Contexte médical et social

Le vieillissement de la population dans les pays développés a fait émerger de nouveaux besoins. Selon l'INSEE, la population de plus de 60 ans représente un sixième de la population française. Les tendances indiquent que d'ici à 2050, 20% de la population mondiale aura plus de 60 ans. Entre 2008 et 2060, l'INSEE prévoit un triplement de la population européenne des plus de 80 ans pour arriver à 61 millions de personnes. Ce vieillissement a pour conséquence l'augmentation du nombre de personnes en dépendance ou en fragilité et l'INSEE prévoit l'arrivée de 375 000 personnes âgées dépendantes en Etablissements d'Hébergement pour Personnes Agées Dépendantes (EHPAD) à l'horizon 2040. Il existe donc une double problématique à la fois de qualité de vie pour les personnes déjà en maison de retraite et de maintien à domicile des personnes souhaitant vivre le plus longtemps chez elles. En effet, 60% des personnes dépendantes vivent encore chez elles.

Dans ce contexte, la chute est un facteur important de passage à la fragilité ou à la dépendance et il convient alors de la prévenir et de la traiter le mieux possible. Les conséquences d'une chute pour une personne âgée peuvent être gravissimes. Celles-ci représentent deux tiers des décès accidentels chez les plus de 75 ans. Plus de deux millions de personnes de plus de 65 ans chutent chaque année. Au-delà du risque médical immédiat en cas de chute, celle-ci a très souvent un impact psychologique important sur la personne qui perd alors confiance et autonomie. Cette perte d'autonomie entraîne une diminution de la qualité et de l'espérance de vie.

Il apparaît alors important de prendre en charge ces tendances afin de limiter les impacts sociaux sur les populations concernées mais aussi de limiter les coûts économiques liés à leur nécessaire prise en charge par la société. En 2012, le secteur du service à la personne comptait 449 000 emplois avec 2 millions de personnes bénéficiant de ces services en France. L'augmentation des besoins et les ressources limitées pour leur subvenir imposent de trouver de nouvelles façons de prendre en charge la fragilité.

1.1.2 Solutions capteurs existantes pour la détection de chute

Un certain nombre d'algorithmes et de systèmes commerciaux existent permettant la détection de chute. Ceux-ci se répartissent en deux grandes catégories : les solutions portées et les solutions déportées.

Systèmes portés

Lorsque l'on pense aux systèmes portés sur la personne, on pense d'abord aux systèmes d'alerte déclenchés par l'utilisateur. Ceux-ci se présentent souvent sous la forme de médaillons portés autour du cou sur lesquels se trouve un bouton sur lequel la personne appuie après une chute. Ces systèmes ne détectent pas la chute directement mais permettent son signalement. Ils sont inutiles cependant lorsque la personne tombe inconsciente ou lorsqu'elle est trop confuse pour se servir du système, ce qui est généralement le cas pour les personnes atteintes de la maladie l'Alzheimer.

Certains systèmes portés utilisent des accéléromètres. Généralement portés à la ceinture, ces systèmes peinent à différencier les chutes des mouvements rapides vers le bas tel que le fait de s'asseoir.

Ces systèmes portatifs possèdent des avantages et des inconvénients liés à leur nature même. Un avantage majeur est qu'ils se déplacent avec la personne dont on cherche à détecter la chute et possèdent alors un rayon d'action important. Parmi les inconvénients, citons le fait que la personne doit porter ces dispositifs toujours sur elle pour qu'ils soient efficaces. En pratique, les personnes oublient de porter ce dispositif ou négligent de le mettre lorsqu'ils se lèvent la nuit par exemple. Cela est problématique car un grand nombre de chutes surviennent de nuit. Une des raisons pour lesquelles les gens négligent de porter ces dispositifs est leur aspect invasif ce qui nous amène au deuxième inconvénient principal de ces systèmes. Certaines personnes trouvent désagréable ou encombrant ce genre de système. Le but recherché étant entre autre une amélioration de la qualité de vie, cela est alors préjudiciable en plus d'être nuisible à l'efficacité de ces solutions.

Systèmes déportés

Les systèmes déportés ne souffrent pas de l'inconvénient de l'instrumentation de la personne qui apparaît avec les dispositifs portatifs. Cependant ces solutions sont limitées par le champ de vue inhérent aux capteurs considérés. Ils sont toutefois capables de détecter une chute sans intervention de la personne surveillée, que ce soit en amont ou au moment de la chute. Parmi les capteurs déportés se trouvent les capteurs vidéo tels que ceux proposés par Link Care Services. Cette solution, en particulier, pose des problèmes d'invasion de la vie privée car les images sont envoyées à un centre de contrôle. Dans la littérature, un certain nombre de travaux s'intéressent à la détection automatique de chute par vision. Nous allons présenter un rapide état de l'art sans prétention d'exhaustivité. Une approche

utilisant comme nous un capteur de profondeur est proposée par Diraco et al. [24]; la chute est détectée grâce à des seuils de hauteur. Mastorakis et al. [41] utilisent comme nous un capteur RGB-D de type Kinect [42] et la segmentation fournie avec celui-ci. Dans Rougier et al. [49], le suivi de la tête est utilisé pour détecter la chute dans une séquence d'images couleur. Rougier et al. dans [50] utilisent la déformation d'une silhouette segmentée dans l'image couleur. Un algorithme utilisant plusieurs caméras couleur et la classification de postures est utilisé par Cucchiara et al. [18]. Malheureusement, soit ces approches utilisent des caméras couleur ne fournissant pas d'images la nuit, soit elles ne se généralisent pas à l'estimation de posture.

D'autres solutions existent telles que les capteurs au sol. Le plancher de la personne est recouvert d'une surface qui détecte la chute. Cette solution reste cependant coûteuse du fait de l'installation d'une infrastructure conséquente.

1.2 Besoins de l'entreprise

La société ORME est une entreprise de traitement du signal et de l'image située à Labège dans la région toulousaine [4]. En côtoyant des médecins et des personnels soignants en contact avec des personnes en fragilité, la société ORME a perçu un réel besoin pour des systèmes permettant de les aider à détecter mais aussi prévenir les chutes. Le but premier de ce travail de recherche est, dans ce contexte, le développement d'un capteur de chute par vision à destination des EHPAD. Le système doit répondre au cahier des charges suivant :

- Fonctionnement de nuit comme de jour, ce qui nous empêche d'utiliser des caméras couleur classiques.

- Coût limité et instrumentation minimale de l'espace privatif. Cela nous oriente vers une solution avec un seul capteur léger et grand public.

- Respect de la vie privée. En effet pour des questions légales et éthiques, il est problématique de proposer une solution de vidéo-surveillance où des personnes visualisent les images. Seule une alarme est envoyée au système d'appel malade de l'établissement en cas de chute et seuls les soignants des EHPAD iront voir la personne après l'envoi de l'alarme.

- Reconstruction de la dynamique de la chute afin de donner aux médecins une première idée de diagnostic, notamment dans le cas d'une personne inconsciente. Il est demandé de préserver la vie privée de la personne tout en permettant la transmission d'informations.

Par ailleurs, il a été envisagé une action de prévention à travers la détection de comportements dangereux grâce à l'analyse de la posture humaine.

- Robustesse aux situations variées rencontrées (postures, distances, position relative du capteur par rapport au sujet, apparence et morphologie du sujet)

- Fonctionnement temps réel pendant des périodes de temps longues sans intervention humaine et sans problèmes majeurs affectant la capacité du système à réaliser sa fonction et mettant alors en danger la santé des personnes surveillées.

Au-delà de la détection de chutes, la société ORME a identifié des besoins autres auxquels ces travaux doivent répondre et en particulier le transfert de technologie en estimation de posture de manière générale. En effet, pour certains projets, actuels ou futurs et impliquant ORME, il est nécessaire de développer ce type de compétences. Citons par exemple le projet SEFA-IKKY (pour "Intégration du cocKpit et de ses sYstèmes") d'estimation de la position et de l'orientation de la tête d'un pilote d'avion. Notre objectif est alors, dans le cadre de cette problématique de détection, de présenter une solution générique permettant de répondre aux besoins émergents de la société ORME.

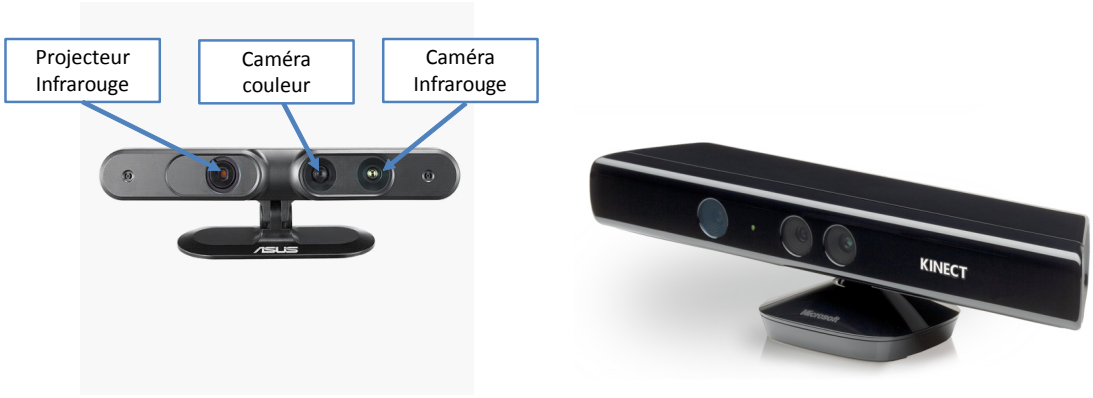
1.3 Notre solution

Comme expliqué précédemment, nous privilégions un capteur optique et les techniques de Vision par Ordinateur pour répondre au problème de la détection de chute et, au-delà, de l'estimation de postures humaines. Dans un premier temps nous présentons le capteur utilisé et justifions ce choix. Puis nous présentons un résumé de notre solution algorithmique ; chaque composante sera décrite et évaluée dans un chapitre dédié.

1.3.1 Notre solution matérielle : le capteur RGB-D

Nous privilégions un capteur RGB-D (Red Green Blue - Depth) Asus Xtion [10] pour notre solution. Celui-ci est présenté sur la figure 1-1a. Ce capteur actif intègre un système de capture de profondeur et une caméra couleur classique. Le système de calcul de la profondeur se compose d'un projecteur infrarouge projetant un motif structuré sur la scène et d'une caméra infrarouge qui infère des mesures 3D par triangulation active à l'instar des capteurs à lumière structurée. Le capteur Kinect de Microsoft [42] montré sur la figure 1-1b est un packaging différent du même capteur. Nous avons préféré un Xtion pour des raisons de prix et de taille de capteur ; le Xtion est de surcroît plus petit et plus

léger. Nous avons préféré ce type de solutions à une paire stéréoscopique pour plusieurs raisons. Tout d'abord, l'information de profondeur est donnée directement par le capteur et les pilotes associés comme OpenNI [46] sans besoin d'algorithmes supplémentaires. De plus, l'information de profondeur fonctionne aussi la nuit car il s'agit d'un capteur actif à lumière infrarouge. Enfin, les difficultés des algorithmes stéréoscopiques à déterminer la profondeur des surfaces non texturées sont bien connues. Pour les applications visées par ORME, la présence ou non de texture est inconnue a priori.



a) Capteur RGB-D Asus Xtion. b) Capteur Kinect de Microsoft.

Figure 1-1: Capteurs RGB-D.

1.3.2 Solution logicielle

Notre solution logicielle est dite "bottom-up" comme illustré sur la figure 1-2. Cette approche est inspirée de Agarwal et al. dans [6]. On peut aussi supposer qu'elle est utilisée dans OpenNI [46] et NITE [3] qui sont respectivement les pilotes et la solution de détection de posture fournie avec le Xtion.

Figure 1-2: Résumé de notre approche "Bottom-up".

Le chapitre 2 décrit notre algorithme de segmentation de personnes dans une séquence

vidéo RGB-D. En effet, un premier besoin s'est dégagé d'améliorer le processus de segmentation, les algorithmes existants et en particulier ceux fournis avec les capteurs s'adaptant mal à notre contexte. Notre contribution principale se situe ici dans l'exploitation originale de l'information de profondeur pour ce problème et celui de la soustraction de fond en général. Nous avons pu constater que notre algorithme supplante l'approche développée dans OpenNI [46] en termes de robustesse et de portée en distance.

Le chapitre 3 décrit l'algorithme d'estimation 3D de postures humaines à l'aide d'une seule image de capteur RGB-D que nous avons développé. Nous avons réalisé notre propre implémentation des algorithmes de Girshick et al. [27] et NITE [3] dans le but de les adapter à notre problématique de chute. Nous avons de plus explicité les valeurs optimales des paramètres libres de ces algorithmes, non étudiés dans la littérature bien qu'ils impactent fortement les performances de ceux-ci. Enfin, nous avons proposé quelques amendements prometteurs à l'algorithme existant.

Le chapitre 4 présente deux algorithmes de filtrage temporel multi-modal de la posture humaine adaptés aux sorties engendrées par l'algorithme décrit au chapitre précédent. Notons que la reconstruction par image à l'instar de Girshick et al. [27] ne garantit aucune cohérence temporelle car il s'agit d'une reconstruction indépendante à chaque image donnant plusieurs hypothèses la position 3D de chaque articulation. La cohérence spatio-temporelle permet de déterminer la position 3D la plus probable à chaque instant. Par ailleurs, nous abordons spécifiquement le problème de levée d'ambiguïtés droite/gauche sur la position des articulations qui survient lors de la reconstruction image par image.

Le chapitre 5 constitue la conclusion de nos travaux, évoque le prototype commercial réalisé et livre quelques perspectives.

Chapitre 2

Segmentation de silhouettes humaines par capteur RGB-D

Alors que la littérature sur la soustraction de fond sur des images couleur est riche comme le recensent Dhome et al. dans [23] , beaucoup moins de travaux de recherche ont été effectués sur la soustraction de fond et la détection de personnes à partir d'images RGB-D. Ceci est probablement dû au coût prohibitif qu'ont pu avoir dans le passé les systèmes d'imagerie permettant de produire des images de profondeur comme les caméras "Time of Flight". Avec l'arrivée du Kinect [42] et du Xtion [10], les images RGB-D se sont vulgarisées et, avec ces capteurs, sont arrivés les algorithmes de détection de silhouettes humaines utilisés dans les librairies OpenNI [46] et Kinect SDK [42] fournies par les constructeurs. Leurs limitations sont hélas nombreuses. Citons :

- Leur limitation en distance : leur fonctionnement nominal correspond à des personnes situées à moins de $6m$ du capteur, la segmentation de la silhouette n'est plus assurée au-delà.

- En fonctionnement nominal, le capteur doit être situé à une hauteur correspondant à celle d'un meuble de télévision. Leur technique traite très mal le cas où le capteur est accroché au plafond de façon plongeante et, en particulier, ils fonctionnent très mal lorsque la personne est au sol. Ils sont en effet destinés à une utilisation avec la personne debout, face au capteur (situation typique du jeu vidéo).

- Ces librairies ne sont pas ou très peu documentées et leur licence d'utilisation interdit une utilisation commerciale.

- Un objet faussement détecté comme une personne le restera et ne sera jamais intégré

au fond comme le montre la figure 2-1 où l'on voit que le tableau situé derrière la chaise est détecté comme une personne. On voit aussi que la chaise est fusionnée dans une même silhouette avec la personne située à côté. En particulier la figure 2-1b montre un objet qui n'intégrera jamais le fond car OpenNI ne possède pas de mécanisme le permettant.

- Ces algorithmes n'utilisent pas le flux RGB du capteur alors qu'il existe un grand nombre de techniques performantes dans ce domaine pour la segmentation de régions mobiles.

a) Fusion de silhouettes de la personne et de la chaise.

b) Détection d'un objet immobile comme silhouette.

Figure 2-1: Erreurs de segmentation par OpenNi.

Forts de ces constats, dans ce chapitre, nous proposons une méthode combinant l'information de couleur RGB et l'information de profondeur D pour segmenter les silhouettes des différentes personnes présentes dans la scène. L'information de profondeur est extrêmement utile car elle permet de s'affranchir des conditions d'illumination. En particulier pour les capteurs tels le Kinect [42] ou les caméras "Time of Flight", cette information est aussi exploitable indépendamment des conditions d'illumination. L'information de profondeur est très discriminante comme nous allons le montrer et possède une spécificité dont la prise en compte permet de s'affranchir de problèmes classiques de la soustraction de fond tels que les fantômes qui apparaissent lorsqu'un objet du fond se déplace. Ceci est courant pour des scènes humaines donc variables et évolutives.

2.1 Positionnement de notre approche

Nous présentons ci après un état de l'art synthétique afin de positionner notre approche et les choix sous-jacents.

Algorithmes basés sur la détection de personnes

Certains travaux comme Gulshan et al. dans [29] et Xia et al. dans [40], se focalisent sur des images provenant de capteurs Kinect [42] ou Xtion [10] et utilisent des méthodes de détection de personnes pour amorcer leur segmentation. Alahari et al.dans [7] présentent un algorithme sur des séquences stéréoscopiques de films en 3D. Malheureusement, ces travaux ne s'intéressent qu'à la segmentation de personnes debout et ne permettent pas de détecter des personnes allongées ou dans des positions assises ou penchées. De plus, ils ne permettent de segmenter des personnes qu'à une distance limitée ($< 6m$). Ils ne conviennent donc pas à notre application de détection de chutes à plus de $6m$. Jafari et al. dans [33] ou Munaro et al. dans [44] utilisent des images RGB-D pour détecter des personnes sur des robots mobiles sans les segmenter. Un certain nombre d'approches récentes s'intéressent à la segmentation conjointement à l'estimation de pose : par exemple Ladicky et al. dans [38] avec des images couleurs ou encore Kholi et al. dans [35] avec plusieurs caméras couleur. L'utilisation de caméras couleur de nuit, lorsque la scène observée est obscure, est impossible ce qui rend ces approches inopérantes dans notre cas.

Algorithmes basés sur le mouvement

Beaucoup d'algorithmes existants de soustraction de fond en RGB tels que ceux cités dans la "survey" par Dhome et al. dans [23] ou plus récemment Barnich et al. dans [11] pourraient être appliqués en l'état à des images RGB-D en considérant la composante de profondeur comme une quatrième composante que l'on traitera comme les autres composantes couleurs. Certains travaux exploitent l'ajout de la profondeur de façon spécifique pour la soustraction de fond : en plus de présenter un algorithme inspiré du Codebook Kim et al. dans [34], Fernandez-Sanchez et al. dans [25] recensent un certain nombre de méthodes. Kolmogorov et al. dans [36] et Ivanoc et al. dans [32] se concentrent sur les spécificités que possèdent les images venant d'une paire stéréoscopique. Crabb et al. dans [17] et Zhu et al. dans [62] s'intéressent principalement au remplacement du fond en utilisant la profondeur pour créer différents plans parallèles à la caméra mais ne sont pas adaptés à un contexte de vidéosurveillance.

Harville et al. dans [31] proposent l'approche la plus similaire à la nôtre. Ils utilisent un mélange de gaussiennes RGB-D mais la prise en compte de la spécificité de l'information de profondeur est naïve ce qui fait qu'ils ne peuvent se débarrasser des fantômes liés aux objets intégrés dans le fond qui se déplacent ensuite.

2.2 Notre approche

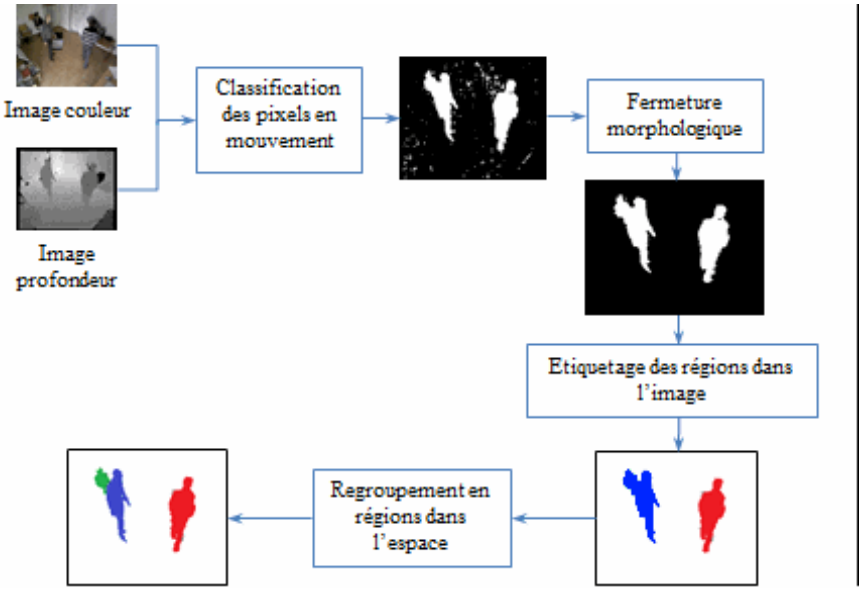


Figure 2-2: Synoptique de notre approche.

Notre approche vise à segmenter les différentes régions mobiles correspondant a priori à des personnes à partir d'une image RGB-D. La présence de la composante de profondeur nous permet de facilement distinguer les composantes en mouvement qui seraient confondues dans l'image mais séparées les unes des autres dans la réalité comme une personne en masquant partiellement une autre.

L'approche est schématisée par la figure 2-2. Tout d'abord nous allons utiliser une méthode pixellique basée sur des mélanges de gaussiennes qui vont classer chaque pixel en plan mobile ou arrière-plan sans tenir compte des relations de voisinage entre pixels. Le premier plan correspond aux pixels des objets en mouvement, l'arrière-plan aux pixels des objets statiques. Cette méthode utilise spécifiquement l'information de profondeur pour gérer les ambiguïtés propres aux algorithmes de soustraction de fond ce qui n'est pas le cas de la littérature actuelle. Après un filtrage morphologique, ces pixels sont ensuite regroupés pour former des composantes connexes dans l'image. Enfin, un algorithme d'étiquetage en 3D va séparer les composantes connexes de l'image en groupes de points dans l'espace.

Nous allons d'abord formaliser notre algorithme avant de décrire son implémentation pratique. Des évaluations seront ensuite présentées à la fois dans le contexte applicatif

et au-delà. Enfin, nous présenterons les résultats obtenus en utilisant le seul canal de profondeur à l'instar d'OpenNI.

2.3 Classification des pixels

Notre approche exploite ici des mélanges de gaussiennes à l'instar de Stauffer et al. [65]. Les évaluations montrées dans Benezeth et al. [12] démontrent la pertinence de cette approche dans un contexte RGB à la fois du point de vue des performances et du point de vue du temps de calcul. De plus, le formalisme présenté dans Stauffer et al. [65] se prête particulièrement bien aux améliorations proposées. Cette modélisation permet de prendre en compte à la fois le bruit inhérent au capteur et le fait que différents objets constituent la scène (ne serait-ce qu'une personne et le fond devant lequel elle est située). Ces objets seront représentés localement pour chaque pixel par une distribution gaussienne. **Notre algorithme s'inspire de ces approches en fixant l'appartenance de chaque composante au fond ou au premier plan. L'idée clé de notre approche est d'utiliser la profondeur comme une dimension spéciale pour laquelle la relation d'ordre existante possède des propriétés directement liées au fait qu'un pixel appartient au premier plan ou au fond.** Nous formaliserons d'abord cet algorithme avant de parler des problèmes spécifiques à l'implémentation de cet algorithme.

2.3.1 Modèle associé à chaque pixel

Chaque pixel x_i de l'image à segmenter est représenté par un vecteur 4D $x_i = (r_i, g_i, b_i, d_i)$ qui représente dans l'ordre le canal rouge, vert, bleu et de profondeur du pixel i.

Toutes les variables utilisées dans cette section décrivent l'état d'un seul pixel x_i alors noté ici simplement x pour simplifier les notations. Un pixel peut appartenir soit au fond soit au premier plan. On note $P(x|B)$ la distribution de probabilité du pixel i sachant qu'il appartient à l'arrière-plan et $P(x|F)$ la distribution de probabilité du pixel i sachant qu'il appartient au premier plan.

Pour chaque pixel de l'image, chaque plan est représenté par une distribution normale. L'espérance de cette distribution représente la position réelle du plan dans l'espace 4D (à estimer), la variance correspond au bruit inhérent à la capture de couleur et à l'estimation de profondeur par le capteur ainsi qu'à l'évolution possible de la position de ce plan. Il n'existe qu'une seule composante de premier plan pour chaque pixel alors qu'il existe un

nombre arbitraire N_b de plans du fond. La distribution de probabilité est représentée par un mélange de gaussiennes décrit comme suit :

L'appartenance d'un pixel à une composante $k \in 1..N_b$ particulière du fond est décrite par la variable qualitative B_k. L'appartenance d'un pixel à une composante quelconque du fond est décrite par la variable qualitative B. Les matrices de variance-covariance des composantes sont notées $\Sigma_\bullet$ et considérées comme diagonales.

Pour le premier plan :

$$\Sigma_f = \begin{bmatrix} \sigma_{f,r} & 0 & 0 & 0 \\ 0 & \sigma_{f,g} & 0 & 0 \\ 0 & 0 & \sigma_{f,b} & 0 \\ 0 & 0 & 0 & \sigma_{f,d} \end{bmatrix} \tag{2.1}$$

Pour les composantes de fond :

$$\Sigma_k = \begin{bmatrix} \sigma_{k,r} & 0 & 0 & 0 \\ 0 & \sigma_{k,g} & 0 & 0 \\ 0 & 0 & \sigma_{k,b} & 0 \\ 0 & 0 & 0 & \sigma_{k,d} \end{bmatrix} \tag{2.2}$$

-L'espérance de la composante de premier plan est notée $f = (f^r, f^g, f^b, f^d)$.

-L'espérance d'une composante $k \in 1..N_b$ du fond est notée $b_k = (b_k^r, b_k^g, b_k^b, b_k^d)$.

Avec ces définitions nous pouvons écrire les probabilités ci-après de la valeur des pixels selon leur appartenance aux différents plans :

D'après notre hypothèse gaussienne, nous définissons :

$$\begin{aligned} P(x, x \quad \in \quad B_k) &= P\left(x | x \in B_k\right) P\left(x \in B_k\right) \\ &= \quad \rho_{b,k} \Gamma(x, b_k, \Sigma_{b,k}) \end{aligned}$$

où $\Gamma\left(x, \bar{x}, Q\right)$ désigne la densité de probabilité gaussienne de moyenne $\bar{x}$ et de variance Q et où $\rho_{b,k} = P\left(x \in B_k\right)$. Les densités de probabilité des composantes de fond et de

premier plan sont alors définies comme suit :

$$P(x, x \in B) = \sum_{k=1}^{N_b} P(x, x \in B_k) \tag{2.3a}$$

$$= \sum_{k=1}^{N_b} \rho_{b,k} \Gamma(x, b_k, \Sigma_{b,k}) \tag{2.3b}$$

$$P(x, x \in F) = P(x \in F)\, P(x | x \in F) \tag{2.3c}$$

$$= \rho_f \Gamma(x, f, \Sigma_f) \tag{2.3d}$$

La distribution de probabilité du pixel est alors :

$$\begin{aligned} P(x) &= P(x, x \in B) + P(x, x \in F) \\ &= \rho_f \Gamma(x, f, \Sigma_f) + \sum_{k=1}^{N_b} \rho_k \Gamma_{b,k}(x, b_k, \Sigma_{b,k}) \end{aligned} \tag{2.4}$$

2.3.2 Relation d'ordre via le canal profondeur

Notre approche considère qu'il existe plusieurs niveaux de fond, ce qui en fait sa spécificité par rapport aux travaux déjà existants utilisant la profondeur. En effet, le canal de profondeur permet d'ajouter des considérations spatiales et nous permet d'ordonner les plans par leur profondeur. L'information de profondeur permet de donner un sens physique aux termes "premier plan" et "arrière-plan". Le premier plan est toujours situé devant l'arrière-plan ce qui se traduit par le fait que les pixels appartenant au premier plan se trouvent plus près du capteur que s'ils appartenaient à l'arrière-plan. Ainsi, les pixels d'un premier plan auront, par définition, toujours une profondeur plus faible que les pixels d'arrière-plan. Nous pouvons ainsi savoir pour une composante si elle modélise le premier plan ou l'arrière-plan. L'appartenance des différentes composantes gaussiennes au fond ou au premier-plan est donc fixée (et ne dépendra alors plus des poids $\rho_{b,k}$ et ρ_f des composantes) et permet ainsi de supprimer la majorité des fausses détections.

La relation d'ordre sur la profondeur et les nouvelles considérations que nous en tirons nous permettent d'écrire les relations suivantes entre les espérances des composantes gaussiennes :

$$f^d < b_k^d, \ \forall k = 1..N_b \tag{2.5}$$

$$b_k^d < b_1^d, \ \forall k = 2..N_b \tag{2.6}$$

La figure 2-3 montre l'ordre des plans pour deux plans. Le plan de fond 1 est considéré comme étant le plan de "mur". Ce terme fait référence de façon générale à tout ce qui délimite l'espace de la pièce dans lequel peuvent se trouver des objets qui bougent. Un meuble qui ne bougera jamais sera modélisé de fait comme étant un mur. Le sol et le plafond aussi sont alors considérés comme des murs. Aucun objet de premier plan ou autre objet du fond ne peut se trouver plus loin. Les autres objets de fond sont par exemple des meubles ou des chaises pouvant bouger mais destinés à incorporer le fond.

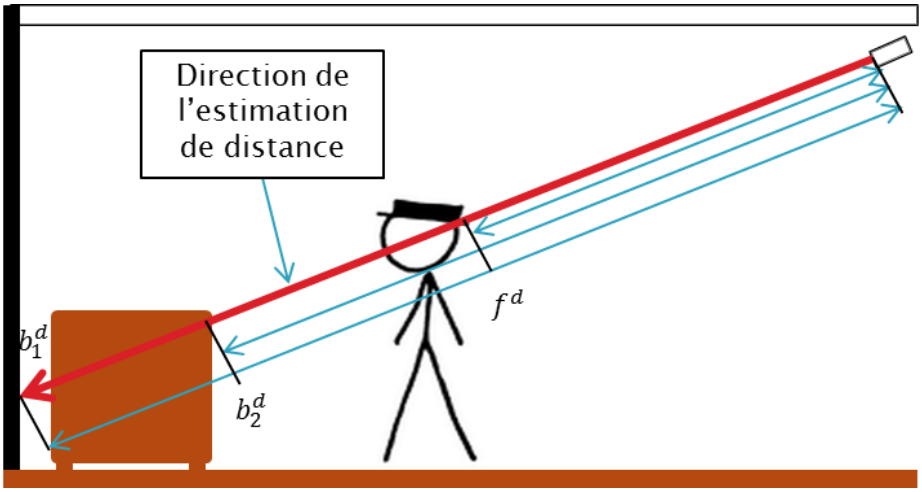


Figure 2-3: Schéma illustrant l'ordre des plans

L'objectif est d'associer un plan (F, B_k) à chaque pixel de l'image. Il serait naturel d'utiliser les équations 2.7 et de chercher la composante pour laquelle la probabilité est la plus forte.

$$P(x \in F|x) = \frac{\rho_f P(x|x \in F)}{P(x)} \tag{2.7a}$$

$$P(x \in B_k|x) = \frac{\rho_{b,k} P(x|x \in B_k)}{P(x)} \tag{2.7b}$$

Cependant, le calcul de ces probabilités pour chaque pixel possède un coût calculatoire élevé. On préférera utiliser la distance de Mahalanobis à l'espérance de chaque composante comme le fait Stauffer et al. [65]. On cherche quelle est la composante dont le pixel est le plus proche. Si cette distance est inférieure à un seuil prédéfini T et que le poids ρ_f ou $\rho_{b,k}$ de cette composante est assez élevé, on associe alors ce pixel à cette composante.

2.3.3 Mise à jour des paramètres du modèle

Afin de s'adapter à l'évolution du fond ou du premier plan, nous faisons ensuite évoluer les paramètres associés à chaque pixel de l'image avec la nouvelle valeur du pixel à l'instant t. L'approche utilisée pour l'évolution de ces paramètres est la suivante : le terme α est une constante d'apprentissage avec $0 < \alpha < 1$. Ce paramètre détermine l'importance de l'association du pixel à la composante et est un paramètre libre à régler décrivant en quelque sorte la bande passante des mouvements. Encore une fois nous tirons parti de Stauffer et al. [65].

Nous définissons la variable $M_{f,t}$ qui représente l'association à la gaussienne de premier plan. On a donc $M_{f,t} = 1$ si le pixel est associé à la composante de premier plan et $M_{f,t} = 0$ sinon.

Nous définissons les variables d'association $M_{k,t}$ de façon similaire pour les N_b composantes de fond : $M_{k,t} = 1$ si le pixel est associé à la composante de fond k et $M_{k,t} = 0$ sinon.

$(\bullet)_t$ est la valeur d'une variable pour l'image à l'instant t. Pour toutes les composantes, nous faisons évoluer le paramètre de poids avec les équations 2.8. Cela permet éventuellement d'intégrer dans le fond des objets qui restent immobiles après avoir bougé.

$$\rho_{k,t+1} = (1 - \alpha)\rho_{k,t} + \alpha M_{k,t} \tag{2.8a}$$

$$\rho_{f,t+1} = (1 - \alpha)\rho_{f,t} + \alpha M_{f,t} \tag{2.8b}$$

Pour la composante à laquelle le pixel est associé, nous faisons aussi évoluer l'espérance et la matrice de covariance de la distribution normale :

$$b_{k,t+1} = (1 - \alpha)b_{k,t} + \alpha x_t \tag{2.9}$$

$$f_{t+1} = (1 - \alpha)f_{,t} + \alpha x_t \tag{2.10}$$

$$\Sigma_{k,t+1} = (1 - \alpha)\Sigma_{k,t} + \alpha(x_t - b_{k,t})^t(x_t - b_{k,t}) \tag{2.11}$$

$$\Sigma_{f,t+1} = (1 - \alpha)\Sigma_{f,t} + \alpha(x_t - f_t)^t(x_t - f_t) \tag{2.12}$$

L'évolution des matrices $\Sigma_\bullet$ doit laisser les matrices diagonales. Nous forçons alors à 0 les valeurs en dehors de la diagonale.

Si le pixel se trouve trop loin de toutes les composantes, nous utilisons une heuristique qui se fonde sur les considérations de relation d'ordre sur la profondeur :

Si le pixel est derrière la gaussienne de mur, cela signifie que cette composante modélisait un meuble qui a été déplacé. L'espérance de cette composante est alors directement déplacée à la valeur du pixel observé.

Si le pixel est situé entre deux composantes de fond, nous affectons un poids nul aux gaussiennes situées plus près, nous déplaçons la gaussienne de premier plan à la place du pixel et nous lui affectons un poids faible et une variance élevée afin de permettre la ré-acquisition de celui-ci.

Si le pixel est plus près que toutes les gaussiennes de fond, nous déplaçons la gaussienne de premier plan au niveau du pixel et nous lui affectons un poids faible et une variance élevée.

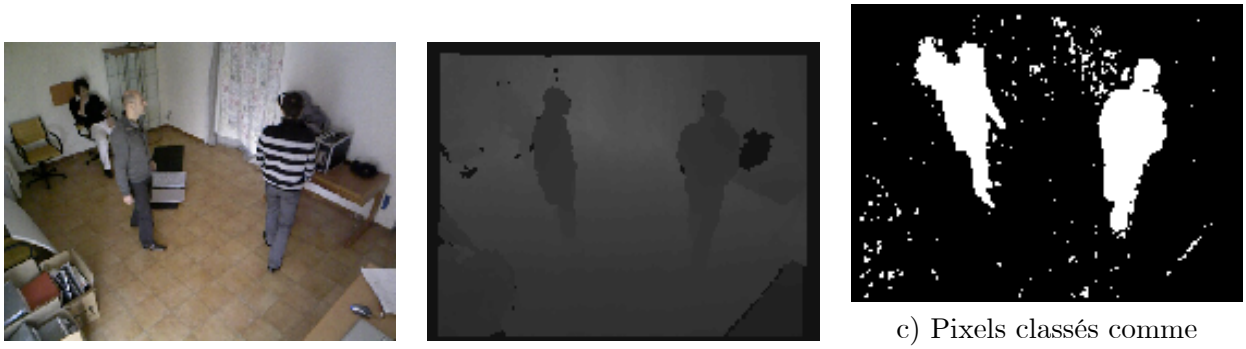
Lorsque le poids ρ_f de la composante de premier plan atteint un poids assez élevé (supérieur au seuil ρ_T), l'objet de premier plan est alors incorporé dans le fond. Ainsi lorsque l'on déplace un objet de la scène celui-ci ne reste pas un temps infini dans le premier plan. Pour assimiler une composante de premier plan au fond, on remplace les paramètres de la composante de fond ayant le poids $\rho_{b,k}$ le plus faible par les paramètres de la composante de premier plan. La gaussienne de premier plan prend alors un poids nul et la composante de profondeur f^d est alors déplacée à une distance nulle de la caméra. L'algorithme heuristique permet alors de redéplacer cette composante à l'endroit d'un premier plan réel lorsque que cela sera nécessaire.

A la fin de cette étape, on aura alors associé tous les pixels de l'image au fond ou au premier plan. La figure 2-4 illustre un exemple de soustraction de fond après la première étape.

La section 2.4 détaille l'étape suivante de regroupement de ces pixels en régions.

2.3.4 Implémentation du processus de classification

En pratique, on utilise seulement deux composantes de fond ($N_b = 2$). Nous avons réalisé des tests avec les séquences utilisées pour les autres évaluations en utilisant jusqu'à $N_b = 5$ composantes et observé que l'ajout d'autres composantes ne fait pas augmenter les



a) Image couleur. b) Image de profondeur. c) Pixels classés comme premier plan.

Figure 2-4: Pixels classés par mélange de gaussiennes.

performances de l'algorithme. En effet, il est extrêmement rare que deux objets ou plus se déplaçant viennent à rentrer dans le fond sur un même pixel.

Dans la présentation de Binney [13] est présenté l'écart-type de la valeur de l'estimation de profondeur d'un pixel en fonction de sa profondeur réelle. Afin de déterminer les paramètres de notre algorithme, nous avons utilisé un modèle polynomial pour modéliser cet écart-type ; celui-ci est illustré sur la figure 2-5. L'équation est alors la suivante 2.13 :

$$\sigma_{reg,d} = C_2 d^2 + C_1 d + C_0. C_0 = 7.713, C_1 = -0.005, C_2 = 2.61 * 10^{-6}. \tag{2.13}$$

Lors de l'introduction d'une nouvelle composante suite à l'utilisation de notre heuristique, son espérance est fixée égale à la valeur mesurée du pixel à ce moment-là. Pour déterminer la valeur de la variance de la profondeur de cette nouvelle composante, nous prenons en compte à la fois l'incertitude sur la position de la composante lors de son introduction ainsi que le bruit inhérent au capteur. Cette variance est alors initialisée à une valeur supérieure à la valeur de bruit correspondant à cette profondeur donnée par notre modèle polynomial (typiquement 2 fois cette valeur). Les variances sur la couleur sont fixées à des valeurs relativement élevées pour la même raison. Le poids ρ_k ou ρ_f de la composante est initialisé à une valeur supérieure à la valeur minimum pour qu'un pixel soit associé. En pratique, les valeurs exactes d'initialisation importent peu à condition qu'elles vérifient les conditions mentionnées précédemment. Nos expériences avec différentes valeurs n'ont pas montré de sensibilité significative à ces paramètres. L'évolution des paramètres définie par les équations 2.8 permet d'estimer de façon rapide les paramètres des composantes après leur introduction.

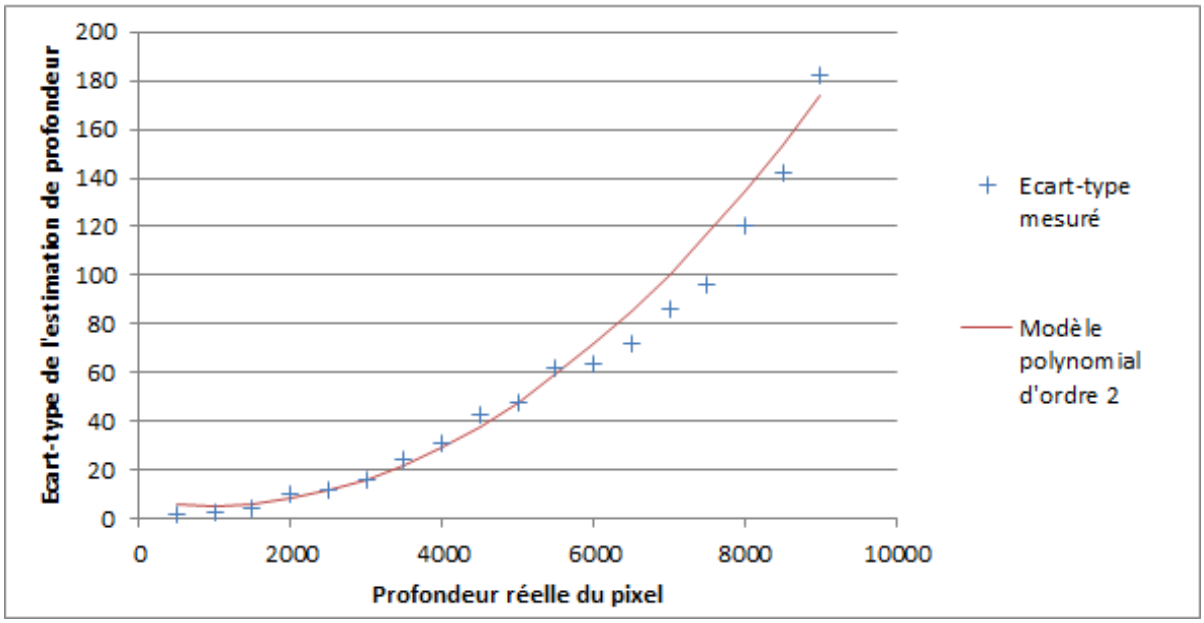


Figure 2-5: Ecart-type de l'estimation de profondeur en fonction de la valeur réelle.

Comme expliqué plus haut, nous prenons une valeur de $T = 2.5$ pour la distance maximum d'appartenance à une composante. Cela correspond à une probabilité de 98.8%

Le bruit sur l'estimation de la profondeur d'un pixel présente des statistiques non gaussiennes comme montré par Andersen et al. [8]. La variance sur la profondeur tend alors à prendre des valeurs trop faibles ce qui amène à un grand nombre de pixels faussement détectés comme premier plan. Pour s'en affranchir, nous empêchons cette variance de prendre une valeur inférieure à celle donnée par notre modèle polynomial 2-5.

Pour notre application il est important de connaître le temps qu'une personne immobile mettra à intégrer le fond. Ce temps doit être largement supérieur au temps que peut prendre une chute (qui peut potentiellement prendre plusieurs dizaines de secondes). Ce temps peut être calculé théoriquement pour un pixel complètement immobile grâce à l'équation 2.15 :

$$\rho_{f,t} - 1 = (1 - \alpha)^t (\rho_{f,0} - 1) \tag{2.14}$$

$$t = \log_{1-\alpha}(\frac{\rho_{f,t} - 1}{\rho_0 - 1}) \tag{2.15}$$

Ce paramètre α peut donc être réglé en fonction de l'application. Nous avons choisi de prendre une valeur pour laquelle un pixel mettra environ 1 min à intégrer le fond . La

valeur de α est modifiée à chaque image proportionnellement au temps exact de traitement de l'image pour atteindre ce résultat. Le paramètre ρ_T n'est important que pour son interaction avec le paramètre α ; nous l'avons donc fixé à $\rho_T = 0.7$ comme mentionné précédemment.

Initialisation

L'initialisation des composantes au lancement de l'algorithme n'est importante que pour celle associée au mur. Les autres composantes peuvent être simplement initialisées avec des poids $\rho_b = 0$. L'algorithme déplace automatiquement ces composantes grâce à l'heuristique sur la profondeur lorsque les pixels seront en mouvement et leur affectera un poids minimum pour que des pixels des images suivantes lui soient associés.

L'initialisation de la composante de mur se fait exactement comme lors de l'introduction d'une composante par l'heuristique à la différence près que son poids ρ_k est initialisé à 1.

Prise en compte des changements de luminosité

Du fait de la contrainte de diagonalité des matrices de covariance, il peut être judicieux de choisir un espace couleur différent de RGB pour les valeurs de couleur d'un pixel afin de prendre en compte plus facilement les changements de luminosité. En utilisant un espace tel que YCrCb, la variance sur la dimension Y correspond à la luminosité du pixel. La variance de la composante gaussienne sur cette dimension modélisera alors les changements de luminosité qui peuvent survenir sur la scène. Pour accommoder des changements de luminosité plus soudains on peut même décider de fixer la variance sur Y à une valeur très élevée.

2.4 Regroupement des pixels en régions mobiles

Pour la seconde étape de l'algorithme, nous passons d'une approche locale par pixel à une approche globale. Chaque pixel déterminé comme faisant partie d'un objet en mouvement doit maintenant être regroupé avec d'autres pixels du même objet. Nous allons d'abord expliciter l'algorithme avant d'évoquer les problèmes propres à l'implémentation.

2.4.1 Formalisation

Dans un premier temps, nous supprimons avec une fermeture morphologique de taille T_f les petites composantes, car nous ne cherchons pas à détecter de petits objets mais des personnes en mouvement. Puis nous cherchons à regrouper les pixels en composantes connexes dans l'image sans considération de profondeur. Nous utilisons une solution basée sur la recherche des contours qui est dans OpenCV [45]. On recherche les contours fermés qui ne sont pas confinés dans un autre contour fermé. Ces contours sont considérés comme les contours des composantes connexes de l'image. Les pixels à l'intérieur de ces contours sont alors les pixels des différentes composantes connexes.

Pour chaque composante connexe de l'image produite par le regroupement précédent, nous allons chercher à séparer les différents objets. En effet, il peut y avoir plusieurs objets dans une même composante connexe. L'information de profondeur permet de calculer la position de chaque pixel dans le repère en 3D de la caméra. Une composante connexe correspond alors à un nuage de points dans l'espace 3D. On utilise ensuite l'algorithme de recherche-fusion présenté dans l'algorithme 2.4.1 issu de Cormen et al. [16] pour chercher dans ce nuage de points les différents objets. Chaque groupe de pixels est décrit comme une liste chaînée.

Deux pixels font partie du même objet lorsque la distance euclidienne dans l'espace 3D entre ces deux pixels est inférieure à une certaine distance T_r i.e $crit(x, y) := (||x - y||^2 < T_r^2)$. Chaque pixel va alors se voir attribuer un label. Tous les pixels ayant le même label correspond alors à un même objet.

2.4.2 Implémentation et valeurs des paramètres

La valeur de T_r pour la clusterisation dans l'espace 3D doit être de choisie de façon à bien séparer des objets relativement proches mais elle ne doit pas être choisie trop faible pour pouvoir être robuste au bruit de mesure de la profondeur. Deux pixels adjacents à $8m$ du capteur seront séparés d'une distance d'environ $15cm$ dans l'espace. Typiquement la valeur est fixée à $T_r = 20cm$ afin d'accommoder cela en plus du bruit de mesure.

Cet algorithme possède une complexité $O(n^2)$ où n est le nombre de points dans la composante connexe. En pratique, en fonction de la taille des composantes connexes, ce coût peut être prohibitif pour une intégration en temps réel. On va alors ne prendre qu'une fraction déterminée à l'avance (1/16 typiquement) des pixels de la composante pour leur appliquer l'algorithme, puis étendre la labélisation aux pixels voisins détectés comme étant

Algorithm 1 Union-Find

```
1: Function{Find}{x}
2: if x.parent = x then
3:    return x
4: else
5:    return Find(x.parent)
6: end if

7: Function{Union}{x,y}
8: xRoot ← Find(x)
9: yRoot ← Find(y)
10: xRoot.parent ← y
11: EndFunction

12: Function{Partition}{S}
13: for each x in S do
14:    xRoot ← x
15: end for
16: for each x in S do
17:    for each y in S \ x do
18:       if crit(x, y) then
19:          Union(x, y)
20:       end if
21:    end for
22: end for
23: EndFunction
```

en mouvement mais qui n'ont pas été utilisés pour l'algorithme de clusterisation.

2.5 Expérimentations et évaluations associées

Nous avons conduit une série d'expérimentations pour évaluer les performances de notre algorithme. Les valeurs privilégiées de paramètres libres que nous avons utilisées pour ces expérimentations sont résumées dans le tableau 2.1. Nous avons choisi de caractériser les performances de notre algorithme sur plusieurs situations types : personne debout, personne au sol, présence de fantôme. Pour une image, nous définissons à l'instar de Benezeth et al.[12], deux critères appelés précision et recall :

- $précision = \frac{Nb\ pixels\ correctement\ détectés\ comme\ F}{Nb\ pixels\ détectés\ comme\ F}$

- $recall = \frac{Nb\ pixels\ correctement\ détectés\ comme\ F}{Nb\ pixels\ du\ premier\ plan}$

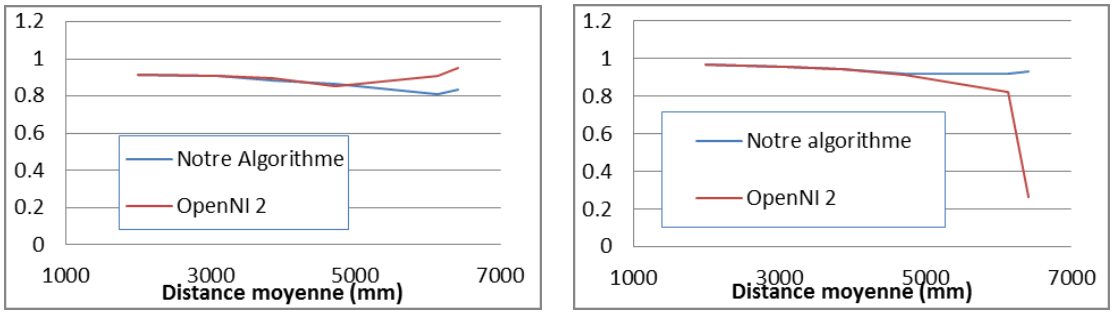
La précision exprime la capacité de l'algorithme à éviter les fausses détections de pixels, alors que le recall exprime la capacité à bien détecter les pixels qui sont effectivement des pixels en mouvement.

Taille de l'élément morphologique pour la fermeture	T_f	3
Poids maximal de la composante de premier plan	ρ_T	$0,7$
Seuil de distance de Mahalanobis pour appartenir à une composante	T	$2,5$
Poids d'initialisation des composantes		$0,05$
Facteur d'apprentissage	α	$10^{-4} * Temps\ de\ calcul$
Distance maximale entre deux pixels de la même composante	T_r	$20cm$
Variance d'initialisation couleur	σ_r, σ_g, σ_b	60
Coefficient du modèle d'écart-type	C_0, C_1, C_2	$C_0 = 7.713$, $C_1 = -0.005$, $C_2 = 2.61.10^{-6}$.
Variance d'initialisation profondeur	σ_d	$2.\sigma_{reg,d}$
Nombre de composantes de fond	N_b	2

Table 2.1: Valeurs des paramètres libres de notre algorithme.

2.5.1 Evaluations pour une personne debout

Les courbes 2-6 montrent les performances en recall et en précision de notre algorithme comparé à celui d'OpenNI. Pour tracer ces courbes, la vérité terrain a été déterminée à la main sur différentes images de couleur de différentes séquences de personnes se déplaçant face à la caméra de plus en plus loin. Trois séquences ont été utilisées, dans chacune desquelles 15 images de la personne à différentes distances ont été utilisées.

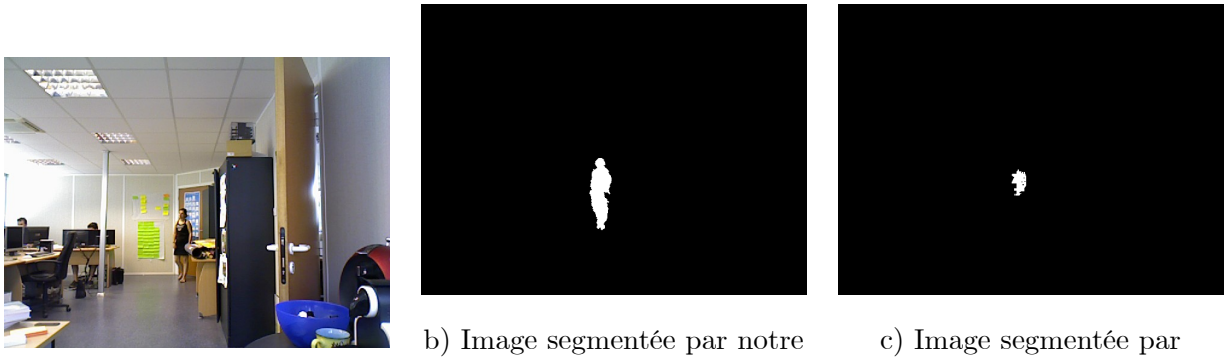


Précision. Recall.

Figure 2-6: Precision et recall pour personnes debouts.

La distance en abscisse correspond à la distance moyenne des pixels de la silhouette. Cette distance est calculée avec la valeur des pixels de l'image de profondeur.

Dans le domaine d'utilisation nominale du capteur ($< 6m$), les deux algorithmes ont des performances très proches. Cependant, au-delà de $6m$, alors que les performances en précision de l'algorithme d'OpenNI se dégradent, notre algorithme continue à détecter les personnes avec des performances qui restent suffisantes pour l'application qui nous intéresse. La chute du critère de recall et non de la précision traduit le fait que très peu de pixels sont détectés en dehors du corps humain. La dégradation des performances avec la distance vient principalement des pixels faussement détectés comme faisant partie du fond.



a) Image Couleur.

b) Image segmentée par notre algorithme.

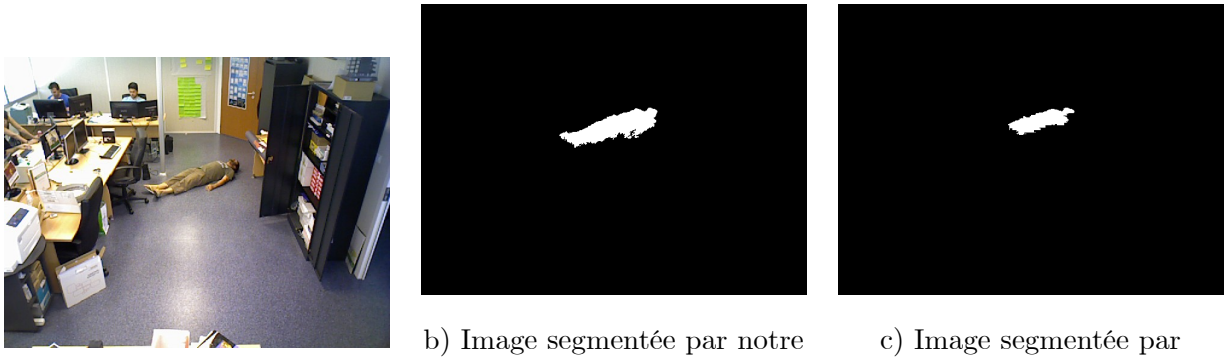
c) Image segmentée par OpenNI.

Figure 2-7: Segmentation d'une personne debout.

La figure 2-7 illustre les différences de segmentation entre notre algorithme et OpenNI pour une personne debout à une distance importante du capteur. Nous voyons que notre algorithme est substantiellement plus performant que celui d'OpenNI.

2.5.2 Evaluations pour une personne au sol

Pour rappel, notre algorithme doit être capable de détecter des personnes allongées au sol. La figure 2-8 illustre un exemple de segmentation par notre algorithme et montre la différence entre celui-ci et OpenNI. Nous voyons qualitativement que notre segmentation est meilleure.



a) Image couleur.

b) Image segmentée par notre algorithme.

c) Image segmentée par OpenNI.

Figure 2-8: Segmentation d'une personne allongée.

Le tableau 2.2 présente les résultats des tests effectués avec des personnes allongées au sol à différentes distances de la caméra. Dix images de la même personne au sol ont

été prises à différents endroits d'une pièce et dans différentes orientations par rapport à la caméra.

	Notre algorithme	OpenNI2
Précision	0.85	0.99
Recall	0.90	0.53

Table 2.2: Precision et recall pour personnes allongées au sol.

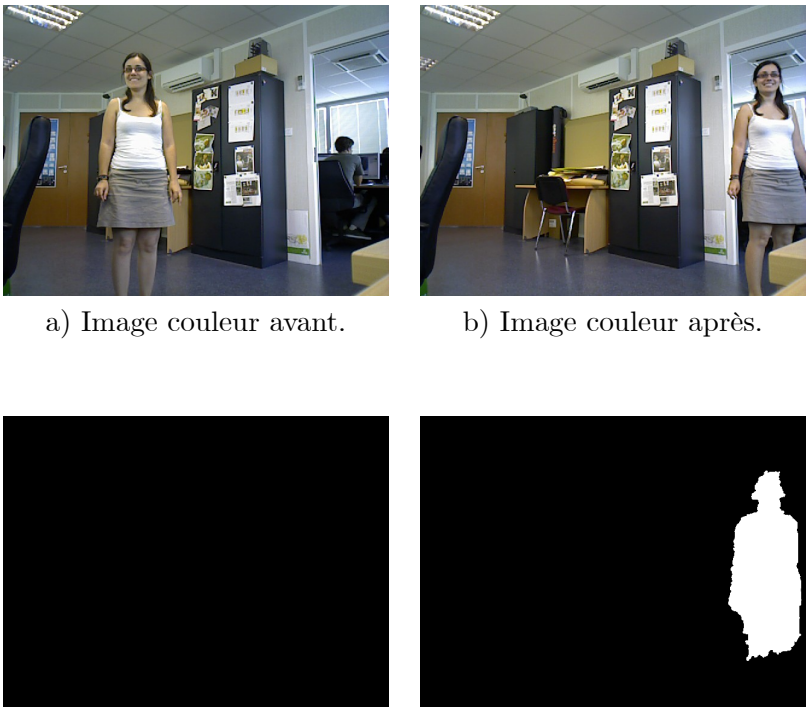
La précision très élevée et le recall très faible de l'algorithme d'OpenNI au sol vient du fait que peu de pixels sont détectés mais leur quasi-totalité sont des bonnes détections. Au contraire, notre algorithme possède un taux de précision plus faible bien que satisfaisant mais un taux de recall bien plus élevé qu'OpenNI. L'analyse que l'on peut faire de ces résultats est la même que pour la segmentation de personnes debout. La cause de la chute du critère de recall vient du grand nombre de pixels non détectés comme faisant partie de la personne. Notre algorithme fournit donc une segmentation de meilleure qualité que celle fournie par OpenNI pour une personne au sol. Cela est essentiel dans notre contexte applicatif de détection de chute.

La figure 2-9 montre un exemple de segmentation pour plusieurs images d'une séquence de marche et de chute. Quelle que soit la distance de la personne au capteur, sa position dans la scène ou sa posture, notre algorithme parvient à segmenter la personne.

Figure 2-9: Segmentation d'une séquence de marche et chute.

2.5.3 Suppression des fantômes

La figure 2-10 illustre un exemple de gestion des fantômes par notre approche. La figure 2-10c montre le résultat de la segmentation après qu'une personne ait été intégrée dans le fond. La figure 2-10d montre le résultat lorsque la personne s'est déplacée. Nous voyons alors l'apport de notre heuristique sur la profondeur pour l'élimination des fantômes. Comme relevé en introduction, OpenNI ne permet pas aux fausses détections d'intégrer le fond contraitrement à notre approche.



a) Image couleur avant. b) Image couleur après.

c) Segmentation avant. d) Segmentation après.

Figure 2-10: Exemple de suppression des fantômes.

2.5.4 Détection de plusieurs personnes

La figure 2-11 illustre la capacité de notre algorithme à segmenter différentes personnes même lorsque celles-ci s'occultent mutuellement. Un problème demeure : lorsqu'une personne est séparées en deux composantes distinctes dans l'image comme c'est le cas sur la figure 2-11c où la personne en rouge est séparée en deux par la personne en vert, deux composantes distinctes sont produites.

Notre algorithme ne permet pas de séparer deux personnes qui sont trop proches.

a) Image couleur. b) Image de profondeur. c) Image couleur avec segmentation.

Figure 2-11: Segmentation de plusieurs personnes avec occultation.

OpenNI permet de le faire dans certains cas mais seulement lorsque les personnes sont nettement séparées avant de se rapprocher.

2.5.5 Autres limites observées

Une autre faiblesse de l'algorithme proposé est le fait que les personnes immobiles finissent par ne plus être détectées. OpenNI souffre du problème inverse où certains objets détectés comme étant des personnes restent indéfiniment détectées comme premier plan. Ce problème est illustré sur la figure 2-12.

a) Segmentation de deux personnes avec silhouettes distinctes. b) Segmentation de deux personnes dans la même silhouette.

Figure 2-12: Segmentation de deux personnes proches.

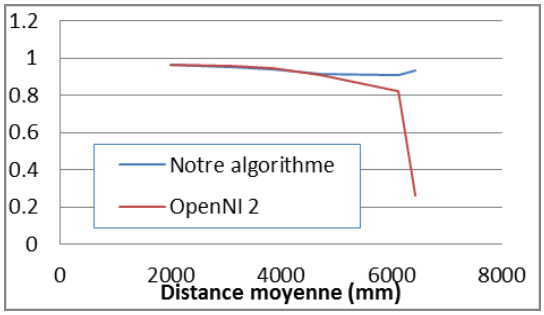
Un objet en mouvement qui n'est pas une personne (comme une chaise par exemple) sera détecté comme une composante mobile mais cela est aussi le cas pour OpenNI 2 comme montré précédemment. Avec notre algorithme cependant, l'objet finira par être incorporé dans le fond.

2.5.6 Coût CPU

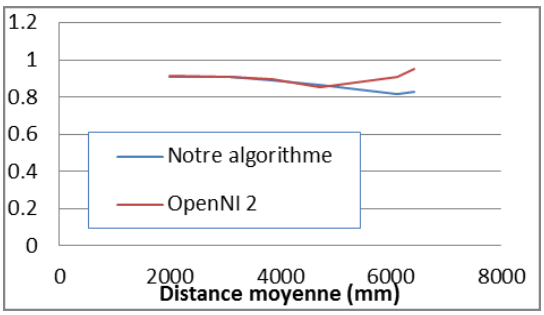
Le temps de calcul de l'algorithme est variable, ceci à cause de l'étape de regroupement en régions mobiles dont le temps d'exécution dépend du nombre de pixels en mouvement à l'étape précédente. En moyenne, le temps d'exécution est de $20ms$ sur un coeur à 2Ghz dont $18ms$ pour l'étape de regroupement en régions mobiles.

2.5.7 Approche avec canal profondeur seul

Pour rappel, le système doit être capable de fonctionner la nuit. Cela nous oblige donc à n'utiliser que l'information de profondeur car l'information de couleur n'est pas exploitable pour une scène sous-éclairée ou obscure. Nous avons donc évalué notre algorithme sur la même base mais en utilisant cette fois que l'information de profondeur.



Précision.



Recall.

Figure 2-13: Performances en utilisant la profondeur seulement

Les performances uniquement avec la profondeur sont extrêmement proches des performances obtenues en associant la mesure de couleur comme illustré sur la figure 2-13. Les faibles gains observés avec l'ensemble des canaux RGB-D sont très dépendants des conditions d'illuminations et du contraste de couleur entre les objets à détecter et le fond de la scène. Les canaux RGB semblent donc moins discriminants que le seul canal profondeur.

2.6 Conclusion

Nous avons présenté une approche fondée sur des mélanges de gaussiennes en quatre dimensions RGB-D auxquelles nous ajoutons des heuristiques permettant d'exploiter les caractéristiques propres à la mesure de profondeur. Grâce à cela, et contrairement à d'autres approches, notre algorithme ne souffre pas de l'apparition de fantômes qui peuvent appa-

raître avec des algorithmes uniquement en RGB ou lorsque la profondeur est simplement utilisée comme une 4ème composante. Ainsi, lors de la réapparition du fond, on ne constate pas de fausse détection. De plus, notre algorithme permet de segmenter des personnes dans des postures variées et pas seulement debout. Il est donc utilisable dans un grand nombre de cas et est bien adapté au contexte applicatif des chutes où les postures sont très diverses. Cependant notre algorithme présente des limites lorsque plusieurs personnes sont présentes dans la pièce et que celles-ci sont trop proches les unes des autres. Ceci pourrait être amélioré en modifiant l'algorithme de clusterisation 3D. En résumé, nous avons montré que notre algorithme est particulièrement bien adapté à la segmentation de personnes seules dans une pièce de dimension limitée. Il a de surcroît été montré que l'algorithme est utilisable la nuit lorsque l'image de profondeur seule est utilisée. Par ailleurs, nous montrons, dans un contexte similaire de vidéo-surveillance, des performances meilleures que les algorithmes de OpenNI.

Chapitre 3

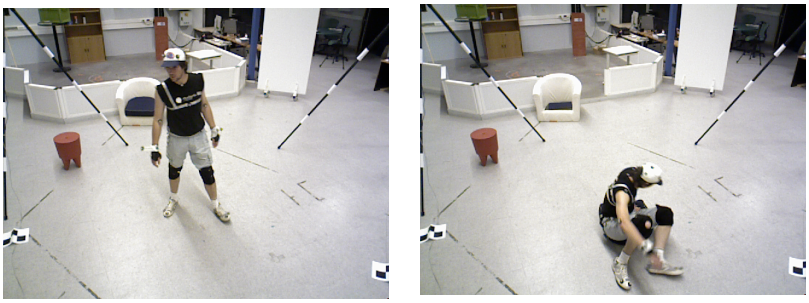
Estimation de la posture humaine par image de profondeur

3.1 Introduction

Rappelons ici notre problématique d'estimation de posture humaine et résumpns notre approche avant d'énoncer le plan du chapitre.

Le but est d'inférer les positions 3D des articulations à l'instar de Nite [3]. Nous fournissons un ensemble de candidats avec un seuil de confiance pour les positions d'une sélection des articulations d'une personne présente dans le champ de vision du capteur et dont la silhouette a été segmentée par l'algorithme décrit au chapitre précédent. Seule l'information de profondeur est utilisée pour cela. L'image de couleur fournie par le capteur n'est pas exploitée ici pour s'affranchir des conditions d'illumination. Les solutions existantes sur Kinect sont soumises à un droit de licence ce qui justifie le re-développement et la maîtrise du "pipeline" complet de reconstruction de posture. Par ailleurs, cet aspect est important pour les autres besoins de la société ORME. Notre approche est fortement inspirée de Girshick et al. [27] dont les performances et la robustesse dans un contexte industriel ont été démontrées pour des applications de "Motion Gaming" grand public à domicile. Cette solution s'est montrée capable de fonctionner en environnement peu contrôlé, toutefois en se limitant à des mouvements proches des mouvements fronto-parallèles comme illustré sur la figure 3-1a.

De plus, cet algorithme est prévu pour fonctionner avec le capteur posé dans le bon sens et à une hauteur correspondant à celle d'un meuble de télévision. Pour notre application, nous avons en revanche besoin de pouvoir estimer la posture de personnes alors que le



a) Posture fronto-parallèle. b) Posture de chute.

Figure 3-1: Exemples de postures à estimer.

capteur est accroché à l'envers au plafond de façon plongeante et lorsque la personne se trouve dans une variété de postures non prises en charge par les implémentations existantes telles que Nite [3], en particulier des postures de personnes chutant ou allongées au sol comme illustré sur la figure 3-1b.

Nous présentons une approche fondée sur la classification des parties du corps couplée à une régression de la position des articulations. Elle se fonde sur l'utilisation des Random Forests appliquées à chaque pixel de la silhouette préalablement segmentée. Les contributions de chaque pixel à la position de chaque articulation sont potentiellement prises en compte et agrégées pour inférer la position 3D de chaque articulation. Nous privilégions deux types de caractéristiques : (1) la profondeur à l'instar de Shotton et al. [51], (2) la hauteur du pixel par rapport au sol qui se justifie en particulier par notre application de détection de chutes. Pour l'apprentissage des Random Forests, nous avons besoin d'une grande base de données d'images représentatives des morphologies et des scénarios (postures) pour notre application. Pour cela, l'utilisation d'images de synthèse facilite grandement la création d'une base au prix d'un réalisme diminué.

Dans un premier temps nous présentons des travaux similaires et exposons le principe des Random Forests pour l'application qui nous intéresse. Nous présentons alors notre approche et la positionnons par rapport à l'état de l'art. Nous continuons en évaluant les performances de notre algorithme et ses différentes variantes une à une de façon autonome et à nouveau par rapport à l'état de l'art. Enfin nous concluons en résumant nos travaux et en présentant de possibles extensions.

3.2 Travaux similaires

La communauté Vision a largement investigué l'estimation de posture humaine par des techniques de Vision par Ordinateur. Des "surveys" ont été réalisées, notamment par Moeslund et al. dans [43]. Nous nous intéressons d'abord aux travaux utilisant la profondeur. Puis nous nous concentrons sur ceux utilisant les Random Forests ; ce classifieur multi-classes est très adapté dans notre contexte.

3.2.1 Estimation à partir de la profondeur

Des travaux récents ont été réalisés qui exploitent les images de profondeur et en particulier le Kinect [42].

Anguelov et al. dans [9] segmentent une image de profondeur en fond et différentes parties du corps en utilisant des champs de Markov. Grest et al. dans [28] se placent dans une logique de suivi et utilisent un algorithme ICP (pour Iterative Closest Point) pour suivre un squelette dont ils connaissent la taille et la posture initiale dans une image de profondeur. Utilisant encore une fois la segmentation en parties du corps, Zhu et al. dans [63] [64] partent d'une pose en T pour calibrer un modèle qui sera ensuite suivi par programmation linéaire. Avec des détecteurs adaptés à chaque partie du corps, Siddiqui et al. dans [53] utilisent un modèle de MCMC (pour Monte Carlo Markov Chains) pour suivre les positions de certains membres spécifiques. Plagemann et al. dans [48] construisent un modèle 3D complet à partir du nuage de points de la silhouette et trouvent les extrémités géodésiques de ce modèle qui correspondent aux pieds, à la tête et aux mains. Aucune distinction n'est cependant faite entre la droite et la gauche. Récemment, Ye et al. dans [60] utilisent un modèle 3D articulé dont ils estiment les paramètres de forme en l'ajustant au nuage de points. Ils propagent ensuite la pose par optimisation itérative. Cet axe de recherche est en plein développement et a connu de grandes avancées ces dernières années, notamment grâce à l'utilisation de Random Forests comme présenté ci-après.

3.2.2 Travaux utilisant les Randoms Forests

Les Random Forests ont été largement utilisées en vision, notamment pour la classification d'images (Leistner et al. dans [39]), la segmentation d'images et la détection d'objets par Gall et al. dans [26] et Kontshieder et al. dans [37] ou encore la reconnaissance d'actions par Gal et al. dans [26]. Cet outil a permis une grande avancée dans l'estimation de

pose par images de profondeur avec Shotton et al. dans [51]. Ils utilisent les images de profondeur avec des Random Forests pour classifier les pixels en parties du corps et estimer la position. Inspirés par ces travaux, différentes méthodes ont amélioré cet algorithme. Construisant au-dessus du principe de reconnaissance par parties, Girshick et al. dans [27] ajoutent une dimension de régression pour estimer directement les parties du corps dans les feuilles de l'arbre. Kholi et al. dans [55] ajoutent une composante conditionnelle à la régression pour contraindre a priori l'espace des poses à estimer. Une généralisation supplémentaire à la régression par Taylor et al. dans [56] utilise un modèle en 3D du corps humain pour créer des correspondances entre pixels de l'image et points de la surface de ce modèle. Cet algorithme est cependant coûteux et difficile à mettre en oeuvre. Yu et al. dans [61] utilisent des Regression Forests pour estimer la position 3D de personnes dans des images 2D. La détection d'action est combinée à une détection des membres dans l'image 2D. Hara et al. dans [30] utilisent un graphe de dépendance entre points du corps, les positions des membres sont estimées successivement par des Regression Tree en utilisant ce graphe. Dantone et al. dans [19] cherchent à obtenir des descripteurs non-linéaires des parties du corps grâce à des Random Forests en deux couches dans des images couleur. Le nombre important de travaux récents utilisant les Random Forests pour l'estimation de posture humaine et les bonnes performances de ceux-ci laissent penser que cette approche est prometteuse.

Nous présentons dans ce qui suit en détail les travaux de Shotton et al. dans [51] et Girshick et al. dans [27]. Notre approche s'appuie largement sur ces travaux qui ont été résumés par Shotton et al. dans [52].

3.3 Randoms Forests pour l'estimation de posture humaine

Nous présentons dans cette section en détail le formalisme des Random Forests pour l'estimation de posture à l'aide d'une seule image de profondeur.

Les travaux de Shotton et al. dans [51] constituent un jalon majeur dans l'estimation de posture humaine. Leurs résultats utilisant les Random Forests se sont avérés largement plus performants que ceux de l'état de l'art sur un ensemble de postures très variées. Leurs travaux, dénommés BPC (pour Body Part Classification), se placent dans un premier temps dans une approche de pure classification. Cette approche vise à déterminer la classe de chacun des pixels de l'image correspondant à une personne en termes de partie

du corps. Les classes utilisées dans notre implémentation sont représentées sur la figure 3-2.

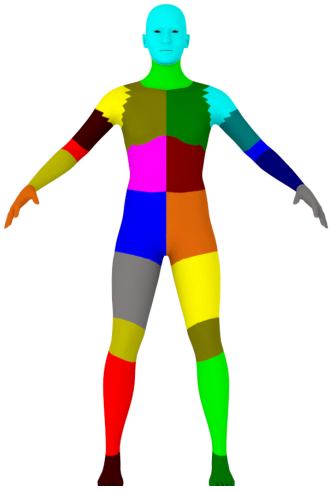


Figure 3-2: Classes des parties du corps humain considérées.

Aux feuilles des arbres sont associées des distributions empiriques de probabilité visant à labelliser en parties corporelles chacun des pixels issus de la silhouette préalablement segmentée. Grâce à un ensemble de postures et d'images d'apprentissage très important, les auteurs parviennent avec de très bons résultats à estimer ces classes sur de nouvelles images, qu'elles soient simulées ou réelles. Une fois ces classes estimées, les auteurs utilisent un algorithme de Mean-Shift [15] pour trouver la position des articulations 3D. Ils trouvent alors dans l'espace, le centre des parties corporelles et appliquent un "offset" pour trouver la position des articulations.

Les travaux de Girshick et al. dans [27] dénommés OJC (pour Offset Joint Regression OJC) s'inspirent de Shotton et al. [51] mais montrent de réels gains de performance. Les mêmes caractéristiques et les mêmes jeux de données sont utilisés. Leur approche cependant, utilise la régression pour estimer la position des articulations directement avec l'arbre en utilisant une information différente dans les feuilles. En effet, dans l'approche de Girschick, une information directe sur la position des articulations est associée aux feuilles. Le critère d'apprentissage des structures des arbres reste le même que celui de Shotton et al. dans [51] avec un critère de classification. La figure 3-3 illustre les deux approches BPC et OJC en terme de précision de reconstruction pour chaque membre (Cf. section 3.5 pour la définition exacte du critère "average precision").

Malheureusement, le réglage de certains paramètres libres utilisés dans ces deux ap-

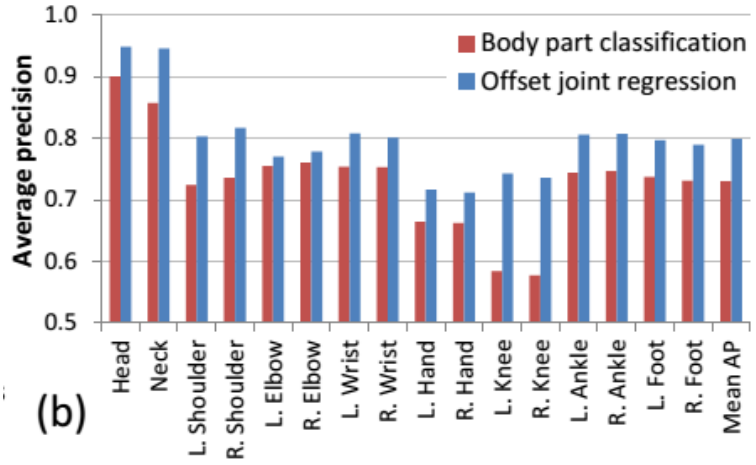


Figure 3-3: BPC vs. OJC en termes de performance (Shotton et al 2013)[51].

plications est peu argumenté. Pour obtenir de bons résultats avec une implémentation propre, il nous faut alors régler les valeurs de ces paramètres.

3.3.1 Formalisation de l'algorithme de Random Forest

Une forêt est un ensemble de T arbres de décision. Un arbre de décision est un arbre binaire contenant des noeuds de séparation ayant chacun deux enfants et des noeuds de décision appelés aussi feuilles qui n'ont pas d'enfants. Chaque noeud de séparation contient une caractéristique f_θ et un seuil τ. La caractéristique f_θ est une fonction de l'ensemble $\{f_\theta : (x, I) \rightarrow R | \theta \in \Theta\}$ prenant en paramètre un pixel x et son image associée I et renvoyant un scalaire. L'ensemble Θ est l'ensemble des paramètres possibles pour la caractéristique. Un noeud de décision est donc représenté par un couple $\phi = (\theta, \tau)$.

Chaque feuille contient l'information à inférer sur les pixels. Cette information est la classe du pixel (resp la position des articulations) dans une approche BPC (resp OJC).

Exploitation de la forêt

Lors de la phase de prédiction par une Random Forest, nous faisons traverser chaque arbre par le pixel pour lequel nous cherchons à faire une prédiction comme illustré sur la figure 3-4. La feuille de chaque arbre à laquelle le pixel arrive fournira alors une prédiction pour ce pixel. Afin de faire une prédiction d'un arbre pour un pixel x de l'image I, on commence par le premier noeud de l'arbre i.e le noeud racine. Nous utilisons l'indice n pour indicer les différents noeuds de l'arbre. Le noeud racine possède alors l'indice 0. Les paramètres

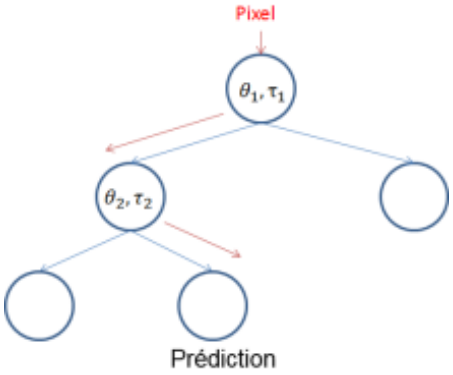


Figure 3-4: Schéma réduit d'un pixel traversant un arbre.

du noeud de décision n sont alors notés $\phi_n = (\theta_n, \tau_n)$. La fonction binaire $h_{\phi_n}(x, I)$ de l'équation 3.1 de chaque noeud de décision permet de déterminer quel noeud doit être testé selon :

$$h_{\phi_n}(x, I) = [f_{\theta_n}(x, I) \leq \tau_n] \tag{3.1}$$

A chaque noeud traversé par le pixel, nous comparons la valeur de la caractéristique à ce noeud pour ce pixel soit $f_\theta(x)$. Si $f_\theta(I, x) \leq \tau$, le processus continue pour le noeud suivant à gauche sinon on continue pour le noeud suivant à droite. Ce processus est répété jusqu'à arriver à une feuille. Les prédictions pour chaque arbre et chaque pixel sont ensuite agrégées pour former un ensemble final de prédiction pour chaque partie corporelle ou articulation.

Apprentissage de la structure des arbres d'une forêt

Une Random Forest est un ensemble d'arbres de décision. Ces arbres sont entraînés indépendamment les uns des autres avec une collection de pixels différente pour chaque arbre. Le nombre T d'arbres utilisés est un paramètre libre de l'algorithme.

Pour l'entraînement d'un arbre, une collection de pixels avec leurs images I associées $Q_t = \{(x, I)\}$ avec $t = 1..T$ est utilisée. Le but est d'apprendre les paramètres (θ, τ) à chaque noeud de chaque arbre.

Pour l'apprentissage des caractéristiques d'un noeud, l'algorithme suivant est utilisé :

L'ensemble Q_n est l'ensemble des pixels utilisés pour l'apprentissage du noeud n.

1. Un ensemble Φ de candidats ϕ est choisi pour les paramètres (θ, τ).

Pour chaque $\phi \in \Phi$:

2. Deux ensembles $Q_g(\phi)$ et $Q_d(\phi)$ respectivement l'ensemble des pixels allant à gauche et l'ensemble des pixels allant à droite sont créés pour ces paramètres Φ :

$$Q_g(\phi) = \{(I, x) | f_\theta(I, x) \leq \tau\} \tag{3.2}$$

$$Q_d(\phi) = Q / Q_g(\phi) \tag{3.3}$$

3. On cherche $\phi^* \in \Phi$ qui donne le plus grand gain d'information $G(\phi^*)$. La fonction $H(Q)$ est une mesure d'information que nous explicitons plus loin :

$$\phi^* = \arg\max_{\phi} G(\phi) \tag{3.4}$$

$$G(\phi) = H(Q) - \sum_{s \in \{g,d\}} \frac{|Q_s(\phi)|}{|Q|} H(Q_s(\phi)) \tag{3.5}$$

Si le gain d'information $G(\phi^*)$ est suffisant et que la profondeur du noeud est inférieure à la profondeur maximum, on recommence la procédure pour les noeuds enfants de gauche et droite avec respectivement les ensembles de pixels $Q_g(\phi^*)$ et $Q_d(\phi^*)$. Si on atteint le critère d'arrêt, soit parce que l'on se trouve à la profondeur maximale de l'arbre, soit parce que le gain d'information $G(\phi^*)$ est trop faible, le noeud est considéré comme un noeud feuille.

Pour l'apprentissage des noeuds, nous avons besoin d'une fonction $H(Q)$ qui évalue l'information statistique contenue dans une collection de pixels. Pour chaque élément $c \in C$ définissant la classe d'un pixel, il est possible de calculer l'entropie statistique de la distribution de c pour l'ensemble des pixels de la collection. La probabilité empirique d'une classe de pixels est définie par :

$$P_Q(c) = \frac{|\{x \in Q | C(x) = c\}|}{|Q|} \tag{3.6}$$

où $|Q|$ désigne le cardinal de l'ensemble Q. L'opposé de l'entropie d'information définie dans l'équtation 3.7 est alors une mesure d'information que nous retiendrons comme critère de classification :

$$H(Q) = -\sum_{c \in C} P_Q(c) \log(P_Q(c)) \tag{3.7}$$

Les investigations de Girshick et al. dans [27] montrent la pertinence de ce critère de

classification pour la construction des arbres dans l'approche de régression. C'est donc celui-ci que nous allons utiliser pour notre implémentation.

Les approches de régression classiques de Random Forests utilisent des critères fondés sur la distribution de l'information à inférer. Or, dans notre cas, comme montré par Girshick et al. [27], ces distributions sont extrêmement multi-modales. Par conséquent, les approches classiques mono-modales de modélisation de ces distributions ne sont pas adaptées à notre contexte, celles-ci ayant l'inconvénient supplémentaire d'être très coûteuses en temps de calcul.

3.3.2 Caractéristiques

Nous utilisons celles décrites par l'équation 3.8 introduite par Gal et al. [26] et utilisées par Shotton et al. [52]. Leur expression est la suivante :

$$f_{\theta,1}(I,x) = d_I(x + \frac{u}{d_I(x)}) - d_I(x + \frac{v}{d_I(x)}) \tag{3.8}$$

Le terme $d_I(x)$ désigne la profondeur du pixel x dans l'image I. Le paramètre $\theta = (u, v)$ est celui de la caractéristique et correspond à des décalages dans l'image. La figure 3-5 illustre deux exemples de caractéristiques calculées sur un même pixel. Leur normalisation par $d_I(x)$ permet de rendre la caractéristique indépendante de la profondeur. En tout point du corps, nous avons alors un décalage fixe dans l'espace, quelle que soit la distance du pixel à la caméra. Pour un pixel x' ne faisant pas partie du corps ou en dehors des limites de l'image, nous donnons à $d_I(x')$ une valeur positive très élevée.

Une caractéristique de ce type donne peu d'information pour classifier un pixel lorsqu'elle est isolée. Cependant, la combinaison dans un arbre de nombreuses caractéristiques permet de pallier cela tout en préservant un temps de calcul très limité.

3.3.3 Modèle de prédiction des feuilles

Dans chaque feuille de chaque arbre est stocké un modèle de prédiction. Deux modèles de prédiction distincts peuvent être utilisés, un modèle de classification des pixels (Shotton et al. [51]) ou un modèle de régression (Girshick et al. [27]).

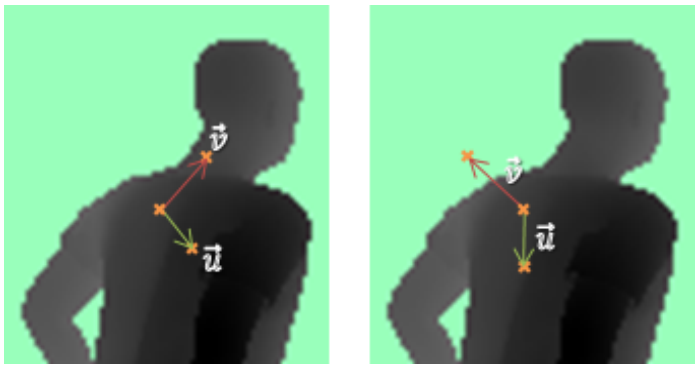


Figure 3-5: Exemples de deux caractéristiques θ sur un même pixel.

Modèle de classification

Dans le modèle de classification qui est celui utilisé par Shotton et al. [51], nous cherchons à inférer directement la classe d'un pixel. Pour chaque feuille l de chaque arbre t, nous stockons une distribution $p_{t,l}(c)$ sur les classes c pixels. Un pixel x va alors descendre chaque arbre t pour aboutir dans une feuille l. Nous avons alors pour chaque pixel $p(c|x, I, t, l) = p_{t,l}(c)$. Les prédictions sont ensuite moyennées sur la forêt avec l'équation 3.9 pour donner la prédiction finale de la classe d'un pixel :

$$p(c|x, I) = \frac{1}{T} \sum_{t=1..T} p(c|x, I, t, l) \tag{3.9}$$

Pour obtenir ensuite la position des membres, les prédictions sont agrégées grâce à la technique de Mean-Shift de Comaniciu et al. [15]. Celle-ci infère alors un ensemble de prédictions des positions 3D des centres des classes. Un "offset" dépendant de l'articulation considérée et appris a priori est ajouté pour donner enfin la position de l'articulation.

Lors de l'apprentissage, il s'agit de d'apprendre conjointement la structure de l'arbre ainsi que l'information à stocker dans chaque feuille. Le but de l'apprentissage des feuilles, dans le cas de la classification, est d'estimer de façon non paramétrique la distribution $p_{t,l}(c)$ des classes qui arrivent à ce noeud.

Chaque pixel de la collection Q traverse les arbres. Nous comptons alors le nombre de pixels de chaque classe qui arrivent à chaque feuille. On obtient alors la distribution empirique $p_{t,l}(c)$ définie par l'équiation 3.6 que l'on obtient en divisant le nombre de pixels de chaque classe par le nombre total de pixels qui ont atteint la feuille. L'ensemble des

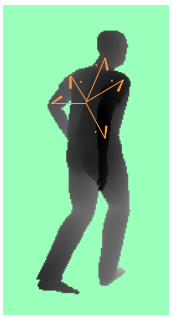


Figure 3-6: Différents "offsets" sur la position des articulations à partir d'un pixel du dos.

pixels ayant atteint la feuille l de l'arbre t est noté $Q_{t,l}$. Nous avons donc :

$$p_{t,l}(c) = \frac{|\{x \in Q_{t,l} | C(x) = c\}|}{|Q_{t,l}|} \tag{3.10}$$

Modèle de régression

Le modèle de régression permet de prédire directement la position des articulations $j \in \{1, \ldots, J\}$ dans l'espace $\mathbb{R}^3$. Contrairement à la classification, cette approche ne nécessite pas d'étape intermédiaire pour passer de la prédiction fournie par la forêt à une prédiction de position des articulations. A chaque feuille l est alors associé un ensemble de positions des membres (appelées "offsets"), relativement à la position des pixels qui sont arrivés à cette feuille. Chaque pixel peut alors potentiellement fournir une prédiction pour toutes les articulations du corps humain. Les offsets sont représentés par des flèches oranges sur la figure 3-6. Après apprentissage d'un arbre, les distributions empiriques d'offsets observées aux feuilles apparaissent multi-modales, en particulier pour les ambiguïtés entre membre droit et membre gauche. A noter qu'il est impossible en pratique de stocker tous les votes de la base d'apprentissage afin de les utiliser lors de la phase de prédiction. La distribution empirique est alors modélisée par un ensemble "relativement petit" de positions relatives $\Delta_{t,l,j,k} \in \mathbb{R}^3$ où $j \in \{1, \ldots, J\}$ désigne le membre considéré et où $k \in \{1, \ldots, K\}$ désigne l'indice du mode. Pour $K = 2$ on modélise ainsi l'ambiguïté droite/gauche du corps humain. Girshick et al. [27] mentionnent que $K > 1$ n'augmente que marginalement les performances mais son implémentation souffre moins de ces ambiguïtés car les postures d'apprentissage utilisées ne représentent pas une rotation complète du corps humain. Chaque vote est associé à un poids $w_{t,l,j,k}$ calculé à partir de la taille du mode.

De la même manière que l'on crée le modèle de classification, notre implémentation du

modèle de régression utilise une collection Q de pixels. Ces pixels traversent chaque arbre, et, pour chaque feuille, nous obtenons une collection $Q_{t,l}$ des pixels arrivés dans cette feuille. A chaque pixel x de la collection $Q_{t,l}$ utilisée pour l'apprentissage, est associé un ensemble $P_i = (p^i_{1,..,}p^i_n)$, où $p^i_{1,..,}p^i_n$ représente la position 3D de chacune des articulations pour l'image de la base d'apprentissage d'où provient le pixel. La position relative (offset) de chaque membre p^i_j de la pose P_i par rapport à la position dans l'espace du pixel x est notée $\delta^i_j = p^i_j - x_k$. Nous avons alors pour chaque membre et à chaque feuille l'ensemble $O_{t,l,j} \subset \mathbb{R}^3$ des offsets dans l'espace pour le membre j à la feuille l. Dans ce cas, nous devons estimer la position des $\Delta_{t,l,j,k}$, les modes de la distribution empirique de l'ensemble $O_{t,l,j}$.

Pour créer la distribution nous utilisons l'algorithme du Mean-Shift avec l'estimateur de densité de Parzen :

$$p(\hat{\Delta}) \propto \sum_{\delta \in O_{t,l,j}} \exp(-||\frac{\delta - \hat{\Delta}}{\sigma}||^2) \tag{3.11}$$

Nous avons alors une collection de points qui correspondent aux modes de la distribution de $O_{t,l,j}$. Cependant, des points d'initialisation différents peuvent être situés sur le même mode et le Mean-Shift fournira alors deux points très proches l'un de l'autre. Deux modes sont considérés comme équivalents si leur distance dans l'espace est inférieure à σ. Seuls K modes sont stockés à chaque feuille pour chaque articulation. Les poids $w_{t,l,j,k}$ de tous les modes trouvés sont calculés comme étant le nombre de points de $O_{t,l,j}$ se trouvant à un distance inférieure à σ de la position du mode.

Le nombre de prédictions fournies par la forêt peut être relativement élevé et de ce fait certaines peuvent être redondantes. Nous pouvons choisir aléatoirement un nombre $N_r = 200$ de prédictions de l'ensemble $O_{t,l,j}$ et effectuer la clusterisation sur cet ensemble réduit. On observe que la réduction de ce nombre de pixels n'affecte pas la performance de l'algorithme et permet de surcroît d'augmenter significativement la vitesse d'exécution.

La figure 3-7 représente un ensemble $O_{t,l,j}$ pour le poignet gauche. Apparaît également sur cette figure la position des deux modes prépondérants de la distribution donnés par notre algorithme. La feuille représentée possède le poignet gauche comme classe c majoritaire après apprentissage. La plupart des pixels arrivés à cette feuille se trouvent donc proches de l'articulation représentée.

Cette figure montre que d'autres modes sont présents dans la distribution mais notre

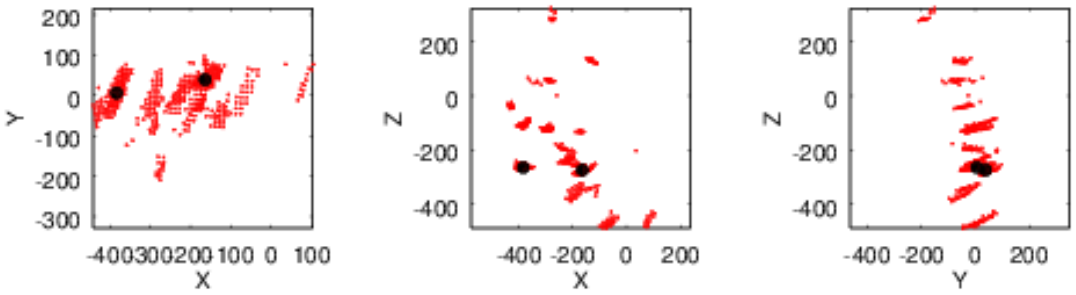


Figure 3-7: Exemple de clusterisation de la position 3D d'une articulation dans une feuille. En noir la position des deux plus grands modes de la distribution.

choix est de ne pas en tenir compte. Ceux-ci sont considérés comme étant des "outliers".

3.3.4 Prédictions finales de la position des articulations

Nous présentons ici notre implémentation de l'état de l'art pour la prédiction finale de la position des articulations sur une image grâce aux offsets contenus dans les feuilles. Les articulations dont nous cherchons à estimer la position sont présentées sur la figure 3-8 : tête, cou, épaule gauche, épaule droite, torse, coude gauche, coude droit, hanche gauche, hanche droite, genou gauche, genou droit, cheville gauche, cheville droite.

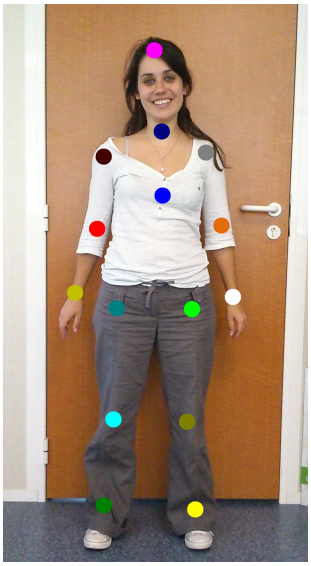


Figure 3-8: Position des articulations à estimer pour notre approche.

Tous les pixels de la personne traversent les arbres et fournissent une prédiction pour chaque membre. Cette prédiction est calculée comme étant la somme de la position absolue

du pixel x_i dans l'espace et de la position relative $\Delta_{t,l,j,k}$ donnée par la feuille à laquelle est arrivé ce pixel.

$$\hat{p}_{i,t,l,j,k} = \Delta_{t,l,j,k} + x_i \tag{3.12}$$

Le poids de chaque prédiction $\hat{p}_{i,t,l,j,k}$ est alors $w_{t,l,j,k}$. Pour l'articulation j, nous avons un ensemble E_j de prédictions de sa position. Les prédictions $\Delta_{t,l,j,k}$ dont la norme est trop élevée, c'est à dire les prédictions fournies par des pixels loin du membre sont potentiellement mauvaises. Pour éviter cela, nous utilisons les distances maximales fournies dans Girshick et al. [27] pour éliminer ces prédictions.

Le nombre de prédictions reste très important du fait des nombreux pixels fournissant une prédiction pour chaque articulation. Cela rend cette information inutilisable en pratique. Il faut donc réduire le nombre de prédictions pour chaque membre. Nous agrégeons les prédictions fournies par tous les pixels pour fournir un nombre relativement faible de prédictions pour chaque articulation. Les prédictions seront ensuite utilisées dans un filtre multi-modal présenté dans le chapitre suivant. De plus nous prenons en compte l'ambiguïté droite/gauche qui apparaît pour les membres en proposant plusieurs positions pour le membre.

La figure 3-9 montre un exemple de prédictions données par les pixels pour la position du poignet gauche. Nous voyons alors que plusieurs modes sont présents dans le nuage de points. Il apparaît alors important de fournir plusieurs prédictions pour la position de chaque articulation. Le choix de la prédiction finale se fera ensuite grâce à un algorithme de filtrage décrit dans le chapitre suivant.

Pour trouver les modes dans l'ensemble E_j, nous utilisons encore une fois un algorithme du Mean-Shift avec l'estimateur de Parzen défini par l'équation 3.13. Pour simplifier les notations, la position d'un point de prédiction sera notée $\hat{p}_i$ et le poids associé sera noté w_i. La position du mode est notée p_j. Le paramètre σ_j est la bande passante de test associée à l'articulation j. Sa valeur est un paramètre de l'algorithme dont la valeur est déterminée dans la section 3.5.

$$p(p_j) \propto \sum_{i \in |E_{j|}} w_i \exp(-||\frac{\hat{p}_i - p_j}{\sigma_j}||^2) \tag{3.13}$$

Le point de départ du Mean-Shift est choisi aléatoirement parmi les points. Chaque point $\hat{p}^i$ a une chance d'être pris comme point de départ avec une probabilité $\frac{w_i}{\sum_j w_j}$.

Le poids de chaque mode est défini comme étant la valeur de l'estimateur de Parzen en ce point. De même que lors de la clusterisation des feuilles, nous regroupons les modes trop proches les uns des autres. Nous avons alors un ensemble de prédictions pour la position du membre j. Le nombre de prédictions dans cet ensemble est variable. A chaque mode est associé un poids $\hat{w}^i$ qui correspond à la somme des poids des points situés à moins de σ_j de la position du mode. Les modes ayant des poids trop faibles (inférieurs à S_j) sont éliminés de l'ensemble des prédictions. La valeur de ce seuil est un paramètre dont la valeur doit être déterminée. Celle-ci n'est pas donnée dans les travaux précédents. Dans la section 3.5, nous présentons les performances de l'algorithme en fonction de ce seuil.

La figure 3-9 montre un nuage de points de prédictions donné par une image de test pour le poignet gauche avec en rouge les quatre plus grands modes trouvés par le Mean-Shift et en vert la position réelle du membre.

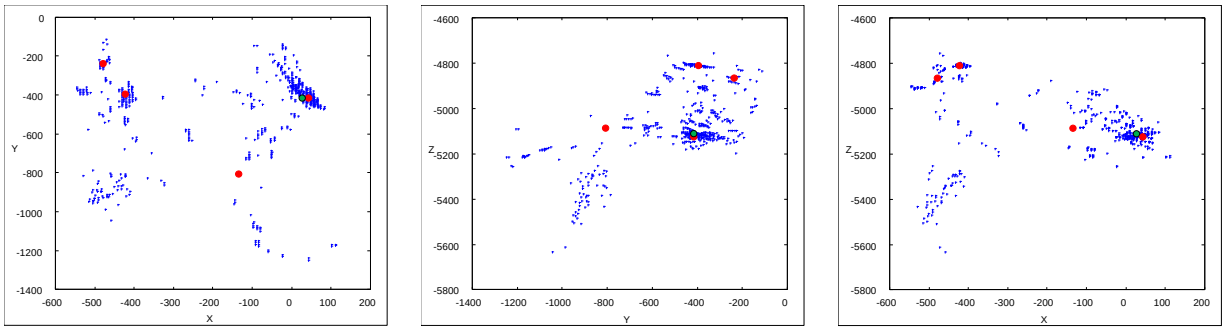


Figure 3-9: Nuage de prédictions pour le poignet gauche. En bleu le nuage de prédiction, en rouge les modes de la distributions, en vert la position réelle du membre.

La figure 3-10 illustre les positions des plus grands modes pour une image simulée et une image réelle. Les couleurs des points représentants les positions des articulations correspondent à celles illustrées sur la figure 3-8.

3.4 Notre Approche

Nous cherchons à estimer la position de 15 articulations du corps de la personne illustrées sur la figure 3-8. Notre méthode est une extension de Girshick et al. [27]. Nous allons donc utiliser des Random Forests en apprentissage supervisé et leur faire apprendre les postures pour ensuite les reconnaître dans nos scénarios. Soulignons que l'information stockée dans les feuilles de nos Randoms Forest donne directement la position des articulations du corps par rapport à la position d'un pixel et correspond à une approche de régression.

a) Image de synthèse. b) Image réelle.

Figure 3-10: Prédiction de la position des membres pour deux images.

Les membres dont nous choisissons d'estimer la position ne sont pas les mêmes que dans Shotton et al. [52]. En effet, leur application de jeu demande de posséder une grande granularité d'estimation de la position des membres terminaux. Ils estiment la position des mains et des poignets, ce qui n'est pas notre cas. Ils ne font pas non plus l'estimation de la position de la poitrine. Dans notre application nous sommes davantage intéressés par la position générale du corps alors que nous n'avons pas besoin d'une grande granularité sur la position des membres terminaux pour avoir une idée générale de la dynamique de la chute. C'est pourquoi la selection des articulations que nous cherchons à estimer est différente.

Pour l'apprentissage des Random Forests nous allons utiliser des données d'apprentissage synthétiques à l'instar de Shotton et al. dans [52]. Ces données permettent d'avoir une grande quantité et variété de données (morphologie humaine, apparence vestimentaire,...) plus simplement qu'avec des données d'apprentissage réelles. Notre système de motion capture permettra d'enregistrer les poses que nous voulons faire apprendre pour ensuite animer des personnages avec ces poses et produire des images de synthèse. L'apprentissage s'effectuera alors avec une base de données d'images de synthèse.

3.4.1 Nos contributions

Notre ré-implémentation de Girschik, adaptée à notre contexte, met l'accent sur deux volets.

Tout d'abord nous montrons que l'équilibrage par classe de la base d'apprentissage

est un facteur de performance. L'équilibrage de la base d'apprentissage est un problème classique des algorithmes de "machine learning" et nous voulons en évaluer l'impact dans notre cas. Nous proposons alors deux façons de mettre en oeuvre cet équilibrage.

Nous montrons ensuite comment l'introduction de la hauteur du pixel en tant que nouvelle caractéristique pour l'apprentissage accroît les performances de l'algorithme.

Les sections suivantes reprennent chacun de ces volets.

3.4.2 Equilibrage de la collection de pixels d'apprentissage

Pour réduire le temps d'apprentissage, tous les pixels de chaque image ne sont pas utilisés, ceux-ci sont choisis aléatoirement parmi les pixels de la silhouette de l'image. Lorsque l'on choisit ces pixels de façon uniforme, on obtient une distribution des pixels très déséquilibrée comme on peut la voir sur la figure 3-11. Cela est dû à la proportion des parties corporelles dans les images d'apprentissage. Le déséquilibre de la base donne un apprentissage biaisé en faveur des classes qui possèdent un plus grand nombre de pixels que les autres. Pour remédier à ce problème, nous proposons deux façons d'équilibrer cette base d'apprentissage par prétraitement des données.

Equilibrage du nombre de pixels

La plupart des travaux cherchant à faire apprendre des ensembles de données déséquilibrés à des Random Forests se penchent exclusivement sur un problème à deux classes. Citons Chen et al. dans [14] et Thomas et al. dans [59]. Pour une personne, le nombre de pixels de chaque classe/partie corporelle c est disparate. La figure 3-11 montre le nombre de pixels de chaque classe pour une seule image. On voit en effet la grande disparité entre les valeurs.

De manière générale, lorsqu'on utilise des algorithmes d'apprentissage, il est préférable d'avoir une base d'apprentissage équilibrée où le nombre d'instances de chaque classe est proche. Pour équilibrer cette base nous prenons le même nombre de pixels de chaque classe dans chaque image d'apprentissage et nous choisissons ces pixels aléatoirement de façon uniforme sur la surface de la classe. De fait, nous équilibrons le nombre de pixels de chaque classe pris pour chaque pose, ce qui permet d'équilibrer la base d'apprentissage en général. A chaque image, nous souhaitons prendre aléatoirement un nombre N_p de pixels de cette image. Ainsi, nous choisissons chaque pixel indépendamment avec une probabilité qui dépend de sa classe de telle sorte que l'espérance du nombre de pixels total pris dans

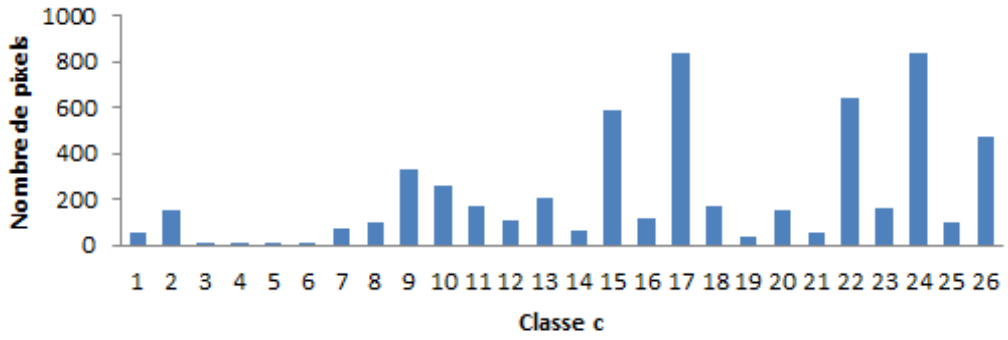


Figure 3-11: Distribution du nombre de pixels par classe sur une image.

cette image soit N_p. $P_I(c)$ désigne la probabilité qu'un pixel de l'image I de la classe c soit utilisé pour l'apprentissage et $N_I(c)$ désigne le nombre de pixels de la classe c dans l'image I. Nous avons alors l'égalité 3.14 :

$$\sum_{c \in C} N_I(c) P_I(c) = N_p \tag{3.14}$$

Par ailleurs, il nous faut prendre en compte la contrainte qui impose que le nombre de pixels de chaque classe doit être le même.

$$\forall (c_1, c_2) \in C^2, N_I(c_1) P_I(c_1) = N_I(c_2) P_I(c_2) \tag{3.15}$$

La résolution du système nous donne :

$$\forall c \in C, P_I(c) = \frac{N_p}{N_I(c)|C|} \tag{3.16}$$

Dans certains cas, le calcul de cette probabilité donne une valeur supérieure à 1 pour certaines classes. Nous choisissons de saturer cette valeur à 1 sans modifier les autres valeurs, ce qui modifie l'espérance du nombre de pixels pris au total et pour chaque classe. En pratique cela varie peu et le nombre de pixels pris par image reste proche de N_p dans la grande majorité des cas.

La figure 3-12 montre la distribution de pixels résultante de notre algorithme. Nous voyons que l'équilibrage par cette méthode permet de s'affranchir du déséquilibre initial de la base. Celle-ci reste cependant légèrement déséquilibrée. Nous présentons dans la section suivante une façon de gérer ce déséquilibre par pondération des classes lors de

l'apprentissage.

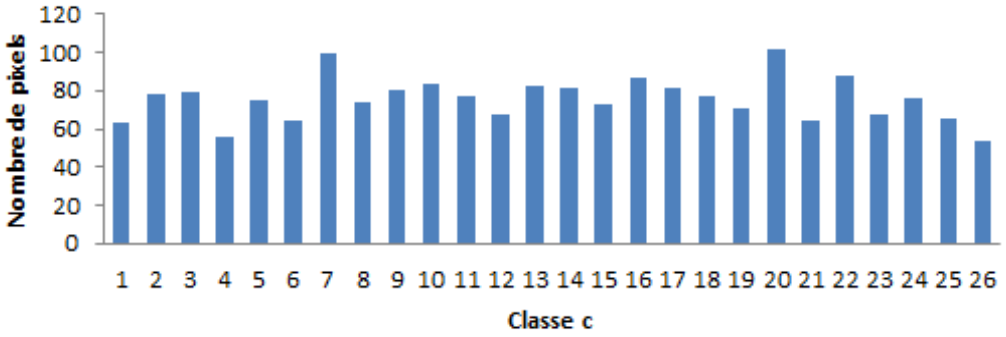


Figure 3-12: Distribution équilibrée du nombre de pixels par classe sur une image.

Pondération des classes

Une autre approche peut être utilisée moyennant une légère modification du critère d'apprentissage. Cette approche peut être mise en pratique de concert avec l'approche précédente et permet de gérer la situation $P_I(c) > 1$. Nous utilisons pour cela la distribution empirique des classes des pixels de la collection Q_t et nous calculons $P(c) = \frac{|\{x \in Q \backslash classe\ de\ x=c\}|}{|Q|}$ qui représente la probabilité a priori de la classe d'un pixel. On affecte alors un poids à chaque classe tel que $w_c = \frac{1}{P_Q(c)}$. Le critère 3.7 se réécrit :

$$H_2 = -\sum_{c \in C} \frac{w_c P_Q(c)}{\sum_{c \in C} w_c P_Q(c)} \log\left(\frac{w_c P_Q(c)}{\sum_{c \in C} w_c P_Q(c)}\right) \tag{3.17}$$

Dans le cas d'une collection déséquilibrée, l'utilisation du critère simple de l'équation 3.7 introduit des biais lors du choix du paramètre ϕ lors de l'apprentissage en faveur d'une meilleure classification des classes majoritaires au détriment des classes minoritaires. Ce nouveau critère permet alors une meilleure prise en compte des classes minoritaires qui sont pour nous par exemple les poignets ou les pieds. Nos évaluations décrites en section 3.5, démontrent une meilleure estimation de la position de ces articulations avec une collection d'apprentissage équilibrée.

3.4.3 Hauteur du pixel comme nouvelle caractéristique

Dans notre application, la position du capteur permet de calculer facilement la hauteur d'un pixel par rapport au sol. Cette information est d'une nature différente de celle de

Shotton car elle est dépendante de l'environnement. Nous avons alors une information absolue et non relative par rapport aux autres pixels. De plus, il semble naturel d'utiliser la hauteur lorsqu'on cherche à estimer des postures de chute.

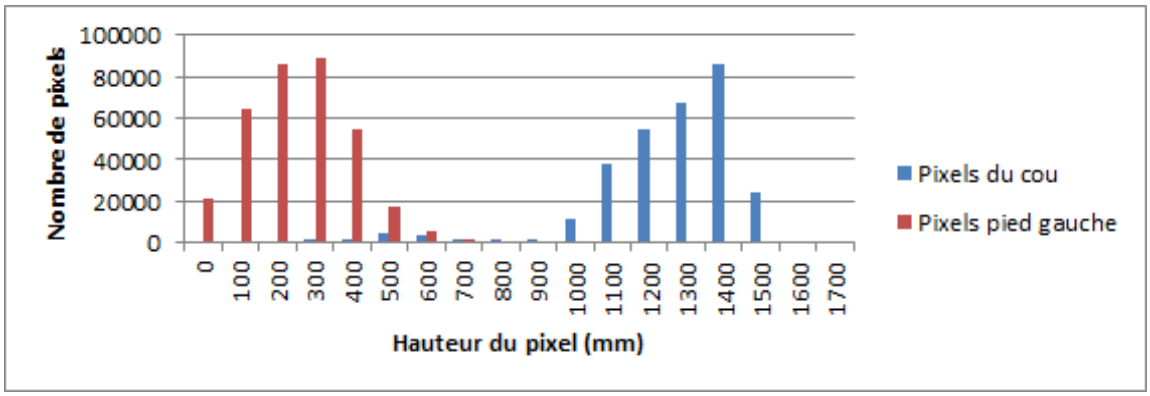


Figure 3-13: Hauteur des pixels de deux classes différentes pour la base d'apprentissage.

Si l'on visualise les distributions empiriques des hauteurs des pixels de deux classes différentes pour toute la base d'apprentissage comme sur la figure 3-13, on remarque que cette information permet de très facilement séparer ces deux classes. On s'attend à ce que l'utilisation de cette information soit faite dans les noeuds de l'arbre. On comprend en effet qu'avec la hauteur, on ne puisse pas distinguer finement la nature du pixel, ce qui est le travail des noeuds les plus profonds.

Formalisation

Nous proposons alors de combiner les caractéristiques de Shotton et l'information de hauteur dans une caractéristique hétérogène présentée en 3.18 :

$$f_{\theta,2}(I,x) = \begin{cases} d_I(x + \frac{u}{d_I(x)}) - d_I(x + \frac{v}{d_I(x)}), & \text{si } p = 1 \\ h(x), & \text{si } p = 2 \end{cases} \tag{3.18}$$

Les paramètres de la caractéristique sont alors $\theta = (u, v, p)$. Le paramètre p code la nature de la caractéristique. Pour la valeur $p = 1$, nous retrouvons le calcul classique de caractéristique utilisé par Shotton. Pour une valeur $p = 2$, la caractéristique fournit alors la hauteur $h(x)$ du pixel x. Lors de l'apprentissage, les deux valeurs de p seront testées à chaque noeud. L'algorithme va alors choisir de prendre la hauteur comme caractéristique la plus pertinente pour la classification ou prendre la caractéristique de Shotton et cela

sans a priori et à chaque noeud. Si cette information est utile pour l'apprentissage, nous allons alors avoir des noeuds pour lesquels le meilleur gain d'information correspond à une valeur $p = 2$.

3.4.4 Génération des données d'apprentissage et de la base de test

Les Random Forests sont des algorithmes d'apprentissage supervisés et demandent donc d'avoir un ensemble d'images d'entraînement avec vérité terrain. Il est difficile d'avoir une vérité terrain avec des images d'apprentissage réelles et la production de celles-ci est fastidieuse et coûteuse. Nous avons donc privilégié les images d'apprentissage de synthèse. Nous allons donc décrire dans cette section le processus de génération de ces images.

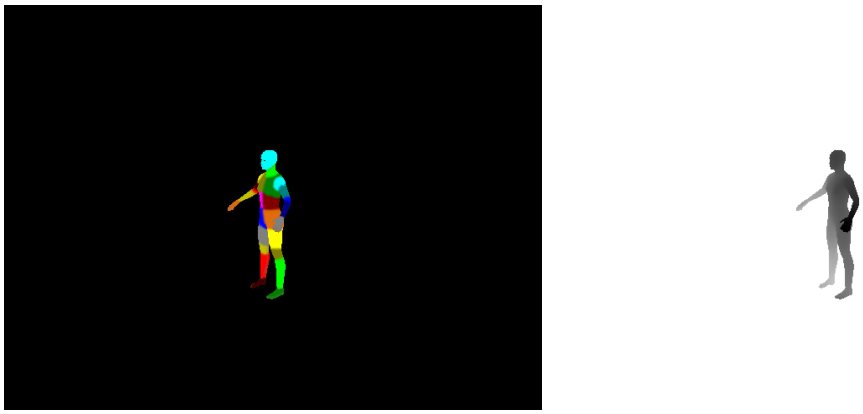
Ce processus est inspiré de Shotton et al. [51]. Une base de données de mouvements tirés d'un système de capture de mouvement est utilisé en conjonction de modèles humains de différentes morphologies et habillés de différentes façons. La difficulté de création de cette base de données d'images vient du grand nombre de variations possibles à la fois dans les mouvements, les morphologies, les vêtements et la position de la caméra.

Comme point de départ pour le développement de nos algorithmes nous avons utilisé la CMU Mocap Library [2]. Cette base de données de capture de mouvement possède une grande variété de mouvements différents. Cependant elle contient peu d'images de chutes. Nous avons alors complété cette base de données avec différents mouvements de chute. D'autres types de mouvements ne sont pas présents dans cette base mais notre application étant focalisée sur la détection de chute et nos moyens techniques et humains étant limités, nous nous sommes concentrés sur la capture de chutes. Dans cette partie nous explicitons d'abord la nature de la base d'apprentissage puis nous détaillons les différentes étapes permettant la génération de cette base de données.

Nature de la base d'apprentissage

La base d'images d'apprentissage est en fait composée à la fois d'images et de métadonnées associées. Une "unité image" comprend :

- l'image de labels où chaque pixel correspond au fond ou à une partie du corps c illustrée sur la figure 3-14a.
- l'image de profondeur illustrée sur la figure 3-14b.
- la position dans l'espace 3D des articulations.
- les paramètres caméra.



a) Image des parties corporelles. b) Image de profondeur.

Figure 3-14: Exemple d'unité image de la base d'apprentissage.

Capture de mouvements et génération de la base de test

Dix différents mouvements de chute ont été enregistrés ainsi que d'autres mouvements communs tels que la marche. Ces captures ont, de plus, été filmées avec 5 caméras Xtion en utilisant le montage présenté sur la figure 3-15 ce qui nous a permis par ailleurs de créer une base de tests avec des images réelles sur les mouvements que nous cherchons à estimer. La position des caméras est présentée sur la figure 3-16. Evidemment, il est impossible en pratique d'avoir une base contenant la totalité des mouvements que l'on cherche à estimer. Cependant, nous espérons que la capacité de généralisation de l'algorithme d'apprentissage permettra de reconnaitre des poses qui ne sont pas dans la base d'apprentissage. Il est important toutefois que chaque membre indépendamment possède un grand nombre de poses différentes et c'est le caractère semi-local de l'apprentissage qui permettra la généralisation.

Elimination des poses redondantes

Le caractère temporel des captures de mouvement n'est pas pris en compte dans ce chapitre et l'on considère celles-ci comme simplement un ensemble de poses non corrélées. Cette cohérence spatio-temporelle implique cependant que deux poses consécutives d'une capture de mouvement seront très similaires et donc probablement redondantes pour l'apprentissage. Afin de créer une base de données de motion capture représentant le nombre maximum de poses tout en restant relativement petite, nous utilisons un algorithme de clusterisation que nous appliquons sur l'ensemble des poses dont nous disposons dans

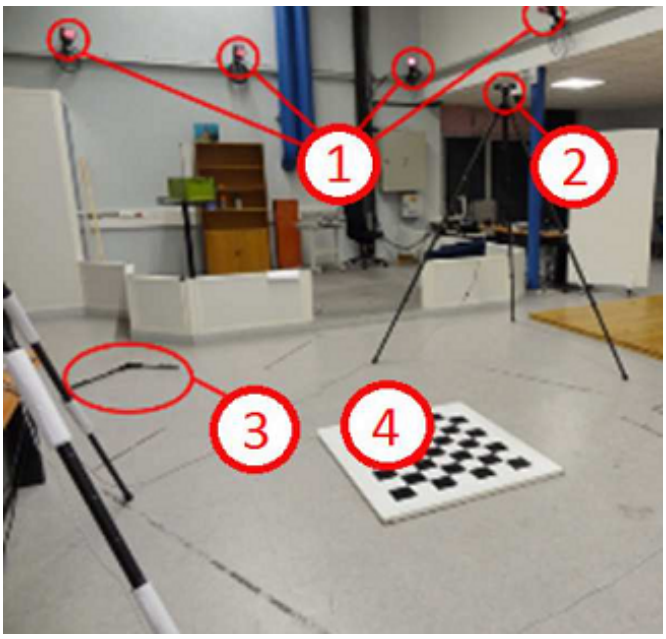


Figure 3-15: Montage de capture de mouvement. 1.Caméras MoCap 2.Capteur RGB-D 3. Mire de calibration MoCap 4. Mire de calibration RGB.

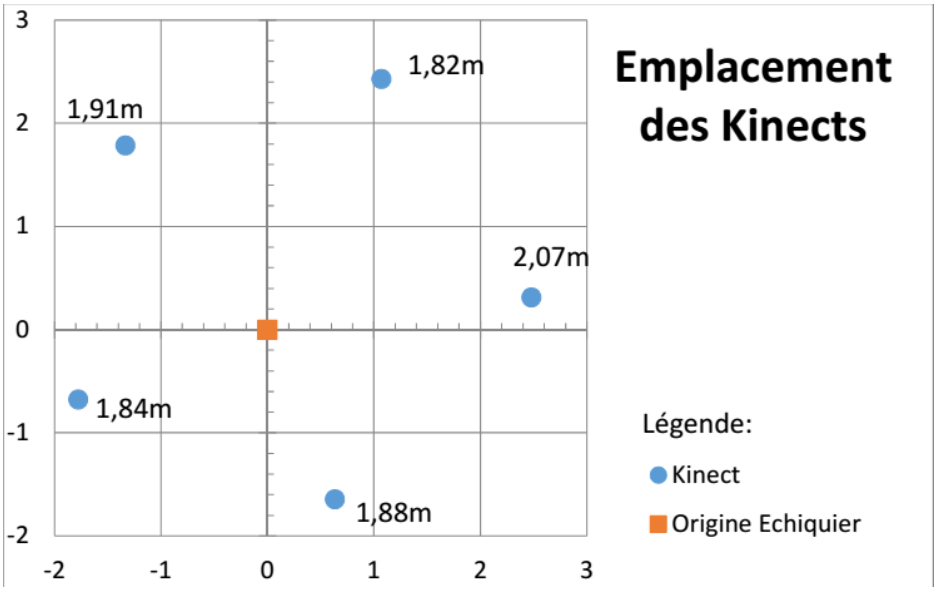


Figure 3-16: Position des capteurs RGB-D avec leurs hauteurs associées.

notre base de données. Cet algorithme permet de choisir un sous-ensemble de poses qui sera utilisé pour animer les modèles en éliminant les postures considérées comme redondantes. Pour cela nous utilisons l'algorithme de Furthest Neighbour. Chaque pose est représentée comme une collection de positions d'articulation $P = (p_{1,..,}p_n)$. Nous choisissons une pose initiale P comme premier élément d'un ensemble $\wp$ qui sera l'ensemble des poses utilisées pour l'apprentissage. Cet ensemble est ensuite augmenté à chaque nouvelle itération grâce à l'équation 3.19. L'ensemble total des poses dont nous disposons est noté $\wp_{all}$.

$$\wp := \wp \cup \{\underset{P \in \wp_{all} \setminus \wp}{\operatorname{argmax}} \min_{P' \in \wp} d_{pose}(P, P')\} \tag{3.19}$$

La distance entre deux poses est définie par 3.20

$$d_{pose}(P, P') = \max_{j \in \{1,..,j\}} ||p_j - p'_j||_2 \tag{3.20}$$

L'algorithme se termine lorsque qu'il n'existe plus de pose P telle que $d_{pose}(P, P') > D_{pose}$. L'ensemble des poses $\wp$ est alors l'ensemble d'apprentissage retenu pour la forêt.

Modèles et animation

Afin de créer différents modèles de corps humain pour différentes corpulences et vêtements, nous utilisons le logiciel MakeHuman [5]. Ce logiciel permet de créer différentes morphologies et permet d'habiller les modèles de façon variée. Ces modèles sont ensuite exportés sous le logiciel Blender [1] où ils seront animés avec la base de données de poses que nous avons créée. Les différentes unités images sont alors exportées. Dix modèles différents avec des paramètres de hauteur et de corpulence sont utilisés pour créer la base de données. Des exemples de ces modèles sont donnés sur la figure 3-17 avec une sélection des vêtements utilisés pour l'apprentissage. Nous utilisons un pantalon, un teeshirt à manches longues et un teeshirt à manches courtes.

La figure 3-18 montre une sélection des postures utilisées pour l'apprentissage.

Labellisation des données

Les labels des classes c de chaque pixel correspondent à la partie du corps dont ils font partie. Afin de pouvoir générer l'image correspondante pour chaque unité image (image 3-14a), nous utilisons des textures de notre fabrication qui définissent les 26 différentes parties du corps de la personne comme présenté sur la figure 3-2. Contrairement à [52]

Figure 3-17: Exemples de modèles utilisés pour l'apprentissage.

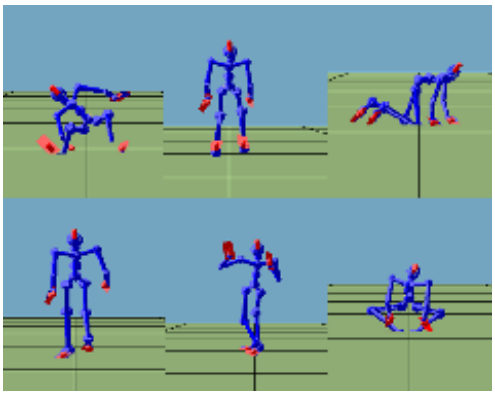


Figure 3-18: Exemples de postures utilisées pour l'apprentissage.

nous n'utilisons qu'une seule classe pour la tête. Il est important que les classes du corps ne soient pas trop étendues afin de limiter la variation d'apparence des pixels d'une même partie du corps et ainsi faciliter leur classification.

Paramètres caméra

Nous cherchons à simuler la position d'un capteur attaché au plafond d'une chambre. La hauteur typique d'un plafond se situe entre $2,20m$ et $2,70m$. L'angle du capteur par rapport au plafond prend des valeurs entre 30° et 60°. La personne est vue de tous les points de vue à 360°. Pour chaque unité image, nous prenons des paramètres aléatoirement et de façon uniforme dans ces plages.

3.5 Expérimentations et évaluations associées

Dans cette partie, nous évaluons les performances de notre algorithme et investiguons l'influence de différents paramètres sur celles-ci. Nous utilisons à la fois des images simulées et des images réelles pour réaliser ces évaluations. Les images synthétiques sont utilisées pour l'évaluation car elles représente un nombre plus important de postures que notre base de test réelle. Nous définissons d'abord la mesure de l'erreur que nous utilisons et détaillons les conditions d'apprentissage de la forêt que nous avons utilisées. Puis nous évaluons les gains de performances réalisés avec l'ajout d'un nouveau type de caractéristique ainsi que l'influence de l'équilibrage de la base d'apprentissage. L'influence du paramètre de bande passante de test σ_j est également étudié. Enfin, nous évaluons les performances de notre algorithme sur notre base de test d'images réelles avant de nous comparer à l'état de l'art.

3.5.1 Mesure de l'erreur de l'estimation de pose

Nous utilisons la même mesure d'erreur que celle présentée dans Shotton et al. [52]. Pour toutes les images de test et pour chaque articulation, nous gardons tous les modes dont le poids $\hat{w}^i$ se situe au-dessus d'un seuil S. La première prédiction située à une distance inférieure à une distance D_{tp} est considérée comme une bonne mesure. Toute autre prédiction est considérée comme un faux positif. Une articulation qui ne possède aucune prédiction avec un poids plus grand que le seuil S est considérée comme un faux négatif. En faisant varier le seuil, nous avons alors une courbe precision-recall complète. La précision et le recall sont définis ci-après :

$$precision = \frac{Nombre\ bonnes\ mesures}{Nombre\ bonnes\ mesures\ +\ Nombre\ de\ faux\ positifs} \tag{3.21a}$$

$$recall = \frac{Nombre\ bonnes\ mesures}{Nombre\ bonnes\ mesures\ +\ Nombre\ de\ faux\ négatifs} \tag{3.21b}$$

La valeur de la distance D_{tp} utilisée est $10cm$. Un point de prédiction est alors considéré comme valable s'il se situe à moins de $10cm$ de la position réelle du membre. L'aire sous la courbe precision-recall nous donne la valeur de la précision moyenne. Celle-ci peut être calculée par articulation (Pm, j) ou en général (Pmg) pour toutes les articulations. Cette

mesure permet de prendre en compte le problème qui se pose lorsque trop de prédictions sont présentes pour un membre ce qui impacterait le coût d'un algorithme de filtrage qui viendrait ensuite.

3.5.2 Conditions d'apprentissage de la forêt

Dans la plupart des évaluations menées dans cette section, trois arbres ont été entraînés avec chacun 5000 images et 2000 pixels par image à une profondeur de 20. Chaque noeud a été entraîné avec un choix parmi 500 caractéristiques $p = 1$ différentes et 20 seuils possibles. De plus un gain minimum d'information de $G_{\max} = 0.1$ a été pris pour qu'un noeud puisse être un noeud de branchement. Ces conditions se situent à la limite des capacités d'un ordinateur du commerce. Le temps d'apprentissage d'une forêt est d'environ une demi-journée et la quantité de mémoire vive nécessaire se situe aux alentours de 10 Go.

3.5.3 Caractéristique de hauteur du pixel

Dans cette partie nous montrons l'influence de notre caractéristique de hauteur du pixel. Nous montrons que celle-ci est bien utilisée lors de l'apprentissage de l'arbre et nous quantifions les gains de performance.

Position des noeuds de hauteur dans l'arbre

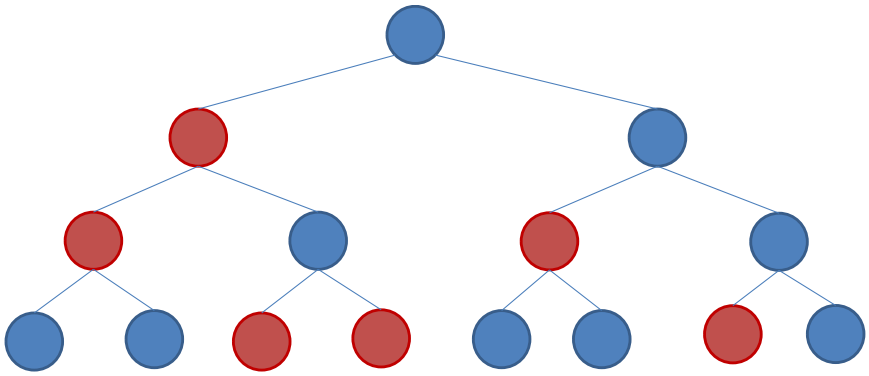
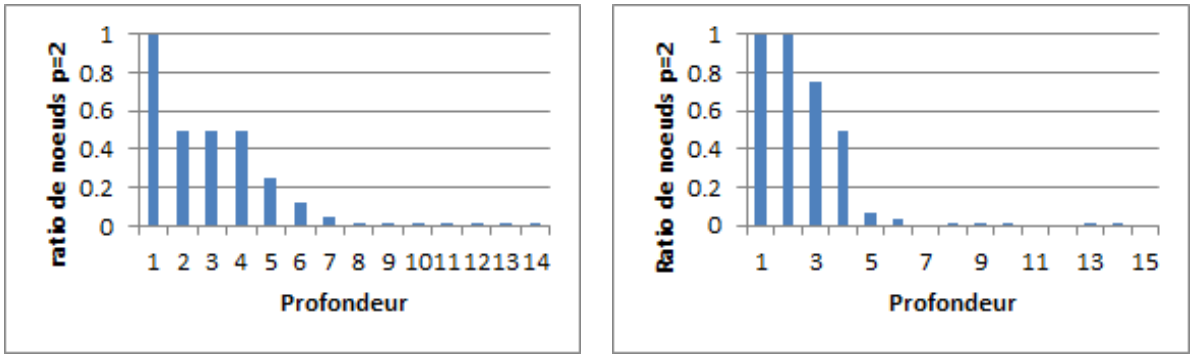


Figure 3-19: 4 premiers niveaux d'un arbre avec caractéristiques de hauteur.

Nous cherchons à déterminer si l'utilisation de la hauteur est utile au regard du contexte de détection de chute. Nous voulons de plus savoir si cette information est parfois plus utile que l'information donnée par des caractéristiques $p = 1$. Deux forêts ont été apprises,

une où les noeuds avaient la possibilité de choisir une caractéristique $p = 2$ et une sans cette possibilité. La figure 3-19 montre les 4 premier niveaux de l'arbre généré avec en bleu les noeuds avec une caractéristique de type $p = 2$ et en rouge les noeuds avec une caractéristique de type $p = 1$. Alors que rien n'est mis en place pour forcer certains noeuds à utiliser la hauteur du pixel comme caractéristique, celle-ci a été choisie pour certains noeuds et particulièrement pour les premiers noeuds. Cela indique la pertinence de cette information de hauteur dans les premières profondeurs de l'arbre. Le graphique 3-20 montre la proportion de noeuds de type $p = 2$ pour chaque profondeur de l'arbre.



a) Sans chutes. b) Avec chutes.

Figure 3-20: Ratio de noeuds avec p=2 en fonction de la profondeur de l'arbre.

Comme espéré, on voit qualitativement que cette information est utile au début de l'apprentissage car l'évaluation de gain d'information lui a été favorable par rapport à $p = 1$ c'est à dire par rapport à la caractéristique $p = 1$ seule. En effet, les pixels situés au-dessus de la taille sont, dans la grande majorité des poses, plus hauts que les pixels situés en dessous. Une fois la hauteur utilisée, on voit que cette information devient moins importante au fur et à mesure que la profondeur augmente et ce sont des caractéristiques avec $p = 1$ qui sont choisies pour les couches les plus profondes de l'arbre.

Un arbre a aussi été appris avec des postures variées mais sans postures de chutes où la personne se trouve au sol. La figure 3-20 représente la proportion de noeuds $p = 2$ pour chaque profondeur. On voit que lorsqu'il n'y a pas de postures de chutes, le nombre de noeuds avec $p = 2$ chute plus vite avec la profondeur mais l'information semble très utile pour les 2 premières couches de l'arbre. On peut supposer que l'information de hauteur est prise en compte différemment dans les deux cas. Dans le cas de l'apprentissage avec chutes, la hauteur est utilisée pour différencier les poses de chutes des autres poses. Dans

le cas de l'apprentissage sans chutes, l'information est utilisée principalement pour séparer les parties du bas du corps des parties du haut du corps.

Distribution des seuils

Il est intéressant de comprendre quels seuils (notés τ) sont choisis par les noeuds lorsque l'information de hauteur est utilisée.

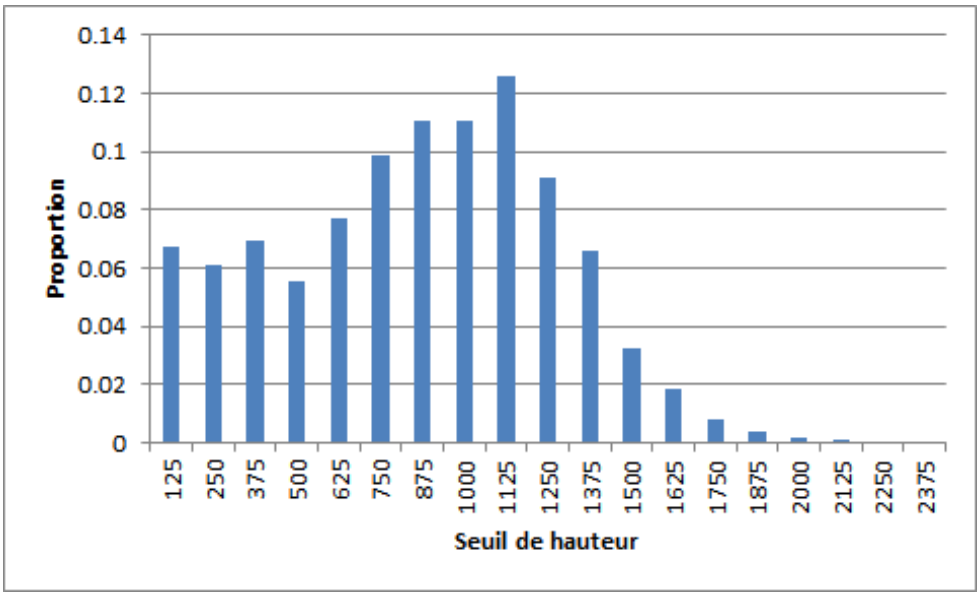


Figure 3-21: Distribution du seuil de hauteur dans les noeuds.

La figure 3-21 montre la distribution empirique des seuils τ choisis pour l'information de hauteur pour tous les noeuds de la forêt. Comme nous pouvions nous y attendre, la majorité des noeuds possède un seuil inférieur à $1m$ de hauteur. Beaucoup de seuils sont choisis autour de cette valeur qui correspond à la fois à une hauteur qui correspond à celle des hanches de la personne debout et une hauteur au-dessus de laquelle les postures de chutes ne possèdent pas de pixels.

Comparaison des performances avec et sans hauteur

Les valeurs de σ_j (bande passante de test pour le membre j) permettant d'obtenir les meilleurs Pmg donnent les performances suivantes présentées sur la figure 3-22.

Nous voyons que l'apport de la hauteur permet une amélioration des performances pour tous les membres. La Pmg pour tous les membres est de 0.756 avec la hauteur et 0.733 sans la hauteur.

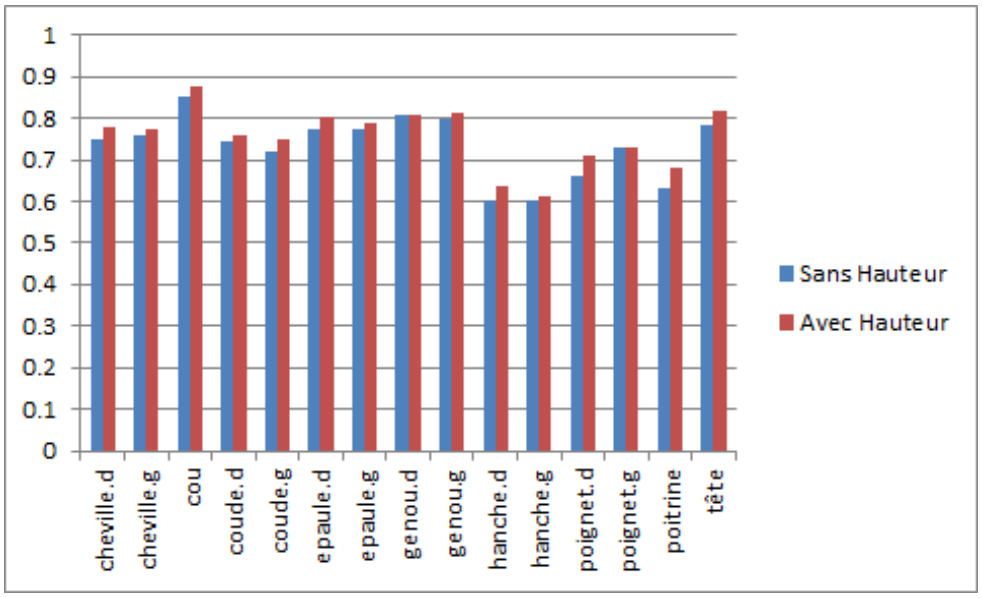


Figure 3-22: Performances avec et sans utilisation de la hauteur.

3.5.4 Influence du paramètre de bande passante de test

Le "tuning" de certains paramètres libres est omis dans les articles de Shotton et al [52]. Notamment, aucune valeur n'est donnée pour les valeurs de σ_j qui sont les valeurs de bande passante des estimateurs de Parzen pour l'agrégation et la formation de propositions lors du test. Ces paramètres sont importants car ils contrôlent notamment le nombre de prédictions finales pour chaque membre et la position de ces prédictions. La valeur de ces paramètres correspond au compromis biais-variance qui est fait lors de l'estimation. Une valeur faible de ce paramètre donnera un grand nombre de prédictions biaisées, alors qu'une valeur élevée de ce paramètre donnera un faible de nombre de prédictions possédant chacune une variance importante. Nous avons alors testé, pour chaque membre sur une base de test de 5000 images de synthèse, différentes valeurs pour les paramètres σ_j.

Nous voyons sur la figure 3-23 que la valeur $\sigma_{t,,j}$ correspondant au maximum de Pmg dépend de l'articulation. Nous voyons aussi que la valeur de ce paramètre influe beaucoup sur les performances de l'algorithme et qu'il est, de fait, important de bien le choisir pour chaque membre.

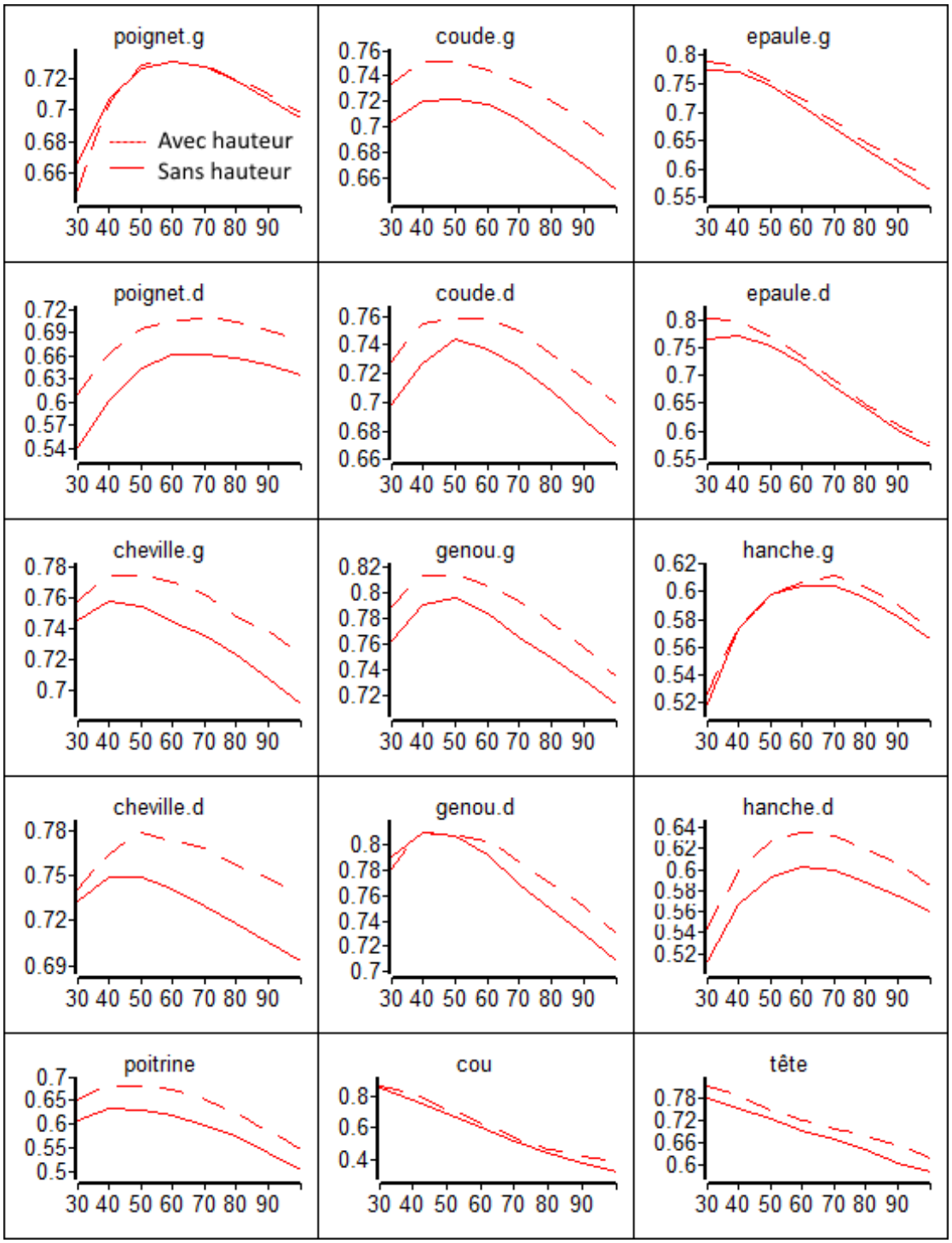


Figure 3-23: Influence de la hauteur sur la bande passante de test. En abscisse la valeur de la bande passante. En ordonnée la valeur de Pmg.

3.5.5 Influence de Dtp sur $\sigma_{t,,j}$.

La valeur de Dtp est importante car elle correspond à la précision que nous requérons pour notre système. Dans les résultats présentés précédemment, cette valeur est de $10cm$. Si nous requérons une autre valeur, la valeur des $\sigma_{t,,j}$ permettant d'atteindre le maximum de performance est susceptible de changer.

La figure 3-24 montre les différences de performance pour deux valeurs différentes de Dtp à savoir $10cm$ et $5cm$. Nous voyons que pour certains membres, la valeur du $\sigma_{t,,j}$ permettant d'obtenir le meilleur score varie en augmentant lorsque l'on choisit $Dtp = 5cm$. En effet la nature de notre critère fait que la diminution de la distance Dtp augmente potentiellement le nombre de prédictions situées plus loin que cette distance.

3.5.6 Influence de l'équilibrage des pixels

Nous avons réalisé un apprentissage en n'équilibrant pas les données comme décrit précédemment. Nous présentons les résultats dans la figure 3-25.

Pour tous les membres nous avons une Pmg de 0.756 pour l'apprentissage équilibré et 0.748 pour l'apprentissage non équilibré. L'influence de l'équilibrage de la collection impacte donc les performances. Certains membres se retrouvent moins bien détectés lorsque l'apprentissage est équilibré. Les membres dont la détection est moins bonne sont ceux situés près des zones de grande taille comme la poitrine ou les hanches par exemple. En effet, moins de pixels de ces zones sont utilisés lorsque l'on équilibre la collection et cela diminue la capacité de détection de ces articulations. En revanche l'équilibrage tend à améliorer les performances pour les membres situés près des classes de petite taille.

Valeur du seuil de confiance

Il est important de choisir le meilleur seuil S_j pour chaque membre selon le compromis precision-recall que nous voulons. Pour rappel, les predictions dont le poids associé est inférieur à ce seuil sont éliminées. Un seuil trop faible donnera un grand nombre de prédictions potentiellement éloignées de la position réelle du membre alors qu'un seuil trop haut éliminera peut-être la prédiction la plus proche. Nous cherchons le seuil S_j qui maximise la mesure F_1 définie par :

$$F_1 = 2\frac{precision * recall}{precision + recall} \tag{3.22}$$

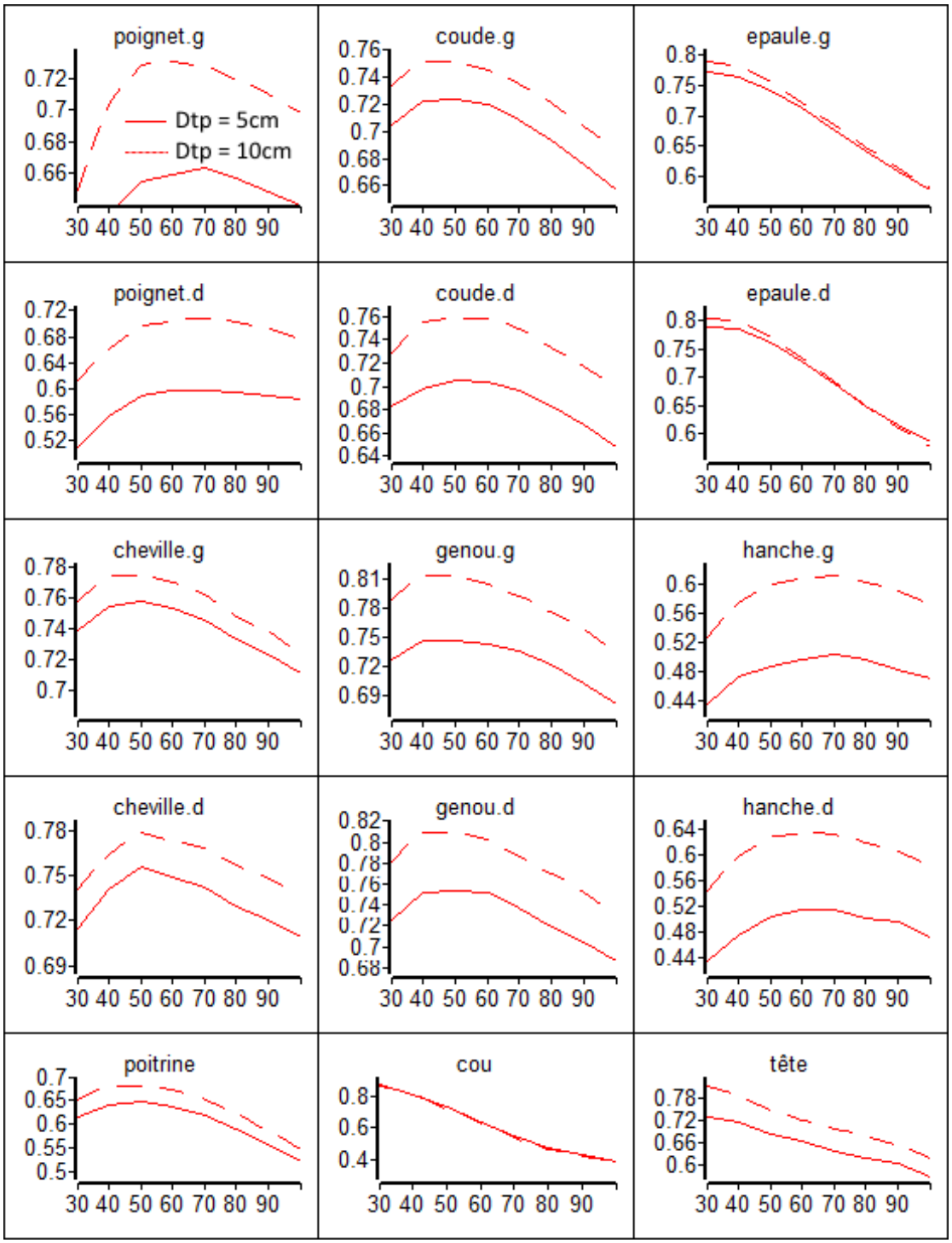


Figure 3-24: Influence de la distance de détection (Dtp) sur la valeur de bande passante de test optimale. En abscisse la valeur de bande passante. En ordonnée la valeur de Pmg.

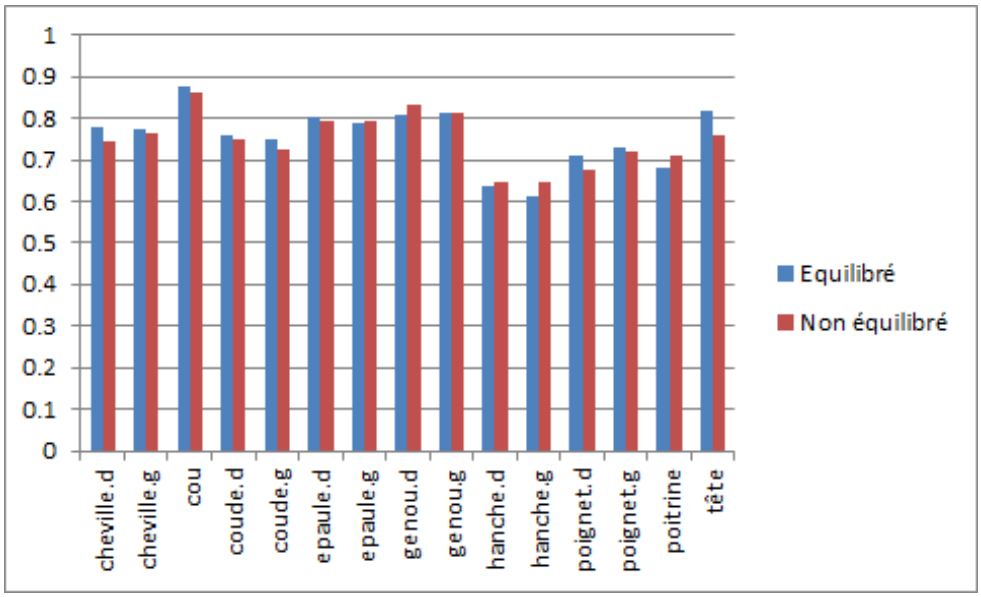


Figure 3-25: Performance avec équilibrage de la base d'apprentissage

3.5.7 Evaluations images réelles

Nous avons créé une base de test avec des images réelles et notre système de motion capture commercial. Nous utilisons 10000 images prises avec 5 capteurs disposés autour du sujet en mouvement comme présenté précédemment. Nous avons donc 2000 images pour chaque capteur. Chacun des capteurs observe le sujet d'un angle différent. Les mouvements observés sont variés et continus dans le temps. Différents mouvements de chutes sont présents dans la base. Nous prenons alors 2000 images aléatoirement dans cette base pour effectuer nos évaluations. La figure 3-26 présente la Pmg par membre pour les images réelles.

Pour l'ensemble des membres nous avons une Pmg de 0.591. Les performances moins bonnes sur la base réelle proviennent d'une grande augmentation de l'ambiguïté droite/gauche par rapport à la base d'images de synthèse. De plus, les images produites par notre installation expérimentale sont de moins bonne qualité que des images produites avec un seul capteur du fait des interférences existantes entre ces capteurs. Est représenté sur la figure 3-27 l'ensemble des prédictions retenues pour le poignet droit pour différentes images. Un mode est toujours présent sur le poignet droit mais d'autres modes apparaissent.

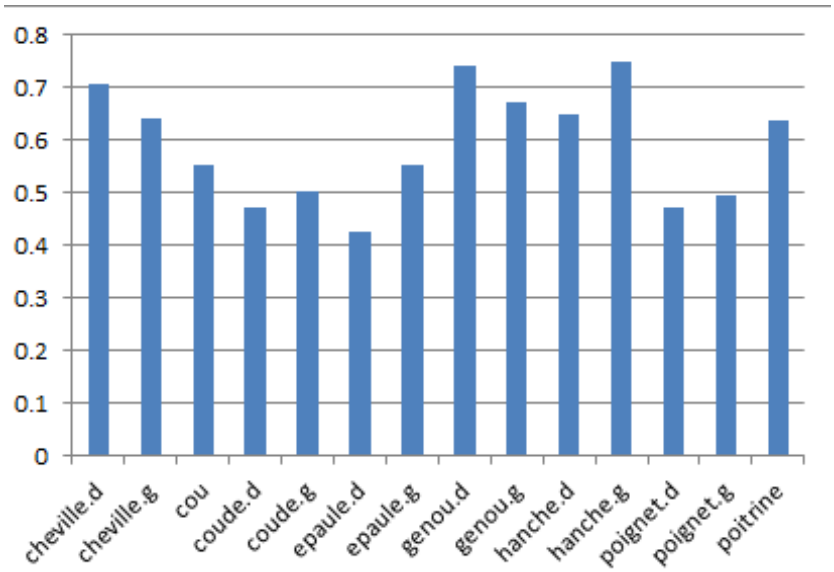


Figure 3-26: Performances sur base d'images réelles.

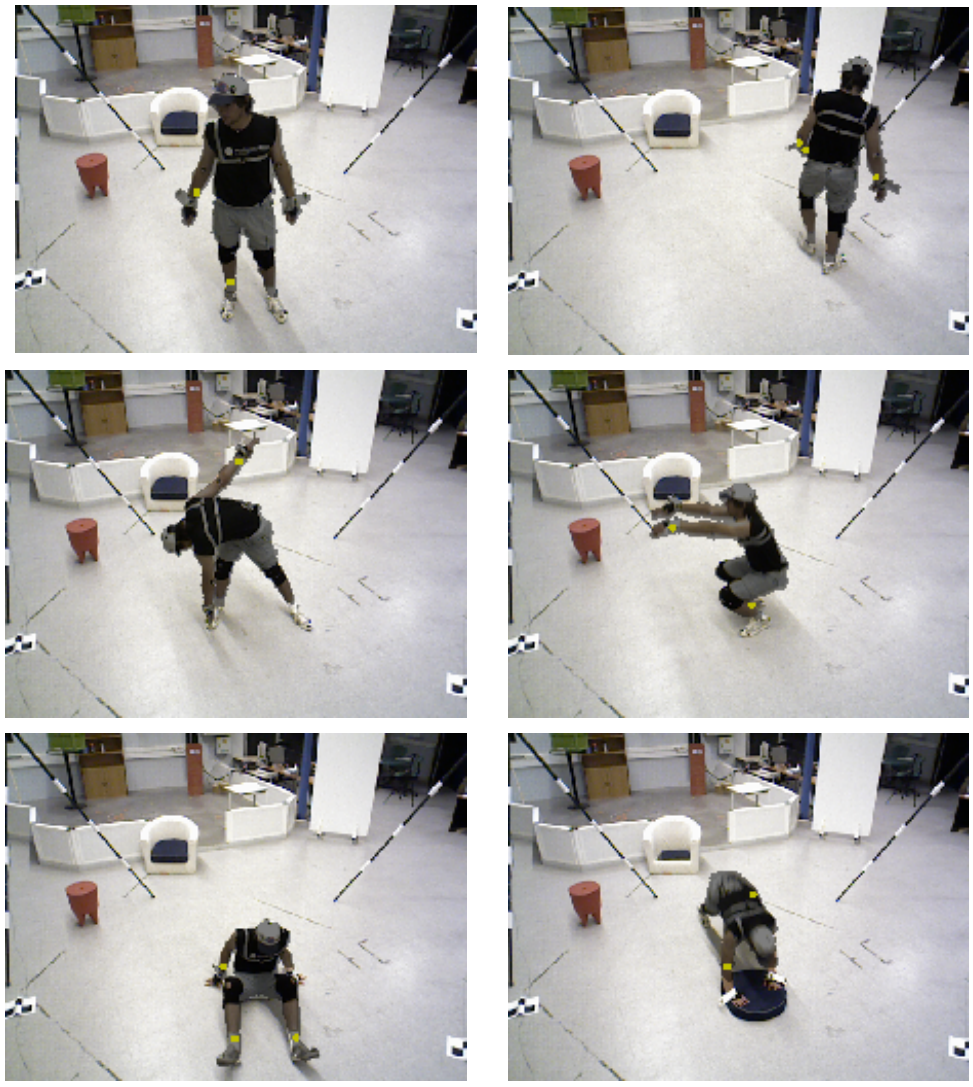
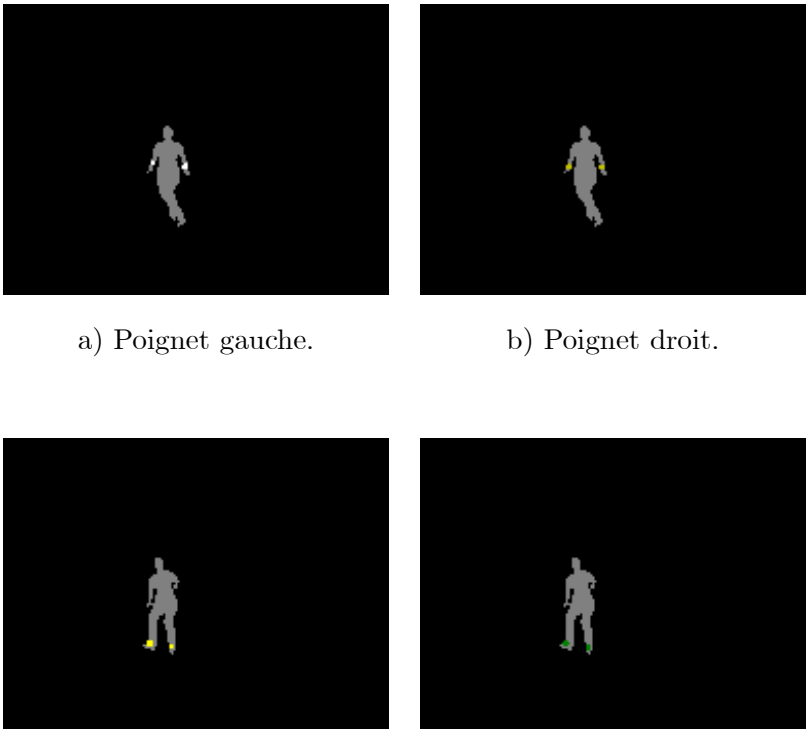


Figure 3-27: Positions des modes retenus pour le poignet droit pour différentes images.

3.5.8 Mise en évidence de l'ambiguïté droite/gauche

Rappelons que nous faisons un apprentissage d'un ensemble de postures avec rotation complète du corps. De ce fait, l'apparence de deux membres de même nature droite et gauche est similaire. Par conséquent, nous avons une grande ambiguïté droite/gauche sur les membres où cela a un sens.



a) Poignet gauche. b) Poignet droit.

c) Cheville gauche. d) Cheville droite.

Figure 3-28: Ambiguité droite/gauche.

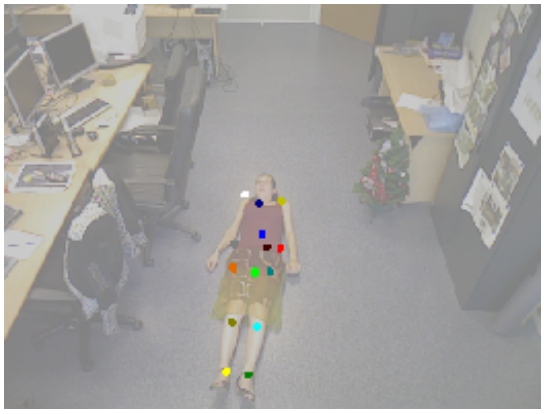
Les figures 3-28 montrent cette ambiguïté sur des images simulées. Elles indiquent, sur des images choisies à cet effet, la position des deux modes possédant le plus grand poids. On voit que sur ces images, ces deux modes se situent sur chacun des deux membres. Toutefois, ce n'est pas le cas pour toutes les images. Afin d'améliorer les résultats, il parait donc important de trouver une stratégie permettant de limiter cette ambiguïté en tirant parti de la cohérence spatio-temporelle. Nous présentons au chapitre suivant de telles stratégies.

3.5.9 Comparaison à l'état de l'art

Nous comparer à Shotton et al. [52] membre par membre n'est pas possible car nous n'estimons pas la position des mêmes articulations. Cependant, nous pouvons fournir une comparaison sur les performances générales de l'algorithme.

Pour notre ensemble de postures de test qui contient un ensemble de postures variées avec des personnes au sol et une rotation complète du corps, nous obtenons une Pmg de 0.591. Girschick obtient, avec 300000 images dans un contexte similaire, une Pmg de 0.711. L'implémentation des algorithmes de Shotton et al. [52] n'est pas disponible, il est donc impossible de nous comparer à eux sur notre base de test de postures de chute. Les moyens limités dont nous disposons ne nous permettent pas de faire des apprentissages avec autant d'images et par conséquent, la comparaison est non pertinente.

Les implémentations de ces algorithmes dans Nite fournissent une posture filtrée. La comparaison avec ces algorithmes sera réalisée dans le chapitre 4 de filtrage. On peut cependant donner un exemple de comparaison qualitative sur une posture de chute comme illustré sur la figure 3-29 où sont représentées les positions des modes les plus importants pour notre algorithme et la sortie finale pour Nite. Nous pouvons voir que sur cette posture, Nite ne parvient pas a donner des positions satisfaisantes pour la grande majorité des articulations. Notre algorithme, malgré une erreur pour le coude gauche, est bien plus performant.



a) Estimation de pose par Nite.

b) Estimation de posture par notre implémentation.

Figure 3-29: Position des articulations sur une posture de chute par Nite et notre algorithme.

3.6 Conclusion

Nous avons développé et mis en oeuvre un pipeline complet de génération de données d'apprentissage ; allant de la motion capture à la création d'images d'apprentissage pour notre algorithme. L'apprentissage a ensuite été réalisé grâce aux données générées à l'aide de ce pipeline. La création de cet ensemble algorithmique nous permet, en un temps court, de nous adapter à de nouvelles applications.

Sur des algorithmes existants de Random Forests pour l'estimation de posture, nous avons réglé certains paramètres qui n'avaient pas été étudiés dans les approches nominales.

Au niveau des caractéristiques à calculer sur les images, nous avons montré que l'ajout de la hauteur du pixel dans la scène permet un gain de performances. L'équilibrage de la collection d'apprentissage a aussi été évalué et nous en avons déterminé les apports. Les résultats étant prometteurs, il pourrait être intéressant de les approfondir, par exemple, en évaluant d'autres idées pour l'équilibrage de la collection ou d'autres types de caractéristiques. Il faudra toutefois garder à l'esprit que celles-ci doivent-être rapides à calculer sous peine de voir le temps d'apprentissage augmenter de façon prohibitive.

Notre algorithme fournissant un certain nombre de prédictions pour la position 3D de chaque articulation, il nous faut maintenant utiliser la cohérence spatio-temporelle pour trouver à chaque instant la plus probable. C'est l'objet des travaux présentés dans le chapitre suivant.

Chapitre 4

Filtrage multi-modal et levée d'ambiguïté entre parties corporelles

4.1 Introduction

L'algorithme décrit dans le chapitre précédent nous fournit un certain nombre d'hypothèses pour la position de chaque membre du corps humain. Ce nombre dépend du membre considéré et varie à chaque nouvelle image. A la sortie de notre algorithme, nous ne souhaitons avoir qu'une seule prédiction la plus robuste possible pour la position de chacune des articulations du corps. Il faut alors choisir à chaque instant la prédiction la plus probable en s'appuyant sur l'information spatio-temporelle. De plus, nous avons montré qu'un certain nombre de points de prédiction pour un membre se trouvent proches du membre du côté opposé pour les articulations en paires. Cela induit une ambiguïté droite/gauche qu'il convient de traiter.

Pour traiter ce problème, nous utilisons une approche de filtrage bayésien mis en oeuvre sous la forme d'un mélange de gaussiennes. Rappelons que l'approche filtrage bayésien consiste à propager la probabilité de l'état d'intérêt conditionnellement aux observations. Cette propagation est classiquement réalisée en deux étapes : prédiction puis correction.

Nous allons d'abord présenter quelques travaux similaires puis présenter nos deux algorithmes de filtrages et leurs performances respectives. Le premier algorithme permet de filtrer chaque membre indépendamment en propageant plusieurs hypothèses. Le second

algorithme incrémente sur le premier et traite spécifiquement l'ambigüité droite/gauche sur les paires de membres. Les performances sur des séquences de chute surpassent celles de l'implémentation de Nite.

4.2 Travaux similaires

Nous présentons dans cette partie les travaux de filtrage les plus similaires aux nôtres. Dans le cas linéaire gaussien, la solution optimale est le filtre de Kalman. Dans le cas non-linéaire gaussien, une approximation linéaire locale des fonctions de dynamique et d'observation conduisent au filtre dit de Kalman étendu. A noter que cette approche n'est valide que dans le cas où la solution est mono-modale. Dans le cas général, non-linéaire, non-gaussien et donnant lieu à une solution multi-modale, on opte souvent pour une solution nommée filtrage particulaire comme présenté par Tenorth et al dans [57] ou encore Deutscher et al dans [21] et [22]. Cette approche repose sur l'approximation de la solution, la densité de probabilité de l'état conditionnellement aux observations, sous la forme d'une somme pondérée de mesures de Dirac. Un très grand nombre d'approches utilisent cette stratégie comme recensé par Peursum et al dans [47]. Les stratégies se sont révélées payantes dans le contexte du suivi 2D comme présenté dans Agarwal et al. dans [6]. Les stratégies de filtrage particulaire sont toutefois coûteuses en temps CPU. Certains travaux utilisent les contraintes sur la posture humaine comme Riu et al. dans [58].

Les travaux de Sminchisescu et al. dans [54] utilisent une modélisation gaussienne des modes dans une distribution d'état du filtrage particulaire pour traquer une posture humaine dans des images couleurs monoculaires. Mais les travaux utilisant les mélanges de gaussiennes pour le filtrage restent rares.

Nous présenterons dans un premier temps notre algorithme de filtrage multi-modal considérant les membres indépendants les uns des autres. Puis nous présenterons une amélioration de cet algorithme qui filtre les membres par paires droite/gauche.

4.3 Filtrage multi-modal sur un membre

Dans cette partie, nous traitons les membres indépendamment les uns des autres. Les distributions de probabilité de l'état, conditionnelles aux observations, sont par nature multi-modales. En effet, l'algorithme de prédiction décrit au chapitre précédent fournit un certain de nombre de candidats possibles pour la position de chaque articulation, ces

positions étant dispersées dans l'espace. Une représentation par mélange de gaussiennes est donc parfaitement appropriée dans ce contexte. Dans un premier temps nous présentons la formalisation mathématique de notre filtre, puis nous présentons les évaluations de performances réalisées sur notre propre base de séquences. Dans la suite, afin de simplifier les écritures, nous omettrons l'indice du membre dans nos notations.

4.3.1 Formalisation

La position de chaque membre est donnée par ses coordonnées 3D dans le repère de la caméra :

$$X_t = \begin{bmatrix} x_t \\ y_t \\ z_t \end{bmatrix}$$

Les positions des articulations étant considérées ici indépendantes, on représente la densité de probabilité solution (conditionnelle aux observations), pour chaque membre, sous la forme d'une somme pondérée de densités gaussiennes. Celle-ci possède donc un nombre N_t^e de modes qui représentent les différentes hypothèses pour la position réelle du membre à un instant t. Cette densité de probabilité se représente donc comme suit :

$$P(X_t|Y_{0\ldots t}) = \sum_{i=1}^{N_t^e} \rho_t^i \Gamma(X_t, \hat{X}_{t|t}^i, P_{t|t}^i) \tag{4.1}$$

où $\Gamma\left(X, \hat{X}, P\right)$ désigne la densité de probabilité gaussienne de moyenne $\hat{X}$ et de variance P. L'ensemble $Y_{0\ldots t}$ désigne l'ensemble des observations disponibles sur l'horizon $[0, t]$. Si les moyennes $\hat{X}_{t|t}^i$ de ces gaussiennes sont suffisamment dispersées dans l'espace, et que les variances $P_{t|t}^i$ sont suffisamment petites pour que les densités gaussiennes ne se chevauchent que peu, on peut interpréter ρ_t^i comme étant la probabilité que X_t se trouve au voisinage de $\hat{X}_{t|t}^i$. Ainsi, cette modélisation permet d'entretenir plusieurs hypothèses pour la position de chaque membre. En effet, il n'est pas garanti que le mode possédant le poids ρ_t^i le plus élevé à un instant t soit le "bon mode", c'est-à-dire la position la plus proche de la position réelle du mode.

4.3.2 Equations d'évolution

Nous cherchons à filtrer un ensemble de mouvements très divers. Il est donc difficile de proposer un modèle de prédiction complexe pour la position des membres, en particulier

lorsque l'on considère un membre isolé des autres. Le modèle proposé pour l'évolution de la position d'un membre en fonction du temps est alors un modèle de marche aléatoire de covariance Q qui ne dépend pas du temps. L'équation d'évolution est alors définie comme suit :

$$P(X_t|X_{t-1}) = \Gamma(X_t, FX_{t-1}, Q) \tag{4.2}$$

Dans notre cas, $F = I_3$ et la variance Q dépend du membre. Certains membres comme les bras ou les jambes possèdent des mouvements plus amples et plus incertains que d'autres, comme le torse ou les épaules, ils possèdent alors naturellement des matrices de covariance de bruit de processus plus élevées. Ces matrices sont définies comme étant diagonales. En effet, les trois coordonnées de l'espace sont modélisées de façon indépendante dans notre approche. Toutes ces composantes possèdent la même valeur de variance et nous avons $Q = (\sigma_Q)^2 \times I_3$.

4.3.3 Equation d'observation

Pour chaque membre, l'algorithme de prédiction décrit au chapitre précédent fournit un certain nombre de positions candidates que nous noterons $Y_t = \left\{Y_t^1, \ldots, Y_t^{N_t^{obs}}\right\}$. Comme nous n'avons a priori aucune raison de donner un poids plus fort à une position qu'à une autre, on considérera que la probabilité a priori de chaque hypothèse est identique pour chacune d'entre-elles est donc égale à $1/N_t^e$. La densité de probabilité de l'observation conditionnelle à l'état pourra donc être écrite sous la forme :

$$P(Y_t|X_t) = \frac{1}{N_t^{obs}} \sum_{i=1}^{N_t^{obs}} \Gamma(Y_t^i, X_t, R) \tag{4.3}$$

où R désigne la variance du bruit d'observation. Toutes les composantes possèdent la même valeur de bruit d'observation et nous avons $R = (\sigma_R)^2 \times I_3$.

4.3.4 Filtrage

Rappelons que le but du filtrage est de calculer récursivement dans le temps la densité de probabilité $p\left(X_t|Y_{0\ldots t}\right)$ où $Y_{0\ldots t}$ désigne l'ensemble des observations disponibles sur l'horizon $[0, t]$.

Prédiction Cette étape consiste à calculer la transition de $p\left(X_{t-1}|Y_{0\ldots t-1}\right)$ à $p\left(X_t|Y_{0\ldots t-1}\right)$. Pour ce faire, on utilise la formule de Chapman-Kolmogorov, soit :

$$p\left(X_t|Y_{0\ldots t-1}\right) = \int p\left(X_t|X_{t-1}\right) p\left(X_{t-1}|Y_{0\ldots t-1}\right) dX_{t-1}$$

Si à l'instant $t-1$, la densité de probabilité conditionnelle était exprimée selon l'expression 4.1, on substituant celle-ci on obtient :

$$p\left(X_t|Y_{0\ldots t-1}\right) = \sum_{i=1}^{N_{t-1}^e} \rho_{t-1}^i \int \Gamma(X_t, FX_{t-1}, Q)\Gamma(X_{t-1}, \hat{X}_{t-1|t-1}^i, P_{t-1|t-1}^i)dX_{t-1}$$

Chacune des intégrales sont celles qui apparaissent au moment du calcul de l'étape de prédiction d'un filtre de Kalman. Le calcul classique de cette somme de convolutions conduit alors à :

$$p\left(X_t|Y_{0\ldots t-1}\right) = \sum_{i=1}^{N_{t-1}^e} \rho_{t-1}^i \Gamma(X_t, \hat{X}_{t|t-1}^i, P_{t|t-1}^i)$$

avec

$$\begin{aligned} \hat{X}_{t|t-1}^i &= F\hat{X}_{t-1|t-1}^i \\ P_{t|t-1}^i &= FP_{t-1|t-1}^i F^T + Q \end{aligned}$$

Correction Cette étape consiste à passer de $p\left(X_t|Y_{0\ldots t-1}\right)$ à $p\left(X_t|Y_{0\ldots t}\right)$, c'est-à-dire d'intégrer la dernière observation Y_t au processus de filtrage. Elle s'accomplit en utilisant la formule de Bayes, soit :

$$p\left(X_t|Y_{0\ldots t}\right) = \frac{p\left(Y_t|X_t\right) p\left(X_t|Y_{0\ldots t-1}\right)}{p\left(Y_{0\ldots t}\right)}$$

Rappelons que le dénominateur est un terme de normalisation. Si on omet celui-ci, on pourra écrire, en exploitant notre représentation de la fonction d'observation 4.3 :

$$p\left(X_t|Y_{0\ldots t}\right) \propto \frac{1}{N_t^{obs}} \sum_{j=1}^{N_{t-1}^e} \sum_{i=1}^{N_t^{obs}} \rho_{t-1}^j \Gamma(Y_t^i, X_t, R)\Gamma(X_t, \hat{X}_{t|t-1}^j, P_{t|t-1}^j)$$

A nouveau, ce produit de densités gaussiennes peut se récrire sous la forme d'une seule gaussienne par les formules standards utilisées dans le filtre de Kalman. On aura donc :

$$\Gamma(Y_t^i, X_t, R_t^i)\Gamma(X_t, \hat{X}_{t|t-1}^j, P_{t|t-1}^j) = \Gamma\left(Y_t^i, \hat{X}_{t|t-1}^j, P_{t|t-1}^j + R\right)\Gamma\left(X_t, \hat{X}_{t|t}^{i,j}, P_{t|t}^j\right)$$

avec :

$$\begin{aligned}
\hat{X}_{t|t}^{i,j} &= \hat{X}_{t|t-1}^j + K_t^j\left(Y_t^i - \hat{X}_{t|t-1}^j\right) \\
K_t^j &= P_{t|t-1}^j\left(P_{t|t-1}^j + R\right)^{-1} \\
P_{t|t}^j &= P_{t|t-1}^j - K_t^j P_{t|t-1}^j
\end{aligned}$$

Ainsi, après correction, en normalisant cette expression, on obtient une expression de la densité de probabilité a posteriori de la même forme qu'à l'étape précédente, soit :

$$p\left(X_t|Y_{0\ldots t}\right) = \sum_{j=1}^{N_{t-1}^e}\sum_{i=1}^{N_t^{obs}} \rho_t^{i,j}\Gamma\left(X_t, \hat{X}_{t|t}^{i,j}, P_{t|t}^{,j}\right)$$

avec :

$$\rho_t^{i,j} = \frac{\rho_{t-1}^j\Gamma\left(Y_t^i, \hat{X}_{t|t-1}^j, P_{t|t-1}^j + R\right)}{\sum_{j=1}^{N_{t-1}^e}\sum_{i=1}^{N_t^{obs}}\rho_{t-1}^j\Gamma\left(Y_t^i, \hat{X}_{t|t-1}^j, P_{t|t-1}^j + R_t^i\right)}$$

Les poids des modes $\rho_t^{i,j}$ peuvent être vus comme la probabilité a posteriori que le mode correspondant soit le mode de la position réelle du membre. A chaque itération, le nombre de modes dans l'estimation de l'état est multiplié par N_{obs}. Il faut donc avoir une stratégie permettant de réduire le nombre de modes à chaque itération du filtre sans quoi celui-ci augmenterait de manière exponentielle.

4.3.5 Choix des modes

Différentes stratégies sont mentionnées dans la littérature pour les choix des modes. Nous considérons que les modes ayant les $\rho_t^{i,j}$ avec les valeurs les plus élevées sont les modes les plus intéressants. Le nombre N_t^{obs} d'observations à l'instant t possède une valeur qui n'est pas constante en fonction du temps car elle dépend du nombre de modes fournis par l'algorithme d'estimation de posture par image du chapitre précédent. Une stratégie consiste à garder constant le nombre N_t^e de modes dans l'estimation de l'état. Nous choisissons alors pour chaque observation j, de garder le mode possédant le $\rho_t^{i,j}$ le plus élevé. Cette

stratégie permet de prendre en compte les nouvelles observations situées potentiellement loin des modes existants $\hat{X}^i_{t|t-1}$ et ainsi il n'est jamais besoin de ré-initialiser le filtre.

4.3.6 Implémentation et évaluations associées

Valeurs des paramètres

Nous donnons dans cette section les valeurs de paramètres utilisées pour nos évaluations.

Le tableau 4.1 donne les valeurs de bruit de process σ_Q pour chacune des articulations considérées. A noter que les distances considérées sont exprimées en millimètres. Le nombre de composantes d'état a été fixé à $N^e_t = 4$. Nos expérimentations n'ont pas montré de meilleures performances avec l'augmentation de cette valeur. La valeur de bruit d'observation a été fixée à $\sigma_R = 50$. Toutes ces valeurs ont été déterminées par expérimentations successives.

Poignets	Coudes	Epaules	Chevilles	Genoux	Hanches	Poitrine	Cou	Tête
100	70	50	100	70	50	50	50	50

Table 4.1: Valeur de bruit de process pour chaque articulation.

Mesure de l'erreur

Pour mesurer les performances de notre algorithme, nous définissons la mesure suivante. A chaque image, une articulation est considérée comme étant bien détectée si le mode ayant le plus grand poids dans la distribution d'état se trouve à une distance inférieure à un seuil ($10cm$ dans les tests) de la position réelle de l'articulation. Cela correspond au nombre de vrais positifs. Pour un membre et sur une séquence, la mesure utilisée est alors le taux de bonnes mesures (Tbm) qui est le nombre de vrais positifs divisé par le nombre total d'images dans la séquence.

Base de test

Pour l'évaluation, nous utilisons un certains nombre de séquences d'images réelles avec différents mouvements que nous explicitons dans le tableau 4.2.

Evaluations

La figure 4-1 montre des exemples d'images pour deux des séquences de test. La couleur des points des articulations est définie comme sur la figure 3-8. Sur certaines images, les

Séquence	Mouvement
1	Marche aller-retour
2	Marche, chute avant
3	Marche, chute arrière
4	Mouvements face caméra

Table 4.2: Séquences de test

deux membres d'une paire sont détectés au même endroit. L'algorithme de la section 4.4 permet de remédier à ce problème.

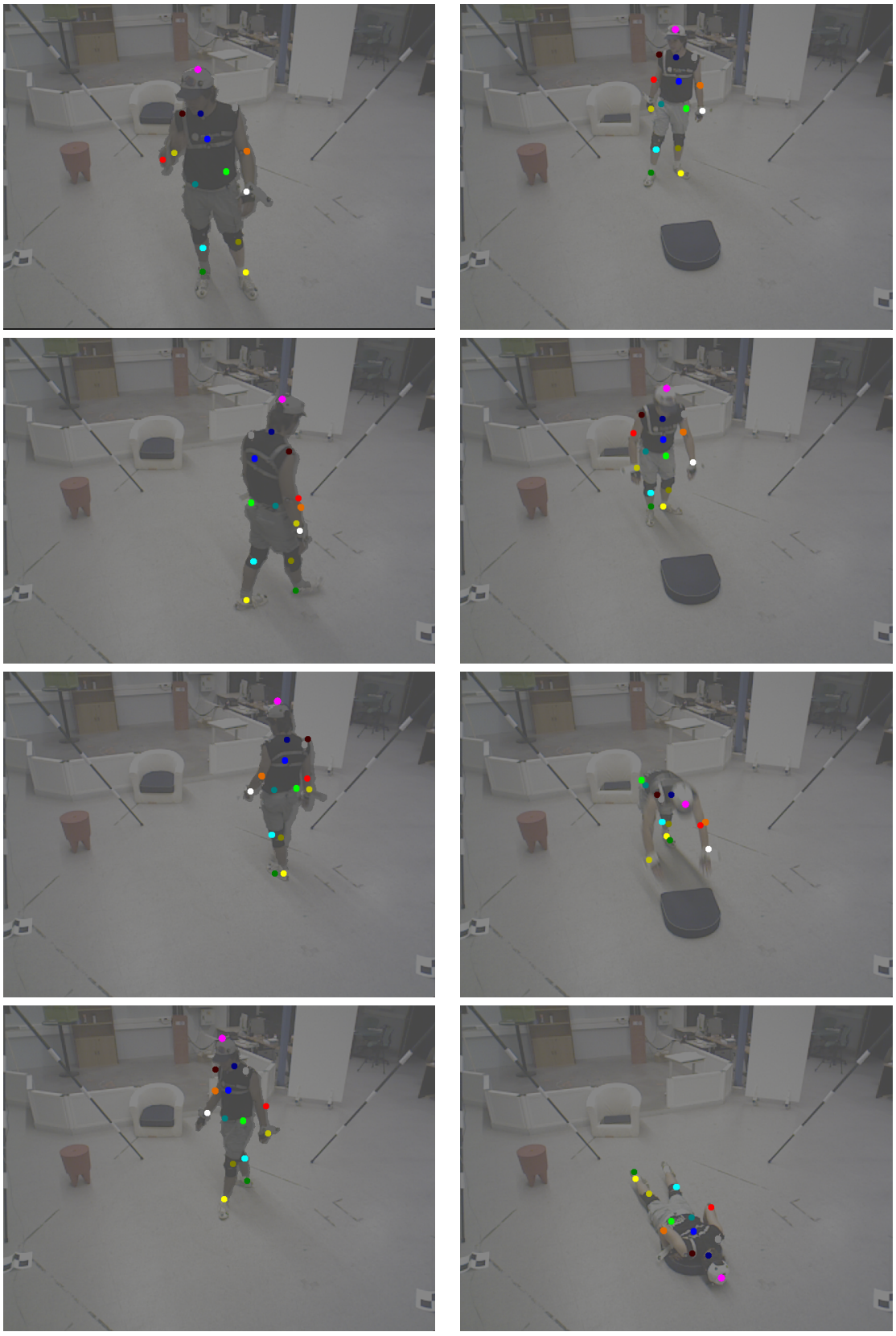
La figure 4-2 présente les résultats pour les différentes séquences de notre base de test. Les résultats sont présentés dans le repère de la caméra. Nous voyons que pour les membres sans paire gauche/droite tels que la tête ou le cou, les performances sont meilleures que pour les membres avec paire. Pour les membres avec paire, il y a parfois une différence importante entre le membre de gauche et le membre de droite. Cela est dû au fait que sur un certain nombre d'images, les estimations pour les deux membres se trouvent à proximité du même membre.

La figure 4-3 montre le resultat du filtrage pour les deux poignets sur la séquence 2. Nous voyons que sur un certain nombre d'images, les positions des deux poignets sont estimées au même endroit. Nous voyons que aussi que sur quelques images aux alentours de l'image 80, une mauvaise estimation de la position du membre par l'algorithme de Random Forests donne une estimation fausse. Le filtre parvient toutefois à suivre le membre.

4.4 Filtrage multi-modal par paire de membres

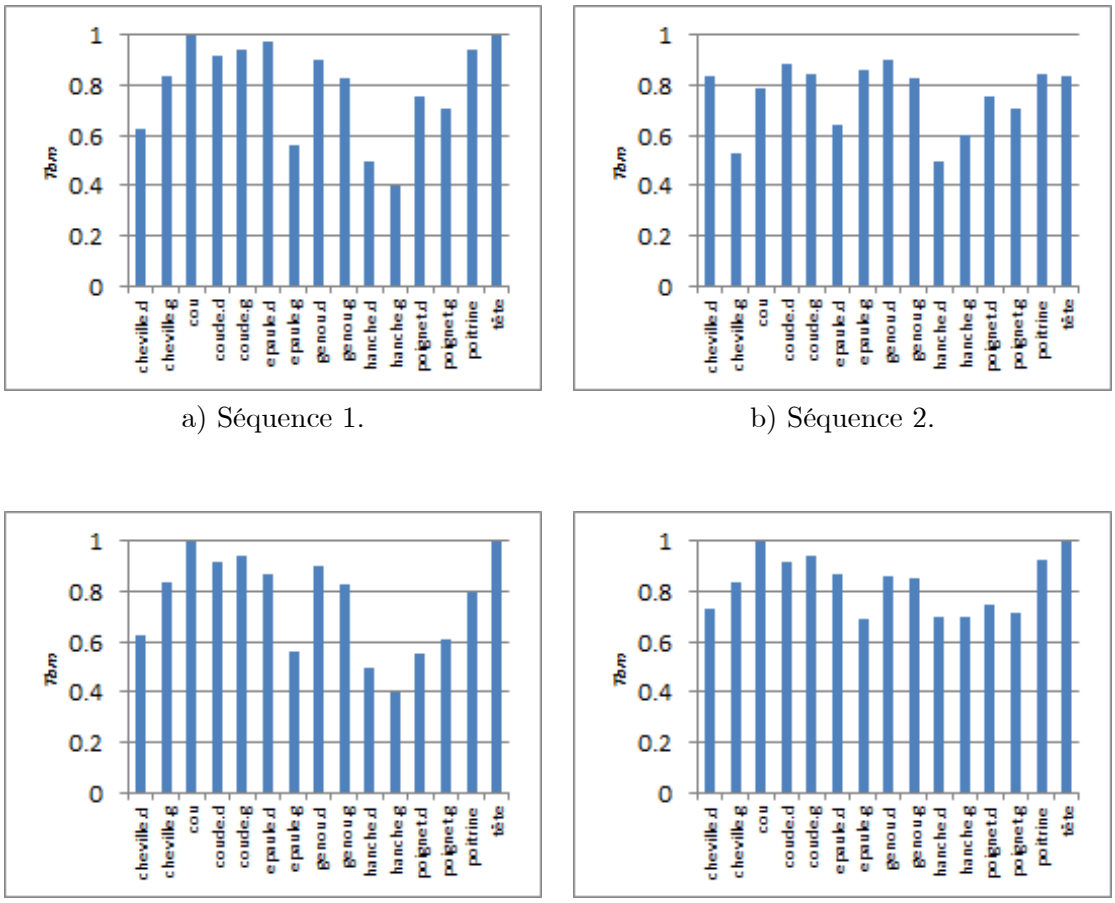
Le filtrage par membre ne permet de répondre que partiellement au problème de levée d'ambiguïté droite/gauche des paires de membres. Afin de rendre plus robuste l'algorithme, nous proposons une approche originale consistant à filtrer les membres par paires de même nature. Ainsi, les positions des membres droit et gauche de la personne sont estimées conjointement avec un filtrage pour lequel on utilise deux hypothèses de la fonction d'observation. L'approche est donc dédiée au traitement spécifique de l'ambigüité droite/gauche.

Ce filtrage n'a de sens naturellement que pour les membres en paire c'est-à-dire les poignets, les coudes, les épaules, les hanches, les genoux et les chevilles.



a) Séquence 1. b) Séquence 2.

Figure 4-1: Exemple de capture pour deux séquences (verticalement) pour le filtrage par membre.



a) Séquence 1. b) Séquence 2.

c) Séquence 3. d) Séquence 4.

Figure 4-2: Evaluations quantitatives pour les différentes séquences pour le filtrage par membre.

4.4.1 Génération des paires d'observations

La sortie de l'algorithme du chapitre précédent nous donne différents modes pour chaque membre. Nous voulons regrouper ces modes pour former des prédictions pour chaque paire de membres. Pour cela nous prenons deux à deux les estimations de position de chaque membre et nous gardons la paire si les deux observations se situent à une distance acceptable l'une de l'autre. Cette distance dépend du membre considéré. En effet, la distance entre deux épaules varie très peu alors que celle entre les poignets ou les genoux possède une variabilité bien plus grande. Lorsqu'aucune paire n'est située à une distance acceptable, ce qui correspond à un cas où toutes les prédictions pour gauche et droite sont situées très proches les unes des autres, nous gardons simplement le mode ayant le poids le plus élevé. Nous n'avons donc qu'une seule observation pour un seul des deux membres

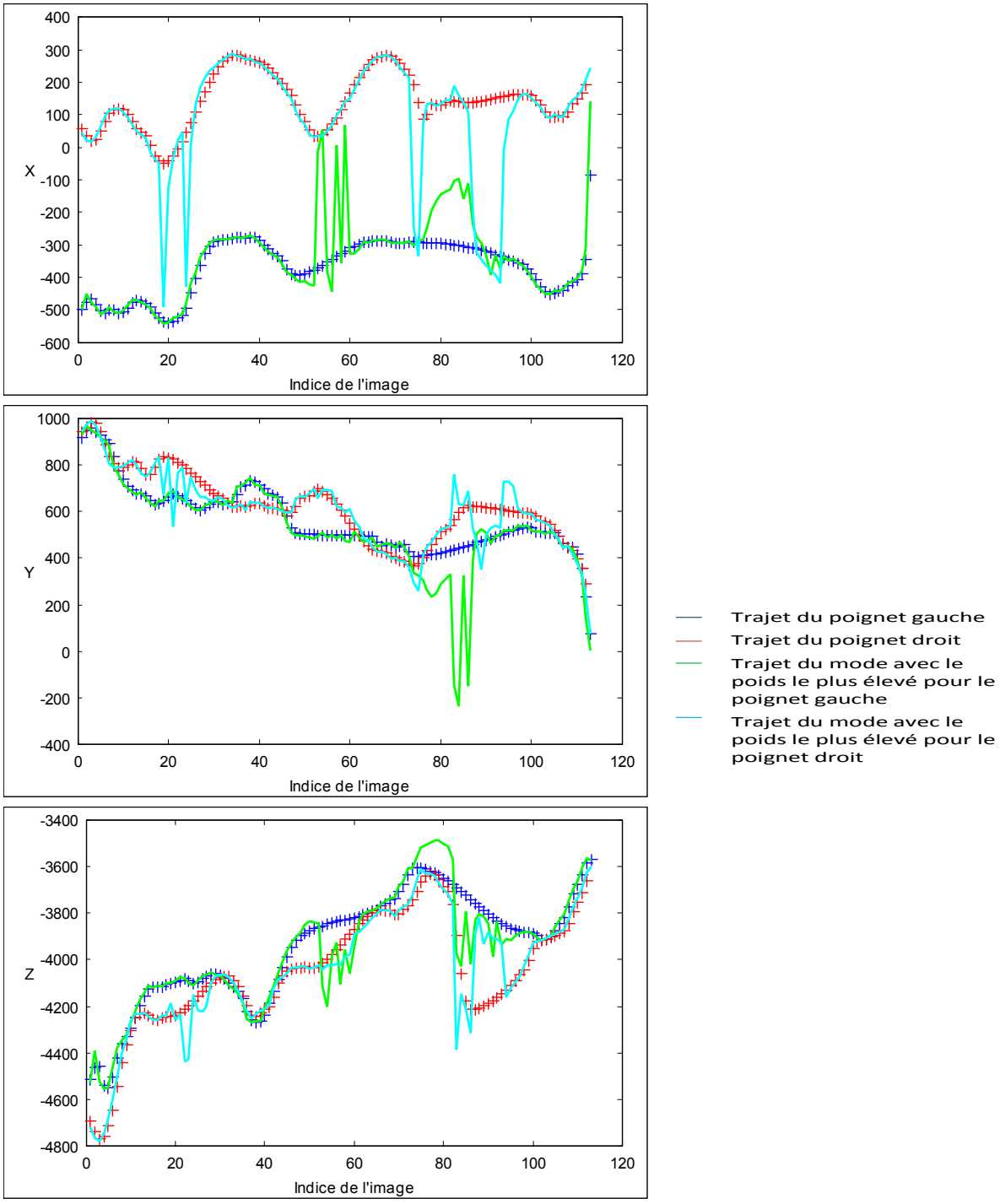


Figure 4-3: Trajectoires filtrées des poignets avec algorithme par membre.

et le filtrage à l'instant t est alors fait selon l'algorithme précédent. L'autre membre est considéré comme non détecté.

4.4.2 Formalisation

A un instant t, nous nous intéressons à la position de deux membres formant une paire droite/gauche. Une paire p est constituée de deux membres m et n. Comme précédemment, puisque nous considérons chaque paire de membres indépendamment les unes des autres, nous omettrons l'indice qui caractérise ce couple. Nous cherchons donc à estimer le vecteur :

$$X_t = \begin{bmatrix} x_{n,t} & y_{n,t} & z_{n,t} & x_{m,t} & y_{m,t} & z_{m,t} \end{bmatrix}^T.$$

Les observations sont formées de paires correspondants aux observations de chaque membres :

$$Y_t^i = \begin{bmatrix} y_{n,t}^{x,i} & y_{n,t}^{y,i} & y_{n,t}^{z,i} & y_{m,t}^{x,i} & y_{m,t}^{y,i} & y_{m,t}^{z,i} \end{bmatrix}^T \tag{4.4}$$

La densité de probabilité a posteriori de $X_{p,t}$ est encore une fois modélisée par un mélange de gaussiennes, nous avons donc l'équation 4.5 :

$$P(X_t|Y_{0\ldots t}) = \sum_{i=1}^{N_t^e} \rho_t^i \Gamma(X_t, \hat{X}_{t|t}^i, P_{t|t}^i) \tag{4.5}$$

Les arguments évoqués dans la partie précédente pour cette modélisation restent valables.

4.4.3 Equation d'évolution

Ici encore, peu de changement, nous considérons toujours une marche aléatoire :

$$P(X_t|X_{t-1}) = \Gamma(X_t, X_{t-1}, Q) \tag{4.6}$$

Les matrices de covariance Q peuvent simplement être définies comme la concaténation des deux matrices du membre précédent comme défini dans la partie précédente.

4.4.4 Equation d'observation

Dans ce cas, deux hypothèses d'observation sont alors considérées :

- La première considère que l'observation pour le membre de gauche correspond bien au membre de gauche et le membre de droite au membre de droite.

- La seconde hypothèse considère que l'observation pour le membre de gauche correspond au membre de droite et vice-versa.

Nous avons alors deux matrices d'observation possibles H_1 et H_2 correspondant à chacune de ces hypothèses :

$$H_1 = \begin{bmatrix} I_3 & 0 \\ 0 & I_3 \end{bmatrix}, H_2 = \begin{bmatrix} 0 & I_3 \\ I_3 & 0 \end{bmatrix} \tag{4.7}$$

Nous avons alors une fonction d'observation 4.8 qui peut être représentée par le mélange de densités gaussiennes suivant :

$$P(Y_t|X_t) = \frac{1}{N_t^{obs}} \frac{1}{h_1 + h_2} \sum_{i=1}^{N_t^{obs}} \sum_{k=1}^{2} h_k \Gamma(Y_t^i, H_k X_t, R) \tag{4.8}$$

Les poids h_1 et h_2 sont respectivement les probabilités d'être dans l'hypothèse 1 ou dans l'hypothèse 2. L'hypothèse 1 est considérée comme la plus probable. Cela revient à dire que l'algorithme nous fournit plus souvent des paires de membres "dans le bon sens" que des paires "dans le mauvais sens". En pratique le réglage exact de ces valeurs n'a pas une grande importance à condition de s'en tenir à $h_1 > h_2$. De même que pour Q, R est définie comme la concaténation des deux matrices pour chacun des deux membres.

4.4.5 Filtrage

Prédiction

La prédiction se fait de manière identique au cas précédent. On aura donc :

$$p\left(X_t|Y_{0\ldots t-1}\right) = \sum_{i=1}^{N_{t-1}^e} \rho_{t-1}^i \Gamma(X_t, \hat{X}_{t|t-1}^i, P_{t|t-1}^i)$$

avec :

$$\begin{aligned} \hat{X}_{t|t-1}^i &= F\hat{X}_{t-1|t-1}^i \\ P_{t|t-1}^i &= FP_{t-1|t-1}^i F^T + Q \end{aligned}$$

Correction

A nouveau, la formule de Bayes permet d'écrire la densité de probabilité a posteriori sous la forme

$$p\left(X_t|Y_{0...t}\right) \propto \frac{1}{N_t^{obs}} \frac{1}{h_1+h_2} \sum_{k=1}^{2} \sum_{i=1}^{N_t^{obs}} \sum_{j=1}^{N_{t-1}^e} \rho_{t-1}^j h_k \Gamma(Y_t^i, H_k X_t, R) \Gamma(X_t, \hat{X}_{t|t-1}^j, P_{t|t-1}^j)$$

et la récriture des produits de densités gaussiennes conduit à

$$\Gamma(Y_t^i, H_k X_t, R) \Gamma(X_t, \hat{X}_{t|t-1}^j, P_{t|t-1}^j) = \Gamma\left(Y_t^i, H_k \hat{X}_{t|t-1}^j, H_k P_{t|t-1}^j H_k^T + R\right) \Gamma\left(X_t, \hat{X}_{t|t}^{i,j,k}, P_{t|t}^{i,j,k}\right)$$

En définitive, la mise à jour de la densité de probabilité a posteriori s'écrit

$$p\left(X_t|Y_{0...t}\right) = \sum_{k=1}^{2} \sum_{i=1}^{N_t^{obs}} \sum_{j=1}^{N_{t-1}^e} \rho_t^{i,j,k} \Gamma(X_t, \hat{X}_{t|t}^{i,j,k}, P_{t|t}^{j,k})$$

avec

$$\begin{aligned} \hat{X}_{t|t}^{i,j,k} &= \hat{X}_{t|t-1}^j + K_t^{j,k}\left(Y_t^i - H_k \hat{X}_{t|t-1}^j\right) \\ K_t^{j,k} &= P_{t|t-1}^j H_k^T \left(H_k P_{t|t-1}^j H_k^T + R\right)^{-1} \\ P_{t|t}^{j,k} &= P_{t|t-1}^j - K_t^{j,k} P_{t|t-1}^j H_k^T \end{aligned}$$

et où les nouveaux poids sont obtenus par

$$\rho_t^{i,j,k} = \frac{\Gamma\left(Y_t^i, H_k \hat{X}_{t|t-1}^j, H_k P_{t|t-1}^j H_k^T + R\right) \rho_{t-1}^j h_k}{\sum_{k=1}^{2} \sum_{i=1}^{N_t^{obs}} \sum_{j=1}^{N_{t-1}^e} \Gamma\left(Y_t^i, H_k \hat{X}_{t|t-1}^j, H_k P_{t|t-1}^j H_k^T + R\right) \rho_{t-1}^j h_k}$$

On constate que, comme on pouvait s'y attendre, le poids de chaque hypothèse prend en compte :

- l'ancienne probabilité que le mode j soit le "bon mode" à travers le terme ρ_{t-1}^j,

- la vraisemblance que la mesure i corresponde à ce mode et que l'inversion soit présente ou non à travers le terme $\Gamma\left(Y_t^i, H_k \hat{X}_{t|t-1}^j, H_k P_{t|t-1}^j H_k^T + R\right)$,

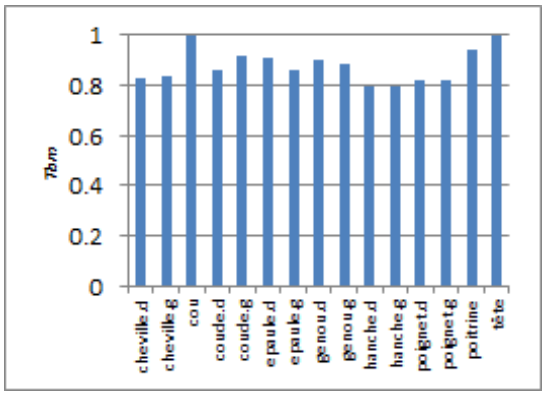
- le fait qu'a priori l'algorithme de prédiction du chapitre précédent donne plus souvent un bon appariement que le contraire à travers le terme h_k.

4.4.6 Choix des modes

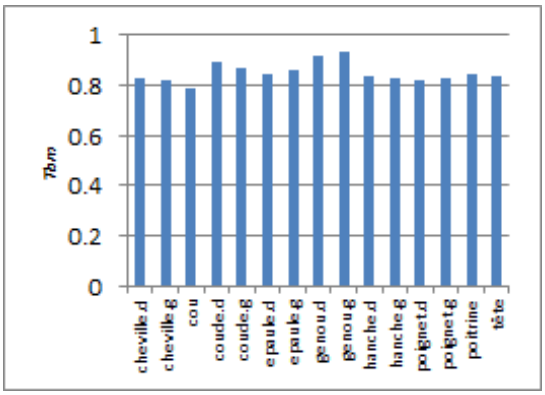
Le choix des modes est fait de la même façon que lors du filtrage multimodal par membre simple.

4.4.7 Evaluations

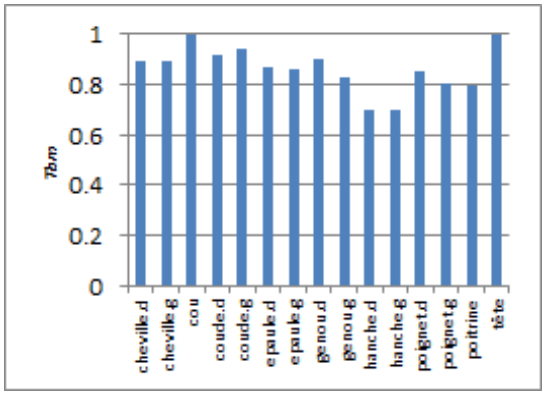
La figure 4-4 montre les résultats sur nos séquences de test avec notre algorithme de filtrage par paire. Nous voyons une nette amélioration des résultats sur les membres par paires et en particulier une diminution importante de l'ambigüité droite/gauche.



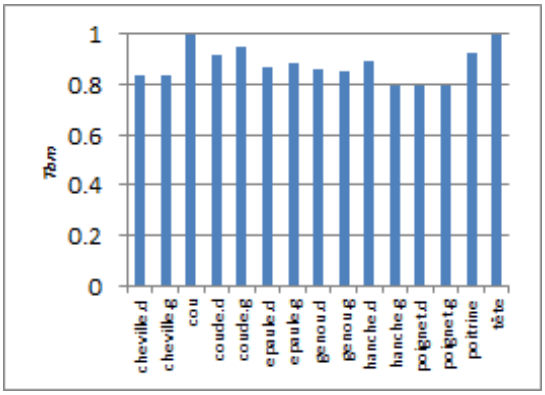
a) Séquence 1.



b) Séquence 2.



c) Séquence 3.



d) Séquence 4.

Figure 4-4: Evaluations quantitatives pour les différentes séquences pour le filtrage par paires.

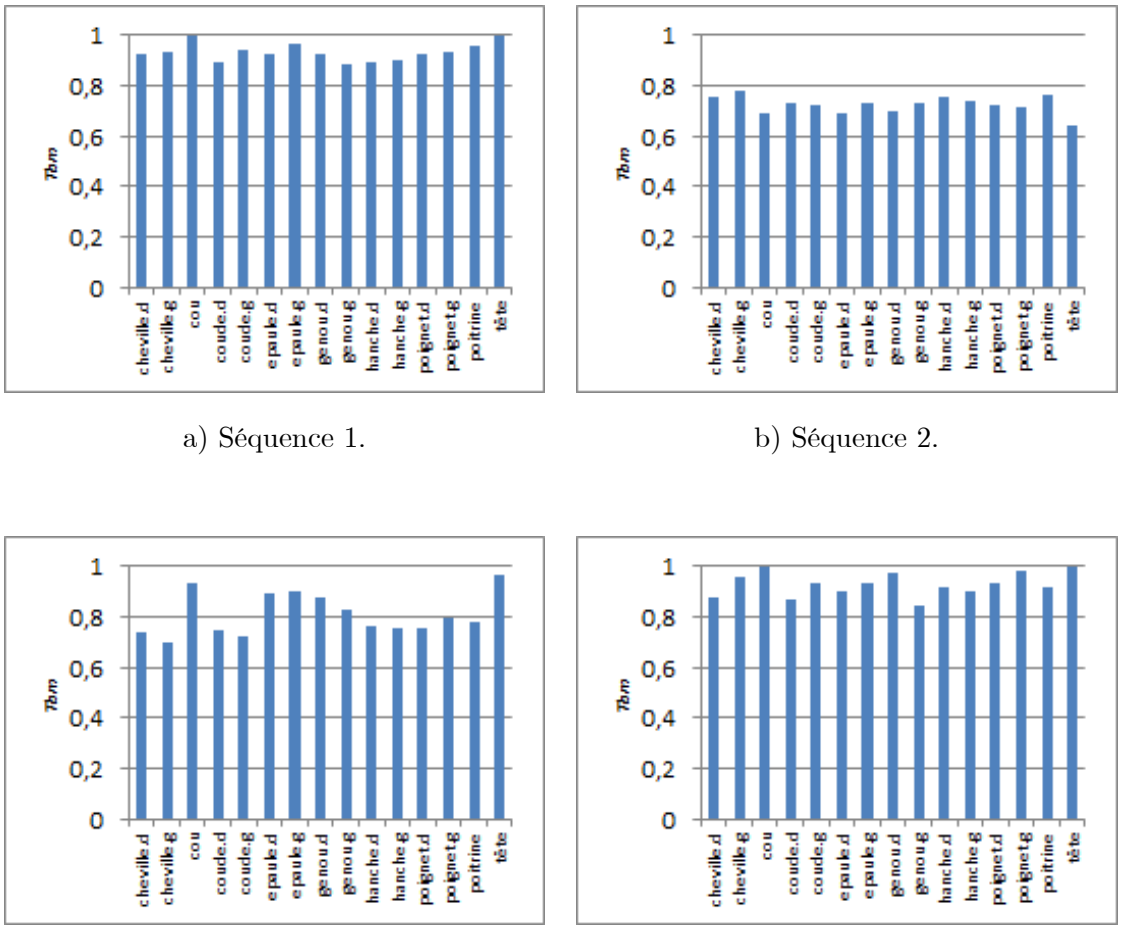
Cette diminution est aussi illustrée sur la figure 4-5. Certains membres ne sont pas détectés plûtot que d'être détectés au mauvais endroit comme c'est le cas avec l'algorithme

précédent.

La figure 4-6 montre les résultats du filtrage avec l'algorithme par paire. Un grand nombre des problèmes d'ambigüité a été supprimé. Il reste toutefois des erreurs importantes lors des erreurs d'observation.

4.4.8 Comparaison à Nite

Nous avons évalué Nite sur nos séquences de test. Les résultats sont présentés sur la figure 4-7.



a) Séquence 1.

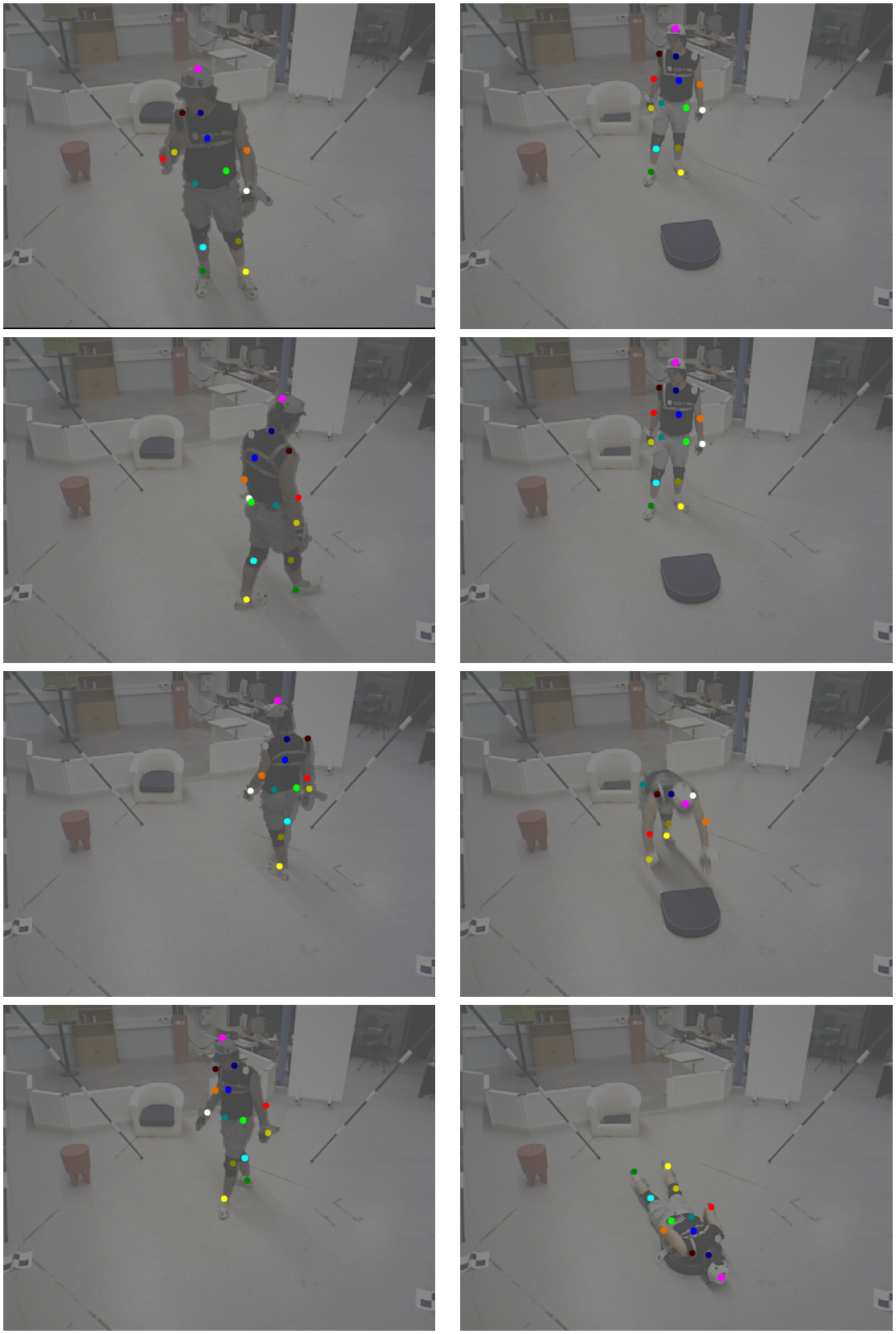
b) Séquence 2.

c) Séquence 3.

d) Séquence 4.

Figure 4-7: Evaluations quantitatives de Nite pour les différentes séquences.

Sur les séquences debout, Nite reste meilleur que notre implémentation. Mais nous voyons que sur les séquences de chutes, notre algorithme présente des performances supérieures aux algorithmes de Nite.



a) Séquence 1. b) Séquence 2.

Figure 4-5: Exemples de capture sur deux séquences (verticalement) avec filtrage par paires.

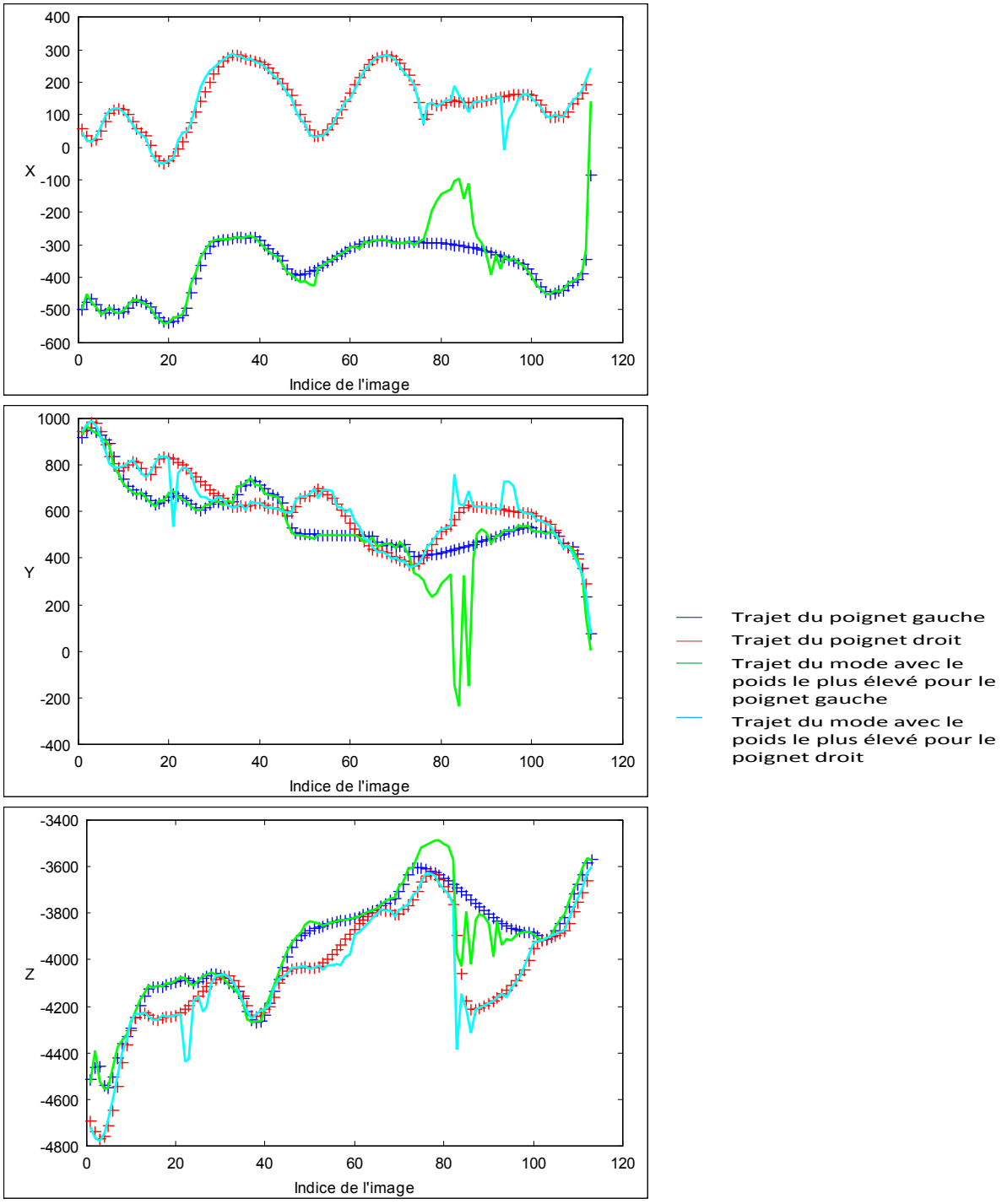


Figure 4-6: Trajectoires filtrées des poignets avec l'algorithme par paires.

4.5 Conclusion et perspectives

Nous avons présenté dans ce chapitre deux algorithmes de filtrage bayésien utilisant le mélange de gaussiennes comme représentation d'état.

Le premier traite les membres indépendamment et présente de bonnes performances sur les membres simples. Sur les membres par paire cependant, une ambigüité droite/gauche subsiste.

Le second algorithme filtre les membres par paire et permet de s'affranchir de la majorité des problèmes d'ambigüité droite/gauche. Notre approche originale dédiée au traitement de l'ambigüité droite/gauche permet de nous comparer favorablement à l'implémentation de Nite sur les séquences de chute.

L'amélioration de ces algorithmes est envisagée avec la prise en compte d'une contrainte de longueur sur la taille des membres qui permettrait de mieux gérer les fausses observations.

Chapitre 5

Conclusion et perspectives

Cette partie résume nos travaux et les contributions associées; nous listons alors quelques perspectives possibles à ces travaux. La dernière section évoque le prototype de système de détection de chute développé par ORME.

5.1 Conclusion des travaux

Les travaux de cette thèse ont permis le développement d'un "pipeline" complet d'estimation de la posture humaine en relation avec notre problématique de détection de chute des personnes âgées dans les EHPAD.

Le chapitre 1 introduit le contexte médical et social de la problématique de détection de chute qui a initié cette thèse CIFRE. Au-delà, nous définissons les besoins plus généraux en estimation de postures de la société ORME. Le cahier des charges établi justifie une solution mono-capteur type Asus Xtion et des algoritmes de segmentation de silhouettes, reconstruction et filtrage 3D de postures.

Le chapitre 2 décrit notre algorithme de segmentation de silhouettes de personnes dans une séquence vidéo RGB-D. La contribution se situe dans une gestion intelligente du canal profondeur dans une modélisation par mixtures de gaussiennes pour labelliser les pixels de la silhouette humaine. Notre approche présente des performances meilleures que celles des algorithmes existants et notamment que OpenNI. Grâce à ces travaux nous avons pu produire un premier prototype de détecteur de chute que nous présentons dans la section 5.3.

Le chapitre 3 décrit notre propre implémentation des algorithmes de Random Forests inspirée de Shotton et al. dans [52]. Les travaux comprennent la génération de données

d'apprentissage de synthèse représentant des postures classiques et des postures de chute, l'apprentissage de ces données et l'évaluation de notre implémentation. Nous avons considéré un système commercial de capture de mouvements pour générer des dynamiques de chutes réalistes et constituer une vérité terrain pour ces évaluations quantitatives. Nous avons alors un "pipeline" complet permettant de s'adapter rapidement à de nouveaux besoins. Notre implémentation inclut deux amendements majeurs à l'algorithme générique. D'abord en proposant d'utiliser la hauteur d'un pixel comme caractéristique. Puis en équilibrant la base d'apprentissage. Les résultats montrent une amélioration substantielle des performances. L'algorithme fournit plusieurs propositions pour la position 3D de chaque articulation et une limitation forte reste la présence d'ambiguïtés. Pour les gérer, nous utilisons un algorithme de filtrage présenté dans le chapitre 4.

Le chapitre 4 présente notre stratégie de filtrage spatio-temporel et multi-modale. Les observations sont logiquement issues de la reconstruction de posture du chapitre 3. Deux variantes de filtrage bayésien multi-modaux sont présentées. Notamment nous avons traité de façon spécifique le cas de l'ambigüité droite/gauche qui apparait pour certains membres. Notre méthode permet de se débarrasser de la grande majorité de ces ambiguïtés à l'aide d'un filtrage par paire de membres.

5.2 Perspectives

Pour notre algorithme de segmentation de silhouette de personnes, l'algorithme de regroupement des pixels en silhouettes pourrait être amélioré. La prise en compte d'une information temporelle permettrait d'éviter la fusion des silhouettes lorsque celles-ci se rapprochent trop. L'implémentation sur un processeur graphique (GPU) est aussi envisagée pour améliorer la vitesse de détection.

Nous souhaitons aussi évaluer nos algorithmes sur le capteur Kinect 2. Ce capteur fournit une image de profondeur de bien meilleure qualité que celle fournie par le Xtion. La taille de l'image est plus importante et le bruit sur chaque pixel est plus faible. De plus, le champ de vision est plus élevé ce qui constitue un avantage conséquent pour notre application de vidéo-surveillance.

Nous souhaiterions aussi faire apprendre à nos Random Forests un plus grand nombre d'images. L'obstacle majeur est la mémoire vive nécessaire. L'apprentissage distribué sur plusieurs ordinateurs est une piste possible.

A plus long terme, nous souhaitons tester de nouvelles caractéristiques pour les noeuds des Random Forests. Le coût CPU du calcul de ces caractéristiques sur l'image doit rester faible.

Nous voulons aussi utiliser les postures estimées pour détecter des comportements dangereux dans notre application de détection de chute. Le but est de faire de la prévention au-delà de la simple détection pour donner une réelle plus-value à notre système par rapport à la concurrence.

Pour le filtrage, nous voulons intégrer des contraintes sur la taille des membres afin d'améliorer l'estimation de la position des articulations.

Les développements sur les Random Forests effectués dans nos travaux vont être utilisés dans le projet SEFA-IKKY impliquant la société ORME. La détection de la position et de l'orientation de la tête d'un pilote d'avion exploitera directement ces travaux. Le système de motion capture commercial associé au Kinect permettra encore une fois de créer une base de test et d'apprentissage avec vérité terrain.

5.3 Prototype de détecteur de chute commercial

A la suite des travaux de segmentation de silhouette du chapitre 2, un prototype de système de détection de chute a été réalisé. Le capteur a été associé à un petit ordinateur comme illustré sur la figure 5-1.

Figure 5-1: Capteur et ordinateur constituant notre prototype de système de détection.

Ce prototype a été installé dans dix chambres à l'EHPAD La cocagne de Sainte Foy d'Aigrefeuille en Haute-Garonne. La figure 5-2 montre le capteur installé dans une chambre de cet EHPAD.

La détection de chute est réalisée grâce à la position des silhouettes dans l'espace 3D. Le taux de fausses alarmes reste légèrement trop important pour la commercialisation. Celui-ci se situe aux environs de 2 fausses alarmes par nuit pour l'ensemble des systèmes

Figure 5-2: Capteur installé dans une chambre d'EHPAD.

installés dans l'EHPAD. Le personnel de l'établissement a réagi très positivement aux tests et est prêt à les poursuivre aussi longtemps que nécessaire pour préparer une solution commerciale.

Liste des figures

Bibliographie

[1] "Blender." [Online]. Available: https://www.blender.org

[2] "CMU Mocap Database." [Online]. Available: http://mocap.cs.cmu.edu/

[3] "Nite." [Online]. Available: http://www.primesens.com

[4] "Orme." [Online]. Available: http://www.orme-toulouse.com

[5] "MakeHuman," 2015. [Online]. Available: http://www.makehuman.org/

[6] A. Agarwal and B. Triggs, "Recovering 3D human pose from monocular images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 1, pp. 44–58, Jan. 2006. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/16402618

[7] K. Alahari, G. Seguin, J. Sivic, and I. Laptev, "Pose Estimation and Segmentation of People in 3D Movies," in *Computer Vision (ICCV), 2013 IEEE International Conference on*, 2013, pp. 2112–2119.

[8] M. Andersen, T. Jensen, P. Lisouski, A. Mortensen, M. Hansen, T. Gregersen, and P. Ahrendt, "Kinect Depth Sensor Evaluation for Computer Vision Applications," Tech. Rep., 2012. [Online]. Available: http://eng.au.dk/fileadmin/DJF/ENG/PDF-filer/Tekniske_rapporter/Technical_Report_ECE-TR-6-samlet.pdf

[9] D. Anguelov, B. Taskar, V. Chatalbashev, D. Koller, D. Gupta, G. Heitz, and A. Ng, "Discriminative learning of Markov Random fields for segmentation of 3D scan data," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, no. 2, pp. 169–176, 2005. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1467438%*backslash*nhttp://ieeexplore.iee

[10] Asus, "Xtion." [Online]. Available: http://www.asus.com/Multimedia/Xtion_PRO/

[11] O. Barnich and M. Van Droogenbroeck, "ViBe: A universal background subtraction algorithm for video sequences," *IEEE Transactions on Image Processing*, vol. 20, no. 6, pp. 1709–1724, June 2011. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/21189241

[12] Y. Benezeth, P.-M. Jodoin, B. Emile, H. Laurent, and C. Rosenberger, "Comparative study of background subtraction algorithms," *Journal of Electronic Imaging*, vol. 19, no. 3, p. 033003, July 2010. [Online]. Available: http://electronicimaging.spiedigitallibrary.org/article.aspx?doi=10.1117/1.3456695

[13] D. Binney and J. Boehm, "Performance Evaluation of the PrimeSense IR Projected Pattern Depth Sensor," 2011. [Online]. Available: http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Performance+Evaluation+of

[14] C. Chen, A. Liaw, and L. Breiman, "Using random forest to learn imbalanced data," Tech. Rep., 2004.

[15] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, pp. 603–619, 2002.

[16] T. H. Cormen and Collaborators, *Introduction to Algorithms*, Sept. 1994, vol. 42, no. 9. [Online]. Available: http://euler.slu.edu/ goldwasser/courses/loyola/comp363/2003_Spring/handouts/course-info.pdf http://www.jstor.org/stable/2583667?origin=crossref http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Introduction+to+Algorithms

[17] R. Crabb, C. Tracey, A. Puranik, and J. Davis, "Real-time foreground segmentation via range and color imaging," in *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops*, 2008.

[18] R. Cucchiara, A. Prati, and R. Vezzani, "A multi-camera vision system for fall detection and alarm generation," *Expert Systems*, vol. 24, no. 5, pp. 334–345, 2007.

[19] M. Dantone, J. Gall, and C. Leistner, "Human Pose Estimation Using Body Parts Dependent Joint Regressors," in *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, 2013, pp. 3041–3048. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6619235%*backslash*npapers3:/

[20] B. G. Dargent-Molina P, "Épidémiologie Des Chutes Et Des Traumatismes Liés Aux Chutes Chez Les Personnes Âgées," *Revue Epidémiologique de Santé Publ ique*, vol. 43, no. 1, pp. 72–83, 1995. [Online]. Available: http://cat.inist.fr/?aModele=afficheN&cpsidt=3429265

[21] J. Deutscher, a. Blake, and I. Reid, "Articulated body motion capture by annealed particle filtering," *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No.PR00662)*, vol. 2, 2000.

[22] J. Deutscher and I. Reid, "Articulated body motion capture by stochastic search," *International Journal of Computer Vision*, vol. 61, no. 2, pp. 185–205, 2005.

[23] Y. Dhome, N. Tronson, A. Vacavant, T. Chateau, C. Gabard, Y. Goyat, and D. Gruyer, "A benchmark for background subtraction algorithms in monocular vision: A comparative study," in *2010 2nd International Conference on Image Processing Theory, Tools and Applications, IPTA 2010*, vol. 2, no. 2. Ieee, July 2010, pp. 66–71. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5586792

[24] G. Diraco, a. Leone, and P. Siciliano, "An active vision system for fall detection and posture recognition in elderly healthcare," *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2010*, no. June 2015, 2010.

[25] E. J. Fernandez-Sanchez, J. Diaz, and E. Ros, "Background subtraction based on color and depth using active sensors." *Sensors (Basel, Switzerland)*, vol. 13, no. 7, pp. 8895–8915, Jan. 2013. [Online]. Available: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3758628&tool=pmcentrez&rendertype=abstrac

[26] J. Gall and V. Lempitsky, "Class-specific hough forests for object detection," in *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2009*, 2009, pp. 1022–1029.

[27] R. Girshick, J. Shotton, P. Kohli, A. Criminisi, and A. Fitzgibbon, "Efficient regression of general-activity human poses from depth images," *Proceedings of the IEEE International Conference on Computer Vision*, pp. 415–422, Nov. 2011. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6126270

[28] D. Grest, J. Woetzel, and R. Koch, "Nonlinear Body Pose Estimation from Depth Images," in *Pattern Recognition*, 2005, pp. 285–292. [Online]. Available: http://www.springerlink.com/content/319v00jj6t4yj8pv

[29] V. Gulshan, V. Lempitsky, and A. Zisserman, "Humanising GrabCut: Learning to segment humans using the Kinect," *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1127–1133, Nov. 2011. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6130376

[30] K. Hara and R. Chellappa, "Computationally efficient regression on a dependency graph for human pose estimation," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Ieee, June 2013, pp. 3390–3397. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6619279

[31] M. Harville, G. Gordon, and J. Woodfill, "Foreground segmentation using adaptive mixture models in color and depth," *Proceedings IEEE Workshop on Detection and Recognition of Events in Video*, pp. 3–11, 2001. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=938860 http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=938860

[32] Y. Ivanov, A. Bobick, and J. Liu, "Fast lighting independent background subtraction," *International Journal of Computer Vision*, vol. 37, no. 2, pp. 199–207, 2000.

[33] O. H. Jafari, D. Mitzel, and B. Leibe, "Real-Time RGB-D based People Detection and Tracking for Mobile Robots and Head-Worn Cameras," 2014, pp. 5636–5643.

[34] K. Kim, T. H. Chalidabhongse, D. Harwood, and L. Davis, "Real-time foreground-background segmentation using codebook model," *Real-Time Imaging*, vol. 11, no. 3, pp. 172–185, 2005.

[35] P. Kohli, J. Rihan, M. Bray, and P. H. S. Torr, "Simultaneous segmentation and pose estimation of humans using dynamic graph cuts," *International Journal of Computer Vision*, vol. 79, no. 3, pp. 285–298, 2008.

[36] V. Kolmogorov, A. Criminisi, A. Blake, G. Cross, and C. Rother, "Bi-layer segmentation of binocular stereo video," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 2, 2005.

[37] P. Kontschieder, P. Kohli, J. Shotton, and A. Criminisi, "GeoF: Geodesic forests for learning coupled predictors," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Ieee, June 2013, pp. 65–72. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6618860

[38] L. Ladicky, P. H. Torr, and A. Zisserman, "Human Pose Estimation Using a Joint Pixel-wise and Part-wise Formulation," *2013 IEEE Conference on Computer Vision and Pattern Recognition*, no. c, pp. 3578–3585, June 2013. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6619303

[39] C. Leistner, A. Saffari, J. Santner, and H. Bischof, "Semi-supervised random forests," in *Proceedings of the IEEE International Conference on Computer Vision*, 2009, pp. 506–513.

[40] X. Man, P. Yuan, and Y. YANG, "HAU3D11_Xia," *Journal of Computational Information ...*, 2011. [Online]. Available: http://www.jofcis.com/publishedpapers/2011_7_3_807_815.pdf

[41] G. Mastorakis and D. Makris, "Fall detection system using Kinect's infrared sensor," pp. 1–12, 2012.

[42] Microsoft, "Kinect." [Online]. Available: http://www.xbox.com/en-US/kinect

[43] T. B. Moeslund, A. Hilton, and V. Krüger, "A survey of advances in vision-based human motion capture and analysis," pp. 90–126, 2006.

[44] M. Munaro and E. Menegatti, "Fast RGB-D people tracking for service robots," *Autonomous Robots*, vol. 37, no. 3, pp. 227–242, 2014.

[45] OpenCV, "OpenCV." [Online]. Available: http://opencv.org/

[46] OpenNI, "OpenNI." [Online]. Available: http://www.openni.org

[47] P. Peursum, S. Venkatesh, and G. West, "A study on smoothing for particle-filtered 3D human body tracking," *International Journal of Computer Vision*, vol. 87, no. 1-2, pp. 53–74, 2010.

[48] C. Plagemann, V. Ganapathi, D. Koller, and S. Thrun, "Real-time identification and localization of body parts from depth images," in *Proceedings - IEEE International Conference on Robotics and Automation*. Ieee, May 2010, pp. 3108–3113. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5509559

[49] C. Rougier and J. Meunier, "Demo: Fall Detection Using 3D Head Trajectory Extracted From a Single Camera Video Sequence," *In Proceedings of the Journal of Telemedicine and Telecare*, vol. 11, no. 4, p. 2, 2005.

[50] C. Rougier, J. Meunier, A. St-Arnaud, and J. Rousseau, "Robust video surveillance for fall detection based on human shape deformation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 21, no. 5, pp. 611–622, May 2011. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5733403

[51] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, "Real-time human pose recognition in parts from single depth images," *Studies in Computational Intelligence*, vol. 411, pp. 119–135, June 2013. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5995316

[52] J. Shotton, R. Girshick, A. Fitzgibbon, T. Sharp, M. Cook, M. Finocchio, R. Moore, P. Kohli, A. Criminisi, A. Kipman, and A. Blake, "Efficient human pose estimation from single depth images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 12, pp. 2821–2840, Dec. 2013. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/24136424

[53] M. Siddiqui and G. Medioni, "Human pose estimation from a single view point, real-time range sensor," *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*, pp. 1–8, 2010. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5543618

[54] C. Sminchisescu and B. Triggs, "Covariance scaled sampling for monocular 3D body tracking," *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, vol. 1, 2001.

[55] M. Sun, P. Kohli, and J. Shotton, "Conditional regression forests for human pose estimation," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Ieee, June 2012, pp. 3394–3401. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6248079

[56] J. Taylor, J. Shotton, T. Sharp, and A. Fitzgibbon, "The Vitruvian manifold: Inferring dense correspondences for one-shot human pose estimation," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2012.

[57] M. Tenorth, J. Bandouch, and M. Beetz, "The TUM kitchen data set of everyday manipulation activities for motion tracking and action recognition," in *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops 2009*, 2009, pp. 1089–1096.

[58] R. Therón, V. Moreno, B. Curto, and F. J. Blanco, "Posture constraints for Bayesian human motion tracking," *AMDO*, vol. 4069, no. 4th Conference on Articulated Motion and Deformable Objects, pp. 414–423, 2006. [Online]. Available: http://hdl.handle.net/10261/30364

[59] J. Thomas, P. E. Jouve, and N. Nicoloyannis, "Optimisation and evaluation of random forests for imbalanced datasets," *Foundations of Intelligent Systems*, pp. 622–631, 2006. [Online]. Available: http://www.springerlink.com/content/v2x2013145v81985/

[60] M. Ye, H. Wang, N. Deng, X. Yang, and R. Yang, "Real-time Human Pose and Shape Estimation for Virtual Try-On Using a Single Commodity Depth Camera." *IEEE transactions on visualization and computer graphics*, vol. 20, no. 4, pp. 550–9, 2014. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/24650982

[61] T. H. Yu, T. K. Kim, and R. Cipolla, "Unconstrained monocular 3D human pose estimation by action detection and cross-modality regression forest," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2013, pp. 3642–3649.

[62] J. Zhu, M. Liao, R. Yang, and Z. Pan, "Joint depth and alpha matte optimization via fusion of stereo and time-of-flight sensor," in *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2009*, 2009, pp. 453–460.

[63] Y. Zhu and K. Fujimura, "Constrained Optimization for Human Pose Estimation from Depth Sequences," in *Computer Vision âĂŞ ACCV 2007*, 2007, vol. 4843, pp. 408 – 418. [Online]. Available: http://www.springerlink.com/index/10.1007/978-3-540-76386-4

[64] ——, "A bayesian framework for human body pose tracking from depth image sequences," *Sensors*, vol. 10, no. 5, pp. 5280–5293, 2010. [Online]. Available: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3292173&tool=pmcentrez&renderty

[65] W. G. Zijlstra, A. Buursma, and W. P. Meeuwsen-van der Roest, "Absorption spectra of human fetal and adult oxyhemoglobin, de-oxyhemoglobin, carboxyhemoglobin, and methemoglobin," *Clinical Chemistry*, vol. 37, no. 9, pp. 1633–1638, 1991. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=784637