Capture de mouvements humains par capteurs RGB-D

Jean-Thomas Masse

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par l'Université Toulouse III - Paul Sabatier
Discipline ou spécialité : Robotique et informatique

Présentée et soutenue par Jean-Thomas Masse
Le 25 Septembre 2015

Titre : Capture de mouvements humains par capteurs RGB-D

JURY
- Jenny Benois-Pineau - Rapporteur
- Sylvie Treuillet - Rapporteur
- Guillaume Oller - Examinateur
- Mohamed Daoudi – Examinateur – Président du Jury
- Frédéric Lerasle - Directeur de thèse
- Michel Devy - Directeur de thèse

École doctorale : EDSYS
Unité de recherche : LAAS-CNRS (Groupe RAP)
Directeur(s) de Thèse : Frédéric LERASLE, Michel DEVY

Résumé

L'arrivée simultanée de capteurs de profondeur et couleur, et d'algorithmes de détection de squelettes super-temps-réel a conduit à un regain de la recherche sur la capture de mouvements humains. Cette fonctionnalité constitue un point clé de la communication Homme-Machine. Mais le contexte d'application de ces dernières avancées est l'interaction volontaire et fronto-parallèle, ce qui permet certaines approximations et requiert un positionnement spécifique des capteurs. Dans cette thèse, nous présentons une approche multi-capteurs, conçue pour améliorer la robustesse et la précision du positionnement des articulations de l'homme, et fondée sur un processus de lissage trajectoriel par intégration temporelle, et le filtrage, des squelettes détectés par chaque capteur. L'approche est testée sur une base de données nouvelle acquise spécifiquement, avec une méthodologie d'étalonnage adaptée spécialement. Un début d'extension à la perception jointe avec du contexte, ici des objets, est proposée.

Abstract

Simultaneous apparition of depth and color sensors and super-realtime skeleton detection algorithms led to a surge of new research in Human Motion Capture. This feature is a key part of Human-Machine Interaction. But the applicative context of those new technologies is voluntary, fronto-parallel interaction with the sensor, which allowed the designers certain approximations and requires a specific sensor placement. In this thesis, we present a multi-sensor approach, designed to improve robustness and accuracy of a human's joints positionning, and based on a trajectory smoothing process by temporal integration, and filtering of the skeletons detected in each sensor. The approach has been tested on a new specially constituted database, with a specifically adapted calibration methodology. We also began extending the approach to context-based improvements, with object perception being proposed.

Sommaire

Introduction

La capture de mouvements humains, ou Motion Capture, consiste, à partir des données de capteurs, à inférer les positions 3D des parties corporelles dans le temps. Cette capture peut être exploitée dans toutes sortes d'applications :

- Le divertissement, permettant aux données ainsi récupérées d'animer un avatar virtuel. À la fois le cinéma d'animation et les jeux interactifs entrent dans cette catégorie ;
- La surveillance : raffinement extrême de la détection de personne, la posture complète permet d'induire des informations supplémentaires telles que l'intentionnalité. Pour la surveillance de personnes âgées, il s'agira de détecter des postures dangereuses (chutes) ou de déduire des mesures sur l'actimétrie de la personne. Pour la vidéo-surveillance dans des lieux publics, il s'agira de détecter des comportements anormaux: ce contexte est plus complexe puisqu'il nécessite généralement de capturer les mouvements de plusieurs personnes dans la scène ;
- La robotique, la capture de mouvements permet au système de réagir au service de la personne dans des buts très variés. Il s'agira par exemple de détecter des gestes de la personne, gestes exploités pour l'interaction entre la personne et le système robotique: ce type d'interaction est indispensable pour la réalisation de tâches conjointes Homme-Robot avec des robots collaboratifs (ou cobots) tels que les bras KUKA LWR ou IIWA. Même sans interaction, l'apprentissage de tâches par démonstration pour un rejeu par un robot nécessite aussi cette modalité.

La capture de mouvements est une opération, en tout cas en cinématographie, généralement menée grâce à un système de caméras détectant des marqueurs posés sur l'utilisateur. C'est la technique communément désignée quand on utilise l'expression Motion Capture, ou MoCap. Nous disposons au LAAS d'un tel système (commercialisé par Motion Analysis (Maloney, Movement Analysis Products, 2013), avec une dizaine de caméras infrarouges) ; cette technologie est très onéreuse, et requiert une phase d'installation lourde. Par contre les marqueurs sont localisés avec une grande précision de l'ordre de quelques millimètres.

Du fait de ces contraintes de coût et d'installation, il est intéressant de s'affranchir des marqueurs, en faisant de la capture de mouvements par simple traitement d'images. Comme le cerveau humain, une image contient suffisamment d'informations pour retrouver la posture humaine, cependant au prix d'une grande complexité des calculs, et sou-

vent des baisses de performances lorsque la posture du sujet cache la vision de certains de ses membres. Il est possible de combiner plusieurs caméras pour remédier à cela.

(a) *(b)* *(c)*

Figure 1 Techniques de motion capture.

a : Système commercial de capture de mouvement avec marqueurs.
b : Capture mono-caméra couleur (Agarwal & Triggs, Monocular Human Motion Capture with a Mixture of Regressors, 2005).
c : Capture Kinect (Microsoft) (image de GamerHub.tv).

Dans les années 2000, sont apparues les caméras optiques à temps de vol (abrégé en TOF pour Time Of Flight), technologie aussi appelée PMD (Photonic Mixer Device), qui permettent d'acquérir des images 3D, ainsi qu'une image de réflectance souvent très bruitée. Par exemple, les capteurs SWISS RANGER® réalisés initialement au CSEM en Suisse, qui sont maintenant commercialement exploités par la spin-off Heptagon (propriétaire de MESA Imaging). Ce type de capteur a été exploité pour la capture du mouvement, notamment dans le contexte du projet européen COGNIRON sur le robot compagnon. Mais le coût très élevé de ces caméras en a limité initialement la diffusion. Notons que cette technologie permet maintenant de produire des capteurs bas coût: citons la Kinect® 2, mais aussi les capteurs produits par l'entreprise pmdtechnologies gmbH en Allemagne. Le très médiatisé projet Tango® de Google en partenariat avec pmdtec vise à intégrer de telles caméras dans nos smart phones.

En 2010, des capteurs de profondeur bon marché sont apparus sur le marché du divertissement : le capteur Kinect de Microsoft en 2010 et sa deuxième version présentée en 2014. Ces capteurs sont souvent appelés 'capteur RGBD', car ils donnent à la fois une image de profondeur (Depth) et une image couleur, les deux étant superposés (on peut généralement négliger les erreurs de calibrage). Cette commercialisation est accompagnée d'une méthode de capture de mouvements pour l'utiliser comme périphérique des produits de divertissement de la marque. Les trois systèmes sensoriels sont illustrés Figure 1.

Les systèmes commerciaux de capture de mouvements (dont fait partie le système utilisé dans notre laboratoire, un produit de Motion Analysis (Maloney, Movement

Analysis Products, 2013)) permettent en général d'effectuer des prises de vues différées : la donnée est traitée et exploitée ultérieurement, possiblement plusieurs fois. Dans un contexte robotique, cela peut être utilisé pour acquérir les mouvements d'un opérateur, puis pour faire exécuter ce même mouvement par un robot intéractif : cet apprentissage hors ligne de mouvements a été exploité hors ligne par exemple, pour réaliser des démonstrations bien popularisées au Japon, sur le robot humanoïde Asimov exécutant des mouvements synchrones avec une danseuse. Nos collègues au LAAS ont réalisé une démonstration du même type à la Novela 2012 de Toulouse, le robot HRP2 réalisant les mêmes mouvements qu'un danseur hip-hop.

De tels systèmes ne sont pas exploitables pour des systèmes interactifs ou de surveillance. De plus le système doit généralement être adapté à la morphologie spécifique d'un utilisateur. Et même si récemment la technologie a permis la capture temps-réel pour les besoins des acteurs du cinéma, le besoin d'instrumentalisation de l'acteur reste un handicap pour une utilisation naturelle et usuelle de la technologie. Ces constats ont motivé nos premières investigations sur la capture de mouvements multi-Kinects, une solution *a priori* bas coût, simple d'utilisation car dépourvue d'instrumentation, qui démocratiserait les systèmes de capture de mouvements.

Les capteurs vidéo, bien que moins directement adaptés à la motion capture, ont été la source des nombreux travaux sur la perception de l'homme. Ces méthodes exploitent généralement un ensemble de caméras : le corps de la personne suivie est extrait par une segmentation fond-forme, ce qui rend ces approches très dépendantes de l'apparence de la personne et du fond. Souvent dans les expérimentations, les personnes sont habillées en sombre, et se déplacent dans une pièce peinte en blanc, ce qui permet d'extraire une silhouette "propre" de la personne suivie.

Mais peu exploitent les percepts liés au contexte courant de prise de vue pour robustifier cette perception de l'homme. Les éléments de la scène qui peuvent occulter l'utilisateur ou qui peuvent faire perdre le suivi du mouvement humain, peuvent aussi apporter de l'information de façon indirecte si on a les bonnes connaissances *a priori.* Une contrainte de perception peut a contrario contraindre et donc aider le suivi, et c'est l'intuition qui nous a poussés à exploiter de la perception d'objets en parallèle de la perception de l'homme, afin que la combinaison de ces deux percepts souvent considérés séparément dans la littérature leur soit mutuellement bénéfique.

Figure 2 Capteurs de profondeur à lumière structurée.

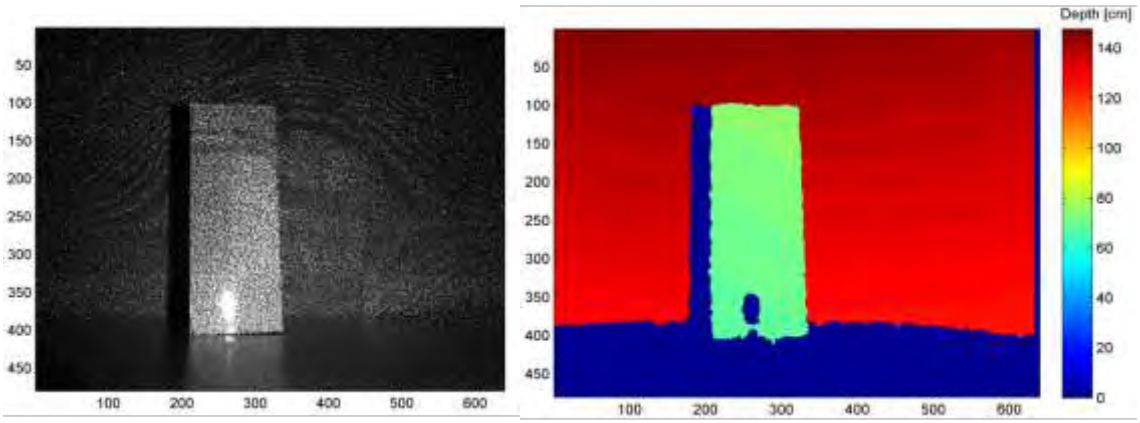
À gauche, Kinect® de Microsoft. À droite, Xtion® Pro Live, de Asus

Les capteurs RGBD bas coût de type Kinect, comme illustrés par la Figure 2, offrent de nouvelles possibilités. Même si on se limite toujours à un seul point de vue, ce

capteur apporte cependant une donnée plus riche que la silhouette (parfois difficile à obtenir, car ambigüe) qui est généralement extraite à partir des images acquises par des capteurs vidéo classiques. Ces capteurs s'inspirent du principe des capteurs à lumière structurée. Un projecteur infrarouge émet un mouchetis connu *a priori,* obtenu par diffraction d'un laser à travers une grille pseudo aléatoire et un élément de convolution. L'image infrarouge du mouchetis projeté sur la scène, est acquise par une caméra dotée d'un filtre infrarouge, décalée par rapport au projecteur: la base est d'environ 7 à 8 cm. Par décorrélation locale du motif projeté et calcul de l'intensité de chaque point, on obtient la distance au capteur pour toute la zone image appartenant à ce point. Ce principe, résumé dans les brevets (Garcia & Zalevsky, 2008) et (Shpunt & Zalevsky, 2011), semble être celui utilisé par les capteurs Kinect et assimilés. Une image du mouchetis, et l'image des valeurs de profondeur associées, sont illustrées par la Figure 3. Par ailleurs, une image RGB est acquise par une caméra couleur classique positionnée à côté de la caméra infrarouge avec une base d'environ 2 à 3 cm ; un étalonnage en usine permet de recaler les deux images infrarouge et couleur.

1. Contexte de la thèse

Cette thèse a été initiée avec le projet PRACE (pour « Productive Robot ApprentiCE ») : mon employeur, Magellium, collaborait au sein d'un consortium d'entreprises européennes. En parallèle de ce projet européen, il a bénéficié du dispositif CIFRE de l'ANRT, convention passée avec l'équipe Robotique, Action et Perception du LAAS-CNRS. Le but de l'entreprise était l'élaboration à des fins manufacturières d'un système robotique manipulateur d'objets variables, comme ceux listés dans la Figure 4, entraînable tel un apprenti par un utilisateur non expert en robotique, au contraire de la robotique industrielle actuelle. Cet objectif d'apprentissage de tâches de manipulation à partir de démonstrations faites par un opérateur et observées par notre plateforme multi-capteurs RGB-D, requiert donc une approche adaptée de capture de mouvements humains.



(a) *(b)*

Figure 3 Mouchetis infrarouge de Kinect (a) et image de profondeur (b) correspondante (Khoshelham & Elberink, 2012).

Cette thèse a donc eu une forte connotation ingénierie. Tous les détails sur le projet, et les développements implémentant l'approche décrite dans ce manuscrit, sont dans le Chapitre VI: Ingénierie.

(a) (b) (c)

Figure 4 Quelques exemples de robots manipulateurs.

(a) Baxter, de Rethink Robotics (http://www.rethinkrobotics.com/products/baxter/, photo de Flickr). (b) PR-2, de Willow Garage (www.willowgarage.com/pages/pr2/overview, photo de Wikimedia). (c) Frida, de ABB (http://www.abb.com/cawp/abbzh254/8657f5e05ede6ac5c1257861002c8ed2.aspx).

2. Problématique et cahier des charges

Notre problématique se borne en apparence à estimer la position et l'orientation des parties corporelles du corps humain, ou les positions 3D des extrémités de ceux-ci. Mais les challenges à relever sont de le faire pour une grande variabilité de morphologies et d'apparences, et que le système soit suffisamment peu encombrant et sans instrumentation du sujet pour permettre un usage naturel pour l'utilisateur et l'éventuelle intégration dans un système robotique. Nous devons trouver le meilleur compromis entre les performances du système (précision et robustesse) et l'efficacité (coût CPU qui influe sur la fréquence de traitement des images), afin d'obtenir les données en temps réel, et laisser suffisamment de capacité de calcul au reste des processus graphiques ou algorithmiques.

Un des problèmes connus des capteurs RGBD et surtout, des techniques de capture de mouvements fournis avec la Kinect, est la grande sensibilité à la position de la personne devant le capteur. Si la personne ne fait pas face au capteur, les résultats sont généralement détériorés. Aussi nous proposons d'exploiter plusieurs capteurs RGBD afin de couvrir un champ de vue plus large, et surtout, afin d'avoir un suivi des mouvements de la personne même si elle change d'attitude pendant la séquence.

Nous pouvons dresser ainsi un cahier des charges de notre approche de reconstruction 3D de la posture humaine :

1. Contrainte temps réel.

2. Un seul sujet observé.
3. Plateforme multi-capteurs RGB-D à forts champs joints, synchronisés et étalonnés, sans marqueurs sur le sujet observé.
4. Couplé à un système commercial de capture de mouvements (pour la vérité terrain uniquement).

3. Notre approche

Plusieurs constats ont motivé nos travaux.

Premièrement, l'arrivée récente de nouvelles technologies, comme les capteurs de profondeur type Kinect® (deux exemples en Figure 2, dont le capteur susnommé de Microsoft et le Xtion® de ASUS), et les travaux associés de (Shotton, et al., 2011), dont les caractéristiques s'approchent des besoins de notre problématique (rapidité, robustesse par rapport à l'apparence des personnes suivies…). Nous utiliserons les deux noms Kinect® et Xtion® de manière équivalente car leurs caractéristiques, à l'exception d'une motorisation de l'angle vertical et du nombre de microphones, tous les deux non considérés dans nos travaux, sont identiques.

Deuxièmement, en remarquant que le Kinect fournit une donnée certes de profondeur, mais sous forme d'une matrice 2D comme une caméra vidéo classique, nous avons exploré la littérature sur le suivi de mouvements dans un flux vidéo. En guise de plus ample préambule, une synthèse fait l'objet du premier chapitre de ce mémoire.

Conséquence de ces constats et de nos propres expérimentations, ce mémoire présente une approche multi-capteurs, mais fusionnant les données squelettes détectées dans chaque point de vue séparément. Un lissage trajectoriel permet de lever les ambigüités, géométriques en recoupant dans tous les points de vue, et dynamiques en considérant les trajectoires des articulations.

Toutes les questions que l'on peut se poser, telles que comment valider l'approche, ses caractéristiques et ses choix, alors devraient trouver réponses dans les chapitres suivants.

4. Plan du mémoire

Ce mémoire est structuré en 6 chapitres, clôturé par les conclusions et perspectives générales.

Le premier chapitre introduit le contexte de notre approche de capture de mouvements : état de l'art et positionnement des travaux. Le second chapitre présente quant à lui un état de l'art sur les bases publiques existantes et descriptif de nos bases de données multi-Kinects avec vérité terrain par MoCap. Le chapitre suivant décrit et formalise notre approche de capture de mouvements, ainsi que l'analyse des premiers résultats. Le chapitre 4 présente une analyse des performances cette fois sur un ensemble de configurations de 2 à 5 Kinects. Le chapitre 5 présente nos investigations préliminaires sur une perception conjointe homme-objets. Le dernier chapitre est plus technologique: d'une part il donne le contexte « industriel » dans lequel nous avons effectué ces travaux en tant

que doctorant CIFRE. Nous décrirons rapidement le projet PRACE et nos contributions techniques réalisés dans le cadre de ce projet. Ce sera l'occasion de préciser l'environnement technique dans lequel nous avons produit les logiciels implémentant les algorithmes décrits dans les chapitres précédents.

Le mémoire se termine sur une partie conclusions et perspectives où nous allons nous projeter dans l'avenir des possibles extensions aux travaux présentés.

Chapitre I: Capture de mouvements humains : techniques existantes en Vision par Ordinateur

Introduction

Le capteur Kinect est un capteur de distance, dans la droite lignée des caméras actives à temps de vol. Son exploitation pour la capture de mouvements humains a été initiée par Microsoft, mais il a attiré aussi la curiosité dans les domaines classiques de la métrologie et l'étude spatiale. Au-delà des travaux sur les caractéristiques du capteur lui-même (Andersen, et al., 2012), il existe des études dans des cas précis de localisation dans un bâtiment (Khoshelham & Elberink, 2012) ou de l'utilisation sur des chantiers de construction (Rafibakhsh, Gong, Siddiqui, Gordon, & Lee, 2012), qui mettent en œuvre un unique Kinect. Aussi, les solutions multi-Kinects sont principalement pour des environnements non-humains (Berger, 2013) et dans l'écrasante majorité sans vérité terrain associée à l'homme.

À ce stade nous pensons qu'il est important pour justifier nos choix méthodologiques et technologiques de nous positionner par rapport à l'existant, d'une part pour asseoir l'originalité de notre approche, et d'autre part car cela a orienté notre recherche dans une ou plusieurs directions alors jugées pertinentes.

Le problème de la perception des mouvements humains passe historiquement par les approches basées vision exclusivement. Le survey de (Moeslund, Hilton, & Krüger, 2006) montre la richesse et l'intérêt de cette problématique pour la communauté Vision par Ordinateur.

Notre approche se démarque des approches existantes sur deux aspects essentiels : la stratégie de filtrage mise en œuvre, et la gestion intelligente de reconstruction de postures émanant simultanément de plusieurs Kinects, mais chaque aspect est présenté ci-dessous et notre approche est positionnée relativement à la littérature.

A. Approches bas-niveau vs. haut-niveau

Une première distinction peut être faite en considérant la nature des données traitées. Dans le suivi vidéo on considère généralement le flux vidéo sous forme de séquence d'images prises en rafale à intervalle constant. Cependant on peut d'abord exploiter une segmentation des silhouettes vis-à-vis de l'arrière-plan, l'approche devient alors tributaire, mais aussi agnostique de la segmentation sous-jacente, qui peut être réalisée de plusieurs manières. De ces deux manières d'aborder une même problématique de capture de mou-

vements, nous pouvons donc distinguer les approches bas-niveau et haut-niveau, selon la nature des percepts considérés : sensoriels ou symboliques.

Les approches originelles de (Shotton, et al., 2011) et (Girshick, Shotton, Kohli, Criminisi, & Fitzgibbon, 2011) utilisent l'image de profondeur, ou carte de disparité, donnée par le capteur Kinect®. La valeur de chaque pixel n'est pas une couleur ou une intensité lumineuse, mais la distance mesurée à l'objet le plus proche dans ce rayon passant par le centre optique du capteur. L'image de profondeur est d'abord segmentée en zones correspondant à un utilisateur. Le principe de l'estimation subséquente de squelette doit donc permettre d'estimer les positions des membres à l'intérieur de cette zone à partir de la donnée de profondeur.

Nous allons détailler ici 4 approches d'intérêt extraites de la littérature utilisant le Kinect® ou équivalent. Leurs caractéristiques seront cependant aussi analysées plus tard dans les autres classifications présentées dans ce chapitre.

1. Approche initiale Microsoft mono-Kinect® par labellisation des pixels

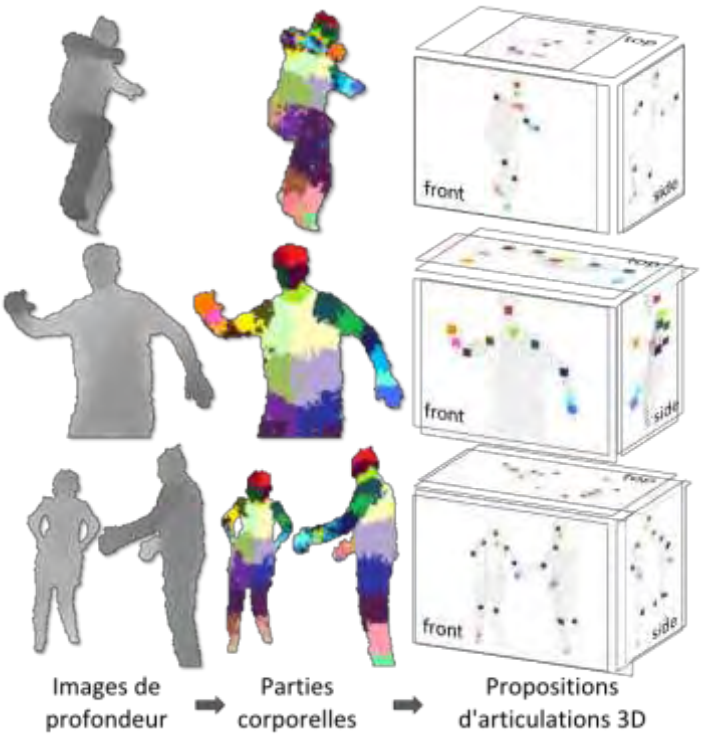


Figure 5 Approche utilisée dans le capteur Kinect® telle que décrite dans (Shotton, et al., 2011)

Cette approche initée par (Shotton, et al., 2011) est illustrée Figure 5. Pour obtenir la labellisation en parties corporelles, une approche par apprentissage type « *Random Forest* » a été utilisée. Ce type de meta-classifieur est constitué de multiples classifieurs en arbre. Dans la droite lignée de l'approche en cascade de (Viola & Jones, 2001), historique introduction du Machine Learning dans la détection de parties corporelles, ils utilisent plutôt ici une structure en arbre car il ne s'agit pas seulement d'une détection (autrement dit, classification binaire) mais aussi de classifier chaque pixel. Le problème de

détection de visages présente un faible nombre de positifs, ce qui favorisait nettement la structure en cascade.

Ensuite, pour déduire la position 3D, un dérivé de l'algorithme *mean-shift* est utilisé pour obtenir la position image des centres des articulations, puis la valeur de profondeur à cet endroit, décalée d'un déplacement spécifique à l'articulation, est utilisée pour déterminer la profondeur réelle de l'articulation, déterminant pleinement la position de l'articulation.

2. Seconde approche Microsoft Research mono-Kinect® par régression

Cette approche plus récente initiée par (Girshick, et al., 2011), est illustrée Figure 6. C'est une variante qui requiert moins de données d'apprentissage que la précédente et donne de meilleurs résultats, notamment avec les membres occultés. Ceci se justifie car l'approche a été entraînée pour faire des propositions de position 3D (et non de labellisation de partie corporelle) pour toute articulation, même occultée. La figure illustre bien ce fait avec les occultations de l'épaule droite et du genou gauche.

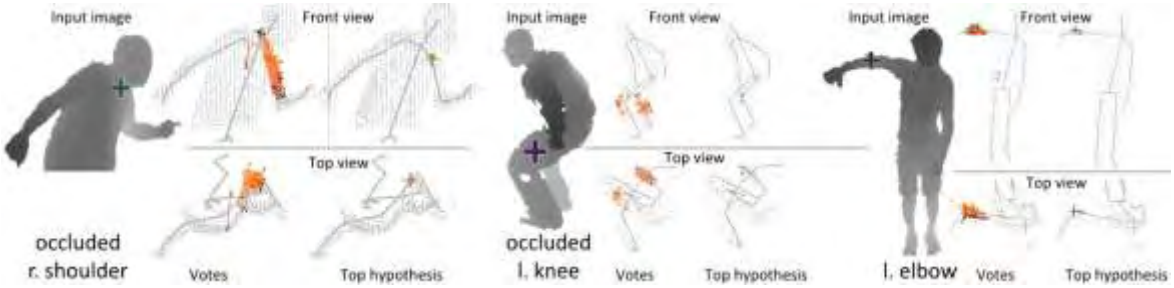


Figure 6 Approche alternative par Kinect® telle que décrite dans (Girshick, Shotton, Kohli, Criminisi, & Fitzgibbon, 2011).

L'analyse des performances d'OpenNI/NiTE ne montre pas un tel comportement des membres occultés. Comme (Shotton, et al., 2011) aucune information n'est donnée. Une fonctionnalité logicielle renvoie cependant une position neutre (membre tendu à la verticale du buste) mais nous l'avons désactivée. Nous supposons donc que nous utilisons (Shotton, et al., 2011) en utilisant OpenNI/NiTE, mais notre approche pourrait se coupler indifféremment aux deux.

3. Travaux préliminaires de fusion multi-capteurs bas-niveau

Nos investigations très préliminaires du multi-capteur s'inscrivaient elles aussi dans la catégorie des approches bas-niveau. Dans (Filali, Masse, Lerasle, Boizard, & Devy, 2013) le principe est de voxelliser l'espace commun aux Kinects, puis définir des caractéristiques sur les voxels pour labelliser ceux-ci en parties corporelles. Nous utilisons du Machine Learning afin de parvenir à ce résultat, puis un Mean-Shift est effectué pour localiser le centre des articulations en 3D. La Figure 7 synthétise l'approche par un schéma-bloc.

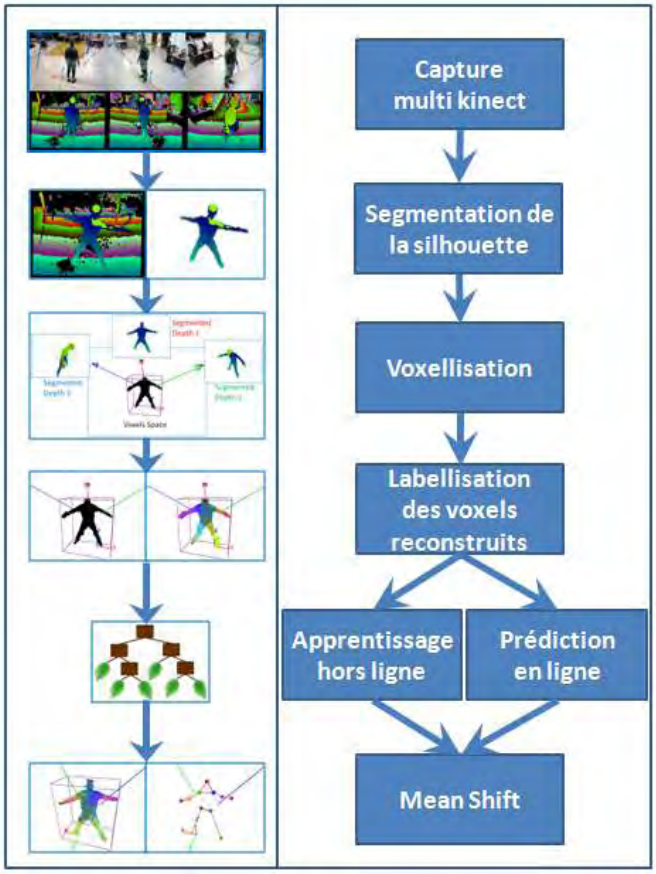


Figure 7 Approche multi-capteurs de (Filali, Masse, Lerasle, Boizard, & Devy, 2013).

4. Autre approche de fusion multi-capteurs

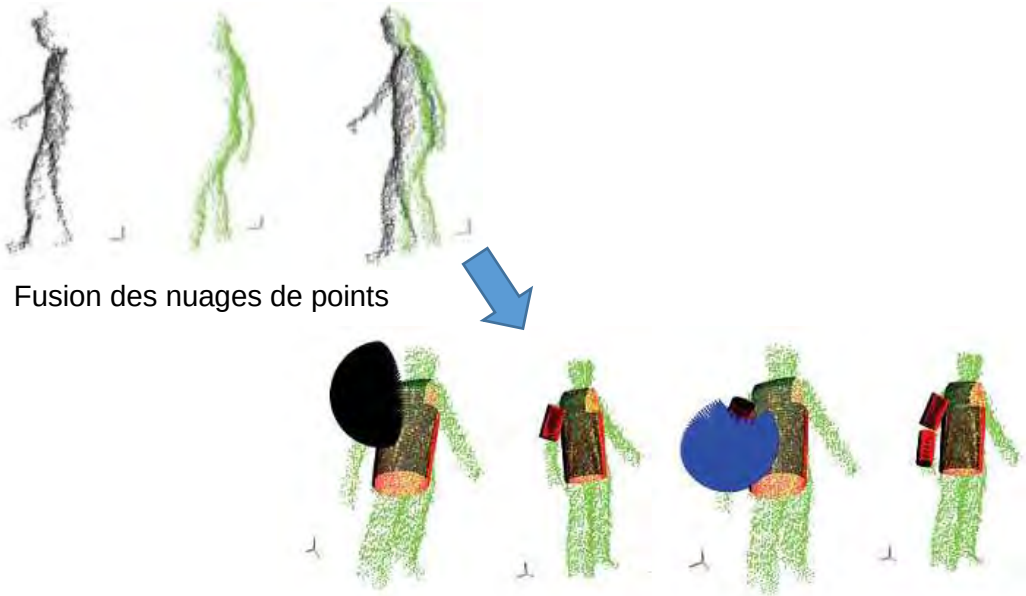


Figure 8 Approche multi-capteurs de (Zhang, Sturm, Cremers, & Lee, 2012).

Dans (Zhang, Sturm, Cremers, & Lee, 2012) on considère les données de profondeur sous la forme de nuages de points. Ici, chaque image de profondeur est transformée en un nuage de points, qui couvrent l'intégralité du corps humain s'il y a suffisamment de capteurs. Ensuite, la configuration spatiale d'un ensemble d'enveloppes représentant un modèle de corps humain, est optimisée afin de mieux coller à ces nuages de points. On déduit de la configuration de ce modèle est la pose de l'utilisateur. La Figure 8 illustre ces deux étapes.

5. Positionnement de notre approche

Nous privilégions une approche multi-Kinect mais sans fusionner les données au niveau bas. En effet, nous utilisons un détecteur par capteur qui infère la posture indépendamment des autres capteurs. En l'espèce, ce détecteur est OpenNI/NiTE. Comme nous l'avons vu précédemment, une approche bas-niveau exploiterait directement et simultanément les données de profondeurs de tous les capteurs et formerait sa propre estimation de la posture humaine. Notre approche est donc une approche haut-niveau, et pourra avantageusement tirer parti des performances de toute méthode robuste de reconstruction de postures, OpenNI ou autre, dans un futur immédiat.

B. Directe vs. indirecte

On peut noter également deux grands types d'approches pour l'estimation du mouvement humain : les approches basées détection, ou directe, et les approches génératives, ou indirectes.

Le premier type infère l'estimation (dans notre problématique, la posture 3D) directement de l'observation. Cette évaluation s'effectue éventuellement en plusieurs étapes de plus en plus précises (méthode dite *top-down*) ou par morceaux du tout estimé (*bottom-up*). Au rang de ces approches, on peut citer ici certaines que nous avons déjà abordées au point précédent : (Shotton, et al., 2011), (Girshick, Shotton, Kohli, Criminisi, & Fitzgibbon, 2011) et (Filali, Masse, Lerasle, Boizard, & Devy, 2013). Dans la littérature vidéo, on peut citer (Agarwal & Triggs, 2006), (Poppe & Poel, 2006), (Gond, Sayd, Chateau, & Dhome, 2009), (Corazza, Mündermann, Gambaretto, Ferrigno, & Andriacchi, 2010) et (Zhuang, Zhao, Ahmad, Chen, & Low, 2012).

Les méthodes indirectes, ou génératives, se basent sur un modèle « d'apparence » *a priori*, donc ici un modèle de corps humain, qui sert à recaler l'estimation sur les données d'entrée. Ce modèle est dit génératif, car il peut générer la partie de ces données, image, ou carte de disparité, correspondant à l'état estimé. On mesure ensuite dans l'espace des données d'entrée la cohérence de cette apparence générée avec les données observées. La méthode de (Zhang, Sturm, Cremers, & Lee, 2012) rentre parfaitement dans cette catégorie en cela que leur modèle humain est une surface dont le nuage de point constitue un échantillonnage, ils comparent donc bien les données capteurs à leur modèle. Ces méthodes sont de nature régressive, appliquant des modifications à l'état visant à améliorer

cette cohérence, jusqu'à ce qu'un critère d'arrêt soit satisfait. L'initialisation est particulièrement longue mais les recherches subséquentes sont plus rapides.

Cependant les approches génératives sont généralement quand même plus lentes car la comparaison dans l'espace des données capteurs est assez coûteuse, surtout lors d'une recherche exploratoire dans l'espace associé au vecteur d'état. Par exemple dans (Zhang, Sturm, Cremers, & Lee, 2012) leurs performances avec deux capteurs montent à 2 Hz en CPU contrairement aux approches mono-Kinect de (Shotton, et al., 2011) qui atteignent 30 Hz sans utiliser plus qu'un cœur de CPU.

Notre approche multi-Kinects vise à améliorer les limitations observées dans l'approche mono-Kinect de (Shotton, et al., 2011), qui certes est une approche directe, mais qui fournit une segmentation silhouette (image binaire de présence/absence d'utilisateur pour chaque pixel) et une reconstruction du squelette associée. Nous utiliserons ces multiples données dérivées en provenance de chaque capteur, constituant ainsi une approche de type indirecte. Aussi, nous allons nous concentrer sur les propositions de chaque capteur, et non faire une recherche systématique dans l'espace d'état des postures comme les approches indirectes classiques, car cela constitue souvent la source de la lenteur de ces approches.

C. Des aspects filtrage et modélisation du corps humain

Les approches basées filtrages supposent un modèle cinématique, voire dynamique, du corps humain, afin de réduire les incohérences temporelles, et de faire des prédictions qui peuvent être utilisées afin d'améliorer la vitesse des algorithmes. Citons comme exemples les travaux de (Deutscher & Reid, 2005) ou (Brubaker, Fleet, & Hertzmann, 2010). Le problème de telles approches reste qu'elles ont besoin de (ré)initialisation automatique pour contrecarrer les dérives ou décrochages du filtre dans le temps.

Il apparaît donc comme pertinent de coupler reconstruction (détection) et filtrage spatio-temporel. Cette stratégie, lorsque l'inférence se borne au plan image, s'appelle « tracking-by-detection » (Okuma, Taleghani, de Freitas, Little, & Lowe, 2004) (Perez, Vermaak, & Blake, 2004). C'est pourquoi nous nous proposons de transposer et évaluer cette démarche, probante en suivi 2D, à notre problématique de capture 3D, et de l'adapter au multi-Kinect.

Il est à noter que le filtrage possède plusieurs variantes de techniques, qui n'ont cessé de se multiplier avec les années de travail passés à étudier ce type de problèmes. On peut citer classiquement les techniques dites markoviennes, dont certaines peuvent alors être d'ordre 1 (l'état actuel ne dépend que de l'état estimé à l'instant précédent et pas plus), e.g. le filtrage de type Kalman, ou le filtrage particulaire (Tenorth, Bandouch, & Beetz, 2009), (Deutscher & Reid, 2005). Il est notable que ces stratégies sont effectivement efficaces dans le contexte de suivi 2D (Pirsiavash, Ramanan, & Fowlkes, 2011), (Goyat, Chateau, & Bardet, 2010), (Yu, Medioni, & Cohen, 2007). A contrario, une stratégie de filtrage par intégration, comme la logique différée, raisonne sur plusieurs images succes-

sives, et permet d'extraire une trajectoire de meilleure cohérence spatio-temporelle (Larson & Peschon, 1966), (Viterbi, 2006).

En suivi de mouvement, les approches de type filtrage bénéficient en général d'une modélisation dynamique, ou cinématique, ou à minima géométrique, le plus approchée possible de la réalité physique du corps humain, et donc de la posture que l'on cherche à estimer. Le principe est classiquement de modéliser chaque articulation par un ou plusieurs degrés de liberté et les amplitudes associés. Par exemple, le coude qui n'a qu'un degré de liberté, et dont la valeur ne s'étend pas sur 360°. Le principe est alors de recaler le modèle dans le flux vidéo (Deutscher & Reid, 2005).

Un enjeu est clairement de s'affranchir de toute connaissance anthropomorphe, *a priori*, cinétique ou dynamique, sur la cible car un modèle compliqué rend difficile toute généricité. Nous approchons donc le problème en faisant le choix d'aucun modèle dynamique ou cinématique humain pour cette fusion de données.

Intégration temporelle

Nous considérons une approche prenant en compte non seulement l'état précédent, mais une antériorité de plusieurs instants.

La principe initié dans (Hofmann & Gavrila, 2012) compose des techniques de détection dans l'image avec une désambiguïsation temporelle afin de raffiner les hypothèses déjà classées selon leur cohérence multi-vue.

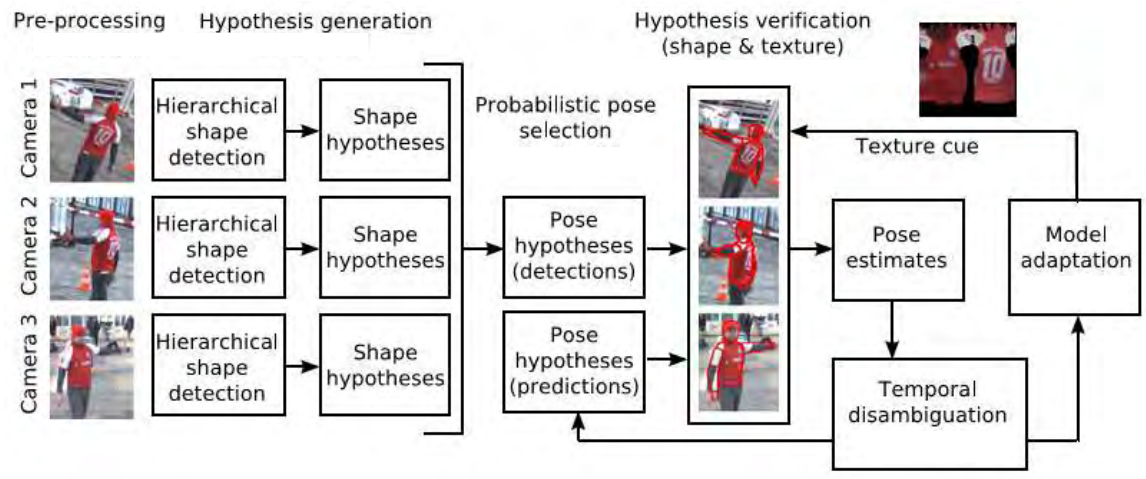


Figure 9 Résumé de l'approche de (Hofmann & Gavrila, 2012)

Chaque donnée image donne lieu à une hypothèse de forme 2D, qui combinées entre elles donnent une hypothèse de Pose 3D. Cette hypothèse de détection est mise en compétition avec une hypothèse de prédiction provenant de leur modèle dynamique utilisé dans la désambiguïsation temporelle. L'estimation est l'hypothèse correspondant le plus aux indices de forme et de texture dans l'image. Enfin, le modèle de texture est mis à jour. La Figure 9 montre en un schéma l'organisation de ce procédé complexe.

Cette approche d'intégration temporelle apporte des caractéristiques de lissage trajectoriel et ainsi exploite la continuité spatio-temporelle de notre problème.

D. De la multiplication des points de vues

La reconstruction de la posture à partir d'une vue unique présente des ambiguïtés comme le montre la Figure 10. Multiplier les vues est une manière triviale de régler le problème. De plus, les problèmes d'occultation sont traditionnellement résolus par la multiplication des points de vue capteur, comme dans (Deutscher & Reid, 2005), (Ogawara, Li, & Ikeuchi, 2007), (Gupta, Mittal, & Davis, 2008), (Li, Dai, & Xu, 2011) pour des caméras RGB, ou des caméras de profondeur à l'instar de (Ziegler, Nickel, & Stiefelhagen, 2006), (Zhang, Sturm, Cremers, & Lee, 2012) ou (Gond, Sayd, Chateau, & Dhome, 2009). Plus globalement, multiplier les vues améliore les performances du système, notamment la robustesse aux occultations. Il existe certes des travaux sur le multi-Kinect (Berger, 2013) mais à notre connaissance marginalement exploités dans un contexte de capture de mouvements humains (Zhang, Sturm, Cremers, & Lee, 2012), (Caon, Yue, Tscherrig, Mugellini, & Abou Khaled, 2011).

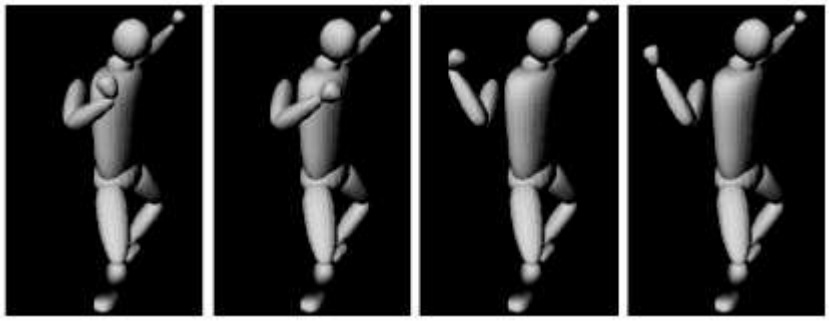


Figure 10 Quatre postures 3D pour une même reprojection 2D.
"Kinematic jump processes for monocular 3D human tracking", Sminchisescu and Triggs, Proc. Computer Vision and Pattern Recognition, 2003, ©IEEE.

E. Synthèse des caractéristiques des travaux examinés

Nous avons compilé dans le Tableau 1 un échantillon des approches présentées pour un examen rapide.

On retrouve par ailleurs d'autres caractéristiques non encore mentionnés dans ces travaux, telles que la nature modèle utilisé, la comparaison à une vérité terrain, et une vitesse d'exécution suffisante pour atteindre un traitement temps-réel du flux de données d'entrée.

On peut déplorer l'absence de travaux depuis 2013. En effet, la littérature continue de s'étoffer, mais dans les dérivés de capture de mouvement : détection d'action, classification, la capture de mouvement semblant acquise. De plus, l'arrivée de nouveaux capteurs plus évolués ont peut-être refroidi les ardeurs à la recherche logicielle en attendant que le paysage technologique se stabilise.

	Nombre de capteurs	*Nature des données*	*Initialisation automatique*	*Cohérence Spatio-temporelle*	*Modèles du corps humain*	*Comparaison à une vérité terrain*	*Temps-réel*
Agarwal & Triggs	*1*	*RGB*	*Oui*	*Oui*	*Cinématique*	*Non*	*Non*
Deutscher & Reid	*multiple*	*RGB*	*Non*	*Non*	*Non*	*Non*	*Non*
Hofmann & Gavrila	*multiple*	*RGB*	*Oui*	*Oui*	*Cinématique Texture*	*Non*	*Non*
Shotton et al.	*1*	*RGB-D*	*Oui*	*Non*	*Non*	*Virtuelle*	*Oui*
Zhang et al.	*multiple*	*RGB-D*	*Oui*	*Non*	*Non*	*Identification manuelle*	*Oui*
Wassim et al.	*multiple*	*RGB-D*	*Oui*	*Non*	*Non*	*Oui*	*Oui*
Notre approche	*multiple*	*Squelettes (RGB-D)*	*Oui*	*Oui*	*Non*	*Oui*	*Oui*

Tableau 1 Comparaison des caractéristiques de plusieurs approches de capture de mouvements humain.

F. Conclusion

Nous avons positionné notre approche de capture de mouvements humains par fusion de données par filtrage (lissage trajectoriel) de multiples données squelette provenant de capteurs RGB-D, dans une littérature riche dans chacun de ces domaines, mais où l'association précise de tous ces aspects nous semble originale et prometteuse. Nous nous inspirons aussi du concept « tracking-by-detection » initié en vision 2D. Tous ces choix, ainsi justifiés, nous semblent pertinents eu égard à la littérature en analyse vidéo.

Nous allons donc appliquer le concept d'intégration temporelle indiqué précédemment à notre problématique de capture de mouvements 3D, mais contrairement à (Hofmann & Gavrila, 2012) nous considérons une plateforme multi-Kinects, et non des caméras conventionnelles, et exploitons une reconstruction de posture type OpenNI, i.e mono-capteur.

Notre contribution majeure est donc l'élaboration d'une approche originale « tracking-by-detection, » dite encore « tracking-by-reconstruction, » tirant partie à haut-niveau d'un détecteur mono-kinect existant (Shotton, et al., 2011) étendu à de multiples points de vue capteur de profondeur, sans modèle géométrique/cinématique du corps humain, et avec intégration temporelle permettant un lissage trajectoriel.

Chapitre II: Nos plateformes multi-Kinects et bases de données acquises

Introduction

Dans la lignée de (Bailey & Bodenheimer, 2012) nous proposons de comparer qualitativement, mais surtout quantitativement les performances de la capture de mouvements issues de l'approche Mono Kinect® Microsoft et un système commercial de MoCap. Pour rappel, (Shotton, et al., 2011) a surtout évalué l'approche Kinect® sur des images synthétiques. Ainsi nous pourrons quantifier les performances de cette approche mono-Kinect puis évaluer les gains obtenus par notre approche multi-Kinects. Ceci afin d'exhiber les limitations associées au premier, et motiver notre approche.

L'utilisation de plusieurs capteurs RGB-D engendre quelques verrous à lever, notamment dû à leur nombre : étalonnage géométrique, synchronisation, bande passante, etc… Alors que la technique de Motion Capture commerciale est très précise, la coupler avec les Kinect® était également un défi.

A. Bases de données existantes

Nom	Nombre de			Vérité terrain	Nombre d'acteurs	Nombre de séquences
	Caméras	Kinects	Autres			
RGBD-HuDaAct	1	0	0	Annotations d'activité	30	14
ETHZ	2 (stéréo)	0	0	Annotations	piétons	8
PETS	2	0	1 Caméra 360°	Annotations d'activité	6	5
HumanEva	4 N&B, 3 RGB	0	0	MoCap	4	6 (marche, jogging, gestes…)
Human3.6M	4	0	1 TOF	MoCap	11	17
MHAD	2 stéréo + 2 quads	2	6 IMUs	MoCap	12	11, répétées 6 fois

Tableau 2 Comparaison de bases de données de la communauté Vision.

Il existe plusieurs bases de capture de mouvements humains en mono-caméra classique, comme TUD (Andriluka, Roth, & Schiele, 2008), ou multi-caméra comme PETS (INRIA Rhones-Alpes, 2003), ETHZ (Leibe, Schindler, & Van Gool, 2007),…) et avec MoCap commercial comme HumanEva (Sigal, Balan, & Black, 2010) et MHAD (Ofli, Kurillo, Vidal, & Bajcsy, 2013). Cependant il en existe peu avec des capteurs de profon-

deurs, surtout pour raisons historiques ; les capteurs Kinect sont assez récemment exploités dans la communauté Vision, et l'utilisation de cartes de disparité aussi.

Quelques bases de données, ainsi que leurs caractéristiques, sont résumées dans le Tableau 2. On peut voir notamment que la base contenant le plus de Kinect® n'en contient que deux, et même si il y a d'autres types de capteurs compatibles, rendre les données exploitables par notre algorithme n'était pas immédiat. On peut aussi saluer la variabilité des bases de données acquises, ce qui montre que ces bases auraient été plus propices à la création d'un algorithme de détection fusionnant les données à un niveau bas plutôt que haut.

B. Présentation de notre plateforme d'acquisition de données

Nous avons tiré parti du système commercial de motion capture installé dans les locaux du laboratoire pour acquérir par nos propres moyens une telle base multi-Kinect synchronisée et étalonnée géométriquement afin de quantifier les performances des approches développées.

Ce double étalonnage temporel et géométrique, MoCap commercial vs. plateforme multi-Kinects est peu référencé dans la littérature nous semblait-il. Aussi avons-nous élaboré une stratégie de synchronisation des systèmes utilisés à la fois sur le plan spatial et temporel.

1. Configuration matérielle globale

Figure 11 Système de Motion Capture jumelé à un capteur RGBD.

① : 4 des 10 caméras IR ou proche IR du système MoCap. ② : capteur RGBD (ici une Xtion.) ③ : référentiel MoCap ④ : mire de ~1 m² d'étalonnage RGB

Notre plateforme, illustrée par la Figure 11 avec le système de Motion Capture (①), se compose de plusieurs capteurs RGBD (②) montés sur trépied, à environ 2,0 mètre du sol, l'horizontale étant au bord supérieur du champ de la caméra couleur. Ceci permet donc normalement au système de fonctionner avec les personnes de moins de 2 m. Par exemple, pour l'expérience avec 5 kinects, leurs positions obtenues par étalonnage par OpenCV sont montrées Figure 12.

Chaque Kinect, produisant des flux vidéo RGB et un flux D (profondeur), nécessite la quasi-totalité de la bande passante d'un bus USB (de telle sorte qu'il ne reste plus assez de bande passante pour un autre flux provenant d'une autre Kinect, mais deux flux de profondeur ou deux flux vidéo provenant de deux périphériques distincts étaient possibles). Si on ajoute la bande-passante nécessaire afin de sauvegarder le flux sur un disque dur, il est préférable d'utiliser un ordinateur par capteur. La synchronisation temporelle se fait *a posteriori* avec la MoCap, multiplier les ordinateurs n'est donc pas un handicap.

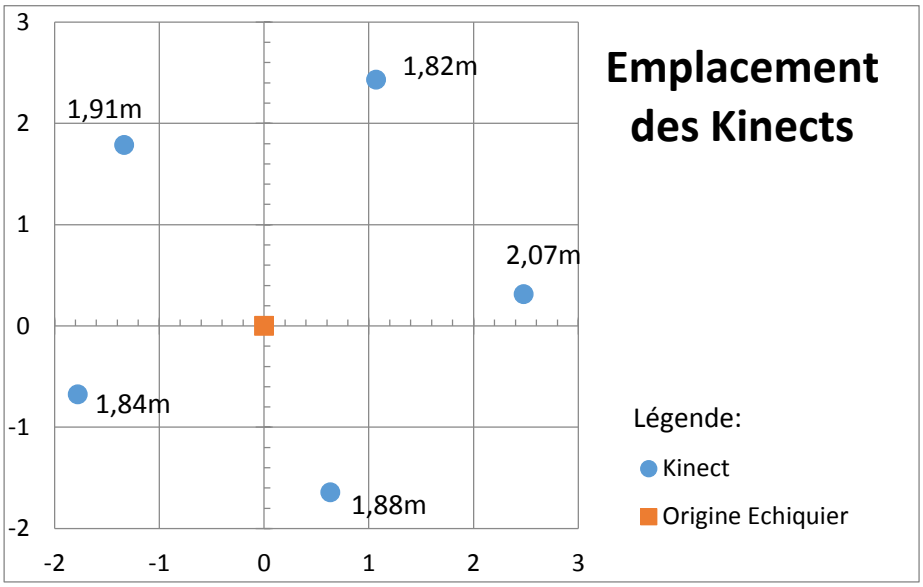


Figure 12 Emplacement des Kinects dans l'expérience avec 5 capteurs.

La distance à côté de chaque Kinect est la hauteur par rapport au plan de l'échiquier (qui fait 1,5cm d'épaisseur).

Les interférences entre les capteurs Kinect ne sont pas significatives. En effet, pour une Kinect donnée, la surface où pourrait se produire l'interférence est réduite au maximum par la répartition des Kinects autour de l'utilisateur. De plus, (Andersen, et al., 2012) montrent que seulement 4% des pixels au maximum sont perdus (aucune valeur n'est donnée) là où deux Kinects illuminent la même surface, en étant parallèles. Vu l'approche statistique de (Shotton, et al., 2011), où chaque pixel contribue à l'estimation d'une articulation, une perte éparse de surface aurait peu d'impact sur la reconstruction inférée du mouvement humain.

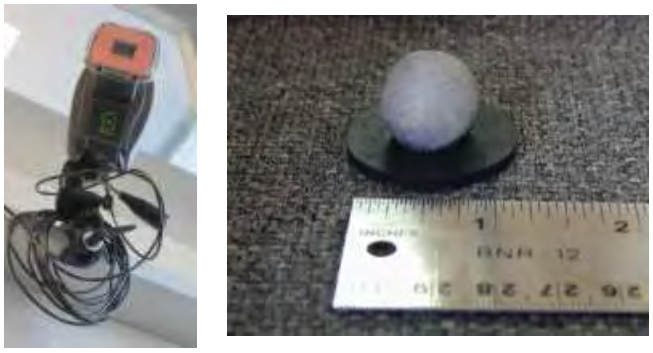
2. Le système commercial de capture de Motion Analysis (MoCap)

Nom	Hawk	Eagle
Résolution	0.3 mega pixel	1.3 mega pixel
Fréquence	1-200 Hz	1-500 Hz
Nombre de DEL	237	237
Contrôle logiciel du zoom, déclenchement, de la focale, fermeture,	Oui	Oui
Zoom embarqué avec lentille à faible distortion		Oui

Tableau 3 Caractéristiques des capteurs Motion Analysis.

Pour acquérir la vérité terrain à laquelle se référer dans l'évaluation de nos résultats, nous avons utilisé le système de capture de mouvements installé au laboratoire du LAAS. Ce système, commercialisé par Motion Analysis (Maloney, Movement Analysis Products, 2013), est constitué d'un ensemble de caméras en réseau et du logiciel d'acquisition : Cortex.

Les caméras du MoCap, dont les caractéristiques sont résumées dans le Tableau 3, sont visibles dans la Figure 11①, et Figure 13(a), captent dans l'infrarouge. L'objectif de chaque caméra est entouré d'un ensemble de diodes électroluminescentes émettant dans la même gamme de longueur d'onde. Les marqueurs sont des boules de 1cm recouvertes d'une matière catadioptrique, renvoyant la lumière plus intensément dans la direction d'origine. Ainsi, la position des boules peut être facilement déterminée par seuillage, réglable dans le logiciel, comme le gain du capteur, ainsi que l'intensité des diodes. Un exemple d'ensemble de 34 marqueurs nécessaire à la capture de mouvements humains est montré Figure 15.



(a) (b)

Figure 13 Gros plan sur une caméra infrarouge et le réseau de LEDs de la Motion Capture (a), et d'un marqueur (b).

D'autres réglages sont possibles, tels que le nombre de lignes image sur lesquelles le capteur doit s'étendre, et la méthode du calcul du centroïde de la boule dans l'image proche infrarouge. Ces positions image permettent de générer des droites de vue dont l'intersection définisse la position 3D d'un marqueur. Le nombre de droites de vue, et

donc de points de vue nécessaires à la détection d'un marqueur par le système est paramétrable, et suivant la couverture des caméras, il est essentiel pour la réduction de fausses détections de marqueurs.

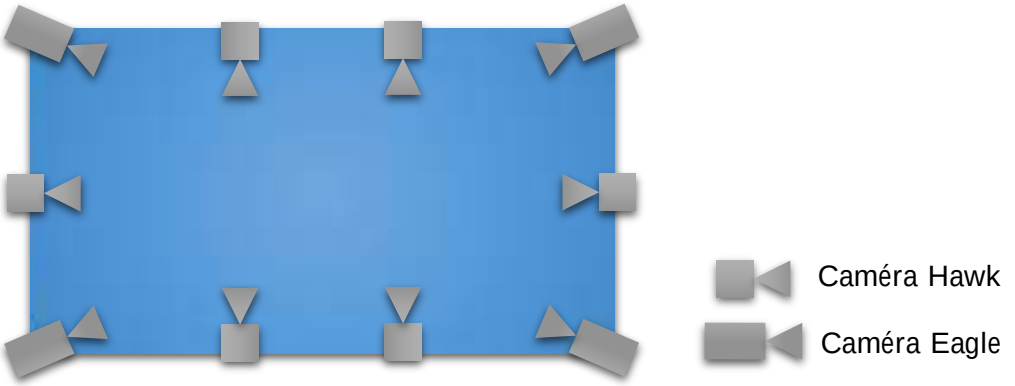


Figure 14 Configuration par défaut des caméras Infrarouge dans la salle MoCap (~8 mètre par 4).

Le logiciel Cortex n'enregistre que la position de tous les marqueurs. Pour reconnaitre les marqueurs corporels, il peut s'appuyer sur un patron qui contient les domaines de variabilité de certaines distances entre les marqueurs. Malgré ces contraintes, le système ne peut s'affranchir d'ambiguïtés lots de l'identification des marqueurs dans les divers flux vidéo enregistrés.

1	HEADB
2	HEADR
3	HEADL
4	RSHO
5	RELB_INT
6	RELB_EXT
7	RWRIST_U
8	RWRIST_B
9	RMP
10	LSHO
11	LELB_INT
12	LELB_EXT
13	LWRIST_U
14	LWRIST_B
15	LMP
16	RPEL
17	RHIP
18	RKNEEO
19	RANK
20	RHEEL
21	RPINKY
22	RTHUMB
23	LPEL
24	LHIP
25	LKNEEO
26	LANK
27	LHEEL
28	LPINKY
29	LTHUMB
30	BACKNECK
31	BACK_MID
32	BACK_DOWN
33	STERNUM
34	SCAP

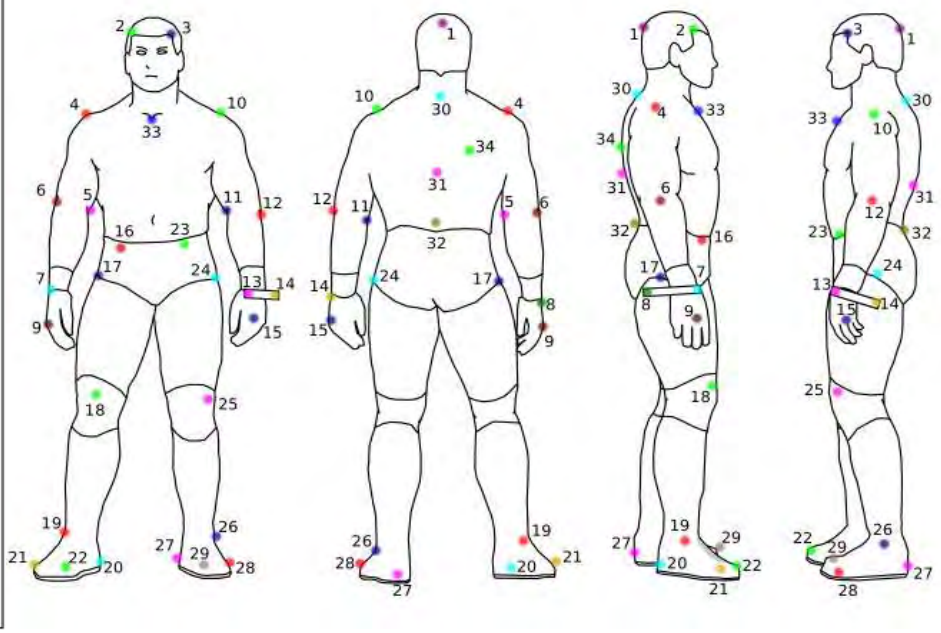


Figure 15 Noms et positionnement des marqueurs de Motion Capture.

La Figure 14 montre la configuration initiale des caméras MoCap. Le système couvre toute la salle, cependant, la quantité de rayons disponibles ne dépassait pas 2. Nous avons reconfiguré l'installation afin d'avoir toutes les caméras à fort champ joint sur la zone d'acquisition, afin de réduire les marqueurs « hallucinés » par le système

(qualificatif utilisé parce qu'ils n'existent pas dans la réalité), formant un « bruit » parasite autour du sujet et contraignant la reconnaissance des vrais marqueurs.

3. Récupération des données des articulations

Le logiciel Cortex gère les squelettes (modélisant le corps humain) comme une structure hiérarchique. Cette information est séparée des données de mouvement, et ne fournit aucun contrôle, la définition étant totalement arbitraire. Il existe cependant un module payant, nommé « Calcium Solver (Maloney, Calcium Solver) », un outil de création, configuration et positionnement d'un squelette par rapport aux marqueurs. À la place, nous avons généré les articulations à partir des marqueurs disposés sur l'acteur, listés Figure 15. Ce procédé est nommé « Marqueur Virtuel » dans Cortex.

Les marqueurs virtuels sont constructibles de plusieurs façons. Notamment à partir de 3 autres marqueurs (virtuels eux-mêmes, ou physiques) formant un repère. Les coordonnées du marqueur virtuel peuvent être exprimées en valeurs absolues le long des axes du repère (X centimètres dans le premier axe, Y dans le second axe, et Z dans l'axe orthogonal) ou en pourcentage des axes. Par utilisation de deux marqueurs virtuels il est ainsi également possible de positionner un marqueur absolument ou relativement selon les différents axes. Il est aussi possible de définir un marqueur sur la droite passant par 2 autre marqueurs.

Nom	Définition	Origine	Extrêmité axe X	Marqueur du plan	Valeur X	Valeur Y	Valeur Z
V_head	3 pt. Ratio	HEADR	HEADL	HEADB	50%	33%	40%
V_handR	3 pt. Ratio	RMP	RWRIST_U	RWRIST_B	10%	30%	11%
V_handL	3 pt. Ratio	LMP	LWRIST_U	LWRIST_B	10%	30%	-11%
V_footR	3 pt. Ratio	RHEEL	RTHUMB	RANK	50%	10%	-1%
V_footL	3 pt. Ratio	LHEEL	LTHUMB	LANK	50%	10%	-1%
V_elbowR_const	2 pt. Ratio	RELB_int	RELB_ext		50%		
V_shoulderR	3 pt. Ratio	V_elbowR_const	RSHO	BACK_DOWN	93%	20%	1%
V_elbowR	3 pt. Mesure	V_shoulderR	V_elbowR_const	RELB_ext	285 mm	1 mm	-35 mm
V_elbowL_const	2 pt. Ratio	LELB_int	LELB_ext		50%		
V_shoulderL	3 pt. Ratio	V_elbowL_const	LSHO	BACK_DOWN	93%	20%	1%
V_elbowL	3 pt. Mesure	V_shoulderL	V_elbowL_const	LELB_ext	285 mm	1 mm	35 mm
V_hipR	3 pt. Mesure	RHIP	LHIP	BACK_DOWN	55 mm	-20 mm	55 mm
V_kneeR	3 pt. Mesure	RANK	RKNEEO	RHIP	430 mm	20 mm	25 mm
V_ankleR	3 pt. Mesure	RANK	RTHUMB	RHEEL	10 mm	20 mm	-20 mm
V_hipL	3 pt. Mesure	LHIP	RHIP	BACK_DOWN	55 mm	-20 mm	-55 mm
V_kneeL	3 pt. Mesure	LANK	LKNEEO	LHIP	430 mm	20 mm	-25 mm
V_ankleL	3 pt. Mesure	LANK	LTHUMB	LHEEL	10 mm	20 mm	20 mm
V_neck1	2 pt. Ratio	BACKNECK	STERNUP		30%		
V_Neck	3 pt. Ratio	V_neck1	V_head	BACKNECK	50%	0%	0%
V_waist	2 pt. Ratio	V_hipL	V_hipR		50%		
V_chest	3 pt. Ratio	V_waist	V_neck1	BACK_DOWN	56%	-1%	0%
V_wristR	3 pt. Mesure	RWRIST_U	RWRIST_B	RMP	40 mm	20 mm	15 mm
V_wristL	3 pt. Mesure	LWRIST_U	LWRIST_B	LMP	40 mm	20 mm	-15 mm
V_Spine	3 pt. Ratio	BACKNECK	BACK_MID	STERNUP	50%	33%	0%
V_LeftToeBase	3 pt. Ratio	V_footL	LTHUMB	LPINKY	70%	10%	0%
V_RightToeBase	3 pt. Ratio	V_footR	RPINKY	RTHUMB	70%	10%	0%
V_LowerBack	3 pt. Ratio	RPEL	LPEL	BACK_DOWN	50%	66%	0%

Tableau 4 Définitions des marqueurs virtuels dans la Motion Capture.

En utilisant cette technique, nous avons essayé de localiser les centres des articulations traquées. Il est à noter que ne bénéficiant pas de moyen de vérifier la qualité de cette vérité terrain, un soin tout particulier a été apporté à son élaboration pour qu'elle soit la plus authentique possible. En cela, nous avons contrôlé que les os étaient de longueur fixes dans les acquisitions et avons ainsi obtenu des variations inférieures à 5 cm avant de mesurer nos estimations basées Kinect à cette vérité terrain.

Les valeurs utilisées pour construire les marqueurs virtuels sont énumérées dans le Tableau 4. Comme écrit plus haut, chaque marqueur est construit en fonction des précédents et des marqueurs réels qui se trouvent sur le corps de l'acteur (voir Figure 15). Chaque méthode de définition est listée en première colonne, indiquant si le marqueur est défini en fonction de 3 ou 2 points, et si les valeurs sont dynamiques en fonction de l'écartement des marqueurs, ou bien si elles sont fixes. Par exemple on voit que le mar-

queur **V_elbowR_const** est construit comme le point médian du segment [**RELB_int**, **RELB_ext**].

C. Étalonnage des deux systèmes

Cette section présente les étalonnages effectués pour synchroniser temporellement et colocaliser les données entre les deux systèmes : Kinect® et système de capture de mouvement commercial Motion Analysis. Cette procédure a à notre connaissance aucun précédent dans la littérature, et nous espérons qu'elle aussi sera utile à la communauté, en complément des données acquises, en tant que contribution.

1. Étalonnage géométrique du système MoCap dans le repère Kinect

La procédure d'étalonnage comporte trois étapes.

La première, l'étalonnage intrinsèque des capteurs RGB. L'image est sensiblement sous l'effet d'une distorsion, assez pour fausser un étalonnage extrinsèque. Les images de profondeur et de couleur doivent alors être corrigées en distorsion avant d'être utilisées. La transformation profondeur-vers-RGB est déterminée en usine, et assurée par OpenNI. Certains travaux (Herrera C., Kannala, & Heikkila, 2012) montrent qu'on peut améliorer de manière significative cette transformation, mais seulement dans une zone à proximité du capteur que nous n'utilisons pas.

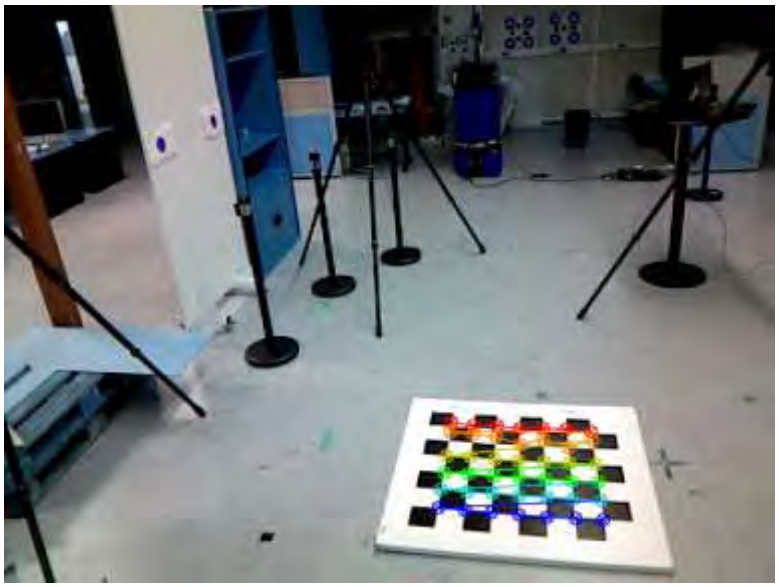


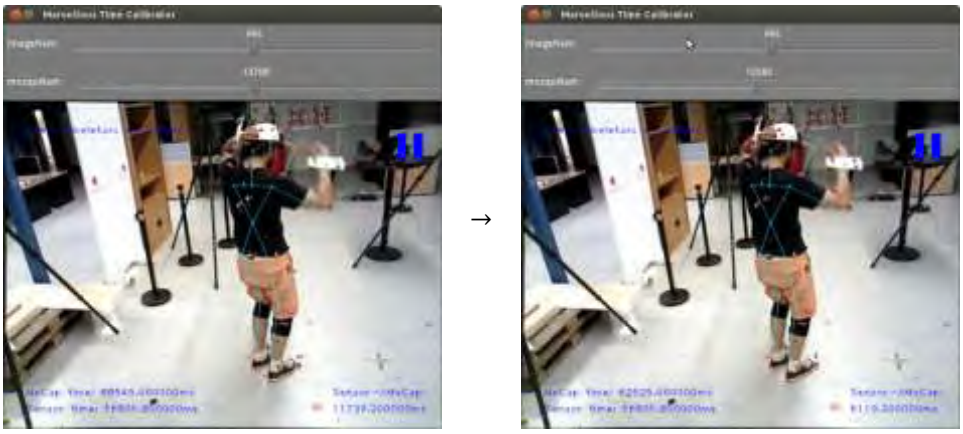
Figure 16 Image de la mire pour étalonnage géométrique.

La seconde étape consiste à obtenir la transformation de chaque repère capteur (Kinect®) vers un repère commun. Dans notre cas, un échiquier et une fonction OpenCV suffit (comme l'illustre la Figure 16.)

Enfin, on place et oriente le référentiel MoCap sur la mire d'étalonnage couleur, respectivement ③ et ④ dans la Figure 11, afin de localiser tous les capteurs dans le référentiel MoCap.

2. Synchronisation temporelle

La synchronisation temporelle a été faite *a posteriori* par projection des marqueurs MoCap sur l'acteur (Figure 15) dans les images couleurs. Sur ces images, un mouvement rapide laissait une traînée floue par marqueur. Puisque nous avions 10 fois plus de mesures MoCap que d'images, nous avons finement ajusté la synchronisation de telle sorte que le marqueur était projeté au milieu de sa traînée. Enfin, puisque tous les flux, squelette, couleur, profondeur et de segmentation, étaient synchronisés entre eux car ils étaient générés par un même processus utilisant OpenNI qui fournit un horodatage, tous les flux étaient synchronisés. Nous avons aussi supposé que les séquences étaient trop courtes (90 secondes en moyenne) pour que toute dérive temporelle des horloges soit non significative par rapport à la précision attendue ne se produise entre la MoCap et OpenNI.



non synchronisé *synchronisé*

Figure 17 Outil de synchronisation temporelle.

La Figure 17 montre le petit outil de synchronisation temporelle élaboré. Il ouvre un dossier d'images, un fichier MoCap et les paramètres d'étalonnage de la caméra et du MoCap. Les marqueurs sont reprojetés en rouge. Optionnellement, un fichier de données Kinect peut être chargé, il est alors affiché en cyan, et un nouveau fichier sera recréé avec des temps MoCap à la fermeture du programme.

D. Bases de séquences acquises

Nous avons fait l'acquisition de 2 bases de données. La première avec seulement 3 Kinects était le premier essai. La seconde base a eu pour but de tester la configuration optimale des capteurs, en utilisant non pas 3 Kinects, mais 5, et de tester leurs différentes combinaisons : deux-à-deux, trois-à-trois... Et évaluer toutes les configurations géométriques.

		3KAL35		5KAL35
Vues Couleurs		3		5
Vues Profondeurs		3		5
Capteurs MOCAP		10		9
Marqueurs MOCAP		35		35
Durée		Environ 16 minutes		Environ 7 minutes
Fréquence		20 images / s		30 images / s
Nb séquences		9		4
Total Nb Postures		21 569		11193
Séquences	C1	Posture en T, mouvements bras, mouvements jambes, mouvements genoux, accroupis, bascule	C1	Posture en T, Tennis, Smash, haltérophilie, gymnastique, danse, répété.
	C2	Posture en T, haltérophilie, tennis, volley ball, ping-pong, natation (bras), pétanque, lancement de poids	C2	Posture en T, marche (sortie de champ)
	C3	Posture en T, haltérophilie, tennis, volley ball, ping-pong, Natation (bras), Pétanque, Lancement de poids	C3	(dans cette séquence, le bas du corps est occulté par le devant.) Manipulation sur table, jonglage, répété.
	C4	Posture en T, Tour sur soi, marche, course, assis debout, assis par terre, accroupi	C4	Posture en T, boxe, kicks, brasse, crawl, plongée, sautillements.
	C5	Posture en T, Tour sur soi, marche, Course, assis debout, assis par terre, accroupi		
	C6	Posture en T, saut américain, mouvements bras, équilibre, étirement, boxe, bowling, danse, chute avant, chute arrière, conduite, équilibre		
	C7	Déplacer chaise, s'asseoir, balayer assis, déplacer meuble, balayer, bouger et filmer, jouer avec des balles		
	C8	Posture en T, tour sur soi, mouvement articulations, équilibre, mouvement accroupi, karaté, échauffement, toucher les pieds, mouvements sur genoux		
	C9	Posture en T, tour sur soi, haltérophilie, marche, tennis, volley ball, course, ping-pong, assis par terre, natation (bras), pétanque, lancement de poids, saut à la corde		

Tableau 5 Récapitulatif des bases de données.

Le Tableau 5 récapitule les caractéristiques des deux bases capturées tandis que quelques images de la première base sont présentées dans la Figure 11 pour illustration.

E. Conclusion

Une première contribution est ici l'acquisition de ces bases de séquences multi-Kinects avec vérité terrain rigoureuse. Elle répond à un manque de la littérature en

termes de nombreuses données Kinect. Les bases seront rendues publiques via l'interface web sur http://homepages.laas.fr/lerasle/hmc/dataset.php.

La vérité terrain que nous avons construite en parallèle d'une base de donnée capteur et détecteur squelette nous permettra d'évaluer les performances de notre approche et l'influence des paramètres libres du système. Nous quantifierons au Chapitre IV les gains obtenus lors des chapitres suivants, après la formalisation détaillée de notre approche qui se trouve dans le Chapitre III ci-après.

Figure 18 Exemples de snapshots de séquences de la base 3KAL35.

Chapitre III: Formalisation et description de notre approche

Introduction

Ce chapitre traite de la capture de mouvements humains. Rappelons que la démarche est de fusionner les reconstructions de squelettes au sens de (Shotton, et al., 2011) issues de plusieurs Kinects. Nous visons des gains en robustesse et précision que nous pourrons alors quantifier avec la vérité terrain (voir le Chapitre I:F).

Nous rappelons que nous visons des applications temps-réel. Ce type d'application implique des contraintes CPU afin que d'autres traitements soient effectués en parallèle de la perception pour le confort de l'utilisateur. Sans pour autant arriver au niveau de performances d'un système commercial de capture de mouvements avec marqueurs, nous essayons d'atteindre une précision suffisante pour, par exemple, détecter les mouvements de prise d'objets. La comparaison avec un système commercial doit aussi être modérée car au contraire de ce dernier, notre approche non-invasive et bas-coût car elle ne requiert aucune instrumentation de l'utilisateur.

À l'instar de (Shotton, et al., 2011), où la détection est un processus parallèle et au service du système de jeu, la perception ici doit être au service de l'application, et ne doit pas nécessiter une puissance de calcul élevée. Une vitesse de traitement temps-réel permet par ailleurs de voir ce que comprend le robot et réagir en conséquence si besoin. On parle de vitesse de traitement « super-temps-réel » quand le traitement prend fin bien avant la réception d'une nouvelle donnée capteur, ou sur un fraction des ressources matérielles disponibles, comme par exemple un seul cœur processeur pour un processeur multi-cœurs.

Ce chapitre est structuré en trois parties. Premièrement, nous exposons la formalisation de notre approche. Ensuite, nous présentons et discutons la première évaluation réalisée. Nous tirons les conclusions dans la troisième et dernière partie.

A. Formulation du problème

Le but de cette partie est de présenter la problématique et d'en établir une modélisation formelle. Dans un premier temps nous présentons donc le problème, montrons qu'il s'agit d'un problème connu et résolvable facilement par une approche décrite dans un second temps.

On considère $\{X_j\}_{j\in J}$ les positions d'un ensemble J (de l'anglais Joint) d'articulations du squelette humain, variables à estimer, définies par leurs coordonnées euclidiennes sous

la forme

$$X = \{X_j\}_{j\in J} = \left\{\begin{bmatrix} x^j \\ y^j \\ z^j \end{bmatrix}\right\}_{j\in J}$$

Il est à noter que même si le nombre de degrés de liberté réel du corps humain n'est pas aussi grand, c'est un espace cible convenable.

À chaque instant image t, chaque capteur $k \in K$ (pour Kinect®, même si les capteurs spécifiques ne sont pas importants, seules les données comptent) produit une bitmap de segmentation du premier plan correspondant à la silhouette humaine S_t^k par rapport à la bitmap de profondeur, et génère une reconstruction du squelette $Y_t^k = \left[Y_t^{j,k}\right]_{j\in J}$. Nous choisissons de ne pas nous limiter à ces dernières, donc nous considérons à la place un sur-ensemble $L \supset K$ d'hypothèses au sens large. Chaque hypothèse d'index $l \in L$ peut s'exprimer

$$Y_t^l = X_t + V_t^l, \forall l \in L$$

où V_t^l est l'erreur de reconstruction de squelette. Cette erreur n'est certainement ni un bruit blanc, ni un bruit gaussien.

Nous faisons l'hypothèse qu'il suffit de sélectionner sur l'ensemble $\{Y_t^l\}_{l\in L}$ des hypothèses, Y_t^* celle minimisant cette erreur, et que nous pouvons le faire en maximisant la probabilité $\mathbb{P}\left(Y_t^l \middle| \{S_t^k\}_{k\in K}\right)$. Soit :

$$\hat{X}_t = Y_t^* = \underset{\{Y_t^l\}_{l\in L}}{\operatorname{argmax}} \mathbb{P}(Y_t^l | \{S_t^k\}_{k\in K}) \tag{1}$$

Pour tous les ensembles tels que J, K et L, nous écrirons K au lieu de $|K|$ ou $card(K)$ pour alléger les notations.

Pour des raisons de coût CPU, nous choisissons de limiter la recherche à ces L hypothèses, recherchant la reconstruction avec la plus petite erreur eu égard à l'ensemble $\{S_t^k\}_{k\in K}$ représentant l'ensemble des points de vue. La cohérence spatio-temporelle des mouvements humains est une propriété naturelle à exploiter. Ainsi, pour tirer parti de cette cohérence, nous optimisons cette erreur sur un ensemble de $M \in \mathbb{N}$ instants images successifs.

Le but est donc de trouver, comme estimation :

$$\begin{aligned} Y_{t-M:t}^* &= \underset{\{Y_s^{l_s}\}_{s=t-M:t}}{\operatorname{argmax}} \mathbb{P}\left(\{Y_s^{l_s}\}_{s=t-M:t} \middle| \{S_{t-M:t}^k\}_{k\in K}\right) \\ &= \underset{\{Y_s^{l_s}\}_{s=t-M:t}}{\operatorname{argmax}} \frac{\mathbb{P}\left(\{S_{t-M:t}^k\}_{k\in K} \middle| \{Y_s^{l_s}\}_{s=t-M:t}\right) \times \mathbb{P}\left(\{Y_s^{l_s}\}_{s=t-M:t}\right)}{\mathbb{P}\left(\{S_{t-M:t}^k\}_{k\in K} \cdot \{Y_s^{l_s}\}_{s=t-M:t}\right)} \\ &= \underset{\{Y_s^{l_s}\}_{s=t-M:t}}{\operatorname{argmax}} \mathbb{P}\left(\{S_{t-M:t}^k\}_{k\in K} \middle| \{Y_s^{l_s}\}_{s=t-M:t}\right) \times \mathbb{P}\left(\{Y_s^{l_s}\}_{s=t-M:t}\right) \end{aligned} \tag{2}$$

par le théorème de Bayes.

Cette problématique ainsi formulée, communément appelé intégration temporelle, ou « Modal Trajectory Estimation », est déjà résolue par programmation dynamique. Dont la sous-section suivante présente le formalisme.

B. Programmation dynamique

D'après (Larson & Peschon, 1966), le point terminal de la trajectoire optimale, et donc notre estimation à l'instant t, peut être trouvé par :

$$Y_t^* = \underset{Y_t^{l_t}}{\operatorname{argmax}}\, \mathcal{J}_t^*\left(Y_t^{l_t}, \{S_{t-M:t}^k\}_{k\in K}\right) \tag{3}$$

Où $\mathcal{J}_t^*$ est la vraisemblance Bayésienne marginale :

$$\mathcal{J}_t^*\left(Y_t^{l_t}, \{S_{t-M:t}^k\}_{k\in K}\right) \triangleq \max_{Y_{t-M:t-1}} \mathbb{P}\left(\{S_{t-M:t}^k\}_{k\in K} \middle| \{Y_s^{l_s}\}_{s=t-M:t}\right) \times \mathbb{P}\left(\{Y_s^{l_s}\}_{s=t-M:t}\right) \tag{4}$$

En commençant par la connaissance de la valeur initiale $\mathcal{J}_{t-M}^*\left(Y_{t-M}^{l_{t-M}}, \{S_{t-M}^k\}_{k\in K}\right) = \mathbb{P}\left(\{S_{t-M}^k\}_{k\in K} \middle| Y_{t-M}^{l_{t-M}}\right) \times \mathbb{P}\left(Y_{t-M}^{l_{t-M}}\right)$, elle peut être récursivement calculée par

$$\mathcal{J}_t^*\left(Y_t^{l_t}, \{S_{t-M:t}^k\}_{k\in K}\right) = \max_{Y_{t-1}^{l_{t-1}}} \mathbb{P}\left(\{S_t^k\}_{k\in K} \middle| Y_t^{l_t}\right) \times \mathbb{P}\left(Y_t^{l_t} \middle| Y_{t-1}^{l_{t-1}}\right) \times \mathcal{J}_{t-1}^*\left(Y_{t-}^{l_t}, \{S_{t-M:t-1}^k\}_{k\in K}\right) \tag{5}$$

Où $\mathbb{P}\left(\{S_t^k\}_{k\in K} \middle| Y_t^{l_t}\right)$ peut être considéré comme une probabilité d'observation pour laquelle nous devons trouver un modèle d'observation, et $\mathbb{P}\left(Y_t^{l_t} \middle| Y_{t-1}^{l_{t-1}}\right)$ une probabilité de transition pour laquelle nous devons trouver un modèle dynamique.

Enfin, une implémentation exploitant cette démonstration est le célèbre algorithme de Viterbi (Viterbi, 2006).

1. Algorithme de Viterbi

L'algorithme travaille sur des données structurées en un treillis de L états et de M sections. On appelle aussi L la largeur et M la longueur du treillis. Une illustration est proposée dans la Figure 19.

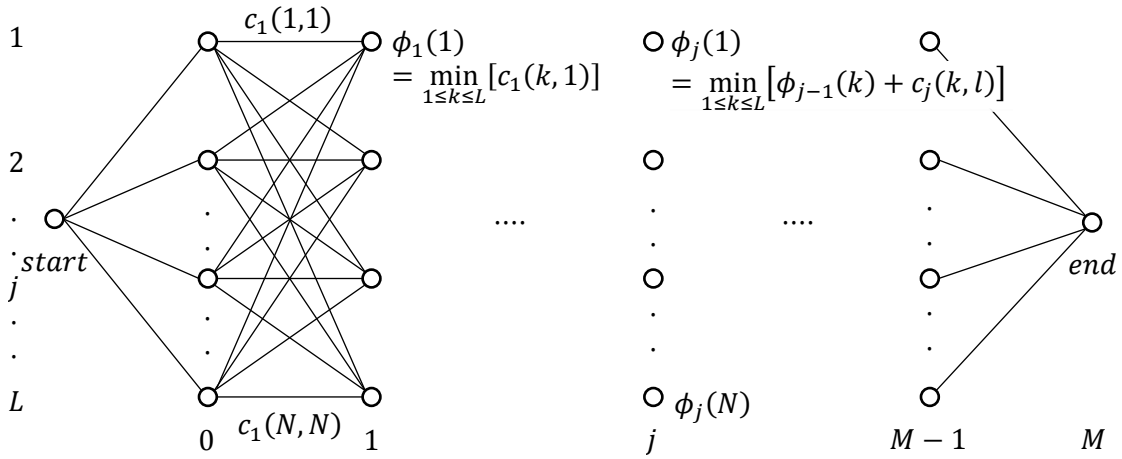


Figure 19 Treillis entièrement connecté de N Etats.

Le nombre total de séquences possibles est donc L^2M. Le coût incrémental associé à une transition de l'état i à l'instant $t-1$ à l'état j à l'instant t se note $c_t(i,j)$. Le but est de trouver le chemin le moins coûteux à travers le treillis. Soit $\phi_t(j)$ le coût minimum pour accéder à l'état j au temps t, et l'historique du chemin associé stocké dans $\xi_t(j)$.

C'est donc un état du treillis, et le chemin complet peut être reconstruit à rebours en commençant par la fin.

L'algorithme de Viterbi peut être résumé comme suit dans l'Algorithme 1 :

1. Initialisation : (t=0)

$$\phi_1(l) = c_0(start, l) \quad (6)$$
$$\xi_1(l) = start$$
$$1 \le l \le L$$

2. Récursion :$(0 < t < M)$

$$\phi_t(l) = \min_{1 \le k \le L} [\phi_{t-1}(k) + c_t(k, l)] \quad (7)$$
$$\xi_t(l) = \arg\min_{1 \le k \le L} [\phi_{t-1}(k) + c_t(k, l)]$$
$$1 \le l \le L$$

3. Terminaison : $(t = M)$

$$\phi_M(end) = \min_{1 \le k \le L} [\phi_{M-1}(k) + c_M(k, l)] \quad (8)$$
$$\xi_t(end) = \arg\min_{1 \le k \le L} [\phi_{M-1}(k) + c_M(k, l)]$$
$$= i_{M-1}$$

4. Remontée de la solution :

Le meilleur chemin est

$$(start, i_0, \dots, i_{M-1}, end)$$

où

$$i_t = \xi_{t+1}(i_{t+1}) \ , \quad (9)$$
$$0 \le t < M \ .$$

Algorithme 1 Algorithme de Viterbi.

L'algorithme de Viterbi est un algorithme minimisant le coût ϕ_M. La valeur du coût doit donc être déduite de la probabilité que nous cherchons à maximiser. Cette transformation est directe pour des probabilités gaussiennes de la forme $\mathbb{P}(c) = k \cdot e^{-c}$, telles que celles que nous utilisons et décrivons ci-après, en assimilant le terme c au coût à minimiser.

2. Modélisation des probabilités

a) Probabilité d'Observation

Afin de déterminer la vraisemblance d'une hypothèse $Y_t^{l_t}$ de squelette à l'instant t, la donnée la plus adaptée de par sa disponibilité et sa robustesse est $S_t^K = \{S_t^k\}_{k \in K}$, les segmentations des régions d'intérêt produites durant la détection des squelettes par NiTE dans les K capteurs. Nous cherchons donc $\mathbb{P}\left(\{S_t^k\}_{k \in K} \middle| Y_t^{l_t}\right)$ comme probabilité d'observation du squelette $Y_t^{l_t}$ étant donné l'ensemble $\{S_t^k\}_{k \in K}$ dans tous les capteurs.

La segmentation produite est une image binaire, une matrice d'identifiants, isolant la silhouette de l'arrière-plan. Pour un unique utilisateur, la donnée dégénérée est une matrice binaire (le nom de bitmap est ici pleinement justifié) de la taille la résolution du capteur, notée WxH (dans notre cas, $W = 640$ et $H = 480$) :

$$S_t^k = \left\{S_t^k \ (\mathrm{i}, \mathrm{j}) \in \{0,1\}\right\}_{\substack{0 \le i < H \\ 0 \le j < W}}$$

où 1 désigne un pixel appartenant au premier plan (l'utilisateur), et 0 l'arrière-plan.

Pour autant que nous puissions en juger, le middleware réalise une extraction de la silhouette, i.e. du premier plan, puisqu'il est perdu lorsque le capteur est déplacé. Les nouveaux acteurs entrant dans le champ de vision sont identifiés et maintenus indépendants ensuite, et l'exploitation du canal profondeur permet de s'affranchir de toute considération d'apparence (texture, couleur) de la cible.

Puisque la segmentation utilisateur $S_t^k(i,j)$ est la segmentation de premier plan en i,j de l'utilisateur tel que vu par le capteur k, elle peut être comparée à une segmentation artificielle $\mathbb{S}_t^{l,k}(i,j)$ basée sur la projection du squelette Y_t^l faite de rectangles, cercles et triangles blancs sur fond noir dans plan image du capteur k. Cette modélisation s'approche d'un modèle cylindre mais est facile à dessiner pour des raisons de calcul.

La Figure 20 donne un exemple des images $\mathbb{S}^{l,k} \oplus S^k$ pour $L = K$ où K est un ensemble de 3 capteurs. On peut y voir que la silhouette est entourée d'un halo blanc, signifiant une valeur de 1, où les images $\mathbb{S}^{l,k}$ et S^k diffèrent. Chaque imagette de la diagonale présente très peu de disparité car le squelette est alors reprojeté dans la vue capteur qui a permis d'inférer la posture. Mais une fois reprojetée dans un autre point de vue (les imagette en dehors de la diagonale), le squelette d'apparence correct dans son propre point de vue peut produire des résultats variables une fois reprojeté dans une vue autre que celle d'origine..

On peut en particulier voir que le squelette 2, provenant du capteur de profil, ne peut voir une jambe, et cela est observable sur les deux autres points de vue. Dans cette situation, le squelette 3, pris de face, est le meilleur d'un point de vue aire totale et ceci pour tous les capteurs.

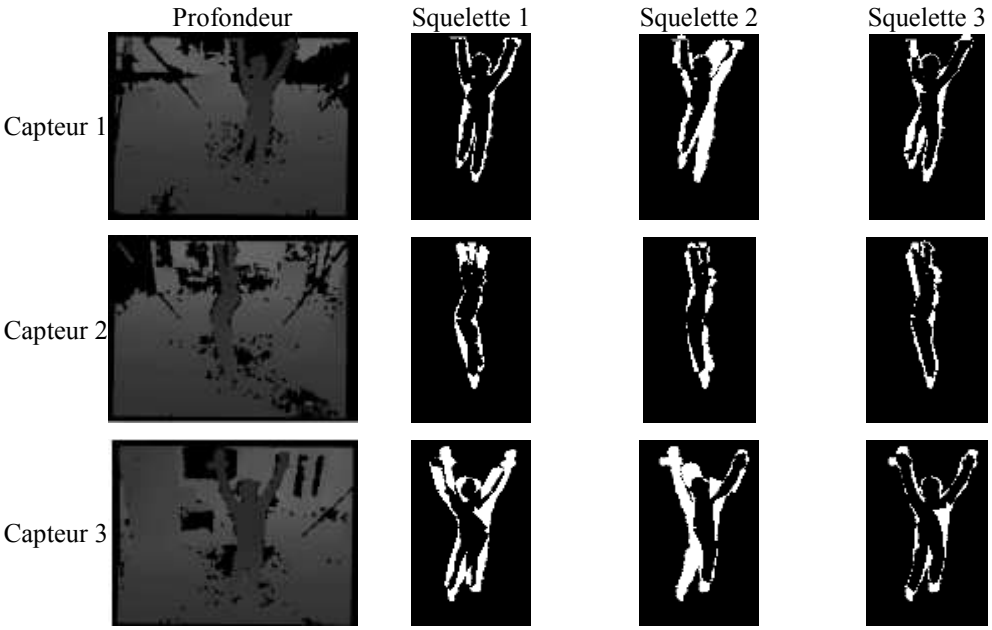


Figure 20 Exemple de « ou exclusif » entre squelette reprojeté et segmentation silhouette.

Pour calculer une distance entre des cartes de segmentations, nous comptons simplement les pixels blancs dans exclusivement l'une ou l'autre image, c'est-à-dire, le « ou exclusif »; ou bien la somme des différences absolues entre chaque pixel : $d_t(k,l) = \sum_{i,j} \mathbb{S}^{l,k}(i,j) \oplus S^k(i,j) = \sum_{i,j} |S_t(i,j) - \mathbb{S}(i,j)|$.

L'algorithme devra logiquement attribuer une meilleure vraisemblance à l'hypothèse qui se reprojette au mieux dans l'ensemble des vues K.

En l'état, d_t varie avec la résolution et la place qu'occupe le corps de l'acteur dans l'image, qui est fonction de sa distance au capteur. Ainsi, il faut normaliser. Si nous le faisons par rapport à l'aire totale occupée sur l'image réelle, nous obtenons une mesure en grande partie insensible à la distance, si nous ignorons les effets d'échantillonnage dus au travail avec une bitmap. Il en résulte:

$$\mathrm{D_t}(k,l) = \frac{d_t(k,l)}{\sum_{i,j} S_t^k(i,j)} \tag{10}$$

Muni de cette « distance » entre reprojection squelette et segmentation utilisateur, nous pouvons ensuite modéliser la densité de probabilité $\mathbb{P}\left(\{S_t^k\}_{k\in K}\middle|Y_t^{l_t}\right)$ d'avoir une observation correcte, comme la distribution normale autour d'un D_t d'une moyenne $\mu = 0{,}1$ (10%) pour cause de bruit de mesure de profondeur et l'approximation des membres par des cylindres, et d'écart-type $\sigma = 0{,}1$, c'est à dire

$$\mathbb{P}\left(\{S_t^k\}_{k\in K}\middle|Y_t^{l_t}\right) \triangleq \prod_{k\in K} \frac{1}{\sqrt{2\cdot\pi\cdot\sigma^2}} e^{-\frac{(\mathrm{D_t}(k,l)-\mu)^2}{2\cdot\sigma^2}} \tag{11}$$

Les valeurs de μ, σ ainsi que les valeurs et formules de dessin des silhouettes ont été déterminées empiriquement, et gagneraient à être optimisées automatiquement sur un ensemble de donnée, ainsi que déterminées en fonction des personnes, dans le cadre du dessin de la silhouette.

b) Probabilité de transition entre instants image successifs

Considérant la nature cartésienne des degrés de liberté de notre modèle issu du format de données squelettes OpenNI, par opposition à un modèle intrinsèquement articulaire, il est difficile d'obtenir une cinématique proche de celle de l'humain. Nous pensons qu'utiliser un modèle articulaire aurait plus de sens, mais en contrepartie serait trop long à calculer, et donc incompatible avec nos contraintes de coût CPU.

Pour l'instant, l'objectif du modèle de dynamique est de réduire le jitter et de favoriser la cohérence spatiotemporelle inter-images, en premier lieu filtrer les inversions de labellisation gauche-droite de parties corporelles classiquement observées lors de la reconstruction OpenNI d'une personne de dos.

Ainsi, nous utilisons une densité gaussienne centrée sur chaque articulation, avec un écart-type ajusté pour chaque articulation, basé sur l'écart-type réglé via la vérité terrain.

$$\mathbb{P}\left(Y_t^{l_t}\middle|Y_{t-1}^{l_{t-1}}\right) \triangleq \prod_{j\in J} \frac{1}{\sqrt{2\cdot\pi\cdot{\sigma_j}^2}} e^{-\frac{\left(Y_t^{j,l_t}-Y_{t-1}^{j,l_{t-1}}\right)^2}{2\cdot{\sigma_j}^2}} \tag{12}$$

C. Estimation supplémentaire : filtres de Kalman à réjection

Le filtre de Kalman que nous avons employé utilise un modèle d'état très simple pour limiter son coût CPU. Chaque état X^j de l'articulation j est traqué séparément comme suit. Les unités sont le mm, et le pas de temps $30^{-1}s$:

$$X^j = (x \quad y \quad z \quad \dot{x} \quad \dot{y} \quad \dot{z})^T \quad (S)$$

(S) Définit l'état du filtre

$$X_t^j = A \cdot X_{t-1}^j + W_{t-1} \quad (E)$$

(E) définit l'évolution temporelle de l'état à partir de la matrice d'évolution A et du bruit dynamique W.

$$Z_t^j = H \cdot X_{t-1}^j + V_{t-1} \quad (O)$$

(O) définit l'observation Z en fonction de l'état courant, la matrice d'observation H et le bruit d'observation V.

W et V sont supposés être distribués selon un bruit blanc gaussier centré sur 0 et de matrices de variance Q et R respectivement :

$$W \sim \mathcal{N}(0, Q) \quad et \quad V \sim \mathcal{N}(0, R) \qquad (N)$$

Dans notre cas, voici les valeurs des matrices (en millimètres pour les variances) :

$$H = (I_3 \quad 0_3) \quad (O_M), \quad A = \begin{pmatrix} I_3 & I_3 \\ 0_3 & \mathrm{k} \cdot I_3 \end{pmatrix} \quad (E_M)$$

$$Q = \begin{pmatrix} {\sigma_q}^2 \cdot I_3 & 0 \\ 0 & {\sigma_q}^2 \cdot I_3 \end{pmatrix} \quad (N_{M1}), \quad R = {\sigma_r}^2 \cdot I_3 \quad (N_{M2})$$

Voici les principales équations d'un filtre de Kalman pour obtenir $\hat{X}_{t+1}^j$ l'estimation de X_{t+1}^j :

$$\begin{aligned} \hat{X}_{t+1}^{j-} &= A \cdot \hat{X}_t^j \\ P_{t+1}^{j-} &= A \cdot P_t^j \cdot A^{-1} + Q \end{aligned} \tag{13.1}$$

$$\begin{aligned} S_{t+1} &= H \cdot P_{t+1}^{j-} \cdot H^T + R \\ \boldsymbol{r}_{t+1}^j &= Y_{t+1}^j - H \cdot \hat{X}_{t+1}^{j-} \end{aligned} \tag{13.2}$$

$$\begin{aligned} K_{t+1} &= P_{t+1}^{j-} \cdot H^T \cdot S_{t+1}^{-1} \\ \hat{X}_{t+1}^j &= \hat{X}_{t+1}^{j-} + K_{t+1} \cdot \boldsymbol{r}_{t+1}^j \\ P_{t+1}^j &= P_{t+1}^{j-} - K_{t+1} \cdot H \cdot P_{t+1}^{j-} \end{aligned} \tag{13.3}$$

L'étape (13.1) est effectuée à chaque pas de temps. On calcule la prédiction $\hat{X}_{t+1}^{j-}$ à partir de l'estimation précédente $\hat{X}_t^j$ d'après (E). La matrice d'estimation des covariances P_t^j est aussi mise à jour.

La matrice des résidus S_{t+1} et le résidu de l'innovation r_{t+1}^j sont ensuite calculés en (13.2). Ce sont ces deux valeurs qui sont utilisées afin de tester la réjection expliquée ci-après. La réjection annule éventuellement le calcul à ce moment, sinon, l'étape (13.3), la correction, est effectuée avec le gain K_{t+1}. Sinon, l'estimée est seulement la prédiction.

Pour chaque mesure supplémentaire, l'étape (13.1') ci-après est faite en remplacement de (13.1) et l'algorithme réitère à partir de (13.2) comme décrit précédemment.

$$(\hat{X}_{t+1}^{j-}, P_{t+1}^{j-}) = (\hat{X}_{t+1}^{j}, P_{t+1}^{j}) \quad (13.1')$$

Cependant, toutes les articulations ne suivent pas un tel modèle même très approximativement. Par exemple, si l'utilisateur est de profil, les deux pieds peuvent se retrouver positionnés au même endroit. Pour éviter l'assimilation de telles mesures (ce qui dégrade sévèrement les performances), une réjection (ou vérification des résidus) est mise en place.

Cette technique, publiée pour la première fois dans (Maybeck, 1979), consiste à rejeter l'assimilation de Y_t^j si

$$(\boldsymbol{r}_t^j)^T \cdot S_t^{-1} \cdot \boldsymbol{r}_t^j > g_t^2 \quad (R)$$

où $\boldsymbol{r}_t^j$ est l'innovation, S_t la matrice de covariance des résidus, et g_t est le seuil de réjection. g_t est généralement 1, 2 ou 3, signifiant que « la norme L2 de l'innovation est à plus de g_t sigmas résiduels. » Ce qui indique généralement un capteur défaillant, car un tel phénomène a respectivement 32%, 5% ou 0,2% de probabilité de se produire pour $g_t = 1{,}2$ ou 3. Puisque nous avons redondance des mesures, une mesure correcte est suffisante. Si aucune mesure est prise en compte, la variance de l'erreur P_t^j augmente à chaque instant image, assez pour à un moment permettre à une mesure de passer au travers de l'étape de la réjection.

Ce modèle de réjection n'a qu'un désavantage, c'est qu'il se repose sur la modélisation cartésienne de l'état articulaire humain, et donc il ne bénéficie pas d'un nombre de degrés de liberté restreint et adapté. Mais en pratique, puisque les reconstructions que nous exploitons s'y conforment généralement, les résultats après filtrage demeurent approchant. En contrepartie, cette modélisation ne nécessite aucune conversion depuis le format fourni par le détecteur, et la paramétrisation reste simple.

D. Implémentation

1. Sous-résolution

On peut noter que la pleine résolution capteur de 640×480 n'a pas été utilisée dans l'implémentation, au contraire de (Shotton, et al., 2011). Nous utilisons une sous-résolution finale de 160×120, ce qui correspond à une sous-résolution d'un quart de la résolution source. Cet ajustement permet accélérer les calculs d'un facteur 16 afin de rester super-temps réel.

2. Note sur les flottants

De plus, au lieu de maximiser la probabilité $\mathbb{P}\left(\{S_t^k\}_{k\in K} \middle| Y_t^{l_t}\right)$, l'algorithme de Viterbi, tel que présenté précédemment, est appliqué tel-quel, minimisant pour le même effet, $\sum_{k\in K} \frac{(D_t(k,l)-\mu)^2}{\sigma^2}$ (car la fonction ln est strictement croissante), mais plus rapidement et sans dépassement de la capacité des nombre flottants par valeur inférieure. En effet, avec 45 degrés de liberté, les densités de probabilités (notamment dynamique) atteignaient des valeurs inférieures à $2{,}22507.\,e-308$, arrondies à 0. Cet arrondissement provoquait alors

une singularité dans les calculs de probabilités par produits. Nous avons transposé ce principe au modèle dynamique car il utilise lui aussi une probabilité gaussienne.

E. Critère pour les évaluations

Nos bases de données vidéo (Chapitre II:D) et la vérité terrain associée nous permettent de quantifier et comparer les performances de toute approche de capture de mouvement.

Plusieurs critères sont privilégiés pour caractériser une capture de mouvements humains. Sa précision vient à l'esprit immédiatement, mais la disponibilité, c'est-à-dire la proportion de temps où une mesure est donnée par le détecteur, est tout aussi importante. Comment juger de la qualité d'une mesure, et quelle précision donner à une absence d'estimation de la part du détecteur ?

L'appréciation de la mesure des distances spatiales dans le temps n'est pas simple. La qualité d'une mesure squelette dépend des nœuds examinés, savoir la position des épaules est utile pour la lecture d'émotion, les mains, la manipulation. Nous nous sommes concentrés sur les valeurs obtenues pour les extrémités : mains, pieds, tête.

Pour pouvoir comparer les différentes approches, nous avons donc calculé la quotité d'images $Q_j(d) = \frac{1}{n_{images}} \sum_{images} \begin{cases} 1 \; si \; \|X_j - Z_j\| < d \\ 0 \; sinon \end{cases}$ où une articulation j se trouve à moins d'une certaine distance de la vérité terrain Z_j. En faisant varier la distance d, nous obtenons des courbes cumulatives. Ce critère est directement inspiré du « *mean Average Precision* » de (Shotton, et al., 2011).

Présentées sous forme de radar, nous pouvons apprécier d'un seul regard la précision globale par l'aire des courbes, et en regardant la taille sous chaque branche, se focaliser sur une articulation. La Figure 21 montre un exemple de tel radar avec légende.

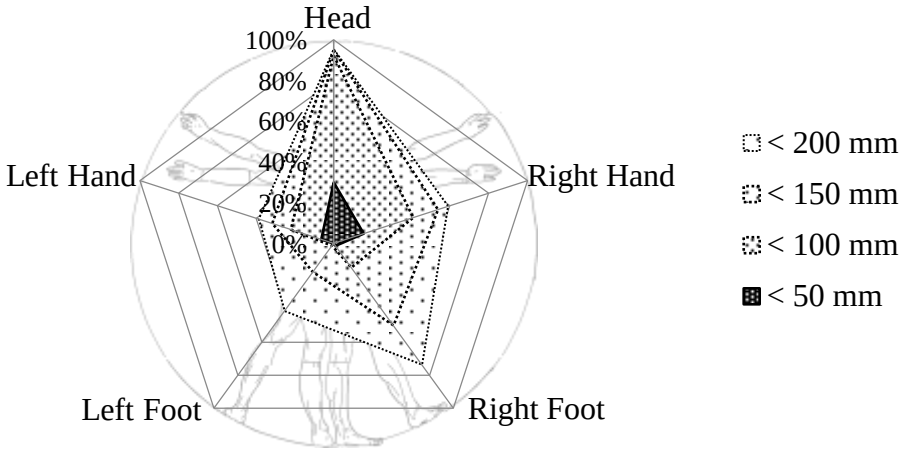


Figure 21 Exemple de radar de précision.

Puisque nous avons les valeurs tirées de notre expérience présentée ci-après, nous pouvons d'ores et déjà aussi émettre une évaluation quantitatives sur les performances

mono-Kinect du détecteur de (Shotton, et al., 2011). En effet, bien que ce détecteur de postures 3D s'avère rapide et suffisamment précis la plupart du temps, certaines configurations personne/Kinect donnent lieu à des reconstructions clairement erronées. Ici, le détecteur n'est pas idéalement positionné : le côté gauche de la personne n'est pas visible, l'auto-occultation du côté gauche se voit avec une moins grande précision sur les articulations de ce côté. Notre approche, en combinant plusieurs Kinects, prend alors pleinement son sens. Le chapitre suivant présente les évaluations associées.

F. Conclusion

Nous avons prototypé, sur la base du détecteur de posture OpenNI/NiTE, un système avancé de reconstruction multi-Kinects et filtrage spatiotemporel par lissage trajectoriel.

Nous avons ici tiré parti de concepts « tracking by detection » et « lisseur trajectoriel » qui ont été éprouvés en analyse vidéo 2D, combinés de manière innovante. Nous avons rajouté comme autres hypothèses explorées, un filtrage de Kalman à réjection de mesures, à raison d'un par articulation.

Pour quantifier les performances du système ainsi formé, nous choisissons d'utiliser la distance des nœuds à la vérité terrain dans notre base acquise, et décrite précédemment. Les résultats obtenus, et les discussions associées, font l'objet du chapitre suivant.

Chapitre IV: Évaluations de notre approche de capture de mouvements

Introduction

Dans ce chapitre, nous allons étudier les performances de notre approche (et ses variantes) formalisé au chapitre précédent, avec deux des expériences faites durant cette thèse, présentées dans le Tableau 5 du Chapitre II:D, et nommées 3KAL35 et 5KAL35.

Une évaluation préliminaire avec 3 Xtions a été faite. L'acquisition a été réalisée sur une unique machine permettant de réduire le nombre de synchronisations temporelles à effectuer. Cependant, 3 capteurs était le nombre maximal sur cette machine. Il s'agit de l'acquisition notée 3KAL35. Et fait l'objet de la première section. Nous évaluons les résultats des capteurs pris séparément pour comparaison mutuelle, et ainsi mettre en lumière les bonnes configurations spatiales mono-capteur.

La section B porte sur une seconde expérimentation illustrant la mise à l'échelle de notre plateforme multi-Kinects. L'évaluation sur 3 Kinects montrée précédemment ayant montré des gains probants, multiplier les Kinects est clairement payant, mais jusqu'où pouvons-nous percevoir des gains ? Quelles sont les configurations spatiales multi-capteurs les plus pertinentes ? Nous avons donc poursuivi cette mise à l'échelle avec ici 5 Kinects. C'est ainsi que nous avons souhaité acquérir et traiter la base 5KAL35, ne voulant pas tester tout d'abord 4 Kinects, afin de retrouver une configuration approchant celle de 3KAL35 avec un capteur de face et deux derrière.

Ces deux premières sections qui se concentrent sur une seule base contiennent chacun une analyse et discussion des valeurs obtenues, et le protocole associé.

Enfin, la section C conclut ce chapitre.

A. Evaluations préliminaires (3 capteurs maxi)

Une évaluation préliminaire avec 3 Xtions a été faite. L'acquisition a été réalisée sur une unique machine permettant de réduire le nombre de synchronisations temporelles à effectuer. Cependant, 3 capteurs était le nombre maximal sur cette machine. C'est l'acquisition 3KAL35 dans le Tableau 5 du Chapitre II:D.

Nous évaluons les résultats des capteurs seuls afin de nous y comparer, et de les comparer entre eux afin de mettre en évidence les faiblesses de chacun.

1. Réglage des paramètres libres

Nous avons fixé les valeurs des paramètres libres mentionnés dans la formalisation (voir Chapitre III:A) de la manière suivante.

σ_q et σ_r sont respectivement fixés grâce aux mouvements inter-images des articulations et la prise en compte de la précision du système. k est fixé empiriquement mais varie avec le frame-rate, comme σ_q. Nous avons fixé $M = 30$ *a priori* pour que la fenêtre de temps fasse 1 seconde, mais plus important, une fenêtre temporelle (et glissante) permet de lisser efficacement la trajectoire inférée.

Les paramètres ainsi réglés sont énumérés dans le Tableau 6 :

Symbole	*Signification*	*Valeur*
K	*Nombre de Kinects*	*2..5*
M	*Taille de la fenêtre temporelle*	*30*
(σ_q, σ_r)	*Variances du filtre de Kalman*	*(100,200)*
k	*Coefficient de dégradation de vitesse*	*0.8*
(σ_o, σ_d)	*Variance dans les modèles d'observation et dynamique*	*(0.1,*)*
g_t	*Seuil de réjection dans le filtre de Kalman*	*2*

Tableau 6 Valeurs des paramètres libres pour notre implémentation.

* :depend de l'articulation: head/neck/torso/shoulders: 5; elbows/hips/knee: 10; hands/feet: 20

La valeur de 0.8 dans (EM) est totalement arbitraire. Le but d'avoir une valeur inférieure à 1 est de rendre le système stable. La valeur de 100 mm correspond à une estimation de la vitesse maximale de déplacement de l'extrémité d'un membre. En fait la valeur dépend de l'articulation dans l'implémentation. La valeur de 200 mm correspond à la précision, car c'est la précision maximale attendue pour un de nos capteurs.

Ce réglage empirique des paramètres n'a pas fait l'objet d'une étude approfondie pour étudier leur impact, positif comme négatif, sur les performances.

Des vidéos d'illustration sur les résultats qui suivent sont accessibles à l'adresse suivante :

http://homepages.laas.fr/lerasle/hmc/videos.php

2. Nomenclature

Nous présentons ici les résultats obtenus sur la base 3KAL35 mentionnée au Chapitre II:D. Les différents jeux de données de la Figure 22 sont nommés en gras dans ce qui suit.

Les capteurs numérotés 1, 2, et 3 étaient placés en triangle autour de la personne, respectivement, en arrière à gauche, de profil à droite et en face légèrement sur la droite, du point de vue de l'acteur. C'est la même numérotation que sur la Figure 20.

Dans cette première expérience, nous avons effectué plusieurs essais et utilisé des méthodes moins raffinées au premier abord. Nous rappelons que notre méthode utilise un lissage trajectoriel et une mesure supplémentaire de type Kalman à réjection dont les mesures sont les mesures capteur.

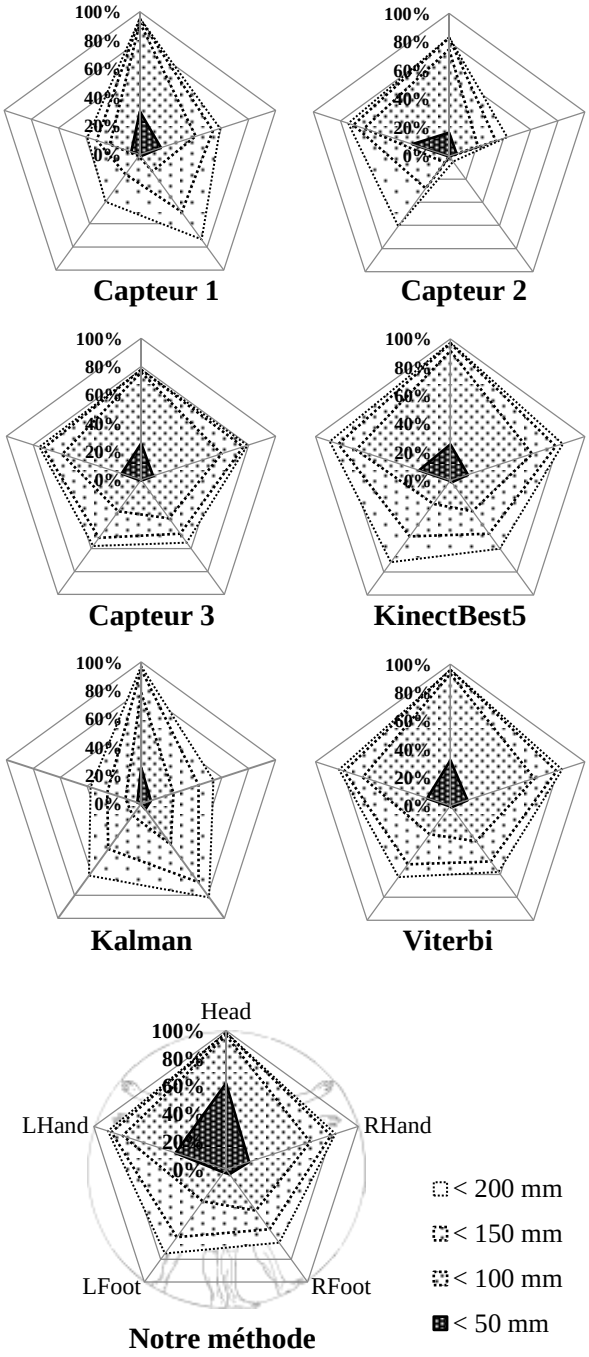


Figure 22 Figures radar de la première expérience.

Nous avons d'abord essayé une approche de type Viterbi seul. Cette approche était donc un sélecteur de Capteur, qui se basait sur un historique de positions, et la vraisemblance de chaque hypothèse dans tous les points de vue au même instant. Le résultat était la position à la fin d'une trajectoire la plus probable au sens dynamique et à chaque instant.

Nous avons comparé cette approche à une fausse approche qui sélectionne automatiquement le squelette avec la plus petite erreur cumulée aux 5 articulations qui sont les extrémités corporelles. Cet « oracle » est baptisé KinectBest5.

Enfin, nous avons aussi implémenté un filtre de Kalman basique (sans réjection de mesure) afin de donner une base de comparaison dans la théorie de filtrage.

Toutes ces différentes approches prototypes de la solution finale se trouvent donc dans les résultats ci-après.

	< 50 mm					< 100 mm				
Capteur 1	9,3%					29,9%				
Head RHand RFoot LFoot LHand	25%	14%	1%	0%	6%	80%	36%	12%	1%	20%
Capteur 2	8,4%					28,3%				
Head RHand RFoot LFoot LHand	14%	5%	0%	0%	23%	65%	18%	0%	5%	53%
Capteur 3	8,9%					43,7%				
Head RHand RFoot LFoot LHand	24%	8%	0%	0%	12%	67%	51%	28%	23%	50%
KinectBest5	11,4% (+2,1%)					47,1% (+3,4%)				
Head RHand RFoot LFoot LHand	24%	11%	1%	0%	20%	82%	53%	22%	18%	61%
Kalman	7,0% (-2,3%)					27,7% (-15%)				
Head RHand RFoot LFoot LHand	21%	6%	4%	0%	3%	69%	21%	30%	9%	11%
Viterbi	11,5% (+2,2%)					48,3% (+4,6%)				
Head RHand RFoot LFoot LHand	29%	11%	1%	1%	15%	84%	54%	26%	21%	56%
Notre méthode	21,7% (+12,4%)					53,0% (+4,7%)				
Head RHand RFoot LFoot LHand	55%	15%	3%	2%	33%	87%	56%	31%	24%	68%
	+30%	+1%	+2%	+2%	+10%	+37%	+5%	+5%	+1%	+15%

	< 150 mm					< 200 mm				
Capteur 1	43,0%					53,2%				
Head RHand RFoot LFoot LHand	83%	48%	42%	15%	28%	84%	52%	62%	35%	34%
Capteur 2	37,2%					45,7%				
Head RHand RFoot LFoot LHand	72%	29%	1%	24%	59%	73%	36%	5%	52%	63%
Capteur 3	55,5%					59,2%				
Head RHand RFoot LFoot LHand	69%	65%	39%	42%	62%	69%	68%	46%	48%	65%
KinectBest5	62,0% (+5,5%)					70,3% (+11,1%)				
Head RHand RFoot LFoot LHand	87%	69%	40%	42%	72%	88%	73%	51%	61%	78%
Kalman	46,8% (-8,7%)					57,9%(-2,7%)				
Head RHand RFoot LFoot LHand	83%	37%	58%	34%	23%	85%	47%	70%	53%	33%
Viterbi	62,1% (+5,6%)					67,2% (+8%)				
Head RHand RFoot LFoot LHand	87%	69%	42%	44%	69%	87%	72%	50%	54%	72%
Notre méthode	66,4% (10,9%)					72,2% (+13%)				
Head RHand RFoot LFoot LHand	89%	70%	46%	52%	76%	89%	72%	56%	65%	78%
	+6%	+5%	+4%	+10%	+14%	+5%	+4%	-6%	+13%	+13%

Tableau 7 Statistiques complets relatifs aux données 3KA35.

3. Diagrammes radar

On note sur les graphes que les Capteurs 1, 2, et 3 ont des performances variables : quasiment 100% de la tête à moins de 100mm pour 1, mais 20% restent au-delà de 200mm pour 2 et 3. KinectBest5 semble former une enveloppe convexe des 3 graphes Capteur. Les valeurs de Viterbi semblent quasiment identique à KinectBest5, l'aire du polygone <50mm semble égale mais de forme différente : les performances se sont redistribués différemment selon les articulations. Kalman présente des valeurs semblables à Capteur 1, mais une précision <50mm moins élevée. Enfin, Notre méthode atteint 100% de l'articulation tête à <100mm, dont 60% à <50mm, et des valeurs proches de 90% pour la main gauche à <150mm, et 80% pour la droite et le pied gauche. Tous les résultats sont listés dans le tableau ci-après :

4. Tableaux de résultat

Sur le Tableau 7, nous pouvons voir les valeurs pour chaque articulation, de gauche à droite : Head, RHand, RFoot, LFoot, LHand, et surplombant, la moyenne arithmétique de ses 5 valeurs, pour chaque approche/détecteur énuméré précédemment.

Sur les trois premières lignes, en gras on trouve le résultat du meilleur détecteur mono-Kinect®. Sur les lignes suivantes, précédée du signe + (ou -), se trouve la différence à ce meilleur détecteur (pour des raisons de place, ceci a été omis pour les articulations uniquement pour quelques détecteurs). En gras se trouve la meilleure amélioration parmi les fusions multi-Kinect®.

5. Analyse des résultats

a) Performances de NiTE

En comparant les relevés des capteurs entre eux, la supériorité de la 3ième mesure apparaît clairement. Cet état de fait permet de vérifier que la position optimale pour un capteur seul se situe face à l'acteur, et donc pour des mouvements fronto-parallèles. La position de profil provoque la dégradation, voir la perte totale, des membres occultés par le reste du corps : le côté droit pour le capteur 2 et dans une moindre mesure le côté gauche pour le capteur 1.

Les performances relevées lors de notre expérience sont en-deçà de la littérature (Shotton, et al., 2011), qui sont de 80% des mesures à moins de 200mm, même pour le Capteur 3 qui est dans une position optimale (59,2%). Nous imaginons que leur base de donnée de travail, notamment fortement orienté fronto-parallèle, devait être plus adaptée à leur approche, ou que notre vérité terrain diffère et est plus stricte, ou encore que le point de vue est trop plongeant.

En comparaison brute à un filtre de Kalman simple, ce dernier ne tient pas la route, excepté pour la tête, où il rejoint le capteur 1, leader pour cette articulation parmi les capteurs.

b) Selection d'un capteur

Essayons de vérifier si la reprojection de silhouettes dans plusieurs points de vue est un bon critère de sélection de l'hypothèse avec la plus petite distance cumulée aux 5 articulations.

Pour des comparaisons de performance, nous avons l'oracle KinectBest5 qui, rappelons-le, correspond à la mesure du capteur constituant le meilleur choix en termes de distance cumulée.

Avec un algorithme de Viterbi simple, nous obtenons le résultat Viterbi, qui arrive à légèrement surpasser l'oracle dans les distances courtes (11,5%, 48,3%, et 62,1% contre respectivement 11,4%, 47,1%, et 62,0% pour Best5), mais est plus en retrait sur les distances de 150mm à 200mm (67,2% contre 70,3%). Il nous semble que les deux approches se valent alors, l'une échangeant de la précision contre de la disponibilité de la mesure.

Ainsi, nous concluons qu'il est possible, grâce à une simple sélection basée sur plusieurs points de vue (et la dynamique liée à une trajectoire plausible) de trouver la meilleure position squelette parmi N lorsque des points de vue complémentaires sont fournis.

c) Apport du filtrage de Kalman

Notre méthode combine donc une approche de type Viterbi avec une hypothèse squelette supplémentaire. Ce type d'approche a été envisagé suite à la conclusion précédente, car cela ne pouvait qu'améliorer ou modifier le type de performance de l'estimateur basé sur la programmation dynamique.

La méthode conserve, et amplifie même la performance globale : on atteint maintenant 72,2% d'images à moins de 200mm au lieu des 70,3% de l'oracle, la meilleure performance jusqu'alors.

Le gain le plus significatif se révèle pour les toutes petites distances : de 11,5% à 21,7% pour 50mm, de 48,3% à 63% pour 100mm et de 62,1% à 66,4% pour 150mm. La proportion de mesures précises à 50mm a donc presque doublée par rapport à l'approche basée Viterbi seule.

Ces performances confirment la précédente conclusion, et de plus démontrent qu'un filtrage de Kalman avec réjection de mesure, bien qu'une hypothèse requérant peu de calcul, permet d'obtenir des mesure souvent plus efficaces que toutes les mesures des capteurs. Notamment pour la tête, ou la proportion d'images à moins de 50mm a doublé en ajoutant le Kalman.

6. Conclusion préliminaire

Les résultats que nous obtenons sont probants. Nous nous positionnons au-dessus des performances des détecteurs pris séparément mesurées sur le même dataset. Nous avons réalisé deux publications pour montrer ces résultats : (Masse, Lerasle, Devy, Monin, Lefebvre, & Mas, Human Motion Capture Using Data Fusion of Multiple Skeleton Data, 2013) et (Masse, Lerasle, Devy, Monin, Lefebvre, & Mas, Capture de Mouvements Humains par Fusion de Multiples Données Squelettes, 2014).

Cependant, il nous semble opportun de valider ces performances par des évaluations à plus large échelle, donc sur la seconde base, nommée 5KAL35 mentionnée au Chapitre II:D. Dans le chapitre qui suit, nous allons étudier l'impact du placement et du nombre de Kinects.

B. Evaluations à large échelle

L'utilisation de 3 capteurs a été probante (voir Chapitre IV:A) mais la multiplication des détecteurs est coûteuse en matériel, ressources CPU et espace. Afin de mesurer l'impact du nombre et du placement des capteurs autour de l'utilisateur, nous allons acquérir à partir de 5 capteurs disposés uniformément, et analyser les performances de notre approche en prenant en compte des combinaisons de 2, 3 capteurs… etc. jusqu'aux 5 capteurs ensemble.

Une des caractéristiques voulues du système était sa robustesse. En conséquence, quand une erreur a corrompu le flux d'un capteur qui s'en est retrouvé décalé, nous avons gardé l'acquisition afin de mesurer l'impact d'une mesure squelette erronée, plutôt que de refaire l'acquisition avec 5 capteurs parfaitement opérationnels.

Des vidéos d'illustration sont accessibles à l'url :

http://homepages.laas.fr/lerasle/hmc/videos.php

Le coût CPU n'a ici pas diminué par rapport à la première expérience suivant le nombre de capteurs. En effet, le coût CPU provient de la comparaison dans les points de vue des squelettes, et est donc en $O(card(K)^2)$. Une optimisation permet de soustraire à la valeur de $card(K)$ l'ensemble des Kinects ne voyant pas d'utilisateur, ce qui permettrait de rajouter des Kinects à champ disjoint sans surcoût CPU.

Vu la taille conséquente des évaluations, nous avons utilisé des abréviations que nous allons d'abord présenter avant les évaluations, suivies des discussions associées, et enfin présenter leurs conclusions.

1. Nomenclature

L'expérience comporte des résultats combinant de 1 à 5 capteurs positionnés de manière circulaire et équitablement espacés autour de l'acteur comme le montre la Figure 12. Un des cinq capteurs était face à l'acteur. Il est donc désigné comme Frontal, abrégé en F. Ceux en face à droite et à gauche de l'acteur sont donc Front Left et Front Right, abrégés en FL et FR respectivement. Ceux de derrière Back Left et Back Right (BL et BR.)

Both Frontal sont les deux capteurs à gauche et à droite de Frontal. All Frontal est l'ensemble des trois capteurs faisant face à l'acteur. Left Bearing est l'ensemble des capteurs à gauche y compris Frontal, respectivement Right Bearing pour la droite.

Pour faciliter la lecture, des balles se trouvent dans le coin du diagramme radar où la camera se situait vis-à-vis de l'utilisateur, en vue de dessus. L'utilisateur faisait face au haut.

2. Diagrammes radar

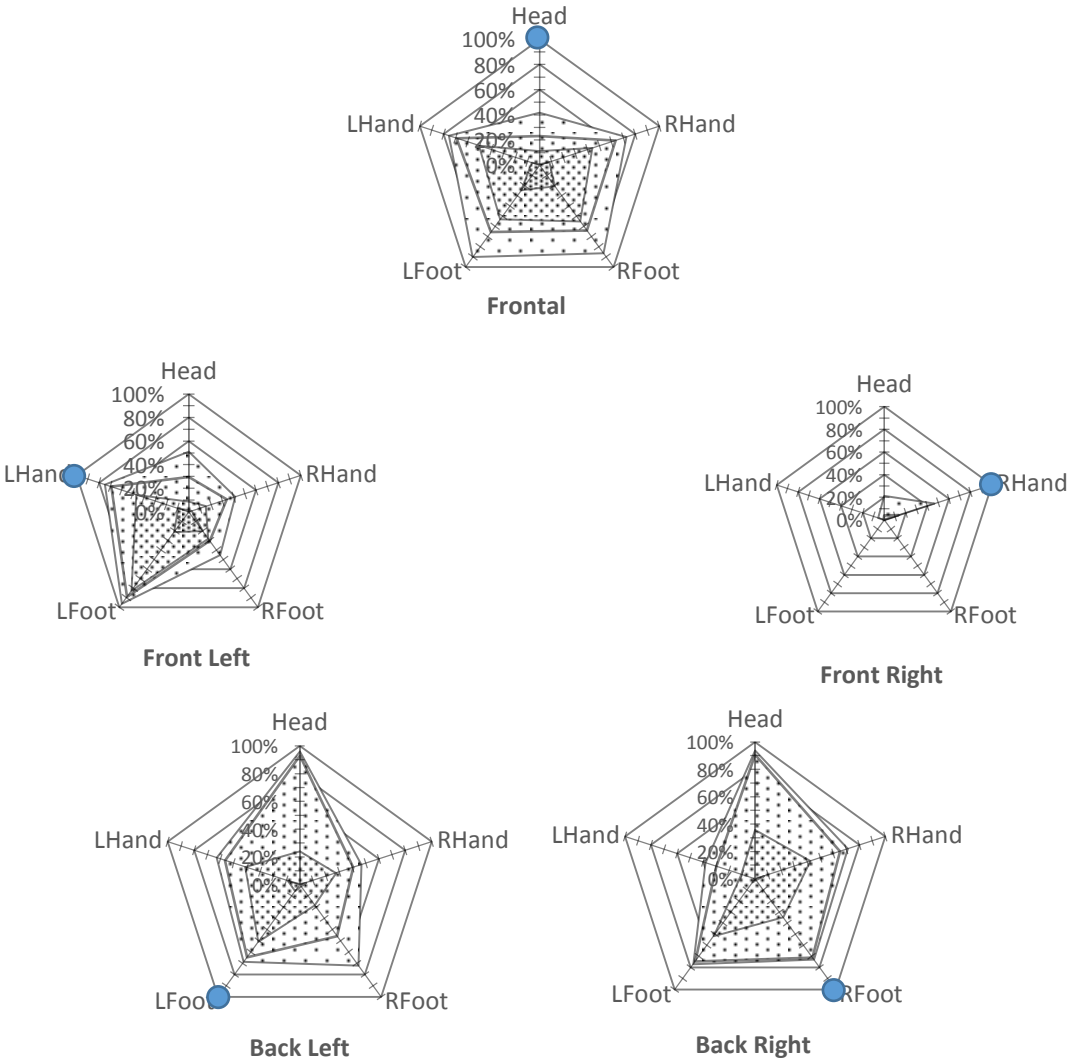


Figure 23 Figures radar de la seconde expérience.

Voici tout d'abord les résultats de chaque capteur pris indépendamment. On remarque que la forme du diagramme dépend du positionnement. Deux capteurs symétriques ont un diagramme symétrique aussi. Par exemple, la main à 40% de précision est la gauche (resp. droite) pour Back Right (resp Left). La tête est assez mal perçue par le devant, par plus de 50% même à <200mm. Enfin aucun diagramme ne donne des valeurs inférieures à 50mm pour la tête, plus de 10% pour les mains ou plus de 25% pour les pieds.

On remarque que les valeurs de Front Right sont totalement abyssales.

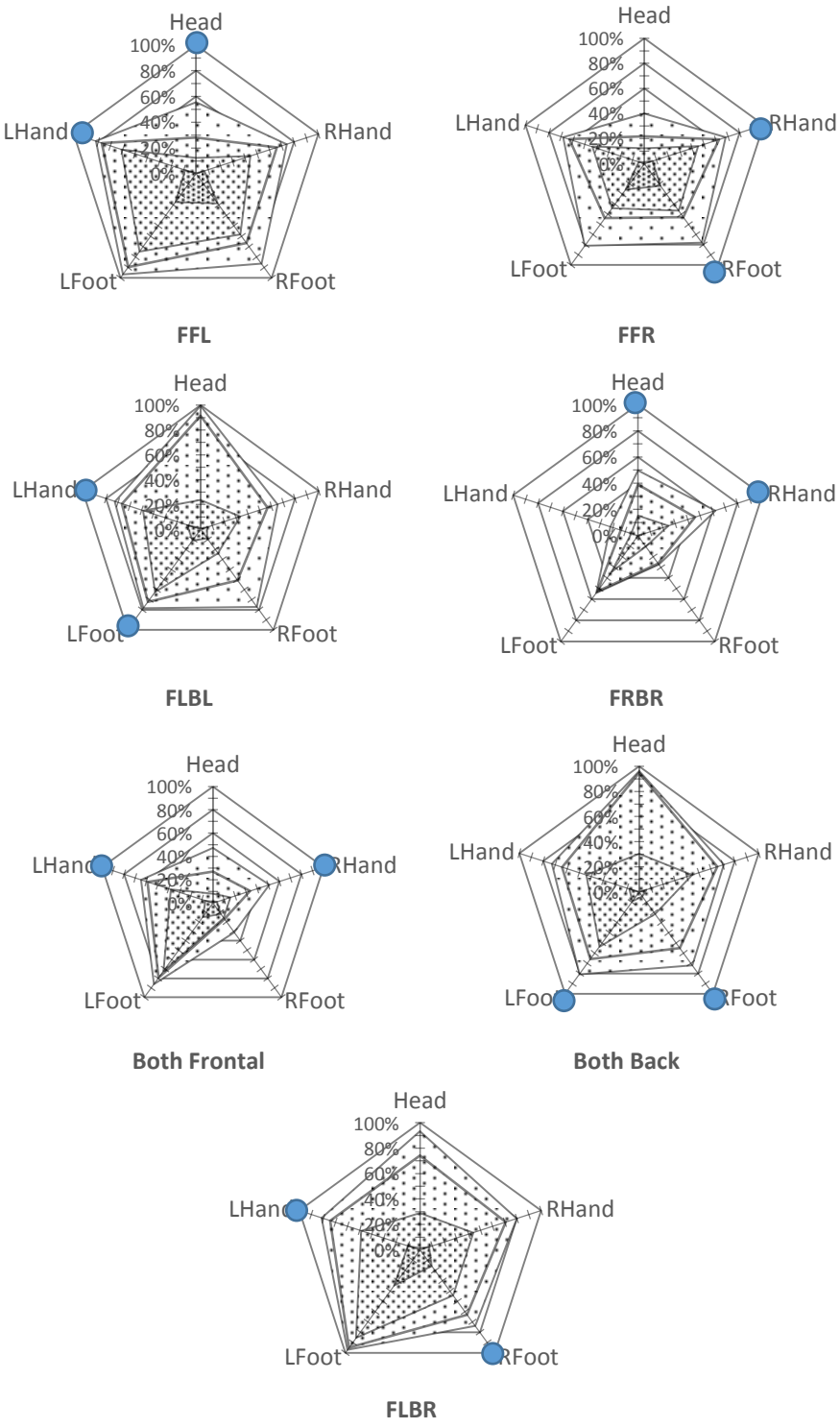


Figure 23 (suite) Figures Radar de la seconde expérience. (Configurations 2 Kinects)

Nous voyons ici des résultats plus probants pour des configurations à deux capteurs. Par exemple la configuration FFL atteint 10% pour les mains et 30% pour les pieds à moins de 50mm de leurs positions MoCap. FLBR, Both Back et FLBL atteignent 90, 95 et 100% à <200mm pour la tête. Ce dernier atteint d'ailleurs les meilleures performances globales pour cette distance parmi les performances avec 2 capteurs, ainsi que <150mm mais la performance semble équivalente à Both Back.

On remarque encore que les données fournies par Front Right sont mauvaises, mais les performances avec deux capteurs, surtout FFR, ne sont pas aussi fortement fausses que le capteur défectueux. En fait le diagramme de FFR est équivalent à celui de Front.

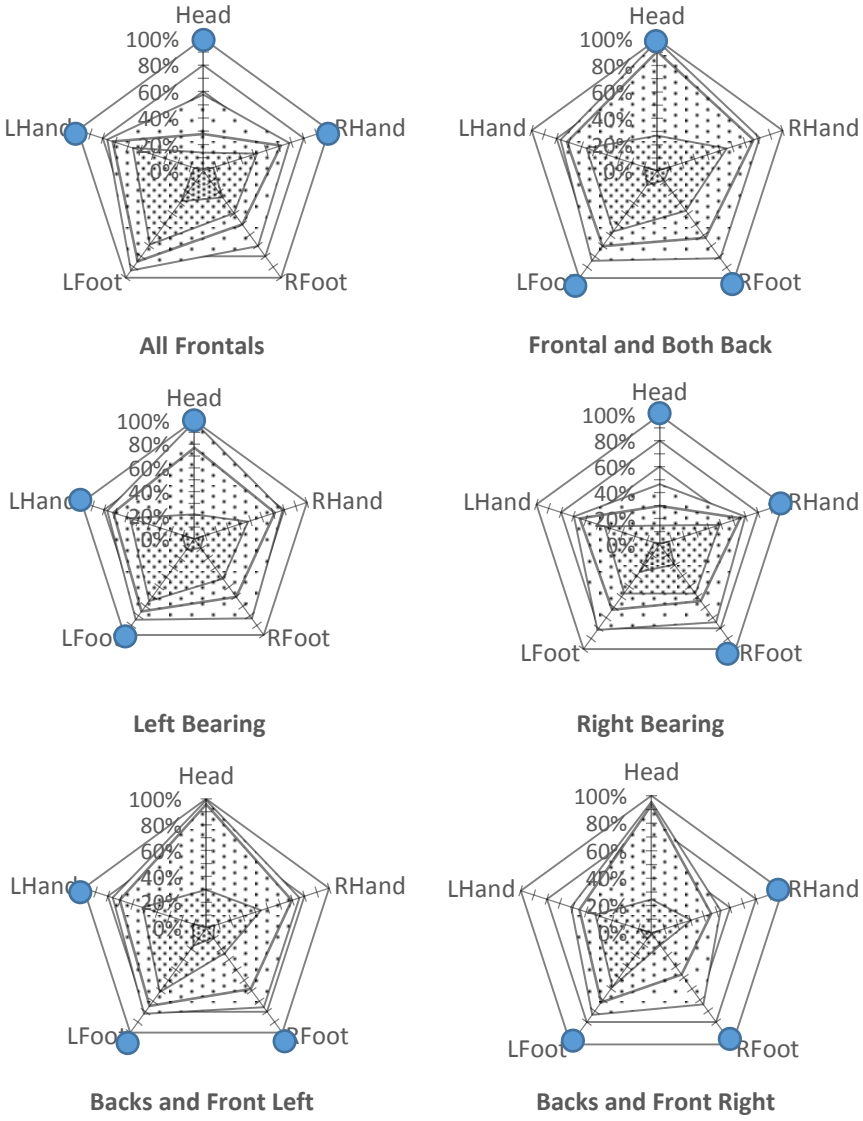
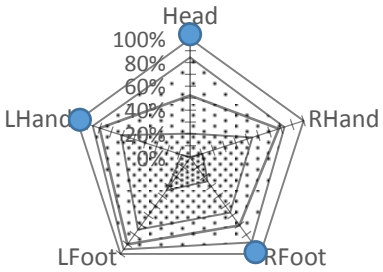


Figure 23 (suite) Figures Radar de la seconde expérience. (Configurations 3 Kinects)

Ici quelques approches avec 3 flux capteurs. On obtient des radars similaires à la première expérience (voir Chapitre IV:A) pour la première fois, mais le positionnement Frontal and both back, pourtant le plus similaire à 3KAL35 (Chapitre II:D), n'est pas le meilleur, c'est Backs and Front Left, avec 80% à 200mm sur les extrêmités des membres et pratiquement 100% sur la tête jusqu'à moins de 150mm, et pas moins de 70% sur les autres extrêmités à l'exception du pied droit (60%).

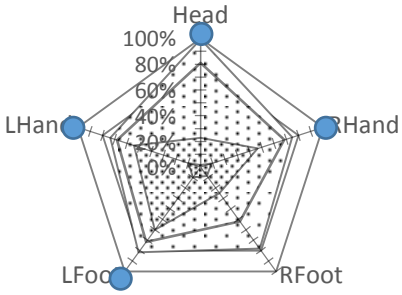
La meilleure quantité d'images à <50mm s'obtient avec All Frontals et FFLBR, avec un léger avantage pour ce dernier. En effet il détecte sur légèrement plus de 10% des images contre légèrement moins de 10%.



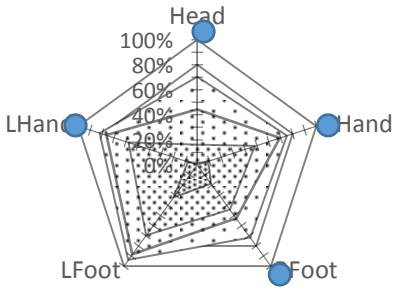
FFLBR

Figure 23 (suite) Figures Radar de la seconde expérience. (Configuration 3 Kinects)

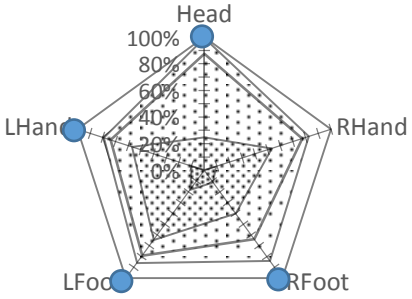
C'est aussi FFLBR qui a la perception à 200mm la plus équilibrée, en effet, la précision se borne entre 95% et un minimum de 80%.



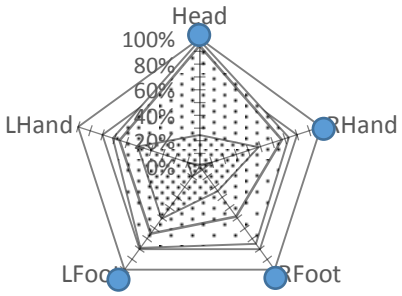
All except Back Right



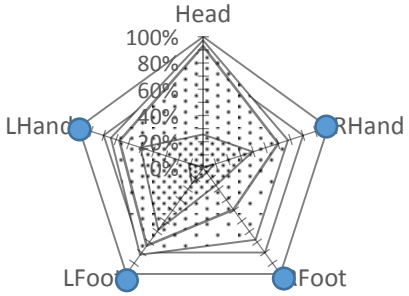
All except Back Left



All except Front Right



All except Front Left



All except Front

Figure 23 (suite) Figures Radar de la seconde expérience. (Configurations 4 Kinects)

Voici les figures radars à 4 et 5 capteurs. All except Front Right obtient les mêmes valeurs à 200 et 150mm que Backs and Front Left, c'est-à-dire la deuxième meilleure valeur des approches à 3 capteurs. L'ajout de Front a eu un effet négatif. En revanche on note que la précision à 50mm est plus élevée que Back and Front Left. La comparaison est donc difficile.

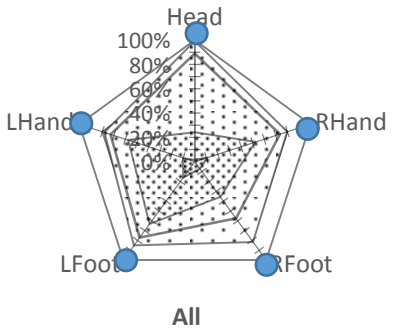


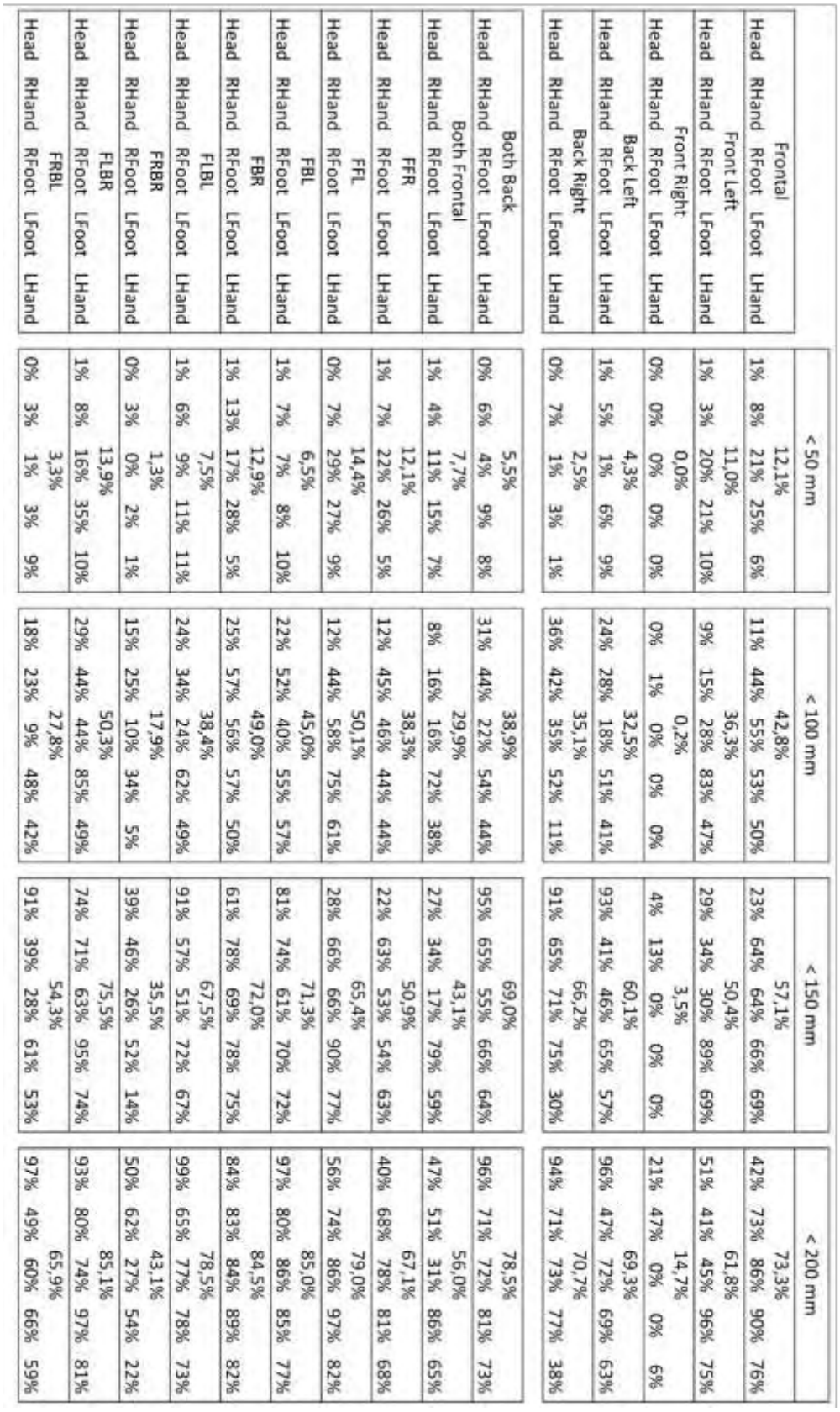
Figure 23 (suite) Figures Radar de la seconde expérience. (Configurations 4 et 5 Kinects)

On peut noter que sur All except Back Left, la performance de la tête est la pire de toutes les combinaisons de 4 capteurs, avec seulement 70% <200mm et 45% <150mm. Comme pour compenser, la précision au niveau du pied gauche est elle la plus importante, avec 90% à <150mm et 95% à <200mm. La valeur à <50mm est aussi la meilleure, peut-être est-ce lié au fait que Back Left n'était pas parmi les meilleurs détecteurs mono-capteur.

La configuration capteurs All avec tous les capteurs montre des performances similaires à All except Front Right. Cette dernière n'est d'ailleurs inférieure à All except Front Left que sur la tête, avec 90% de tête <150mm à la vérité terrain, contre 95%, mais est supérieure sur toutes les autres articulations et distances.

Avant de passer à la discussion nous affichons les valeurs détaillées (numériques) dans la Tableau 8 afin de compléter les statistiques obtenue sur les résultats de cette expérience.

3. Tableaux de résultat



					< 50 mm	< 100 mm	< 150 mm	< 200 mm
Frontal					12,1%	42,8%	57,1%	73,3%
Head	RHand	RFoot	LFoot	LHand	1% 8% 21% 25% 6%	11% 44% 55% 53% 50%	23% 64% 64% 66% 69%	42% 73% 86% 90% 76%
Front Left					11,0%	36,3%	50,4%	61,8%
Head	RHand	RFoot	LFoot	LHand	1% 3% 20% 21% 10%	9% 15% 28% 83% 47%	29% 34% 30% 89% 69%	51% 41% 45% 96% 75%
Front Right					0,0%	0,2%	3,5%	14,7%
Head	RHand	RFoot	LFoot	LHand	0% 0% 0% 0% 0%	0% 1% 0% 0% 0%	4% 13% 0% 0% 0%	21% 47% 0% 0% 6%
Back Left					4,3%	32,5%	60,1%	69,3%
Head	RHand	RFoot	LFoot	LHand	1% 5% 1% 6% 9%	24% 28% 18% 51% 41%	93% 41% 46% 65% 57%	96% 47% 72% 69% 63%
Back Right					2,5%	35,1%	66,2%	70,7%
Head	RHand	RFoot	LFoot	LHand	0% 7% 1% 3% 1%	36% 42% 35% 52% 11%	91% 65% 71% 75% 30%	94% 71% 73% 77% 38%
Both Back					5,5%	38,9%	69,0%	78,5%
Head	RHand	RFoot	LFoot	LHand	0% 6% 4% 9% 8%	31% 44% 22% 54% 44%	95% 65% 55% 66% 64%	96% 71% 72% 81% 73%
Both Frontal					7,7%	29,9%	43,1%	56,0%
Head	RHand	RFoot	LFoot	LHand	1% 4% 11% 15% 7%	8% 16% 16% 72% 38%	27% 34% 17% 79% 59%	47% 51% 31% 86% 65%
FFR					12,1%	38,3%	50,9%	67,1%
Head	RHand	RFoot	LFoot	LHand	1% 7% 22% 26% 5%	12% 45% 46% 44% 44%	22% 63% 53% 54% 63%	40% 68% 78% 81% 68%
FFL					14,4%	50,1%	65,4%	79,0%
Head	RHand	RFoot	LFoot	LHand	0% 7% 29% 27% 9%	12% 44% 58% 75% 61%	28% 66% 66% 90% 77%	56% 74% 86% 97% 82%
FBL					6,5%	45,0%	71,3%	85,0%
Head	RHand	RFoot	LFoot	LHand	1% 7% 7% 8% 10%	22% 52% 40% 55% 57%	81% 74% 61% 70% 72%	97% 80% 86% 85% 77%
FBR					12,9%	49,0%	72,0%	84,5%
Head	RHand	RFoot	LFoot	LHand	1% 13% 17% 28% 5%	25% 57% 56% 57% 50%	61% 78% 69% 78% 75%	84% 83% 84% 89% 82%
FLBL					7,5%	38,4%	67,5%	78,5%
Head	RHand	RFoot	LFoot	LHand	1% 6% 9% 11% 11%	24% 34% 24% 62% 49%	91% 57% 51% 72% 67%	99% 65% 77% 78% 73%
FRBR					1,3%	17,9%	35,5%	43,1%
Head	RHand	RFoot	LFoot	LHand	0% 3% 0% 2% 1%	15% 25% 10% 34% 5%	39% 46% 26% 52% 14%	50% 62% 27% 54% 22%
FLBR					13,9%	50,3%	75,5%	85,1%
Head	RHand	RFoot	LFoot	LHand	1% 8% 16% 35% 10%	29% 44% 44% 85% 49%	74% 71% 63% 95% 74%	93% 80% 74% 97% 81%
FRBL					3,3%	27,8%	54,3%	65,9%
Head	RHand	RFoot	LFoot	LHand	0% 3% 1% 3% 9%	18% 23% 9% 48% 42%	91% 39% 28% 61% 53%	97% 49% 60% 66% 59%

Tableau 8 Statistiques complètes relatives aux données 5KAL35.

						< 50 mm	< 100 mm	< 150 mm	< 200 mm
All Frontals						14,0%	44,2%	59,3%	73,3%
Head	RHand	RFoot	LFoot	LHand		2% 8% 25% 28% 7%	14% 43% 40% 69% 56%	28% 62% 50% 85% 72%	58% 68% 70% 93% 77%
Frontal and Both Back						8,5%	46,4%	74,3%	84,6%
Head	RHand	RFoot	LFoot	LHand		1% 10% 9% 13% 10%	27% 55% 37% 56% 56%	91% 77% 62% 70% 72%	99% 82% 82% 84% 77%
Left Bearing						7,9%	46,3%	71,2%	84,2%
Head	RHand	RFoot	LFoot	LHand		0% 8% 9% 11% 11%	21% 48% 41% 65% 57%	77% 72% 60% 76% 72%	99% 78% 83% 84% 77%
Right Bearing						11,6%	40,3%	55,2%	68,3%
Head	RHand	RFoot	LFoot	LHand		1% 7% 19% 26% 4%	15% 49% 47% 47% 45%	30% 65% 54% 62% 65%	47% 70% 74% 82% 70%
Backs and Front Right						4,0%	31,4%	58,7%	69,4%
Head	RHand	RFoot	LFoot	LHand		0% 4% 1% 6% 9%	25% 30% 11% 48% 43%	93% 46% 37% 62% 55%	96% 53% 64% 73% 61%
Backs and Front Left						9,0%	42,2%	73,8%	81,8%
Head	RHand	RFoot	LFoot	LHand		1% 7% 9% 17% 11%	30% 44% 24% 61% 52%	96% 70% 58% 75% 70%	99% 76% 76% 82% 76%
Fronts and Back Left						5,5%	33,3%	58,9%	71,6%
Head	RHand	RFoot	LFoot	LHand		1% 6% 3% 6% 11%	19% 32% 10% 58% 48%	87% 52% 26% 66% 63%	97% 61% 56% 75% 68%
Fronts and Back Right						10,2%	35,7%	53,1%	62,7%
Head	RHand	RFoot	LFoot	LHand		1% 5% 10% 28% 8%	15% 27% 17% 80% 39%	43% 48% 22% 90% 62%	59% 59% 33% 93% 69%
FFLBR						15,6%	53,7%	74,2%	87,4%
Head	RHand	RFoot	LFoot	LHand		0% 11% 25% 34% 8%	20% 55% 57% 75% 61%	52% 78% 70% 90% 80%	84% 83% 88% 95% 86%
FFRBL						5,5%	34,8%	63,1%	74,5%
Head	RHand	RFoot	LFoot	LHand		1% 7% 4% 5% 10%	25% 39% 16% 46% 48%	90% 59% 43% 62% 62%	99% 64% 71% 71% 67%
All except Front						8,0%	38,1%	66,7%	77,9%
Head	RHand	RFoot	LFoot	LHand		1% 8% 5% 15% 12%	26% 40% 16% 59% 50%	93% 60% 39% 73% 67%	97% 68% 68% 82% 74%
All except Front Right						9,7%	48,0%	76,4%	86,2%
Head	RHand	RFoot	LFoot	LHand		0% 9% 11% 18% 10%	25% 53% 40% 65% 57%	88% 77% 64% 79% 74%	99% 82% 84% 86% 79%
All except Front Left						6,8%	39,7%	68,3%	79,4%
Head	RHand	RFoot	LFoot	LHand		1% 10% 3% 10% 10%	25% 49% 24% 51% 51%	94% 68% 48% 65% 66%	99% 73% 75% 79% 71%
All except Back Left						13,8%	47,2%	66,3%	78,7%
Head	RHand	RFoot	LFoot	LHand		1% 9% 19% 32% 8%	16% 48% 44% 70% 58%	44% 69% 53% 89% 77%	70% 75% 72% 94% 82%
All except Back Right						7,8%	41,8%	68,2%	81,5%
Head	RHand	RFoot	LFoot	LHand		2% 9% 8% 10% 10%	23% 47% 25% 60% 54%	81% 68% 52% 72% 68%	99% 75% 78% 82% 74%
All						9,1%	46,7%	74,1%	84,8%
Head	RHand	RFoot	LFoot	LHand		2% 11% 7% 16% 10%	24% 53% 36% 63% 58%	89% 74% 58% 77% 73%	99% 80% 82% 85% 78%

Tableau 8 (suite) Statistiques complètes relatives aux données 5KAL35.

4. Analyse des résultats

a) Comparaison avec les résultats précédents

Si nous examinons d'abord les résultats de chaque capteur, nous remarquons encore une fois que le positionnement capteur/utilisateur a un impact négatif sur les membres occultés. Les capteurs situés à gauche ont des performances moindre sur le côté droit du corps et vice-versa. Un capteur de face a une précision équilibrée.

On observe bien le mauvais positionnement de la tête dans la performance capteur, et particulièrement le problème mentionné ci-dessus avec le capteur devant à droite.

En comparaison de la première expérience, les capteurs de derrière et celui de face ont des performances semblables au capteur de face de la première expérience. Si on corrige la remarque précédente sur le cadrage de tête, la supériorité du positionnement fronto-parallèle est rétablie. Ainsi, nos évaluations corroborent la littérature, notamment les évaluations de (Shotton, et al., 2011) ce qui prouve que les performances mesurées sur la base précédente 3KAL35 complétaient, sans contredire, les résultats de notre confrère.

Il est à noter que les pieds sont plus précis quand ils sont évalués par les capteurs de devant que les capteurs de derrière sans que cela puisse avoir une explication directe.

b) Étude de la configuration capteurs

Pour ce qui est des approches à deux capteurs, on peut voir des gains probants par rapport aux approches mono-capteurs.

On peut noter par exemple une amélioration systématique des résultats vis-à-vis du capteur le moins bon de la paire, pour toutes les combinaisons de mesures correctes. Plus particulièrement, l'impact de la mesure erronée Front Right est minimal sur Frontal dans FFL, sensible en combinaison avec son homologue Front Left dans Both Frontal, mais dégrade fortement le résultat de Back Right dans FRBR.

Le cas de FFL montre la synergie produite par la méthode entre deux mesures aux performances complémentaires. L'ajout de Front Left apporte la précision sur le pied gauche, et le côté droit est fourni par Frontal. Ce résultat confirme que notre approche permet de fusionner harmonieusement des détecteurs mono-capteur qui, par définition, n'ont qu'une vision partielle de la scène.

De manière surprenante, alors que FLBL est la combinaison des deux capteurs du côté gauche, on voit revenir de la précision sur le côté droit. Cela peut se comprendre car la statistique se fait dans le temps : comme la zone d'auto-occultation du corps humain est faible, les images responsables de la petite précision d'un capteur sont celles où l'autre capteur n'a aucun problème, même avec une faible distance entre les capteurs. C'est le même phénomène que pour Both Back où les mains sont consolidées par la symétrie. La fusion en base (:distance entre les deux capteurs) faible semble donc fonctionner dans une moindre mesure, même si ce n'est pas la configuration typique en capture de mouvements.

Contre toute attente, la meilleure des fusions bi-capteur ne se trouve par contre pas être une combinaison avec Frontal, mais FLBR. Elle reste néanmoins une fusion de capteurs se faisant face. Ainsi, un capteur en position fronto-parallèle n'est pas nécessaire pour de bonnes performances bi-capteur.

Pour ce qui est des traitements tri-capteurs, nous avons enfin un positionnement similaire à la première expérience afin de nous comparer à celle-ci : Frontal and Both Back. Ses performances sont similaires à Back and Front Left et Left Bearing, et forment à eux trois les meilleures performances à trois capteurs. Notons que d'autres combinaisons ont plus de précisions sur certaines parties seulement, par exemple FFLBR qui est meilleur sur les pieds. La combinaison d'emplacements choisie a donc un impact non pas sur la performance globale, mais sur quelle articulation serait la mieux détectée.

Nous pouvons aussi comparer l'impact de performances quand nous ajoutons Front Right à un duo précédent. Par example, Right Bearing et FBR, Back and Front Right et Both Backs, et All Frontals et Both Frontals. L'impact de la mauvaise mesure est donc minimisé par la présence des deux autres bonnes mesures, comme un algorithme de vote.

L'impact de la mauvaise mesure diminue d'ailleurs encore suivant le nombre d'autres bonnes mesures comme le montre All.

Cependant, nous avons une subtile baisse de la qualité de la mesure plus on ajoute des capteurs. Même All except Front Right est inférieure à FFLBR. Ainsi plus de capteurs ne signifie pas toujours mieux, au-delà de 3 capteurs.

5. Conclusion préliminaire

Cette évaluation fait suite à celle sur 3KAL35 où nous avions procédé à la fusion des flux de 3 Kinects sans vérifier la configuration, ou le nombre, optimal de capteurs pour nos besoins.

Les gains observés mettent en évidence une bonification de la mesure avec l'ajout d'un second capteur, dans un même ordre de grandeur qu'un 3ième. Le 4ième capteur n'a cependant qu'un impact minime. L'ajout d'un cinquième capteur n'aurait probablement plus d'incidence notable si la tendance se poursuit.

L'approche se robustifie à la manière d'un système de vote, où les mauvaises mesures ne sont pas prises en considération à moins qu'elle ne soit en nombre égal aux bonnes mesures.

Cependant, l'approche reste tributaire des meilleures performances de chaque capteur et ne permet pas des gains importants de précision quand aucun capteur ne peut fournir d'information, même si des gains moyens peuvent être observés.

Les résultats présentés dans ce chapitre ont permis la soumission d'un article en revue internationale (Masse, Lerasle, Devy, & Lefebvre, Human Motion Capture by Pose Recovery and Temporal Integration from Multi-Depth Sensor Setup (soumis), 2015).

La précision capteur semble déterminante pour les performances finales de notre approche. Ainsi, il serait intéressant de mesurer les gains avec des approches plus modernes encore, mais auxquelles nous n'avions pas encore accès. Aussi, l'utilisation non pas de la

seule segmentation, mais aussi de la profondeur, pourrait donner des résultats convainquant avec un nombre réduit de capteurs (2) mais cela reste aussi à vérifier.

C. Conclusion

Nous avons prototypé au chapitre précédent, un système de détection de posture multi-capteur avancé, basé sur le détecteur de posture OpenNI/NiTE, avec reprojection dans chaque vue de plusieurs hypothèses squelette, avec une méthode d'intégration temporelle.

Pour quantifier les performances du système ainsi formé, nous choisissons d'utiliser la distance des nœuds à la vérité terrain dans notre base acquise, et décrite précédemment.

Sur la base de gains avancés dans la littérature en matière de tracking vidéo, nous avons combiné cette approche tracking-by-detection avec une méthode d'intégration temporelle afin d'obtenir un détecteur-traqueur hybride. Nous avons rajouté comme autres hypothèses explorées, un filtrage de Kalman à réjection de mesures, à raison d'un par articulation.

La perception d'objet dans un contexte de manipulation peut fournir des informations sur le corps humain, si le système contient les informations de prise, permettant l'interaction de percepts relatifs aux objets et à l'homme durant la manipulation. En revanche la gestion de ces percepts hétérogènes car de nature différente requiert une adaptation de notre approche multi-Kinects. Ces nouvelles investigations sont décrites dans le chapitre suivant.

Chapitre V: Vers une extension à la perception conjointe homme-objet

Introduction

Cette partie traite d'un travail plus prospectif donc moins abouti sur l'utilisation du contexte dans la perception des mouvements humains.

Comme nous l'avons évoqué précédemment, la capture de mouvements humains représente de multiples enjeux. Le nombre de travaux est grand comme le montre (Moeslund, Hilton, & Krüger, 2006), mais on utilise peu souvent le contexte. Pourtant l'utilisation des multiples percepts inférés peuvent être utilisés en même temps afin de robustifier la tâche globale de perception.

Le couplage perception de l'homme-contexte est souvent exploité dans un cadre de reconnaissance d'actions humaines (Koppula, Gupta, & Saxena, 2013) (Moore, Essa, & Hayes, 1999) ou d'activités humaines (Park & Kautz, 2008), (Shapovalova, Gong, Pedersoli, Roca, & Gonzalez, 2011). L'estimation de posture, si elle est faite, est alors ici globale ou partielle... en lien avec l'activité ou action à reconnaitre.

Les travaux les plus similaires sont probablement ceux de (Kjellström, Romero, & Kragic, 2011). Mais ces travaux ne considèrent qu'un seul objet très spécifique et occulté partiellement lors de sa manipulation.

Par notre approche, nous souhaitons gérer les interactions homme-objets, i.e. saisie, pose, changement de main, etc. et les coupler avec la capture des mouvements haut du corps.

La modalité perception d'objets est considérée comme acquise, et dépassant le cadre de cette thèse. En l'occurrence, nous utilisons ARToolkit (github.com/artoolkit) afin de nous fournir les percepts relatifs aux objets, à partir des flux image RGB des capteurs.

Dans le problème que nous essayons de résoudre, présenté dans les détails ci-après, l'état du système dynamique est de dimension variable : il y a des sauts d'un espace d'état à un autre selon les événements de manipulation associés aux objets (pose, dépose, etc.). Ainsi, un objet saisi va induire un saut dans le vecteur d'état puisque il est considéré comme une partie supplémentaire dont il faut gérer l'état i.e. sa position spatiale à l'instar des autres membres corporels. Pour comparer avec le contexte du suivi multi-personnes par exemple, comme dans (Khan, Balch, & Dellaert, 2004) ou (Mekonnen, Lerasle, & Herbulot, 2013), et où le vecteur d'état est de dimension variable selon l'entrée ou sortie de cibles, ces sauts doivent aussi être gérés. Ici le principe est le même mais appliqué à des objets manipulés. Cette particularité nous invite donc à utiliser une cer-

taine stratégie, le RJ-MCMC, afin de résoudre le problème de la perception jointe homme-objets.

Dans ce chapitre, nous allons tout d'abord commencer par décrire le formalisme utilisé au travers de notre analyse du problème. Dans la seconde section, nous finissons d'énumérer les variables d'implémentations et leurs valeurs choisies pour la première expérimentation que nous décrivons dans la troisième section avant la conclusion de ce chapitre.

A. Formalisation du RJ-MCMC

1. Analyse et modélisation du problème

Dans notre contexte, les positions 3D des membres humains sont déjà filtrées par l'approche vue précédemment, donc la perception jointe n'effectue pas de gestion de cet état en dehors des articulations liées à la manipulation. L'état courant du filtre inclut donc le résultat du filtrage précédent, seulement à but de calculs, sous la forme de la concaténation des coordonnées $X_j, j \in J$ l'ensemble des articulations (joints en anglais). On note $N = card(J)$.

Pour ce qui est des objets, la liste des détections d'objets $z_t = \{z_i = (Y_i \in \mathbb{R}^3, \Omega_i \in \mathbb{R}^3), i \in I\}$ l'ensemble des objets détectés dans l'image à l'instant t, est fournie par ailleurs. Il faut savoir que cette liste peut être non exhaustive quand e.g. un objet est partiellement caché dans l'image. Ces résultats sont mis à disposition par ARToolkit dans notre cas, qui renvoie une position 6D. On note $M = card(J)$ le nombre total d'objets. Avant la première détection, l'emplacement de l'objet est non initialisé. Les détections sont des observations. Cette information, provenant d'un simple détecteur et non d'un filtrage, a donc uniquement pour but d'aider à la décision, et n'est pas le résultat de l'approche conjointe. La position des objets qui s'appuie sur la perception faite par ARToolkit doit être filtrée. Contrairement aussi aux articulations de l'homme qui ne sont pas impliquées dans la manipulation : nous supposons que la manipulation ne va affecter que les mains de l'utilisateur et l'approche conjointe doit donc les filtrer aussi.

Indépendamment de l'estimation renvoyée par la perception des objets, nous gérons donc la position des objets manipulés. La position ne change que lors des manipulations. Et suite à une prise-dépose, la fonction de filtrage doit prendre en compte la possibilité de faire la dépose à l'endroit précédent. En conséquence, la position de l'objet avant sa prise peut être un facteur. D'où le changement d'état, car la position de l'objet doit être différente et de la main, et de celle de l'objet qui ne bouge que suite à une dépose. Durant la manipulation, ce sont les mouvements de l'objet composite main-objet qui sont traqués indépendamment.

Notre approche de manipulation d'objet doit donc gérer les positions 3D des objets et de l'homme. Les objets auront cependant deux états : statiques et manipulés, encodés dans les espaces d'état plutôt que dans une variable d'état. Les objets sont supposés ne pas se mouvoir en dehors des manipulations, mais leur position fait partie de l'état cou-

rant. Durant la manipulation, leur position dépend des deux percepts : c'est donc un quatrième état différent des états précédents : ce n'est ni la position de l'objet telle que perçue, ni la position perçue de la main, ni celle avant la prise.

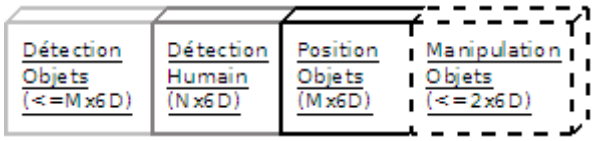


Figure 24 Schématisation de l'état interne du filtre particulaire.

Les détections objets (z_i) et humain (X_j) ne sont pas filtrées. La détection humain peut être filtrée seulement lors de manipulation, et seulement les mains. Les positions objets sont le résultat d'un filtrage : lorsqu'ils sont manipulés, les objets sont filtrés indépendamment de leur dernière position qui reste indépendante dans le vecteur d'état.

La Figure 24 illustre schématiquement la composition du vecteur d'état résultant de notre analyse.

Par abus de notation, l'ensemble des variables ainsi citées devant être estimées, changeant suivant l'état courant, est noté x_t, et l'ensemble des variables de détection à disposition (humain et objet) est noté z_t.

L'estimation de notre approche consiste donc à maximiser $p(x_t|z_t)$.

2. Présentation générale et limite du filtrage particulaire MCMC

Le filtrage particulaire MCMC (pour Markov Chain Monte Carlo) repose sur l'estimation d'une distribution par un nuage d'états, appelées particules, interagissant les unes avec les autres et pouvant représenter toute distribution.

$$p(x_t|z_t) \approx \sum_{k=1}^{K} w_t^{(k)} \delta\left(x_t - x_t^{(k)}\right), \sum_{k=1}^{K} w_t^{(k)} = 1$$

Le processus de filtrage consiste à régénérer un nuage de particules $x_t^{(k)}$ avec la probabilité (ou poids) $w_t^{(k)}$ à chaque instant image t afin d'estimer la distribution réelle. $k = 1..K$ est l'indice de la particule. Une fois ce nuage obtenu, l'état à minimum d'erreur quadratique moyenne (ou MMSE) $\tilde{x}_t = \sum_{k=1}^{K} w_t^{(k)} x_t^{(k)}$ peut être évalué et retourné.

Les particules $x_t^{(k)}$ évoluent stochastiquement dans le temps. Elles sont échantillonnées selon une fonction d'importance visant à explorer adaptativement les zones « pertinentes » de l'espace d'état. Dans l'approche MCMC que nous utilisons, le principe est d'estimer la chaîne ergodique de distribution stationnaire $p\left(x_t^{(k)} \middle| x_t^{(k-1)}, z_t\right)$.

Vu que la création du nuage de particule à l'instant t démarre d'une particule de l'instant $t-1$, il est d'usage de procéder à une étape dite de burn-in avec N_B itérations afin de laisser la chaîne atteindre la distribution stationnaire. Les états traversés ne sont pas retenues dans le nuage de particules et une acceptation de la transition remplace la particule.

La transformation $x_t^{(k-1)*}$ d'un état $x_t^{(k-1)}$ étant tirée avec une distribution $q\left(\cdot \middle| x_t^{(k-1)}, z_t\right)$, l'acceptation de la transition $x_t^{(k-1)} \rightarrow x_t^{(k-1)*}$ est soumise au ratio

$$\mathcal{A} = \frac{p(z_t|x_t^{(k-1)*})}{p(z_t|x_t^{(k-1)})} \times \frac{q\left(x_t^{(k-1)}\middle|x_t^{(k-1)*}, z_t\right)}{q\left(x_t^{(k-1)*}\middle|x_t^{(k-1)}, z_t\right)}$$

Où, dans le cas contraire, i.e. la transformation n'est pas acceptée, $x_t^{(k)} = x_t^{(k-1)}$.

Cependant, cette approche n'est pas compatible avec un vecteur d'état x_t à dimension variable, en effet la différence d'espace implique des différences d'ordre dans les probabilités qu'il convient d'égaliser pour avoir un ratio convenable et approchant de la distribution stationnaire souhaitée.

Dans la partie qui suit nous allons donc justifier le changement d'espace du vecteur d'état dans notre contexte et introduire les changements dûs au changement de paradigme MCMC pour RJ-MCMC (Reversible Jump – Markov Chain Monte Carlo).

3. Extension au filtre particulaire RJ-MCMC

Dans notre cas, la position de l'objet au repos n'est jamais mis à jour sauf dans un pouvement de dépose d'objet par l'homme, c'est ainsi que sa position est filtrée. Ainsi, durant la manipulation, la position de l'objet, devenu une articulation du corps humain, devient une partie supplémentaire du vecteur d'état, et il sera retiré lors de la dépose pour affecter la nouvelle valeur à la position de l'objet.

Dans le paradigme RJ-MCMC en général, l'espace d'état est une union d'espaces de dimensions inégales, et requiert de définir un ensemble de mouvements $Moves$ inversibles pour aller d'un espace à l'autre.

On définit, pour chaque mouvement une fonction permettant de transformer l'état courant en état tel que le mouvement le prescrit. $\forall m \in Moves$, la fonction f_m a donc comme paramètre un état de la taille escompté et forme un état de dimension différente, mais mathématiquement cette fonction n'est inversible et d'inverse f_n, $n \in Moves$ que si les dimensions d'espace de départ et d'arrivée sont en même nombre. Ainsi, des variables intermédiaires sont introduites, u_m et u_n pour donner $f_m(\cdot, u_m)$ et $f_n(\cdot, u_n)$ telles que $f_m(x_n, u_m) = (x_m^*, u_n)$ implique $f_n(x_m^*, u_n) = (x_n, u_m)$ et vice-versa.

Par abus de notation dans notre cas, on dit toujours que l'on « passe de l'état (n, x_n) à (m, x_m^*). »

Pour chaque mouvement m, un modèle de distribution $q_m(\cdot\,|m, x_n, z_t)$ pour le tirage de u_m, ainsi que la probabilité $p(z_t|m, x_m^*)$ doivent être définis.

Le taux d'acceptation d'un mouvement $m \in Moves$ peut ainsi être défini par rapport à son mouvement inverse n :

$$\mathcal{A}_m = \min\left(1, \frac{p(z_t|m, x_m^*)}{p(z_t|n, x_n)} \times \frac{q(n|m)}{q(m|n)} \times \frac{q_n(u_n|m, x_m^*, z_t)}{q_m(u_m|n, x_n, z_t)} \times \mathcal{J}_{f_m}\right)$$

Où $\mathcal{J}_{f_m}$ est la jacobienne de la transformation f_m et $q(m|n)$ la probabilité du choix du mouvement m si n est le mouvement inverse subséquent qui sont des probabilités « fixées » *a priori*, (terme ambigu car elles dépendent de l'espace de départ et d'arrivée, qui semble être intégré dans l'état mais ne fait pas l'objet direct d'une variable discrète du point de vue théorique).

4. Mouvements et probabilité de tirage *a priori*

L'ensemble des mouvements possibles, modélisés par $Moves$ dans notre implémentation est le suivant :

- Prise d'un objet dans une main
- Pose de l'objet tenu par l'une des deux mains
- Echange d'objets entre les deux mains

À cela s'ajoute un mouvement creux « Mise à jour » pour les instants image où il ne se passe rien vis-à-vis de la manipulation d'objets (pas de saut dans le vecteur d'état) mais on met à jour les composantes actuelles du vecteur d'état du système eu égard à leur dynamique courante, par exemple le simple mouvement entre deux évènements. D'où la définition de $Moves \triangleq \{PickObject, PlaceObject, SwapHands, Update\}$.

Pour l'analyse des probabilités de tirage *a priori* $q(m|n)$, prenons trois exemples.

Dans le premier, si les objets sont $Objets = \{1,2,3,\dots\}$, que aucune des mains n'est prise, et que le mouvement m est $PickObject$, il faut choisir un objet à prendre au hasard parmi $Objets$, et une main au hasard parmi $Mains = \{gauche, droite\}$ car $Mains\ Libres = Mains$. Ainsi,

$$q(m|n) = \frac{1}{card(Objets)} \times \frac{1}{card(Mains)}$$

Et n est la pose de l'objet en question, en partant de la main prise. Sachant qu'il n'y a qu'une manipulation en cours, $q(n|m) = \frac{1}{card(Mains\ Prises)} = 1$.

Dans un second temps, considérons une prise avec déjà un objet dans une main. Alors, $q(m|n) = \frac{1}{card(Objects)-1}$ car il ne reste plus qu'une main de libre, et un objet n'est plus disponible à la prise. En revanche, $q(n|m) = \frac{1}{2}$.

On peut donc déduire les formules générales suivantes :

$$q(PickObject|PlaceObject) = \frac{1}{card(Objets\ Libres)} \times \frac{1}{card(Mains\ Libres)}$$

$$q(PlaceObject|pickObject) = \frac{1}{card(Mains\ Occupées)}$$

Il ne reste qu'à définir : $q(SwapHands|swapHands) = q(Update|Update) = 1$

Il est à noter que chaque probabilité $p(m|n)$ ci-dessus doit être modulée par la probabilité de tirage fixe du mouvement avant même de tirage de la cible (main et objet). Ces probabilités sont fixées empiriquement avec pour valeurs :

$$P_{Update} = 0.50; P_{PickObject} = 0.20; P_{PlaceObject} = 0.20; P_{SwapHands} = 0.10$$

5. Transitions entre espaces par mouvement

La Figure 25 illustre les différents mouvements et leurs transformations.

Le mouvement de prise crée une articulation supplémentaire au modèle humain qui n'est pas traqué par la capture de mouvements, ou bien de manière erronée. La détection d'objet détecte avec une meilleure précision (position et orientation) l'objet manipulé, ce

qui donne plus d'information. La position de l'objet en dehors de ceci ne change jamais de valeur sauf au moment de la dépose d'un objet.

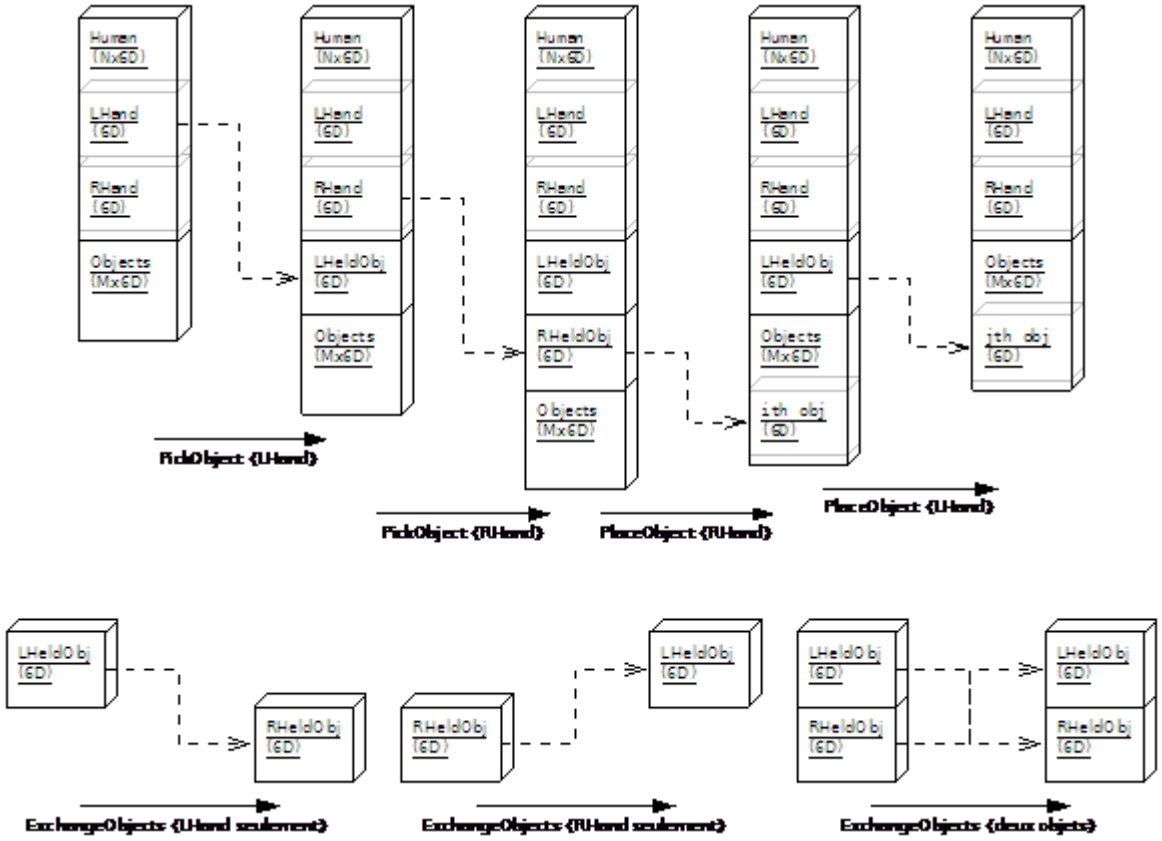


Figure 25 Illustration des mouvements d'état RJ-MCMC

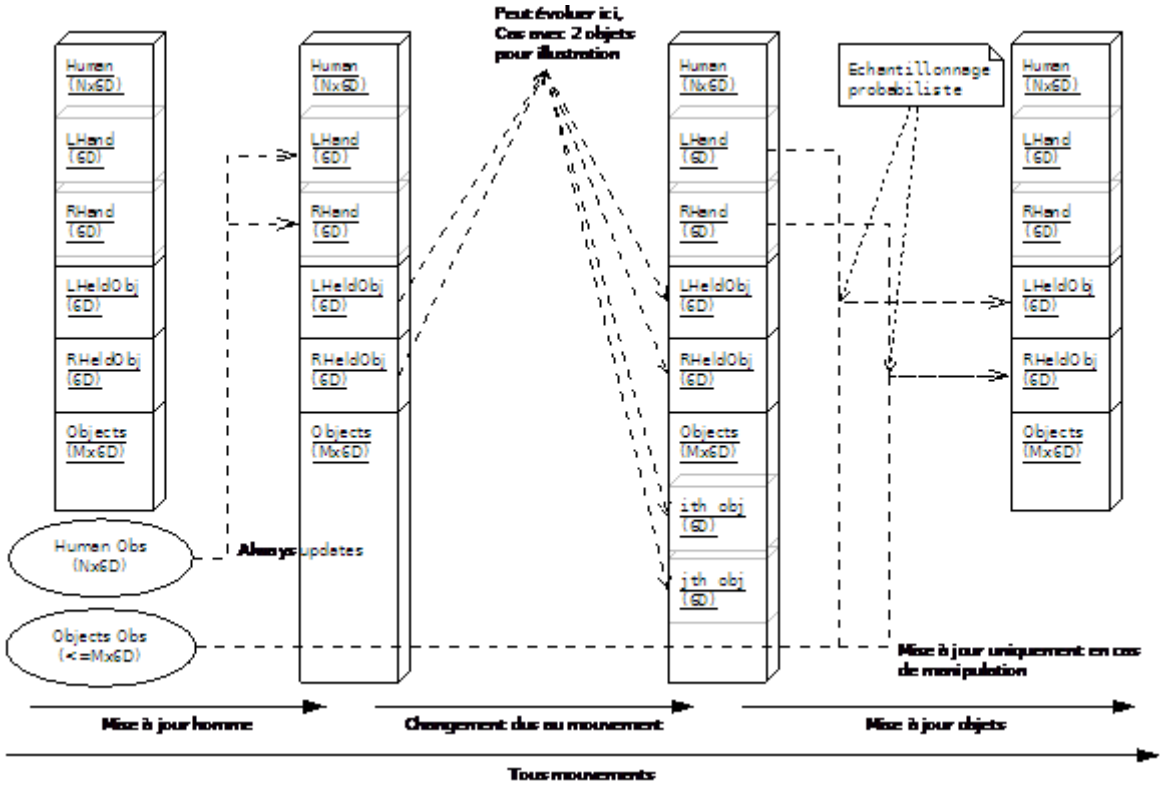


Figure 26 Illustration des transformation dans un mouvement

Le mouvement d'échange des contenus des mains échange simplement les variables du vecteur d'état.

Dans la Figure 26, le changement d'état que nous avons illustré par la Figure 25 ne se passe par conception qu'après mise à jour des articulations humaines depuis la capture de mouvements, et avant l'utilisation éventuelle de la détection d'objet pour les objets manipulés uniquement. En effet, puisque la prise/dépose d'objet peut influer sur les objets manipulés, il fait sens d'utiliser les informations de détection de ceux-ci qu'après changement lié aux transitions d'état.

Par abus de notation, $x_{t,object}$ signifie toujours la partie « active » de l'état de l'objet, c'est-à-dire dans les objets (de taille M) quand l'objet n'est pas pris, et en tant qu'articulation (conjointe à la partie de taille N) quand il est manipulé.

B. Implémentation de notre approche

Le Tableau 9 liste les formules restantes de probabilité pour le ratio d'acceptation $\mathcal{A}_m$ vu en A.3. Les formules d'échantillonage ne sont pas définies pour les mouvements n'introduisant pas de dimension, et dans un premier temps, sont laissées neutres pour l'échantillonage de Prise et dépose d'objet. La priorité est mise sur la probabilité d'observabilité de l'état x_m^*.

La probabilité lors de la prise est modélisée selon une distribution de probabilité gaussienne de la position de la main, centrée sur la position enregistrée de l'objet. La variance de cette gaussienne est modulée par la différence de position mesurée pour l'objet entre ce qui est enregistré et ce qui est détecté.

Pour la probabilité de placement, nous modélisons l'inverse de la distribution gaussienne centrée autour de la détection de l'objet de la position de la main. Nous considérons que ne pas détecter la main implique que l'objet n'a pas été lâché, ceci est une supposition du système dans ce cas où l'information est en défaut.

Probabilité	Mouvement	Formule
D'échantillonage $q_m(\cdot \mid m, x_n, z_t)$	Pick Object	1
	Place Object	1
	Swap Hands	1
	Update	1
D'observabilité $p(z_t \mid m, x_m^*)$	Pick Object	$e^{\left(-\frac{\lVert x_{t,object}-x_{t,main}\rVert^2}{{\sigma_{PickObject}}^2+\lVert x_{t,object}-z_{t,object}\rVert^2}\right)}$
	Place Object	$1-e^{-\frac{\lVert x_{t,main}-z_{t,objet}\rVert^2}{{\sigma_{PlaceObject}}^2}}$ ou 0 si $x_{t,main}=\emptyset$
	Swap Hands	$\left(1-e^{\left(-\frac{\sum_{o\in Objets}\sum_{a\in A}\lVert x_{t,o}-x_{t,a}\rVert^2}{(\sum_{o\in Objets}\sum_{a\in A}1)\cdot{\sigma_{swapHands}}^2}\right)}\right)\times e^{-\frac{\lVert x_{t,main\ gauche}-x_{t,main\ droite}\rVert}{{\sigma_{swapHands}}^2}}$ où A est l'ensemble des articulations {main, coude, épaule} visibles du côté qui tient l'objet o.
	Update	1

Tableau 9 valeurs des probabilités pour notre implémentation RJ-MCMC

Enfin, pour la permutation des objets dans les mains, nous modélisons la probabilité par l'inverse de la gaussienne des distances articulations-objet pour chaque objet, avec l'ensemble des articulations visible du bras. En effet, même si ceci n'est pas centré, nous obtenons alors une expression finale du ratio d'acceptation comme étant le rapport de ces deux produits de probabilité favorable à la permutation de bras si la distance cumulée aux articulation est plus petite ou plus grande. Enfin nous rajoutons le terme d'espacement des mains qui n'a d'effet que si un seul objet est saisi (l'échange des deux objets tenus échangerait aussi les positions des mains lors de prise)

Le Tableau 10 résume l'ensemble restant des paramètres libres utilisés précédemment.

Symbole	Signification	Valeur
N_B	Itérations de Burn-in	20
K	Nombre de particules	100
$\sigma_{PickObject}$	Ecart-type de prise d'objet	$0{,}1m$
$\sigma_{PlaceObject}$	Ecart-type de pose d'objet	$0.3m$
$\sigma_{SwapHands}$	Ecart-type d'échange de main	$0{,}1m$

Tableau 10 Valeurs des paramètres libres pour notre implémentation RJ-MCMC

C. Évaluations préliminaires et discussion

Nous avons réalisé des tests avec un petit tag de réalité augmenté et un plus grand, afin de tester la réaction du nuage de particule à la manipulation d'objets occultables et pouvant occulter le sujet.

Les scénarios suivants ont été testés :

- Aucune interaction avec les objets.
- Prise d'un objet.

- Pose d'un objet.
- Passage d'un objet d'une main à l'autre.
- Prise d'un second objet.
- Echange des objets dans les mains.
- Pose d'un des deux objets.
- Occultation du bras sujet avec un grand objet.
- Occultation d'un petit objet.
- Occultation du sujet avec un grand objet et un objet dans l'autre main/
- Occultation d'un petit objet, avec un autre objet dans l'autre main.

La Figure 27 illustre quelques mouvements typiques lors de certains des scénarios ci-dessus.

1. Critère de l'expérience

Lors de l'expérience, nous avons essayé d'utiliser une règle de décision naïve : l'état du traqueur était la particule avec l'état de manipulation (quel objet dans quelle main) le plus représenté. Pour les valeurs quantitatives (positions), nous utilisions la moyenne des particules représentant l'état le plus représenté dans la population.

Aucune interaction

Prise petit objet

Echange petit objet

Echange grand objet

Occultation par grand objet

Occultation du petit objet

Figure 27 Illustrations de l'expérience de manipulations d'objets.

Cependant, cette règle affichait des problèmes de stabilité et il était difficile d'évaluer le résultat final. La Figure 28 illustre la différence entre la prise d'un petit objet, où tout se passe bien (nous montrons 3 secondes car la stabilité n'est pas un problème dans le cas de cet objet) et la prise d'un petit objet où la stabilité fait défaut et cela se voit image par image seulement. Nous présentons donc plutôt ici des statistiques sur les particules durant les répétitions des scénarios susmentionnés.

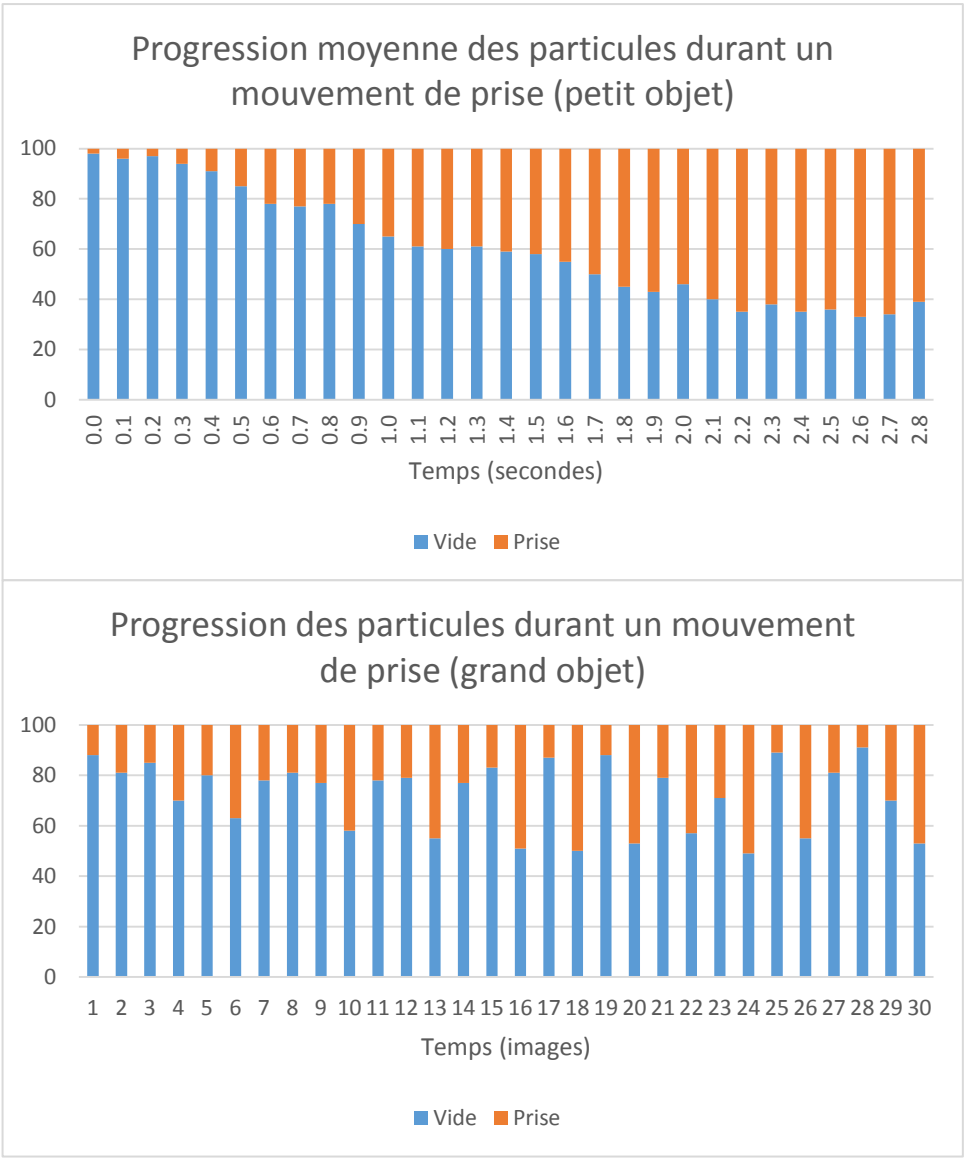


Figure 28 Statistiques des particules durant la prise d'un grand et d'un petit objet, en temps et en images.

Comme dans la première expérience, nous faisons un histogramme du nombre d'instants image où l'état attendu était représenté par une certaine quantité de particule. Cependant nous devons observer cette quantité pour une quantité d'images. Pour une plus grande lisibilité nous présentons seulement la valeur minimale et la valeur maximale par les extrémités d'un trait, ainsi que les 1er et 3ème quartiles (où au moins 25% et 75% des instants image affichaient le nombre de particules indiqué) avec une boîte. La Figure 29 illustre ce diagramme.

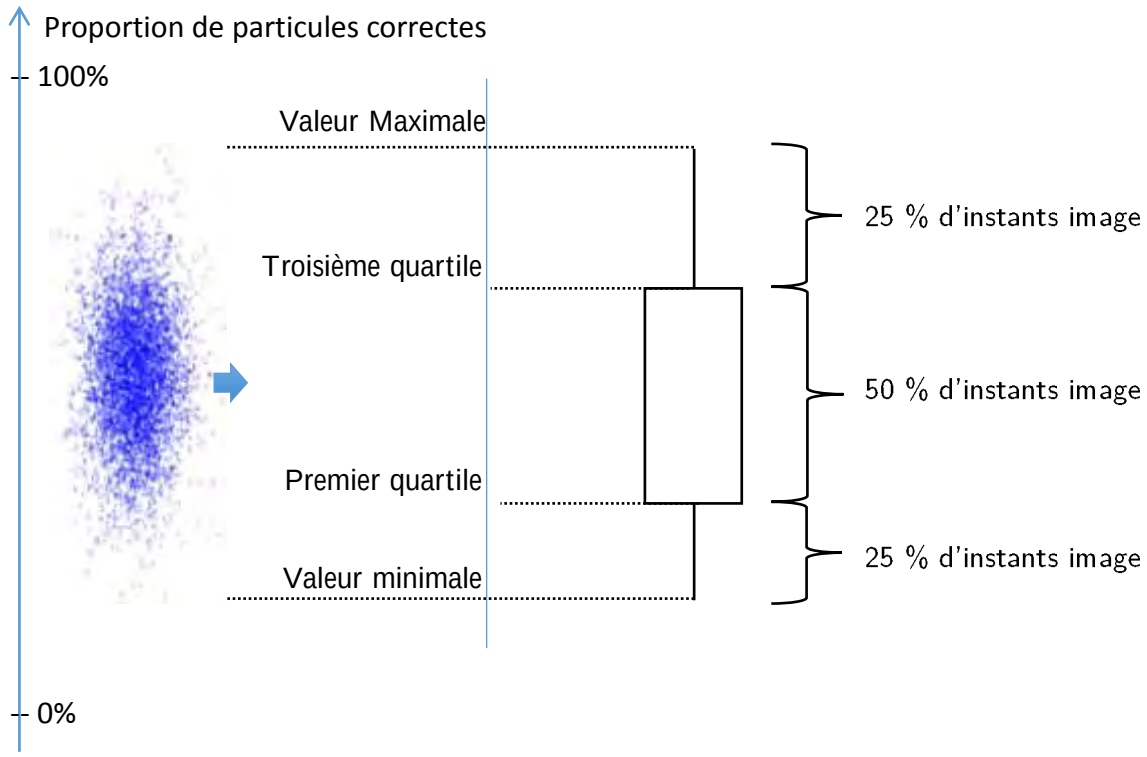


Figure 29 Anatomie du diagramme de confiance utilisé (à droite) pour une expérience, représentant un ensemble d'images, représentées par un point du nuage de point (à gauche).

2. Résultats

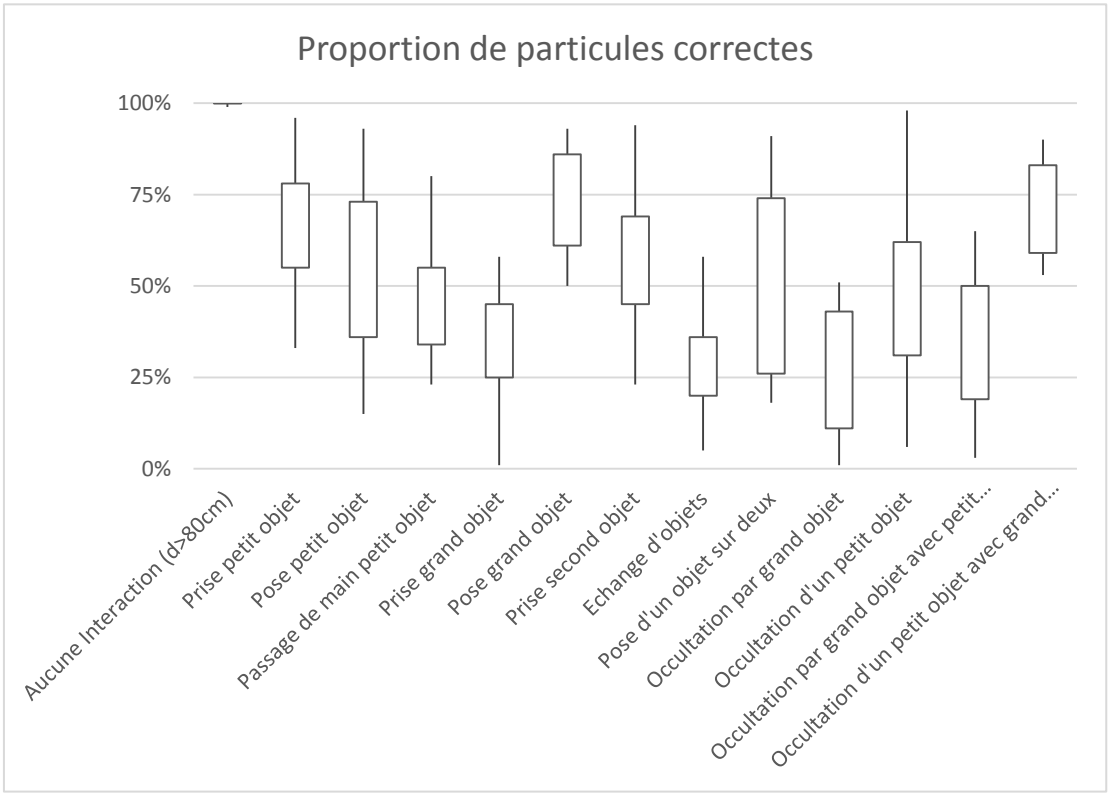


Figure 30 Diagrammes de confiance de l'expérience de manipulations d'objets.

La Figure 30 montre les résultats de notre expérience.

Nous vérifions bien que les mouvements qui n'amènent pas la main à moins de quelques dizaines de centimètres d'un objet ne provoquent aucune véritable interaction au niveau du modèle. La statistique descend à 99% plutôt qu'à 100% donc les erreurs sont négligeables.

Pour les manipulations avec un petit objet, nous voyons des résultats globalement plus satisfaisants qu'avec le grand objet : Lors de la prise d'un grand objet, on peut avoir 0% de particules correctes, même si c'est rare, et rarement plus de 50%. Avec un petit objet, les résultats sont variables mais au moins 30% des particules sont correctes. Nous notons la tendance inverse pour la pose des objets : la pose d'un grand objet est facilement détectée mais pour le petit c'est plus ambigüe avec parfois 30% seulement de particules où l'objet n'est plus manipulé.

Pour les échanges et passages d'objet, les performances sont mitigées par la confusion et l'augmentation du nombre de possibilités. D'autant plus dans le cas de l'interversion d'objets. Cependant, ceci est à compenser par le fait que plus il y a d'états différents présents, moins il y a besoin pour un état d'être représenté par 50% de la population pour être majoritaire. En règle générale, l'état est bon mais instable.

L'occultation d'un petit objet a tendance à être problématique car le principal état en compétition est celui où l'objet est dans l'autre main. La variabilité est énorme entre 6% et 98%, mais cette variabilité diminue fortement quand un autre objet occupe l'autre main. D'ailleurs, nous avons réduit la statistique aux images où la prise du gros objet était bien détectée par le système.

L'occultation par le grand objet présente des valeurs similaires à la prise mais légèrement moins bonnes : le premier quartile passe de 25% à 11% et la valeur maximale a aussi légèrement réduite, de 58% à 51%.

Notons que la répétition des mesures étant rendue difficile par la faible fiabilité du détecteur de tags réalité augmentée, nous ne pouvons que donner ces résultats qualitatifs.

3. Discussion

La comparaison des statistiques entre la prévention d'interactions et les tentatives de prises montrent que les particules détectent généralement mais pas toujours majoritairement la prise d'un objet.

La différence de comportement entre petit et grand objet montre un problème dépendant de la taille de l'objet : un objet grand est moins compatible avec notre système actuel. Il faut noter que nous utilisons que la position de l'objet pour détecter la prise/dépose, et le centre de l'objet se trouve au milieu de l'objet, ce qui peut expliquer ce comportement. Pour réduire cette différence, il faudrait donc perfectionner la modélisation afin d'introduire une transformation objet/main dans le vecteur d'état.

Un bémol global se trouve aussi être la stabilité du système. En effet, il est très fréquent que la prise soit détectée correctement, mais qu'une pose soit faite très rapidement par la suite. En fait par la modélisation de la chaîne de Markov « verrouille » la position de l'objet « en l'air » en mettant à jour sa position dans le vecteur d'état des objets par

une transition de pose, même peu probable. Il est à noter que ce comportement est encore plus mis en évidence lors de tests effectués avec un plus grand nombre de particules. L'intuition pourtant logique que seule la pose d'un objet puisse mettre à jour la position de l'objet pourrait donc ne pas être adaptée au niveau d'une particule.

Pour le passage d'objet entre les deux mains et les échanges de mains, il est normal que le système souffre de confusion. Afin de pallier à ceci une stratégie post-filtrage serait peut-être à adopter afin de lisser la mesure et mieux détecter le point d'inversion.

Pour l'occultation d'un petit objet, une différence évidente selon la présence d'un autre objet dans l'autre main révèle un autre problème trivial. En effet, sans information de la part du détecteur d'objet, l'échange d'une main à l'autre se fait quelques fois même avec les mains écartées. Ceci n'est pas corrigible dans la modélisation actuelle sans rendre impossible tout échange de mains. La nature de la distribution d'observabilité est probablement à reformuler.

L'occultation de l'homme par le grand objet souffre par ailleurs d'un problème amont, où la détection de l'homme, non conçue pour la manipulation d'objet, fournit quand même une mesure, et une mesure fortement dégradée, alors que les membres sont occultés (voir Figure 27. Le repère nommé « right_hand » se trouve au niveau de la main gauche…) Notre expérience ayant été menée avec une seule Kinect, nous devrions réitérer l'expérience avec un système complet. Cependant les statistiques des particules ne sont pas inexploitables et un modèle pourrait être ajoute, surtout avec une intégration totale humain-objet, pourrait permettre de résoudre le problème en identifiant les pixels à ne pas utiliser pour la recherche de squelette.

En revanche, si ces problèmes étaientt réglé, et avec une modélisation plus complète de la manipulation (avec une matrice de transformation 6D entre la main et l'objet), la capture de l'homme profiterait donc bien de la précision du système de détection d'objet, et l'occultation d'objet serait compensée par la capture de mouvements humains.

4. Coût CPU

Le coût CPU de la solution complète tournant sur une unique machine suit la répartition suivante (notons que la machine était équipée de plusieurs cœurs et que la consommation est indiquée en pourcentage d'utilisation d'un seul cœur, ce qui explique que le total dépasse 100%)

- 60% pour la capture d'image et détection d'objets avec deux objets
- 38% pour la détection de squelettes (avec un seul capteur)
- 20% la fusion humain-objets avec deux objets.

D. Conclusion

Notre approche combine les percepts distincts du mouvement humain et de détection d'objet. Il assure le lissage trajectoriel des positions des objets en fonction des mouve-

ments humains, et fusionne les données rendues corrélées par la manipulation par l'homme des objets de la scène.

Les résultats sont prometteurs mais préliminaires, mais nous restons confiants sur l'aboutissement possible dans cette approche, et bien que des problèmes restent encore à régler, des solutions existent.

Les simplifications faites durant cette première modélisation pourraient être étendues et nous pourrions modéliser le problème comme une chaîne de Markov plutôt qu'un filtrage particulaire, mais cela n'accomplirait plus de lissage trajectoriel sur l'état des objets qu'il faudrait alors faire par ailleurs.

La prochaine étape de l'évaluation consistera à faire une évaluation quantitative à l'aide d'un système MoCap afin de mesurer les gains obtenus en précision et en robustesse aux occultations.

Chapitre VI: Ingénierie

Introduction

Ce chapitre résume des développements logiciels que nous avons effectués pour les besoins de la société Magellium, partenaire de cette thèse CIFRE. En effet, côté Recherche, nous avons donc contribué comme vu dans les chapitres précédents, sur la capture de mouvements humains à partir de plusieurs capteurs RGBD, puis sur la perception conjointe Homme-Objet. Plusieurs collègues de Magellium ont suivi mes travaux : leur principale motivation était qu'à la fin de cette convention CIFRE, Magellium dispose de résultats exploitables de ceux-ci. Donc côté Technologie, l'accent a été mis sur la production, et la « mise aux normes » pour la société, de logiciels, algorithmes et codes utilisables.

Cette thèse s'inscrit dans le projet PRACE, auquel j'ai participé ponctuellement, et que nous décrivons ici. Puis, se trouvent la description et la discussion sur les productions logicielles faites durant la thèse.

A. Projet PRACE

PRACE signifie « Productive Robot ApprentiCE. » Ce projet, évoqué en introduction, a été financé par le septième programme-cadre de l'Union Européenne pour la recherche et le développement technologique, FP7.

L'objectif de PRACE est le développement d'une plateforme robotique mobile pour l'automatisation d'opérations d'assemblage de routine, dotée de deux bras manipulateurs et hautement adaptable. La principale caractéristique visée est le réglage rapide et intuitif du système par apprentissage.

Les conditions d'automatisation de l'assemblage ont changé de manière radicale ces dernières années. Les clients ont des demandes de plus en plus spécifiques, poussant la complexité des productions. Comme les systèmes actuellement disponibles à l'exploitation ne peuvent satisfaire cette extrême flexibilité des applications, modifiées à un rythme rapide, en fonction des besoins du marché, le but d'un projet tel que PRACE est d'aboutir à un nouveau concept de système robotique. Dans le cadre des programmes actuellement mis en œuvre sur l'Usine du Futur (industrie 4.0 en Allemagne), il existe plusieurs contraintes importantes pour l'élaboration d'un tel système robotique:

- la flexibilité: le robot doit pouvoir assembler sur la même chaîne, des variantes d'un objet donné.

- la reconfiguration: les robots sont des moyens de production génériques. Ils doivent pouvoir être amortis sur de longues périodes, sur différentes chaînes d'assemblage.
- la programmation intuitive pour accélérer les temps de mises en oeuvre.

Dans ce contexte, il est indispensable d'améliorer le temps d'apprentissage des tâches par un robot. Traditionnellement ceci était fait sur des robots industriels, en exploitant une interface manuelle (joystick) pour guider l'organe terminal du robot sur les différentes positions à atteindre. Cela devait se faire à distance pour des raisons évidentes de sécurité. Avec les robots collaboratifs (KUKA LWR et IIWA, ABB FRIDA, YASKAWA Motoman...) qui répondent aux normes en vigueur pour être exploités à proximité, ou même au contact des opérateurs, cette technique est plus facile; l'opérateur peut même emmener à la main l'organe terminal dans les positions requises, enregistrer ces positions, et reexécuter les trajectoires ainsi apprises.

Ces techniques existent. L'apprentissage par démonstration est une stratégie plus ambitieuse qui s'apparente à ce que fait un ouvrier senior avec un apprenti : la main d'oeuvre ouvrière peut être formée rapidement (en quelques répétitions de la tâche seulement) et intuitivement à des compétences manufacturières nouvelles, en contraste total avec ce que la robotique industrielle actuelle exige comme compétences avancées et une exploitation longue afin d'amortir les coûts d'élaboration des solutions d'automatisation de tâches industrielles à l'aide de robots. C'est ce verrou que le projet PRACE essaie de lever avec sa solution.

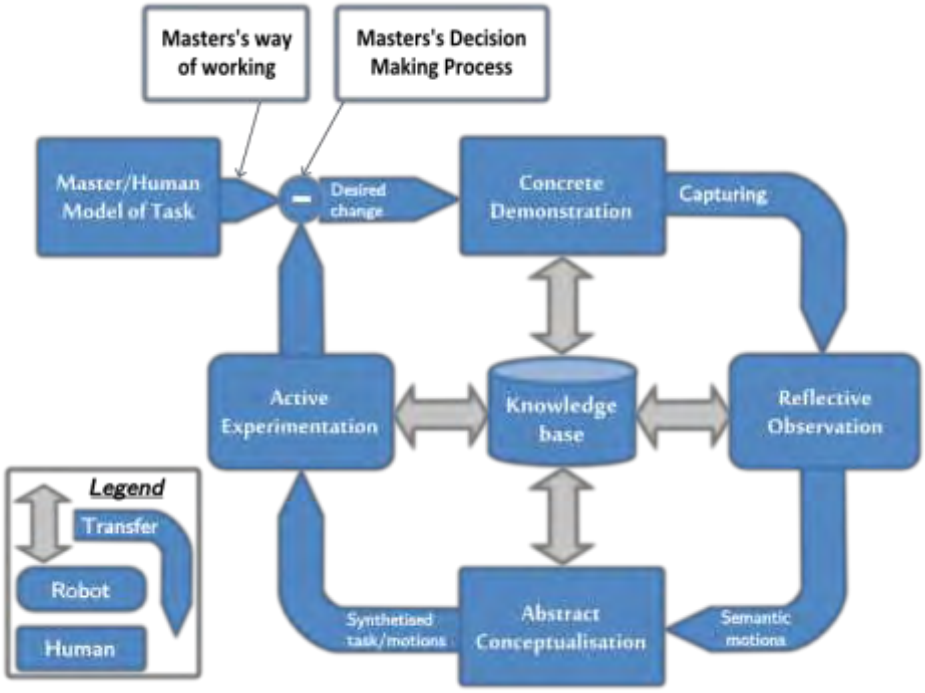


Figure 31 Le concept PRACE d'un point de vue logiciel. (http://prace-fp7.eu/)

Le concept PRACE repose fondamentalement sur l'apprentissage par démonstration. La relation de l'utilisateur avec le système est comparable à celle entre un maître et son apprenti. Dans notre cas, le maître enseigne à l'apprenti en démontrant des compétences. L'apprenti regarde les actions et les effets pour catégoriser ce nouveau savoir dans sa base de connaissances. La trajectoire est analysée pour en déduire d'une part les actions abstraites (suggérées par l'opérateur humain) puis les éléments géométriques associés d'autre part. Ensuite, en appliquant cette nouvelle compétence, en synthétisant une tâche et une trajectoire pour ses propres actionneurs avant de l'exécuter dans un environnement virtuel ou avec la plateforme, le maître corrige l'exécution en affinant l'expérience. Ces itérations sont répétées jusqu'à ce que le maître soit satisfait du résultat. La Figure 31 illustre ce processus.

Un autre aspect important du système robotique PRACE est le fonctionnement sans garde-fou pour atteindre les performances visées d'installation rapide. Le fonctionnement sans garde-fous limite cependant la vitesse maximale du robot. Une approche à base de double bras manipulateur est envisagée afin de rester compétitif avec un ouvrier humain.

Ce type de système combinant la manipulation à deux bras et une plateforme mobile pour pourvoir à une mobilité locale à l'intérieur du lieu de travail permet donc d'automatiser de manière économique de nouvelles tâches applicatives. En utilisant une approche modulaire, le système PRACE peut même être recombiné pour n'utiliser qu'une partie du robot pour des applications spécifiques, par exemple, utiliser seulement un bras ou le système sans mobilité.

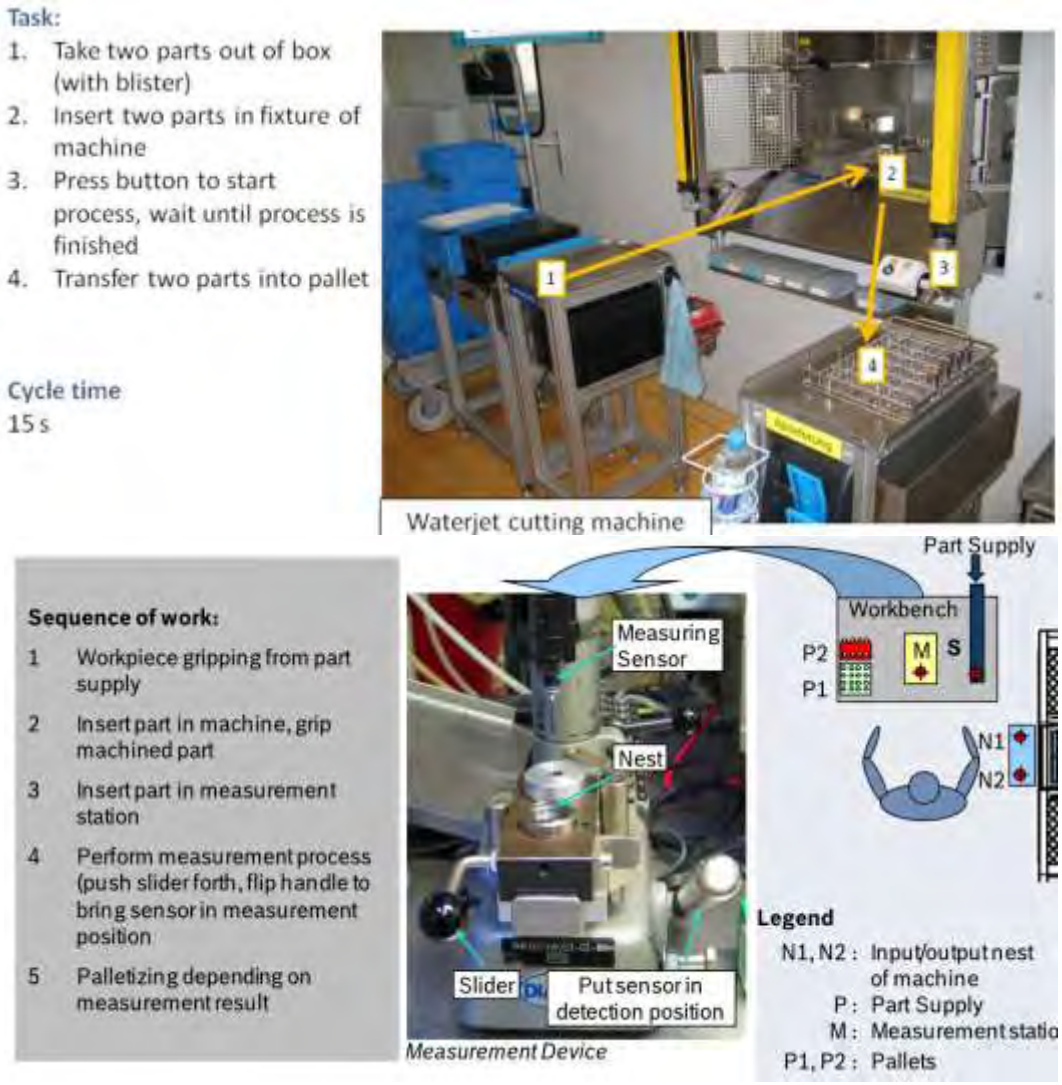


Figure 32 Scenarios d'utilisation du système PRACE. (http://prace-fp7.eu/)

Différents cas d'utilisation d'assemblage étaient définis en tant que tâche-type pour le concept PRACE. Deux des scénarios considérés sont montrés sur la Figure 32:

- d'une part le chargement d'une machine-outil, ici une machine de découpe à jet d'eau: l'opérateur démontre au système robotique, où prendre et comment positionner les deux pièces à introduire dans la machine, puis comment sortir et mettre en palettes les objets après découpe
- d'autre part le chargement d'un site d'inspection. C'est la même tâche de chargement/déchargement, sauf que le positionnement de l'objet à inspecter sous le capteur, doit être précis (dans le "nest"), et que le choix de la palette en sortie dépend du résultat de la tâche d'inspection.

Afin d'accomplir cette tâche, le projet est classiquement structuré en groupes de travail, et chaque partenaire devait participer en proportions variées en fonction de leurs compétences. Deux exemples de work-packages spécifiques : « Skill-Supporting End-Effectors » et « Tracking and Recording Movements and Actions in Manual Assembly

Processes. » Celles de Magellium, mon employeur, dans la seconde tache englobaient l'expertise capteur, la fusion de données, et l'architecture de communication.

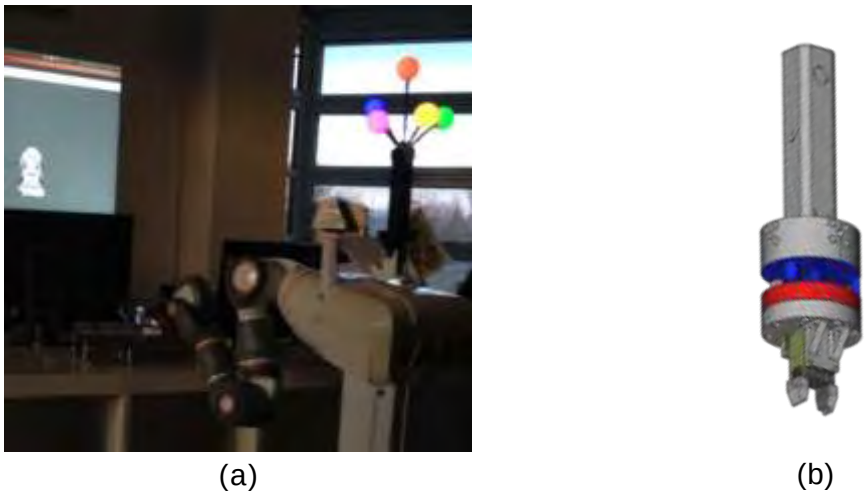
Dans l'optique de développer ses compétences dans le domaine des capteurs, financer ce travail de thèse portant sur l'acquisition de données squelette et la fusion avec des données d'objet concordait parfaitement avec les attributions de Magellium au sein du consortium.

Les autres membres du consortium, je cite, Robert Bosch Gmbh, Fraunhoffer IPA, le Teknologisk Institut Danois (DTI), l'université de Lund, et Axelius Automation. Ce dernier a été remplacé par ABB Automation au cours du projet.

Mon implication dans le projet a été celle d'un collaborateur à part entière. En ma qualité d'employé de chez Magellium, membre du consortium, j'assistais aux réunions, meetings et participais au suivi de l'activité.

Le cahier des charges présenté en introduction a été inspiré du contexte PRACE : la démonstration d'une tâche de manipulation à un robot, mais nous avons aussi voulu garder les possibilités d'élargir ce contexte.

Pour les besoins du projet, deux approches ont été prototypées : celle présentée dans ce manuscrit, et un « teach-in handle, » ou TIH, un manche démontable et instrumenté permettant de manipuler et traquer l'outil du robot manuellement, sans manipuler tout le bras du robot. La Figure 33 montre une photo et le modèle 3D du TIH. L'instrumentation consistait en un ensemble de marqueurs, en l'occurrence des sphères colorées assez faciles à détecter par segmentation chromatique. Connaissant le modèle 3D de cette interface, donc la position des sphères vis-à-vis d'un repère défini, un algorithme de localisation PnP (à base des appariements centre des sphères dans le modèles/ centre des ellipses détectées dans l'image), est exploité afin de localiser l'interface à tout moment dans la scène. La tâche subséquente du robot étant de reconstruire sa propre trajectoire afin de reproduire les mouvements de l'outil au bout du TIH.



(a) (b)

Figure 33 Teach-In Handle.

a : le TIH avec ses marqueurs colorés et illumines, sur un emplacement dédié sur le robot.
b : modèle CAD du TIH avec système de tolérance aux efforts et outil du robot.

Une telle approche a été vite validée et notre approche présentant des caractéristiques encore inconnues, n'a pas été retenue pour le projet, bien qu'il fut avoué qu'elle était plus dans l'esprit de la démarche PRACE. Mais il fallait comprendre que le projet durant 3 ans comme une thèse, prévoyait cependant un timing de fin de conception afin d'intégrer et de raffiner durant le dernier tiers du projet, incompatible avec le timing d'une thèse au final.

Notons que la démonstration de tâches d'assemblage avec un tel objet n'est pas triviale : la détection, le suivi et la localisation de ce « teach-in handle » permet certes de reconstruire la trajectoire de l'organe terminal d'un manipulateur, mais pour transformer cette trajectoire en une trajectoire de la chaîne cinématique complète, peut se révéler impossible, si le bras passe dans des singularités.

L'approche teach-in handle, concurrente de prime abord, a aussi inspiré notre recherche d'intégration de percepts objets. Cependant elle n'a pas été exploitée car nous voulions explorer la manipulation de plusieurs objets et l'approche TIH était limitée en termes d'exemplaires visibles, car les marqueurs sur l'objet étaient de couleur afin d'automatiser leur identification.

À la fin du projet, en Octobre 2014, une phase d'évaluation en environnement de production réel était prévue afin de tester les fonctionnalités du système et pour assurer la capacité à entraîner le système en moins d'une demi-journée par des utilisateurs non-experts. Lors de la revue finale du projet, l'Université de Lund a renouvelé son regret de ne pouvoir intégrer la perception de l'homme présentée avant dans la phase d'apprentissage.

Plusieurs librairies ont dues être développées, et pour une meilleure transmission du développement du contenu scientifique de ma thèse, il semblait naturel de structurer le plus possible pour faciliter le travail de mes collaborateurs, collègues, ou successeurs futurs. Ces librairies se trouvent détaillées après un point préliminaire sur les librairies utilisées.

B. Environnements de développement produits (librairies, MagBot, ROS, etc.)

Dans cette partie, les noms de fonction et les extraits de code seront écris en police à chasse fixe afin de mieux les différencier.

Il est à noter que d'autres librairies utilisées uniquement pour les expériences hors-ligne ont été développées mais ne sont pas mentionnées ici, mais ont été partagées avec les collègues doctorants du LAAS (L. Marti) ayant utilisé la base de données acquise et seront peut-être mis à la disposition avec les acquisitions.

Présentons tout d'abord les librairies utilisées avant de présenter les librairies créées.

1. Librairies utilisées

a) ROS

ROS, pour Robot Operating System, est pour simplifier une librairie de communication inter-processus et compatible multi-langages de programmation. Python et C++ sont les deux langages supportés officiellement, et d'autres implémentations de ROS sont disponibles dans la communauté par exemple en Java. Ce qui en a certainement fait un framework très populaire, car toutes les communautés peuvent très facilement interfacer leurs programmes avec d'autres en utilisant ROS, générant ainsi une grande variété d'outils accessibles.

ROS est donc un framework, divisant les programmes et les données en plusieurs concepts listés et expliqués brièvement ci-après :

Computation graph : c'est le réseau pair-à-pair des processus ROS qui traitent ensemble les données du système.

Nœuds : Les nœuds sont des processus qui exécutent des calculs. ROS est conçu pour être modulaire à une échelle très fine. Un système de contrôle robotique contient en général plusieurs nœuds. Par exemple, un nœud contrôle un télémètre laser, un nœud contrôle les moteurs, un nœud s'occupe de la localisation, un nœud effectue la planification de mouvements, un nœud fournie une vue graphique du système, etc. Un nœud ROS est écrit à l'aide de librairie client ROS, telle que roscpp or rospy.

Master : le ROS Master fournit un registre et une recherche de noms au reste du computation graph. Sans le Master, les nœuds seraient incapables de se trouver, d'échanger des messages, ou invoquer un service.

Message : Les nœuds communiquent entre eux en se passant des messages. C'est simplement une structure de données, qui contient des champs nommés et typés. Plusieurs types de base (entier, caractère, valeur à virgule flottante…) sont compatibles, ainsi que les tableaux à taille fixe et variables. L'imbrication de messages est possible.

Topic : Les messages sont transmis selon une méthode de publication/souscription. Un nœud envoie un message en le publiant sur un certain topic. Un nœud qui est utilisateur des données va y souscrire. Il peut y avoir plusieurs émetteurs et plusieurs récepteurs à un unique topic. En général on peut considérer un topic comme un bus de communication fortement typé : chaque bus est unique, et tout nœud peut y lire et écrire du moment que le message est du type adéquat.

Service : Un service est plus adapté pour les interactions demande/réponse. C'est une paire de messages, et c'est aussi publié par un nœud qui devient alors serveur, et répond par un message réponse à tout message demande qu'il reçoit.

Serveur de paramètres : Le serveur de paramètres est une base de données centrale permettant de stocker des données et d'y accéder par leur nom. Il est plus adapté aux données susceptibles de ne pas changer régulièrement.

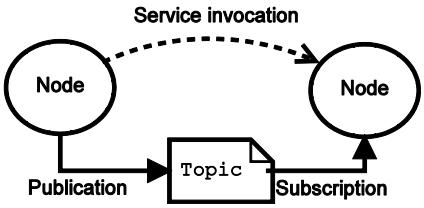


Figure 34 Schéma illustrant la communication entre deux nœuds ROS.

La figure Figure 34 montre la représentation de deux nœuds interagissant par service et message sur un topic.

b) OpenCV/HighGui

OpenCV (pour Open Computer Vision) est une bibliothèque graphique libre, initialement développée par Intel, spécialisée dans le traitement d'images en temps réel. La société de robotique Willow Garage assure le support de cette bibliothèque depuis 2008.

La librairie contient des algorithmes de traitement d'image, vidéo, de calcul matriciel et de machine learning. Elle contient aussi des fonctions utilitaires d'entrée-sortie fichier et d'affichage graphique.

OpenCV a surtout été utilisée durant ma thèse pour enregistrer et lire sur disque dur les flux des Xtions pour l'acquisition des bases de données mentionnées Chapitre II:D.

Plus notablement, OpenCV est la seule librairie requise par l'utilitaire de synchronisation temporelle illustré Figure 17.

2. Librairie créées

Les librairies ont pour but d'aboutir à un système temps-réel basé sur le framework ROS décrit précédemment. Nous allons commencer par les plus petits aspects : l'interfaçage avec les capteurs grâce à OpenNiTools, le filtrage de Kalman, le filtrage de Viterbi, pour ensuite terminer sur l'architecture logicielle globale et comment chaque composant s'insère à sa place.

a) OpenNiTools (peut-être mettre dans librairies utilisées ?)

Le périphérique RGB-D utilisé était utilisable sous Linux et Windows par la librairie OpenNI. Cette librairie était développée comme un framework de générateurs de données, autour du concept de plug-in. L'interface de développement fournit un ensemble de classes génératrices définies, mais le code exécuté est implémenté par des librairies tierces, et enregistré pour être mis à la disposition des programmes utilisant la librairie au moment de l'exécution. Le framework pouvait créer tous les générateurs dont les prérequis (listables à l'intérieur du programme par les fonctions de manipulation des « neededNodes ») étaient satisfaits. Un exemple des prérequis, dérivé de l'accès que le framework donnait aux nœuds requis, est montré Figure 35.

Dans notre cas, les plugins nécessaires étaient le driver pour communiquer avec le périphérique, afin d'avoir les générateur d'images de couleur et profondeur, et le middleware NiTE pour la génération de squelettes.

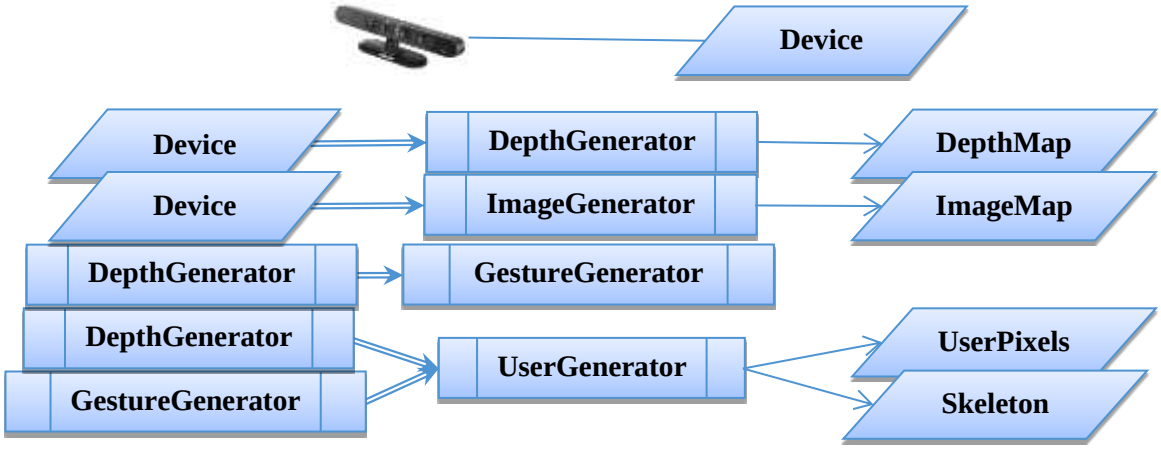


Figure 35 Règles de création des générateurs OpenNI.

Le framework était à la base dynamique. Un périphérique branché signifiait un générateur d'image et un générateur de profondeur de disponible, un générateur de gestes, et un d'utilisateur (fournissant entre autres le squelette) se reposant aussi sur le générateur de gestes, pouvaient donc être instanciés à partir du générateur de profondeur.

Figure 36 Graphe de dépendances des générateurs pour un périphérique.

Le problème du framework en revanche est qu'il n'était pas réellement prévu pour plusieurs périphériques. Tout se déroulait normalement comme le montre la Figure 37. Cette figure ne montre d'ailleurs pas d'autres générateurs invalides.

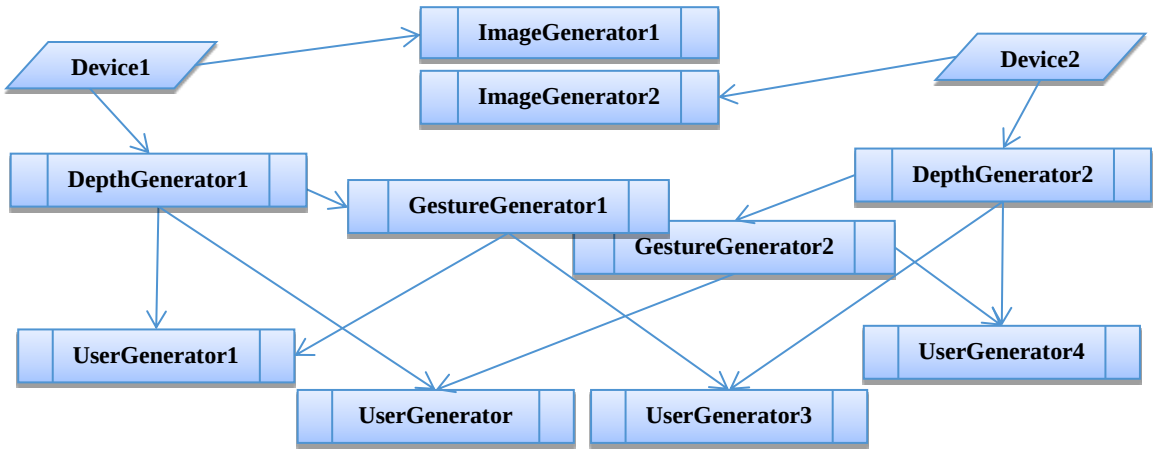


Figure 37 Graphe non-exhaustif de générateurs pour deux périphériques.

Par ailleurs, les UserGenerator pouvaient être instanciées avec une version spécifique. Ce qui pouvait passer pour une fonctionnalité superflue s'est révélée cruciale quand

s'est révélé le singleton global du DepthGenerator source d'un UserGenerator. En effet, malgré le diagramme des « neededNodes » indiquées dans la Figure 37, UserGenerator1 et UserGenerator4 retournaient des images d'utilisateurs et des squelettes identiques, provenant du DepthGenerator1.

Dans le but de remédier à ces problèmes, mais aussi d'assurer une bonne transmission des connaissances sur le Framework OpenNI, une librairie simplifiée a été mise en place. Nommée « OpenNITools », cette librairie partagée avec le LAAS permet une meilleure gestion des générateurs en présence de multiples périphériques. Elle permet de (dans l'ordre nominal) :

- Lister les numéros de série (ou à défaut un identifiant unique) des périphériques branchés.
- Initialiser les générateurs d'un périphérique.
- Obtenir les générateurs correspondant à un périphérique spécifique.

Le fonctionnement en présence de plusieurs périphériques est sensible à l'ordre, ce qui est non documenté. Par exemple l'instanciation du DepthGenerator avant l'ImageGenerator est nécessaire pour obtenir les images à la résolution pleine de 640x480, et spécifier cette résolution de force provoque un plantage. Une partie du code standard d'utilisation d'OpenNI, inspiré des exemples fournis avec, a été réuni dans la fonction d'initialisation avec la connaissance de ces sensibilités. De plus, si des modifications devaient être apportées sur le fonctionnement, telle que la registration (transformation de l'image provenant du capteur de profondeur dans le point de vue du capteur couleur) ou la synchronisation, ces choix pouvaient être spécifiés dans le profil passé en paramètre de la fonction d'initialisation.

Au total plus de 1000 lignes de codes d'implémentation ont été créées pour un fichier en-tête de 200. La documentation du code est soignée comme pour toute librairie destinée à être partagée.

b) KalmanFiltering

Afin de poser les bases d'un estimateur basé Viterbi, un estimateur basé sur un filtre de Kalman a été recodé.

Les objectifs de l'implémentation était une utilisation simple de l'extérieure, une lecture évidente permettant de retrouver des étapes du calcul, et cependant une optimalité de ce dernier en terme de coût CPU. De plus, le code devait fonctionner sur plusieurs librairies de calcul C++ utilisées, car les librairies Magellium et LAAS différaient.

Un estimateur possède deux fonctionnalités : prédiction (predict) et correction (correct). Ces deux fonctions sont clairement identifiées et exposées. La réjection de mesure a été implémentée dans une fonction séparée (smartMultiCorrect) pour une correction à partir de multiples observations.

Une utilisation lourde de la fonctionnalité de template a permis d'accorder les autres objectifs. Pour rappel, un template est du proto-code variable en fonction de paramètres connus à la compilation, mais en dehors du code templaté. Il permet d'instancier des

copies différentes en fonction des valeurs des variables utilisées. Il devient évident que cela a permis l'adaptation à plusieurs librairies scientifiques. De manière générale, les algorithmes sont aussi génériques en taille des vecteurs d'états ou des matrices. Mais en calcul informatique, l'implémentation avec taille fixe permet de se passer de tests sur la taille des variables, et donc de gagner du temps à l'exécution. Ou en tout cas laisser le choix au compilateur qui est généralement étudié pour optimiser au besoin, mais qui ne peut en aucun cas le faire s'il n'a pas l'information.

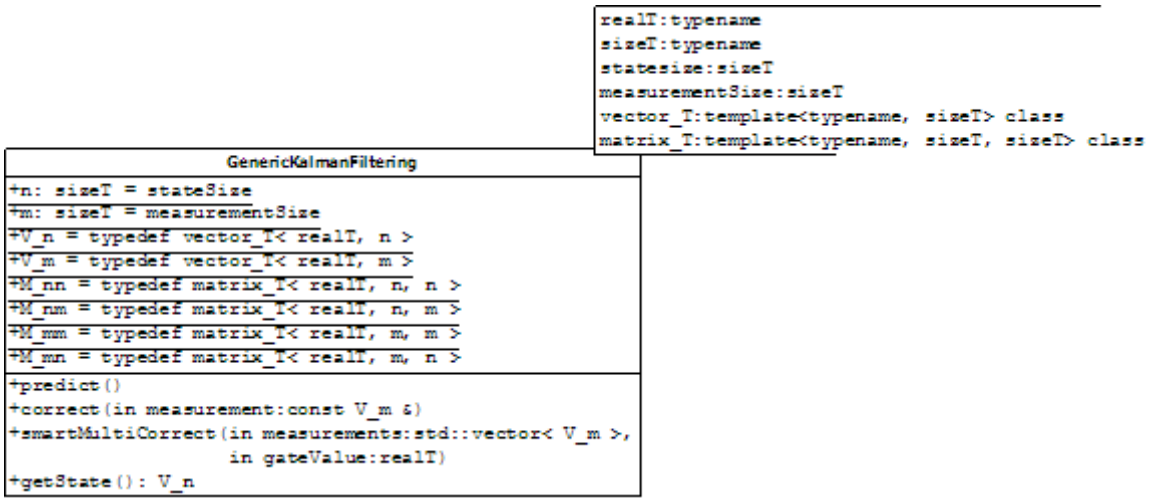


Figure 38 Diagramme UML de la classe-modèle GenericKalmanFilter.

Le filtre peut fonctionner sans mesure fournie, ou avec plusieurs mesures. Chaque pas de temps devant absolument faire l'objet d'exactement un traitement, quel que soit le nombre de mesures.

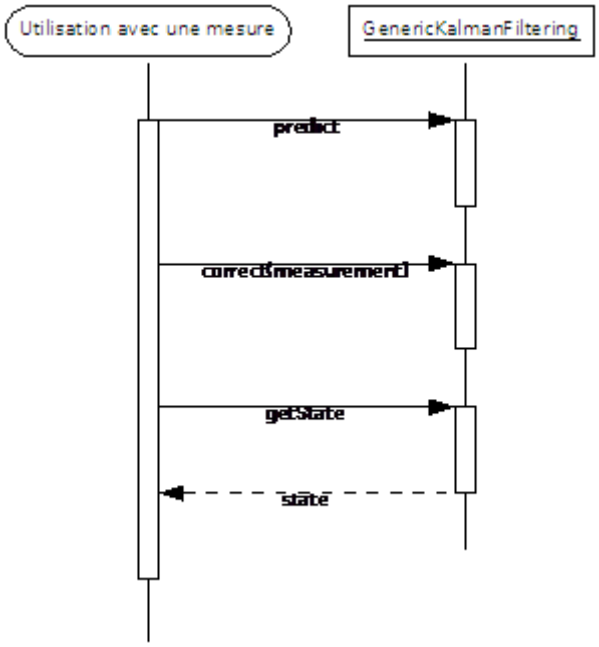


Figure 39 Diagrammes d'utilisation de la classe-modèle GenericKalmanFilter.

La Figure 39 montre les deux cas d'utilisation de la classe GenericKalmanFilter. La phase de prédiction pouvant être exécutée seule.

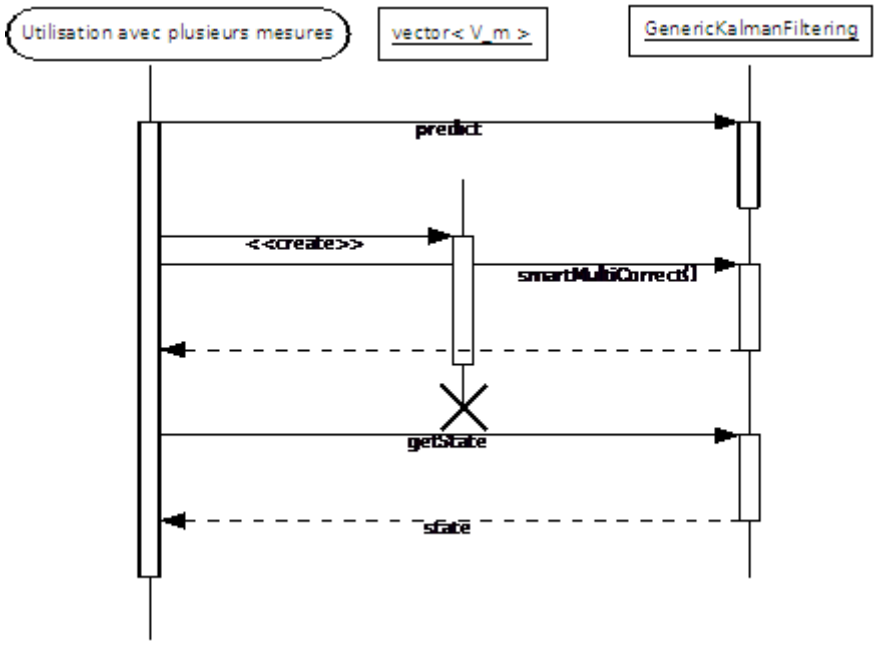


Figure 39 (suite) Diagrammes d'utilisation de la classe-modèle GenericKalmanFilter.

La Figure 40 contient le code de la classe Generic Kalman Filtering. On peut voir en premier lieu l'utilisation de noms parfois longs pour expliciter leur usage, et le surnommage cours pour avoir du code lisible.

```
/*!
 * Based on http://www.cs.unc.edu/~welch/kalman/kalmanIntro.html:
 * http://www.cs.unc.edu/~welch/media/pdf/kalman_intro.pdf
 */
template<typename realT, typename sizeT, sizeT stateSize, sizeT measurementSize,
template<typename, sizeT> class vector T,
template<typename, sizeT, sizeT> class matrix T,
template<typename mat T, typename mat TTranspose> class transpose functor,
template<typename mat_T> class invert_functor>
class GenericKalmanFiltering
{
    static const sizeT n = stateSize;
    static const sizeT m = measurementSize;
public:
    typedef matrix_T<realT, n, n> M_nn;
    typedef matrix_T<realT, n, m> M_nm;
    typedef matrix T<realT, m, m> M mm;
    typedef matrix T<realT, m, n> M mn;

    typedef vector_T<realT, m> V_m;
    typedef vector_T<realT, n> V_n;

private:
    /*! procedure returning success of
     *          inv<n>
     *  (M_nn) -----------> M_nn */
    template<sizeT _n>
    inline bool invert(const matrix_T<realT, _n, _n>& M_in,
                       matrix T<realT,  n,  n>& M out)
    {
        bool success = invert_functor<matrix_T<realT, _n, _n> >()(M_in, M_out);
        return success;
    }

    /*!          trans<n>
     *  (M nm) -----------> M mn */
    template<sizeT _n, sizeT _m>
    inline matrix_T<realT, _m, _n> transpose(const matrix_T<realT, _n, _m>& M)
    {
        return transpose functor<matrix T<realT,  n,  m>, matrix T<realT,  m,  n> >
               ()(M);
    }

    /*! Current State */
    V n  xk;
    /*! Current error covariance */
    M nn  Pk;

    /*! _H: Matrix to compute a measurement from a state by
     * zk =  H *  xk
     */
    M mn  H;
    /*! _H^T */
    M_nm _HT;
    /*! _A: Matrix to predict next state from currentState by
     *  xk =  A *  xk
     */
    M nn  A;
    /*! _A^T  */
    M_nn _AT;

    /*!  Q: Covariance of the normal noise on state dynamics
     *  Pk =  A *  Pk *  A^-1 +  Q
     */
    M_nn _Q /*stateNoiseCovariance*/;
    /*! _R: Covariance of the normal noise on measurement */
    M mm  R;
```

Figure 40 Extraits du code du filtre de Kalman.

Le nommage des membres commençant par le caractère underscore '_' est une spécification du code Magellium.

```
/*!
     * corrects the filtering with no measurement (only predicts)
     * Use this if you have no measurement
     */
    inline void predict()
    {
         xk =  A *  xk;
         Pk =  A *  Pk *  AT +  Q;
    }

    inline bool smartMultiCorrect(
                      std::vector<vector T<realT, measurementSize> > measurements,
                      realT gateValue){
        bool madeACorrection;
        while(measurements.size()){
            std::vector<realT> sigmaDistances(measurements.size());
            M mm totalMeasurementNoise =  H *  Pk *  HT +  R;
            M mm& totalMeasurementNoise m1 = totalMeasurementNoise;

            if( !invert(totalMeasurementNoise, totalMeasurementNoise_m1) )
                throw std::string("noise singularity exception !");

            V m bestInnovation;
            realT minimumSigmaDistance = std::numeric limits<realT>::infinity();
            int bestMeasurement_sIndex;

            for(int meas_i = measurements.size() - 1 ; meas_i >= 0 ; --meas_i){
                V_m innovation = measurements[meas_i] - _H * _xk;
                V m backProject = totalMeasurementNoise m1 * innovation;
                realT squareProject = 0.0;

                for(unsigned int i = 0 ; i < m ; ++i)
                    squareProject += backProject[i] * innovation[i];

                if( (sigmaDistances[meas i] = squareProject) < minimumSigmaDistance ){
                    minimumSigmaDistance = sigmaDistances[meas i];
                    bestInnovation = innovation;
                    bestMeasurement_sIndex = meas_i;
                }
            }

            if(minimumSigmaDistance > gateValue)
                break;

            measurements.erase(measurements.begin() + bestMeasurement sIndex);

            M nm K =  Pk *  HT * totalMeasurementNoise m1;

            _xk = _xk + K * bestInnovation;
             Pk =  Pk - K *  H *  Pk;
            madeACorrection = true;
        }
        return madeACorrection;
    }
```

Figure 40 (suite) Extraits du code du filtre de Kalman.

On peut remarquer que la notation des variables et l'utilisation de la surcharge des opérateurs ('*', '+' …) pour les opérations matricielles permettent de lire aisément le code comme on lit l'algorithme (crédité en commentaire du code, au début de la classe, pour référence).

Par ailleurs, l'accent a aussi été mis sur l'optimalité du code sans sacrifier la lecture : ainsi, une variable créée est réutilisée sous un autre nom (totalMeasurementNoise en totalMeasurementNoise_m1) pour sauvegarder l'espace.

Il est aussi possible d'utiliser ponctuellement la correction avec des mesures d'un espace différent du moment qu'on passe à correct les matrices H, H^T et R idoines.

```
/*!
     * corrects the filtering with the casual measurement. Uses the embedded H and R
matrices
     */
    inline void correct(const vector_T<realT, measurementSize>& measurement)
    {
        correct(measurement,  H,  HT,  R);
    }

    /*!
     * @brief corrects the filtering with the casual measurement.
     * @param measurement
     * @param  H
     * @param  HT
     * @param _R
     *
     * Since the default parameters cannot be expressed as depending on the this
pointer,
     * use the previous function if use the embedded H, H^T and R matrices.
     */
    template<sizeT otherMeasurementSize>
    inline void correct(const vector_T<realT, otherMeasurementSize>& measurement,
        const matrix T<realT, otherMeasurementSize, n>&  H /*= this-> H*/,
        const matrix T<realT, n, otherMeasurementSize>&  HT /*= this-> HT*/,
        const matrix_T<realT, otherMeasurementSize, otherMeasurementSize>& _R /*=
this->_R*/)
    {
        matrix_T<realT, otherMeasurementSize, otherMeasurementSize>
            totalMeasurementNoise =  H *  Pk *  HT +  R;

        //this ref is to make more sense, yet not use more space than needed.
        //Note we don't use totalMeasurementNoise later anyway.
        //it's the inverse of totalMeasurementNoise;
        matrix_T<realT, otherMeasurementSize, otherMeasurementSize>&
            totalMeasurementNoise m1 = totalMeasurementNoise;

        if( !invert(totalMeasurementNoise, totalMeasurementNoise_m1) )
            throw std::string("noise singularity exception !");

        vector T<realT, otherMeasurementSize> innovation = measurement -  H *  xk;

        //filter gain 'K'
        matrix_T<realT, n, otherMeasurementSize> K =
             Pk *  HT * totalMeasurementNoise m1;

         xk =  xk + K * innovation;
         Pk =  Pk - K *  H *  Pk;
    }
}
```

Figure 40 (suite) Extraits du code du filtre de Kalman.

c) ViterbiFiltering

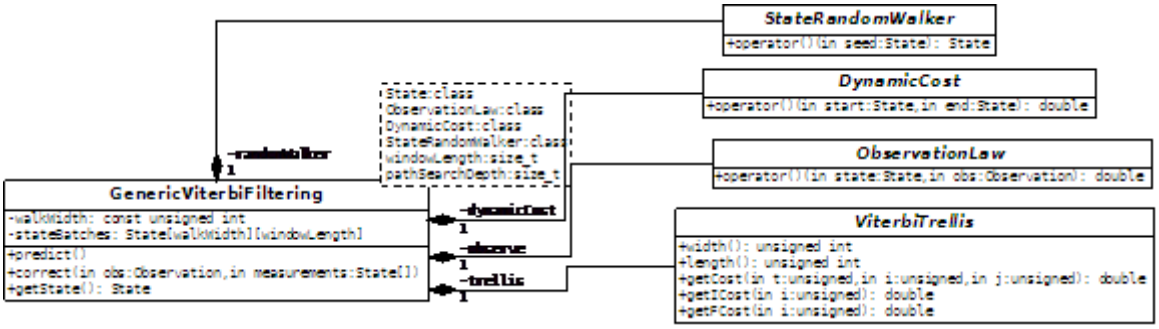


Figure 41 Diagramme UML de la classe-modèle GenericViterbiFiltering.

La classe GenericViterbiFiltering, basée sur l'algorithme de Viterbi, a été conçue afin de gérer un front d'hypothèses sur l'estimation de l'état recherché, et de trouver le plus probable parmi ceux-ci en fonction des observations et des hypothèses précédentes. Ce n'est pas tout à fait un lisseur à proprement parler car le point terminal de la trajectoire la plus probable (dont la probabilité dépend de considérations dynamiques) ne passe pas forcément par le point ainsi estimé à l'instant précédent.

Le front d'hypothèses peut être totalement, partiellement, ou aucunement constitué d'hypothèses fournies. Les hypothèses restantes sont générées par marche aléatoire comme dans un filtre à particule. La Figure 42 montre justement ce comportement variable en fonction du nombre d'hypothèses passées.

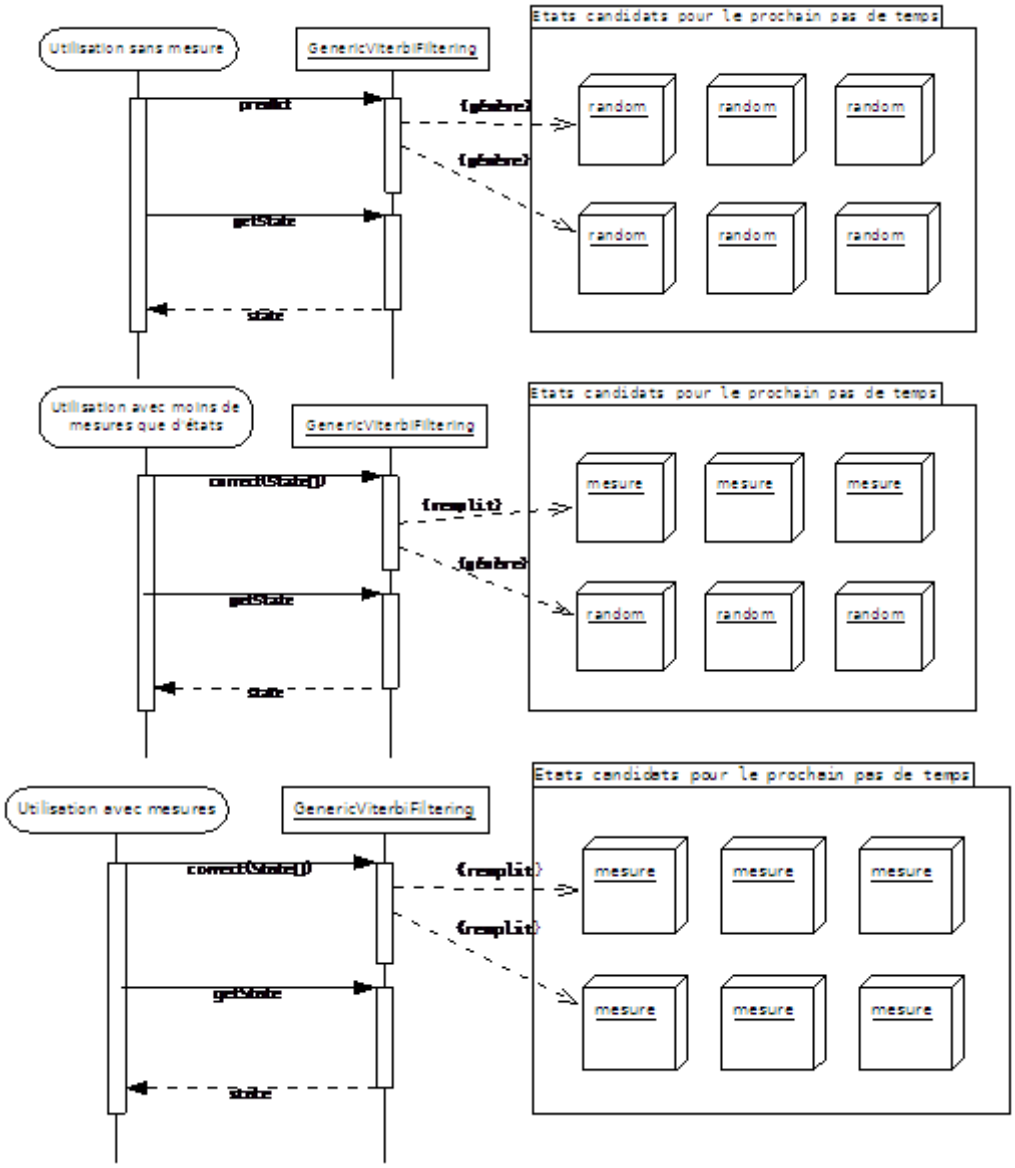


Figure 42 Diagrammes d'utilisation de la classe-modèle GenericViterbiFiltering.

La Figure 43 montre la globalité du fonctionnement de l'algorithme tel qu'il a été implémenté dans un exécutable à part avant d'être transformé en classe telle que décrite précédemment. Le fonctionnement interne est le même mis à part les entrées-sorties.

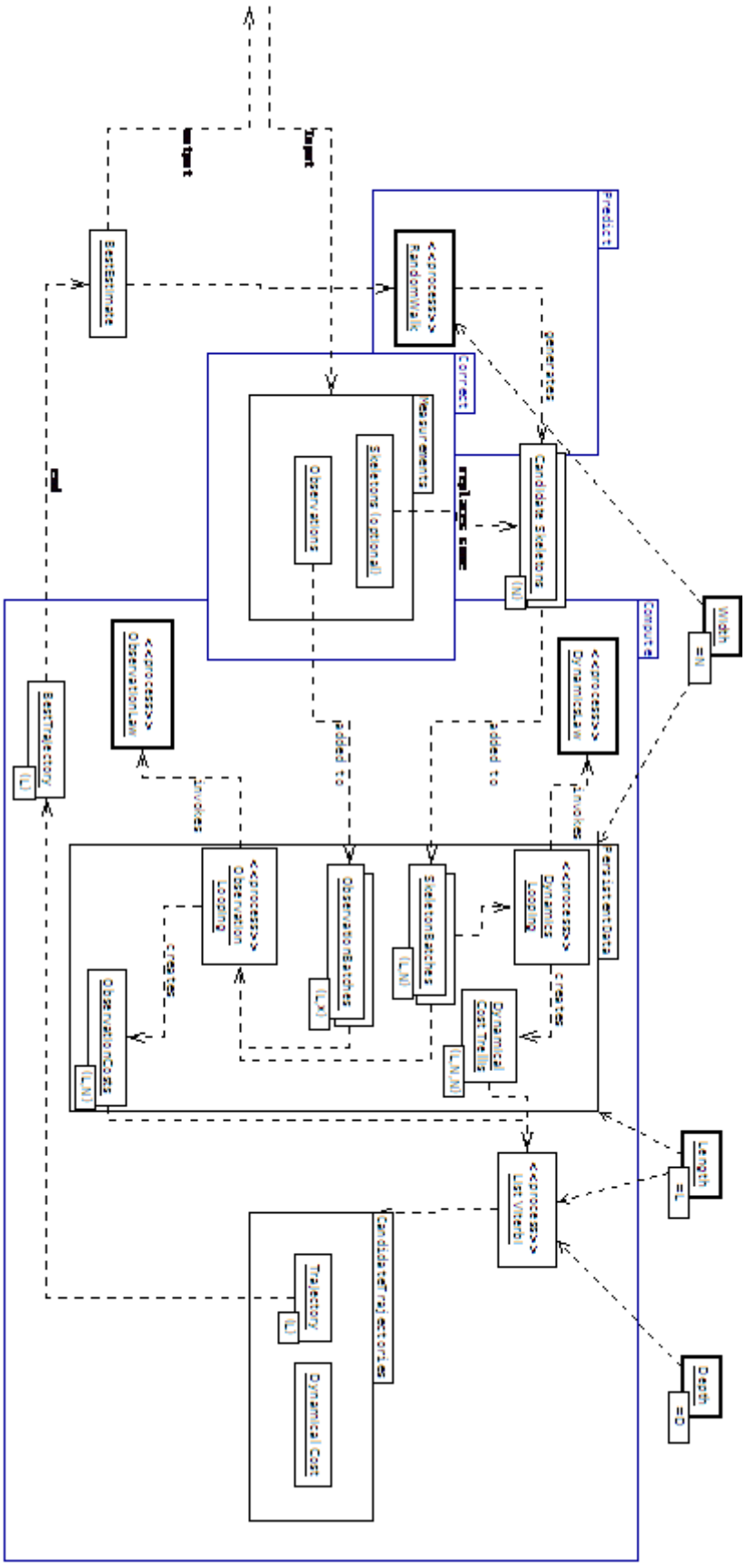


Figure 43 Schéma de fonctionnement global du filtrage de Viterbi.

Les mesures Xtion® sont fournies en entrées. Ce sont à la fois des images et des postures provenant de OpenNI. Les deux modalités sont séparées. Les images sont encapsulées et mises au bout d'une suite d'« observations, » les postures à la fin d'une grille de positions. Ces ensembles sont utilisés pour créer des coûts dynamiques et d'observation pour chaque transition dans le treillis de viterbi. On voit en encadré gras les processus et les variables (largeur du trellis : nombre d'hypothèses considérées ; longueur du treillis : nombre d'instants successifs considérés ; et profondeur : nombre de trajectoires considérées au maximum, non utilisé ici.)

L'algorithme ListViterbi renvoit donc un ensemble de trajectoire dans l'ordre décroissant de probabilité, ainsi la meilleure trajectoire seulement est choisie, et le point terminal est utilisé pour devenir l'estimation courante du filtre.

Le fait de créer des ensembles de postures et d'observations différentes du treillis de Viterbi tient en la volonté de faire de l'algorithme de viterbi un algorithme encore plus générique, et sa classe est aussi un template et donc aussi proche que possible de l'algorithme d'origine.

3. Architecture ROS

Pour les besoins de démonstration, les algorithmes ci-dessus ont été encapsulé dans un ensemble de nœuds ROS afin de faire un démonstrateur temps-réel et un outil abouti pour la communauté.

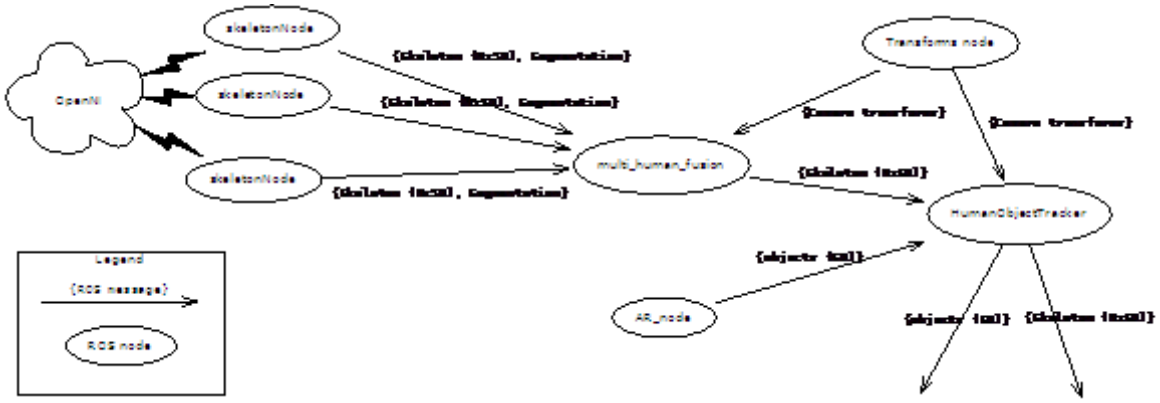


Figure 44 Graphe de communication global ROS.

Plusieurs nœuds sont nécessaire afin d'avoir la fonctionnalité fournie par nos approche :

- Un nœud de perception multi-capteurs. Ce nœud reçoit plusieurs images utilisateur et plusieurs postures. (multi_human_fusion dans la Figure 44)
- Un nœud produisant les images utilisateur et la posture à partir d'un capteur (skeletonNode dans la figure)
- Pour ce qui est de la perception jointe homme-objets, un nœud utilisant une posture (fournie par notre perception multi-capteur) et des détection d'objets, et qui fournira ensuite sa propre version filtrée et traitée.

Chaque nœud est présenté plus en détail dans ce qui suit.

a) KinectNode

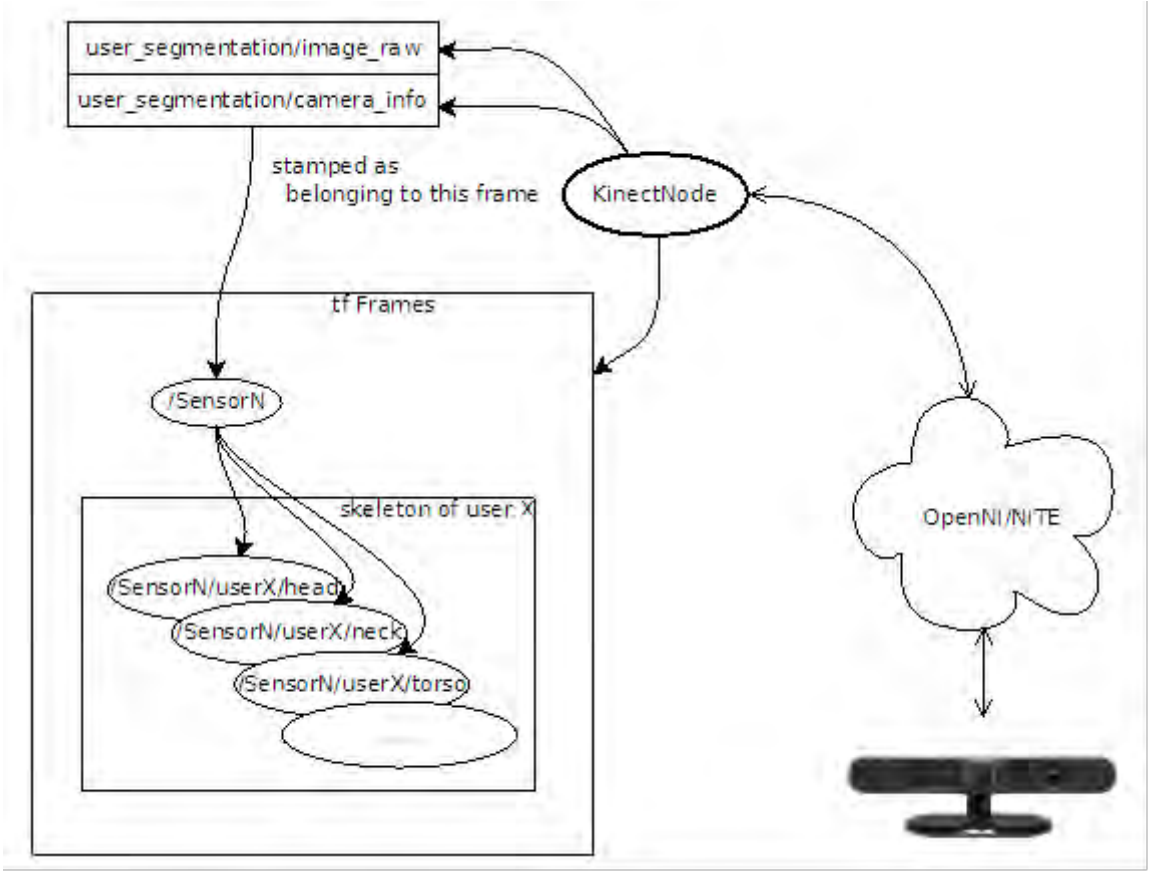


Figure 45 Schéma de communication de KinectNode.

Le nœud KinectNode ici présenté en Figure 45, est singulièrement défini comme publiant des images utilisateurs sur le topic caméra user_segmentation. Plusieurs nœuds sont supposés émettre dans ce topic, ainsi chaque image est étiquetée comme provenant d'un certain capteur, sous la forme d'un référentiel de transformations. Les transformations sont publiées dans ROS avec un mécanisme à part permettant la journalisation des transformations. Ainsi, un nœud souscrivant à ce topic et utilisant ce mécanisme natif peut transformer n'importe quel référentiel représenté par une matrice de transformation dans un autre repère du moment que les repères soient reliées par une unique suite de transformation.

Le nœud publie donc aussi les transformations vers des repères idoines calculés à partir de la posture fournie par OpenNI dans notre cas.

Comme ce nœud est le seul utilisant la librairie OpenNI, n'importe quel nœud utilisant une autre technologie de capture de mouvements fournissant les mêmes données permettra le fonctionnement de notre approche implémentée dans le nœud qui suit, assurant sa généricité avec tout système de détecteur futur.

b) MultiSensorHumanTracker

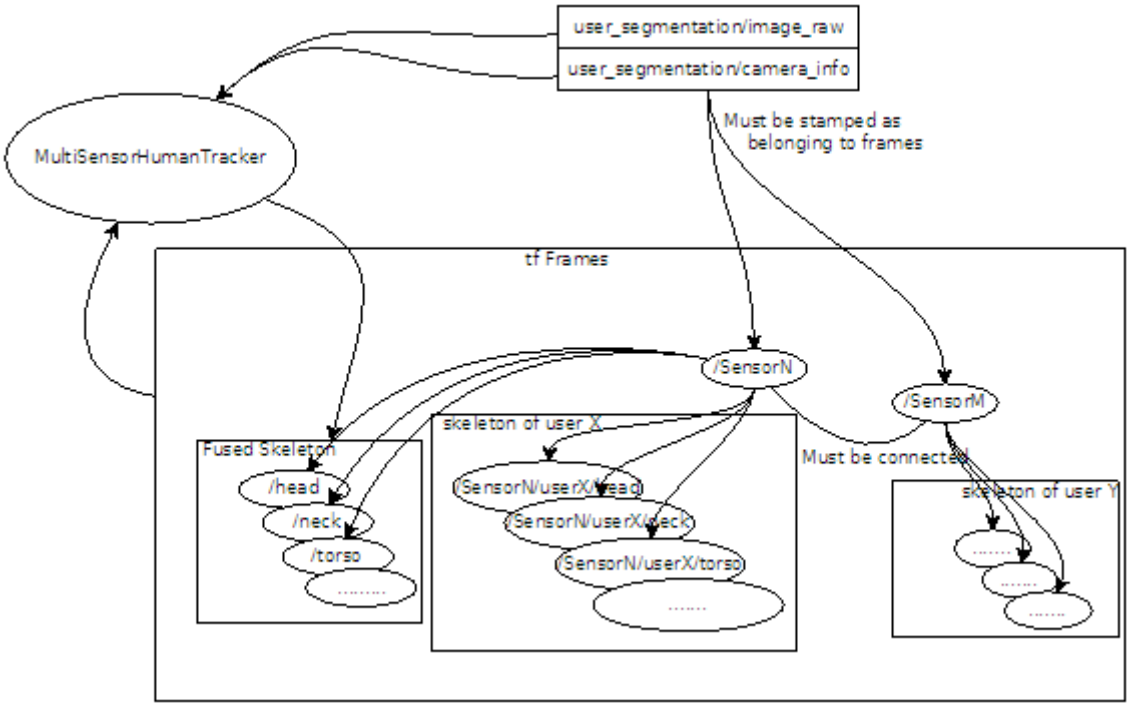


Figure 46 Schéma de communication de MultiSensorHumanTracker

Le nœud MultiSensorHumanTracker est le nœud contenant notre approche. Il souscrit au topic camera user_segmentation et consomme ses images et récupère au même instant les transformations des parties corporelles reliées. Ce nœud a comme paramètre la liste des capteurs qu'il doit utiliser, n'écoutant que ceux-ci s'il existait un réseau important de KinectNode dont certains seraient décorrélés les uns des autres.

Dans le graphe de transformations, un référentiel ne peut avoir qu'un seul parent, et un seul référentiel n'en a aucun : c'est alors le référentiel racine.

Le problème de l'étalonnage spatial est résolu ailleurs dans le graphe et la solution est supposée publiée sur le topic des transformations. Des exécutables ROS permettent de fournir cette fonctionnalité.

Le nœud MultiSensorHumanTracker publie un nouveau squelette, résultat du filtrage. Ce squelette doit être apparenté à un référentiel. Celui du premier capteur de la liste écoutée par le nœud est utilisé comme parent du résultat.

c) Autres nœuds

Le nœud fournissant les données objets est tiré de la communauté Réalité Augmentée. Il encapsule la librairie ARToolkit afin de détecter et publier les marqueurs trouvés dans une image couleur ROS.

4. Contribution à la bibliothèque logicielle Magellium pour la robotique : MagBot

L'entreprise Magellium ayant financé cette thèse a nécessité l'intégration de code produit dans sa propre librairie robotique « MagBot. » Tout le code de cette librairie doit satisfaire à des règles de codages strictes, telles que le nommage des variables et la typographie des structures. Ces règles de codage ont été appliquées et, bien qu'elles soient nombreuses et strictes, elles permettent une meilleure hygiène et homogénéité du code produit.

Ces règles ont été héritées du SDK EDRES du CNES utilisé par Magellium lors de sa participation au projet ExoMars.

C. Conclusion

Dans ce chapitre nous avons exposé tous les détails d'implémentation et le contexte associé : l'entreprise Magellium où j'ai travaillé, et le projet PRACE qui donnait un but général à nos explorations.

Nous pensons que ces développements seront utiles à la communauté tant par leur implémentation conçue pour être à la fois lisible et optimale, d'algorithmes de la communauté, dans un language parfois peu privilégié dans les méthodes scientifiques : le C++.

Conclusion et perspectives

A. Conclusion

Cette thèse portant sur la Capture de mouvements Humains, menée dans le cadre d'un dispositif CIFRE avec Magellium et dans le contexte du projet européen PRACE, a contribué sur plusieurs aspects :

- La formalisation d'une approche originale de capture de mouvements multi-kinects combinant divers atouts : l'utilisation d'intégration temporelle pour un lissage trajectoriel ; une stratégie tracking by detection par reconstruction OpenNI afin de permettre la (ré)-initialisation automatique et prévention des dérives d'estimation ; une plateforme multi-kinects pour la gestion des occultations et auto-occultations. L'approche, conçue pour le temps-réel, tourne à 25 images/secondes pour un système de 3 capteurs. Les performances obtenues (précision, robustesse) sont probantes eu égard notamment à OpenNI.
- L'acquisition pour la validation ci-dessus de bases de données à moyenne et grande échelle de flux Kinects avec en parallèle la vérité terrain fournie par un système de Motion Capture commercial. Trois capteurs d'abord, puis cinq en parallèle pour respectivement 9 et 4 séquences totalisant 21569 et 11139 instants image. A notre connaissance, il n'existe pas de bases publiques multi-Kinects pour la capture de mouvements humain, contrairement à la communauté de vision 2D (détection, suivi, identification de personnes). Elles sont tenues à la disposition de la communauté scientifique sur demande, et nous espérons qu'elles seront utiles pour des études comparatives.
- L'ébauche d'une stratégie originale de perception jointe homme-objets, peu explorée dans la littérature. Cette étude préliminaire est motivée par un constat : il existe beaucoup de travaux sur la capture de mouvements humain mais très peu exploitent des percepts liés aux contextes. Les résultats obtenus, certes qualitatifs, sont prometteurs.

Les choix faits ont été motivés par leur présence dans la littérature mais validés uniquement dans un contexte de suivi 2D. Leur application a été justifiée par les résultats probants dans les deux études réalisées, et publiés dans les articles suivants :

Masse, J.-T., Lerasle, F., Devy, M., Monin, A., Lefebvre, O., & Mas, S. (2013). Human Motion Capture Using Data Fusion of Multiple Skeleton Data. Proceeding of the 15th International Conference, ACIVS 2013 on Advanced Concepts for Intelligent Vision Systems - Volume 8192. Poznan: Springer-Verlag New York, Inc.

Masse, J.-T., Lerasle, F., Devy, M., Monin, A., Lefebvre, O., & Mas, S. (2014). Capture de mouvements Humains par Fusion de Multiples Données Squelettes. Reconnaissance de Formes et Intelligence Artificielle (RFIA) 2014.

Et un article soumis dans la revue internationale Image and Vision Computing.

B. Perspectives

De nombreuses perspectives sont possibles pour ces travaux.

Premièrement, l'approche est extensible à un réseau multi-Kinects à champs disjoints sans perte prévisible de vitesse ou de traque contrairement à l'utilisation indépendante des détecteurs utilisés. Les travaux réalisés conjointement avec le stagiaire G. Marion, qui a implémenté des stratégies de réidentification dans les flux vidéo durant son stage, ont montré des résultats prometteurs comme le montre la Figure 47. Il existe cependant deux possibilités d'intégration de son travail : intégration amont, on utilise les signatures résultantes afin d'instancier un filtre multi-capteur comme présenté ici, par utilisateur. L'intégration aval serait de faire gérer les probabilités d'association par le filtrage de Viterbi en utilisant la signature comme variable d'état.

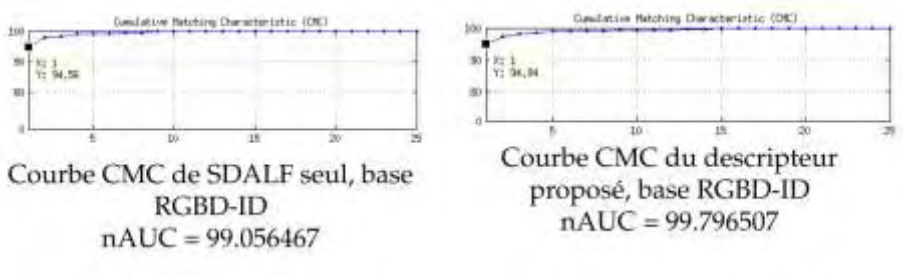


Figure 47 Courbes Cumulative Matching Curve sur dataset ETHZ1 (Leibe, Schindler, & Van Gool, 2007) de méthode de réidentification proposé par G. Marion (Stage 2014).

À moyen terme, des alternatives doivent être explorées pour la loi d'observation. La projection 2D binaire dans de multiples points de vue capteur pourrait être améliorée avec l'utilisation des données couleur ou de profondeur sous le masque que nous exploitons déjà, ou bien la reconstruction d'un modèle binaire entièrement 3D (donc en voxels) recréé à partir des multiples images de profondeur, en s'inspirant du travail de (Filali, Masse, Lerasle, Boizard, & Devy, 2013) validé dans un contexte bas-niveau.

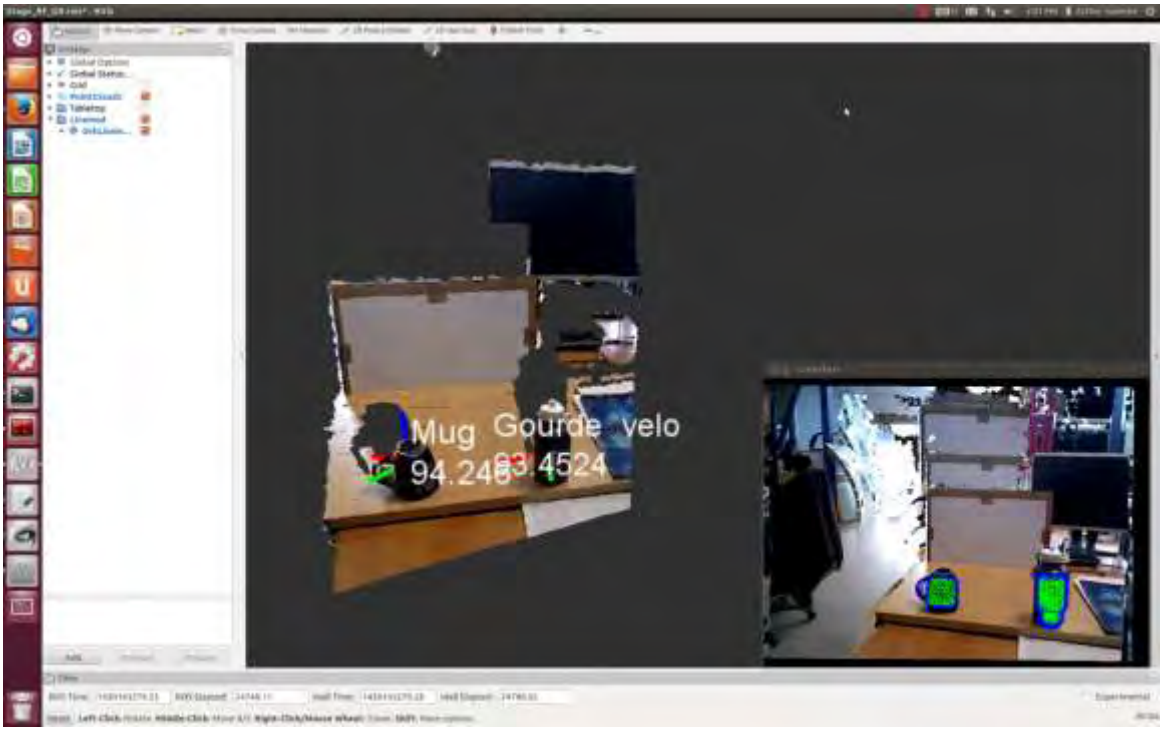


Figure 48 Illustration du fonctionnement de ORK dans RVIZ, l'outil de visualisation de ROS.

L'utilisation des marqueurs de réalité augmentée afin de simuler la perception des objets doit aussi être dépassée. L'utilisation de la librairie ORK (Object Recognition Kitchen), qui est une librairie python/C++ fonctionnant sur ROS créée par Willow Garage, reprise par Vincent Rabaud (Aldebaran Robotics), ancien membre de Willow Garage, pour permettre de faire de la détection d'objets de façon simple, pourrait être une bonne piste. La Figure 48 montre l'intégration et le fonctionnement de ORK dans RVIZ de ROS, pour des objets de la vie de tous les jours (gourde et mug).

A plus long terme, les possibilités sont très nombreuses.

L'approche est visiblement compatible avec un système de MoCap commercial, qui présente cependant des difficultés d'exploitation dûs à la faible couverture et/ou à la présence de marqueurs qui n'existent pas. L'utilisation concomittante d'un système avec des capteurs similaires à ceux utilisés ici permettrait une amélioration du système de MoCap. Inversement, il serait intéressant d'utiliser un système de capteurs hétérogènes, notamment avec l'arrivée du Kinect2, plus précis. Comment gérer la précision de chaque détecteur, est certainement la question la plus intéressante à poser pour l'intégrer à notre approche.

L'utilisation d'image binaire tend à faire penser aux cartes graphiques et à l'avênement de leur utilisation dans la communauté basée vision. Une approche avec des hypothèses stochastiques telle que l'on a écarté précédemment pour cause de la trop grande complexité des calculs pourrait devenir possible.

La perception conjointe homme-objet est faite à posteriori du filtrage basé Viterbi avec plusieurs capteurs. En théorie, il serait possible d'intégrer les perceptions des objets dans le filtrage Viterbi, et de gérer les hypothèses à l'intérieur en multipliant les possibili-

tés d'association comme éléments du treillis. Les positions des objets permettraient aussi d'améliorer la segmentation utilisateur, car cette segmentation, faite par OpenNI, a le désavantage de considérer les objets comme des parties corporelles à labelliser, ce qui fausse la détection en cas de manipulation.

Enfin, l'utilisation concrète de notre approche de capture liée à une perception objet permet tout naturellement d'implémenter un système d'apprentissage par imitation tel qu'inspiré du contexte PRACE. Pour cette modalité, les approches sont diverses : l'utilisation encore une fois d'une recherche de trajectoire dans l'espace des prises et dépose pourrait être faite, avec l'idée d'intégrer des coûts liés aux manipulations et aux observations faites sur les objets et, éventuellement, un input utilisateur afin de guider la reconnaissance.

Table des figures

Tableaux

Remerciements

Jamais je n'aurais imaginé cette thèse comme elle s'est passée, le jour où mon encadrant de stage, futur encadrant principale de ma thèse, m'a parlé d'un partenaire industriel qui cherchait quelqu'un pour un projet européen. C'est ainsi que ma vie prit un tournant que je n'avais pas imaginé. Je remercie donc principalement mon encadrant, Monsieur F. Lerasle, pour m'avoir embarqué là-dedans et m'avoir guidé tout du long.

Je remercie aussi Magellium, surtout en les personnes de O. Lefebvre et S. Mas avec qui je travaillais tous les jours (quand j'étais à Magellium), et tous les autres qui se reconnaitront (Arnaud, Aurélien, Benoit, Denis, Fabrice, Jean-Brice, et les autres que j'oublie !)

Cette thèse aura aussi été l'occasion de connaître le monde du travail (c'est vrai qu'il est important de le connaître !) et les joies de la double hiérarchie.

Et surtout un grand merci, que j'espère sincèrement réciproque, à mes collègues doctorants, avec qui j'ai rigolé, mais aussi réfléchi, échangé et amélioré au court des jours mes compétences et la science en général. Résoudre des problèmes de prépa, de programmation, d'algorithmique, d'apprentissage, a toujours été une joie.

Bibliographie

Agarwal, A., & Triggs, B. (2005). Monocular Human Motion Capture with a Mixture of Regressors. *Computer Vision and Pattern Recognition - Workshops, 2005. CVPR Workshops. IEEE Computer Society Conference on.*

Agarwal, A., & Triggs, B. (2006). Recovering 3D Human Pose from Monocular Images. *IEEE Transcriptions Pattern Analysis and Machine Intelligence, 28*(1), 44-58.

Andersen, M. R., Jensen, T., Lisouski, P., Mortensen, A. K., Hansen, M. K., Gregsersen, T., et al. (2012). *Kinect Depth Sensor Evaluation for Computer Vision Applications.* Aarhus: Aarhus University.

Andriluka, M., Roth, S., & Schiele, B. (2008). People-tracking-by-detection and people-detection-by-tracking. *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, (pp. 1-8).

Bailey, S. W., & Bodenheimer, B. (2012). A Comparison of Motion Capture Data Recorded From a Vicon System and a Microsoft Kinect Sensor. *Proceedings of the ACM Symposium on Applied Perception* (pp. 121-121). New York, NY, USA: ACM.

Berger, K. (2013, Nov 4). A State Of the Art Report on Research in Multiple {RGB-D} sensor Setups. *CoRR*.

Brubaker, M. A., Fleet, D. J., & Hertzmann, A. (2010). Physics-Based Person Tracking Using the Anthropomorphic Walker. *International Journal of Computer Vision, 87*(1-2), 140-155.

Caon, M., Yue, Y., Tscherrig, J., Mugellini, E., & Abou Khaled, O. (2011). Context-aware 3D gesture interaction based on multiple kinects. *AMBIENT 2011, The First International Conference on Ambient Computing, Applications, Services and Technologies* (pp. 7-12). IARIA XPS Press.

Corazza, S., Mündermann, L., Gambaretto, E., Ferrigno, G., & Andriacchi, T. P. (2010). Markerless Motion Capture through Visual Hull, Articulated ICP and Subject Specific Model Generation. *International Journal of Computer Vision, 87*(1-2), 156-169.

Deutscher, J., & Reid, I. (2005). Articulated Body Motion Capture by Stochastic Search. *International Journal of Computer Vision, 61*(2), 185-205.

Filali, W., Masse, J.-T., Lerasle, F., Boizard, J.-L., & Devy, M. (2013). Human Motion Capture Using 3D Reconstruction Based on Multiple Depth Data. *Systems, Man, and Cybernetics (SMC), 2013 IEEE International Conference on*, (pp. 870-875).

Garcia, J., & Zalevsky, Z. (2008, oct 7). *Brevet n° 7,433,024.* United States.

Girshick, R., Shotton, J., Kohli, P., Criminisi, A., & Fitzgibbon, A. (2011). Efficient regression of general-activity human poses from depth images. *Computer Vision (ICCV), 2011 IEEE International Conference on*, (pp. 415-422).

Gond, L., Sayd, P., Chateau, T., & Dhome, M. (2009). A regression-based approach to recover human pose from voxel data. *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, (pp. 1012-1019).

Goyat, Y., Chateau, T., & Bardet, F. (2010). Vehicle Trajectory Estimation Using Spatio-temporal MCMC. *EURASIPJournal on Advances in Signal Processing*, 25:1-25:8.

Gupta, A., Mittal, A., & Davis, L. (2008). Constraint Integration for Efficient Multiview Pose Estimation with Self-Occlusion. *Pattern Analysis and Machine Intelligence, IEEE Transactions on, 30*(3), 493-506.

Herrera C., D., Kannala, J., & Heikkila, J. (2012). Joint Depth and Color Camera Calibration with Distortion Correction. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 34*(10).

Hofmann, M., & Gavrila, D. (2012). Multi-view 3D Human Pose Estimation in Complex Environment. *International Journal of Computer Vision*, 103-124.

INRIA Rhones-Alpes. (2003, Juillet 11). *PETS-ECCV 2004 | Benchmark Data*. Consulté le 04 25, 2015, sur PRIMA: http://www-prima.inrialpes.fr/PETS04/caviar_data.html

Khan, Z., Balch, T., & Dellaert, F. (2004). An MCMC based particle filter for tracking multiple interacting targets. Dans T. Pajdla, & J. Matas (Éd.), *Computer Vision - ECCV 2004. 3024*, pp. 279-290. Springer Berlin Heidelberg.

Khoshelham, K., & Elberink, S. O. (2012). Accuracy and Resolution of Kinect Depth Data for Indoor Mapping Applications. *Sensors, 12*(2), 1437-1454.

Kjellström, H., Romero, J., & Kragic, D. (2011). Visual object-action recognition: inferring object affordances from human demonstrationComputer Vision and

Image Understanding (CVIU'11). *Computer Vision and Image Understanding (CVIU'11)*, *115*, pp. 81-90.

Koppula, H., Gupta, S., & Saxena, A. (2013). Learning human activities and object affordances from RGB-D videos. *Int. Journal of Robotic Research (IJRR'13), 32*(8), 951-970.

Larson, R., & Peschon, J. (1966). A dynamic programming approach to trajectory estimation. *Automatic Control, IEEE Transactions on, 11*(3), 537-540.

Leibe, B., Schindler, K., & Van Gool, L. (2007). Coupled Detection and Trajectory Estimation for Multi-Object Tracking. *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, (pp. 1-8).

Li, K., Dai, Q., & Xu, W. (2011). Markerless Shape and Motion Capture From Multiview Video Sequences. *Circuits and Systems for Video Technology, IEEE Transactions on, 21*(3), 320-334.

Maloney, R. (2013, January 4). *Movement Analysis Products*. (Motion Analysis Corporation) Consulté le January 11, 2013, sur http://www.motionanalysis.com/html/movement/products.html

Maloney, R. (s.d.). *Calcium Solver.* Consulté le Janvier 12, 2015, sur Site Web Motion Analysis: http://www.motionanalysis.com/html/animation/calcium.html

Masse, J.-T., Lerasle, F., Devy, M., & Lefebvre, O. (2015). Human Motion Capture by Pose Recovery and Temporal Integration from Multi-Depth Sensor Setup (soumis). *Image and Vision Computing*.

Masse, J.-T., Lerasle, F., Devy, M., Monin, A., Lefebvre, O., & Mas, S. (2013). Human Motion Capture Using Data Fusion of Multiple Skeleton Data. *Proceeding of the 15th International Conference, ACIVS 2013 on Advanced Concepts for Intelligent Vision Systems - Volume 8192.* Poznan: Springer-Verlag New York, Inc.

Masse, J.-T., Lerasle, F., Devy, M., Monin, A., Lefebvre, O., & Mas, S. (2014). Capture de Mouvements Humains par Fusion de Multiples Données Squelettes. *Reconnaissance de Formes et Intelligence Artificielle (RFIA) 2014.*

Maybeck, P. S. (1979). *Stochastic models, estimation, and control.* Academic Press.

Mekonnen, A. A., Lerasle, F., & Herbulot, A. (2013). External Cameras and a Mobile Robot for Enhanced Multi-person Tracking. *International Conference on Computer Vision Theory and Applications (VISAPP).*

Microsoft. (s.d.). *Kinect for Windows*. Consulté le Janvier 09, 2015, sur http://www.microsoft.com/en-us/kinectforwindows/

Moeslund, T. B., Hilton, A., & Krüger, V. (2006). A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding, 104*(2-3), pp. 90 - 126.

Moore, D., Essa, I., & Hayes, M. (1999). Exploiting human actions and object context for recognition tasks. *Int. Conf. on Computer Vision (ICCV'99)*, (pp. 80-86). Corfou, Grêce.

Ofli, F. a., Kurillo, G., Vidal, R., & Bajcsy, R. (2013). Berkeley MHAD: A comprehensive Multimodal Human Action Database. *Applications of Computer Vision (WACV), 2013 IEEE Workshop on* (pp. 53-60). IEEE.

Ogawara, K., Li, X., & Ikeuchi, K. (2007). Marker-less Human Motion Estimation using Articulated Deformable Model. *Robotics and Automation, 2007 IEEE International Conference on* (pp. 46-51). IEEE.

Okuma, K., Taleghani, A., de Freitas, N., Little, J. J., & Lowe, D. G. (2004). A Boosted Particle Filter: Multitarget Detection and Tracking. *Computer Vision - ECCV 2004. 3021*, pp. 28-39. Springer Berlin Heidelberg.

Park, S., & Kautz, H. (2008). Hierarchical recognition of activities of daily living using multi-scale, multi-perspective Vision and RFID. *Int. Conf. on Intelligent Environments.* Seattle.

Perez, P., Vermaak, J., & Blake, A. (2004, Mars). Data fusion for visual tracking with particles. *Proceedings of the IEEE, 92*(3), pp. 495-513.

Pirsiavash, H., Ramanan, D., & Fowlkes, C. (2011). Globally-optimal greedy algorithms for tracking a variable number of objects. *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on* (pp. 1201-1208). IEEE.

Poppe, R., & Poel, M. (2006). Comparison of silhouette shape descriptors for example-based human pose recovery. *Automatic Face and Gesture Recognition, 2006. FGR 2006. 7th International Conference on*, (pp. 541-546).

Rafibakhsh, N., Gong, J., Siddiqui, M., Gordon, C., & Lee, H. (2012). Analysis of XBOX Kinect Sensor Data for Use on Construction Sites : Depth Accuracy and Sensor Interference Assessment. Dans *Construction Research Congress 2012* (pp. 848-857).

Shapovalova, N., Gong, W., Pedersoli, M., Roca, F., & Gonzalez, J. (2011). On importance of interactions and context in human action recognition. *Iberian Conf. on Pattern Recognition and Image Analysis (IbPRIA'11).* Las Palmas.

Shotton, J., Fitzgibbon, A., Cook, M., Sharp, T., Finocchio, M., Moore, R., et al. (2011). Real-time human pose recognition in parts from single depth images.

Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on, 1297 -1304.

Shpunt, A., & Zalevsky, Z. (2011, Nov 1). *Brevet n° 8,050,461.* United States.

Siegwart, R., Nourbakhsh, I. R., & Scaramuzza, D. (2011). *Introduction to autonomous mobile robots.* MIT Press.

Sigal, L., Balan, A. O., & Black, M. J. (2010, March). HumanEva: Synchronized Video and Motion Capture Dataset for Evaluation of Articulated Human Motion. *International Journal of Computer Vision, 87*(1-2), 4-27.

Tenorth, M., Bandouch, J., & Beetz, M. (2009). The TUM Kitchen Data Set of everyday manipulation activities for motion tracking and action recognition. *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on* (pp. 1089-1096). IEEE.

Viola, P., & Jones, M. (2001). Rapid Object Detection using a Boosted Cascade of Simple Features. *Conference on Computer Vision and Pattern Recognition. 1.* Kauai, Hawaii: IEEE Computer Society.

Viterbi, A. (2006). A personal history of the Viterbi algorithm. *Signal Processing Magazine, IEEE, 23*(4), 120-142.

Yu, Q., Medioni, G., & Cohen, I. (2007). Multiple Target Tracking Using Spatio-Temporal Markov Chain Monte Carlo Data Association. *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on* (pp. &-8). IEEE.

Zhang, L., Sturm, J., Cremers, D., & Lee, D. (2012). Real-Time Human Motion Tracking using Multiple Depth Cameras. *Proc. of the International Conference on Intelligent Robot Systems (IROS)*.

Zhuang, H., Zhao, B., Ahmad, Z., Chen, S., & Low, K. S. (2012). 3D depth camera based human posture detection and recognition Using PCNN circuits and learning-based hierarchical classifier. *Neural Networks (IJCNN), The 2012 International Joint Conference on*, (pp. 1-5).

Ziegler, J., Nickel, K., & Stiefelhagen, R. (2006). Tracking of the Articulated Upper Body on Multi-View Stereo Image Sequences. *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on* (pp. 774-781). IEEE.

Human Motion Capture by RGB-D sensors

Simultaneous apparition of depth and color sensors and super-realtime skeleton detection algorithms led to a surge of new research in Human Motion Capture. This feature is a key part of Human-Machine Interaction. But the applicative context of those new technologies is voluntary, fronto-parallel interaction with the sensor, which allowed the designers certain approximations and requires a specific sensor placement. In this thesis, we present a multi-sensor approach, designed to improve robustness and accuracy of a human's joints positionning, and based on a trajectory smoothing process by temporal integration, and filtering of the skeletons detected in each sensor. The approach has been tested on a new specially constituted database, with a specifically adapted calibration methodology. We also began extending the approach to context-based improvements, with object perception being proposed.

AUTEUR : JEAN-THOMAS MASSE

TITRE : Stratégies Kinect pour la Capture de mouvements Humains

DIRECTEURS DE THESE : FREDERIC LERASLE et MICHEL DEVY

LIEU ET DATE DE SOUTENANCE : 25 Septembre 2015, Salle Hourgade

---

RESUME

L'arrivée simultanée de capteurs de profondeur et couleur, et d'algorithmes de détection de squelettes super-temps-réel a conduit à un regain de la recherche sur la capture de mouvements humains. Cette fonctionnalité constitue un point clé de la communication Homme-Machine. Mais le contexte d'application de ces dernières avancées est l'interaction volontaire et fronto-parallèle, ce qui permet certaines approximations et requiert un positionnement spécifique des capteurs. Dans cette thèse, nous présentons une approche multi-capteurs, conçue pour améliorer la robustesse et la précision du positionnement des articulations de l'homme, et fondée sur un processus de lissage trajectoriel par intégration temporelle, et le filtrage des squelettes détectés par chaque capteur. L'approche est testée sur une base de données nouvelle acquise spécifiquement, avec une méthodologie d'étalonnage adaptée spécialement. Un début d'extension à la perception jointe avec du contexte, ici des objets, est proposée.

MOTS-CLES Robotique industrielle ; Perception multi-sensorielle ; Intégration Temporelle ;

DISCIPLINE ADMINISTRATIVE Robotique et Informatique

INTITULE ET ADRESSE DE L'U.F.R. OU DU LABORATOIRE :

LAAS-CNRS
7, avenue du Colonel Roche
BP 54200
31031 Toulouse cedex 4
France