Interprétation de Mouvements Temps Réel

Mathieu Barnachon[1] Saïda Bouakaz[1] Boubakeur Boufama[2] Erwan Guillou[1]

[1] Université de Lyon, CNRS
Université Lyon 1, LIRIS, UMR5205, F-69622, France
firstname.lastname@liris.cnrs.fr
[2] University of Windsor
School of Computer Science, Windsor, Ontario Canada N9B 3P4
boufama@uwindsor.ca

Résumé

Dans cet article, nous proposons une méthode pour l'interprétation des mouvements d'un humain en temps réel, en s'appuyant sur des données fournies par une étape de capture (MoCap). Nous nous basons sur le paradigme examplars-based afin d'apprendre des actions à l'aide d'un représentant unique. Nous avons formalisé le concept de flux d'actions reconnaissables – provenant d'un système de capture de mouvements temps réel – à l'aide d'un automate de mouvements. Cet automate est utilisé pour reconnaître les actions, ainsi que pour ajouter de nouvelles actions à la volée. La méthode proposée est linéaire et incrémentale. Elle permet une utilisation temps-réel du système. De plus, chaque action peut être reconnue en ligne et ne nécessite pas d'attendre la fin de l'action pour l'identifier. Des expérimentations sur des données synthétiques, provenant de la base de MoCap de CMU [1], complétées par des données réelles, issues du dispositif Kinect®, montrent l'efficacité de notre méthode.

Mots Clef

Capture de mouvements, Reconnaissance d'actions, Interprétation temps-réel, *Examplars-based*.

Abstract

This paper proposes a new method for real-time human motion recognition using Motion Capture (MoCap) data. In particular, our method uses examplars to learn actions, without the need for learning intra class variations. We have formalized streamed recognizable actions, coming from an online MoCap engine, into a motion graph, similar to an animation motion graph. This graph is used as an automaton to recognize known actions as well as to add new ones. We have defined and used a spatio-temporal metric for similarity measurements to achieve more accurate feedbacks on classification. The proposed method has the advantage of being linear and incremental, making the recognition process very fast and the addition of a new action straightforward. Furthermore, actions can be recognized with a score, even before they are fully completed. Thanks to the use of a skeleton-centric coordinate system, our recognition method has become view-invariant. We have successfully tested our action recognition method on both synthetic and real data. The latter was obtained from live input videos using either our online markerless MoCap engine or the Kinect acquisition system.

Keywords

Motion Capture, Motion Recognition, Real-Time Motion Interpretation, Examplars-based.

1 Introduction

Des besoins en interactions dans des domaines, tels que les jeux vidéo [2], la surveillance « intelligente » [3], la télévision interactive et la réalité augmentée posent de nouveaux problèmes dans le domaine de l'interprétation de mouvements. Résoudre ces problèmes grâce à la vision par ordinateur, ouvre des perspectives intéressantes dans le domaine des interfaces Homme - Machine et permet de s'affranchir du triangle « écran – clavier – souris ». Pour qu'elles puissent être adoptées par les utilisateurs, les interactions Homme - Machine doivent être simples, naturelles et conviviales. Bien que des solutions aient été proposées, elles restent globalement assez simplistes, limitées par le type d'interaction proposées et leurs faibles abstractions. Certaines limitations sont liées au fait que dans la plupart des modes d'interaction, seules certaines parties du corps sont exploitées. Ce dernier point est largement influencé par la difficulté d'identifier les différentes parties du corps. Des réponses ont été apportées par l'utilisation de marqueurs. Zhang *et al.* [4] sont parmi les premiers à proposer une solution basée vision, pour interagir avec un ordinateur. Les auteurs utilisent une feuille de papier comme marqueur, ils

sont capables de déplacer la souris, de cliquer, de tourner une image et de dessiner sur un tableau blanc virtuel. Le besoin de s'affranchir de marqueurs pour simplifier l'interaction a également été proposé dans [5] Bien que de telles solutions soient suffisantes pour se substituer à une souris ou à quelques raccourcis claviers, elles ne suffisent pas à rendre les interactions naturelles.
Parallèlement, d'autres recherches dans le domaine de l'analyse d'activités humaines (*human activity monitoring*) [6, 7, 8] ont permis d'interagir d'une façon plus riche avec un ordinateur. Basées sur des algorithmes d'apprentissage par ordinateur (*machine learning*), ces méthodes requièrent une variance intra-classe importante pour reconnaître des activités. Cette exigence en apprentissage limite le nombre d'activités reconnues : l'extension à de nouvelles actions implique une ré-évaluation de l'ensemble de la base. Un tel processus est généralement beaucoup trop coûteux pour être fait en temps réel. Pour une étude sur les méthodes d'*activity monitoring*, on peut se référer à l'étude de Poppe [9].

Dans la suite de l'état de l'art, nous avons fait le choix d'organiser la bibliographie suivant trois méthodologies : les techniques 2D, généralement une caméra ; les méthodes basées sur la vision multi-vues (stéréo, ...) et enfin les méthodes purement 3D, qui s'appuient sur un volume ou un squelette d'animation 3D, *etc.*

Méthodes 2D Fujiyoshi *et al.* [10] furent parmi les premiers à proposer une extraction du squelette d'un personnage à partir du contour de la silhouette, afin d'identifier des actions de marche et de course. Leur solution possède l'avantage de la simplicité, mais elle n'est réellement efficace que sur des actions simples. Bobick et Davis [11] ont proposés une méthode reposant sur des canevas spatio-temporels (*spatio-temporal template*). Ceux-ci permettent, à l'aide d'une base d'actions précédemment extraites, de reconnaître des activités humaines. La méthode fonctionne en temps réel, mais ne permet pas l'ajout rapide des nouvelles activités à cette base. À partir des primitives extraites de silhouettes, Elgammal *et al.* [12] ont conjugués le paradigme des *examplars* et les chaînes de Markov pour reconnaître des actions. Cette solution s'avère flexible et efficace en temps de calcul, néanmoins, elle reste trop dépendante des primitives 2D, et ne sont pas assez discriminantes dans bien des cas. Xiong et Liu [13] utilisent les chaînes de Markov sur des descripteurs extraits des silhouettes. Cette méthode a l'avantage d'être simple, mais son application peut difficilement dépasser le cadre d'interactions élémentaires. En étendant le concept d'historique de mouvements au cas des silhouettes, Ahmad et Lee [14] utilisent un apprentissage par machine à support de vecteurs (SVM). Les inconvénients de cette démarche sont le nombre important de paramètres nécessités par le SVM et la sensibilité de leur système à la qualité des silhouettes extraites et au point de vue de la caméra.

Nous pouvons noter que la plupart des méthodes utilisant des silhouettes souffrent de l'extraction d'un squelette trop simple. Ce dernier se présente souvent sous la forme d'une structure en étoile à cinq branches. En outre, l'ensemble des méthodes 2D sont dépendantes du point de vue de la caméra – à l'apprentissage, ainsi qu'à la reconnaissance – et sont vulnérables aux ambiguïtés visuelles résultant de la projection du monde réel sur la surface d'un capteur 2D.

Méthodes 2D Étendues Van den Berg *et al.* [15] ont proposés une solution générique pour interagir avec un ordinateur. Pour cela, ils détectent des configurations de doigts, définis à priori, ainsi que la position des yeux à l'aide d'un système de stéréovision. Le principe général consiste à manipuler la souris à l'aide des deux mains : l'une des mains manipule le pointeur, la seconde détermine et valide l'action visée (clique, clique-droit, zoom, ...). Cette solution a initié une voie vers des interactions naturelles – elle exploite les mouvements des mains et la direction du regard – sans recours à des marqueurs. Son utilisation reste néanmoins adaptée aux écrans de grandes tailles (vidéoprojecteurs). La rigidité des configurations, peu intuitives, limite son adoption par le grand public. Li *et al.* [5] proposent de simuler un écran tactile à l'aide d'un système de stéréovision. Ce mode d'interaction, calqué sur les habitudes des écrans tactiles, permet de s'affranchir d'un matériel onéreux, cependant, il n'apporte pas de réelle innovation d'interaction.

Par delà les utilisations classiques des interfaces, Okada et Stenger [16] ont présentés une méthode, basée silhouettes, capturant le mouvement et le transférant dans un environnement virtuel temps réel. Ils offrent ainsi une liberté d'interaction au travers d'un avatar. Pour cela, ils construisent une hiérarchie de formes à partir de silhouettes virtuelles de différentes statures, et retrouvent le mouvement (MoCap). Ils n'utilisent qu'une caméra, mais font une interprétation tridimensionnelle du mouvement. La dépendance aux silhouettes impose une connaissance *a priori* des configurations, et est très coûteuse en terme d'évaluation – utilisation d'un calculateur de type *Cell* ^{TM} – ce qui la rend difficilement utilisable sur un PC classique. Leur travail est parmi les premiers à exploiter l'ensemble du corps humain pour proposer des interactions. Néanmoins, il souffre, au niveau interprétation, de l'absence de mécanisme de reconnaissance, et ne propose que des interactions « directes » par l'intermédiaire de l'avatar.

Méthodes 3D Yilmaz et Shah [17] ont proposés une méthode basée sur l'extraction de silhouettes, qui effectue une extrusion spatio-temporelle. Ils détectent des points importants sur ce volume qu'ils utilisent comme caractéristiques pour catégoriser leurs actions. La correspondance temporelle, utile à la détection et à la construction du volume, est très couteuse en temps de calcul et sensible aux erreurs liées à la précision des silhouettes. Huang et Trivedi [18] ont créé le concept d'histogramme cylindrique, dans une représentation voxelique. En utilisant une reconnaissance par chaînes de Markov cachées, ils sont en mesure d'assurer une indépendance au point de vue. De même, Weinland *et al.* [19] utilisent des chaînes de Markov cachées en s'appuyant sur le paradigme *examplars*. Une autre méthode a été proposée par Parameswaran et Chellappa [20], qui utilise la capture de mouvements avec marqueurs. Les auteurs préconisent l'utilisation d'un espace où le mouvement serait invariant (exprimé par la trajectoire). Dans un tel formalisme, chaque action doit être modélisée séparément. Il ressort de cette lecture que pour assurer l'indépendance par rapport au point de vue, il est souvent nécessaire d'utiliser un squelette d'animation [20]. Le recours à un squelette d'animation facilite l'utilisation des trajectoires décrivant le mouvement, et permet ainsi une meilleure abstraction de l'action [21, 7].

Certains travaux, comme ceux proposé par Baak *et al.* [22] ont utilisés l'interprétation et la reconnaissance du mouvement pour améliorer les résultats de la capture de mouvements. Les résultats de l'interprétation leur permettent de créer une base de contraintes : pieds en contact avec le sol durant la marche, *etc.* Ces contraintes leur permettent de corriger les résultats de la capture par des caméras. Bien que cela soit utile en capture de mouvements, la méthode n'est pas adaptée à la reconnaissance d'actions quelconques.

Signalons que le but de nos travaux n'est pas de revenir sur les méthodes de MoCap. Nous partons du principe qu'un squelette d'animation est extrait de données de capture, ces données provenant d'un système d'acquisition multi-caméras ou Kinect. Dans ce dernier cas, les travaux de Shotton *et al.* [23] fournissent des données suffisamment précises pour l'extraction du squelette. Ainsi le choix que nous avons fait dans le présent article est l'interprétation d'actions à partir d'un squelette cinématique.

La suite de l'article est organisée de la façon suivante : en suivant le principe de [21], nous représentons les données de capture par des trajectoires. La section 2 introduit la représentation des données de capture par des trajectoires réduites. Dans la section 3, nous introduisons une métrique spatio-temporelle qui permettra de comparer des actions. Notre système de reconnaissance est implémenté sous forme d'un automate de reconnaissance en ligne des actions, qui sera présenté en section 4. Enfin, des résultats provenant de données synthétiques et de données réelles, ainsi que la conclusion, seront présenté en section 5 et 6.

2 Estimations des Trajectoires

Dans cette section, nous considérons les données de MoCap sous forme d'une chaîne cinématique hiérarchique – le squelette d'animation – exprimé dans un repère tridimensionnel. Nous abordons le problème de la transformation de ces données sous forme de trajectoires. Pour donner un cadre formel à la notion d'apprentissage et la reconnaissance d'actions, nous introduisons les définitions suivantes, où une action est découpée en actions élémentaires (*action element*).

Definition 1. *Un* `partial action cut` *est une discontinuité d'ordre 1 de la trajectoire d'un point matériel.*

$$\widetilde{T^X} = \left\{ \begin{array}{l} t_i \mid (X(t), t_i) \text{ une discontinuité} \\ \text{de la trajectoire sur l'axe } Ox \end{array} \right\} \quad (1)$$

On adoptera la même notation pour $\widetilde{T^Y}$ et $\widetilde{T^Z}$ Il correspond à un changement drastique sur l'un des axes X, Y ou Z.

Definition 2. *Un* `action cut` *est défini récursivement de la façon suivante :*

$$AC = \left\{ \begin{array}{lcl} AC_0 & = & 0 \\ AC_j & = & \max\limits_{axe \in \{X,Y,Z\}} \left\{ \min\limits_{\substack{t_i \in \widetilde{T^{axe}} \\ t_i > AC_{j-1}}} \{t_i\} \right\} \end{array} \right. \quad (2)$$

Definition 3. *Un* `action element` *(AE) représente le mouvement pour un unique point du squelette, entre deux* `action cuts` *successifs. Il correspond à une portion de la trajectoire entre deux* `action cut` *pour un intervalle de temps Δt. Ceci nous permet de rapporter toutes les* `action cut` *à la même origine t_0.*

$$AE = \left[P_{deb} \; P_{fin} \; \Delta_t \right] \quad (3)$$

où P_{deb} et P_{fin} sont les positions spatiales de début et de fin de l'`action element`*.*

À l'aide des définitions 2 et 3, nous pouvons représenter les trajectoires de chacune des articulations du squelette d'animation, comme une succession d'`action element`, figure 1. Dans la pratique, chaque projection de trajectoires (selon X, Y et Z) est approximée par une suite de segments de droite.

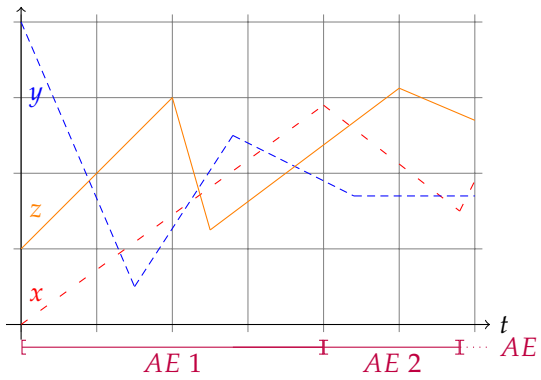

Figure 1: Exemple d'*action elements* définis à partir de régression linéaire des trajectoires d'une articulation de MoCap.

Definition 4. *Une* `Spatio-Temporal Pose` *est l'ensemble des* `action elements` *successifs de toutes les articulations du squelette d'animation obtenus pour un un intervalle de temps* δt.

$$STP = \bigcup_{t\in[t_i,t_j]} \bigcup_{\alpha\in\{A\}} AE_t^\alpha \quad (4)$$

où $\{A\}$ *est l'ensemble des articulations du squelette d'animation,* AE_t^α *est l'*`action element` *relative à l'articulation* α *au temps* t.

3 Métrique Spatio-Temporelle

Nous cherchons à comparer deux `spatio-temporal poses`. Dans notre démarche, nous avons fait le choix de donner plus d'importance à une variation spatiale qu'à une variation temporelle lors de la comparaison d'actions, *i.e.* la vitesse de l'exécution d'une action n'influe que peu sur sa nature. Soit $AE = [P_{deb}\ P_{fin}\ \Delta t]$ et $AE' = [P'_{deb}\ P'_{fin}\ \Delta t']$ deux `action elements`. Nous introduisons le critère suivant, noté ε :

$$\varepsilon(AE,AE') = \left(d\left(P_{deb},P'_{deb}\right) + d\left(P_{fin},P'_{fin}\right)\right) \cdot \left(1 + \frac{|\Delta t - \Delta t'|}{\max\{\Delta t,\Delta t'\}}\right) \quad (5)$$

où $d(P,P')$ est la distance entre deux positions 3D.

En tenant compte de la définition 4, la différence entre deux STP peut être donnée sous la forme suivante :

$$D(STP_1,STP_2) = \sum_{\delta t} \sum_{\alpha\in\{A\}} \varepsilon(AE_1^\alpha,AE_2^\alpha) \quad (6)$$

où $\{A\}$ est l'ensemble des articulations du squelette d'animation, AE_1^α et AE_2^α sont les *action elements* des STP_1 et STP_2 pour l'articulation α.

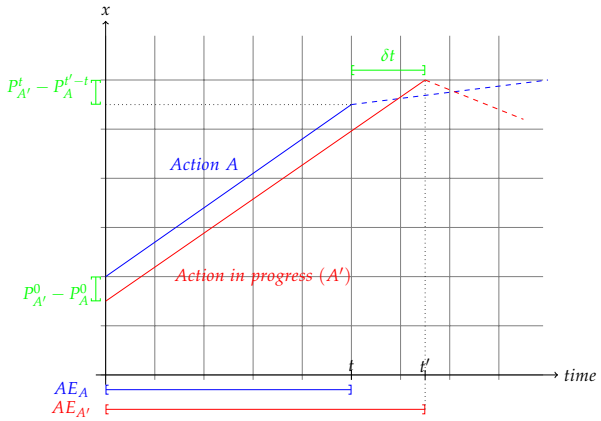

Figure 2: L'action A (apprise) est comparée à l'action A', en cours d'acquisition. La comparaison se fait à l'aide de l'expression 5. Par soucis de clarté, seul la projection sur l'axe x est présentée.

Comme les actions sont décomposées en STP, comparer deux actions revient à comparer ses STP. Ceci peut être fait de façon incrémentale, en particulier avec une stratégie de vote. La ressemblance entre deux actions A_1 et A_2 est donnée par l'expression 7 :

$$C(A_1,A_2) = \sum_{i\in\{STP\}} \left(1 - \frac{D(STP_1^i,STP_2^i)}{T_D}\right) \quad (7)$$

où $\{STP\}$ est l'ensemble des STP, D est défini par l'expression 6. T_D est un facteur de normalisation, qui permet d'avoir $\frac{D(STP_1,STP_2)}{T_D} \in [0,1]$.

4 Automate de reconnaissance

Construction de l'automate : La reconnaissance d'action, par l'intermédiaire des STP, se fait à l'aide d'un automate, qui est défini comme suit. Les nœuds de l'automate sont les *Spatio-Temporal Poses* des actions apprises. Chaque transition est constituée des *Action Elements* conduisant de la STP courante à la STP suivante. Une STP peut appartenir à plus d'une action, dans ce cas, elle aura plus d'une transition dans le graphe. En effet, deux actions peuvent avoir des STP identiques, elles sont alors fusionnées au sein de l'automate.
Nous distinguons différents types d'états dans notre graphe, voir figure 3. Les états initiaux, nommés *Starting STP*, sont créés à partir de la première STP de chaque action. De façon similaire, les dernières STP extraites donnent les états d'acceptation. Elles indiquent la reconnaissance d'une action. Dans certains cas, un état d'acceptation peut être également un état initial. Cela se traduit par une boucle au sein de l'automate. D'autres boucles peuvent se produire

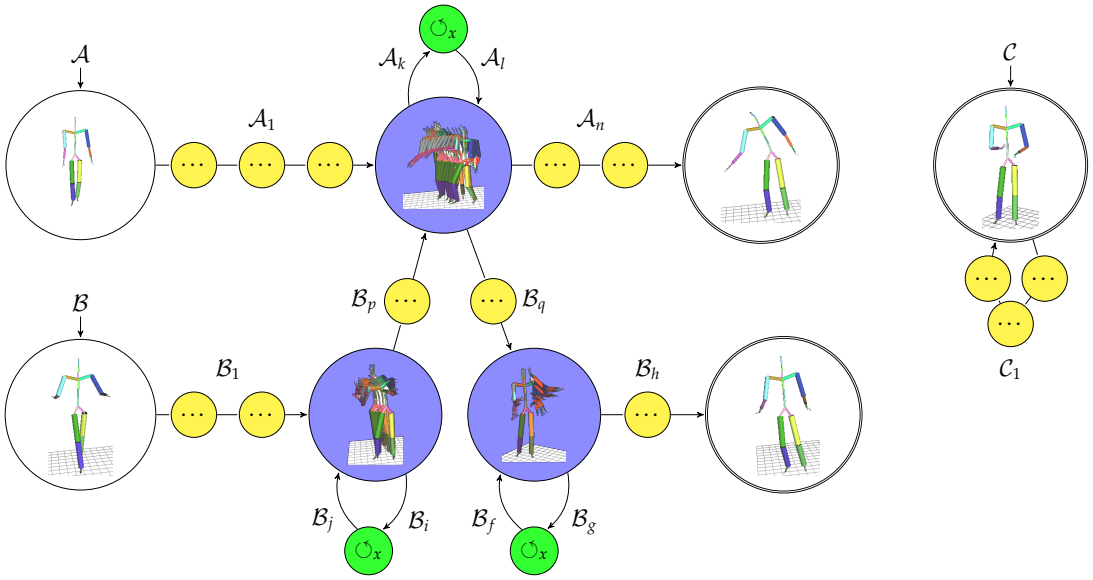

Figure 3: Exemple d'automate pour trois actions apprises : A, B et C. Les *Starting STP* sont représentées par une flèche au dessus, les *Accepting STP* par un double cercle, et les autres STP par les états jaunes.

au sein des actions. Elles sont différentiées car elles servent : *(i)* à identifier les actions répétitives et donc à s'affranchir, si besoin, du nombre de répétitions ; *(ii)* à détecter le bruit, de mouvement ou d'extraction des actions.

Ajout d'une nouvelle action : L'efficacité de notre processus de segmentation des actions en éléments atomiques (`action elements`, ainsi que l'exploitation des STP, nous permet d'ajouter facilement, et à tout moment, une action à notre automate. Une nouvelle action, subit la réduction linéaire décrite en section 2, et est décomposée en STP. Si la première STP de la nouvelle action, est déjà présente dans l'automate, elle devient une *Starting STP* (si elle ne l'était pas), sinon, elle est ajoutée à l'automate. Les STP suivantes sont ajoutées, en évitant les doublons, sous forme d'états intermédiaires.

Reconnaissance d'une action : De façon similaire à l'ajout d'une action, la reconnaissance d'une action est décomposée linéairement (*cf.* section 2). Puis, la première STP est comparée à l'ensemble des états initiaux. En cas de reconnaissance d'une STP initiale, le processus se poursuit le long des transitions de l'automate. Ainsi, reconnaître une action revient à trouver un chemin dans un graphe. Chaque transition est valuée par un score de similarité (équation 5) entre les AE. Ce score permet de quantifier un degré de confiance concernant l'action en cours de reconnaissance.

5 Résultats et discussion

Dans cette section, nous présentons des résultats sur des données synthétiques et réelles.

Ajouter du bruit à une action est un véritable défi. En effet, pour introduire des perturbations dans un mouvement, il n'est pas suffisant d'utiliser un bruit gaussien, mais il faut agir sur les paramètres du mouvement. Dans notre cas, nous avons emprunté des techniques utilisées en animation pour perturber un mouvement. Dans nos expérimentations, nous avons surtout utilisé des perturbations temporelles qui correspondent au mieux à la ré-exécution d'une action par une personne. Nous avons utilisé la méthode développée par McCann *et al.* [24] pour générer nos données bruitées. Cette méthode nous a permis de produire des actions de synthèse qui nous servent de validation (vérité terrain), ainsi qu'à générer des variations sur des cas réels (provenant de MoCap par exemple), afin d'enrichir l'automate.

Discussion

Une comparaison quantitative avec d'autres méthodes est assez difficile, car la plupart les bases d'actions utilisent des données 2D (caméras vidéo).

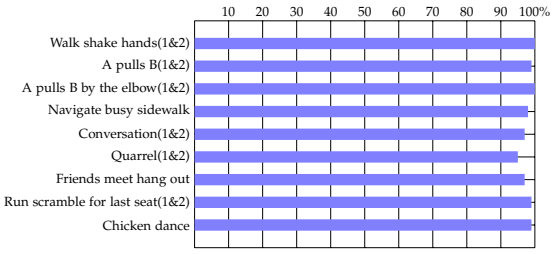

Figure 4: Taux de reconnaissance pour 15 actions synthétiques, générées aléatoirement en utilisant la méthode [24].

#	Motion	# STP	Reduction Rate	Learning Time	Video length
1	Walk, shake hands (1)	87	71.10%	0.002	2.50s
2	Walk, shake hands (2)	87	71.10%	0.101	2.50s
3	A pulls B (1)	87	85.30%	0.168	4.93s
4	A pulls B (2)	89	78.02%	0.946	3.38s
5	A pulls B by the elbow (1)	87	80.13%	0.994	3.65s
6	A pulls B by the elbow (2)	87	78.62%	0.517	3.39s
7	Navigate busy sidewalk	69	76.28%	0.586	2.42s
8	Conversation (1)	87	95.83%	0.647	17.38s
9	Conversation (2)	70	62.50%	0.948	1.93s
10	Quarrel (1)	87	95.18%	1.122	15.04s
11	Quarrel (2)	87	93.06%	1.404	10.44s
12	Friends meet, hang out	87	95.05%	1.471	10.44s
13	Run, scramble for last seat (1)	87	81.33%	1.934	3.88s
14	Run, scramble for last seat (2)	87	73.80%	2.013	2.77s
15	Chicken dance	87	92.96%	2.113	12.8s

Table 1: Résultats obtenus à partir de 15 actions issues de la base [1], où A et B sont les deux acteurs. Les résultats ont été fait sur un Intel® Core™ 2 Quad CPU@2.83GHz, avec 8Gb RAM.

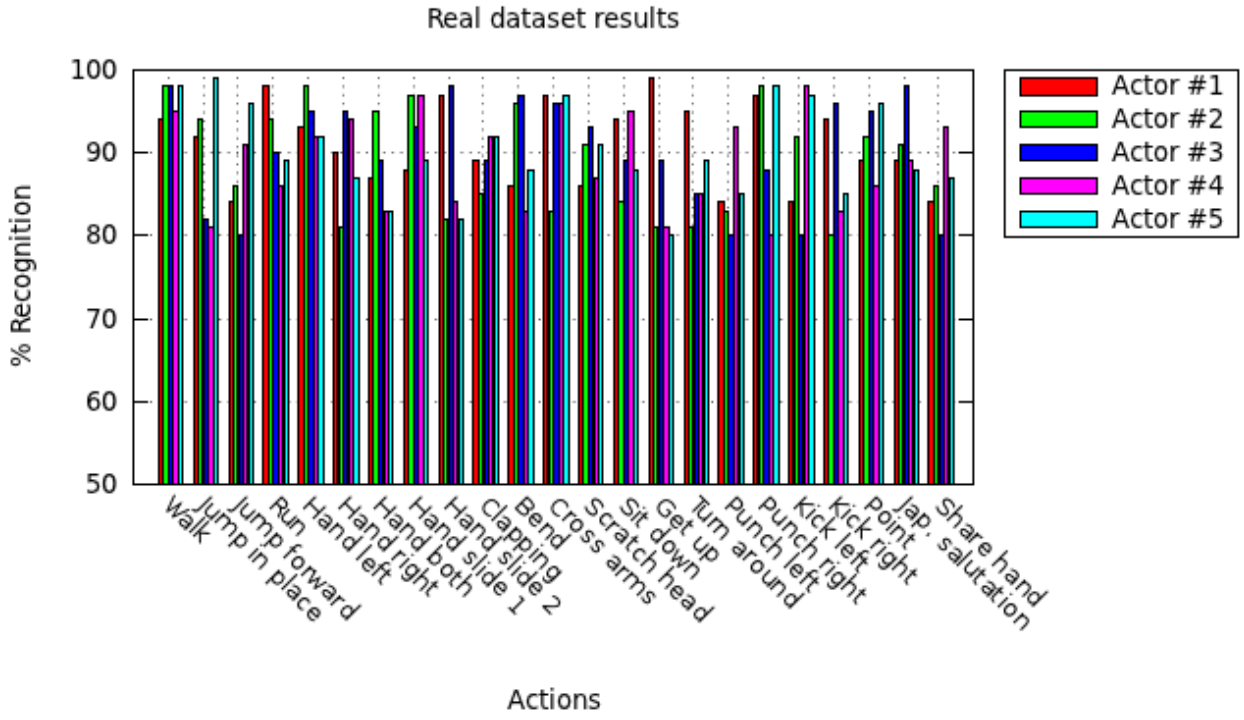

Figure 5: Résultats obtenus à partir d'actions réelles, par la méthode de Shotton *et al.* [23]. Le taux de reconnaissance est présenté entre 50% et 100% pour plus de clarté.

Souvent, ces solutions reposent sur des extractions de points caractéristiques et/ou de squelettes 2D. Très peu utilisent des squelettes 3D (*cf.* section 1). De plus, elles sont généralement basées sur des méthodes d'apprentissage par ordinateur (*machine learning*), qui ne sont pas en mesure de fournir un apprentissage temps réel. Néanmoins, nous pouvons faire un parallèle avec deux méthodes que nous avons évoqué en section 1.

La méthode de Weinland *et al.* [19] utilise des informations 3D pour assurer une invariance au point de vue.Elle requiert un système de caméras calibrées pour l'apprentissage des *examplars* 3D, et se base sur une reconstruction par enveloppe visuelle. Celle-ci présente de bons résultats, mais elle est sensible à l'extraction de silhouettes, et semble assez peu dis-

criminante des actions proches.
La méthode de Okada et Stenger [16], a l'avantage de n'utiliser qu'une seule caméra pour capturer le mouvement. Leur approche est très coûteuse en temps de calcul pour extraire la capture de mouvement de la personne. Elle doit être évaluée sur un calculateur de type Cell^{TM} afin d'assurer une capture temps réel. De plus, cette méthode exige un grand nombre de modèles afin d'estimer des poses pour les différentes morphologies. Soulignons toutefois qu'il ne s'agit pas d'une méthode de reconnaissance d'action, mais d'une méthode de capture de mouvement monocaméra. Ainsi le but n'est pas d'identifier les actions, mais de transférer le mouvement humain sur un avatar virtuel.
Nous proposons une solution capable de surmonter les difficultés des précédentes méthodes par l'utilisation d'un squelette d'animation. Nous pouvons – par cette structure – différencier facilement des actions très proches comme « s'assoir » (*sitting*) de « se plier » (*bend*). Des résultats pour 15 actions synthétiques, caractéristiques des comportements humain, sont présentés dans la figure 4. Chaque action est apprise sans ajout de bruit, puis, 100 variations de cette même action, par la méthode [24], sont générées. Le taux de reconnaissance est toujours supérieur à 90%.
Afin d'évaluer notre méthode avec des données réelles, nous avons utilisé une base de 23 actions, effectuées trois fois par 5 acteurs. La capture de mouvement a été fournie par la méthode [23]. La première représentation de chaque action a été utilisée pour l'apprentissage. Nous avons ensuite utilisé les deux autres répétitions, ainsi que des variations synthétiques des trois actions, afin de fournir des résultats de reconnaissance, voir la figure 5. Au vue de ces résultats, nous sommes en mesure de créer « à la volée » une base d'actions reconnaissables, utilisable pour exploiter une application comme un logiciel de présentation. De plus, comme notre modèle est capable de distinguer des actions proches, nous ne sommes que faiblement impactés par les gestes parasites, comme les moulinets des bras pendant une présentation.

6 Conclusion

Dans cet article, nous avons proposé une méthode permettant d'apprendre et de reconnaitre des actions à partir de squelettes issus de capture de mouvements. Notre système est capable de fonctionner avec l'acquisition 3D sans marqueur de Shotton *et al.* [23], ce qui lui permet d'être adapté au contexte des interactions Homme - Machine. Nous avons présenté deux contributions importantes. (i)Une méthode de réduction des données brutes d'un squelette issue de MoCap en des actions élémentaires : compacts et faciles à calculer. (ii)Une solution temps réel qui utilise un graphe d'animation implémenté sur un automate pour apprendre et reconnaître en ligne des actions. Notre méthode propose une représentation efficace en mémoire et une solution incrémentale qui permet l'apprentissage d'actions à la volée.
Les expériences menées sur des données synthétiques, à partir de la base *CMU Graphics Lab Motion Capture Database*, ont montrées un très haut taux de reconnaissance. L'ajout de bruit nous a permis de valider la robustesse en simulation. Les expériences menées sur des données réelles ont montrées la robustesse de notre méthode dans des conditions usuelles d'interaction Homme - Machine.
Nous projetons d'étendre ses résultats aux actions extrêmement dynamiques : comme le sport ou les jeux vidéo, dans lesquelles l'accélération des mouvements des membres du corps peut être déterminante pour la victoire ou avoir une signification sémantique importante. Nous cherchons par ailleurs à exhiber les éléments intrinsèques à une action : comme les pas de danse dans une chorégraphie. Une telle évolution de la décomposition nous permettrait de coder nos actions en briques élémentaires sémantiques et non seulement numériques. Par ailleurs, notre méthode se base sur un automate déterministe, et une évolution vers un automate stochastique nous permettrait de palier la rigidité actuelle du système.

References

[1] MoCap CMU: The data used in this project was obtained from mocap.cs.cmu.edu. the database was created with funding from nsf eia-0196217. `http://mocap.cs.cmu.edu/`.

[2] Freeman, W.T and Tanaka,K. and Ohta, J. and Kyuma, K.: Computer vision for computer games. Automatic Face and Gesture Recognition, IEEE International Conference on **0** (1996) 100

[3] Kilambi, P. and Masoud, O. and Papanikolopoulos, N.: Crowd Analysis at Mass Transit Sites. In: IEEE International Conference on Intelligent Transportation Systems. (2006) 753–758

[4] Zhang, Zhengyou and Wu, Ying and Shan, Ying and Shafer, Steven: Visual panel: virtual mouse, keyboard and 3D controller with an ordinary piece of paper. In: PUI '01: Proceedings of the 2001 workshop on Perceptive user interfaces, New York, NY, USA, ACM (2001) 1–8

[5] Shanqing Li and Jingjun Lv and Yihua Xu and Yunde Jia: EyeScreen: A Gesture Interface for

Manipulating On-Screen Objects. In: Human-Computer Interaction. HCI Intelligent Multimodal Interaction Environments. (2007) 710–717

[6] Ayers, D., Shah, M.: Monitoring human behavior from video taken in an office environment. Image and Vision Computing **19**(12) (2001) 833–846

[7] Cuntoor, N., Yegnanarayana, B., Chellappa, R.: Activity modeling using event probability sequences. IEEE Trans. Image Processing **17**(4) (April 2008) 594–607

[8] Ivanov, Y.A., Bobick, A.F.: Recognition of Visual Activities and Interactions by Stochastic Parsing. IEEE Transactions on Pattern Analysis and Machine Intelligence **22** (2000) 852–872

[9] Poppe, R.: A survey on vision-based human action recognition. Image and Vision Computing **28**(6) (2010) 976 – 990

[10] Fujiyoshi, H., Lipton, A.J.: Real-time human motion analysis by image skeletonization. Applications of Computer Vision, IEEE Workshop on **0** (1998) 15

[11] Bobick, A.F., Davis, J.W.: The recognition of human movement using temporal templates. IEEE Trans. Pattern Anal. Mach. Intell. **23**(3) (2001) 257–267

[12] Elgammal, A., Shet, V., Yacoob, Y., Davis, L.S.: Learning dynamics for exemplar-based gesture recognition. Computer Vision and Pattern Recognition, IEEE Computer Society Conference on **1** (2003) 571

[13] Xiong, J., Liu, Z.: Human motion recognition based on hidden markov models. In: Advances in Computation and Intelligence. (2007) 464–471

[14] Ahmad, M., Lee, S.W.: Variable silhouette energy image representations for recognizing human actions. Image and Vision Computing **28**(5) (2010) 814 – 824

[15] den Bergh, M.V., Servaes, W., Caenen, G., Roeck, S.D., Gool, L.V.: Perceptive user interface, a generic approach. In: Computer Vision in Human-Computer Interaction. (2005) 60–69

[16] Okada, R., Stenger, B.: A single camera motion capture system for human-computer interaction. IEICE - Trans. Inf. Syst. (2008) 1855–1862

[17] Yilmaz, A., Shah, M.: A differential geometric approach to representing the human actions. Comput. Vis. Image Underst. **109**(3) (2008) 335–351

[18] Huang, K.S., Trivedi, M.M.: 3d shape context based gesture analysis integrated with tracking using omni video array. In: CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Workshops, Washington, DC, USA, IEEE Computer Society (2005) 80

[19] Weinland, D., Boyer, E., Ronfard, R.: Action recognition from arbitrary views using 3d exemplars. In: Proceedings of the International Conference on Computer Vision, Rio de Janeiro, Brazil. (2007) 1–7

[20] Parameswaran, V., Chellappa, R.: View invariants for human action recognition. Computer Vision and Pattern Recognition, IEEE Computer Society Conference on **2** (2003) 613

[21] Han, L., Wu, X., Liang, W., Hou, G., Jia, Y.: Discriminative human action recognition in the learned hierarchical manifold space. Image and Vision Computing (2010) 836 – 849

[22] Andreas Baak and Bodo Rosenhahn and Meinard Müller and Hans-Peter Seidel: Stabilizing Motion Tracking Using Retrieved Motion Priors. In: IEEE 12th International Conference on Computer Vision. (September 2009) 1428–1435

[23] Jamie Shotton and Andrew Fitzgibbon and Mat Cook and Toby Sharp and Mark Finocchio and Richard Moore and Alex Kipman and Andrew Blake: Real-Time Human Pose Recognition in Parts from a Single Depth Image. In: CVPR. (2011)

[24] McCann, J., Pollard, N.S., Srinivasa, S.: Physics-based motion retiming. In: SCA '06: Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation. (2006) 205–214