Reconnaissance d'actions en temps réel à partir d'exemples

Mathieu Barnachon

Dpt Informatique

Université de Lyon

Thèse de Doctorat

Reconnaissance d'actions en temps réel à partir d'exemples

Soutenue publiquement le 22 avril 2013
par Mathieu Barnachon

N° ordre : 65 - 2013

Rapporteur :

M. Edmond Boyer	Directeur de Recherche INRIA Grenoble
M. Thierry Chateau	Professeur des Universités, Université de Clermont-Ferrand

Examinateur :

Mme. Marie-Odile Berger	Chargée de Recherche INRIA, Université de Nancy
M. Boubakeur Boufama	*Full Professor*, Université de Windsor
M. Pascal Estraillier	Professeur des Universités, Université de La Rochelle
Mme. Saïda Bouakaz	Professeur des Universités, Université Lyon 1
M. Erwan Guillou	Maître de Conférence, Université Lyon 1

Remerciements

Je remercie mon jury de thèse qui a été un réel honneur, tout autant qu'un grand plaisir ! Merci à monsieur Pascal Estraillier qui a su le présider parfaitement, merci à messieurs Edmond Boyer et Thierry Chateau qui ont su apporter toute leur expertise afin de rapporter mon manuscrit de thèse, merci à madame Marie-Odile Berger qui a su élargir le débat lors d'une séance de questions particulièrement enrichissante et intéressante. Merci à monsieur Boubakeur Boufama qui n'a cessé de me pousser à aller plus à fond dans mes explications, m'a permis de faire un séjour des plus passionnant à Windsor, et sans qui, mes meilleurs publications ne seraient pas aussi bonnes. Un grand merci aussi à madame Saïda Bouakaz et monsieur Erwan Guillou pour tout ce soutien et cette confiance au cours de ses années de thèse. Vous avez su me pousser, empêcher que je me disperse, m'aider à concrétiser et me permettre d'aboutir ce travail. Je vous en suis infiniment reconnaissant.

Ce travail s'est fait en grande parti dans la bonne humeur, et cela je le dois à tous les membres du laboratoire ! Merci à l'équipe R3AM, grande par le cœur, à Jean-Philippe pour son ouverture d'esprit quand le mien se fermait, à Jean-Claude pour son expertise immense dans tous les domaines et ses solutions élégantes à des problèmes « frustres », à François et ses questionnement qui m'ont permis d'avancer, à Maxime, pour tous les échanges que l'on a eu, sur la science, les maths, mais aussi la rando et la Chartreuse ! enfin et biensur à Guillaume : tout ce que tu m'as appris ne rentrerais pas dans cette thèse ! mais *to name a few* Mercurial & Python ! Je remercie aussi tous les membres de SAARA, Élodie et sa vision des détails qui m'ont contraint à une « relative » perfection dans mon travail, Florence et sa rigueur scientifique, son amitié, et sa fraîcheur d'esprit, Fabrice pour toutes ses questions et remarques, Alexandre, pour les idées que j'aurais du avoir ! et vous tous qui m'avez permis d'aller toujours plus loin, mais de façon plus clair. Merci aussi à tous les habitués du coin café : Samir, Éliane, Romuald, Thierry, Baptiste, Bezhad, Parisa, Stéphanie, Sylvain, ... et les autres, occasionnels ou réguliers.

Deux pensés spéciales : mes co-bureaux ! Sans vous, je n'aurais pas autant ri, et je n'aurais pas autant avancé ! Merci pour l'aide, le support, les idées, les rigolades et le soutiens (et merci pour les pauses quand j'en avais besoin ou pas...).

Un grand merci à mes amis, vous avez su me supporter pendant des années pas

toujours facile, ou j'avais si peu de temps à vous consacrer, mais tellement d'envie ! Nos randonnées resterons comme un moment fort de mon existence.

Enfin, et non des moindres, merci à Pauline. Je ne sais toujours pas comment tu as fait pour me supporter tout ce temps, entre stress et galère, joie et déprime, la vie d'un thésard n'est rien à côté de celui de femme de thésard ! Sans toi, rien de tout cela n'aurait vu le jour, je te dois tout, et tu as toute mon affection.

Villeurbanne, avril 2013

Table des matières

Résumé

Le développement de l'image numérique et des outils associés ces dernières années a entrainé une évolution dans les attentes des utilisateurs et des changements dans leurs habitudes de travail. Cette évolution apporte de nouvelles possibilités d'utilisation ouvrant l'usage à un public très large, allant des interactions gestuelles aux jeux vidéo, en passant par le suivi d'activités à domicile, la surveillance, ... Pour qu'elles puissent être performantes et attractives, ces nouvelles technologies nécessitent la mise en œuvre d'outils de reconnaissance et d'interprétation des gestes humains, par des méthodes efficaces, rapides et ouvertes. Actuellement, les méthodes proposées en reconnaissance d'actions peuvent être regroupées en trois catégories principales : les approches de type apprentissage automatique (*Machine Learning*), les modélisations stochastique ou encore les méthodes utilisant le paradigme des *examplars*. Les travaux développés dans cette thèse se rattachent à cette dernière catégorie : « méthodes à base d'exemples » (*examplar-based*) où l'apprentissage peut être fait à partir de quelques instances représentatives. Nous avons fait le choix d'une démarche qui limite le recours à des grandes bases de données, et qui permet la reconnaissance d'action de façon anticipée, c'est-à-dire avant que cette dernière ne soit finie. Pour ce faire, nos travaux ont été menés selon deux visions complémentaires, avec le soucis constant d'aboutir à des traitements qui soient temps réel, précis et ouverts à la reconnaissance de nouvelles actions.

Dans un premier volet, nous nous sommes orientés vers l'analyse du mouvement des membres, par l'intermédiaire des trajectoires des articulations. À chaque articulation est associée une trajectoire au cours de l'action. L'analyse et la modélisation de ces trajectoires ainsi que l'étude de leurs variations permettent, en particulier, de détecter des changements quantitatifs considérer comme des points caractéristiques de l'action. Suite à cette analyse, la trajectoire d'une articulation donnée est décomposée en une suite de trajectoires élémentaires. Cette partition des trajectoires permet d'obtenir une décomposition de l'action en éléments atomiques qui servent à construire un automate de reconnaissance d'actions. Il s'agit d'un automate déterministe à états finis. Un état de l'automate rassemble les éléments atomiques de l'action de chacune des articulations dans le même intervalle de temps. Le problème de reconnaissance d'actions est ainsi modélisé en un par-

cours d'automate. L'un des avantages est qu'il existe une unité spatio-temporelle des éléments. L'automate donne une organisation sémantique des actions, mais elle présente l'inconvénient d'être très sensible au bruit. Ces bruits peuvent être liés à l'environnement de capture, à l'extraction des caractéristiques ou encore à la variation du mouvement lui même (tremblements, rapidité du mouvement, stabilité du membres,...). Ce travail est exposé en détail dans la partie II.

Barnachon et al. [13, 14].

Pour pallier la sensibilité aux variations locales des trajectoires, la deuxième démarche explorée dans cette thèse s'appuie sur la recherche de séquences invariantes dans une action. Pour cela, nous exploitons les statistiques de ces occurrences. En utilisant une méthodologie proche des sacs de mots (*Bag of Words*), nous avons regroupé l'ensemble des poses par classes, chacune identifiée par un représentant. Sachant qu'une action est suite de poses, elle peut donc être caractérisée par la fréquence de ces représentants. Elle est quantifiée par un histogramme de représentants de poses. Bien qu'étant un outil puissant pour la représentation, l'histogramme supprime la notion de temporalité. Par exemple, l'action « se lever » et l'action « s'assoir » sont représentées de façon identique. Pour tenir compte de la composante temporelle, nous avons étendu le principe des histogrammes intégraux. Ceux-ci sont construits à partir d'une partition temporelle de l'histogramme initial. Cette partition est réalisée compte tenue d'un critère de vraisemblance par rapport à une action donnée. Elle est réalisée par un algorithme de programmation dynamique inspiré de *Dynamic Time Warping*. Cette méthode se montre résistante aux bruits dans les données d'apprentissage et ceci en dépit de la perte de la composante sémantique.

Barnachon et al. [10, 11, 12].

Abstract

In the past years, new advances in numerical images and tools have interested users for these usages. These evolutions have changed user's way of interaction with a computer: from touch to gesture, going through video games and home monitoring, to name a few. To be attractive and efficient, these methods need new methodology for human action recognition and interpretation. For now, these solutions have to be efficient, easy to use and opened to new actions adding. We can highlight three kinds of methods: Machine Learning methods, stochastic model-based methods and examplar paradigm-based methods. The works presented in this thesis are related to the examplar paradigm, where the training is made with few instances. This choice allows to tackle huge database for training, and proposes a solution to recognize ongoing actions. We propose two complementary solutions, with real-time, precision and extensible constraints.

In the first hand, we analyse trajectories of articulations. Each articulation is associated to its trajectory along the action. We study changes in trajectories to propose a model of quantitative changes. These changes are considered as critical points for actions. Trajectories are then decomposed in segments to determine atomic parts of the action. This decomposition is used to create states in an automaton. These states are evaluated during the recognition process to identify action in progress. The recognition process of an action is modeled as an automaton path evaluation. One of the major advantages is that elements are spatio-temporally situated. The automaton gives a semantic organization of actions but it is sensitive to noise. Noise is coming from capture environment, characteristics extractions, or human motion variations.

Barnachon et al. [13, 14].

In order to be insensitive to local trajectories variations, our second main contribution presented in this dissertation is dealing with invariant sequences in actions. We exploit the statistical redundancy of poses. Inspired by the Bag of Word approach, we have formalised delegates of poses, where each pose can be represented by its delegates. The frequency of each delegate is computed, by action, to construct an histogram of poses. Histograms are an efficient tool for representation, nevertheless, it suppresses the temporal component of action. As example,

"Stand-up" action and "Sit-down" action have the same histogram. To deal with temporal relation involved in action, we have extended the integral histograms principle. These integral histograms are made from a partition according to time of the complete histogram. The better partition is made by a dynamical programming algorithm, inspired by Dynamic Time Warping. This solution is robust to noise despite that it lacks semantical representation.

Barnachon et al. [10, 11, 12].

Première partie

Contexte

Chapitre 1

Introduction

1.1 Contexte général

Les nouvelles modalités d'interaction basées sur la vidéo ont suscité de nouveaux besoins auprès des utilisateurs. Des périphériques comme l'*EyeToy* de Sony ou le *Kinect* de Microsoft ont offerts aux utilisateurs la possibilité d'utiliser leurs mains, leur corps, leurs mouvements afin d'interagir avec un ordinateur. Ces nouveaux périphériques demandent de nouvelles approches pour l'interprétation de ces mouvements et leur traduction en ordres. Bien loin du simple domaine de la vidéo surveillance, ces nouvelles applications s'adresse à un public large, et exigent plus de rapidité et plus de souplesse. Il s'agit d'offrir à l'utilisateur la possibilité d'ajouter de nouveaux ordres gestuels, que l'application pourra interprété immédiatement.

L'étude du mouvement, bien que antérieur à la vision par ordinateur, a connu un essors avec l'arrivée des système informatique. En effet, depuis que les caméras numériques sont disponibles, l'ambition de la recherche informatique a été d'utiliser de cette technologie pour l'interprétation automatique des actions. Les premiers travaux dans ce domaine se sont intéressés à la marche. Cet intérêt a été porté par de nombreux contexte applicatif, comme la surveillance dans un aéroport, ou l'analyse de mouvement foule dans un rassemblement, *etc.* De nos jours, certains systèmes sont capables de donner des indications d'identification biométrique. Actuellement le champs de recherche en reconnaissance d'actions s'est élargi à d'autres actions de vie courante (course, saut, *etc.*) Des recueils de séquences vidéos (*datasets*) présentant des acteurs marchant, courant, sautant, *etc.* sont disponibles pour permettre la comparaison des méthodes développées. L'accessibilité des équipements vidéo a suscité de nouvelles demandes de la part des utilisateurs comme reconnaitre des actions, à partir de séquences vidéos quelconques (cinéma, vidéos personnelles, clips, *etc.*), rendant le défi de plus en plus ardu. La

démocratisation de ses outils amène naturellement le recours à la reconnaissance d'action pour l'interaction entre utilisateurs et systèmes.

La reconnaissance d'actions peut intervenir dans différents domaines, englobant les applications industrielles (surveillance, maintenance de site, formation), les applications médicales (maintien à domicile), jeux vidéos, *etc.* Chacun de ses domaines possède ses propres contraintes, parfois incompatibles, souvent liées entre elles. Nous pouvons toutefois souligner trois exigences principales : la précision, la souplesse et la rapidité. Suivant l'application, nous sommes parfois amené à faire un compromis entre ces différentes exigences. Pour certaines applications médicales, par exemple, nous auront tendance à privilégier la précision, alors que dans une application ludique (loisir numérique) nous chercherons en priorité la rapidité et l'adaptabilité. Chaque solution doit trouver son propre équilibre entre ses contraintes, en fonction de son contexte d'application. Les trois exigences indispensables en reconnaissance d'actions sont schématisées par la figure 1.1.

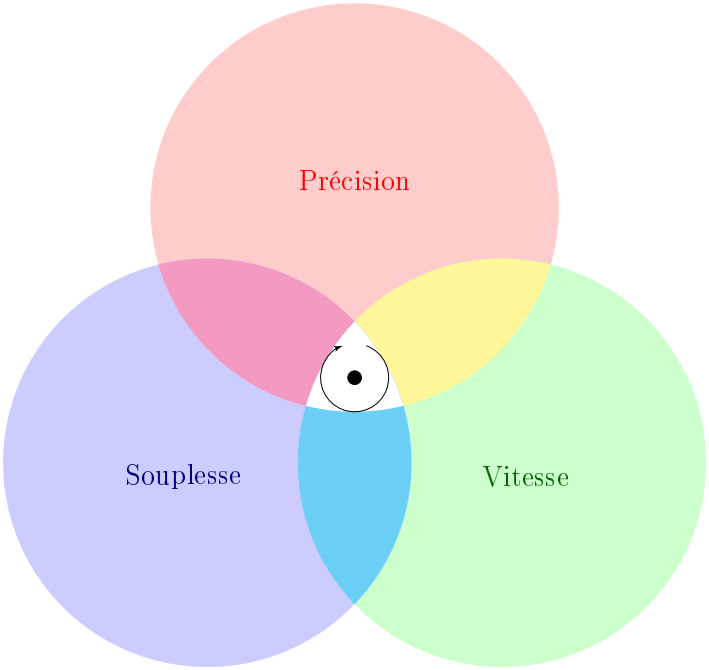


FIGURE 1.1 – Représentation schématique des objectifs à atteindre en reconnaissance d'action. Les intersections indiquent quelles exigences ont été privilégiées.

1.2 Contexte technique

1.2.1 Contraintes

Aux contraintes de la mise en place de méthodologies souples et rapides pour la reconnaissance, d'autres problèmes se posent. Ces problèmes sont dus, d'une part, à la nature des données en entrée, qui sont souvent tâchées de bruit. Et à la difficulté d'extraction des informations de postures de façon fiable et constante au cours du temps, d'autre part. De plus, les êtres humains sont susceptibles de faire de grandes variations de style dans la reproduction d'un même geste. Cela est induit par des vitesses, des amplitudes et des répétitions différentes lors de l'exécution. Ces variations seront plus ou moins importantes selon le contexte et le cadre dans lesquels l'action se déroule. En effet, un jeu vidéo ou une application biométrique n'auront pas les mêmes contraintes d'environnement. Si pour le deuxième, on peut utiliser un cadre contrôlé, pour le premier, souvent dans le salon du joueur, on est soumis à plus de variabilités. Ces difficultés intrinsèque (bruits) et extrinsèque (contexte) ont de fortes répercutions dans l'analyse du problème de la reconnaissance d'actions. Il faut prendre en compte les problèmes de fiabilité dans l'acquisition des données de mouvements par un raffinement des primitives en entrée ou une extraction des points d'intérêts du geste. La grande variabilité dans la reproduction des gestes par un être humain, la précision « requise » en fonction du contexte (jeux vidéo *vs* interaction Homme–Machine) doivent être traitées par une méthodologie plus dépendante du contexte. Cela ne doit pas, le plus souvent, se faire au détriment du temps de reconnaissance et du temps d'apprentissage, si notre méthode est suffisamment rapide pour apprendre des actions « à la volée ».

1.3 Plan

Dans la suite de cette thèse, nous présenterons un état de l'art en reconnaissance d'actions, ainsi qu'une taxonomie adaptée à notre contexte applicatif. Pour proposer des solutions répondant à nos contraintes, nous avons divisé ce document en deux parties. La première partie présentera une solution à la création de primitives de haut niveau, directement à partir des trajectoires des membres d'un corps. Pour cela, nous avons utilisé un changement de base, de même qu'une réduction des dimensions du squelette. Ceci nous permet de construire une structure de type automate, adaptée à l'identification d'un mouvement appris. De plus, cette solution offre une grande souplesse dans l'ajout « à la volée » d'une nouvelle action.

La deuxième partie de cette thèse se concentrera sur la reconnaissance d'actions à partir de poses clairement identifiées. En effet, pour être moins tributaire

du bruit intrinsèque d'une action, nous avons fait le choix d'une approche plus statistique. Les poses importantes sont automatiquement identifiées, et une construction de type histogramme nous permet d'avoir une vision d'ensemble de l'action. Or, comme l'ont montrés de nombreux travaux, la temporalité d'une action est importante. Afin de ne pas la supprimer entièrement, nous avons fait le choix d'une approche reposant sur la programmation dynamique. Ainsi, nous sommes en mesure de garder une granularité temporelle dans notre modèle d'actions. Les résultats obtenus sont encourageants, et permettront, espérons le, une extension de cette solution à des primitives de plus bas niveaux, extraites directement des images par exemple, et non plus tridimensionnelles comme c'est le cas actuellement.

Nos deux contributions reposent sur le paradigme des *examplars*. Cela permet une mise en application quasi immédiate, même lorsque nous disposons de peu de données d'apprentissage. Notons de plus, que toutes ses approches sont temps réel et peuvent être aisément ajoutées à n'importe quels systèmes d'acquisition de poses, comme nous le montrons dans les différents chapitres de résultats.

Le chapitre 3 se veut un rappel sur la capture de mouvement 3D — utilisée tout au long de ce travail — afin de faciliter la compréhension du lecteur et de s'accorder sur les terminologies dans ce domaine. De même l'annexe A présente, de façon plus complète, les différentes données (*datasets*) utilisées pour présenter nos travaux.

Chapitre 2

État de l'art

Sommaire

2.1 Reconnaissance d'actions : le domaine

La reconnaissance d'actions est un sujet étudié depuis de nombreuses années dans la communauté de vision par ordinateur. Cet intérêt est traduit par de nombreuses études bibliographiques proposées aux cours des années Aggarwal et Ryoo [1], Poppe [103], Turaga et al. [131], Weinland et al. [147]. Pour des études relatant l'évolution de ces travaux, il est possible de consulter les références suivantes : Turaga et al. [131] en 2008, Poppe [103] en 2010 qui s'est intéressé aux défis de la reconnaissance d'actions à partir d'images ou de vidéo ; ou Weinland et al. [147] en 2011 qui se sont focalisés sur la représentation des actions ; Aggarwal et Ryoo [1] en 2011 qui ont proposés une approche plus complète du domaine de la reconnaissance d'actions. La volonté de savoir ce que fait un être humain, à un moment donné est présente dans de nombreux domaines : industriel, médical, surveillance, aide à la personne, domestique, *etc.* Les travaux proposés en reconnaissance d'actions se sont intéressés à tous les aspects de ce problème.

Bien que tous les domaines de la reconnaissance aient été étudiés, les différents auteurs ne se sont pas toujours accordés sur une taxonomie générale et complète du domaine. Chaque étude bibliographique (*survey*) présente sa propre taxonomie.

Néanmoins, toutes ou presque font des distinctions entre les actions en fonction de leurs complexités intrinsèques. Ainsi, la décomposition communément admise est la suivante : un « *geste* », une « *action* », une « *interaction* » et une « *activité de groupe* », Aggarwal et Ryoo [1]. Nous avons choisi d'utiliser la même approche pour présenter l'état de l'art, tout en rappelant les définitions.

Un « *geste* » est défini comme le mouvement élémentaire des parties du corps d'une personne. Il s'agit de la décomposition sémantique la plus élémentaire, c'est-à-dire la plus proche du mouvement perçu, comme « agiter le bras », « lever une jambe », *etc.* Une « *action* » est l'activité d'une seule personne qui peut être composée de plusieurs gestes, arrangés temporellement, comme « marcher », « boxer », *etc.* Une « *interaction* » est une activité humaine qui implique soit deux humains au moins agissant ensemble, éventuellement avec un ou des objets, soit un seul humain mais agissant avec au moins un objet. Des exemples d'interaction seraient : « deux personnes se poussant » ou « une personne jetant un objet à la poubelle ». Enfin la dernière distinction concerne les « *activités de groupe* ». La distinction entre cette catégorie et la précédente est faible. Elle tient en la présence de nombreux participants intervenant dans l'action. Néanmoins, telle qu'elle est souvent présentée, elle est définie par un nombre — important — de personnes agissant ensemble avec éventuellement des objets. Une « réunion de groupe » ou « une foule défilant » en sont des exemples.

À partir des définitions suivantes, nous plaçons nos travaux dans le cadre de la reconnaissance d'actions. C'est-à-dire que nous focalisons la reconnaissance sur chaque individu indépendamment de leurs interactions éventuelles.

Les jeux de données proposés pour l'évaluation des méthodes de reconnaissance sont souvent liés à un contexte particulier. En partant des données les plus simples, jusqu'à celles qui posent de nouveaux problèmes en interaction, nous allons les présenter dans le contexte de leurs introductions, en soulignant les défis associés à chacun d'eux.

De nombreux jeux de données ont été créés afin d'identifier la marche, la course et les sauts. Ces gestes ont été considérés comme la première étape nécessaire avant de travailler sur des actions plus complexes. Citons particulièrement les jeux KTH (proposé par Schuldt et al. [116] (2004)) et Weizmann (proposé par Blank et al. [23] (2005)), qui sont composés de 6 et 10 actions respectivement, effectuées par respectivement 25 et 9 acteurs différents. Comme ces données sont les plus anciennes, les résultats obtenus en classification sont proches des 100% de précision. Fathi et Mori [38], Shao et Chen [119], Tran et Sorokin [128], Wang et Mori [142], Yeffet et Wolf [157] ont obtenu un score de classification de 100% sur le jeu de données Weizmann ; Cao et al. [26] et Gilbert et al. [44] ont obtenu plus de 95% de bonne classification sur le jeu de données KTH, Tran et al. [129] ont obtenu $97,83\%$ sur

le jeu de données KTH. Les méthodes vont sans doute bientôt atteindre une classification parfaite des actions sur ce jeu de données (*i.e.* 100% de reconnaissance). Cela tient à la fois à l'amélioration des méthodes de reconnaissance et des avancées faites aux cours des dernières années, mais aussi à la relative simplicité des données, qui présentent des actions très différentes, dans des conditions d'acquisitions assez peu contraignantes.

Le besoin de proposer des jeux de données plus complexes pour évaluer le potentiel des méthodes de reconnaissance a conduit à proposer de nouvelles données pour l'évaluation des méthodes plus récentes. Ainsi, de très nombreux autres jeux de données issus de vidéo déjà accessibles, et donc d'actions acquises dans des conditions réelles, ont été proposés, tels que :

- *The University of central Florida Sport Dataset (UCF Sport)* [107] en 2008, présentant des actions de sports télévisés, comme le golf, la course, ou le plongeon ;
- *The University of Central Florida Youtube Dataset (UCF YouTube)* [73] en 2009, présentant des vidéo issues du site de partage YouTube™comme des enfants faisant du vélo, de la balançoire, *etc.* ;
- *The Hollywood Dataset 1* [67] en 2008 et *The Hollywood Dataset 2* [79] en 2009 présentant des actions issues de films, comme répondre au téléphone, s'asseoir, *etc.* tels qu'ils ont été filmés pour le cinéma, c'est-à-dire présentant un cadrage et une lumière variable, avec une approche artistique plutôt que scientifique de l'acquisition des actions ;
- *Human Motion DataBase (HMDB)* [63] en 2011 et *TV Human Interactions* [97] en 2010 pouvant être vus comme des continuités de ces jeux aux séries télévisées et aux clips YouTube™de façon plus générales ;
- *The Senior Home Monitoring* [28] en 2011, présentant des personnes âgés effectuant des actions de la vie courante : manger, boire, regarder la télévision, *etc.* dans leur propre chambre ;
- *INRIA Xmas Motion Acquisition Sequences* [146] en 2006, présentant 13 actions de la vie quotidienne effectuée par 11 acteurs, chacune 3 fois. Ce jeux de données présente la particularité d'avoir été acquis par 5 caméras synchronisées, permettant d'avoir de multiples vues, mais aussi des reconstructions 3D volumiques et surfaciques.

Ces nouveaux jeux de données ont apporté plus de diversité dans les actions, ils restent néanmoins limités en terme de classes d'actions — entre 4 et 13 catégories différentes [1] — et sont donc des « restrictions » plus ou moins larges des actions à la portée d'un être humain. Leurs limitations en terme de classes d'actions conduit les méthodes de reconnaissance à une « spécialisation » des algorithmes sur chacun de ces jeux de données, au détriment de la généricité de reconnaissance des actions

1. Sauf pour Kuehne et al. [63] qui présente 51 classes d'actions

humaines.

Afin d'aller plus loin dans la reconnaissance d'actions, il semble nécessaire d'avoir des jeux de données avec de nombreuses classes, représentant les multiples variations possibles par les êtres humains. Toutefois, le problème de reconnaitre ces actions parait difficile sans une utilisation de primitives de plus haut niveau que celles basées directement sur les images. À cette fin, des jeux d'actions nouveaux pour la reconnaissance ont été créés ou détournés de leur utilisation première. Citons le *TUM Kitchen Dataset* proposé par Tenorth et al. [126] en 2009. Il s'agit d'un jeu de données représentant la mise en place d'une table pour un repas, dans une cuisine instrumentée par différents capteurs. Quatre caméras ont permis de fournir à la fois des images 2D suivant plusieurs vues de l'activité et une capture de mouvements précise des acteurs. Divers scénarii sont proposés, comme ne transporter qu'un seul ustensile de cuisine à la fois, plusieurs couverts à la fois, *etc.* Des puces RFID permettent de savoir quand il y a contact avec certains objets, telles les portes de placard, et une annotation, en 10 classes, de tous les scénarii est fournie, pour chacune des mains de l'utilisateur (droite et gauche) et pour le torse. Dès lors, une utilisation de ce jeu de données, à des fins de reconnaissance d'actions, plutôt que de *benchmark* des algorithmes de capture de mouvements est tout indiquée. Un autre jeu de données est notablement intéressant et récent, le *MSR Daily Activity 3D* proposé par Wang et al. [138] (2012). Il s'agit d'un ensemble d'acquisition d'actions dans un environnement d'intérieur, le salon, dans lequel des acteurs font 16 actions différentes : « boire », « manger », « téléphoner », « lire un livre », *etc.* Ce jeu possède l'avantage de présenter une acquisition vidéo couleur — caméra RGB classique — en plus de l'acquisition de la profondeur — caméra infrarouge et lumière structurée (Kinect$^{\text{TM}}$) — et de fournir une extraction du squelette d'animation — par la méthode de Shotton et al. [122] — à partir des données de profondeur. Le contexte un peu difficile et la lumière non contrôlée vont faire de ce jeu une référence pour la reconnaissance d'actions dans les années à venir, car il propose des actions complexes, mais aussi des primitives de haut niveau (le squelette simplifié du personnage). Ainsi, il est possible de comparer les solutions utilisant des primitives vidéo, celles utilisant des primitives basées sur la profondeur et les méthodes 3D basées sur le squelette. De plus, il est possible d'exploiter les informations 2D et 3D de façon conjointes, afin d'améliorer la reconnaissance notamment.

Pour une analyse plus complète des jeux de données 2D, et des performances des méthodes au cours du temps, on peut se reporter à Moeslund et al. [85, chap. 20]. Le tableau 2.1 présente un récapitulatif des différents jeux en fonction de leur année de soumission et du nombre d'actions impliquant chacun d'eux. La figure 2.1 synthétise la complexité relative des jeux de données les uns par rapport aux autres. Dans notre cas, nous avons utilisé les données 3D, dont le *TUM Kitchen Dataset*

Jeu	Année	2D/3D	Classes	Clips
Schuldt et al. [116]	2004	2D	6	100
Blank et al. [23]	2005	2D	9	9
Weinland et al. [146]	2006	3D	13	33
Laptev et al. [67]	2008	2D	8	30–129
Rodriguez et al. [107]	2008	2D	9	14–35
Marszałek et al. [79]	2009	2D	12	61–278
Liu et al. [73]	2009	2D	11	100
Tenorth et al. [126]	2009	2D + 3D	10	20 séq.
Patron-perez et al. [97]	2010	2D	4	75
Cheng et al. [28]	2011	2D	12	$\geq$ 10 jours
Kuehne et al. [63]	2011	2D	51	$\geq$ 101
Wang et al. [138]	2012	2D + 3D + Z	16	20

TABLE 2.1 – Récapitulatif des jeux de données couramment utilisés en reconnaissance d'actions.

et le *MSR Daily Activity 3D*, ainsi que des jeux de données issus du monde de l'animation comme le *CMU MoCap Dataset* ou le *HDM Dataset*. Ces données sont présentées dans l'annexe A.

Nous avons divisé l'état de l'art de la reconnaissance d'actions en trois parties distinctes, suivant les objectifs et méthodologies des auteurs. Ainsi, nous distinguerons les parties suivantes : l'identification des parties du corps afin de reconnaitre des actions (2.3), les travaux issus du monde de l'animation s'intéressant à l'extraction de comportements dans de grandes bases de données d'animations (2.4), et la reconnaissance d'actions dans un objectif d'interactions Homme–Machine (2.5). Chacune des catégories sera décomposée suivant la taxonomie de la figure 2.2, faisant le lien entre but et méthodologie. La reconnaissance d'actions n'étant pas basée uniquement sur l'extraction de primitives, mais aussi sur une état de classification de celle-ci, nous présenterons en section 2.6 les différentes stratégies de classifications utilisées dans la littérature.

2.2 Méthodes basées points d'intérêt

Les premiers travaux en reconnaissance d'actions humaines, à partir de caractéristiques spatiales locales, se sont attachés à utiliser les primitives connues en reconnaissance d'objets. Cette analyse repose sur des informations locales, c'est-à-dire à un ensemble de pixels présentant une singularité, que ce soit au niveau du gradient ou du contour. Ces points d'intérêts sont donc les caractéristiques spatiales 2D présentent dans les images. Ces primitives sont invariantes à la taille et

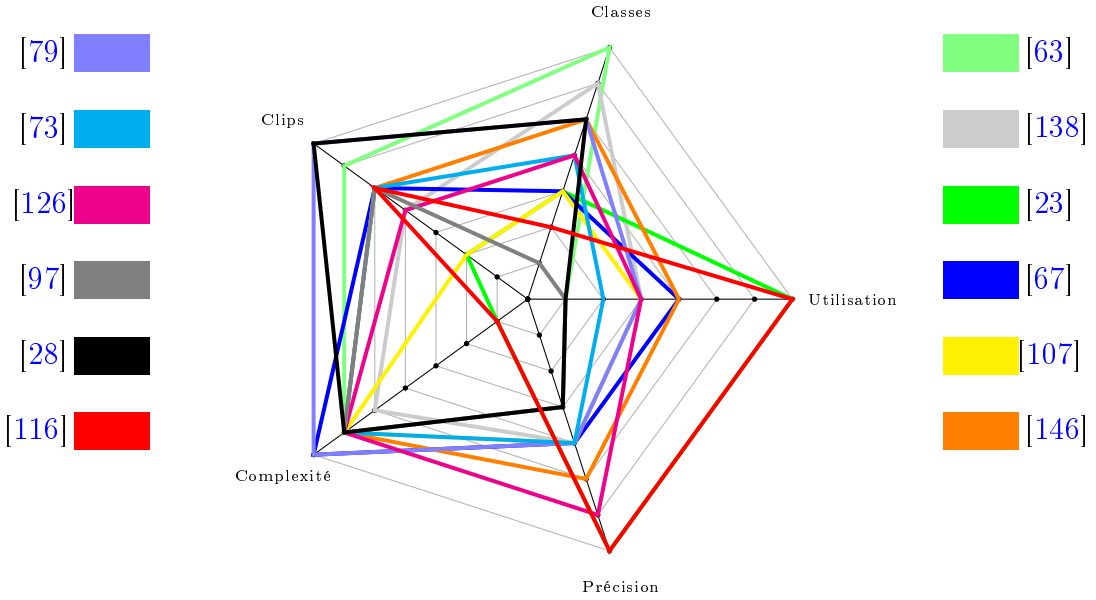


FIGURE 2.1 – Représentation en diagramme araignée des jeux de données courants. (Il s'agit d'une représentation qualitative.)

à l'orientation, elles fournissent un bon moyen d'identifier des objets dans une image. Les recherches se sont alors orientées vers une utilisation de ces primitives afin d'identifier des comportements humains. Pour cela, les auteurs ont appliqué ces descripteurs aux vidéo d'actions pour extraire des informations discriminantes sur l'apparence des comportements humains. La plupart des méthodes de reconnaissance d'actions, à partir de points caractéristiques, se sont construites sur l'analyse du volume spatio-temporel. Le volume spatio-temporel est une extrapolation temporelle de l'information située autour d'un point caractéristique 2D. Le postulat est que ce volume est un objet 3D rigide, qu'il est possible de reconnaitre en utilisant les singularités à sa surface. Ainsi, la reconnaissance est transposée en un problème d'appariement de points 3D. Trois méthodologies concurrentes ont été proposées : la distribution spatiale de points d'intérêt, l'extraction de caractéristiques éparses et l'utilisation de la connectivité temporelle des caractéristiques extraites.

2.2.1 Points d'intérêt spatiaux

Nous pouvons citer les travaux de Chomat et Crowley [30] qui ont utilisé conjointement des filtres de Gabor et des champs d'énergie pour reconnaitre des gestes. Pour cela, ils détectent des caractéristiques locales dans chacune des images qu'ils utilisent pour construire des histogrammes multidimensionnels. Ces histogrammes servent de données d'entrées à un système bayésien afin de calculer la probabilité,

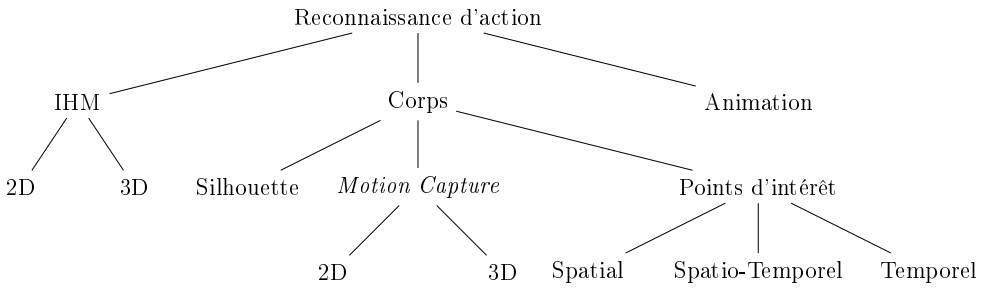


FIGURE 2.2 – Notre taxonomie de l'état de l'art en reconnaissance d'actions. Nous avons utiliser cette hierarchie pour présenter l'état de l'art en reconnaissance d'actions. Trois grandes catégories ont été distinguées : les interfaces Homme–Machine, l'analyse du personnage et l'animation de personnage. Ces catégories sont ensuite divisées en fonction des méthodologies mise en œuvre.

pour chaque pixel, d'appartenir à une action donnée. Ensuite, ces probabilités sont intégrées sur l'ensemble des images de la séquence vidéo pour fournir un score de similarité entre la vidéo et chacune des actions apprises. Ce travail a permis de montrer la validité des approches locales dans la reconnaissance d'actions, même s'il ne s'agissait que de gestes simples. Zelnik-Manor et Irani [159] ont étendu le principe précédent en ajoutant une structure multi-échelle au niveau temporel. Ainsi, leur solution est plus robuste aux différences de vitesses dans l'exécution d'une action, d'un acteur à un autre. Ils ajoutent l'indépendance spatiale des caractéristiques locales, en ne tenant plus compte de leur position dans l'image. Cela se traduit par une meilleure résistance au point de vue. Un algorithme de classification non supervisé (*Modified N-cut*) est employé pour reconnaitre des actions comme le tennis ou le baseball. Blank et al. [23] ont utilisé une extraction de la forme comme la saillance et l'orientation d'un pixel par rapport à son voisinage. Cette extraction est faite en résolvant une équation de Poisson pour chaque pixel du volume spatio-temporel. Dès lors, une classification par plus proches voisins (*kNN*) leur permet de reconnaitre les actions du jeu de données Weizmann, proposé à cette occasion. Efros et al. [35] ont proposé une utilisation du flux optique, dans un contexte bruité ou de faible résolution. Leurs acteurs font environ 30 pixels de haut, et sont donc difficiles à discriminer. Ils proposent une analyse du « bruit » en mouvement plutôt que du flux optique classique. Ainsi, il est plus facile de suivre une personne, notamment dans la diffusion d'un match de foot en plan large. Des descripteurs issus directement du flux optique sont utilisés comme caractéristique du mouvement, et une requête à une base de données de mouvements manuellement annotée permet de reconnaitre l'action. Jhuang et al. [53] ont proposé une utilisation des filtres de Gabor dans un modèle inspiré du comportement biologique humain. Ainsi, ils construisent un modèle multicouches,

composé d'extraction spatio-temporel de motifs par les filtres de Gabor. Grâce à une classification par SVM, ils obtiennent de bons résultats sur les jeux de données constitués uniquement d'une personne, comme Weizmann et KTH. Klaeser et al. [58] ont utilisé un calcul du gradient 3D, *i.e.* dans le volume spatio-temporel. Il s'agit de caractéristiques denses, extraites à partir d'une généralisation de la vidéo intégrale. La construction d'un vocabulaire visuel par quantification des gradients permet d'utiliser un SVM pour la reconnaissance. Willems et al. [149] ont étendu le principe des *examplar* aux caractéristiques denses. En appliquant le principe des sacs de mots, ils sont en mesure de fournir une estimation de la boîte englobante de l'action en plus de sa reconnaissance. Patron-perez et al. [97] ont utilisé des Histogrammes de Gradient Orientés (HOG) afin de détecter des interactions entre personnes dans des vidéo issues de séries télévisées. Bien qu'ils effectuent une détection du corps, par la méthode de Ferrari et al. [40] afin de calculer leurs caractéristiques uniquement sur des parties étiquetées comme appartenant à un personnage, il s'agit d'une méthode dense reposant sur du flux optique et des SVM pour discriminer les interactions entre individus.

Ces approches locales fournissent de bons résultats, mais sont tributaires de primitives parfois instables, parfois connectées de manière artificielle ou inexacte. Il n'est pas évident que tous les pixels d'une image soient pertinents dans la reconnaissance d'une action.

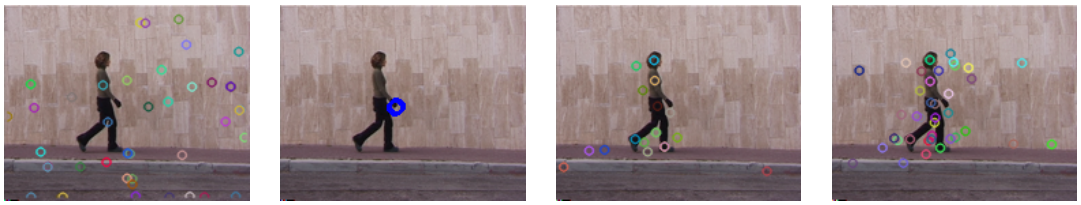
2.2.2 Points d'intérêt éparses

Les approches éparses (*sparse*) offrent une solution plus déconnectée de la réalité de l'action mais plus stable et précise. Le postulat initial repose sur une extraction de caractéristiques lorsqu'elles sont le plus « prononcées », par opposition à une extraction systématique. Ces travaux ont été motivés par le succès de cette approche pour la reconnaissance d'objets, notamment celui du détecteur SIFT par Lowe [75]. Laptev et Lindeberg [66] ont proposé une extension du détecteur de Harris et Stephens [48], utilisé pour la reconnaissance d'objets, à la reconnaissance d'actions. Ils extraient des caractéristiques invariantes à l'échelle, à partir des coins tridimensionnels du volume spatio-temporel. Ce détecteur, communément appelé Harris 3D, leur permet d'extraire des motifs de mouvements (*Motion Pattern*). Schuldt et al. [117] ont utilisé ce détecteur avec un SVM, montrant sa validité dans la reconnaissance d'actions par l'introduction du jeu de données KTH. Dollar et al. [34] ont proposé un détecteur pour les mouvements périodiques. Ils ont ainsi introduit le concept de « cuboïd », nom donné au volume spatio-temporel extrait autour d'un point caractéristique. Il s'agit de capturer l'information portée par un pixel dans l'espace spatio-temporel. Ainsi, un « cuboïd » est composé des pixels voisins du pixel d'intérêt extraient au cours du temps. Le volume est composé des pixels voisins et de leurs valeurs, dans une fenêtre temporelle autour de la première

extraction, avant et après. Une modélisation de l'occurrence des « cuboïds » par histogrammes, à l'aide de l'algorithme *K-Means*, a permis de valider l'approche du « sac de mots » (*Bag of Words*), proposée par Csurka et al. [31], à la reconnaissance d'actions. Dans le même esprit, Niebles et al. [92] ont proposé une utilisation du détecteur de Dollar et al. [34] à l'aide d'un apprentissage probabiliste, le *probabilistic Latent Semantic Analysis (pLSA)*, très employé en reconnaissance de textes. Oikonomopoulos et al. [94] utilisent une extraction des points saillants, proposés par Kadir et Brady [55], dans le volume spatio-temporel. Ils ont adaptés la distance de Chanfrein à cet espace, et utilisent l'alignement par *Linear Space Time Warping* pour aligner leurs séquences vidéo. Leur classification et reconnaissance d'actions s'effectuent par une Machine à Vecteurs Significatifs (*Relevant Vector Machine (RVM)*, proche d'un SVM sur le principe de fonctionnement. Enfin, citons Scovanner et al. [118] qui ont créés une version 3D de l'algorithme SIFT de Lowe [75] pour exploiter le volume spatio-temporel. Leur détecteur a les mêmes propriétés que SIFT en 2D, c'est-à-dire invariance à l'échelle et aux transformations, et permet d'obtenir des points caractéristiques robustes dans le volume spatio-temporel.

2.2.3 Points d'intérêt spatio-temporels

Les solutions précédentes n'utilisent pas de relations spatio-temporelles entre les caractéristiques extraites. Afin de pallier ce manque, Wong et al. [151] ont étendu pLSA en ajoutant une information spatio-temporelle sur les points caractéristiques extraits par rapport au centre de l'action. Les résultats en reconnaissance d'action ont pu être améliorés par rapport à ceux obtenus par Niebles et al. [92] sur le jeu de données KTH. Liu et Shah [74] et Laptev et al. [67] ont proposé de façon similaire la construction de grilles de corrélations entre leurs points caractéristiques temporels et la position spatiale de ceux-ci afin de prendre en compte les deux composantes. Ryoo et Aggarwal [112] introduisent les « appariements relationnels spatio-temporels » (*Spatio-Temporal Relationship Match* ou *STR match*). La structure spatio-temporelle permet une mise en correspondance temporelle des caractéristiques, même au sein de structurations complexes. Niebles et al. [91] ont proposé une approche générique d'exploitation des relations spatio-temporelles dans la reconnaissance d'actions. En se basant sur des points caractéristiques (Harris 3D) et une décomposition en segments temporels, ils effectuent une correspondance segments à segments en pénalisant les recouvrements entre segments. Il s'agit d'une approche globale, qui repose néanmoins sur les performances des approches de détection de points caractéristiques utilisées. Ryoo [111] a proposé une solution utilisant les sacs de mots et des histogrammes intégraux (*Integral Histograms*), pour reconnaitre des actions pendant leurs exécutions. Il s'agit d'une prédiction à proprement parler, de l'action en cours de déroulement. Ceci revêt une importance capitale dans les contextes de surveillance



(a) MSER (b) Harris (c) SIFT (d) SURF

FIGURE 2.3 – Différents type de points d'intérêt.

ou d'assistance aux personnes en danger notamment. L'avantage de la méthode par histogramme intégral est de permettre une reconnaissance précoce de l'action, sans surcout à la reconnaissance. Néanmoins, l'utilisation sur des séquences supérieures à la minute, ou dans le cadre des interactions Homme–Machine auraient été un plus pour la validation de la méthode.

Les méthodes précédentes ne fournissent bien souvent aucune identification de l'acteur de l'action. Les points caractéristiques utilisés « s'accrochent » indifféremment sur l'environnement ou sur le personnage (voir la figure 2.3 pour un exemple, notamment les figures (a), (c) et (d)). Il est difficile de savoir avec les jeux de données utilisés si ces méthodes ont de bons résultats grâce à leurs conceptions ou grâce à une relative variation au sein des arrière-plans entre les actions. Ceci a conduit la recherche à s'intéresser à l'acteur. Les méthodes présentées par la suite fonctionnent sur une détection des personnes dans la scène, pour concentrer l'analyse sur les personnages.

2.3 Méthodes basées apparence

Lorsque l'on cherche à identifier des actions humaines, il est parfois judicieux de savoir où se trouve l'acteur de l'action afin de l'isoler du reste de la scène. Ceci permet de se focaliser sur ses mouvements, indépendamment de ce qui peut se passer à l'arrière-plan. Pour ce faire, deux pistes ont été largement étudiées : l'analyse de la forme du personnage, par le biais de sa silhouette et l'analyse du mouvement de l'acteur par le biais d'une identification de ses membres (mains, tête, jambes, *etc.*). Nous présentons dans la suite un tour d'horizon de ces méthodes.

2.3.1 Analyse de la silhouette

Dans les analyses de la forme par des silhouettes, la modélisation de l'acteur peut être simple : la séparation entre le premier plan et l'arrière plan permet de considérer le premier plan comme un acteur. Une modélisation plus complexe peut

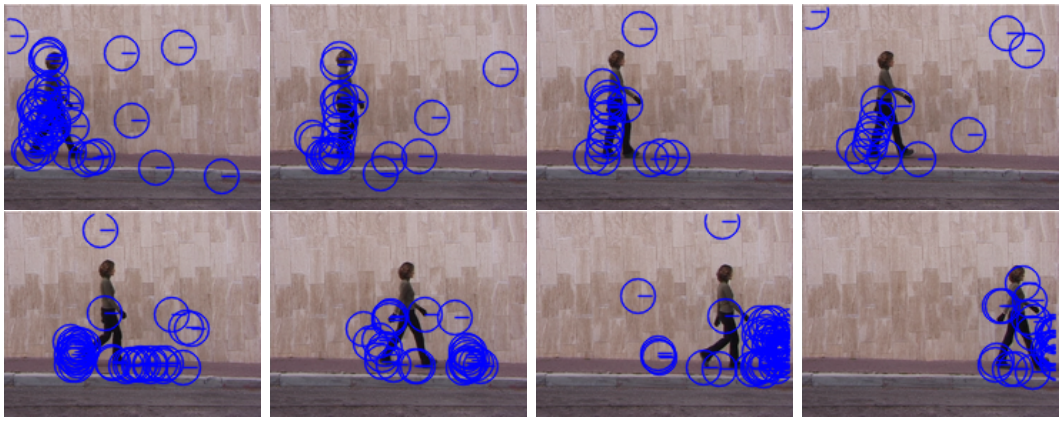
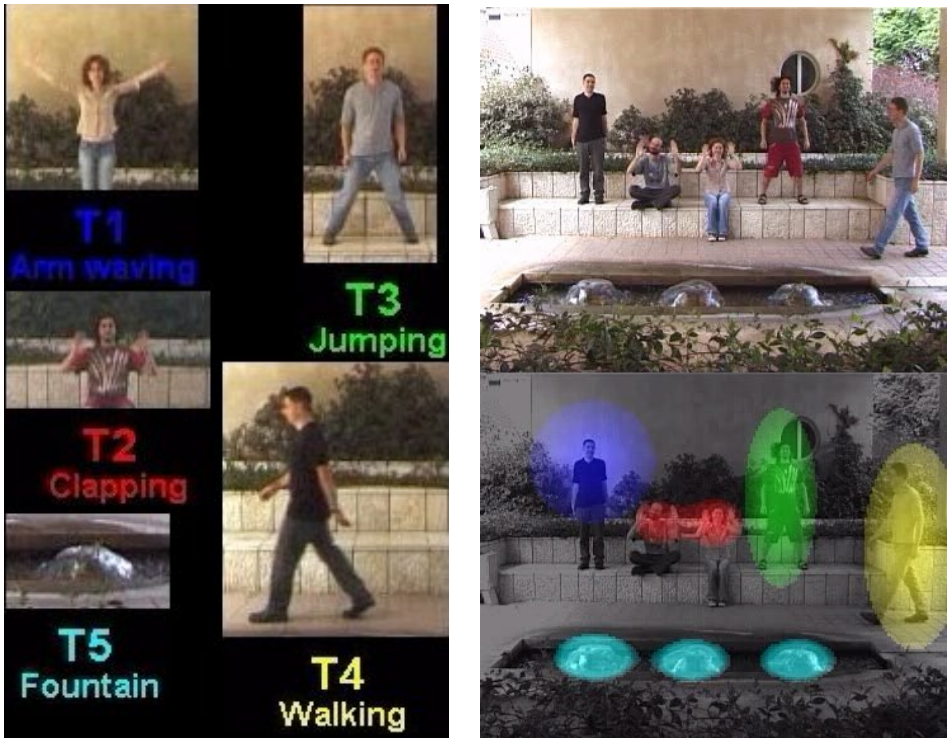


FIGURE 2.4 – *Global Motion* à partir des *Motion History Image* (MHI) sur le jeu de donnée Weizmann.

être mise en œuvre : l'identification du personnage est faite par un algorithme de détection d'humain par exemple.

Bobick et Davis [24] ont proposé le concept de « Motifs Spatio-temporels » (*Spatio-Temporal Template*). Il s'agit d'une représentation des actions qui, tout en étant dépendante de la vue que l'on a de celle-ci, permet de les exprimer par un mouvement au cours du temps. Leur système se base sur des *Motion-Energy Images* ou des *Motion-History Images*, voir figure 2.4, comme représentants du mouvement, afin de reconnaitre des actions par leur forme au cours du temps. Un volume spatio-temporel est construit par une extrusion au cours du temps de la silhouette de l'acteur. Cette modélisation simple du personnage associée à une reconnaissance par *Template Matching* leur permet de reconnaitre des actions comme : « s'assoir », « agiter les bras » ou « s'accroupir ». De plus, leur solution est suffisamment efficace pour être calculée en temps réel. Shechtman et Irani [120] ont appliqué un principe similaire en utilisant des *patchs* au lieu de la silhouette complète. Leur méthode utilise des corrélations de petites parties du volume extrait. Ainsi l'apprentissage est simplifié, seulement quelques *patchs* par activités. De plus, il est possible de reconnaitre de multiples activités se produisant sur la même séquence vidéo en entrainant des *patchs* avec des étiquettes différentes sur une même image. La figure 2.5 présente une illustration de ce principe. Elgammal et al. [36] se sont basés sur le paradigme des *examplars* et une extraction des silhouettes, afin de reconnaitre des mouvements. Un espace des *examplars* est défini par un ensemble d'exemples « représentatifs », issus du jeu de données d'entrainement, muni d'une fonction de distance entre deux points de cet espace. En accord avec la définition de Frey et Jojic [41], les *examplars* sont considérés comme le centre d'un mélange probabiliste, généralement gaussien. Le paradigme des *examplars* permet de modéliser rapidement leurs occurrences à l'aide d'un système de



(a) Patchs d'apprentissage (b) Séquence à reconnaitre

FIGURE 2.5 – Vislab : Reconnaissance par patchs : à partir des patchs de la figure (a), la corrélation maximum avec l'activité est présenté en figure (b). (©Shechtman et Irani [120])

Markov. Frey et Jojic [41] ont utilisé les *examplars* comme état du processus de Markov, alors que Elgammal et al. [36] utilisent les *examplars* comme des états cachés, voir la figure 2.7. Les systèmes Markoviens ont l'avantage d'être efficace en terme de calcul, et simple à étendre. En effet, pour ajouter une action, un seul *examplar* permet de mettre à jour le système probabiliste. Dès lors, il est possible de proposer une reconnaissance de cette action. Néanmoins, la solution proposée par Elgammal et al. [36] demeure contrainte par les *examplars* eux-mêmes, issus des silhouettes 2D, pas assez discriminantes et dépendantes de ce fait du point de vue. Weinland et Boyer [143] ont proposé d'utiliser des *examplars* à partir de silhouettes pour reconnaitre des actions. Pour cela, ils calculent la distance d'une séquence de silhouettes ou de contours par rapport à leur ensemble d'*examplars*. Chaque séquence est ainsi représentée sous la forme d'un point dans un espace à N dimension, $\mathbb{R}^N$. L'espace ainsi créé est partitionner grâce à un classifier de type AdaBoost. Leur méthodologie présente l'avantage d'être indépendante de la variation temporelle d'une action. Ahmad et Lee [2] ont étendu le concept d'historique

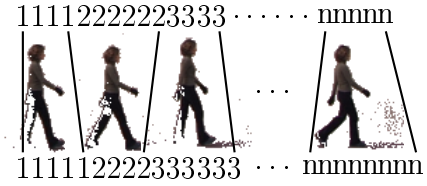


FIGURE 2.6 – Alignement d'*Examplar* par *Dynamic Time Warping*.

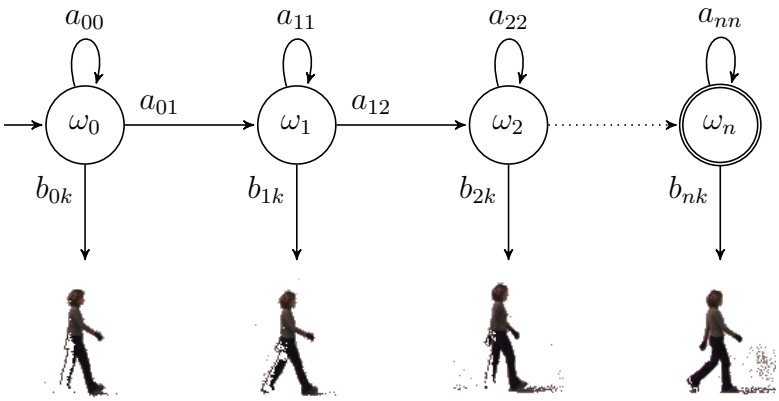


FIGURE 2.7 – Exemple de Modèle de Markov Caché pour l'action « Walk ».

de mouvements, *Motion History*, au cas des silhouettes. Leur reconnaissance fait appel à une classification par machine à vecteurs de support, *Support Vector Machine (SVM)*. Leur solution repose sur de nombreux paramètres et se révèle donc sensible à leurs réglages.

Huang et Trivedi [49] ont proposé le concept d'histogramme cylindrique, dans une représentation basée sur des voxels (cubes élémentaires), couplé à une modélisation par chaînes de Markov cachées. Cet histogramme est basé sur une reconstruction tridimensionnelle du personnage à partir de silhouettes multi-vues synchronisées. Ces histogrammes de fréquences de voxels sont utilisés comme état caché de leur système. Il s'agit donc d'une mesure de la répartition du volume dans l'espace 3D entourant l'acteur. Weinland et al. [144] ont utilisé une reconstruction voxélique ainsi que le paradigme des *examplars* pour entrainer un modèle stochastique indépendant du point de vue de l'action. Dans leur solution, la reconstruction 3D n'est pas nécessaire, contrairement à Huang et Trivedi [49], et aucune connaissance, *a priori* sur le positionnement des caméras n'est nécessaire. Les limitations de cette méthode se retrouvent dans l'extraction des silhouettes, ainsi que dans la nécessité d'avoir un réseau de caméras calibrées. Ceci implique une infrastructure lourde à mettre en œuvre, et restreint le champs des actions reconnaissables. Les silhouettes et les modèles de Markov ont été utilisés conjointement dans le travail

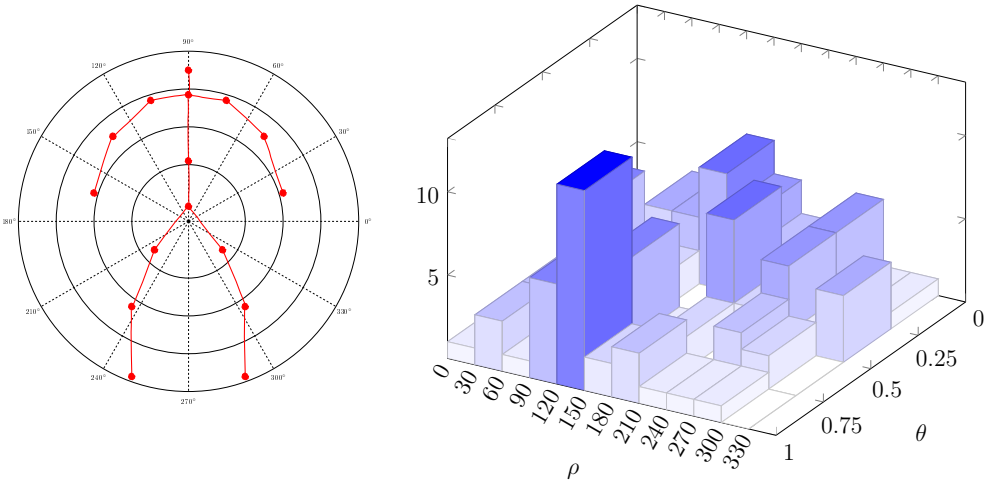
de Xiong et Liu [152]. Les primitives utilisées, principalement le centre de gravité de la silhouette, étaient restreint aux comportements simples des humains. L'extraction des silhouettes comme un volume spatio-temporel a été utilisé par Yilmaz et Shah [158]. Ils construisent un volume par une association des points du contour, à l'aide d'un graphe bipartite, technique empruntée à la stéréovision. La construction de ce volume est différente de celles précédemment proposées, car elle n'utilise pas de voisinage local, mais une correspondance des contours de la silhouette entre deux intervalles de temps (voir la figure 2.8). Ensuite, ils extraient des points caractéristiques sur ce volume afin de catégoriser leurs actions. L'extraction des silhouettes, ainsi que l'extraction des points caractéristiques sur le volume sont coûteuses en temps de calcul. De plus, ces calculs reposent sur un volume sensible aux erreurs du fait de sa dépendance aux silhouettes. Bien que l'approche soit établie sur une forme tridimensionnelle, elle n'est pas totalement invariante au point de vue, car elle souffre toujours des ambiguïtés inhérentes aux silhouettes. Une modélisation différente du volume spatio-temporel des silhouettes a été proposé par Ke et al. [57]. Les auteurs sur-segmentent le volume spatio-temporel et utilisent une reconnaissance par partie de leur volume. Une recherche de régions similaires est faite dans le volume pour reconnaitre une action. Un suivi par flux optique des régions permet d'accélérer le processus de reconnaissance en limitant la zone de recherche. Notons toutefois que cette approche est plus résistante qu'une extraction de silhouette classique dans des environnements complexes, comme une foule. La sur-segmentation de leur volume peut conduire à des erreurs d'interprétations et à l'extraction de patchs incohérents par une agrégation inappropriée des petites régions. Shao et Chen [119] ont proposé une reconnaissance utilisant directement les silhouettes comme des vecteurs caractéristiques d'une action. Ils construisent un sac de mots visuels à l'aide de l'algorithme *KMeans*. Chaque silhouette est projetée dans un espace de dimension inférieure, à l'aide d'une régression spectrale, puis un histogramme des occurrences de chacun des mots visuels est construit par association au voisin le plus proche. La régression spectrale a l'avantage d'être non linéaire, comme l'est le problème de reconnaissance d'actions. Leurs résultats sur le jeu de données Weizmann sont parfait, *i.e.* 100% reconnaissance. Néanmoins, leur solution souffre de manque de robustesse, notamment dès que les silhouettes sont altérées. De façon comparable, Tseng et al. [130] ont proposé une approche construisant un graphe spatio-temporel des actions. Ils lient les silhouettes d'une même action entre elles, et effectuent une réduction de dimension de l'espace des silhouettes afin de séparer chacune des classes d'actions. La reconnaissance est faite à l'aide d'une classification des k plus proches voisins. Notons ici, que la plupart des méthodes utilisant des silhouettes, souffrent d'une sur-sensibilité à ces primitives.

FIGURE 2.8 – Représentation du volume spatio-temporel par stéréo-association des points du contour de la silhouette extraite à chaque image, afin de créer une surface 3D. (©Yilmaz et Shah [158])

2.3.2 Analyse du squelette

Lorsque l'on s'intéresse à l'analyse de l'activité d'un être humain, l'identification de l'acteur peut être un premier pas important. Néanmoins, la forme « brute » du personnage, telle qu'une silhouette peut la fournir, n'est pas toujours discriminante. Certains travaux se sont intéressés à la prise en compte de la posture de l'utilisateur. Leur postulat s'appuie les travaux réalisés en psychologie par Johansson [54]. Johansson a montré que les trajectoires des membres, tête, épaules, genoux, pieds, mains, *etc.*, étaient suffisantes pour qu'un humain puisse reconnaitre une action. Ainsi, les auteurs ont repris ce postulat en l'adaptant à la reconnaissance automatique d'actions. Ils supposent qu'il est plus aisé de reconnaitre une action si l'on peut identifier, même de façon approximative, la position de certains membres clés, comme les mains, les jambes ou la tête. Cette représentation de l'humain par un squelette a conduit les travaux vers une estimation de plus en plus précise de la posture afin de reconnaitre des actions humaines. L'une des premières solution à avoir utilisé une représentation sous forme de squelette humain simplifié est Gavrila et Davis [43]. Les auteurs ont proposé une approche combinant la capture de mouvements 3D, issue de plusieurs caméras, afin d'avoir un squelette simplifié du personnage, et l'algorithme d'alignement de séquence temporelle *Dynamic Time Warping*, pour reconnaitre des actions humaines. La complexité calculatoire était rédhibitoire à l'époque (1995), des simplifications du squelette se sont imposées. Yacoob et Black [153] ont proposé une approche utilisant une segmentation du corps pour en extraire les articulations principales, ainsi qu'une réduction de dimensions par PCA. Chaque activité est modélisée par un *examplar*. La reconnaissance se fait par un alignement temporel des parties du corps. Fujiyoshi et Lipton [42] ont proposé une méthode qui consiste à extraire le contour de la personne, pour en calculer un squelette simplifié. Ils introduisent ainsi le concept de *star skeletonization* ou *star skeleton*. Il s'agit d'une représentation du corps humain par l'intermédiaire d'un squelette à cinq branches maximum, représentant les extrémités des jambes et des bras, ainsi que la tête. De l'extraction de cycles, ils analysent les mouvements de marche et de course. L'extraction de squelette est simple, il s'agit des

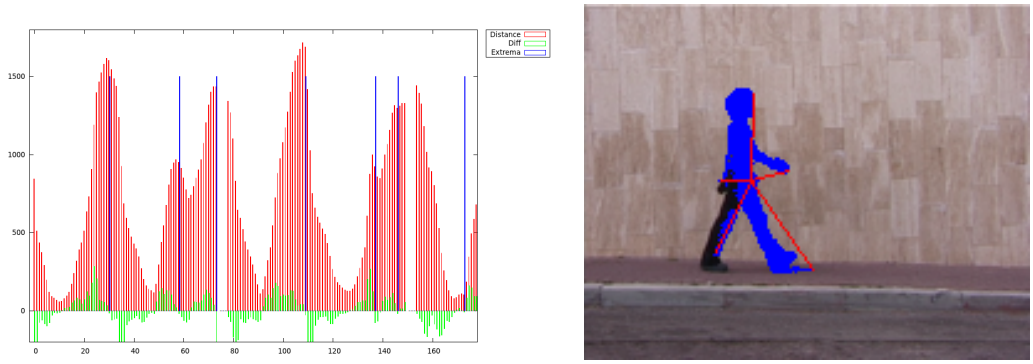
cinq maximum locaux sur le signal unidimensionnel de la distance des points du contour au centre de gravité. La figure 2.10 montre un cas d'utilisation — sur le jeu de données Weizmann — de l'extraction du *star skeleton*. Ali et al. [3] ont utilisé la théorie des systèmes chaotiques et la recherche d'invariants dans cet espace dynamique pour catégoriser des actions par l'intermédiaire de l'algorithme *k-Nearest Neighbors*. Bien que ces approches soient efficaces pour reconnaitre des actions issues d'images, elles sont tributaires de la qualité de l'extraction faite, notamment pour l'extraction 2D du squelette. Ainsi, Ziaeefard et Ebrahimnezhad [161] ont montré que le choix de la méthode d'extraction impactait directement la qualité de la reconnaissance. Dans ces travaux, ils ont obtenu des variations sur la même action allant de 100% à moins de 90% de reconnaissance, en utilisant la même méthode, mais une méthode d'extraction du squelette différente. La précision moyenne varie de 4.5% sur le jeu de données Weizmann. Leur méthodologie reposant sur le calcul d'un histogramme polaire des positions des articulations au cours du temps. Il s'agit d'un histogramme dont les coordonnées s'expriment en (ρ, θ), comme présenté sur la figure 2.9. Cette méthodologie a montré sa validité dans des cas plus complexes, comme l'ont prouvé Tran et al. [129]. En reprenant la méthodologie de l'histogramme polaire, et une réduction des dimensions engendrées — analyse en composantes principales — Tran et al. [129] entrainent une machine à vecteur supports afin de reconnaitre des actions. L'extraction de squelette est celle décrite par Andriluka et al. [5]. Elle se montre plus robuste et plus précise (10 articulations) que celles utilisées par Ziaeefard et Ebrahimnezhad [161], d'où l'amélioration de leurs résultats. Parameswaran et Chellappa [96] ont proposé une méthode de reconnaissance d'actions basée sur une capture de mouvements 3D afin de reconnaitre les actions d'un humain. Ils construisent un espace de mouvements invariant aux actions, pour reconnaitre chacune d'entre elles. Comme le reconnaissent les auteurs, il n'existe pas de tels invariants dans un espace 3D. Dès lors, leur méthode est construite sur une modélisation indépendante chacune de leurs actions. Leur proposition est donc très statique quant à l'ajout de nouvelles actions. Néanmoins, ils ont montré l'importance de la capture de mouvements 3D du corps humain dans le contexte de la reconnaissance d'actions. La dépendance au point de vue est l'un des défis les plus importants dans la reconnaissance d'actions. En effet, la projection du monde réel, tridimensionnel, sur le capteur, bidimensionnel, de la caméra, induit une perte d'information préjudiciable à la discrimination d'une action. Les solutions ayant utilisé une approche tridimensionnelle ont montré leur efficacité. Ramanan et Forsyth [104] ont utilisé un apprentissage de capture de mouvements 3D et une reconnaissance de mouvements se basant sur une capture de mouvements 2D. Une phase d'annotation manuelle des données 3D est nécessaire mais permet ensuite de reconnaitre des mouvements avec une acquisition vidéo 2D. Une modélisation par chaînes de



(a) Calcul de l'histogramme polaire pour un squelette d'animation.

(b) Exemple de représentation de l'histogramme polaire d'une action.

FIGURE 2.9 – Représentation d'un histogramme polaire, tel qu'il a été présenté par Ziaeefard et Ebrahimnezhad [161]. (a) Chaque point du squelette extrait est ajouté à l'une des 48 possibilités (12 orthants × 4 cercles). (b) L'histogramme est la somme de tous les points du squelette de l'action tombant dans la même classe.



(a) Distance du centroïd au contour

(b) *Star skeleton* sur Lena, jeu de données Weizmann

FIGURE 2.10 – Le signal est filtré afin de faire ressortir les maxima locaux. Le signal vert représente la différence entre deux valeurs consécutives de l'histogramme. Les maxima sont identifiés par un passage « positif → négatif » du signal de différence, et indiqués par les barres bleues. D'après Fujiyoshi et Lipton [42].

Markov et un classifieur de Maximum *a posteriori* (MAP) est utilisé pour la reconnaissance. Chacune des articulations contribue à la reconnaissance de l'action, ainsi, un saut pendant une course peut être identifié.

Allant au-delà de la seule reconstruction tridimensionnelle et de l'analyse de trajectoires, Lv et Nevatia [77] ont utilisé des primitives issues du domaine de l'animation, les « poses clés » ou *key-poses*. Il s'agit de poses caractéristiques d'une séquence d'animation présentée sous la forme de squelettes rigides approximés du corps humain, qui servent, généralement, à l'interpolation d'une animation. Lorsqu'un graphiste défini une animation d'un personnage, il ne représente pas l'ensemble des poses du mouvement. Souvent, l'animation se joue à une fréquence trop élevée pour que cela ne soit pas trop fastidieux (30Hz au minimum). Il décrit son animation par les poses importantes — dites poses clés — et laisse le système construire l'interpolation pour les autres poses, à la fréquence désirée. Pour plus d'explication sur les travaux issus du domaine de l'animation, voir la section 2.4. Dans le travail de Lv et Nevatia [77], les poses clés sont extraites à partir des données de capture de mouvements, en cherchant le minimum de variation dans une fonction d'énergie. Les actions sont ensuite modélisées par une chaîne de poses clés. Ils reconstruisent ensuite uniquement les poses identifiées comme clés dans le processus de reconnaissance et font une comparaison avec les poses clés apprises, par l'intermédiaire de l'algorithme *Pyramid Match Kernel* (PMK) qui calcule l'intersection des histogrammes de poses reconnues. Néanmoins, chacune des variations internes à une classe doit être modélisée indépendamment — comme « s'assoir par terre » et « s'assoir sur une chaise » — ce qui peut être un frein à l'apprentissage. Une autre limitation vient de la difficulté d'extraire des poses clés pertinentes, de façon totalement automatique. En animation de personnage, les poses clés sont créées par un graphiste spécialiste. Cette compétence est difficile à mettre en œuvre dans un système informatique. Sheikh et al. [121] ont appliqué une projection affine aux trajectoires afin de reconnaitre des actions, de façon invariante au point de vue, par la mesure des angles entre articulations dans l'espace projeté. Rao et Shah [105] ont utilisé une extraction des pics dans les trajectoires, plutôt que les trajectoires complètes, afin de représenter chacune des actions en terme de caractéristiques constantes. Ces caractéristiques sont invariantes au point de vue et utilisées comme motifs d'actions. Ils effectuent ensuite une recherche de ces motifs dans les actions à reconnaitre. Comme ils utilisent un seul point de vue par acquisition, ils ne sont pas en mesure d'utiliser les trajectoires du corps en entier. Pour cela, ils se sont focalisés sur les mouvements de la main, en utilisant un détecteur de peau pour les extraire.

Lorsque l'on utilise des primitives de plus haut niveau, il est important de ne plus se focaliser sur les modifications locales, mais sur une analyse plus complète de l'action. Dans cette optique, Cuntoor et al. [32] ont suggéré que l'utilisation

des trajectoires fournissait l'approche la plus discriminatoire dans la reconnaissance d'actions humaines. De nombreuses recherches se sont alors orientées dans le sens de l'analyse 3D de trajectoires. Li et Fukui [70] ont appliqué l'analyse de trajectoires à des données de capture de mouvements tridimensionnelles. Les expériences — utilisant uniquement des mouvements de caméras, pour générer le bruit de reproduction d'une action — ne permettent pas de valider leur approche de façon certaine. Néanmoins, elle présente un premier pas quant à l'utilisation conjointe des trajectoires et de la capture de mouvements. Lorsque l'on utilise la capture de mouvements comme primitive d'une action, il peut être profitable d'assurer quelques contraintes en amont. Baak et al. [7] ont ainsi présenté une solution qui utilise une base de connaissance *a priori* afin de créer des dépendances entre des comportements et l'extraction de mouvements associé. Ils ont ainsi ajouté des contraintes à la marche — comme « les pieds doivent être en contact avec le sol » — afin d'augmenter la précision de leur capture de mouvements. Il s'agit d'un des rares cas d'utilisation de l'interprétation de mouvements dans une optique d'amélioration de la qualité d'une animation. Bien que l'amélioration de la capture de mouvements produise des résultats encourageant pour la reconnaissance d'actions, de façon générale, cette précision n'est pas nécessaire dans un contexte de reconnaissance d'actions. Notamment, la méthode de capture de mouvements présenté par Shotton et al. [122] — couplé au dispositif Kinect™ de Microsoft — suffit, dans la plus part des cas, à fournir une capture de mouvements précise et fiable ; sans avoir besoin d'être améliorée par une solution telle que celle-ci. Han et al. [47] exploitent la capture de mouvements afin de représenter les trajectoires de chacune des articulations dans un espace manifold. Leur méthode nécessite de grandes bases d'actions pour l'apprentissage, car elle utilise une simplification hiérarchique du squelette d'animation, au lieu de l'intégralité de celui-ci. Dès lors, des ambiguïtés peuvent apparaitre entre des mouvements ainsi représentés. Un processus complexe et coûteux en terme de temps de calcul est nécessaire à la bonne classification des actions similaires. Allant un peu plus loin avec l'analyse de squelette, Raptis et al. [106] ont montré l'efficacité conjuguée de la capture de mouvements de Shotton et al. [122] et du périphérique Kinect™, dans l'interprétation d'actions de danse. Leur solution est dépendante de la connaissance du nombre initial d'actions ainsi que des scenarii d'utilisation de celle-ci afin de limiter le taux d'erreurs. En contrepartie, la solution fournit un oracle quant à la distance de reproduction de l'action ; méthode indispensable dans le cadre d'un jeu vidéo de danse, comme l'application proposé dans ces travaux. Yao et al. [156] ont proposé la création d'un espace probabiliste de variable latent et non linéaire, afin de discriminer des actions complexes, et un apprentissage par un classifieur de type *Gaussian Process Latent Variable Model* (GPLVM) utilisant une descente de gradient stochastique. L'utilisation du jeu de donnée *TUM Kitchen Dataset*,

relativement complexe en terme d'activités, permet de montrer un réel apport de la capture de mouvements pour la reconnaissance d'actions humaines. Ceci est confirmé par l'article suivant, Yao et al. [155], soulignant le potentiel de la capture de mouvements, seule ou en conjugaison avec des primitives images classiques. Wang et al. [138] ont proposé une approche de reconnaissance d'actions dans un environnement d'intérieur — *i.e.* le canapé devant la télévision — combinant une approche par capture de mouvements et une extension des méthodes 2D usuelles par points caractéristiques aux caméras de profondeur. Leur postulat est d'établir une discrimination entre actions proches, au niveau du squelette d'animation — par exemple « manger » et « boire » — par l'utilisation d'informations discriminantes sur la profondeur autour des articulations impliquées dans l'action. En cela, ils sont proches des volumes spatio-temporels de Blank et al. [23] notamment, à la différence près que leur volume est spatial et non temporel, informations fournit par le capteur de profondeur du dispositif Kinect^{TM}. Ils proposent un algorithme de fouille d'articulations impliquées dans une action (*actionlet mining*) et l'utilisation de modèles locaux d'occupation de l'espace environnant, *Local Occupancy Patterns* (LOP) afin de reconnaitre des actions complexes dans le jeu de données *MSR Daily Activity 3D* (voir le détail du jeu de données dans l'annexe Jeux de données 3D, section A.4). Les *Local Occupancy Patterns* peuvent être vus comme des points caractéristiques sur les images de profondeurs. Ils sont tributaires d'une acquisition de profondeur, en plus de la capture de mouvements.

Guerra-Filho et Aloimonos [46], dans un article d'analyse neurolinguistique ont proposé une ontologie des gestes et des verbes de la langue naturelle. Pour cela, ils ont étudié les variations des membres du corps à l'aide de données de capture de mouvements. Chaque membre peut avoir un état parmi quatre : vitesse positive ou négative et accélération positive ou négative. Ils infèrent une grammaire de mouvements qu'ils corrèlent avec la communication du ou des sujets en interaction. Ils expriment en quelque sorte l'ADN du mouvement d'interaction. Voir la figure 2.11 pour une représentation visuellement proche des caryotypes. Ce travail est une preuve de la corrélation de haut niveau des actions humaines, dans les contextes d'interaction, et de son extraction aisé à partir de données de bas niveau, issues des données de capture de mouvements. Les auteurs montrent les constructions complexes possibles à partir de l'analyse de mouvements, notamment celle représentée par un squelette d'animation rigide.

2.4 Emprunt à l'animation de personnages

L'analyse d'un squelette rigide d'animation est un domaine qui a été étudié à d'autres escients. Dans la communauté d'animation de personnages, certains travaux se sont intéressés à la reconnaissance d'actions afin de fournir une segmen-

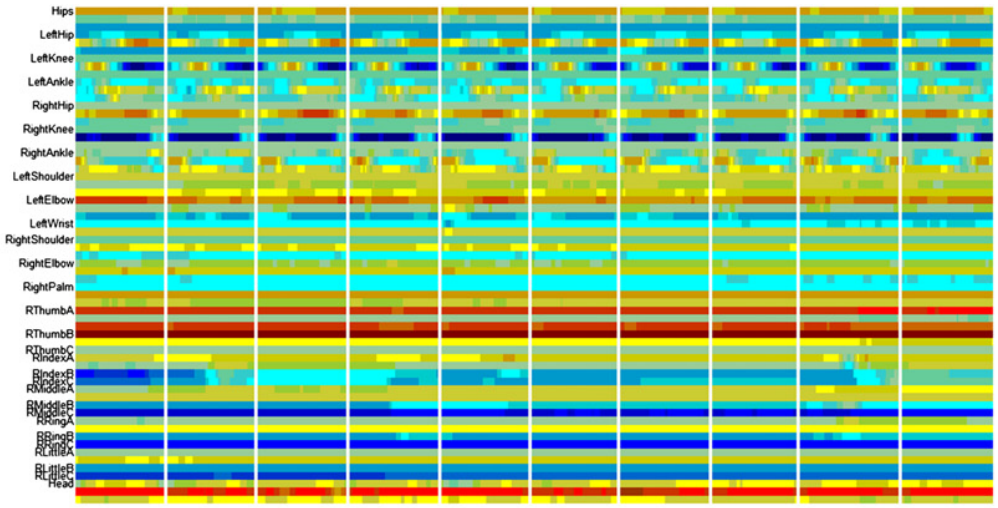


FIGURE 2.11 – Mesure de trois angles d'Euler par articulation — chacune représenté par une couleur — pour dix exécutions du même mouvement. Chaque ligne représente une articulation du squelette d'animation. (©Guerra-Filho et Aloimonos [46])

tation automatique des grandes bases d'animation disponibles, et ainsi d'améliorer les transitions entre animations. Il s'agit d'un problème complexe qui est généralement résolu manuellement. Il s'apparente à la reconnaissance d'action à partir du squelette mais en considérant que l'acquisition d'un squelette précis a déjà été faite au préalable, et que les données de capture de mouvements sont disponibles en grande quantité. Le facteur qualité est très important dans cette découpe et conditionne le résultat visuel. En effet, les utilisateurs de ses résultats (graphistes, joueurs, *etc.*) sont très sensibles aux incohérences dans une animation, comme une transition de marche qui ne se ferait pas sur le bon pied, par exemple. Dans cet esprit, Barbič et al. [9] ont proposé une utilisation de l'Analyse en Composantes Principales (ACP ou *PCA*) et un Modèle de Mélange de Gaussiennes (*GMM*) afin de reconnaitre les changements entre actions. Ils recherchent les maxima locaux (pics) dans la distribution de probabilités des poses. Leur solution impose de connaitre le nombre de comportements initiaux et souffre de faiblesses quant à l'absence de niveau de détails sur l'animation. En effet, chacune des parties du corps est d'une importance égale. Dès lors, des mouvements non pertinents — que l'on peut qualifier de bruit de gestes — sont inclus dans leur segmentation, et viennent parasiter les résultats de l'ACP. De même, en accord avec leur objectif de transitions visuellement lisses entre des animations, ils considèrent que l'orientation initiale de l'action est importante. Ainsi, des mouvements de marche seront différents s'ils représentent une marche en cercle, une marche droite, *etc.* Cette

distinction est importante dans leur modélisation du problème. Dans le cadre plus général de la reconnaissance d'actions, il est souvent nécessaire de regrouper ces différents comportements. Leur solution ne permet pas de le faire aisément, et conduit à une sur-segmentation des comportements reconnus. Toujours dans le même objectif, Beaudoin et al. [18] ont recherché des motifs fréquents dans des bases d'animations. Les auteurs définissent un motif comme une sous-séquence d'animations proches dans l'espace du mouvement. Leur solution exhibe automatiquement les transitions entre les différents motifs afin de construire un graphe d'animations, qui est ensuite étiqueté à la main. Il s'agit avant tout d'un outil de transition fluide entre animations, mais il peut aussi être vu comme une structure de reconnaissance. Le travail de groupement (*clustering*) l'extraction des motifs — processus hors-ligne par nature — coûteux en temps de calcul, et l'étiquetage manuel des transitions, rendent cette solution fastidieuse, et bien loin du temps réel. Elle n'est, de fait, pas adaptée à la reconnaissance automatique d'actions. Müller et al. [87] ont proposé une solution d'annotation de grandes bases d'animations, de façon automatique. Les comportements et sous-comportements — parties d'une action commune à plusieurs actions — sont clairement mis en évidence. Ils proposent un outil à destination des graphistes, qui permet des recherches facilitées, par comportement ou par contrainte (un pied d'appui pour une transition par exemple) dans un but de création de transitions fluides. Néanmoins, il est peu souple à l'ajout de nouvelles actions et ne propose pas de structure de reconnaissance. Toutes les solutions décrites ci-dessus, et généralement toutes celles en fouille de base de données de capture de mouvements se heurtent à la contrainte du temps réel. L'évolution de la base d'apprentissage de ces méthodes est trop contrainte pour une utilisation en reconnaissance d'actions. En effet, la plupart des solutions demandent une phase préalable d'étiquetage manuel des actions.

2.5 Vidéo et interaction Homme–Machine

La reconnaissance d'actions ou plus souvent de gestes a été largement étudié dans une optique d'interaction Homme–Machine, citons notamment [4, 6, 52, 69, 95, 134, 160]. Les méthodes présentées dans cette section ne suivent pas une approche unique pour la reconnaissance d'action. Elles sont parfois des simplifications de méthodes précédemment présentées afin de les rendre compatible avec des contraintes de temps réel. L'interaction Homme–Machine est un nouveau champs d'expression pour la reconnaissance d'action. Elle combine des contraintes fortes : le temps réel, la flexibilité exprimée par les utilisateurs dans la reproduction de leurs gestes, la volonté d'ajouter de nouveaux comportements « à la volée », *etc.* Elle représente un des terrains les plus complexe pour l'application de la reconnaissance d'actions.

Ivanov et Bobick [52] ont proposé un système de reconnaissance d'actions par grammaire formelle. Ils construisent une grammaire, indépendante du contexte et du détecteur de mouvements, qu'ils reconnaissent à l'aide d'un modèle de Markov à états cachés (HMM). Leur méthode repose sur un système de détection stéréoscopique, permettant de contrôler un ordinateur avec les mains. Une application à la vidéo surveillance est présentée pour la détection d'activités « anormales ». Dans leur modèle, une erreur dans l'analyseur de la grammaire est interprétée comme une activité non prévue. Dans ce cas, l'activité est considérée comme anormale. Zhang et al. [160] ont proposé en 2001 une méthode pour interagir, par l'intermédiaire d'une caméra, avec un ordinateur. En utilisant un marqueur — une feuille de papier — ils proposent des interactions simples, telles que : pointer (désigner) une zone, cliquer, tourner, dessiner. Ces interactions sont proches des interfaces tactiles actuelles, mais nécessite une feuille de papier afin de faciliter le suivi de mouvements nécessaire à son fonctionnement. De plus, les actions reconnaissables sont limitées, et assez difficilement extensibles. Van den Bergh et al. [134] ont proposé une approche générique des interactions homme–machine. Leur solution repose sur le suivi des mains des utilisateurs et l'interprétation de configurations des doigts. Ils utilisent de plus la position des yeux pour donner une direction à l'action : pointage d'une zone, déplacement d'une zone à une autre, *etc.* En théorie, toutes les configurations possibles de doigts sont exploitables, en pratique, seules certaines sont reconnues, à cause d'ambiguïtés visuelles possibles. Par ailleurs, une méthode exploitant le regard et la main pour pointer est difficilement exploitable en dehors d'un écran projeté. En effet, dans une configuration de poste de travail classique, l'utilisateur est trop proche de son écran pour que la caméra puisse voir la main et les yeux en même temps. D'autre part, les configurations des doigts sont peu intuitives — au-delà du pointage — pour une utilisation standard du système. Argyros et Lourakis [6] utilisent aussi les configurations des doigts, pour contrôler la souris, à l'aide d'un système de stéréovision. De même que pour la méthode proposée par Van den Bergh et al. [134], les configurations des doigts sont peu intuitives et difficiles à tenir. L'utilisation de la stéréovision, plus complexe à mettre en œuvre, est compensée par l'absence de suivi des yeux. Néanmoins, leur système est restreint aux présentations, car seules les actions à la souris sont possibles. Toujours à l'aide des doigts, Li et al. [69] ont proposé un écran tactile à l'aide d'un système de vision. Ils utilisent deux caméras tournées vers l'écran — *Eyescreen* — et utilisent les principes de stéréovision pour détecter les contacts entre les doigts et l'écran. Ainsi, ils sont en mesure de « simuler » toutes les interactions tactiles, devenues habituelles désormais. Leur système est aussi capable d'identifier des configurations de doigts même lorsqu'ils ne sont pas en contact avec l'écran. Il s'agit de l'un des premiers systèmes utilisant les doigts qui n'est pas destiné aux écrans projetés, mais à une utilisation dans un poste de travail classique. Néanmoins, ils

souffrent des mêmes faiblesses que leurs prédécesseurs, le jeu de 5 configurations de doigts reconnu est trop simple pour des interactions efficaces dans bien des situations. Allant par-delà les interactions des mains uniquement, Okada et Stenger [95] ont proposé une méthode permettant de contrôler un avatar en temps réel à l'aide du corps en entier. Ils utilisent une approche se basant sur des silhouettes afin de construire une hiérarchie de forme. Ils effectuent la capture de mouvements en même temps que l'injection de celle-ci dans un univers 3D. Dès lors, l'utilisateur peut interagir avec un environnement virtuel par l'intermédiaire de son avatar. Cette solution est couteuse en terme de calcul, car elle nécessite une trentaine de morphologies différentes pour construire la hiérarchie et un super calculateur de type *Cell Broadband Engine*™ pour évaluer la pose de l'utilisateur. Bien qu'ils n'utilisent pas de reconnaissance d'actions en tant que tel, ils sont parmi les premiers à exploiter le corps en entier dans des interactions homme–machine. Leur système manque de flexibilité quant à l'ajout d'une nouvelle pose, et gagnerait à utiliser une reconnaissance des actions dans bien des contextes.

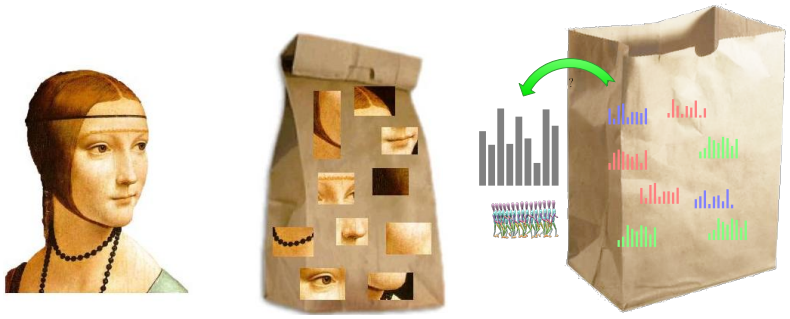
2.6 Stratégies de classifications

Lorsque l'on dispose de primitives pertinentes, comme celles présentées dans les sections précédentes, il est possible de voir la reconnaissance d'actions comme un problème de classification. Étant donné une action, modélisée par ses primitives, et son étiquette associée, *i.e.* son nom, reconnaitre une action revient à (*i*) construire un espace où les actions de même étiquettes sont proches et les actions d'étiquettes différentes sont loin (*ii*) trouver les frontières de cette espace pour créer une partition par étiquette. Dans les sous sections suivantes, nous allons présenter différentes méthodologies de classification, en suivant la décomposition proposée par Poppe [103].

2.6.1 Classification par représentants

Sacs de Mots

Les Sacs de Mots (*Bag of Words*) sont une méthode de description par les occurrences d'un dictionnaire de caractéristiques. Cette méthode considère que les récurrences des caractéristiques d'une action, par exemple, est discriminante au regard des autres. Ainsi, en associant chaque primitive issue d'images, de vidéo, de modèle 3D, *etc.* à un représentant, cette méthode postule que cette distribution est caractéristique de la classe à identifier. Les représentants sont généralement extraits automatiquement à partir de méthodes non supervisées, comme *K-Means*. Il s'agit d'une méthode de classification directe, qui est généralement associée à une autre méthode de classification, comme les Machines à Vecteurs de Support



(a) Sacs de mots images (b) Sacs d'histogrammes

FIGURE 2.12 – Représentation usuelle des sacs de mots. Les caractéristiques sont stockées de manière désorganiser, et servent de dictionnaire pour la construction de vecteurs de caractéristiques à partir de leurs occurrences.

ou les classifieurs Bayésien afin de fournir une représentation plus compacte des classes. La figure 2.12 représente une visualisation habituelle des sacs de mots dans le cas des caractéristiques images ou dans le cas des histogrammes.

Les sacs de mots visuels, présentés par Csurka et al. [31], sont inspirés des sacs de mots utilisés en reconnaissance de textes. Ils présentent une méthodologie adaptée aux caractéristiques visuelles et ont été appliqués avec succès à la reconnaissance d'action, notamment par Dollar et al. [34], Laptev et al. [67], Liu et al. [73], Niebles et al. [92]. D'autres approches ont appliqué le modèles des sacs de mots à la reconnaissance d'actions en prenant en compte les relations spatio-temporelles. Kovashka et Grauman [61] utilisent le voisinage spatio-temporel des points caractéristiques extraits afin d'améliorer la discrimination de leur modèle. De façon similaire, Ta et al. [125] utilisent l'apparence et les relations spatio-temporelles pour décrire une action par points caractéristiques.

La recherche d'un vocabulaire compact et discriminant est l'enjeu le plus important de cette approche. La création d'un vocabulaire compact passe souvent par une réduction de la dimension des vecteurs caractéristiques en entrée. Cette étape peut être opérée par une approche linéaire, comme l'Analyse en Composantes Principales (ACP ou *PCA*) Masoud et Papanikolopoulos [80], Rosales [108]. D'autres approches ont utilisées une réduction des dimensions non linéaires, citons notamment les travaux de Blackburn et Ribeiro [22], Chin et al. [29], Wang et Suter [141]. La réduction des dimensions peut aussi se faire de façon supervisée par rapport à la classe des primitives, citons notamment l'approche de Shao et Chen [119], qui utilisent une régression spectrale sur des silhouettes.

Avoir un vocabulaire compact est important, mais cela ne doit pas se faire

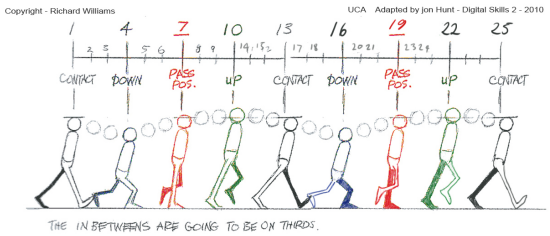


FIGURE 2.13 – Exemple de poses clé dans l'animation de personnage. D'après *The Animator's Survival Kit* de Williams [150].

au détriment de la discrimination de celui-ci. Ainsi, Kong et al. [59] ont proposé de créer un dictionnaire de manière adaptative. Ils construisent le dictionnaire de manière conjointe avec la réduction de dimensions des primitives, à l'aide d'une métrique pondérée.

Poses clés

Lorsque l'on s'intéresse spécifiquement aux poses dans une action, l'approche des sacs de mots peut être vue comme une recherche de poses clés.

Les poses clés sont issues des techniques de dessin, en particuliers les dessins animés. Ceux-ci étant traditionnellement fait à la main, il n'était pas envisageable de dessiner toutes les positions intermédiaires. Voir la figure 2.13 pour un exemple de *The Animator's Survival Kit* de Williams [150]. Cette méthodologie a ensuite été appliqué au monde de l'animation, où, au lieu de créer manuellement l'ensemble d'une action, un animateur ne va créer que certaines poses. Le système se chargera ensuite de faire une interpolation entre les poses créées afin de produire une séquence complète de mouvement. Si l'on s'intéresse à l'extraction de poses clés au lieu de leur création, il est nécessaire de trouver des poses possédant un maximum d'information localement, afin qu'elles puissent être de bonnes représentantes de ses prédécesseurs et successeurs.

Pour extraire les poses clés, certains travaux utilisent des *examplar* comme Weinland et Boyer [143] ; d'autres utilisent une mesure d'énergie, comme Gong et al. [45], Lv et Nevatia [77] ; ou encore une distance géométrique, comme Barnachon et al. [10].

Dans la majorité des cas, l'extraction de poses clés sert de phase intermédiaire entre les données en entrée (silhouettes, séquence de capture de mouvements, *etc.*) et la classification de l'action, par des classifications directes ou indirectes. Nous présenterons dans la suite ses méthodologies, dans le cas plus général de la reconnaissance d'actions. Voir un exemple de poses clés dans la reconnaissance d'action avec la figure 2.14.

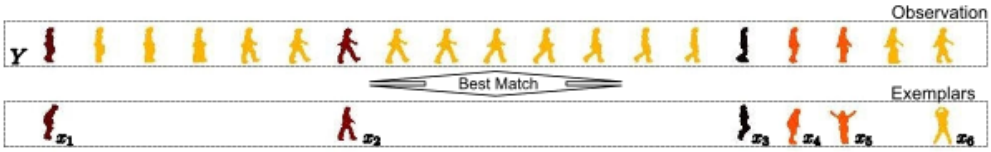


FIGURE 2.14 – Exemple de poses clés en reconnaissance d'actions. Des *Examplars* sont extraits, sous forme de silhouettes et reconnu par une fonction de distance. D'après Weinland et Boyer [143].

2.6.2 Classification directe

Machine à Vecteurs de Support

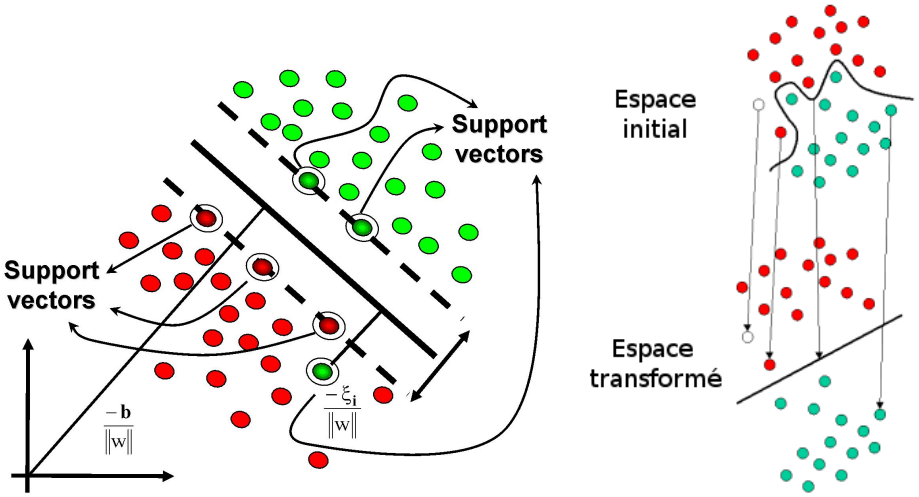
Les Machines à Vecteurs Support, de l'anglais *Support Vector Machine*(SVM), proposées par Vapnik [135], sont des classifieurs permettant d'apprendre une fonction de séparation d'un espace — muni d'un produit scalaire — à partir d'un ensemble d'échantillons. Les SVM reposent sur un apprentissage supervisé, c'est à dire que chaque échantillon d'apprentissage appartient à une classe connue. Ils proposent de construire un optimal hyperplan pour la discrimination les échantillons dans l'espace des caractéristiques (échantillons). Qu'ils soient utilisés pour une classification binaire ou non, les SVM permettent de prédire l'étiquette d'un nouvel échantillon par utilisation des vecteurs supports, c'est-à-dire les vecteurs les plus proches de la marge définie par l'hyperplan dans l'espace des caractéristiques.

Kernels : Les caractéristiques ne sont pas toujours linéairement séparables, voir la figure 2.15b pour un exemple. Dans ce cas, la méthode dite du *Kernel Trick* est appliquée. L'idée sous-jacente est de projeter les caractéristiques dans un autre espace, en utilisant un noyau de fonction. Dans cet espace, si le noyau est bien choisi, les caractéristiques deviennent linéairement séparables, et il est possible d'appliquer la classification par Machines à Vecteurs de Supports afin de trouver l'hyperplan optimal de séparation.

Les SVM ont été utilisés avec des caractéristiques locales de taille fixe, comme les histogrammes de points caractéristiques pour la reconnaissance d'action, notamment par Jhuang et al. [53], Laptev et al. [65], Schuldt et al. [117].

k Plus Proches Voisins

Si les SVM ne sont pas par natures multi-classes, les k Plus Proches Voisins (k *Nearest Neigbors* (kNN) sont une réponse immédiate aux problèmes multi-classes. En effet, il s'agit d'un classifieur permettant de catégoriser un point à partir de la classe associée à ses voisins les plus proches. Ils ont été utilisés dans la reconnaissance d'actions en projetant une séquence observée dans l'espace des carac-



(a) SVM linéaire (b) SVM non linéaire

FIGURE 2.15 – Représentation 2D des caractéristiques ainsi que des vecteurs supports. Chaque caractéristique est représenté par un point de couleur rouge ou vert. L'hyperplan de séparation est indiqué, ainsi que la maximisation des marges. La figure (a) représente un cas linéairement séparable, la figure (b) représente un cas non linéairement séparable dans lequel le *Kernel Trick* est utilisé afin de séparer les caractéristiques par un hyperplan.

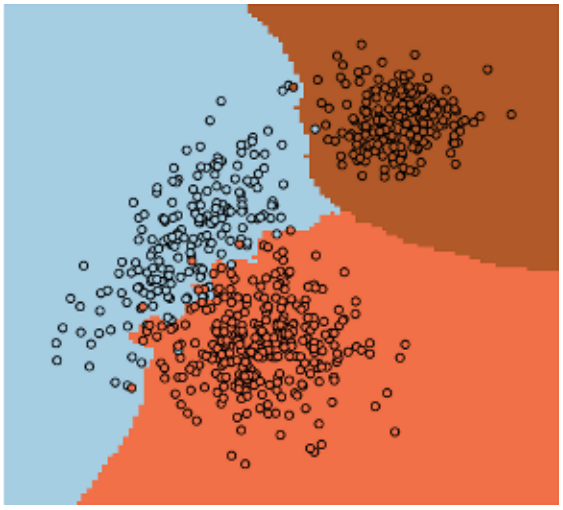


FIGURE 2.16 – Exemple de classification par k Plus Proches Voisins. Dans cet exemple, les caractéristiques en entrées, représentées par les cercles, permettent de classifier l'ensemble de l'espace image en considérant les 3 plus proches voisins, *i.e.* $k = 3$.

téristiques utilisées à l'apprentissage. Dans cet espace, à l'aide d'une fonction de distance, le label de la classe majoritaire parmi les voisins de la séquence observée est associé à cette séquence. Il s'agit d'une approche simple, qui peut néanmoins bénéficier de toutes les structures accélératrices spatiales afin d'obtenir de très bonnes performances. Elle est de plus facilement extensible à de nouvelles classes, si elle utilise une structure accélératrice, ou si l'espace des réponses à été borné. La figure 2.16 représente une classification de l'espace image à partir de trois classes dans le jeu d'apprentissage. Ici, un vote majoritaire a été utilisé, en tenant compte des 3 voisins les plus proches. Notons que le nombres de voisins à considérer n'est généralement pas corrélé au nombre de classes.

La classification par plus proches voisins a été mise en œuvre avec une métrique Euclidienne par Batra et al. [16], Blank et al. [23]. Bobick et Davis [24] ont utilisé les moments de Hu pour reconnaitre des actions, avec une distance de Mahalanobis. Des réductions de dimensions ont été appliqués afin de simplifier l'espace des caractéristiques (Turaga et al. [132], Wang et Suter [139])

La classification par plus proches voisins s'est révélée très efficace lors d'utilisation de poses clés ou d'*examplars*. Citons les travaux de Lin et al. [71], Sullivan et Carlsson [124], Weinland et Boyer [143], Weinland et al. [145].

2.6.3 Modèles Markoviens

Les Modèles de Markov à États Cachés, *Hidden Markov Models* (HMM) font partis des modèles génératifs les plus utilisés pour reconnaitre des actions. La mesure se fait par le biais d'états cachés, c'est-à-dire que l'on ne mesure pas l'état dans lequel le modèle se trouve, mais un état différent (caché) qui renseigne sur l'état courant dans lequel le système doit être. Pour cela, deux mesures sont faites : (*i*) la probabilité de transition entre deux états dans le système de Markov (*ii*) la probabilité d'observation d'un état, par l'intermédiaire de l'état caché. Afin de rendre le système calculable en pratique, deux hypothèses sont faites : (*i*) la dépendance d'un état, au niveau temporel ne dépend que de l'état précédent — modèle de Markov de niveau 1 — ou des n états précédents — modèle de Markov de niveau n. Cette hypothèse d'indépendance des états par rapport aux précédents est appelé Hypothèse de Markov. (*ii*) les observations ne dépendent que de l'état courant : chaque observation est indépendante. Pour les détails sur les modèles de Markov, voir l'annexe B.

Les HMM ont été utilisés dans de nombreuses méthodes de reconnaissance d'actions, pour modéliser la dépendance temporelle le plus souvent. Généralement les auteurs utilisent l'algorithme de Baum-Welch Baum et al. [17], Welch [148] pour entrainer le système, et l'algorithme de Viterbi Viterbi [137] pour déterminer la séquence la plus probable à partir des observations. Lorsque l'on utilise un HMM par action, reconnaitre cette action revient à trouver la chaîne de Markov qui génère la séquence observée avec la plus grande probabilité. Yamato et al. [154] ont utilisés les HMM sur des silhouettes pour catégoriser des actions de tennis. Feng et Perona [39] ont utilisés des poses clés comme états de leur système de Markov. Ils sont en mesure d'entrainer rapidement leur système grâce à des observations simples, mais perdent en flexibilité. Weinland et al. [144] ont eux aussi construit un dictionnaire compact à l'aide de systèmes de Markov. De plus, ils ont conditionnés leurs observations au point de vue afin de proposer une reconnaissance indépendante de celui-ci. De façon similaire, Lv et Nevatia [77] ont utilisés les points de vue et les poses clés afin de construire un réseaux d'actions encodant explicitement les transitions poses clés $\Leftrightarrow$ point de vue. Ahmad et Lee [2] ont proposé une approche hiérarchique des systèmes de Markov pour répondre au problème des observations de multiples points de vue.

Au lieu d'utiliser un HMM par action, certains auteurs ont proposés d'utiliser un HMM par partie du corps suivie. Cette solution a l'avantage de faciliter le processus d'entrainement, car la dynamique apprise l'est par articulation, et non plus pour le corps en entier. İkizler et Forsyth [50] ont utilisé un squelette 3D pour leurs observations. Les bras et les jambes sont traités par deux systèmes indépendants. Ils ont liés les états des différents modèles s'ils partageaient une probabilité d'émission, permettant ainsi une segmentation de l'action. Cette approche par

HMM indépendant a été étendu par Chakraborty et al. [27] a d'autres parties du corps, obtenues par des détecteurs dépendants du point de vue.

De nombreuses autres variations autour des modèles de Markov ont été proposés, comme des factorisations d'états (Peursum et al. [101]), l'utilisation d'hypothèse d'ordre de Markov de longueur variable (Caillette et al. [25]), des transitions hiérarchiques variables (Natarajan et Nevatia [90]), ou l'utilisation de classifieurs faibles et d'algorithme de *boosting* comme *AdaBoost* (Lv et Nevatia [76]). Toutes ces méthodes ont montré la validité des modèles génératifs de type Markovien pour la reconnaissance d'actions, mais aussi l'importance fondamentale que revêt la temporalité dans une action.

2.6.4 Autres modèles

D'autres modèles ont été utilisés pour la reconnaissance d'actions. Citons notamment les grammaires formelles qui modélisent l'ordre dans lequel les parties d'une actions doivent être observées. Cette méthode a été mise en œuvre avec succès par Ogale et al. [93], Turaga et al. [133].

Des modèles discriminants ont aussi été mis en œuvre, afin de prendre en compte la dépendance temporelle des observations. Pour cela, les auteurs ont principalement utilisé des *Conditional Random Fields* (CRF) ou des variantes des CRF. Citons notamment les travaux de Lafferty et al. [64], Mendoza et Pérez de la Blanca [82], Sminchisescu et al. [123], Wang et Suter [140]. Les détails de ses travaux sont disponibles dans l'état de l'art présenté par Poppe [103].

2.7 Comparaisons de l'état de l'art

Nous avons synthétisé l'ensemble des approches de l'état de l'art dans le tableau 2.2. Celui-ci reprend la décomposition précédente des travaux afin de résumer les caractéristiques principales des différentes approches en fonction du contexte que nous avons défini. Ainsi, l'utilisation de primitives 2D ou 3D est mise en avant, de même que les possibilités d'évolution de chacune des méthodes. Nous avons aussi reprécisé les caractéristiques principales en apprentissage et reconnaissance.

Approche	Référence	2D/3D	Méthodologie	Évolution
2.2.1 Points spatiaux	Chomat et Crowley [30]	2D	Gabor	✓
	Zelnik-Manor et Irani [159]	2D	Multi-échelle + *N-cut*	×
	Blank et al. [23]	2D	Poisson + *kNN*	✓
	Efros et al. [35]	2D+t	Flux optique + *kNN*	×
	Jhuang et al. [53]	2D+t	Gabor + SVM	×
	Klaeser et al. [58]	2D + t	Gradient + SVM	×
	Willems et al. [149]	2D + t	STIP + SVM	✓
	Patron-perez et al. [97]	2D	HOG + SVM	×
2.2.2 Points éparses	Laptev et Lindeberg [66]	2D+t	Harris 3D	✓
	Schuldt et al. [117]	2D+t	Harris 3D + SVM	×
	Dollar et al. [34]	2D+t	Cuboïd	✓
	Niebles et al. [92]	2D+t	Cuboïd + pLSA	×
	Oikonomopoulos et al. [94]	2D+t	Kadir + RVM	×
	Scovanner et al. [118]	2D+t	SIFT 3D	✓
2.2.3 Points spatio-temporels	Wong et al. [151]	2D+t	pLSA Spatio-temporel	×
	Liu et Shah [74]	2D+t	Grille corrélation	×
	Laptev et al. [67]	2D+t	Grille corrélation	×
	Ryoo et Aggarwal [112]	2D+t	*STR Match*	✓
	Niebles et al. [91]	2D+t	Harris 3D + Motifs	✓
	Ryoo [111]	2D+t	*BoW* + histogrammes	✓
2.3.1 Silhouette	Bobick et Davis [24]	2D	Motifs Spatio-temporel	✓
	Shechtman et Irani [120]	2D	Patchs	✓
	Elgammal et al. [36]	2D	*Examplar* + Markov	✓✓
	Ahmad et Lee [2]	2D+t	*Motion History* + SVM	×
	Huang et Trivedi [49]	3D+t	Histogrammes + Markov	✓
	Weinland et al. [144]	∼ 3D	*Examplar*	×
	Xiong et Liu [152]	2D	Markov	✓
	Yilmaz et Shah [158]	2D+t→ 3D	Points critiques	✓
	Ke et al. [57]	2D+t	Segmentation + motifs	✓
	Shao et Chen [119]	2D	*BoW* + SRDA	✓
	Tseng et al. [130]	2D	*k*NN	✓

Approche	Référence	2D/3D	Méthodologie	Évolution
2.3.2 Squelette	Gavrila et Davis [43]	3D	DTW	✓
	Yacoob et Black [153]	2D	*Examplar* + PCA	✓
	Fujiyoshi et Lipton [42]	2D	Motifs + cycle	✓
	Ali et al. [3]	2D/3D	Points critiques + kNN	✓
	Ziaeefard et Ebrahimnezhad [161]	2D	Histogrammes + SVM	×
	Tran et al. [129]	2D	Histogrammes + SVM	×
	Parameswaran et Chellappa [96]	3D	Points critiques	✓
	Ramanan et Forsyth [104]	2D+3D	Articulation + HMM	×
	Lv et Nevatia [77]	2D	PMK	✓
	Sheikh et al. [121]	3D	*Examplar* + angles	✓
	Rao et Shah [105]	2D	Motif + pics	✓✓
	Cuntoor et al. [32]	2D	Trajectoires + Markov	×
	Li et Fukui [70]	3D	Trajectoires + Markov	×
	Baak et al. [7]	3D	Motif + contraintes	×
	Han et al. [47]	3D	Trajectoires + SVM	×
	Raptis et al. [106]	3D	Trajectoires + Classifieur	×
	Yao et al. [156]	3D	*Priors* + GPLVM	×
	Yao et al. [155]	2D+3D	Vitesse + Points critiques	×
	Wang et al. [138]	3D+Z	Articulations + Points critiques	×
2.4 Animation	Barbič et al. [9]	3D	PCA + GMM	×
	Beaudoin et al. [18]	3D	Motifs + graphe	×
	Müller et al. [87]	3D	Motifs + poses clés	×
2.5 Interactions	Ivanov et Bobick [52]	2D	Grammaire + Markov	✓
	Zhang et al. [160]	3D	Marqueur	✓
	Van den Bergh et al. [134]	3D	Détection + suivi	✓
	Argyros et Lourakis [6]	3D	Stéréo-vision	✓
	Li et al. [69]	3D	Stéréo-vision	✓
	Okada et Stenger [95]	3D	Hiérarchie + MoCap	×

TABLE 2.2 – Représentation des caractéristiques importantes des méthodes présentées dans l'état de l'art. Pour chaque approche, une mesure qualitative de l'évolution possible est présentée. Elle met en valeur la facilité d'extension de la méthode, principalement en fonction de sa méthodologie d'apprentissage.

2.8 Problématique et contexte

2.8.1 Verrous

De l'état de l'art précédent, nous pouvons tirer les conclusions suivantes :

(i) il n'y a pas de méthode exploitant pleinement les primitives 3D disponibles ;

(ii) les méthodes 2D sont pénalisées par des ambiguïtés visuelles dues aux images ;

(iii) les solutions apportées en 2D sont coûteuses en temps de calculs ;

(iv) les solutions 2D nécessitent une phase importante d'apprentissage de même qu'une reconnaissance complexe en terme de calcul ;

(v) les interactions Homme–Machine doivent apporter de la souplesse à la fois à la reconnaissance, mais aussi à l'apprentissage.

2.8.2 Approches proposées

À partir des verrous que nous avons identifiés précédemment, nous avons proposés deux contributions principales reposant sur l'analyse du squelette du personnage. En effet, la capture du mouvement d'un personnage — sous forme d'un ensemble de points tridimensionnels décrivant ses articulations — fourni les primitives de haut niveau sur le mouvement d'un être humain. Nous pouvons ainsi étudier les trajectoires des membres, de même que les « attitudes » du personnage que nous suivons. Nous voulons fournir une méthode qui soit aussi souple à l'apprentissage de nouvelles actions. Pour cela, nous avons utilisé le paradigme des *Examplars* pour l'apprentissage d'une nouvelle action. C'est-à-dire un apprentissage par des exemples plutôt que par une analyse statistique des données d'apprentissage comme c'est souvent le cas en apprentissage artificiel. Ces deux approches sont à la base de nos deux contributions. L'analyse des trajectoires fournissant une focalisation sur « l'intention gestuelle de l'utilisateur » alors que l'analyse des attitudes, que nous appellerons : poses, permet d'identifier les configurations fréquentes ainsi que les configurations discriminantes. L'analyse des trajectoires des membres du squelette couplée à un automate de reconnaissance permet de décomposer l'action en un ensemble de séries temporelles. Cela permet d'être « au plus près » de l'action. Alors que l'analyse des fréquences des poses permet d'être « un peu plus loin » de l'action à reconnaitre. Ces deux méthodes apportent des réponses complémentaires en fonction du contexte applicatif mais aussi des caractéristiques intrinsèques des actions.

Dans l'état de l'art précédent, les utilisateurs étaient tributaires du système, celui-ci n'ayant aucune souplesse lui permettant d'évoluer en fonction de la volonté des utilisateurs. Seuls les spécialistes pouvaient apprendre de nouvelles actions, par une méthodologie d'apprentissage complexe, nécessitant de nombreuses instances

d'une même action pour permettre sa reconnaissance ainsi qu'une réinitialisation de l'apprentissage avec les nouvelles données. Grâce à l'approche *Examplar* que nous proposons, les utilisateurs peuvent prendre en main leurs interactions, et faire évoluer le système « en lui montrant » de nouvelles actions une seule fois.

Chapitre 3

Représentation du mouvement 3D

Sommaire

3.1 Éléments de base

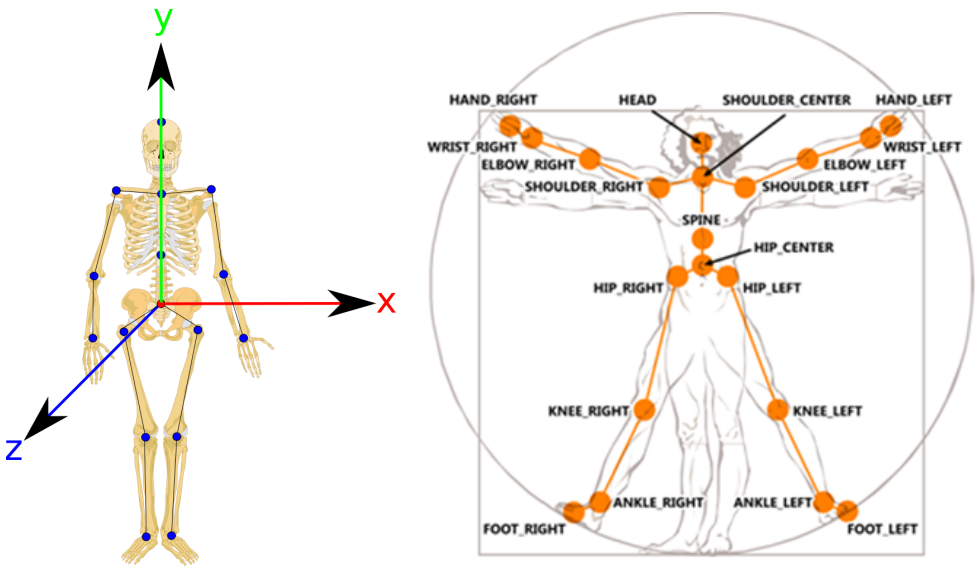
L'étude du mouvement à partir de l'acquisition d'images a été initiée à la fin du 19ème siècle par Muybridge [89] et Marey [78], grâce à l'arrivée de la photographie.

Dès lors que l'on a été capable d'enregistrer des mouvements, les chercheurs se sont intéressés à la composition du mouvement au-delà de ce qu'un œil humain peut percevoir. La photographie fournissant un moyen visuel de « figer » les différentes étapes du mouvement pour pouvoir les analyser *a posteriori*. Il s'agissait principalement de travaux photographiant des mouvements rapides afin d'analyser manuellement leur composition. En 1973, le psychologue Johansson [54] a publié une étude sur la perception du geste par le mouvement. Il a attaché des lumières aux articulations d'un corps humain, afin d'étudier leurs trajectoires. En effet, sans mouvement, il est très difficile d'estimer l'action d'un être humain à partir uniquement de lumière sur ses articulations. Lorsque le sujet est en mouvement, l'action qu'il est en train de faire devient compréhensible même si on ne la représente que par des points lumineux placés sur les articulations. Johansson a ainsi montré que la composante temporelle était fondamentale à un être humain pour la perception d'une action. Son modèle de « capture » des actions — par l'intermédiaire de marqueurs sur les articulations principales — a ensuite inspiré de très nombreux travaux en capture automatique de mouvements. Avec l'arrivée de l'imagerie numérique les travaux sur la capture de mouvements ont connu un réel succès.

Dans un système d'interprétation d'actions, à partir de capture de mouvements, les données utilisées en entrée sont issues des systèmes de « capture de mouvements (MoCap) ». Ceux-ci s'appuient souvent sur une structure hiérarchique simplifiée d'un squelette humain réel, voir la figure 3.1. On appelle « articulations » les points suivis au cours du temps et associés à des degrés de libertés, réels ou virtuels, d'un corps humain. De même, on appellera « os » les segments rigides entres les « articulations ». Chaque extraction du mouvement est appelée pose. Il s'agit d'une configuration du squelette dans une attitude déterminée par le système de capture de mouvements. Dans ce chapitre nous allons présenter la capture de mouvements 3D, c'est-à-dire que les articulations auront des coordonnées tridimensionnelles. Les articulations sont calculées à partir des systèmes 3D, comme un système à base d'exosquelette, voir figure 3.2 pour des exemples, ou depuis des informations 2D fusionnées, systèmes multivues, avec ou sans marqueurs, voir la figure 3.3.

Parmi les systèmes les plus répandus, citons celui de la société Vicon [136], représenté dans la figure 3.3. Il s'agit d'un système à base de marqueurs, très utilisé dans le cinéma ou les jeux vidéo, qui offre une grande précision, mais nécessite un post-traitement long afin d'estimer les positions des articulations. Le sujet porte des marqueurs, pastilles réfléchissantes infrarouges. Il est filmé par un ensemble de caméras sensible à la longueur d'onde des marqueurs. Le réseau de caméras est calibré, et la fusion des informations de position des marqueurs permet d'estimer la position tridimensionnelle des articulations du corps humain.

La tendance actuelle est de s'affranchir des marqueurs. Certaines approches

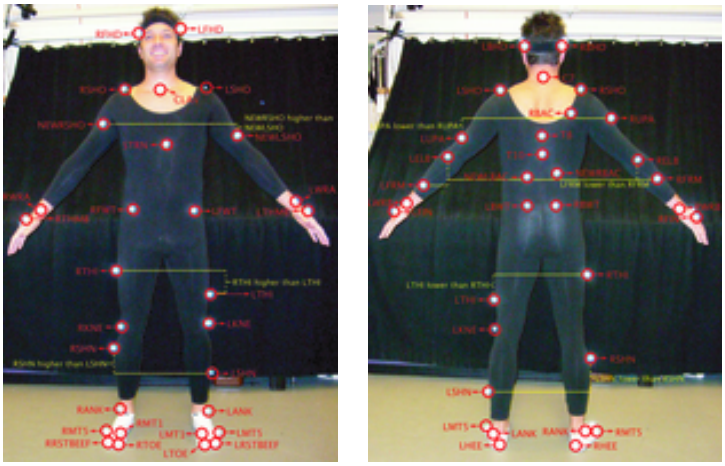


(a) Modèle générique (b) Modèle issue de Wang et al. [138]

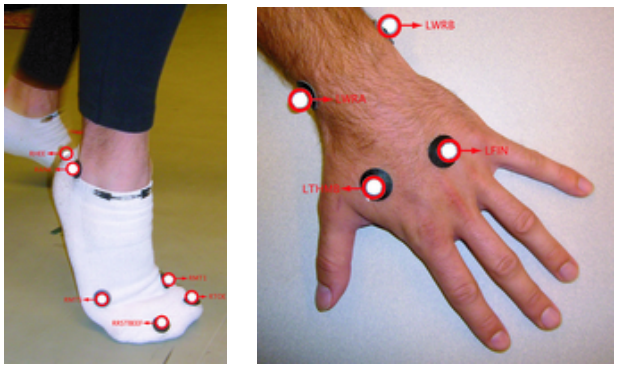
FIGURE 3.1 – Le squelette humain et une superposition d'un squelette simplifié de Capture de Mouvement, représentant les articulations par des points bleus ou oranges.

(a) Exo-squelette mécanique (b) Exosquelette magnétique

FIGURE 3.2 – Exemples d'exosquelette de capture de mouvements. (a) un exosquelette mécanique (système Gypsy^{TM}). (b) un exosquelette magnétique (©Lord of The Rings).



(a) Vue de face (b) Vue de dos



(c) Détail d'un pied (d) Détail d'une main

FIGURE 3.3 – Capture de mouvement avec marqueurs (MoCap CMU [84]).

sont basées sur des solutions multivues, comme Michoud et al. [83], en utilisant des webcams grand public. D'autres sur du matériel spécifique, caméra et projecteur infrarouge, comme la Kinect^{TM}de Microsoft^{TM} pour Shotton et al. [122]. Ces périphériques ont suscité l'engouement des utilisateurs pour la reconnaissance de mouvements dans la vie quotidienne. Les figures 3.4 et 3.5 représentent des exemples d'acquisition par le procédé de Shotton et al. [122], dans des situations domestiques.

Dans notre contexte d'interprétation d'actions à partir de données de capture de mouvements, il nous semble important de préciser les différents concepts issus de la capture. Nous allons définir les poses et actions (section 3.2), les trajectoires des membres (section 3.3) et le repère utilisateur mis en place, appelé auto-centrisme (section 3.4).

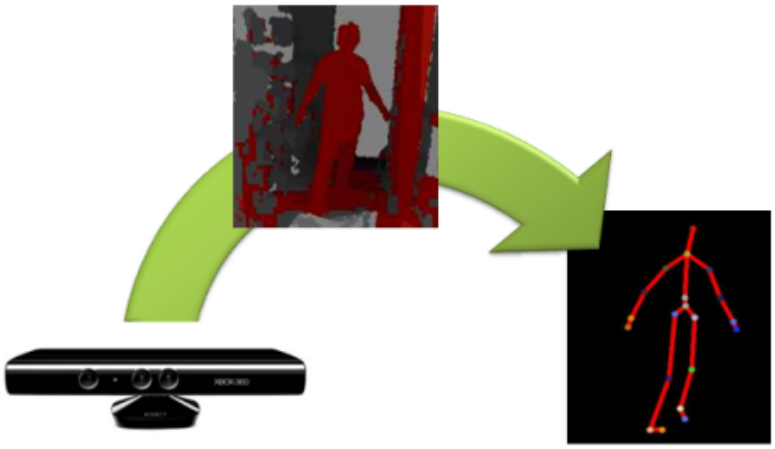


FIGURE 3.4 – Capture de mouvements sans marqueur avec un Kinect™.

3.2 Poses et Actions

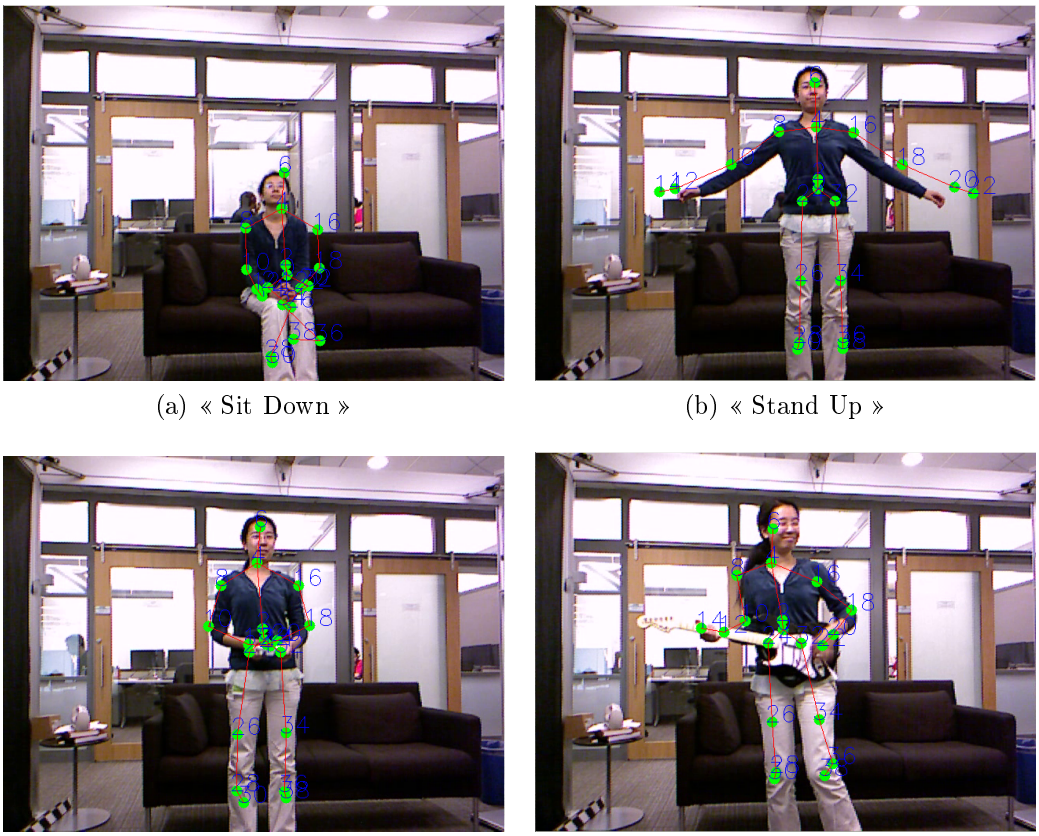
Afin de présenter les différents aspects mis en œuvre dans la capture de mouvements, nous allons suivre la décomposition proposée par Picard [102] dans sa thèse de doctorat. Le geste est défini comme un mouvement véhiculant un sens, une signification. La différence vient de la volonté consciente de l'être humain qui effectue volontairement un geste, contrairement à un système mécanique. Le geste est contextualisé et dépendant de la culture du sujet. Nous définissons la captation de geste par le procédé informatique qui acquière des gestes et les représente sous forme de poses.

Définition 1 Articulation**.** *Une articulation, a, est un point tridimensionnel donné dans un repère de référence.*

$$a = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{3.1}$$

À chaque pas de temps, la position d'un ensemble d'articulations est captée par une observation de la scène. Cette captation peut reposer sur un ensemble de connaissances de la scène, du personnage, *etc.* tel que l'a défini Picard [102] : « la captation est un processus de caractérisation dynamique de la gestuelle d'une entité d'intérêt (un homme, un animal, un objet) ».

La figure 3.1 représente différentes schématisations applicables sur un squelette humain réel.



(a) « Sit Down »

(b) « Stand Up »

(c) « Play Game »

(d) « Play Guitar »

FIGURE 3.5 – Exemples d'actions capturées avec un Kinect™ Wang et al. [138].

Définition 2 Pose. *Une pose est la position tridimensionnelle, de l'ensemble des articulations données, à un instant t. Formellement, nous définissons une pose comme l'union des positions des articulations pour un même temps, définie par l'équation 3.2.*

$$P(t) = \bigcup_{a \in \mathcal{A}} a(t) = \bigcup_{a \in \mathcal{A}} \begin{bmatrix} x_a(t) \\ y_a(t) \\ z_a(t) \end{bmatrix} \tag{3.2}$$

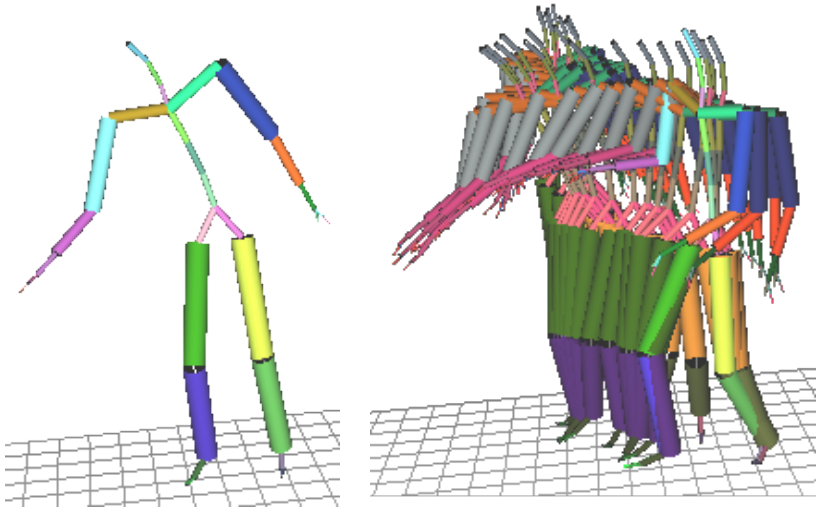
Où $P(t)$ est la pose à l'instant t, $x_a(t)$, respectivement $y_a(t)$ et $z_a(t)$, est la coordonnée sur l'axe X, respectivement Y et Z, au temps t ; $\mathcal{A}$ est l'ensemble des articulations suivies, et $a(t)$ l'articulation à un instant t.

Le premier point 3D d'une pose est généralement la racine, ou *root*. Il s'agit d'un point virtuel ou réel, assimilé au pelvis dans ce cas, qui représente l'origine de la structure hiérarchique du squelette. Lorsque ce point est virtuel, il peut être vu comme une approximation du centre de gravité. Dans le domaine de l'animation par ordinateur, il sert aussi de point de référence pour la position du personnage. Il s'agit du point d'origine du repère local du personnage capté. Nous montrerons aussi en section 3.4, que ce point est fondamental dans bien des approches d'invariance à la localisation du personnage.

Nous pouvons expliciter une pose à partir de l'équation 3.2, sous forme d'un vecteur de dimension $|\mathcal{A}| \times 3$:

$$P(t) = \begin{bmatrix} a_1(t) \\ a_2(t) \\ \vdots \\ a_{|\mathcal{A}|}(t) \end{bmatrix} = \begin{bmatrix} x_1(t) \\ y_1(t) \\ z_1(t) \\ \vdots \\ x_{|\mathcal{A}|}(t) \\ y_{|\mathcal{A}|}(t) \\ z_{|\mathcal{A}|}(t) \end{bmatrix} \tag{3.3}$$

Une action est une suite temporelle de poses partageant une sémantique commune, c'est-à-dire « un mouvement véhiculant un sens »(Picard [102]), par exemple les poses de la marche. Une action sera donc définie par sa suite ordonnée de poses, commençant à t_0 et sera de longueur t_N, où N est le nombre de poses. De plus, généralement on confondra l'étiquette donnée à l'action, « marche », et son nom, « A ». On parlera alors de l'action « marche » au lieu de l'action d'étiquette « marche ».



(a) Une pose (b) Une action

FIGURE 3.6 – Une pose et l'action à laquelle elle se rapporte.

À partir de l'équation 3.3, on définira une action comme une matrice de poses :

$$A = \begin{bmatrix} a_1(t_0) & \dots & a_1(t_N) \\ a_2(t_0) & \dots & a_2(t_N) \\ \vdots & \ddots & \vdots \\ a_{|\mathcal{A}|}(t_0) & \dots & a_{|\mathcal{A}|}(t_N) \end{bmatrix} = \begin{bmatrix} x_1(t_0) & x_1(t_1) & \dots & x_1(t_N) \\ y_1(t_0) & y_1(t_1) & \dots & y_1(t_N) \\ z_1(t_0) & z_1(t_1) & \dots & z_1(t_N) \\ \vdots & \vdots & \ddots & \vdots \\ x_{|\mathcal{A}|}(t_0) & x_{|\mathcal{A}|}(t_1) & \dots & x_{|\mathcal{A}|}(t_N) \\ y_{|\mathcal{A}|}(t_0) & y_{|\mathcal{A}|}(t_1) & \dots & y_{|\mathcal{A}|}(t_N) \\ z_{|\mathcal{A}|}(t_0) & z_{|\mathcal{A}|}(t_1) & \dots & z_{|\mathcal{A}|}(t_N) \end{bmatrix} \tag{3.4}$$

Il est aussi possible d'exprimer une pose, à partir de la définition initiale, équation 3.2, de façon plus compacte, par la formulation suivante :

$$A = \bigcup_{t=t_0}^{t_N} P(t) \tag{3.5}$$

Sans nuire à la généralité, dans la plupart des cas, on peut considérer que $t_0 = 0$, *i.e.* l'action commence au temps $t = 0$.

À partir de la représentation 3.4, nous pouvons mettre en avant chacune des

parties d'une pose, au sein de la représentation d'une action :

$$\mathrm{A} = \underbrace{\begin{bmatrix} x_1(t_0) \\ y_1(t_0) \\ z_1(t_0) \\ x_2(t_0) \\ y_1(t_0) \\ z_1(t_0) \\ \vdots \\ x_{|\mathcal{A}|}(t_0) \\ y_{|\mathcal{A}|}(t_0) \\ z_{|\mathcal{A}|}(t_0) \end{bmatrix}}_{\text{Pose 1}} \underbrace{\begin{bmatrix} x_1(t_1) \\ y_1(t_1) \\ z_1(t_1) \\ x_2(t_1) \\ y_1(t_1) \\ z_1(t_1) \\ \vdots \\ x_{|\mathcal{A}|}(t_1) \\ y_{|\mathcal{A}|}(t_1) \\ z_{|\mathcal{A}|}(t_1) \end{bmatrix}}_{\text{Pose 2}} \cdots \underbrace{\begin{bmatrix} x_1(t_N) \\ y_1(t_N) \\ z_1(t_N) \\ x_2(t_N) \\ y_1(t_N) \\ z_1(t_N) \\ \vdots \\ x_{|\mathcal{A}|}(t_N) \\ y_{|\mathcal{A}|}(t_N) \\ z_{|\mathcal{A}|}(t_N) \end{bmatrix}}_{\text{Pose } N} \begin{matrix} \} \text{Racine} \\ \\ \\ \} 1^{\text{ère}} \text{Articulation} \\ \\ \\ \\ \\ \} |\mathcal{A}|^{\text{ème}} \text{Articulation} \end{matrix} \tag{3.6}$$

3.3 Trajectoires

De même que l'on peut s'intéresser aux positions des articulations, il est possible de regarder leurs trajectoires au cours du temps. Pour cela, on peut remarquer que les lignes de la représentation matricielle d'une action (équation 3.4) sont l'expression des trajectoires des articulations, sur chacun des axes du monde tridimensionnel. Il s'agit de la position des articulations par rapport au temps. Notre matrice, c'est-à-dire une action, est un faisceau de trajectoires. Dès lors, il est possible de voir une action comme un ensemble de trajectoires tridimensionnelles ou monodimensionnelles.

$$A = \bigcup_{t=t_0}^{t_N} \bigcup_{a \in \mathcal{A}} a(t) = \bigcup_{t=t_0}^{t_N} \bigcup_{a \in \mathcal{A}} \begin{bmatrix} x_a(t) \\ y_a(t) \\ z_a(t) \end{bmatrix} \tag{3.7}$$

$a(t)$ est la trajectoire tridimensionnelle de l'articulation a, au cours du temps. $x_a(t)$, respectivement $y_a(t)$ et $z_a(t)$, est la trajectoire de l'articulation a sur l'axe des X, respectivement l'axe des Y et l'axe des Z. La trajectoire d'une action, bien qu'étant potentiellement une simplification de l'action, est porteuse de plus d'information sur l'action, qu'une pose. Il s'agit de la transposition informatique du postulat de Johansson [54], voir la figure 3.7.

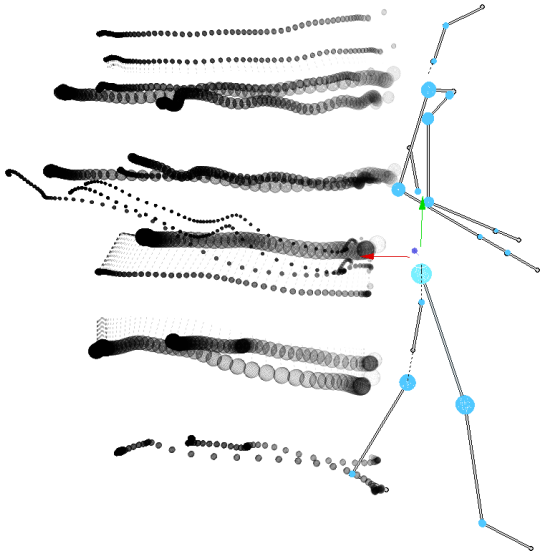


FIGURE 3.7 – Trajectoires des articulations d'un squelette d'animation.

3.4 Auto-centrisme

Il s'agit de centrer les actions par rapport à la capture de mouvements. En effet, lorsque l'on souhaite comparer des actions, il est nécessaire d'avoir un référentiel commun à toutes les exécutions, qu'elles concernent la même action (même étiquette) ou des actions différentes. Nos comparaisons étant essentiellement géométriques, obtenir un référentiel commun n'est pas des plus évidents. De plus, on souhaite que deux actions, géométriquement identiques, mais effectuées dans des lieux différents, soit considérées comme équivalentes. Pour cela, la solution la plus adaptée est de représenter les actions par rapport à l'utilisateur, c'est-à-dire son squelette dans notre cas.

Nous utilisons la racine du squelette comme point de référence. En effet, celle-ci peut être associée au centre de gravité du personnage, ou tout du moins, à une position centrale intéressante. Dès lors, chacune des articulations va être rapportée à ce point 3D. Or, comme il s'agit d'une structure hiérarchique, cela se traduira par une translation des articulations aux temps $t > t_0$ par le vecteur constitué de la racine au temps t, $a_1(t)$ moins la racine au temps t_0, $a_1(t_0)$, L'action auto-centrée, notée A^c, sera exprimée par :

$$A^c = \begin{bmatrix} a_1(t_0) - a_1(t_0) & \dots & a_1(t_N) - a_1(t_0) \\ a_2(t_0) - a_1(t_0) & \dots & a_1(t_N) - a_1(t_0) \\ \vdots & \ddots & \vdots \\ a_{|\mathcal{A}|}(t_0) - a_1(t_0) & \dots & a_{|\mathcal{A}|}(t_N) - a_1(t_0) \end{bmatrix} \tag{3.8}$$

La première articulation du squelette (racine) peut être vue comme le repère local de l'action. Ce repère, constant pour les autres articulations, au cours du temps, permet de positionner le geste par rapport à l'utilisateur. Ce recentrage de l'action est nécessaire mais pas suffisant. Dans la reconnaissance d'actions, différentes personnes peuvent faire les mêmes actions. Il faut donc normaliser la taille de l'individu observé. Dans la plupart des systèmes de capture de mouvements avec marqueurs, cette étape est requise, et faite *a posteriori*. De fait, les formats usuels de gestion de données de capture de mouvements incorporent cette information (BVH, ASF, C3D *etc.*). Dans ce cas, il est aisé d'appliquer le facteur d'échelle. Dans un cas de capture de mouvements sans marqueur et temps réel, comme le système Kinect^{TM}, il n'est pas aisé de déterminer la taille de l'acteur. Dans ces utilisations, un facteur de normalisation doit être appliqué. Il consiste généralement en la plus grande taille admise par le système. Il peut être appliqué sans dommage pour la plupart des méthodes. L'action ainsi normée, notée A^n, sera définie par :

$$A^n = \frac{1}{\alpha} \times A^c \tag{3.9}$$

Où α est la taille de la personne, ou un facteur supérieur à la plus grande taille admise par le système.

Deuxième partie

Approches par trajectoires

Chapitre 4

Analyse des trajectoires

Sommaire

Lorsque l'on s'intéresse aux actions humaines, l'une des approches les plus naturelles est l'analyse des mouvements des membres. Cette analyse entre dans le cadre plus général de l'analyse des trajectoires, que ce soit les trajectoires des membres ou la trajectoire du corps dans son ensemble. Dans cette catégorie, on s'intéresse principalement à des notions de vitesse et accélération (changement intrinsèque) mais aussi aux directions et sens (changement extrinsèque). Les changements dans la géométrie du mouvement sont aussi importants que les changements intrinsèques. En effet, ce sont souvent eux qui sont le plus discriminant, car si un humain est capable d'appréhender facilement une direction — il boxera à peu près toujours de la même façon — il n'est pas en mesure d'évaluer la vitesse de son geste, et celle-ci peut changer de manière importante d'une exécution à l'autre. Nous cherchons à apporter de la souplesse à la reconnaissance d'action, en nous plaçant dans un contexte de reconnaissance usuelle d'action. L'action « boxe » devra être étiquetée de façon identique quel que soit l'intensité du geste ou la hauteur du bras. Nous ne cherchons pas à évaluer des performances sportives. En effet, un sportif de haut niveau va chercher une « perfection » dans la reproduction de son geste. Dans ce contexte, une méthode qui serait « tolérante » à son exécution ne serait pas adaptée.

4.1 Modélisation des trajectoires

Nous avons fait le choix dans cette partie de nous intéresser à la segmentation d'une action en éléments atomiques. Pour cela, nous allons mettre en avant les changements drastiques dans les trajectoires d'une action. En particulier, et grâce à notre approche tridimensionnelle du problème, nous sommes en mesure d'appréhender les variations de trajectoires de chacun des membres du squelette rigide d'animation d'un acteur. Nous exprimons les trajectoires sous forme de segments de droites, projetés sur chacun des axes, du repère général de la scène, à l'aide de régressions linéaires. Ainsi, il nous est possible de décomposer les mouvements en un ensemble d'éléments successifs et appréhender au mieux les variations tant internes aux actions, qu'externes à celle-ci. Notons toutefois, que dans la suite de ce chapitre, nous utiliserons le concept d'auto-centrisme, tel que nous l'avons défini en section 3.4 du chapitre 3.

Dans une grande partie des solutions proposées précédemment, la modélisation des trajectoires des articulations se fait à l'aide de courbes de degrés importants (polynômes de degrés ≥ 3 ou des splines). Ces modélisations permettent d'avoir une modélisation précise du mouvement. Néanmoins, la précision des approximations n'est pas toujours synonyme de précision dans la reconnaissance. En effet, à l'aide d'une réduction linéaire des trajectoires, il est possible d'avoir une approche tout aussi efficace pour détecter les changements « drastiques » tout en étant plus efficace en temps de calcul. La figure 4.1 représente une réduction linéaire d'un signal continu « quelconque » en un ensemble de droite. Le processus de régression peut être calculé de façon incrémentale, comme nous allons le montrer par la suite. Les changements drastiques sont souvent présents dans les zones de changement de courbure ou dans les extrema locaux. Ces points sont importants pour la représentation du mouvement, et bien des solutions les utilisent comme des caractéristiques des actions. Comme nous l'avons souligné dans l'état de l'art, §2.2, les auteurs regroupent ces points dans des classes en rapport avec l'action observée. Diverses méthodes de *Machine Learning* ont été mise en œuvre pour discriminer les actions en fonction de ses caractéristiques. Néanmoins, ces approches ont l'inconvénient de supprimer les informations de vitesse et d'accélération intrinsèques au mouvement étudié. Notre approche, à base de régressions linéaires, permet de conserver les informations sur les changements géométriques, de même que sur les changements de direction et de vitesse.

4.1.1 Régression linéaire

Nous proposons de réduire les trajectoires à des segments de droites. Pour cela, nous allons utiliser un estimateur classique dans la régression de données,

FIGURE 4.1 – Régression linéaire d'un signal continu « quelconque ».

l'estimateur des moindres carrées. Cet estimateur est défini de la façon suivante :

$$\widehat{\beta}^n = \underset{\beta_0,\beta_1}{\arg\min} \sum_{i=1}^{n} (y_i - \beta_1 \cdot x_i - \beta_0)^2 \tag{4.1}$$

Où : x_i est la coordonnée temporelle dans notre cas, y_i est la coordonnée spatiale.

Il s'agit d'un estimateur qui peut être évalué de façon incrémentale. Cela nous donne un critère sur l'erreur admise pour notre régression linéaire. En effet, la minimisation de cet estimateur nous donne la « meilleure » approximation — sous la forme d'une droite — pour un ensemble de points donnés, au sens de l'erreur des moindres carrés.

Règle 1. *Un point, P_{n+1}, est ajouté à la droite D, représenté par la régression linéaire $R_n = \begin{bmatrix} P_0 & P_1 & \cdots & P_n \end{bmatrix}$ si et seulement si $\widehat{\beta}^{n+1} \leq \varepsilon$. Dans le cas contraire, $\widehat{\beta}^{n+1}$ défini l'estimateur, à minimiser, d'une nouvelle régression linéaire incrémentale, et P_{n+1} en est le point initial, $R_{n+1} = \begin{bmatrix} P_{n+1} \end{bmatrix}$.*

La modélisation d'une trajectoire sous la forme de segments de droites se fait par la minimisation d'une erreur. Cette erreur, représenté par ε, indique la distance cumulée de chacun des points à la droite calculée. Comme nous ne cherchons pas une seule courbe pour modéliser la trajectoire d'un membre, l'erreur e doit être bornée. Typiquement, $\varepsilon \in [0; e]$. La valeur de e ne peut être obtenue qu'à partir d'une analyse empirique. Néanmoins, afin de ne pas favoriser une « sur segmentation » des trajectoires, il peut être nécessaire d'ajouter un critère sur le nombre minimum de points d'observation à ajouter à une droite. Quel que soit la trajectoire, deux points produirons toujours une erreur nulle (axiome d'Euclide) et donc ne sera pas une modélisation représentative de la trajectoire d'une articulation. Ce nombre de point, ainsi que l'erreur maximale admissible seront dépendants du contexte et de la variation des actions à apprendre.

En considérant que l'on dispose d'un nombre suffisant de points dans notre régression, nous ne sommes pas à l'abri de points aberrants (*outliers*). Ces observations atypiques sont inévitables lorsque l'on travaille avec des systèmes électroniques. En effet, le simple bruit des capteurs d'acquisition, quels qu'ils soient, produit de l'invraisemblance dans les résultats. La plupart des méthodes de capture de mouvements ne peuvent pas garantir une suppression de ses données incohérentes, qui plus est en temps réel ! Dès lors, l'influence des *outliers* doit être prise en considération dans notre analyse des trajectoires. Une solution naturelle est l'emploi d'un estimateur robuste, comme la régression *biweight* ou les moindres valeurs absolues. Or, ces méthodes, bien qu'efficaces, sont itératives, et donc coûteuse en temps de calcul. Leur principe général suit l'algorithme 4.1.

Algorithme 4.1: Régressions linéaires robustes itératives.

repeat
- Estimate($\widehat{\beta^n}$) ; (Régression des moindres carrés)
- Estimate(ω) ; (W-estimateurs)
- Estimate($\sum \omega_i r_i^2$) ; (Régression pondérée des moindres carrés)

until $\omega^i \approx \omega^{i-1}$;

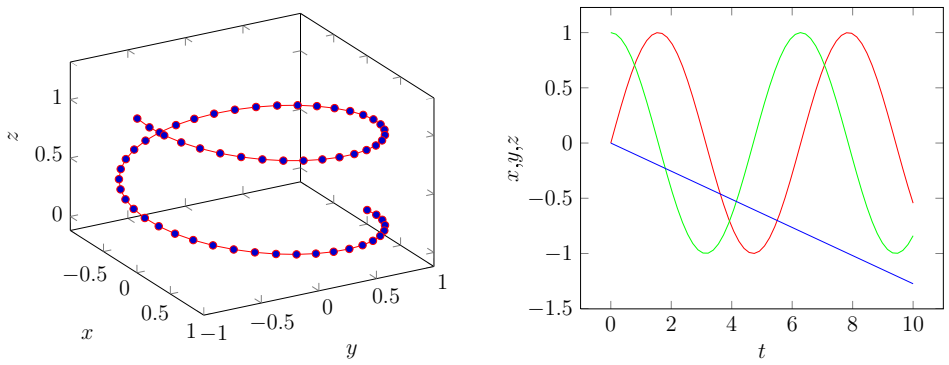
L'algorithme 4.1 est un algorithme itératif. Le surcout se trouve au niveau des multiples itérations dans l'estimation des poids. Le même résultat peut être obtenu avec une somme des moindres valeurs absolues, dès lors, on considèrera cette solution comme plus avantageuse dans notre contexte. Pour de plus amples informations sur les régressions robustes, voir Rousseeuw et Leroy [109].

En utilisant la règle 1, nous définissons une procédure de régression linéaire des trajectoires de capture de mouvements sous forme de segments de droite. Le paramètre ε permet de gérer la granularité de la segmentation des trajectoires. Il s'agit d'ajuster, éventuellement, la précision de la décomposition pour tenir compte du bruit, notamment celui d'extraction des poses de capture de mouvements.

Afin d'augmenter la robustesse de notre estimateur aux différents mouvements non linéaires exécutables par un être humain, nous avons choisi d'exprimer la régression linéaire sur chacun des axes du repère global de la scène. En effet, un mouvement circulaire engendrerait une sur-segmentation de la trajectoire si l'on ne considère que des droites. Or, si l'on exprime le mouvement sur chacun des axes de la base, X, Y et Z, alors, il est possible de limiter les effets de la sur-segmentation, comme précisé sur la figure 4.2. Dans ce contexte, il est évident que le nombre de segments de droites est très inférieur lorsque l'on considère les signaux indépendamment, que lorsque l'on considère le signal 3D initial.

4.1.2 Décomposition atomique

Pour des raisons d'efficacité, nous avons regroupé les régressions des trajectoires par points de capture, *i.e.* après avoir exprimé la régression sur chacun des axes, nous avons fusionné les informations de coupure en un état unique par os du squelette virtuel suivi. Nous avons ainsi créé un ensemble de points de coupure sur les trajectoires du squelette d'animation qu'il faut fusionner de façon efficace. Pour cela, nous proposons de regrouper sous forme d'un élément, pour un unique point de capture, les régressions de chacune des dimensions dès lors qu'elles ont toutes variées au moins une fois. Le regroupement n'est pas effectué à chacune des variations d'un axe, car cela produirait une sur-segmentation des trajectoires, phénomène que nous cherchons à réduire au maximum.



(a) Trajectoire 3D (b) Projection sur les axes X, Y, Z

FIGURE 4.2 – Projection d'une trajectoire circulaire du mouvement d'une main. Cette solution évite une sur-segmentation des régressions linéaires, qui aurait pu arriver avec une régression linéaire appliquée directement sur la trajectoire 3D.

Pour plus de clarté, nous introduisons les définitions suivantes :

Définition 3 Action Cut**.** *Un* Action Cut t_i*, est le temps auquel une discontinuité (singularité), d'ordre* C^1*, sur la trajectoire d'un point de MoCap, se produit.*

Nous définissons la présence d'un *Action Cut* lorsqu'une discontinuité d'ordre C^1 s'est produite sur tous les axes (X, Y et Z). C'est-à-dire qu'un *Action Cut* est présent à la dernière discontinuité d'un des trois axes, *i.e.* le dernier à varier, voir la figure 4.3 pour un exemple d'*Action Cut*. Notons qu'en phase d'apprentissage, un *Action Cut* est mis au début, respectivement à la fin, de chacune des actions apprises. Ces *Action Cuts* portent le nom de *Top-Start* et *Top-End* respectivement.

Définition 4 Action Element**.** *Un Action Element (*AE_P*) d'un point de MoCap* P *est le mouvement de* P *exprimé entre deux* Action Cuts *consécutifs.*

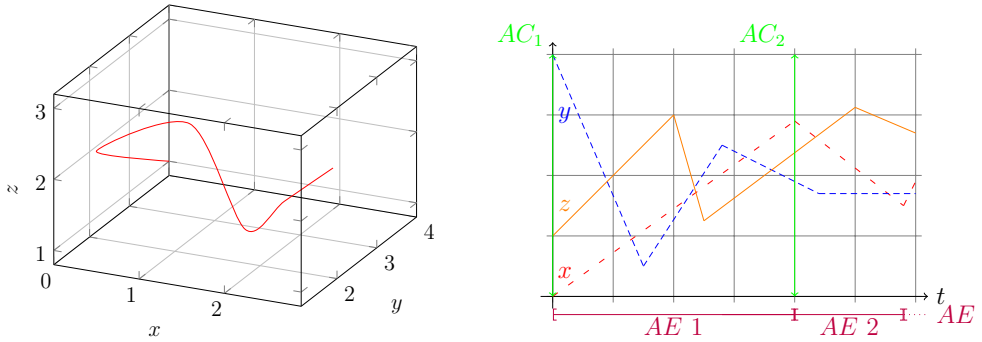
Un Action Element AE_P *est exprimé de la façon suivante :*

$$AE_P \quad = \quad \begin{bmatrix} P_B & P_E & T_B & T_E \end{bmatrix} \tag{4.2}$$

Où : $P_B, P_E \in \mathbb{R}^3$ *sont les coordonnées spatiales du point* P *respectivement au début et à la fin de l'*Action Element*, au temps* T_B *pour le début, et* T_E *pour la fin.*

La figure 4.3 fait le lien entre les régressions linéaires, les *Action Cuts* et les *Action Elements*.

Si l'on considère l'ensemble des *Action Elements*, pour toutes les articulations, on remarque que chaque point de MoCap est représenté par un ensemble continu de segments. Il est dès lors possible de les regrouper en suivant la définition 5.



(a) Trajectoire 3D (b) Trajectoire modélisée

FIGURE 4.3 – Exemple de décomposition d'une trajectoire, réduite linéairement, en *Action Elements*.

Définition 5 Action Segment. *Soit* $\mathbb{A}$*, l'ensemble des articulations du squelette d'animation.* $\forall P \in \{\mathbb{A}\}$ *dans l'intervalle temporel* $[T_i, T_{i+1}]$*, Un* Action Segment *AS est défini par :*

$$AS = \langle\!\langle AE_P \rangle\!\rangle_{T_i}^{T_{i+1}} \tag{4.3}$$

$$= \left\langle\!\!\left\langle \bigcup_{P \in \{\mathbb{A}\}} (AE_P) \right\rangle\!\!\right\rangle_{T_i}^{T_{i+1}} \tag{4.4}$$

Où : $\langle\!\langle \, . \, \rangle\!\rangle_{T_i}^{T_{i+1}}$ *représente l'opérateur de concaténation pour l'intervalle temporel* $[T_i, T_{i+1}]$.

La définition des *Action Segments* nous laisse le soin de définir la longueur de l'intervalle $[T_i, T_{i+1}]$. Nous avons associé la longueur avec la résolution du système de capture de mouvements. Ainsi $|T_i, T_{i+1}| = \frac{c}{fps}$ où : c est une constante — entre 6 et 10 généralement — et fps est la fréquence du système de MoCap. Les *Action Segments* peuvent être vus comme la succession des *Action Elements*, de tous les points du squelette de MoCap, au cours d'un intervalle temporel donné. En d'autres mots, les *Action segments* sont aux poses ce que les *Action Elements* sont aux os du squelette de MoCap.
N.B. : au début d'une action et à la fin de celle-ci, le *Top-Start* et le *Top-End* décrivent respectivement le début et la fin des *Action Segments*.

4.2 Comparaison de trajectoires

Les *Action Elements* ne sont pas invariants au temps. Ils ont donc besoin d'une métrique particulière pour être comparés, qui tienne compte de la variation tem-

porelle inhérente à chaque action humaine. L'équation 4.1 propose un critère d'erreur sur la régression linéaire, et donc par extension sur la distance entre actions. Néanmoins, celui-ci est loin d'être suffisant. En effet, il donne trop d'importance à l'erreur temporelle en comparaison de l'erreur spatiale. Or, ceci est à proscrire lorsque l'on s'intéresse aux actions humaines. Nous avons défini un critère, dont l'objectif est proche de la méthode de *Dynamic Time Warping* — proposée par Sakoe et Chiba [113] — permettant d'attribuer des pondérations différentes aux informations spatiales et temporelles.

Lorsque l'on compare deux *Action Elements*, $AE_P = \begin{bmatrix} P_B^P & P_E^P & T_B^P & T_E^P \end{bmatrix}$ et $AE_Q = \begin{bmatrix} P_B^Q & P_E^Q & T_B^Q & T_E^Q \end{bmatrix}$, la distance peut être représentée sous la forme d'une somme de deux distances : la distance spatiale et la distance temporelle, telle que représentée par l'équation 4.5. De façon simpliste, nous dirons que les éléments atomiques du mouvement sont influencés par une proximité spatiale et une proximité temporelle, que nous sommons afin d'obtenir une mesure de leur similarité.

$$d\left(AE_P, AE_Q\right) = d_{Spatiale} + d_{Temporelle} \tag{4.5}$$

La figure 4.4 met en avant ce principe par l'intermédiaire d'un exemple de comparaison de deux AE, où les distances sont représentées de façon géométrique.

Nous utilisons une distance usuelle pour la distance spatiale, *i.e.* la distance euclidienne entre deux points. Pour la distance temporelle, nous avons défini notre propre distance. Nous devons corréler plus fortement la distance temporelle à la distance spatiale, afin de minimiser l'importance de cette première. Pour cela, notre distance incorpore une normalisation par rapport à la distance spatiale. Nous traduisons donc cette normalisation par une factorisation par la distance spatiale :

$$\begin{aligned} d\left(AE_P, AE_Q\right) = & \overbrace{\left|\left(\left|P_B^P - P_B^Q\right| - \left|P_E^P - P_E^Q\right|\right)\right|}^{\text{composante spatiale}} \\ & \times \left(1 + \underbrace{\frac{\left|T_E^P - T_E^Q\right|}{\max\{T_E^P, T_E^Q\}}}_{\text{composante temporelle}}\right) \end{aligned} \tag{4.6}$$

La figure 4.5 traduit cette imbrication et donne une intuition 2D du calcul de la distance 4.6.

Afin de garder les notations le plus simple possible, et sans perte de généralité, nous considèrerons dans la suite que $T_0 = 0$, comme montré dans la figure 4.5. Il s'agit d'une conséquence de notre stratégie de reconnaissance, basée sur des

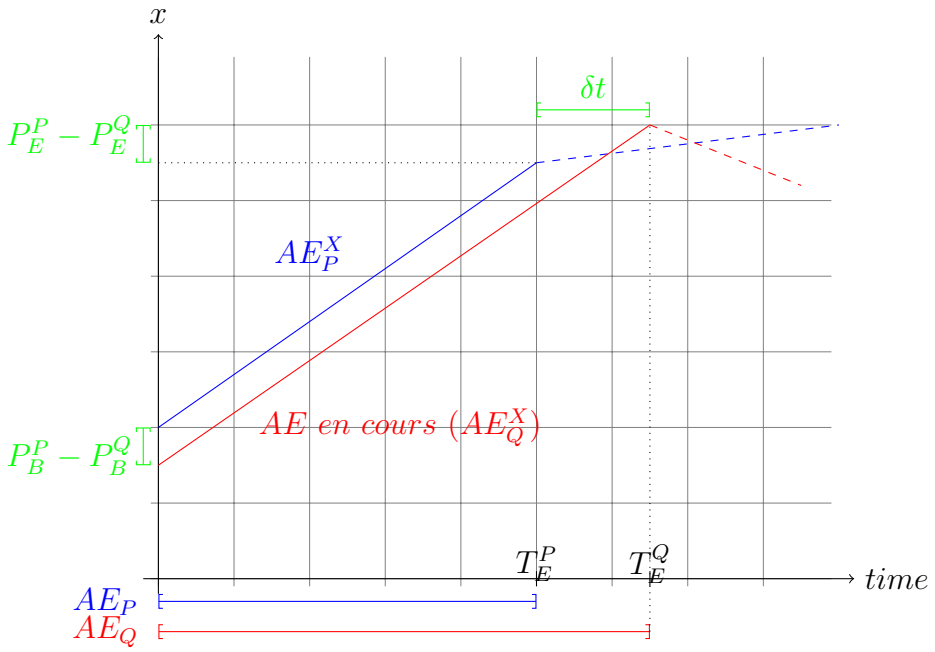


FIGURE 4.4 – AE_P est comparé à l'*Action Element* courant AE_Q Seule la projection sur l'axe X est représentée.

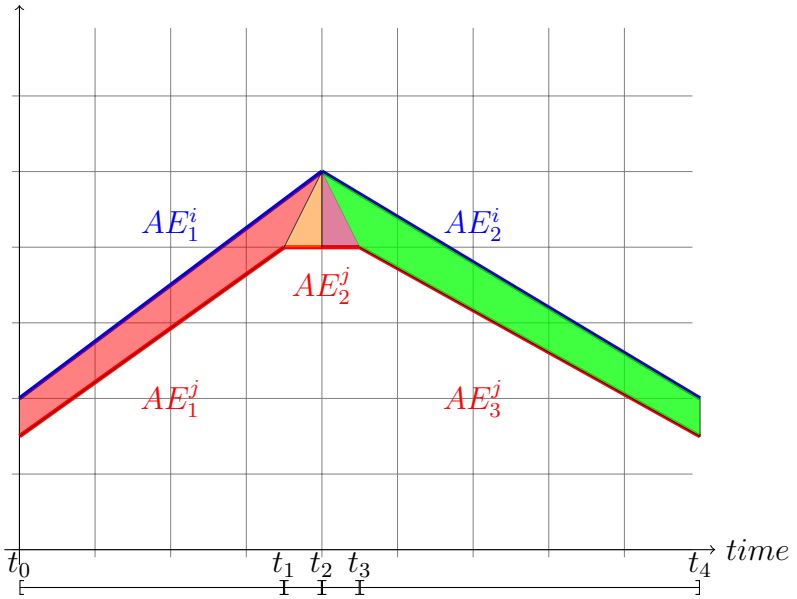


FIGURE 4.5 – Comparaisons entre $AS_i = \langle\!\langle AE_1^i, AE_2^i \rangle\!\rangle_{t_0}^{t_4}$ et $AS_j = \langle\!\langle AE_1^j, AE_2^j, AE_3^j \rangle\!\rangle_{t_0}^{t_4}$, au sein d'un même *Action Segment*.

comparaisons d'*Action Elements* indépendants les uns des autres. Nous justifions la différence d'importance accordée à la partie spatiale par rapport à la partie temporelle, par les variations inhérentes des êtres humains lors d'un mouvement, variations naturelles dues à leur état de forme, à leur état de concentration, notamment, mais aussi à la différence de proportion du corps, aux différences stylistiques, *etc.* induites par la répétition d'une même action. Il est pratiquement impossible pour un être humain de reproduire, deux fois, exactement le même mouvement aux sens géométrique et temporel du terme.

Sachant que chaque *Action Segment* est composé de plusieurs *Action Element*, Nous devons unifier les comparaisons d'actions au niveau supérieur à celui des *Action Elements*, c'est-à-dire, celui des *Action Segment*. Pour cela, nous proposons de sommer l'ensemble des distances entre *Action Elements* à l'intérieur d'un même *Action Segment*. De plus, cette sommation doit tenir compte des trois dimensions spatiales des points de MoCap, contrairement à la distance entre *Action Elements*, qui travaille sur une projection du mouvement sur chacun des axes du repère global. La distance entre deux *Action Segments* AS_i et AS_j est donnée par :

$$D(AS_i, AS_j) = \sum_{t=0}^{T} \left[\sum_{P \in \{\mathbb{A}\}} d\left(AE_P^i, AE_P^j\right) \right] \tag{4.7}$$

La distance entre *Action Segments* étant défini, nous proposons de fournir une évaluation incrémentale de ce score. En particulier, un système de vote (Hough-*like*), similaire à celui proposé par Leibe et al. [68], a été mis en place afin de valider la succession entre *Action Segments*. Si la distance D entre deux *Action Segments* est inférieure à un seuil, alors la correspondance est acceptée et un vote positif est émis. Dans le cas où D est supérieur au seuil, un vote négatif est émis. Il est dès lors possible de calculer un critère de vraisemblance entre deux actions, par l'intermédiaire de leurs *Action Segments*. Ce critère, entre deux actions — l'action en cours A et l'action candidate B — peut être évalué de façon invariante au temps. En effet, la comparaison de deux *Action Segments* ne doit pas impliquer que ceux-ci aient été réalisés à des instants temporels identiques, *i.e.* il n'y a pas de repère temporel absolu dans nos comparaisons d'actions. Nous verrons dans le chapitre 5 comment tirer parti de cette particularité pour regrouper et réarranger les actions au sein de notre structure de reconnaissance. Le score de vraisemblance entre *Action Segments* est donné par l'équation 4.8.

$$C(A, B) = \sum_{i \in [0,\ldots,N]} \left(1 - \frac{D(AS_i^A, AS_i^B)}{\max\limits_{j \in [0,\ldots,N]} \left\{ D(AS_j^A, AS_j^B) \right\}} \right) \tag{4.8}$$

Où : $N = |A|$, *i.e.* le nombre d'*Action Segments* de l'action A.

Notons que ce score est toujours positif, et que plus il est important, meilleur est la correspondance entre le candidat et l'action en cours. Ce score est traduit en une mesure bornée, en divisant par le nombre N, afin d'avoir des similarités indépendantes du facteur de décomposition utilisé (voir le paramètre ε, de la règle 1). Notons aussi que ce score est calculé de façon incrémentale, même lorsque les *Action Segments* ne sont pas définitifs, c'est-à-dire lorsqu'un *Action cut* n'a pas encore été trouvé. Dans ce cas, il est révisé à chaque « adaptation » de l'*Action Element*, *i.e.* l'incorporation d'un nouveau point dans la régression linéaire, de l'action en cours, et ce, jusqu'au prochain *Action Cut*. Dans ce cas, le score devient « définitif » entre les deux *Action Segment*. Il est ainsi possible de fournir une proximité entre deux actions en cours d'exécution par le score de l'équation 4.8, de façon incrémentale, tout en sachant que ce score peut être révisé à la hausse comme à la baisse par notre méthode.

Chapitre 5

Reconnaissance par automate

Sommaire

Dans ce chapitre, nous appliquons le schéma de décomposition des actions, par leurs trajectoires, en éléments atomiques, afin de proposer une reconnaissance des actions. Notre but prioritaire est de faire une reconnaissance dite « en ligne » des actions. Pour cela, notre méthode devra être capable de travailler avec le flux de MoCap, à sa résolution native : au moins 30 images par seconde, mais plus souvent 60 ou 120 images par seconde. La réduction proposée dans le chapitre 4 permet d'obtenir des vecteurs états compacts. De plus cette réduction est temps réel et en ligne. Il est nécessaire de conserver ces propriétés lors du processus de reconnaissance. Nous proposons une structure permettant un parcours rapide des régressions de trajectoires successives. Nous avons fait le choix d'un graphe d'actions, parcouru sous la forme d'un automate. Ce choix s'explique, d'une part par une volonté de ne pas utiliser de bases de données, d'autre part de se passer de structures complexes et coûteuses en mémoire.

Les graphes ont été utilisés — de façon un peu différente — dans la communauté animation, afin de créer des transitions fluides entre différentes animations : Kovar et al. [60] et Beaudoin et al. [18], par exemple. Bien que les objectifs soient différents, à savoir créer des points de convergence entre des animations intrinsèquement différentes, comme passer de la marche à la course, l'organisation de ces méthodes — autour de données issues de capture de mouvement — montre un point de convergence intéressant.

Nous cherchons à mettre en avant les parties communes des actions issues de la capture de mouvement, de même qu'à produire une décision « rapide » concernant la reconnaissance de la séquence de poses courante. Nous proposons de construire un automate dont les états sont les *Action Segments*, des actions du jeu d'apprentissage ; les transitions sont faites par les chaînes d'*Action Elements*, c'est-à-dire que les *Action Segments* successifs dans les actions du jeu d'apprentissage sont reliés entre eux par les *Action Elements* dans l'automate ; l'état initial, respectivement l'état d'acceptation, est le premier, respectivement le dernier, *Action Segment* extrait de l'action du jeu d'apprentissage. Une action est généralement composée d'un ensemble restreint d'*Action Segments*, liés par le temps. Des actions différentes peuvent partager des *Action Segments*. Dans ce cas, ceux-ci peuvent être composés de plus d'une transition, *i.e.* une par *Action Elements* différents dans les actions d'apprentissage.

Dans notre automate, nous mettons en valeur différents états importants. Les états communs à deux actions différentes sont mis en exergue — couleur bleue dans la figure 5.1 — car ils créent des « ponts sémantiques » entre les actions. En effet, que dire de la reconnaissance, valide d'un point de vue structurelle, commençant sur l'état initial $\mathcal{B}_0$ et se terminant sur l'état d'acceptation de l'action $\mathcal{A}$, en passant par la transition $\mathcal{B}_p$? Un autre type d'état est mis en avant dans notre automate : les boucles. Lors de l'apprentissage, les nouveaux états sont recherchés parmi les états existants. Il est ainsi possible de trouver des chaînes d'états récurrents. Or, dans la plupart des actions humaines, le nombre des cycles courts n'a pas d'importance. Ainsi, si l'on apprend un mouvement circulaire de la main, par exemple dans le sens de : « continue », le nombre de tour de la main n'a pas d'importance. Néanmoins, dans une reconnaissance à base d'états successifs, ce nombre est figé. Notre solution de mettre en avant les boucles permet de s'affranchir de ce nombre, et ainsi avoir un parcours simplifié de l'automate, quel que soit l'action impliquée. Notons enfin que l'état initial et l'état d'acceptation peuvent être confondu, cf. action $\mathcal{C}$, figure 5.1.

5.1 Ajouter une action

Grâce à notre méthode de réduction des trajectoires, ainsi qu'à notre structure de représentation des actions, ajouter une action à notre automate, c'est-à-dire « apprendre une nouvelle action », est un processus rapide et efficace en mémoire. Nous abordons le problème de l'apprentissage sous le signe des *examplars*, tel qu'il a été défini par Frey et Jojic [41]. Nous supposons que nos actions, dans le jeu d'apprentissage, sont le centre d'une distribution probabiliste dont la sémantique est associée à cette dernière. C'est-à-dire que nous supposons que chacune des actions du jeu d'apprentissage est un représentant central pour ce type d'action,

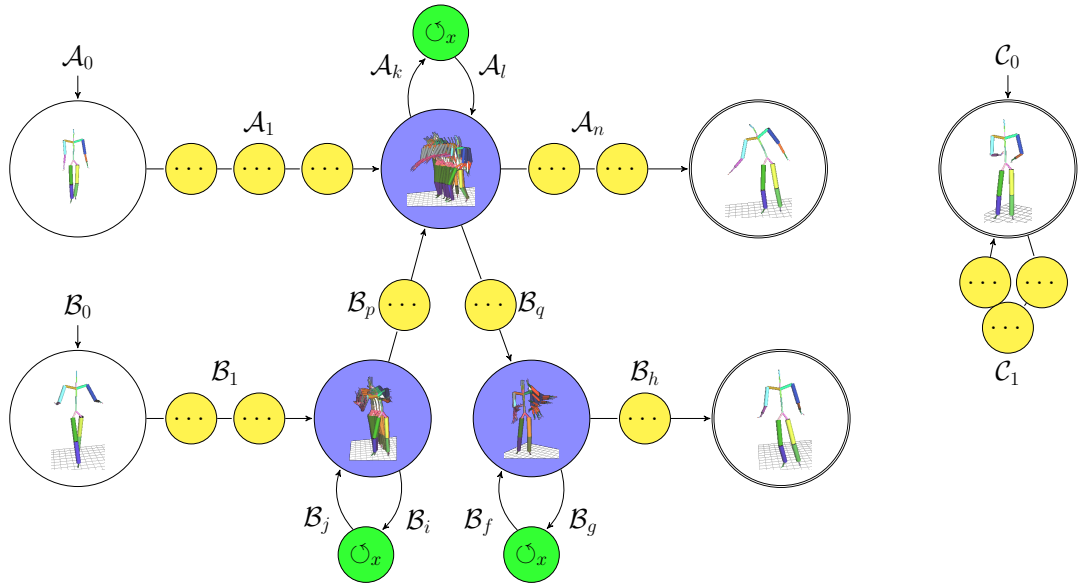


FIGURE 5.1 – Exemple d'un automate reconnaissant trois actions A, B et C. Les *Action Segments* sont représentés par les états jaunes ; les boucles sont mises en valeurs par les états verts ; les *Action Segments* communs à plusieurs actions sont soulignés par les états bleus.

i.e. ayant cette signification. Contrairement à ce qui est classiquement fait avec des *examplars*, nous n'utilisons pas une modélisation Markovienne des actions. Ainsi, apprendre une action, consiste à réduire les poses de cette dernière par la méthode proposée au chapitre 4, et à ajouter chacun des états à l'automate.

5.1.1 Utilisation de l'automate

De façon plus formelle, le processus engagé pour l'apprentissage d'une action est le suivant. À partir d'un flux de poses, issu d'une méthode de capture de mouvement, nous appliquons la régression des premières poses afin d'obtenir le premier *Action Segment*. Une recherche exhaustive dans le graphe est faite, pour savoir si un *AS* similaire est déjà présent. S'il n'est pas présent — ce qui est le plus fréquent pour un début d'action — celui-ci est ajouté, et devient un état initial de l'automate. Dans le cas où cet AS serait un état existant de l'automate, il est transformé en un état initial. Ceci signifie généralement qu'une action dispose d'une « variante initiale », *i.e.* qu'il est possible de commencer cette action de plusieurs façon différentes. Par la suite, les autres *AS* sont ajoutés à l'automate en suivant le même processus, mais sans être des états initiaux. Le dernier *AS* ajouté est désigné état d'acceptation. Lors de l'ajout du dernier *AS*, différents cas peuvent se présenter :

Algorithme 5.1: Adding a new action (learning) into the automaton.

```
Input : Automaton Graph 𝒢 and a video stream 𝒱
repeat
    if Top-Start then
        while not Top-End do
            AS_i ← GetAS( from MoCap(𝒱) );
            forall the AS_i ∈ 𝒢 do
                if AS_i == AS then
                    MergeSuccessors( AS_i );
                    MergePredecessors( AS_i );
                else
                    Add( AS_i, to 𝒢 );
                end
            end
        end
    end
until EndLearningProcess;
```

(i) cet état n'était pas précédemment dans l'automate, dans ce cas, il est ajouté et il devient un état d'acceptation ;

(ii) cet état était déjà un état d'acceptation, dans ce cas, rien n'est fait, et on est probablement dans le cas particulier de l'ajout d'un état initial ;

(iii) cet état est présent dans l'automate, dans ce cas, la logique voudrait qu'il devienne un état d'acceptation, mais cela pose un problème de parcours « sémantique » de l'automate.

En effet, que dire de l'action initiale, qui considérait cet état comme une « simple transition » et qui disposait d'un état d'acceptation différent ? Dans ce cas très particulier, il n'est plus possible de reconnaitre l'action précédente sans reconnaitre, au moins en partie, la nouvelle action. Ceci est probablement dû à un amalgame dans le jeu d'apprentissage. Des actions de granularité différentes sont présentes comme « marcher » et « faire un pas ». Notre structure de reconnaissance permet une interprétation de l'action avec des niveaux différents, une sorte de niveau de détails, mais sur la sémantique du geste plus que sur la précision de la reconnaissance.

L'algorithme 5.1 résume l'apprentissage d'une nouvelle action.

5.1.2 Utilisation des *examplars*

Il est très difficile de s'assurer — de façon *ad hoc* — qu'une action est bien un *examplar* : c'est-à-dire qu'il représente effectivement l'ensemble des actions de ce type. Pour ce faire, nous utilisons plus d'une action en guise d'apprentissage. Cela nous conduit à adopter une stratégie d'identification ou de création des *examplars*. Pour cela, nous avons mis en place une stratégie à base de mixture de gaussiennes. Nous utilisons la première action apprise, comme s'il s'agissait d'un *examplar* (régression et ajout dans l'automate), puis, nous utilisons les éventuelles autres instances de la même action — *i.e.* des actions différentes mais partageant le même label — afin d'établir la moyenne et l'écart type entre cette nouvelle action et celle dans l'automate. Ces calculs se font par l'intermédiaire des AE et de l'équation 4.6. Ensuite, les AE utilisés dans l'automate sont mis à jour de façon à respecter une distribution gaussienne. C'est-à-dire qu'ils sont « remplacés » par la moyenne des AE s'ils ne sont pas « trop » différents de l'écart type, sinon, ils sont considérés comme des *outliers*. Cette stratégie n'introduit pas de surcout mémoire pour l'automate, car le nombre d'états reste le même.

Lorsque l'on utilise de la capture de mouvement, et plus particulièrement lorsque l'on travaille dans une optique d'interaction Homme–Machine, il n'est pas toujours évident d'avoir de multiples variations d'une même action. En effet, il peut être extrêmement rebutant, pour un utilisateur, de devoir faire même plus d'une fois un geste pour qu'il soit appris par le système. Dans ce cas, on ne peut pas garantir la présence de variation interne autour d'une action afin de certifier que notre *examplar* n'est pas un cas « pathologique » de cette action. Pour cela, nous développons une stratégie alternative. Il n'est pas possible, lorsque l'on parle d'actions humaines, d'utiliser un bruit comme le bruit gaussien (bruit blanc) des images. Nous proposons d'utiliser la méthode de variation d'action, proposé par McCann et al. [81] afin de générer de la variété — artificiellement mais de façon plausible — pour nos actions uniques. Ceci nous permet de remplacer nos états par des versions adaptées en fonction d'un contexte de variation temporelle. Les résultats de cette approche, ainsi que l'ensemble de l'explication de la variation temporelle sont présentés dans le chapitre 6.

5.2 Reconnaitre une action

De manière similaire au processus d'apprentissage, le processus de reconnaissance se déroule sur l'exécution d'un flux de capture de mouvement. De même, ce flux est décomposé en *Action Segment* afin d'effectuer la comparaison aux données de l'automate. Dès lors, notre problème de reconnaissance se transforme en la recherche d'un chemin dans notre graphe (automate), avec le respect de certaines

contraintes. Premièrement, le chemin doit partir d'un état initial. Deuxièmement, le chemin doit finir sur un état d'acceptation. Troisièmement, le parcours doit respecter les transitions possibles au sens de l'automate, à savoir, ne pas être à une distance supérieure à un seuil de l'AS en cours. C'est-à-dire :

$$\underbrace{d\left(AE_P, AE_Q\right)}_{\text{équation 4.6}} \leq \Delta \tag{5.1}$$

Durant le processus de reconnaissance, nous utilisons un score de $0,9$, soit 90% de similarité. Il s'agit d'un score classique, qui doit toutefois être validé lors de la phase d'apprentissage. En effet, ce score ne doit pas être supérieur à la distance entre deux actions différentes dans l'automate, *i.e.* il ne doit pas être possible de reconnaitre deux actions étiquetées comme différentes. Il doit aussi permettre de reconnaitre l'ensemble des actions constitutives des *examplars* utilisés pour générer cette branche de l'automate, cf. section précédente. Ce score est calculé de façon incrémentale durant le processus de reconnaissance. Ainsi, il est possible, à chaque étape de savoir si on est en présence d'une action apprise, et de quelle action il s'agit. Cela autorise la reconnaissance précoce d'actions, de même que les divergences possibles à un instant donné, *i.e.* les embranchements si l'on est sur un état à multiples transitions. De plus, cela permet de corréler le score de similarité sur l'ensemble de l'action. Ainsi, une grande diverge sur la fin de l'action, n'impactera pas fortement le score de similarité. Par exemple, lors de la chute d'une personne, les mouvements dits « parasites » une fois au sol n'ont que peu d'importance sur la validité de l'action, à savoir, la personne est tombée.

L'algorithme 5.2 résume le processus de reconnaissance d'une action à l'aide de l'automate.

Une procédure de résistance au bruit est mise en place. En effet, dans la présentation actuelle de la reconnaissance, un seul état non reconnu peut invalider l'ensemble de l'action. Pour cela, nous avons mis en place un solution à base de « tolérance de panne », qui permet de s'affranchir d'un état non reconnu. Ainsi, un état non reconnu dans une séquence ne conduit pas automatiquement au rejet de l'action. Le système tolère un état d'échec, et continue la reconnaissance sur la même branche, soit en passant à l'état suivant, soit en restant sur l'état actuel pour évaluer la suite du flux. Cela permet de vérifier qu'il ne s'agissait ni d'une erreur de MoCap (bruit d'action) ni une tolérance trop faible de l'*examplar*. Le mécanisme de tolérance modifie le score de l'action. Si un état est « sauté », son score est remplacé par le score moyen de reconnaissance, calculé sur les états précédents. Ainsi, l'impact est faible sur le résultat global. Si l'état n'est pas sauté, mais que l'on reprend à celui-ci après une séquence de poses « parasites », la méthode considère que ce problème vient de la MoCap et donc, le score est calculé comme si cette séquence n'existait pas.

Algorithme 5.2: Action recognition using the automaton.

```
Input : Automaton Graph 𝒢 and a video stream 𝒱
repeat
    AS_i ← GetAS( from MoCap(𝒱) );
    CurrentNode ← FindStartingAS( from 𝒢, to match AS_i);
    if no matching AS found then
        // no action found for this AS_i ;
    else
        while CurrentNode is not an accepting AS do
            AS_i ← GetAS( from MoCap(𝒱) );
            CurrentNode ← FindAS( from Succesors(CurrentNode), to
            match AS_i);
            if no matching AS found then
                break
            end
        end
    end
until EndOfVideoStream;
```

5.3 Complexité

Notre solution est temps réel grâce à une complexité maitrisée. Dans cette section, nous allons analyser la complexité de notre solution, en terme de nombre d'opérations. Supposons que l'on ait n actions apprises dans l'automate. Nous avons donc, au plus, n états initiaux (*Starting points*). Dans le pire des cas, nous avons donc n comparaisons à faire pour trouver le début d'une action. Une fois que le point initial a été trouvé, nous avons peu de comparaisons à faire pour évaluer la transition d'un état à l'autre à l'aide de l'automate. Il s'agit d'évaluer les AE, à chaque étape et pour chaque articulation, afin de calculer le score de correspondance. Lorsque l'on arrive sur un état « intermédiaire » commun à plus d'une action, nous évaluons les différentes possibilités. Ce nombre est généralement très faible, de l'ordre de 2 ou 3 au pire, et bien souvent, les transitions sont uniques.

De façon globale, lorsque l'on ne considère plus uniquement la reconnaissance en ligne, le nombre d'AS peut impacter le temps de reconnaissance. Ainsi, une action composée de m *Action segments*, aura une complexité de reconnaissance de $\mathcal{O}(n \times m)$. Dans tous les cas, m et n sont relativement petits. Un jeu d'apprentissage d'une vingtaine d'actions est déjà important dans la littérature et bien souvent compris entre 10 et 15. On peut considérer que $10 \leq n \leq 30$ sont des bornes acceptables. Par ailleurs, m dépendant intrinsèquement de l'action, l'intervalle de ses valeurs est plus large que le nombre d'actions, de l'ordre d'une centaine

environ, pour les actions les plus longues. Notons toujours que le processus de reconnaissance se fait pendant l'acquisition des actions, ce qui fait que ce processus est « en ligne » et donc qu'il ne dépasse pas le temps d'acquisition d'une action.

Comme notre système se veut évolutif, nous devons aussi considérer le temps d'apprentissage, *i.e.* l'ajout d'une action à l'automate. Supposons que nous avons n actions dans l'automate, cela signifie que nous avons, au plus n états initiaux. Supposons de plus, que nos actions ont en moyenne m AS. Notre automate est composé d'environ $n \times m$ états. Lors de l'ajout d'une nouvelle action, nous devons donc parcourir chacun de ces états et les comparer avec notre état initial. Si notre état existe, alors il est déclaré comme initial, s'il n'existe pas, alors nous l'ajoutons à l'automate. Malheureusement, ce processus de parcours complet de l'automate doit être répété à chaque nouvel état, que celui-ci fasse suite à un état préexistant ou qu'il soit issu d'un état nouveau. Nous devons donc effectuer $n \times m \times m$ comparaisons en moyenne pour une nouvelle action. Dès lors, nous pouvons considérer que la complexité de notre algorithme est $\mathcal{O}(n \times m \times m)$. De façon formelle, la complexité est linéaire en nombre d'actions, car m est une constante. Or cette constante intervient de façon plus prononcée que le nombre d'action, en général $n \leq 30$ et $m \in [50; 200]$. Soit pour ajouter une action, le nombre d'opérations est compris entre 75.000 et 1.200.000. Ces nombres peuvent sembler important, mais face à la durée d'une action et à la vue de la puissance actuelle des processeurs, ce processus est temps réel sur la plupart des machines courantes. Ceci est un net progrès par rapport à la majorité des méthodes d'apprentissage statistiques qui sont généralement beaucoup plus longues et loin d'être temps réel. Notons de plus, que le processus d'apprentissage est marginal en comparaison du processus de reconnaissance. On apprend chaque action qu'une seule fois, mais on la reconnait plusieurs dizaines de fois dans un cas d'utilisation « classique ».

Chapitre 6

Résultats

Sommaire

Ce chapitre présente les expérimentations que nous avons mené sur des données synthétiques et réelles.

6.1 Décomposition en *Action Segment*

Nous avons tout d'abord étudié la décomposition des actions en *Action segments*. Pour cela, nous avons utilisé le jeu de données MoCap CMU [84]. Un ensemble de 15 actions, issues du même utilisateur, et extraites de la même séquence, a été utilisé afin d'évaluer la réduction en AS. Le tableau 6.1 présente les résultats sur ces 15 actions. Il présente le temps nécessaire à l'apprentissage de ces actions dans un seul et même automate. L'apprentissage se fait de façon successive et incrémental. On remarquera que le nombre d'AS n'est pas lié à la durée de l'action, mais à son nombre de « variations » internes. Ici, les actions sont issues du même acteur, qui les répète un nombre de fois similaire, d'où un grand nombre d'actions décomposées en 87 AS. On remarquera aussi que le temps d'apprentissage augmente avec le nombre d'actions apprise. Ceci est en adéquation avec l'analyse de la complexité faite au chapitre précédent. Grâce à notre décomposition quasi constante en AS, on voit que le temps d'ajout d'une nouvelle action n'est pas tant lié à la longueur de cette dernière qu'au nombre d'actions déjà apprises dans

#	Action	# *AS*	% Réduction	Apprentissage	Durée
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

Temps cumulatif

TABLE 6.1 – Exemple sur 15 actions du jeu MoCap CMU [84] de la décomposition en *Action Segment*. Calcul fait sur Intel® Core™ 2 Quad CPU@2.83GHz, avec 8Gb de RAM.

l'automate. Ce tableau montre que l'apprentissage de $87,01$ secondes d'actions — sous forme de capture de mouvements à 120 poses par seconde — se fait en $2,113$ secondes uniquement, soit moins de $2,5\%$ de la durée des actions est passée en apprentissage. Ceci permet de conserver une grande partie du temps processeur (CPU) pour le système de capture de mouvements, ou tout autre activité, comme l'application cible dans le cas d'une interface gestuelle.

6.2 Génération du « bruit d'actions »

Ajouter du bruit dans des actions est un défi en soi. En effet, le bruit gaussien peut facilement se rajouter aux positions des articulations, comme utilisé par Yao et al. [155], néanmoins, celui-ci ne recouvre pas de réalité tangible. Il se traduit par des tremblements sur les points de capture, qui montre certes une résistance au bruit de capture, mais pas une tolérance aux variations naturelles d'un corps humain. Pour générer du bruit, dans les animations, qui soit ressemblant aux variations qu'aurait pu faire un être humain, nous nous sommes inspirés des travaux réalisés dans la communauté animation. En effet, ces travaux ont proposés différentes solutions pour « casser » l'effet répétitif d'une animation, c'est-à-dire, générer de la variété avec une unique animation en données d'entrée. Or, le facteur ayant le plus d'impact sur la validité de notre méthode, *i.e.* celui qu'un être humain est le plus sujet à faire, est la différence de tempo dans la reproduction du mouvement. Si l'on demande à une personne de refaire le même mouvement, dans

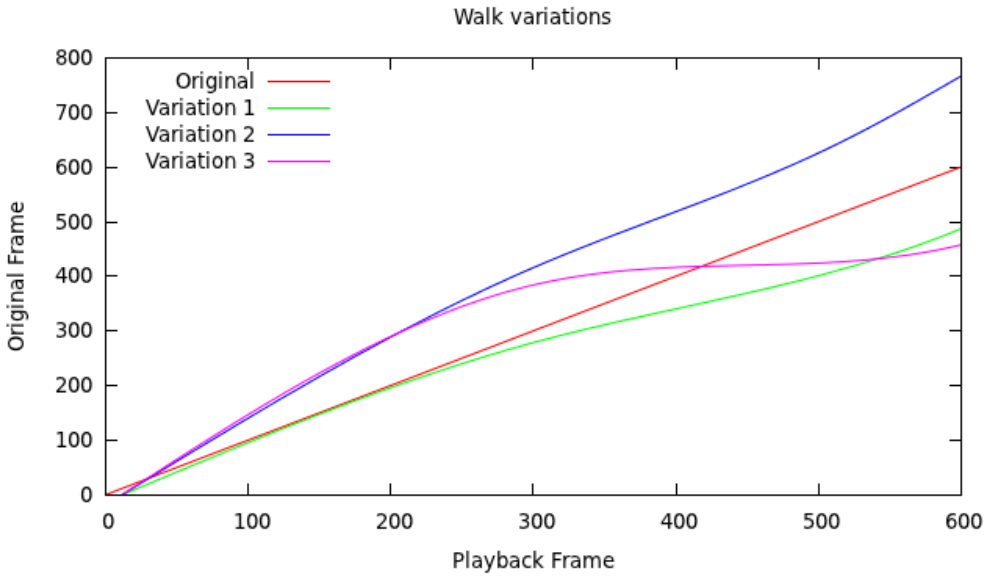
la plupart des cas, son nouveau mouvement sera ou plus rapide ou plus lent que le mouvement initial. Cela tient dans l'extrême difficulté d'une part, de refaire le même geste, d'autre part, de percevoir la vitesse de nos propres mouvements.

Afin d'évaluer la robustesse de notre système aux différents rythmes possible, nous avons mis en place la méthode proposée par McCann et al. [81]. Leur solution se base sur un concept dit de « *retiming* » par des contraintes physiques. Leur méthode propose de changer la position temporelle à laquelle une partie de l'action s'est produite, en tenant compte de contraintes physiques, comme la gravité. Les figures 6.1 et 6.2 montrent les effets de ce « recalage » temporel sur deux actions du jeu de données CMU. Dans chacune de ces figures, quatre réglages différents sont présentés. Ils correspondent aux quatre réglages visibles sur les courbes (6.1a et 6.2a). Les autres sous-figures (6.1b, 6.1c, 6.1d, 6.1e et 6.2b, 6.2c, 6.2d, 6.2e) montrent respectivement le même instant sur l'action initiale et le résultats sur l'animation recalée temporellement, deuxième squelette de l'image. Les détails du recalage temporel, ainsi que l'influence de ses paramètres sur la gravité, la vélocité ainsi que l'accélération se trouvent dans l'article de McCann et al. [81].

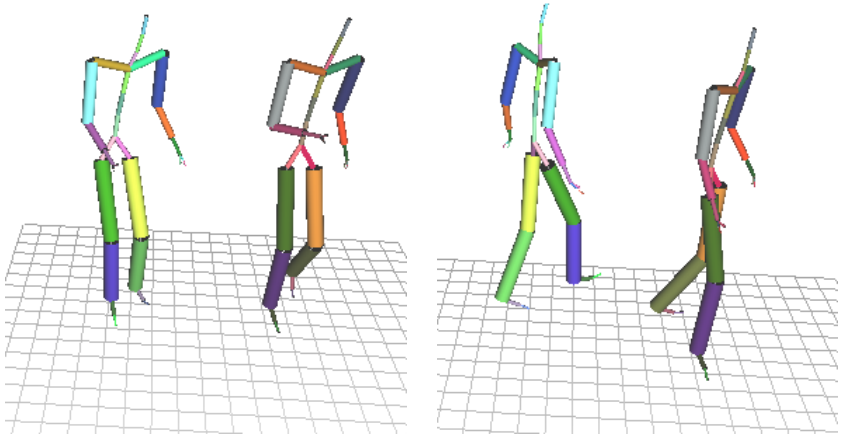
Cette génération de bruit nous permet d'évaluer notre système de reconnaissance sur des variations synthétiques d'actions réelles (issues d'un système de capture de mouvements avec marqueurs). En effet, nous générons aléatoirement une centaine d'actions à partir de chacune des actions initiales. Parmi ces 101 actions (100 synthétiques +1 réelle) nous utilisons successivement chacune d'entre elles pour être l'*examplar* d'apprentissage (*Cross Validation*, tel que présenté par Devijver et Kittler [33]) Le score de reconnaissance est moyenné sur l'ensemble des autres actions. La figure 6.3 montre ces scores. Notre système montre une grande tolérance aux bruits de recalage temporel, en ceci qu'il est proche de 100% et toujours au-dessus de 90%. Dans ce tableau, nous avons fusionné la reconnaissance des exécutions (1) et (2) des actions quand elles existaient, voir le tableau 6.1, car il s'agissait d'actions sémantiquement identiques.

6.3 Reconnaissance d'actions réelles

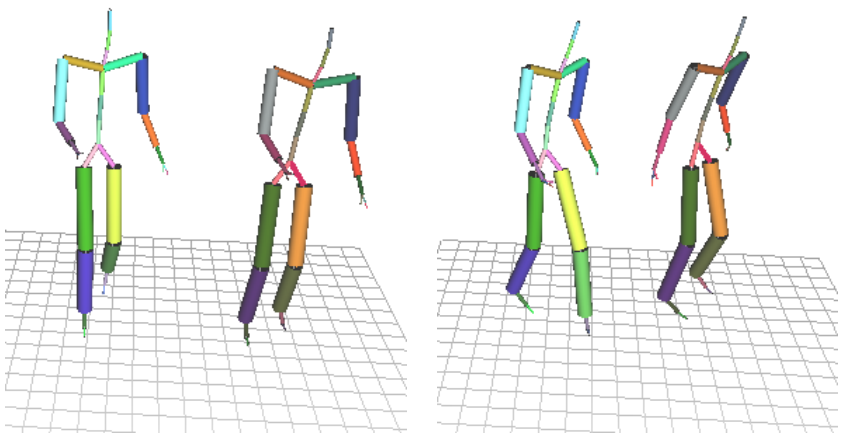
La reconnaissance d'actions « synthétiques » montre de bon résultat. Nous avons aussi testé notre solution sur des données issues d'acquisition temps-réel, plus complexes et plus proche des jeux de données 2D. Pour cela, nous avons créé un jeu de données à partir d'un système de capture de mouvements temps réel, en ligne et sans marqueur, proposé par Shotton et al. [122]. Leur solution repose sur l'utilisation d'un périphérique Kinect^{TM}. Ainsi, ils sont en mesure de réaliser une capture de mouvements dans un environnement non contrôlé, en temps réel, et ce pratiquement quel que soit la morphologie du joueur. Nous avons créé un



(a)

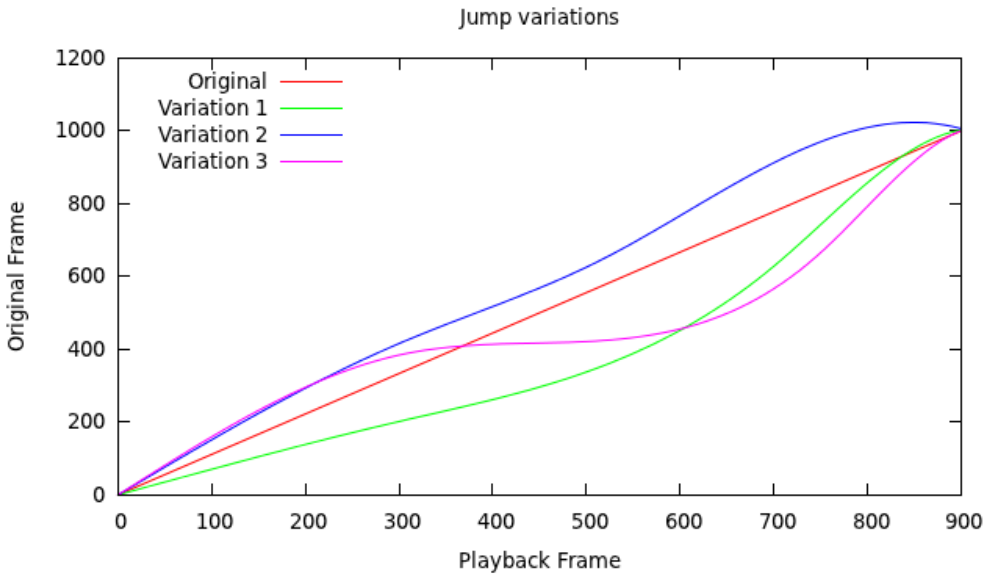


(b) (c)

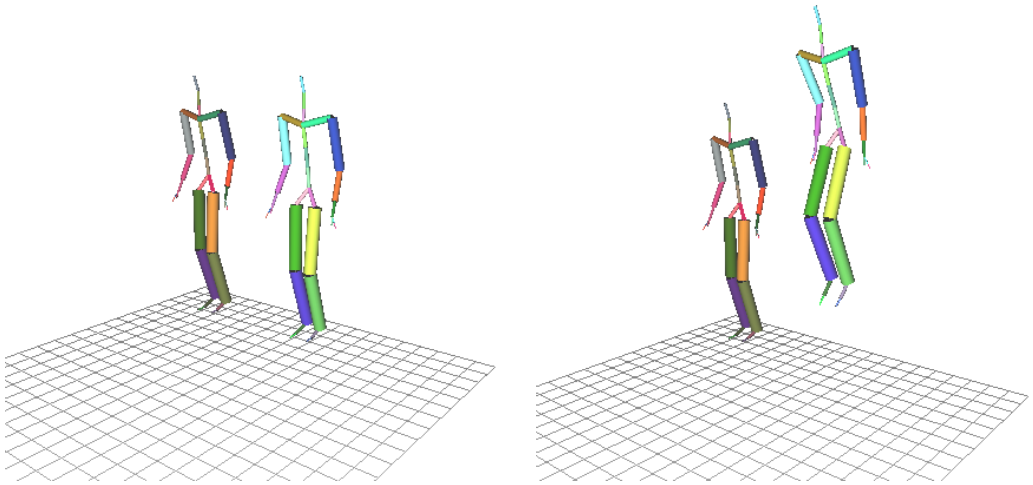


(d) (e)

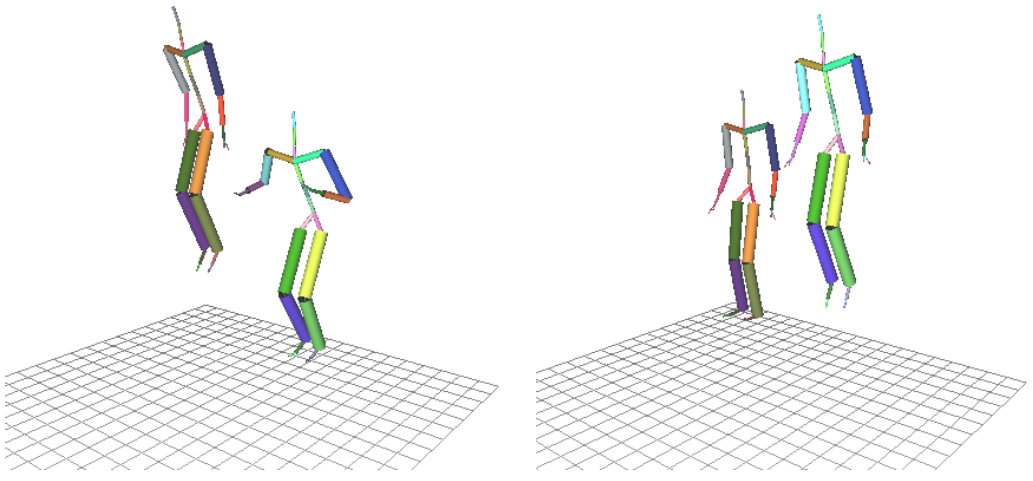
FIGURE 6.1 – Variation sur l'action « walk » à partir de la méthode de McCann et al. [81]



(a)



(b) (c)



(d) (e)

FIGURE 6.2 – Variation sur l'action « jump » à partir de la méthode de McCann et al. [81]

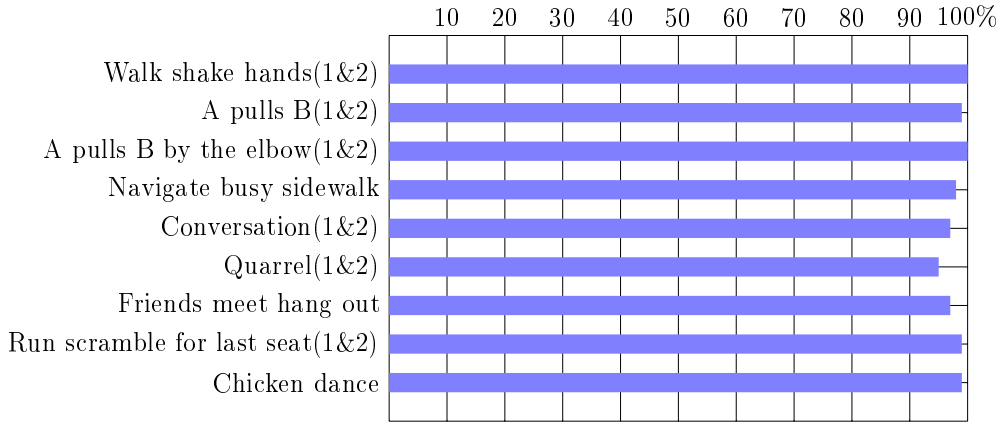


FIGURE 6.3 – Reconnaissance sur 15 actions synthétiques, en présence de bruit par la méthode de McCann et al. [81]. Calculs fait sur 100 variations de chaque action.

jeu de données composé de 23 actions différentes, effectuées chacune par 5 acteurs différents. Chaque acteur effectue 5 fois l'action, sans autre indication que le nom attribué à celle-ci. Les actions sont des actions classiquement présentes dans les jeux de données 2D de reconnaissance d'actions. Ainsi, nous avons : « Marcher », « Courir », « Frapper des mains », *etc.* ceci afin d'approcher au plus près la complexité des jeux de données précédemment proposés. Notre système montre des résultats très encourageants, toujours supérieur à 80% et souvent supérieur à 90%. Nous avons appliqué le protocole classique dans ce genre de situation, à savoir chacune des actions a été utilisée pour l'apprentissage, à tour de rôle, et les scores de reconnaissance pour les autres, exclus de l'apprentissage, ont été moyennés afin de fournir le score final de reconnaissance de l'action. La figure 6.5 présente l'ensemble des résultats sur notre jeu de données. Étant donnée la grande variété présente, et le faible nombre de variations — seulement 5 exécutions d'une même action, dont une utilisée pour l'apprentissage — les résultats montrent la fiabilité du système dans la plupart des cas. Notons par ailleurs que les reconnaissances sont présentées par actions et non moyennés afin de souligner les disparités éventuelles entre les différents acteurs.

Au-delà de la simple présentation numérique de la reconnaissance, nous avons mis en œuvre une application contrôlée par des gestes, afin de montrer le potentiel en terme d'interface Homme–Machine. Notre démonstrateur est une simple visionneuse d'images, possédant les fonctionnalités suivantes : « image précédente », « image suivante », « première image », « dernière image » « sélection de l'image ». Chacune de ces fonctionnalités peut être contrôlée par un geste. Au lieu de fournir

FIGURE 6.4 – Exemple d'actions réelles de notre jeu de données.

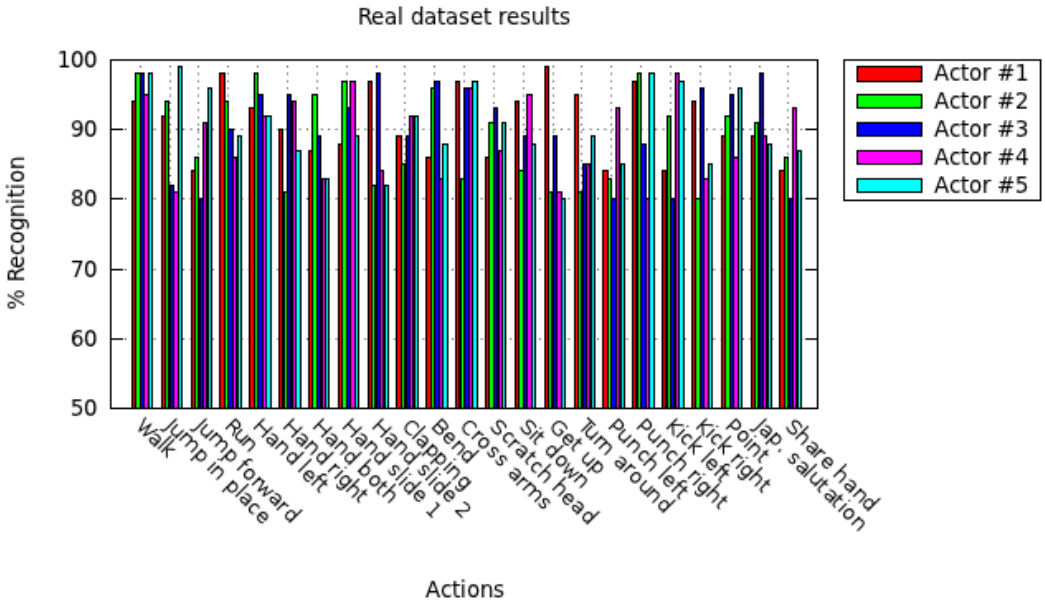


FIGURE 6.5 – Score de reconnaissance sur des actions réelles, limité entre 50% et 100% par clarté.

FIGURE 6.6 – Exemple d'application : une visionneuse d'images contrôlée par des gestes.

une association « figée » et peut être peu intuitive des contrôles, nous avons décidé de laisser à l'utilisateur libre cours quant à son association. Pour cela, il est équipé d'un télécommande sans fil, type WiiRemote^{TM} afin d'effectuer manuellement le *Top-Start* et le *Top-End* nécessaire à l'enregistrement d'une action. La figure 6.6 montre notre réalisation. Afin de se prémunir du bruit environnent, nous avons imposé un minimum de 10 poses avant de créer un *AS*. Cela correspond à $\frac{1}{3}$ de seconde pour ce système de capture de mouvements, *i.e.* la solution de Shotton et al. [122] propose une capture de mouvements à 30 poses par seconde. Les actions apprises sont mémorisées de façon statique et aucune solution d'adaptation, telle que celle proposée par Thevenet et al. [127], n'a été mis en œuvre. Néanmoins, ce démonstrateur a servi de base à la négociation entre un utilisateur et un système dans le cas d'annotation d'image à l'aide de gestes, tâche qui peut vite se révéler fastidieuse si le système n'aide pas son utilisateur, voir Barnachon et al. [15].

6.4 Comparaisons

Il est assez difficile de comparer notre solution avec les méthodes de l'état de l'art. En effet, la plupart d'entre elles reposent sur des algorithmes d'apprentissage statistique, dit *Machine Learning*, bien trop complexes pour être temps réel. De plus, ces algorithmes nécessitent un jeu d'apprentissage très important, et ne peuvent généralement pas se contenter de l'utilisation d'un *examplar* pour l'ap-

Méthode \ Jeux	HDM	CMU	TUM	Approche
SVM (RBF + K)			54,67%	*Machine Learning*
kNN			71,34%	
Tree			75,28%	
Yao et al. [155]	–	–	81,50%	État de l'art
Müller et al. [87]	80%	75%	–	
Tenorth et al. [126]	–	–	62,77%	
Krausz et Bauckhage [62]	–	–	67%	
Notre solution	91,34%	90,78%	85,30%	*Examplar*

TABLE 6.2 – Comparaison de notre méthode de reconnaissance sur différents jeux de données avec les résultats de l'état de l'art. Nous avons mis en œuvre trois solutions utilisant du *Machine Learning* à partir des *examplars* utilisés par notre solution. Pour chacun des jeux de données, la meilleur précision actuellement disponible est reportée afin de comparaison. Notre approche offre des résultats supérieurs aux autres méthodes.

prentissage d'une action, voir le tableau 6.2. Par ailleurs, la plupart des autres méthodes de l'état de l'art en reconnaissance d'actions ne se basent pas sur les mêmes données en entrée. En effet, la majorité utilise des données 2D, comme les images issues des vidéos. Cela rend l'application de notre concept très difficile, si ce n'est impossible, à ces données. Dès lors, nous avons choisi de nous comparer aux méthodes de l'état de l'art qui utilisent des primitives 3D.

Les jeux de données issues de capture de mouvements sont tout indiqués pour faire une comparaison numérique des résultats. Dans ce but, nous avons utilisé les jeux de données suivants : *TUM Kitchen Dataset*, HDM et CMU. Les détails et leurs modalités d'acquisition sont présentés dans l'annexe A.

À des fins de discussion, nous produisons une comparaison qualitative avec les méthodes suivantes : Weinland et al. [144] et Okada et Stenger [95]. La solution proposée par Weinland et al. [144] utilise l'information 3D afin de discriminer les ambigüités visuelles, inhérentes à l'utilisation de caméras. Leur méthode est avant tout destinée à différentier des actions très différentes et ne semble pas avoir suffisamment de précision pour les mouvements fins ou courts. Néanmoins ils utilisent des *examplars* dans un contexte relativement difficile, ce qui fait que nos solutions sont proches. La méthode de Okada et Stenger [95] a l'avantage de n'utiliser qu'une seule caméra. Ils arrivent à faire une capture de mouvements 3D à partir de vision monoculaire, ce qui en soit est remarquable. Malheureusement, leur solution est, avant tout, une solution de capture et peu d'interprétations sont faites. En particulier, ils doivent générer de très nombreuses morphologies humaines afin d'être

capable de reconnaitre le mouvement des personnes. Leur solution d'identification est particulièrement efficace pour l'identification d'une pose humaine, mais n'est pas conçue pour reconnaitre des actions, par exemple sous la forme de successions de poses.

Notre solution permet de dépasser ces problèmes, tout en fournissant une approche rapide et simple à mettre en œuvre. Ceci nous permet aussi de simplement remplacer des actions dans un système informatique par des actions « réelles » de l'utilisateur. Nous ne sommes pas dépendants d'une distance minimale entre actions. Ainsi, la figure 6.5 montre la grande distinction entre les actions : « Sitting » et « Bend » dont les taux de reconnaissance sont proches de 90% contrairement à la plupart des méthodes de l'état de l'art dans ce domaine.

6.5 Conclusion

Nous venons de présenter notre méthode de reconnaissance d'actions par modélisation des trajectoires. Une action est portée par les articulations d'un squelette. Chaque articulation est caractérisée par sa trajectoire, que nous avons modélisé par une fonction continue. Dans notre cas, nous avons opté pour une fonction linéaire pour des raisons d'efficacité. L'étude de l'allure de la trajectoire a permis de la décomposer en une succession d'éléments atomiques (*Action Element*). Le regroupement des éléments atomiques a mis en valeur notre concept d'action élémentaire (*Action Segment*). Au niveau de la reconnaissance, nous avons mis en place un automate dont les états sont les actions élémentaires. Cette structuration s'est montrée très efficace en temps de calculs. Elle présente l'avantage d'être incrémentale aussi bien à l'apprentissage qu'à la reconnaissance. La décomposition des trajectoires d'articulations assure une continuité temporelle du mouvement. De même, le regroupement de ses atomes de trajectoires en poses spatio-temporelles permet d'obtenir une organisation proche du squelette initial. Cet avantage se traduit par une facilité de création des primitives utilisées pour la reconnaissance. La mise en avant dans l'automate des cycles dans les actions, de même que le regroupement des états similaires fourni une représentation sémantique de bas niveau de l'ensemble des actions. Notre système permet d'ajouter de nouvelles actions, si nécessaire, et/ou de modifier certaines en fonction des souhaits de l'utilisateur.

Les résultats de ses travaux ont été publiés dans Barnachon et al. [13, 14].

Troisième partie

Approche par actions

Chapitre 7

Analyse de poses

Sommaire

L'analyse d'actions à partir de données de capture de mouvement, peut se concentrer sur l'analyse des trajectoires, comme nous venons de le montrer en partie précédente. Mais, elle peut aussi se focaliser sur l'apparence de chacune des poses. Ce principe plus général est celui utilisé dans les stratégies de type *Bag Of (Visual) Words*. L'idée étant de trouver des représentants du mouvement, ainsi que leur occurrences particulières à chacune des actions que l'on veut reconnaitre. Pour se faire, nous avons choisi d'étendre le concept des sacs de mots aux poses issues de la capture de mouvements. Nous devons fournir une comparaison entre poses, à l'aide d'une métrique, ainsi qu'un regroupement de ses poses dans une structure d'identification. Notre méthodologie s'appuie sur les histogrammes, en étendant le principe de fréquence à l'apparition des poses dans une action.

7.1 Comparaison de poses

Pour reconnaitre des actions, il est nécessaire de pouvoir comparer des poses. Nous allons introduire le concept d'équivalence entre poses afin de donner une mesure de leur similarité.

Définition 6 ε-équivalence**.** *Soit D_P une distance entre deux poses p_1, p_2, appartenant à l'espace des poses $\mathcal{P}$. L'ε-équivalence entre deux poses est définit par :*

$$p_1 \sim p_2 \quad \Leftrightarrow \quad D_P(p_1, p_2) \leq \varepsilon \tag{7.1}$$

En utilisant la définition 6, il est possible de comparer deux poses, et ainsi de construire des classes de poses. Chaque classe peut être décrite par un représentant unique, noté $\widetilde{p}$. $\widetilde{p}$ est le représentant d'une classe (*cluster*) de poses, définit, à ε près, par :

Définition 7 Représentant**.** *Dans l'espace des poses $\mathcal{P}$, la classe formée par l'ensemble des poses à une distance inférieure à ε les unes des autres peut être représentée par un élément unique, équivalent à tous les autres : $\widetilde{p}$.*

Nous noterons $\widetilde{\mathcal{P}}$ l'espace des représentants de poses.

Généralement, le représentant $\widetilde{p}$ est un des éléments de la classe. Il peut aussi s'agir d'un élément médian. Ce choix est précisé dans chacune des utilisations.

Il peut être lié à la procédure de classifications des poses, en effet, $\widetilde{p}$ sera un élément moyen avec l'algorithme *k-means*, donc pas nécessairement présent dans la classe, *i.e.* $\widetilde{p} \not\subset \mathcal{P}$ dans ce cas. $\widetilde{p}$ peut aussi être le médian de la classe, avec l'algorithme *k-medoids*, *i.e.* $\widetilde{p} \subset \mathcal{P}$. $\widetilde{p}$ peut aussi être vu comme le centre d'une mixture probabiliste, comme dans la définition des *examplars*. Nous donnons la relation suivante :

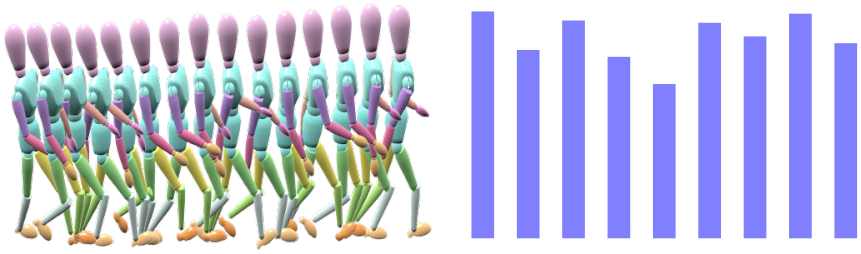
$$\widetilde{p} \in \widetilde{\mathcal{P}} \quad \Rightarrow \quad \widetilde{\mathcal{P}} \subseteq \mathcal{P} \tag{7.2}$$

Nous avons fait le choix de considérer $\widetilde{p} \in \mathcal{P}$ afin d'être similaire au principe des sacs de mots.

Les définitions 6 et 7 nous ont permis de discrétiser l'espace de poses, à l'aide d'une fonction de distance, notée D_P. Nous allons voir comment exploiter la classification des poses pour reconnaitre des mouvements.

7.2 Histogramme de poses

Dans la section précédente, nous avons vu comment construire des regroupements de poses afin d'avoir des classes d'équivalence. Dans cette section, nous allons étendre et dériver le concept d'histogramme à l'utilisation de poses issues de capture de mouvements, et montrer la force de cette approche pour la reconnaissance d'action humaine. Si le chapitre 5 proposait une approche liée aux trajectoires — qualifiée d'approche locale du mouvement, relativement aux données de capture de mouvement — ce chapitre propose une approche plus globale de l'analyse du mouvement.



(a) Action « Walk » (b) Histogramme de pose

FIGURE 7.1 – Histogramme de poses pour l'action « Walk ».

À l'aide des définitions précédentes, nous introduisons le concept de fréquence d'occurrence d'un élément à l'espace des poses. Pour cela, nous allons évaluer le nombre d'occurrence de chacun des représentants $\widetilde{p}$ pour une action donnée.

Définition 8 Fréquence d'occurence**.** *La fréquence f d'un représentant $\widetilde{p} \in \widetilde{\mathcal{P}}$ dans une action $A = (p_0, \ldots, p_N)$, de durée t_N au cours de l'intervalle temporel ΔT est définit par :*

$$f_A^{\Delta T}(\widetilde{p}) \quad = \quad |\{p_{i,t}/t \in \Delta T \wedge p_{i,t} \sim \widetilde{p}, \forall p_{i,t} \in \mathcal{A}\}| \tag{7.3}$$

$f_A^{\Delta T}(\widetilde{p})$ est la fonction définissant la cardinalité de $\widetilde{p}$ pour une action donnée, A, pendant un temps ΔT, tel que :

$$\Delta T \quad = \quad [t_i, t_j]/t_0 \leq t_i < t_j \leq t_N \tag{7.4}$$

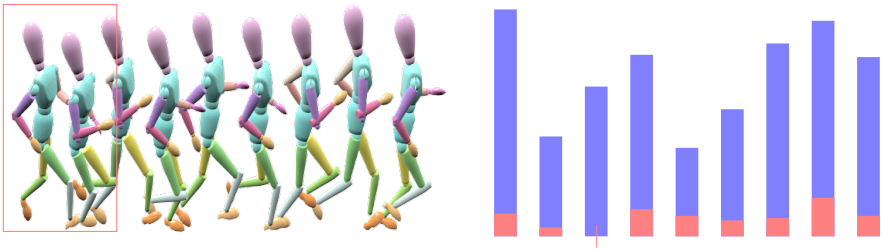
N.B. : *lorsque le temps $t < t_N$, nous considérons la restriction de l'action A à l'intervalle temporel $[0, t]$. Une telle restriction sera exploitée dans la reconnaissance précoce d'actions.*

À partir de la fréquence de chacun des représentants de poses d'une action, nous pouvons avoir une vue statistique d'une action par l'introduction de l'histogramme de ses représentants.

Définition 9 Histogramme de pose**.** *Un histogramme de pose pour une action $A = (p_0, \ldots, p_N)$, de durée t_N, est la distribution statistique des représentants durant son exécution :*

$$\mathcal{H}(A, \mathcal{P}) \quad = \quad \left\{ f_A^{t_N}(\widetilde{p}) / \forall \widetilde{p} \in \widetilde{\mathcal{P}} \right\} \tag{7.5}$$

La définition 9 permet d'avoir une représentation statistique d'une action, en se décorrélant de la temporalité de l'apparition des poses. Comme l'ont montré



(a) Action « Run » (b) Histogramme intégral de pose

FIGURE 7.2 – Histogramme intégral de poses pour l'action « Run », en rouge, et l'histogramme de l'action en bleu. Seules certaines poses sont représentées sur la figure (a). Le rectangle rouge, figure (a) indique les poses utilisées pour la représentation de l'histogramme intégral en figure (b).

nombre de méthodes, [30, 31, 49, 77, 97, 111, 119, 129, 161], l'histogramme est un outil très puissant, à la fois dans la représentation humaine d'un phénomène, mais aussi dans de nombreuses approches automatiques, notamment en traitement d'image. Néanmoins, la suppression complète de la dépendance temporelle peut être problématique dans la reconnaissance d'action. En effet, la succession de sous-ensembles de poses peut décrire une sémantique propre au comportement, et donc avoir un pouvoir discriminant fort lors de la reconnaissance.

Pour résoudre ce problème, nous proposons d'étendre le concept d'histogramme, afin de conserver une part de la temporalité. Nous nous inspirons de la méthode proposée par Ryoo [111], pour introduire le concept d'histogramme intégral de poses.

Définition 10 Histogramme Intégral**.** *Un histogramme Intégral de Poses — noté* $\mathcal{IH}^{\Delta T}(A, \mathcal{P})$*, pour l'action* $A = (p_0, \ldots, p_N)$*, de longueur* t_N*, et l'ensemble des poses* $\mathcal{P}$ *— est l'extension de l'histogramme de poses par une restriction de la fonction temporelle à l'intervalle* ΔT *:*

$$\mathcal{IH}^{\Delta T}(A, \mathcal{P}) \quad = \quad \left\{ f_A^{\Delta T}(\widetilde{p_j}) / \widetilde{p_j} \in \mathcal{P} \right\} \tag{7.6}$$

Un histogramme intégral est un histogramme, dont la fonction de fréquence temporelle a été restreinte à une sous séquence temporelle. L'histogramme intégral d'une action sur l'intervalle $\Delta T = [0, t_N]$ est équivalent à l'histogramme de poses de l'action.

Un histogramme intégral n'est lié qu'à un intervalle. Il est donc possible d'avoir au plus t_N histogrammes intégraux, et au moins un. Ainsi, une action peut être décomposée en un ensemble très grand d'histogrammes intégraux. Une des dé-

compositions les plus intéressantes est celle consistant à obtenir des séries temporelles d'histogrammes intégraux. En considérant une partition d'une action en histogrammes intégraux, il est possible de représenter celle-ci sous forme d'une série temporelle. La partition s'obtient en utilisant la propriété de restriction de la fonction de fréquence d'occurrence — définition 8 — afin de décomposer l'action en sous ensembles de poses qui n'appartiennent qu'à un seul histogramme intégral. Dans cette décomposition « sans chevauchement » des poses entre histogrammes intégraux, une pose n'apparait — par son représentant — que dans un seul histogramme intégral. Notre décomposition respecte les propriétés des partitions.

Considérons l'action A. Il est possible de représenter cette action sous la forme d'un vecteur d'histogrammes intégraux :

$$\mathring{A} = \begin{bmatrix} ha_0 \\ \vdots \\ ha_M \end{bmatrix} \tag{7.7}$$

Où les ha_i sont les histogrammes intégraux de poses de l'action A, restreints afin de créer une partition de l'action, $\mathring{A}$ est la représentation de l'action A sous forme d'une série d'histogrammes, et $M \leq N$ la longueur de la série temporelle, *i.e.* le nombre de sous histogrammes intégraux utilisés.

Nous avons obtenu un vecteur de caractéristiques propres à chaque action. Ce vecteur étant une représentation des actions, nous allons l'utiliser pour catégoriser chacune d'entres elles. Néanmoins, les histogrammes sont déjà des vecteurs de caractéristiques, ainsi $\mathring{A}$ est une matrice de dimension :

$$\dim \mathring{A} = M \times \left|\widetilde{\mathcal{P}}\right| \tag{7.8}$$

$\left|\widetilde{\mathcal{P}}\right|$ est le cardinal de $\widetilde{\mathcal{P}}$, c'est-à-dire le nombre de représentants de l'ensemble des actions. Or ce nombre de représentants dépend de ε, et ne peut être contrôlé directement, comme dans le cas d'une approche non supervisée de type *KMeans*. Dans ce cas, $\left|\widetilde{\mathcal{P}}\right|$ peut être très grand. Dans l'optique de réguler l'explosion de dimension de $\mathring{A}$, il est possible d'augmenter le paramètre ε. Néanmoins, cette approche peut autoriser des associations de poses non pertinentes, en modifiant le pouvoir de discrimination de chaque représentant. Une autre approche couramment utilisée est de réduire la taille des poses en entrée. Au lieu de faire une simplification dirigée par un modèle de la hiérarchie du squelette de capture de mouvements, il est plus efficace d'étudier les variations importantes dans les configurations de squelette, en ne conservant que les plus significatives. C'est cette dernière solution que nous avons choisi de mettre en œuvre, et qui est présentée dans la section

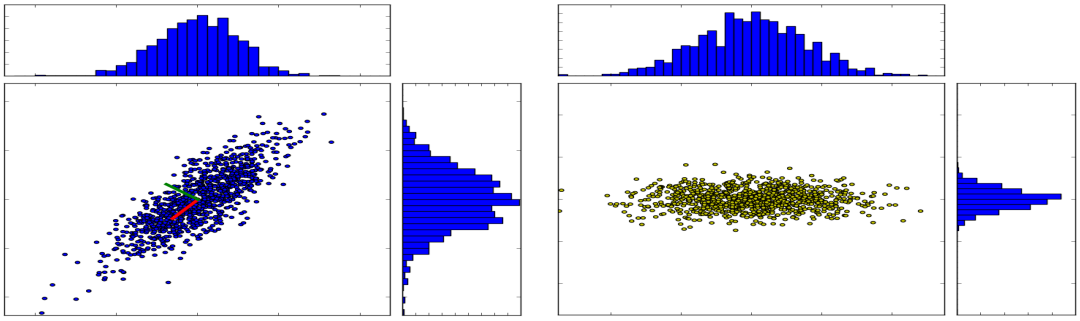
suivante, pour limiter la dimension de $\mathring{A}$ en limitant indirectement la cardinalité de $\widetilde{\mathcal{P}}$.

7.3 Réduction des poses

Utiliser l'ensemble des articulations afin de représenter une pose est une solution peu efficace. En effet, toutes les articulations ne sont pas de même importance en fonction du mouvement considéré. Ainsi, comme l'ont montrés Yao et al. [155] et Raptis et al. [106] une réduction du nombre des articulations impliquées peut être envisagée dès lors que l'on dispose d'une information tridimensionnelle fiable. Dans ce contexte, l'approche la plus utilisée est l'Analyse en Composantes Principales (ACP), ou *Principal Component Analysis*, (PCA), proposée par Pearson [99].

Le principe de l'analyse en composantes principales est de « trouver » une base discriminant « au mieux » les variables initiales. En d'autres mots, il s'agit de changer de point de vue pour avoir une autre représentation des données. Contrairement à d'autres méthodes, le but n'est pas de se reporter à un espace de dimension supérieur, comme Yao et al. [156] par exemple, mais au contraire de trouver les axes d'explication des plus grandes variations. La figure 7.3a représente une distribution aléatoire normale, centrée en $(0, 0)$ et de variance $[(1; 1), (1; 1, 5)]$. La dispersion se fait suivant les axes escomptés. La projection des histogrammes sur chacun des axes « originaux » n'est pas en mesure de représenter efficacement l'orientation de cette variable. La représentation des deux premières composantes principales — issue du calcul de l'ACP — permet de visualiser la dispersion de la variable. La figure 7.3b montre la même variable aléatoire, dont les points ont été exprimés dans le repère de la base déterminée par l'ACP.

Il est possible d'appliquer l'ACP aux poses d'une action. En effet, il s'agit de trouver les axes principaux de représentation d'une variable multidimensionnelle, le squelette d'animation. L'objectif de cette approche est de représenter les données en ne tenant compte que des « parties les plus importantes », en faisant la supposition que cela permette de réduire la quantité de données traités, mais aussi de discriminer plus fortement les actions les unes des autres. Si l'on applique ce principe aux poses d'une activité, composée de plusieurs actions, on obtient les résultats présentés en figures 7.4a et 7.4b. Dès que l'on se confronte à la représentation des poses dans un espace, même projeté, on est limité par la visualisation possible. En effet, représenter plus de 3 dimensions de façon claire est quasiment impossible en 2D. Ainsi, nous n'affichons que les deux et trois premières composantes, qui sont les plus importantes, mais pas nécessairement suffisante dans le

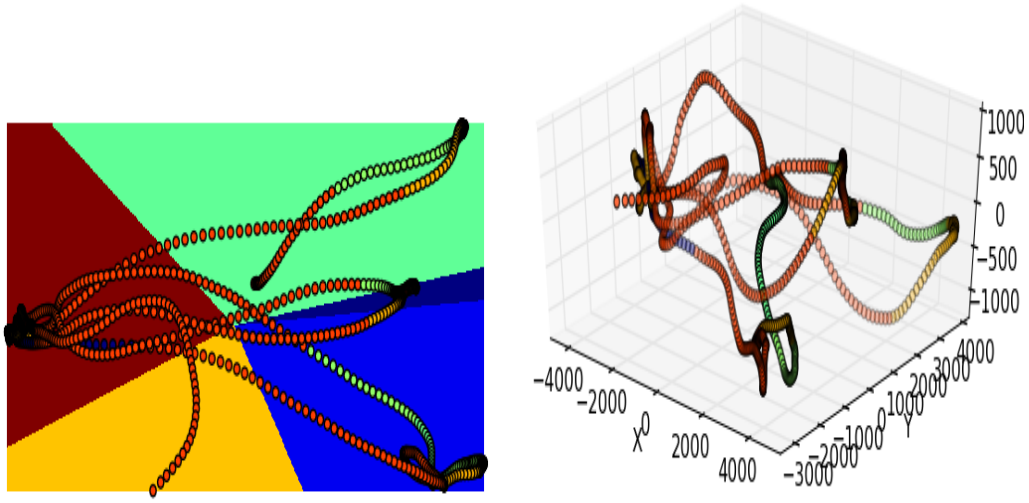


(a) Distribution initiale (b) Distribution transformée

FIGURE 7.3 – Représentation des histogrammes projetés d'une variable aléatoire et des axes des deux composantes principales, ainsi que la même variable transformée suivant les deux premiers axes principaux.

cas d'un espace à 86 dimensions, comme c'est le cas dans la figure 7.4. L'analyse en composantes principales n'est pas la seule solution pour discriminer les actions. On peut citer les approches par espaces latents notamment, voir Yao et al. [156] et le chapitre 2 pour plus d'informations.

Réduire le nombre de dimension des poses permet de conserver un critère de catégorisation stricte, *i.e.* ε petit, tout en réduisant la cardinalité de l'histogramme de poses. En effet, en réduisant la dimension des poses en entrée, on obtient un espace de plus petite dimension pour représenter chacune de ces poses. Nous avons donc la possibilité de limiter la dimension de $\mathring{A}$, sans toutefois pouvoir la contrôler de façon exacte, mais en gardant le même critère de similarité entre poses.



(a) 2 composantes principales (b) 3 composantes principales

FIGURE 7.4 – Analyse en Composantes Principales (ACP) sur l'activité 0-12 du jeu de données TUM. Chaque point est une pose, chaque couleur représente une action différente.

Chapitre 8

Reconnaissance par histogrammes

Sommaire

8.1 Comparaison d'histogrammes

Les histogrammes de poses, qu'ils soient intégraux ou non, complets ou partiels, sont des histogrammes. En ce sens, il est possible de les comparer avec les méthodes classiques dévolues aux histogrammes. Nous utilisons ses méthodes car elles ont fait leurs preuves tant en robustesse qu'en efficacité pour la comparaison de deux histogrammes. En transposant ces méthodes dans le domaine des histogrammes de poses, nous allons nous attacher à définir un coût entre deux actions, représenté par la distance entre les deux histogrammes de ces actions.

Définition 11 Coût entre actions. *Soit deux actions A et B, représentées par leurs histogrammes respectifs, $\mathcal{H}(A,\mathcal{P}) = \mathcal{H}_A$ et $\mathcal{H}(B,\mathcal{P}) = \mathcal{H}_B$, ou par leur histogrammes intégraux respectifs, $\mathcal{IH}^{\Delta T_A}$ et $\mathcal{IH}^{\Delta T_B}$, la similarité entre les actions A et B est définie par :*

$$Cost(A,B) = D_H(\mathcal{H}_A,\mathcal{H}_B) \quad \textit{(Histogrammes de poses)} \tag{8.1}$$
$$Cost(A,B) = D_H(\mathcal{IH}^{\Delta T_A},\mathcal{IH}^{\Delta T_B}) \quad \textit{(Histogrammes intégraux)} \tag{8.2}$$

Où D_H est une distance définie entre deux histogrammes,

De nombreuses distances ont été définies afin de comparer des séries statistiques de façon générale. Néanmoins, des distances plus adaptées aux histogrammes eux-mêmes existent. La plupart de ses distances sont faciles à calculer, mais, elles ne sont pas toutes bornées, ce qui peut poser des problèmes quant à leur mise en œuvre dans des systèmes informatiques. Afin de comparer efficacement des actions entre elles, nous allons faire le choix d'une distance bornée, *i.e.* une distance dans l'intervalle $[0;1]$.

Certaines distances classiques entre histogrammes sont rappelées ci-dessous.

$$\text{Corrélation}\,(H_1,H_2) = \frac{\sum_i H'_1(i)\cdot H'_2(i)}{\sqrt{\sum_i H'^2_1(i)\cdot H'^2_2(i)}} \tag{8.3}$$

$$\text{Intersection}\,(H_1,H_2) = \sum_i \min\left(H_1(i),H_2(i)\right) \tag{8.4}$$

$$\text{Minkovski}\,(H_1,H_2) = \sqrt[p]{\sum_i |H_1(i)-H_2(i)|^p} \tag{8.5}$$

$$\text{Où : } p = \begin{cases} 1 & \text{si on utilise la distance } L_1 \\ 2 & \text{si on utilise la distance } L_2 \end{cases}$$

$$\text{Chi-Square}\,(H_1,H_2) = \sum_i \frac{\left(H_1(i)-H_2(i)\right)^2}{H_1(i)+H_2(i)} \tag{8.6}$$

$$\text{Bhattacharyya}\,(H_1,H_2) = \sqrt{1-\sum_i \frac{\sqrt{H_1(i)\cdot H_2(i)}}{\sqrt{\sum_i H_1(i)\cdot\sum_i H_2(i)}}} \tag{8.7}$$

$$\text{Où : } H'_k(i) = H_k(i) - \frac{1}{N}\times\left(\sum_j H_k(j)\right) \tag{8.8}$$

$$\text{Earth Mover's Distance}\,(H_1,H_2) = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n} d_{ij}f_{ij}}{\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}} \tag{8.9}$$

Où : d_{ij} est la distance entre $H_1(i)$ et $H_2(j)$, basiquement $d_{ij} = |i-j|$, et f_{ij} est le flux entre $H_1(i)$ et $H_2(j)$.

La corrélation entre deux histogrammes (équation 8.3) est bornée entre $[-1;1]$, où -1 indique une dissimilarité totale, 1 une similarité parfaite, et 0 une association aléatoire. L'intersection d'histogramme (équation 8.4) est bornée entre $[0;1]$, si les histogrammes sont normalisés à 1. Un score de 1 indique que les histogrammes sont identiques, et 0 qu'ils sont totalement étrangers l'un à l'autre. La distance de Minkowski est une distance non bornée, qu'elle soit utilisée avec la distance de Manhattan ($p=1$) ou avec la distance Euclidienne ($p=2$). Une identité entre les

histogrammes est traduite par une distance de 0. Le Chi-Square de Pearson [98] est une distance définie entre $[0; +\infty[$, qui vaut 0 en cas de correspondance parfaite. Cette distance n'est pas bornée, ce qui peut poser des problèmes dans les comparaisons futures entre histogrammes. Sa taille dépend de la taille de l'histogramme que l'on évalue, il serait donc possible de la borner, mais au détriment de l'évolution des histogrammes, notamment des histogrammes intégraux. La distance de Bhattacharyya [20] est définie sur l'intervalle $[0; 1]$, où 0 indique une correspondance parfaite des deux histogrammes et 1 une dissimilarité complète. Enfin la distance *Earth Mover's Distance* (EMD), proposée par Rubner et al. [110], est une distance qui n'est pas calculée mode à mode pour les histogrammes, mais qui détermine « le coût du moindre effort » pour passer d'une série statistique (histogramme) à une autre. Elle opère traditionnellement par une minimisation de simplex et possède une complexité exponentielle dans le pire des cas et super-cubic en moyenne ($\mathcal{O}(N^3) \cap \mathcal{O}(N^4)$). Plusieurs approches ont rendu le calcul de cette distance plus « acceptable », citons notamment Ling et Okada [72] et Pele et Werman [100] en utilisant respectivement une distance L_1 ou en ajoutant une saturation aux bornes de la distance. Néanmoins, pour notre contexte, le coût reste trop important. De plus, l'évaluation de la distance entre modes de l'histogramme est un problème à part entière dans ce cas, qu'il n'est pas facile de résoudre.

Dans la suite, nous allons utiliser la distance de Bhattacharyya afin de comparer nos histogrammes d'actions. Or, comme notre but est de fournir une comparaison en ligne des actions, nous allons nous attacher à comparer des actions par l'intermédiaire des histogrammes intégraux de ces dernières. Pour cela, nous produisons la formulation suivante de la distance de Bhattacharyya. Notons toutefois que les distances présentées précédemment peuvent, de la même manière, être rapportées à des comparaisons entre histogrammes intégraux.

Définition 12 Distance de Bhattacharyya entre histogrammes intégraux**.** *Soit deux actions A et B. La distance de Bhattacharyya entre les histogrammes intégraux de ces actions s'exprime par :*

$$\mathit{Bhattacharyya}(\mathcal{H}^{\Delta T_A}(A, \mathcal{P}), \mathcal{H}^{\Delta T_B}(B, \mathcal{P})) = \sqrt{1 - \sum_{\widetilde{p_j}} M} \tag{8.10}$$

où :

$$M = \frac{\sqrt{f_A^{\Delta T_A}(\widetilde{p_j}) \cdot f_B^{\Delta T_B}(\widetilde{p_j})}}{\sqrt{\sum_{\widetilde{p_j}} f_A^{\Delta T_A}(\widetilde{p_j}) \cdot \sum_{\widetilde{p_j}} f_B^{\Delta T_B}(\widetilde{p_j})}} \tag{8.11}$$

Lorsque ΔT_A, respectivement ΔT_B, est la durée de l'action A, respectivement de l'action B, alors la distance entre actions par histogrammes intégraux est égale à la distance entre les actions par histogrammes « classiques ».

8.2 Comparaison incrémentale d'histogrammes

Pour comparer des histogrammes intégraux, il est nécessaire de définir une découpe des histogrammes d'actions en une série d'histogrammes incrémentaux. Or, la qualité de la découpe conditionne le résultat de comparaison. Il est donc nécessaire de fournir une découpe optimale en fonction de l'objectif visé. Pour cela, nous avons fait le choix d'évaluer chacune des découpes afin de trouver la meilleure. Cette recherche exhaustive est coûteuse à mettre en œuvre, et une approche plus pragmatique doit être envisagée. Nous avons rapporté notre problème de découpe à celui de trouver un alignement optimal entre les séries d'histogrammes afin que chacun des scores de comparaison — comparaison des histogrammes intégraux issus de chacune des actions — soit optimaux. Ainsi, trouver une découpe optimale en histogrammes intégraux revient à aligner chacun des sous histogrammes de la série entre eux.

8.2.1 Rappel sur *Dynamic Time Warping*

L'algorithme de *Dynamic Time Warping* (DTW) est l'un des algorithmes les plus populaires pour l'alignement de deux signaux temporellement liés. Il s'agit d'un algorithme reposant sur la « Programmation Dynamique », concept proposé par Bellman [19], qui explore l'ensemble des alignements entre les deux signaux afin de trouver les « recouvrements » qui permettent d'obtenir le meilleur « score » de similarité. Pour cela, l'algorithme évalue toutes les configurations possibles afin de trouver l'alignement de plus faible coût, appelé chemin optimal.

Formellement, soit $A = (a_1, a_2, \cdots, a_N)$ et $B = (b_1, b_2, \cdots, b_M)$ deux séries temporelles échantillonnées régulièrement, dans un même espace Φ, *i.e.* $A, B \in \Phi$. Soit d une distance locale, définit comme suit :

$$d : \Phi \times \Phi \to \mathbb{R}^+ \tag{8.12}$$

d est souvent appelé la fonction de coût, en référence à la programmation dynamique. Le but est de définir l'association de chacun des points de A avec chacun des points de B, par l'intermédiaire d'un chemin de plus faible coût, de longueur L. Ce chemin est défini par $p = (p_1, p_2, \cdots, p_L)$, où : $p_l = (n_l, m_l) \in [1 : N] \times [1 : M]$ et $l \in [1 : L]$, en fixant les contraintes suivantes :

1. **Condition aux frontières** : les premiers et derniers points sont alignés, $p_1 = (1, 1)$ et $p_L = (N, M)$
2. **Condition de monotonie** : l'ordre des points est conservé, $n_1 \leq n_2 \leq \cdots \leq n_L$ et $m_1 \leq m_2 \leq \cdots \leq m_L$
3. **Condition de pas** : le chemin ne fait pas de saut dans l'alignement des signaux, $p_{l+1} - p_l \in \{(1, 0), (0, 1), (1, 1)\}$ pour $l \in [1 : L - 1]$

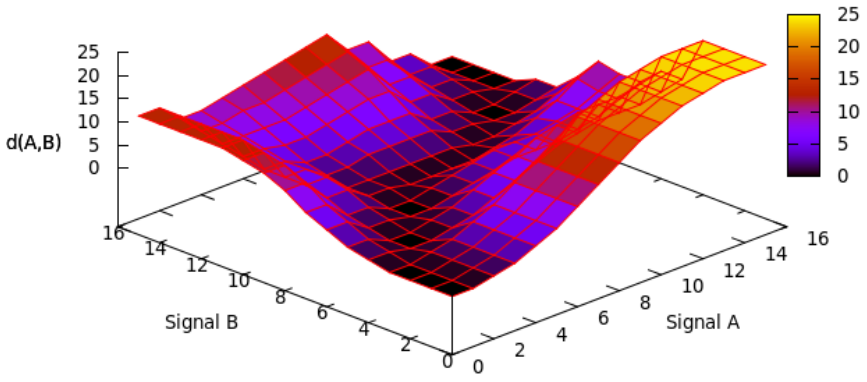


FIGURE 8.1 – Représentation de la matrice de coût sur l'exemple de Salvador et Chan [115]. Le chemin optimal passe par la vallée.

Le calcul du chemin optimal se fait par l'utilisation d'une matrice de coût $C \in \mathbb{R}^{N \times M}$, $C(n,m) = d(a_n, b_m)$, où $d(a_n, b_m)$ est défini par l'équation 8.12, explicitant toutes les distances entre points. Voir la figure 8.1 pour une représentation de la matrice de coût.

Le chemin complet c_p de l'alignement entre deux actions A et B est défini par :

$$c_p(A,B) \quad = \quad \sum_{l=1}^{L} d\left(a_{n_l}, b_{m_l}\right) \tag{8.13}$$

Le chemin optimal, $c_p^\star$ est donc défini par :

$$c_p^\star(A,B) \quad = \quad \min\left\{c_p(A,B) \mid p \text{ est un chemin}\right\} \tag{8.14}$$

Par abus de langage, nous dirons que le coût de l'alignement par *Dynamic Time Warping* est assimilé au chemin de moindre coût, soit :

$$DTW(A,B) \quad = \quad c_p^\star(A,B) \tag{8.15}$$

L'exploitation complète de la matrice peut être très onéreuse. Diverses solutions d'optimisation ont été proposées, comme la bande de Sakoe et Chiba [114] ou le parallélogramme de Itakura [51]. Le lecteur intéressé pourra se référer à Müller [86, chapitre 4].

8.2.2 Coût incrémental d'histogrammes

D'après l'équation 7.7, il est possible de décomposer une action en une série de sous histogrammes intégraux. Dès lors, on cherche à définir la décomposition optimale entre les deux histogrammes intégraux afin de ne pas être tributaire de la décomposition dans la comparaison de deux actions entres elles. Trouver la décomposition optimale nécessite d'explorer l'ensemble des décompositions possibles. Pour cela, nous introduisons la définition 13.

Définition 13 Décomposition en sous actions**.** *Soit $A = (p_0, \ldots, p_n)$ une action composée de $n+1$ poses. L'ensemble des décompositions de A en sous actions s'écrit :*

- *A a 1 pose : $A = (p_0)$*

$$Decomp(A) \quad = \quad \{(p_0)\} \tag{8.16}$$

- *A a 2 poses : $A = (p_0, p_1)$*

$$Decomp(A) \quad = \quad \{(p_0 p_1), (p_0)\ (p_1)\} \tag{8.17}$$

- *A a 3 poses : $A = (p_0, p_1, p_2)$*

$$\begin{aligned} Decomp(A) \quad = \quad & \{(p_0 p_1 p_2), (p_0 p_1)\ (p_2), \\ & (p_0)\ (p_1 p_2), (p_0)\ (p_1)\ (p_2)\} \end{aligned} \tag{8.18}$$

$\vdots$

- *A a $n+1$ poses : $A = (p_0, p_1, \ldots, p_n)$*

$$Decomp(A) \quad = \quad \bigcup_{s \in Decomp(A \setminus \{p_n\})} \{Concat\,((s), \{p_n\})\} \cup ((s), \{p_n\}) \tag{8.19}$$

où :

$$Concat\,((p_a, \ldots,\ p_b)^\star (p_i, \ldots,\ p_{n-1}), \{p_n\}) \quad = \quad \{(p_a, \ldots,\ p_b)^\star\,(p_i \cdots p_{n-1} p_n)\} \tag{8.20}$$

avec : $0 \leq a \leq b < i \leq n-1$, et pour $A = (p_0, p_1, \ldots, p_{n-1}, p_n)$:

$$A \setminus \{p_n\} \quad = \quad (p_0, p_1, \ldots, p_{n-1}) \tag{8.21}$$

c'est-à-dire l'action A privée de la pose p_n.
N.B. : *la formulation de la fonction Concat utilise les expressions régulières afin d'être le plus générique possible.*

Ainsi, une décomposition d'une action A en sous actions pourra être représentée par un vecteur :

$$A = \underbrace{(p_1)}_{h_0}, \underbrace{(p_2 p_3)}_{h_1}, \cdots, \underbrace{(p_{M-1} p_M)}_{h_N} \quad \Rightarrow \quad \mathring{A} = \begin{bmatrix} h_0 \\ h_1 \\ \vdots \\ h_N \end{bmatrix} \tag{8.22}$$

Pour effectivement calculer le coût entre deux actions, nous allons utiliser l'ensemble des décompositions possibles pour les actions A et B, tel qu'il est proposé par la définition 13, en nous inspirant de la méthode de *Dynamic Time Warping* de Sakoe et Chiba [113]. La décomposition optimale est définie par la formulation récursive suivante :

$$\begin{aligned} Cost^{\star}\left(h_0^A, h_0^B\right) &= Cost\left(h_0^A, h_0^B\right) \\ Cost^{\star}\left(h_i^A, h_j^B\right) &= Cost\left(h_i^A, h_j^B\right) \\ &+ \min \left\{ \begin{array}{l} Cost\left(h_{i-1}^A, h_j^B\right), \\ Cost\left(h_i^A, h_{j-1}^B\right), \\ Cost\left(h_{i-1}^A, h_{j-1}^B\right) \end{array} \right\} \end{aligned} \tag{8.23}$$

Où, $\mathring{A} = [h_0^A, \ldots, h_N^A]^T$ et $\mathring{B} = [h_0^B, \ldots, h_M^B]^T$ sont les représentations des actions A et B respectivement, par des histogrammes intégraux.

Posons $A^{\star}$ comme la décomposition optimale de A et $B^{\star}$ comme la décomposition optimale de B lorsque l'on compare les actions A et B ensemble. Nous définissons le coût entre les décompositions des actions A et B par la formulation suivante.

$$Cost(A^{\star}, B^{\star}) = Cost^{\star}(A, B) \tag{8.24}$$

8.3 Multiples hypothèses

La reconnaissance d'actions humaines est tributaire de la variabilité inhérente à chaque réexécution. En effet, peu d'êtres humains sont capables de reproduire à l'identique leurs actions, exécutions après exécutions. Du fait du bruit que peut introduire une discrétisation d'actions par un système de capture de mouvements, il est nécessaire de fournir une solution autorisant de la variabilité. Dans ce cadre, nous avons proposé les histogrammes qui s'attachent à une approche plus statistique de l'action. Dans ce contexte, une action avec des variations occasionnelles

sera prise en compte. C'est-à-dire qu'une approximation dans une partie de l'action n'affectera pas de façon importante l'ensemble du score de reconnaissance. Néanmoins, il reste une possibilité de grande variation. Lorsque l'on exécute une action complexe comme ouvrir un tiroir, il est peu probable que l'utilisateur fasse le même mouvement. Il est aussi peu probable que le nombre de possibilités soit très grand. Dans ce cas, il est possible de créer un sous ensemble d'exécution pour cette action. Dans ce cadre, nous parlerons d'hypothèses pour une action. Chaque instance d'une action, dans l'ensemble d'apprentissage, conduit à une hypothèse différente dans notre modèle pour cette action. Par exemple, chacune des « possibilités » d'ouvrir un tiroir, est vue par notre système, non pas comme une instance différente, mais comme une « autre façon de faire » cette action.

De façon naïve, il semble plus judicieux de faire un modèle intégrant les différentes exécutions possibles pour une action. Or, faire une mixture d'histogrammes, ou faire une moyenne entre deux instances différentes conduit à un modèle trop complexe, voire impossible à exprimer de façon générale. Dans ce contexte, nous faisons le choix de considérer un ensemble d'hypothèses pour chacune des actions apprises.

8.3.1 Coût en multi-hypothèses

Le fait de considérer de multiples hypothèses conduit à avoir un ensemble virtuel d'actions beaucoup plus important. Chacune des actions possède un ensemble d'hypothèses. Dans ce cas, un action A — respectivement B — possédant N hypothèses — respectivement M hypothèses — est représentée par N histogrammes intégraux — respectivement M histogrammes intégraux — chacun pour une hypothèse. Dès lors, la distance entre l'action A et l'action B est définie par l'équation suivante :

$$Cost_{multi}(A,B) = \min_{\substack{h^A \in H^A \\ h^B \in H^B}} \left\{ Cost^{\star}(h^A, h^B) \right\} \tag{8.25}$$

Où : H^A, respectivement H^B, est l'ensemble des hypothèses de l'action A, respectivement de l'action B.

8.3.2 Extraction des représentants multi-hypothèses

Il ne s'agit pas de conserver chacun des exemples du jeu d'apprentissage afin d'avoir une hypothèse pas action. Cela ne serait efficace ni en mémoire ni en temps de calculs. De plus, cela ne garantit pas une meilleure discrimination des actions les unes par rapport aux autres. L'extraction des hypothèses pertinentes

peut recouvrir différentes formes. Il s'agit de trouver un représentant par classe, et donc le principe des *examplar* est applicable. Nous avons besoin d'effectuer un regroupement des hypothèses d'actions en tenant compte du fait que les représentants doivent être dans la classe. En effet, il est difficile de produire un modèle mixte d'histogramme, et une « simple » moyenne, classe par classe, n'a que peu de sens dans notre cadre. Dans ces conditions, nous avons fait le choix de la méthode dite des *K-medoids*, initialement proposée par Kaufman et Rousseeuw [56], et rappelée par l'algorithme 8.1. Le principe de l'algorithme *K-medoids* est de trouver un ensemble de groupes dont le centre est un des éléments du groupe. Pour cela, il sélectionne aléatoirement un ensemble k de « potentiels » centroïds (*i.e. medoids*). Il affecte chaque point au centroïd le plus proche — en considérant une distance, généralement euclidienne, ici définie par l'équation 8.23 — puis, la distance moyenne des éléments au sein d'un même centroïd est évaluée. Ensuite, des permutations entre centroïds et non centroïds sont effectuées, afin de créer de nouveaux regroupements. Le regroupement de moindre coût, *i.e.* celui de variation interne la plus petite, est conservé. Ceci se poursuit jusqu'à ce qu'il n'y ai plus de variation dans l'ensemble des centroïds.

Algorithme 8.1: K-Medoids

```
Input : k : nombres de médoïds
Input : Dataset : hypothèses à regrouper pour une action
[Medoids] ← Aléa(k, Dataset) ;
dist ← Cost*([Medoids]) ;
repeat
    foreach v ∈ Dataset and v ∉ [Medoids] do
        v → PlusProche([Medoids], dist) ;
    end
    foreach m ∈ [Medoids] do
        foreach v ∈ Dataset and v ∉ [Medoids] do
            Échange( m, v ) ;
            Cost*([Medoids]) ;
        end
    end
    [Medoids] ← SélectionCostMin() ;
until stabilité dans [Medoids];
```

Grâce à l'algorithme 8.1 nous pouvons contrôler le nombre d'hypothèses par actions. De plus, il est possible d'étudier la dispersion des hypothèses au sein d'un regroupement, par le calcul de la moyenne et de l'écart type, par exemple. Comme le regroupement en hypothèse se fait de façon indépendante par actions,

nous pouvons utiliser différentes valeurs de k pour chaque actions, et faire varier ce paramètre en fonction de la variation intra-classe présente par actions.

Chapitre 9

Résultats

Sommaire

Afin d'évaluer la validité du processus proposé, nous avons effectué des tests dans différents contextes d'utilisation. Tout d'abord, nous présentons des résultats concernant la reconnaissance d'actions, ceci dans une optique plus « jeux vidéo ». Ensuite, nous exposerons la validité de la méthode pour la reconnaissance d'activités, notamment dans une volonté de *monitoring* d'intérieur, cuisine et salon. Enfin, nous analyserons le comportement des histogrammes lorsque ceux-ci sont confrontés à des reconnaissances précoces d'actions.

9.1 Reconnaissance d'actions

Nous avons testé notre système sur le jeu de données HDM, proposé par Müller et al. [88], ainsi que sur le jeu de données MoCap CMU [84]. Voir l'annexe A pour plus de précision sur la construction des différents jeux de données utilisés.

Des tests de la reconnaissance à l'aide d'une hypothèse unique, ainsi qu'à l'aide de multiples hypothèses ont été fait avec le protocole dit de *Cross validation*. Chacune des exécutions d'action a été utilisé pour l'apprentissage, tandis que les autres étaient utilisées pour la reconnaissance. Dans le cadre multi-hypothèses, nous avons

partitionné aléatoirement l'ensemble du jeux de données entre apprentissage et reconnaissance. Trois hypothèses, considérées comme les plus discriminantes ont été sélectionné par notre méthode parmi l'ensemble d'apprentissage (*training set*). L'ensemble des hypothèses de reconnaissance (*testing set*) ont été utilisé pour évaluer la méthode. La figure 9.1 montre les matrices de confusion pour le cas d'une hypothèse unique 9.1(a),(b) ainsi que pour le cas multi-hypothèses 9.1(c),(d).

Les figures 9.1a et 9.1b présentent un fort taux de similarité entre des actions pourtant assez éloignées les unes des autres. Cela vient de la relative faiblesse de l'apprentissage à l'aide d'un seul histogramme. Notons que nous avons utilisé une distance entre pose de $\varepsilon = 1.0$cm dans l'ensemble de nos tests (voir la section 9.4 pour une justification précise de cette valeur). Notre solution est assez discriminante entre des actions « géométriquement » éloignées. Ainsi, la distinction entre « *Walk* » et les autres actions n'utilisant pas de déplacement, *i.e.* toutes sauf « *Run* », est très importante. *A contrario*, les actions mettant en œuvre les mains, « *Boxing* », « *Drink* » ou «*Eat* », sont vues comme semblables.

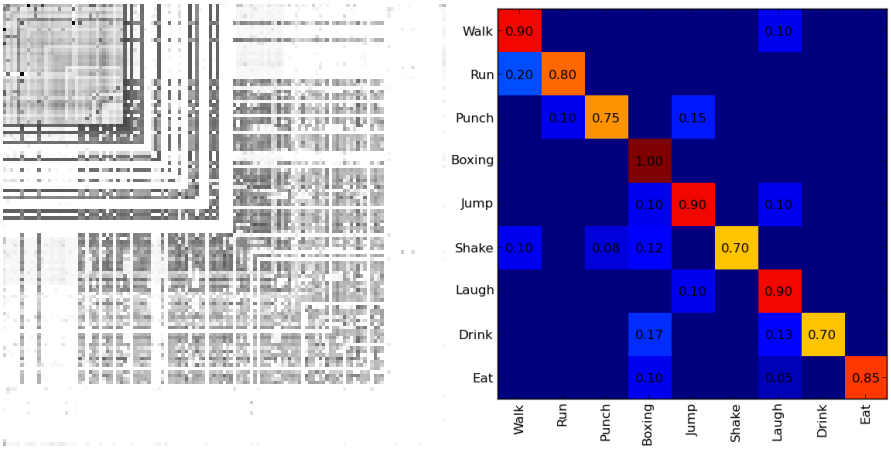
Nous avons calculé la précision moyenne (*Accuracy*) de reconnaissance, équation 9.1, sur toutes les configurations. Nous rappelons ci-dessous les formules couramment utilisées pour évaluer de la pertinence d'une classification :

$$\text{Précision} = \frac{\text{Vrai Positif} + \text{Vrai Négatif}}{\text{Vrai Pos.} + \text{Vrai Nég.} + \text{Faux Pos.} + \text{Faux Nég.}} \tag{9.1}$$

$$\text{Exactitude} = \frac{\text{Vrai Positif}}{\text{Vrai Positif} + \text{Faux Positif}} \tag{9.2}$$

Nous avons aussi étudié la précision (*accuracy*) de notre solution. En comparaison avec l'état de l'art, sur ces données, nous obtenons les scores de précision de $67,89\%$ en simple hypothèse, et 96.67% en multiple hypothèse, alors que la meilleure solution proposée, Müller et al. [88] obtient un score de 80% dans une configuration comparable à notre multi-hypothèse pour le jeu de données HDM. Concernant le jeu de données CMU, nous obtenons les scores de $86,63\%$ (simple hypothèse) et $90,92\%$ (multi-hypothèses) alors que Müller et al. [88] obtient un score de 75% sur un jeu semblable. Il est à noter que nous n'utilisons pas le même jeu de données CMU que celui de Müller et al. [88]. En revanche, notre solution est totalement automatique, alors qu'ils utilisent une sélection manuelle de poses clés (*keyframes*).

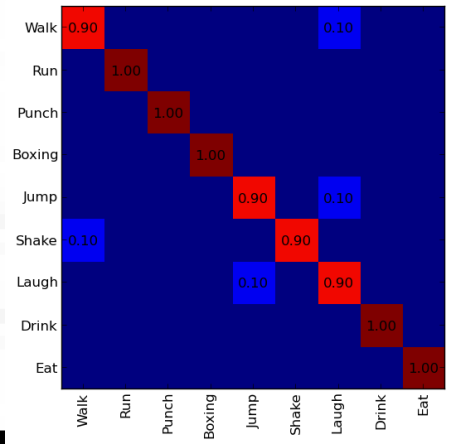
De plus, dans le jeu de données HDM, il y a 130 actions différentes destinées à l'animation de personnages, et non à identifier des comportements humains. Il est donc segmenté selon des principes de transitions entre actions : « la marche avec quatre pas, en partant du pied gauche », *etc.* Nous cherchons, dans notre travail, à faire la différence entre la course et la marche, non à identifier le nombre de pas, ou le pied d'appel. Nous avons donc proposé un regroupement des 130 actions en un ensemble composé de 33 actions, décrivant mieux les comportements humains. Ceci



(a) HDM (130 classes)

(b) CMU

(c) HDM multi (33 classes)



(d) CMU multi

FIGURE 9.1 – Matrices de confusion sur les jeux de données HDM et CMU. (Représentation logarithmique des couleurs)

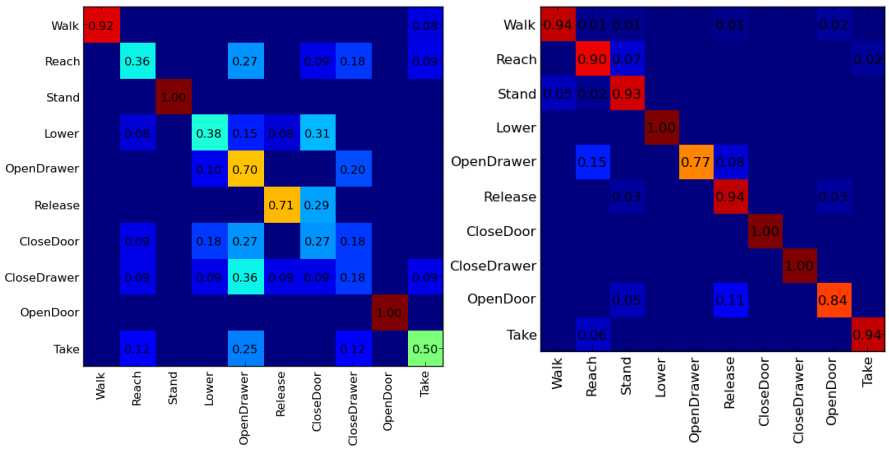
permet d'ajouter de la variance intra-classe aux actions, ce qui est plus en accord avec le but que nous nous sommes fixé, à savoir reconnaitre des comportements humains réels.

9.2 Reconnaissance d'activités

Dans un objectif de comparaison de notre solution avec des méthodes de reconnaissance basées activités, nous utilisons le jeu de données TUM, proposé par Tenorth et al. [126]. Il s'agit d'un jeu de données de personnes dressant une table pour un repas. Un ensemble de 20 séquences sont disponible, chacune annotée. Nous comparons les résultats avec les 4 méthodes actuellement disponible dans l'état de l'art : Tenorth et al. [126], qui ont proposé le jeu de données et fournissent une première reconnaissance, Krausz et Bauckhage [62], Yao et al. [155] et notre solution, Barnachon et al. [10]. Dans toutes ces solutions, le même protocole de test a été appliqué. Les 13 séquences suivantes ont été utilisées pour l'apprentissage : {0-0, 0-1, 0-3, 0-7, 0-9, 0-12, 1-0, 1-1, 1-2, 1-3, 1-4, 1-5, 1-7}. Les 7 séquences restantes sont affectées au jeu de test : {0-2, 0-4, 0-6, 0-8, 0-10, 0-11, 1-6}. Le nombre d'actions impliquées dans la mise en place d'une table a été étendu à 10, pour être en conformité avec Yao et al. [155]. Ainsi, nous séparons : « Walking » et « Standing » en deux actions distinctes. Nous obtenons un score de précision de $92,56\%$ alors que Müller et al. [88] obtiennent $62,77\%$ avec des capteurs RFID en sus, Krausz et Bauckhage [62] obtiennent 67% uniquement avec la capture de mouvements, Yao et al. [155] obtiennent $81,50\%$ avec une utilisation conjointe de la capture de mouvements et des points caractéristiques 2D, issus des images des vidéos. La figure 9.2 présente la matrice de confusion pour ce jeu de données.

Actuellement, aucune solution n'est en mesure de fournir une reconnaissance des actions à l'aide d'un apprentissage plus léger. Afin de fournir une évaluation complète de notre solution, nous avons construit différents classifieurs pour tester la validité de notre approche face à des solutions de type *Machine Learning*. (SVM, kNN, Tree, voir Bishop [21] pour les détails de fonctionnement de ces classifieurs.) Le protocole étant de n'utiliser qu'une seule séquence en guise de jeu d'apprentissage, et toutes les autres séquences en guise de reconnaissance. Nous avons fait la moyenne des scores de précision avec chacune des séquences utilisées comme jeu d'apprentissage. Notre solution surpasse légèrement ces classifieurs, dans toutes les configurations (voir le tableau 9.1). Pour montrer la validité de ces solutions *ad hoc* nous avons aussi fourni leur précision sur le protocole de test complet, *i.e.* les 13 séquences précédentes pour l'apprentissage, et les 7 autres pour le test. Il est à noter que la plupart de nos classifieurs surpassent les premières méthodes de l'état de l'art, Tenorth et al. [126] et Krausz et Bauckhage [62].

L'ensemble de ces résultats est représenté dans le tableau 9.1, ainsi qu'un ré-



(a) TUM simple hypothèse (b) TUM multiples hypothèses

FIGURE 9.2 – Matrices de confusion sur le jeu de données TUM.

capitulatif sur les score de précision de l'état de l'art sur ces jeux de données et de notre solution, dans le tableau 9.2.

La figure 9.3 montre une reconnaissance d'une activité dans la configuration d'un apprentissage unique. Dans la plupart des cas, l'action en cours est assez évidente, *i.e.* son score est bien plus élevé que tous les autres. Néanmoins, certaines activités plus litigieuses, ou pouvant être faite avec de grandes variations d'une exécution à l'autre, sont quelque peu ambigües. Cela s'explique par l'absence d'information concernant la variation interne des actions, dû au fait qu'elles ne sont apprises que sur un *examplar*. Par ailleurs, celui-ci n'est pas garanti comme étant le plus représentatif.

La figure 9.4 montre la reconnaissance d'une activité avec l'apprentissage multiples hypothèses. Les pics de reconnaissance sont clairement identifiés et les actions sont mieux discriminées. Notons toutefois, que la méthode par histogrammes intégraux ne donne que peut de résultats significatifs avant d'avoir vu la moitié de l'action. Ceci se traduit par des plages vides dans la reconnaissance, qui pourront *a posteriori* être associées à l'action reconnue si nécessaire.

9.3 Reconnaissance en ligne

L'avantage des histogrammes intégraux, en comparaison de bien des autres méthodes, est la possibilité de fournir une réponse de façon incrémentale. En effet, à chaque pose, il est possible de construire une représentation intermédiaire, évolutive, de l'action — par le biais d'un histogramme intégral — et ainsi d'avoir une

	TUM (one-vs-one)	TUM (full)
SVM(RBF+K)	49.52%	54.67%
kNN	51.41%	71.34%
Tree	26.67%	75.28%
Tenorth et al. [126]	–	62.77%
Krausz et Bauckhage [62]	–	67%
Yao et al. [155]	–	81.50%
Barnachon et al. [10]	**56.82%**	**92.56%**

TABLE 9.1 – Comparaisons des méthodes de reconnaissance d'activité sur le jeu de données TUM, proposé par Tenorth et al. [126]. Nous avons construit 3 classifieurs afin de tester l'apprentissage simple hypothèse. Pour montrer la validité des approches, nous avons reporter leurs précisions avec le *training set* des multiples hypothèses.

Jeux de données	Précision	
	Notre	Meilleure
HDM (*one-vs-one*)	**67.89%**	–
HDM (multi-hypothèses)	**96.67%**	80% Müller et al. [87]
CMU (*one-vs-one*)	**86.63%**	–
CMU (multi-hypothèses)	**90.92%**	75% Müller et al. [87]
TUM (*one-vs-one*)	**56.82%**	51.41% (kNN)
TUM (multi-hypothèses)	**92.56%**	81.50% Yao et al. [155]

TABLE 9.2 – Comparaisons des reconnaissances sur les jeux de données HDM, CMU et TUM, en simple ou multiples hypothèses.

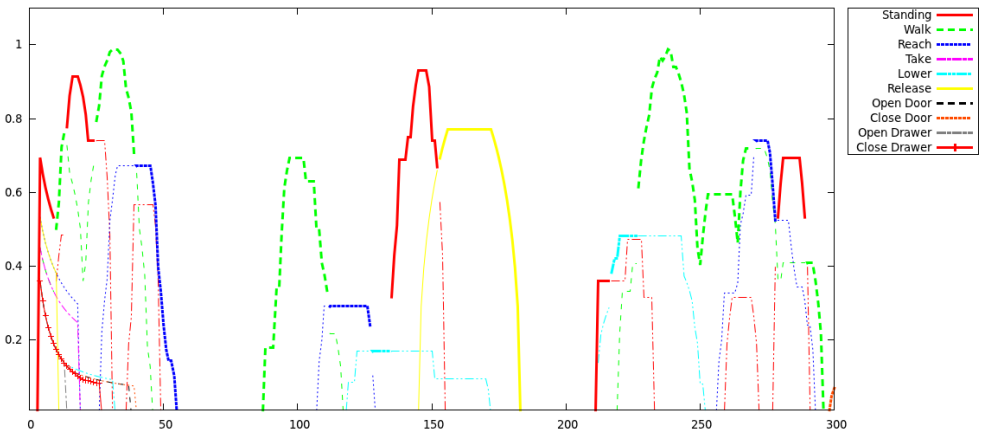


FIGURE 9.3 – Reconnaissance de l'activité « mettre la table » du jeu de données TUM, activité 0-2, apprise sur l'activité 0-12.

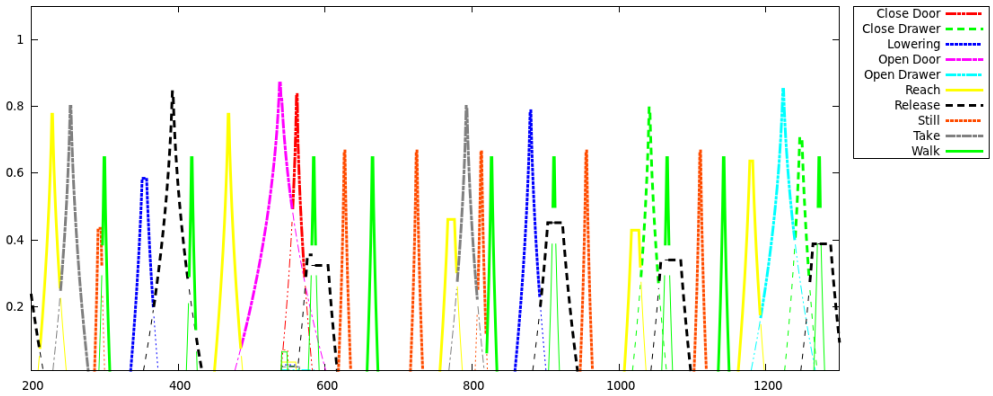


FIGURE 9.4 – Reconnaissance de l'activité « mettre la table » du jeu de données TUM, activité 0-8, apprise en multiples hypothèses.

évaluation de la proximité entre cette action et les actions de la base, temporellement restreintes à la longueur courante de l'action en cours de reconnaissance.

La figure 9.5 montre l'évolution de la précision concernant l'ensemble des jeux de données utilisés. Les deux hypothèses d'apprentissage ont été testées, apprentissage unique et apprentissage « complet ». Les performances sont différentes en fonction des jeux de données, néanmoins, tous montrent une bonne reconnaissance dès lors que l'on considère 50% de l'action. La faible évolution du score de reconnaissance pour le jeu de données HDM s'explique par la présence de cycles au sein de chacune des actions. En effet, la marche par exemple est une succession, plus ou moins régulière, de pas. Or, dans une représentation par histogramme normalisé, les cycles n'apportent pas d'information supplémentaire. Dès lors, les histogrammes de 2 pas ou de $2 \times n$ pas sont semblables, et par conséquence l'action est plus aisément reconnue. *A contrario*, les actions du jeu de données CMU sont moins répétitives. Ceci se traduit par une forte augmentation du score de reconnaissance au cours du temps. En effet, plus on apporte d'information, plus son pouvoir discriminant augmente. Ceci reste vrai, que l'on utilise un apprentissage unique ou l'ensemble du jeu d'apprentissage pour faire la reconnaissance (voir les courbes bleue et magenta sur la figure 9.5). Un autre enseignement intéressant à tirer de la figure 9.5 est la différence importante entre l'apprentissage unique et l'apprentissage multiple, pour la reconnaissance d'activités. On peut remarquer qu'il y a une différence importante entre la reconnaissance par apprentissage unique et l'apprentissage complet, pour les jeux HDM et TUM. Ceux-ci sont représentatifs d'une grande intra-variabilité. En effet, les gestes sont courts, potentiellement très différent les uns des autres, et assez peu reproductibles. L'apprentissage doit pouvoir prendre en considération cette variabilité interne. Or, en utilisant un unique *examplar* par activité, le résultat est nécessairement en dessous des attentes. Néanmoins, on peut

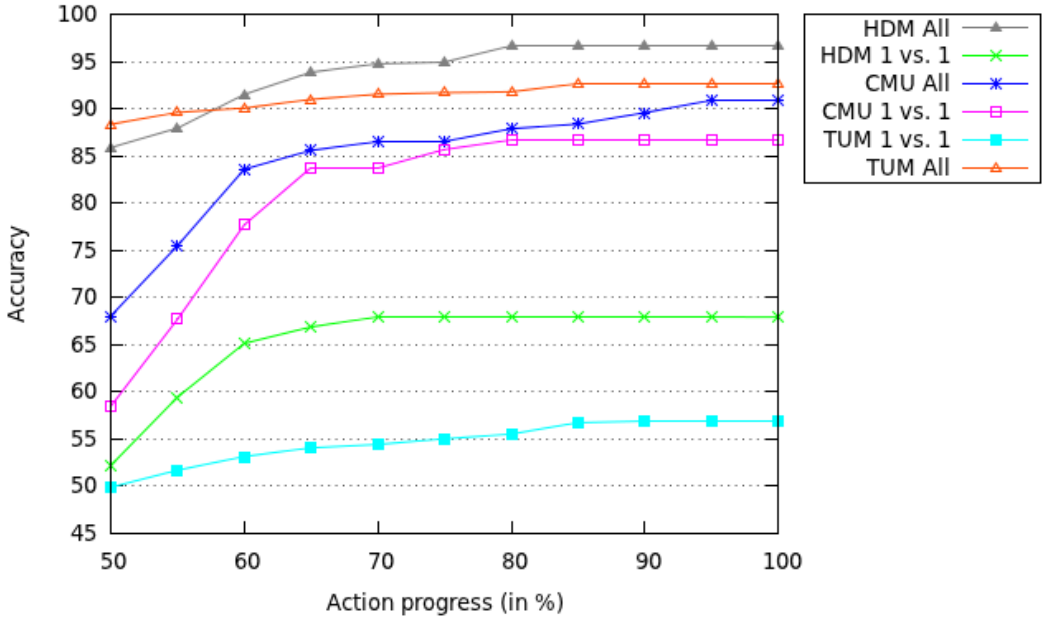


FIGURE 9.5 – Reconnaissance en ligne pour les jeux de données. Score présenté entre 50% et 100%, par pas de 5%.

noter que lorsqu'on s'intéresse à la reconnaissance d'actions, comme dans le jeu CMU, la différence entre apprentissages unique et multiple est plus faible. En effet, les actions possèdent moins de variations internes. Il est dès lors plus évident de les discriminer, même si le modèle initial ne connait pas toutes les subtilités possibles.

9.4 Étude des paramètres

Notre méthode est dépendante de deux paramètres importants : la distance inter-poses, ε, définie dans l'équation 7.1, et le nombre maximum d'hypothèses représentant une action, défini comme le paramètre k de l'algorithme 8.1.

L'ensemble de notre méthode repose sur la construction d'un histogramme de poses. Pour cela, il faut définir un critère de vraisemblance entre poses, afin de pouvoir compter les occurrences de poses « proches ». Pour ce faire, nous avons introduit la notion d'équivalence entre poses par le biais de la définition 6. Ainsi, pour une valeur donnée de ε, nous pouvons regrouper les poses identiques dans une même classe d'histogramme. Or, il est évident que ce paramètre influence directement, à la fois la précision de la reconnaissance, mais aussi, et surtout, le temps de calcul nécessaire pour reconnaitre une action. Le paramètre ε influence

directement le nombre de représentants. De celui-ci dépend directement le temps de calcul car plus l'espace des solutions est important, plus l'exploration de cet espace est coûteuse en temps de calcul. Le temps de calcul et le nombre de représentants pour une action sont exprimés en fonction de la distance inter-poses dans la figure 9.6. Cette figure représente la séquence 0-2 du jeu de données TUM, d'une durée de 39 secondes. La mise en correspondance du nombre de représentants et du temps de calcul est clairement établie. La longueur de l'action, en secondes, est exprimée par la ligne horizontale noire. Lors de l'utilisation d'une distance entre action de 0 ou 1mm l'exploration des solutions n'est pas temps réel. Il est toutefois à noter que ces paramètres ne sont pas réalistes dans la plupart des applications, car une distance entre poses de maximum 1mm est en dessous de la précision générale des systèmes d'acquisition de mouvements, de l'ordre de quelques centimètres pour les systèmes sans marqueur comme Tenorth et al. [126]. Néanmoins, à partir de 2mm de distances inter-poses, le système que nous proposons est temps réel. Notons que le nombre de représentants est trop important pour être efficace, supérieur à 14000. Il s'agit d'une sorte de « *overfitting* ». Voir Bishop [21] pour les conséquences dans les systèmes d'apprentissage statistiques. Le seuil de 20cm, soit 5799 représentants a été choisi de manière empirique. Il satisfait l'ensemble des contraintes précédentes, à savoir, dans la limites de précision des systèmes de capture de mouvements, mais sans introduire trop de représentants. Il est bon de noter qu'un nombre trop important de représentant nuit à la fiabilité de la méthode. En effet, les actions sont trop dissemblables les unes des autres, et les scores de reconnaissances décroissent. Le phénomène inverse se produit en cas de faible nombre de représentants. Dans ce cas, les actions sont trop semblables les unes aux autres, et la précision de reconnaissance s'effondre également. Avec $\varepsilon = 20$, nous avons obtenu, pour une action constituée de 957 poses, soit une durée totale de 39 secondes, un temps de reconnaissance de $12,0756$ secondes. La reconnaissance prend moins de $\frac{1}{3}$ du temps de l'action, ce qui, comme elle se déroule sur l'action en cours, durant son exécution, permet de satisfaire aisément le temps réel.

Le second paramètre d'importance dans notre système est le nombre d'hypothèses retenues pour représenter la variation intra-classe. Dans ce contexte, nous avons étudié le nombre d'hypothèse retenu, *i.e.* la valeur de k dans l'algorithme *KMédoïds* 8.1, en fonction du coût maximum autorisé au sein d'un même médoid. Les résultats présentés dans la figure 9.7 montrent l'évolution du coût — définit par l'équation 8.24 — et du nombre d'hypothèses retenues. Nous avons utilisé le cas emblématique de l'action « Walk », du jeu de données TUM, car il s'agit d'une action courte et possédant une forte variabilité interne dans ce contexte. Nous pouvons remarquer que lorsque le coût maximal autorisé est inférieur à 0.5 — dans le sens de la mesure de l'équation 8.10 — nous avons un nombre d'hypothèses quasi

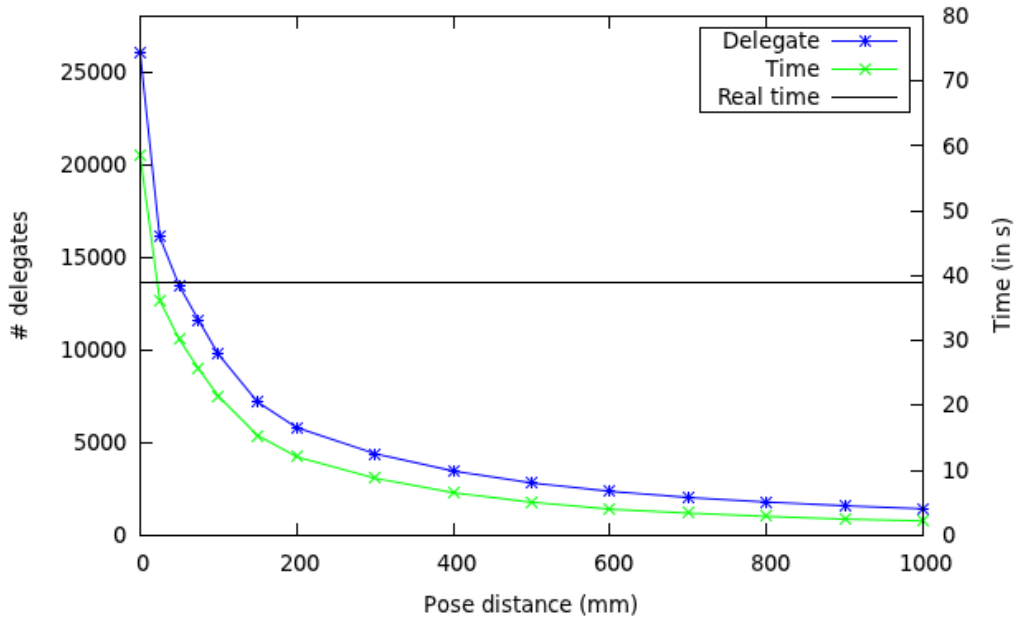


FIGURE 9.6 – Effet du paramètre ε sur le nombre de poses et sur le temps de reconnaissance.

constants. Au-delà de 0.5, le nombre d'hypothèses (de médoids), décroit rapidement. Dans ces conditions, nous pouvons ajuster la valeur de k afin de garantir une certaine « homogénéité » des hypothèses qui représente l'action. Tout en tenant compte du fait de la granularité souhaitée pour chacune des actions, *i.e.* si les actions sont très versatile ou pas.

9.5 Conclusion

Nous venons de présenter notre reconnaissance d'actions par histogrammes intégraux. Notre approche est incrémentale, ce qui est indispensable pour fournir une reconnaissance de l'action au cours de son déroulement. Nous avons appliqué le principe de la programmation dynamique pour augmenter la robustesse de la méthode. Notre méthode repose sur l'utilisation de concepts compréhensibles (histogrammes) et de métriques connue (Bhattacharyya).

De plus, notre solution est capable de travailler avec des séquences de capture de mouvements, sans nécessité de pré-segmentation. L'apprentissage utilisé est semblable à la reconnaissance et très proche du temps réel. Il n'est ni complexe ni coûteux en temps de calcul, ce qui lui confère un avantage indéniable face aux méthodes de *machine learning*, comme les SVM, k-plus-proches-voisins (kNN), *etc.*

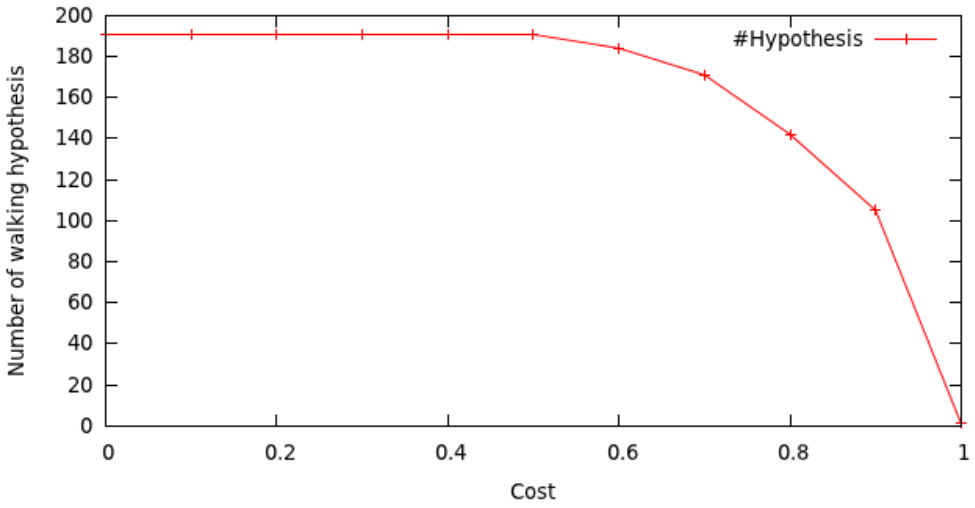


FIGURE 9.7 – Évolution du nombre d'hypothèses en fonction de la distance interposes, action « walk » du jeu de données TUM.

Notre modèle de représentation des actions, basé sur des histogrammes, a permis de transformer un problème complexe en une solution d'ingénierie plus simple, reposant sur l'évaluation de métriques connues. Nous avons aussi étendu le concept des histogrammes à des cas plus complexes, non exclusivement 2D, et sans ordre total. En effet, les poses ne sont pas arrangées dans l'histogramme par proximité. Ainsi, une méthode traitant des histogrammes se doit de prendre en compte cet aspect. L'une des solutions classiques est l'utilisation d'une métrique qui tient compte des distances entre modes dans un histogramme, comme la *Earth Mover's Distance* par exemple. Nous avons pu dépasser ce problème — en utilisant le paramètre ε — qui s'affranchit de l'ordre total en proposant un regroupement afin de modéliser les proximités entre poses sans avoir besoin de les classer. En effet, les poses à une distances supérieure à ε les unes des autres sont suffisamment différentes pour que l'on n'ait pas besoin d'un ordre.

D'autre part, notre solution repose sur deux paramètres simples et facile d'accès permettant de l'adapter aux différents contextes envisageable lors de la reconnaissance d'actions. De par l'étude qui a été mené sur eux, il est aisé de voir leur influence sur les résultats, et donc de les adapter en fonction des besoins : en précision, en vitesse ou en espace mémoire si d'aventure ceux-ci devenait critique.

Les résultats de ses travaux ont été publiés dans Barnachon et al. [10, 11, 12].

Quatrième partie

Bilan

Chapitre 10

Conclusion & Perspectives

10.1 Conclusion

Dans ce travail, nous nous sommes intéressés à la reconnaissance d'actions, à partir de données de capture de mouvements, Pour traiter cette question, nous avons proposé deux approches qui s'appuient sur l'étude des mouvements des articulations, en suivant des visions complémentaires pour répondre à des contextes d'applications différents.

La première approche se base sur les trajectoires des articulations, il est possible de sélectionner certaines articulations ou de travailler en présence d'une capture partielle. Elle résiste à des occultations qui peuvent intervenir au cours du temps. La réalisation de l'apprentissage par un automate permet de s'affranchir de la multiplicité des représentants d'une part. D'autre part, cette structuration de l'apprentissage accélère le traitement de la reconnaissance des actions. La modélisation des trajectoires par segments assure une continuité temporelle des actions. Pour le moment, même si l'automate permet de rassembler les articulations en éléments atomiques du mouvement, l'analyse de base est faite par articulations, et est donc sensible aux conditions d'acquisition des articulations.

La deuxième méthode s'appuie sur la décomposition d'une action en une succession de poses. Pour cela, nous nous basons sur un calcul d'histogramme afin d'obtenir une représentation statistique de l'action. L'histogramme donne une idée de « l'allure générale » de l'action et permet d'être moins sensible aux variations locales qui peuvent survenir au moment de l'acquisition des poses. Des représentants de poses sont extraits, en se servant de la distance de Haussdorf. Sur la base de ses représentants, un histogramme par action est établi. Une comparaison di-

recte des histogrammes est réalisée par un alignement de séquences en utilisant la programmation dynamique. La distance de Bhattacharyya nous permet de faire une comparaison efficace entre les histogrammes. L'adaptation de ces deux techniques nous a permis d'avoir une approche fiable et rapide pour la reconnaissance. Si cette méthode nous offre une bonne stabilité, nous avons constaté une sensibilité aux occultations qui peuvent survenir au cours du temps.

10.2 Perspectives

Pour valider nos méthodes, dans toutes nos expérimentations, nous nous sommes servis de squelettes 3D, aussi bien pour les trajectoires que pour les poses. Dans leurs conceptions, nos méthodes sont suffisamment génériques pour être adaptées à tout autre type de données qui caractérisent une action. En particulier une séquence $2D$, qu'elle soit sous la forme de squelette ou de silhouette. Il serait intéressant d'étudier la stabilité des trajectoires d'articulations ou la notion de représentants de poses, au cours de l'action, sachant que les données seront dépendantes du point de vue et qu'il faudra gérer les cas d'auto-occultations, inhérents aux données. Une telle étude permettra de disposer d'une méthode unifiée pour la reconnaissance d'actions, aussi bien en 3D qu'en 2D.

L'approche par trajectoires souffre d'une dépendance à la régression linéaire des trajectoires, pas toujours représentative du mouvement réalisé. Si nous avons opté pour une modélisation linéaire pour des raisons de temps de calculs, l'une des perspectives immédiate serait de passer à des polynômes de plus haut degré. Une telle modélisation permettrait d'avoir une représentation plus fidèle des trajectoires, et respecterait plus la souplesse du mouvement dans les actions. Nous serons amenés à revoir les critères de décomposition de la trajectoire et repréciser la notion d'action atomique. L'une des solutions possibles serait de s'appuyer sur les propriétés de la courbe : extrema, points d'inflexion, singularité, *etc.* Par ailleurs, étant toujours contraint par le temps réel, le choix des bases de polynômes d'approximation (bases de splines, Tchebychev, *etc.*) sera très important pour conserver la construction incrémentale des trajectoires au cours du temps.

Actuellement, les transitions dans l'automate sont prises suivant un critère géométrique. Un échec dans une transition peut avoir beaucoup d'influence sur la reconnaissance d'une action. Pour relaxer cette rigidité, il serait intéressant d'explorer les possibilités que peut offrir un automate stochastique. Cet apport permettrait de rapprocher, voire de transformer, notre automate en un automate au sens de Markov. Bien sûr, nous sommes conscients que pour pouvoir réaliser l'apprentissage, un minimum de données sera requis dans ce cas. L'apprentissage devrait être modifié, pour utiliser soit plus d'*examplars*, soit pour travailler directement

avec l'ensemble des données d'apprentissage nécessaires.

Dans l'état actuel, le système de reconnaissance d'actions par histogrammes, ne prend pas en compte les actions symétriques, alors qu'elles peuvent être les mêmes. Ainsi, l'action « ouvrir un placard sur la gauche », conduit à la création d'un histogramme incompatible avec un histogramme d'une action « ouvrir un placard sur la droite ». L'une des solutions envisageables est de créer des représentants invariants par symétrie.

Les résultats de l'approche par histogrammes sont dépendants du choix des représentants. Nous avons pris le parti d'extraire les représentants sans contrainte sur leur nombre. Cette méthode permet d'ajouter des représentants au cours du temps. Cependant, le choix d'un représentant ne peut pas être remis en cause, ce qui peut aboutir à un partitionnement non optimal de l'espace des poses, ou encore à créer des classes dégénérées, avec peu d'éléments. Une de nos perspectives serait de réévaluer dynamiquement les représentants de classes générés pour optimiser au cours du temps le positionnement de l'ensemble des poses suivant des critères de fusion et division de classes.

Actuellement, la comparaison des histogrammes lors de la reconnaissance est réalisée de façon exhaustive : l'ensemble des histogrammes appris est comparé à chaque reconnaissance. Une structuration des histogrammes sous forme hiérarchique qui permettrait de limiter le nombre de comparaisons en suivant une branche pour la reconnaissance. Pour cela, nous pourrions nous inspirer des avantages des méthodes d'apprentissage automatique (*Machine Learning*), comme les *Random Tree* ou *Random Ferns*. En plus du gain de vitesse de reconnaissance attendu grâce à leurs propriétés de construction incrémentales, ces structures pourraient être utilisées lors de notre apprentissage.

Les deux méthodes présentées dans cette thèse donnent de bons résultats, mais travaillent de façon indépendante. Actuellement, la première permet de niveler les détails perturbateurs pour la reconnaissance, alors que la méthode sur les trajectoires donne des informations précieuses sur la vitesse dans laquelle elle est faite. L'une des suites naturelles serait d'avoir un processus qui prendrait en compte le meilleur des deux.

Bibliographie

[1] Aggarwal, J. et M. Ryoo. 2011, Human activity analysis, *ACM Computing Survey*, vol. 43, p. 1–43. 7, 8

[2] Ahmad, M. et S.-W. Lee. 2010, Variable silhouette energy image representations for recognizing human actions, *Image and Vision Computing*, vol. 28, n° 5, p. 814 – 824. 18, 36, 38

[3] Ali, S., A. Basharat et M. Shah. 2007, Chaotic invariants for human action recognition, dans *Internationale Conference on Computer Vision*, p. 1–8. 22, 39

[4] Allard, J., J.-S. Franco, C. Ménier, E. Boyer et B. Raffin. 2006, The GrImage Platform : A Mixed Reality Environment for Interactions, dans *4th International Conference on Computer Vision Systems, ICVS'06, January, 2006*, p. 46–46. 28

[5] Andriluka, M., S. Roth et B. Schiele. 2009, Pictorial structures revisited : People detection and articulated pose estimation, dans *Conference on Computer Vision and Pattern Recognition*, p. 1014–1021. 22

[6] Argyros, A. et M. Lourakis. 2006, Vision-based interpretation of hand gestures for remote control of a computer mouse, dans *Computer Vision in Human-Computer Interaction*, p. 40–51. 28, 29, 39

[7] Baak, A., B. Rosenhahn, M. Müller et H.-P. Seidel. 2009, Stabilizing motion tracking using retrieved motion priors, dans *Internationale Conference on Computer Vision*, p. 1428–1435. 25, 39

[8] Bandouch, J. et M. Beetz. 2009, Tracking humans interacting with the environment using efficient hierarchical sampling and layered observation models, dans *IEEE International Workshop on Human-Computer Interaction*, p. 2040–2047. 138, 147

[9] Barbič, J., A. Safonova, J.-Y. Pan, C. Faloutsos, J. K. Hodgins et N. S. Pollard. 2004, Segmenting motion capture data into distinct behaviors, dans *Graphics Interface*, p. 185–194. 27, 39

[10] Barnachon, M., S. Bouakaz, B. Boufama et E. Guillou. 2012, Human actions recognition from streamed motion capture, dans *International Conference on Pattern Recognition*, p. 3807–3810. x, xii, 32, 108, 110, 115

[11] Barnachon, M., S. Bouakaz, B. Boufama et E. Guillou. 2012, Reconnaissance d'Action à Partir de Capture de Mouvements, dans *CORESA*. x, xii, 115

[12] Barnachon, M., S. Bouakaz, B. Boufama et E. Guillou. 2013, Ongoing human action recognition with motion capture, *Pattern Recognition*. x, xii, 115

[13] Barnachon, M., S. Bouakaz, B. Boufama et E. Guillou. 2013, A real-time system for motion retrieval and interpretation, *Pattern Recognition Letters*. x, xi, 84

[14] Barnachon, M., S. Bouakaz, E. Guillou et B. Boufama. 2012, Interprétation de Mouvements Temps Réel, dans *RFIA*. x, xi, 84

[15] Barnachon, M., M. Ceccaroli, A. Cordier, E. Guillou et M. Lefevre. 2011, Intelligent Interactions Based on Motion, dans *Workshop CBR and Games, ICCBR 2011*. 82

[16] Batra, D., T. Chen et R. Sukthankar. 2008, Space-time shapelets for action recognition, dans *Workshop on Motion and video Computing*, p. 1–6. 35

[17] Baum, L. E., T. Petrie, G. Soules et N. Weiss. 1970, A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains, *The Annals of Mathematical Statistics*, vol. 41, n° 1, p. 164–171. 36

[18] Beaudoin, P., S. Coros, M. van de Panne et P. Poulin. 2008, Motion-motif graphs, dans *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, p. 117–126. 28, 39, 67

[19] Bellman, R. 1957, *Dynamic Programming*, Princeton University. 98

[20] Bhattacharyya, A. 1943, On a measure of divergence between two statistical populations defined by their probability distributions, *Bulletin of the Calcutta Mathematical Society*, p. 99–109. 96, 97, 114

[21] Bishop, C. M. 2006, *Pattern Recognition and Machine Learning*, Springer-Verlag New York, Inc., ISBN 0387310738. 108, 113, 152

[22] Blackburn, J. et E. Ribeiro. 2007, Human motion recognition using isomap and dynamic time warping, dans *Proceedings of the 2nd conference on Human motion : understanding, modeling, capture and animation*, p. 285–298. 31

[23] Blank, M., L. Gorelick, E. Shechtman, M. Irani et R. Basri. 2005, Actions as space-time shapes, dans *Internationale Conference on Computer Vision*, p. 1395–1402. 8, 11, 12, 13, 26, 35, 38

[24] Bobick, A. F. et J. W. Davis. 2001, The recognition of human movement using temporal templates, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, n° 3, p. 257–267. 17, 35, 38

[25] Caillette, F., A. Galata et T. Howard. 2008, Real-time 3-d human body tracking using learnt models of behaviour, *Computer Vision and Image Understanding*, vol. 109, n° 2, p. 112–125. 37

[26] Cao, L., Z. Liu et T. S. Huang. 2009, Cross-dataset action detection, dans *Conference on Computer Vision and Pattern Recognition*, p. 1998–2005. 8

[27] Chakraborty, B., O. Rudovic et J. Gonzalez. 2008, View-invariant human-body detection with extension to human action recognition using component-wise hmm of body parts, dans *International Conference on Automatic Face Gesture Recognition*, p. 1–6. 37

[28] Cheng, H., Z. Liu, Y. Zhao et G. Ye. 2011, Real world activity summary for senior home monitoring, dans *International Conference Multimedia and Expo*, p. 1–4. 9, 11, 12

[29] Chin, T.-J., L. Wang, K. Schindler et D. Suter. 2007, Extrapolating learned manifolds for human activity recognition, dans *Internationla Conference on Image Processing*, vol. 1, p. 381–384. 31

[30] Chomat, O. et J. L. Crowley. 1999, Probabilistic recognition of activity using local appearance, dans *Conference on Computer Vision and Pattern Recognition*, p. 104–109. 12, 38, 90

[31] Csurka, G., C. R. Dance, L. Fan, J. Willamowski et C. Bray. 2004, Visual categorization with bags of keypoints, dans *In Workshop on Statistical Learning in Computer Vision, ECCV*, p. 1–22. 15, 31, 90

[32] Cuntoor, N., B. Yegnanarayana et R. Chellappa. 2008, Activity modeling using event probability sequences, *IEEE Transactions on Image Processing*, vol. 17, n° 4, p. 594–607. 24, 39

[33] Devijver, P. A. et J. Kittler. 1982, *Pattern Recognition : a statistical approach*, Prentice Hall, 440 p.. 77

[34] Dollar, P., V. Rabaud, G. Cottrell et S. Belongie. 2005, Behavior recognition via sparse spatio-temporal features, dans *Proceedings of the 14th International Conference on Computer Communications and Networks*, p. 65–72. 14, 15, 31, 38

[35] Efros, A. A., A. C. Berg, G. Mori et J. Malik. 2003, Recognizing action at a distance, dans *Internationale Conference on Computer Vision*, p. 726–733. 13, 38

[36] Elgammal, A., V. Shet, Y. Yacoob et L. S. Davis. 2003, Learning dynamics for exemplar-based gesture recognition, dans *Conference on Computer Vision and Pattern Recognition*, p. 571–578. 17, 18, 38

[37] Elliott, R. J., L. Aggoun et J. B. Moore. 1995, *Hidden Markov models : estimation and control*, Springer-Verlag. 153

[38] Fathi, A. et G. Mori. 2008, Action recognition by learning mid-level motion features, dans *Conference on Computer Vision and Pattern Recognition*, p. 1–8. 8

[39] Feng, X. et P. Perona. 2002, Human action recognition by sequence of movelet codewords, dans *3D Data Processing Visualization and Transmission, First International Symposium on*, p. 717–721. 36

[40] Ferrari, V., M. Marin-Jiménez et A. Zisserman. 2009, Pose search : retrieving people using their pose, dans *Conference on Computer Vision and Pattern Recognition*, p. 1–8. 14

[41] Frey, B. J. et N. Jojic. 2000, Learning graphical models of images, videos and their spatial transformations, dans *Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence*, p. 184–191. 17, 18, 68

[42] Fujiyoshi, H. et A. J. Lipton. 1998, Real-time human motion analysis by image skeletonization, p. 15–21. 21, 23, 39

[43] Gavrila, D. M. et L. S. Davis. 1995, Towards 3-d model-based tracking and recognition of human movement : a multi-view approach, dans *International Workshop on Automatic Face and Gesture-Recognition*, p. 272–277. 21, 39

[44] Gilbert, A., J. Illingworth et R. Bowden. 2011, Action recognition using mined hierarchical compound features, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, n° 5, p. 883–897. 8

[45] Gong, W., A. D. Bagdanov, F. Roca et J. Gonzàlez. 2010, Automatic key pose selection for 3d human action recognition, dans *Articulated Motion and Deformable Objects*, p. 290–299. 32

[46] Guerra-Filho, G. et Y. Aloimonos. 2010, The syntax of human actions and interactions, *Journal of Neurolinguistics*, p. 500–514. 26, 27

[47] Han, L., X. Wu, W. Liang, G. Hou et Y. Jia. 2010, Discriminative human action recognition in the learned hierarchical manifold space, *Image and Vision Computing*, p. 836–849. 25, 39, 143

[48] Harris, C. et M. Stephens. 1988, A Combined Corner and Edge Detection, dans *Proceedings of The Fourth Alvey Vision Conference*, p. 147–151. 14

[49] Huang, K. S. et M. M. Trivedi. 2005, 3d shape context based gesture analysis integrated with tracking using omni video array, dans *Conference on Computer Vision and Pattern Recognition*, p. 80–87. 19, 38, 90

[50] İkizler, N. et D. A. Forsyth. 2008, Searching for complex human activities with no visual examples, *International Journal of Computer Vision*, vol. 80, n° 3, p. 337–357. 36

[51] Itakura, F. 1990, Readings in speech recognition, chap. Minimum prediction residual principle applied to speech recognition, Morgan Kaufmann Publishers Inc., p. 154–158. 99

[52] Ivanov, Y. A. et A. F. Bobick. 2000, Recognition of visual activities and interactions by stochastic parsing, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, p. 852–872. 28, 29, 39

[53] Jhuang, H., T. Serre, L. Wolf et T. Poggio. 2007, A biologically inspired system for action recognition, dans *Internationale Conference on Computer Vision*, p. 1–8. 13, 33, 38

[54] Johansson, G. 1973, Visual perception of biological motion and a model for its analysis, *Perception & Psychophysics*, vol. 14, p. 201–211. 21, 44, 51

[55] Kadir, T. et M. Brady. 2001, Saliency, scale and image description, *International Journal of Computer Vision*, vol. 45, p. 83–105. 15

[56] Kaufman, L. et P. Rousseeuw. 1987, *Clustering by Means of Medoids*, Reports of the Faculty of Mathematics and Informatics. Delft University of Technology. 103

[57] Ke, Y., R. Sukthankar et M. Hebert. 2010, Volumetric features for video event detection, *International Journal of Computer Vision*, vol. 88, p. 339–362. 20, 38

[58] Klaeser, A., M. Marszalek et C. Schmid. 2008, A spatio-temporal descriptor based on 3d-gradients, dans *British Machine Vision Conference*, p. 995–1004. 14, 38

[59] Kong, Y., X. Zhang, W. Hu et Y. Jia. 2011, Adaptive learning codebook for action recognition, *Pattern Recognition Letters*, vol. 32, n° 8, p. 1178–1186. 32

[60] Kovar, L., M. Gleicher et F. Pighin. 2002, Motion graphs, *ACM Transaction on Graphics*, vol. 21, n° 3, p. 473–482. 67

[61] Kovashka, A. et K. Grauman. 2010, Learning a hierarchy of discriminative space-time neighborhood features for human action recognition, dans *Conference on Computer Vision and Pattern Recognition*, p. 2046–2053. 31

[62] Krausz, B. et C. Bauckhage. 2010, Action recognition in videos using nonnegative tensor factorization, dans *International Conference on Pattern Recognition*, p. 1763–1766. 83, 108, 110

[63] Kuehne, H., H. Jhuang, E. Garrote, T. Poggio et T. Serre. 2011, Hmdb : A large video database for human motion recognition, dans *Internationale Conference on Computer Vision*, p. 2556–2563. 9, 11, 12

[64] Lafferty, J. D., A. McCallum et F. C. N. Pereira. 2001, Conditional random fields : Probabilistic models for segmenting and labeling sequence data, dans *International Conference on Machine Learning*, p. 282–289. 37

[65] Laptev, I., B. Caputo, C. Schüldt et T. Lindeberg. 2007, Local velocity-adapted motion events for spatio-temporal recognition, *Computer Vision and Image Understanding*, vol. 108, n° 3, p. 207–229. 33

[66] Laptev, I. et T. Lindeberg. 2003, Space-time interest points, dans *Internationale Conference on Computer Vision*, p. 432–439. 14, 38

[67] Laptev, I., M. Marszałek, C. Schmid et B. Rozenfeld. 2008, Learning realistic human actions from movies, dans *Conference on Computer Vision and Pattern Recognition*, p. 1–8. 9, 11, 12, 15, 31, 38

[68] Leibe, B., A. Leonardis et B. Schiele. 2004, Combined object categorization and segmentation with an implicit shape model, dans *In ECCV workshop on statistical learning in computer vision*, p. 17–32. 65

[69] Li, S., J. Lv, Y. Xu et Y. Jia. 2007, Eyescreen : A gesture interface for manipulating on-screen objects, dans *International Conference on Human-Computer Interaction : Intelligent Multimodal Interaction Environments*, p. 710–717. 28, 29, 39

[70] Li, X. et K. Fukui. 2008, View invariant human action recognition based on factorization and hmms, *Transactions Information and Systems*, vol. E91-D, n° 7, p. 1848–1854. 25, 39

[71] Lin, Z., Z. Jiang et L. S. Davis. 2009, Recognizing Actions by Shape-Motion Prototype Trees, dans *Internationale Conference on Computer Vision*. 35

[72] Ling, H. et K. Okada. 2007, An efficient earth mover's distance algorithm for robust histogram comparison, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, p. 840–853. 97

[73] Liu, J., J. Luo et M. Shah. 2009, Recognizing realistic actions from videos "in the wild", dans *Conference on Computer Vision and Pattern Recognition*, p. 1996–2003. 9, 11, 12, 31

[74] Liu, J. et M. Shah. 2008, Learning human actions via information maximization, dans *Conference on Computer Vision and Pattern Recognition*, p. 1–8. 15, 38

[75] Lowe, D. G. 1999, Object recognition from local scale-invariant features, dans *Internationale Conference on Computer Vision*, p. 1150–1157. 14, 15

[76] Lv, F. et R. Nevatia. 2006, Recognition and segmentation of 3-d human action using hmm and multi-class adaboost, dans *European Conference on Computer Vision*, p. 359–372. 37

[77] Lv, F. et R. Nevatia. 2007, Single view human action recognition using key pose matching and viterbi path searching., dans *Conference on Computer Vision and Pattern Recognition*, p. 1–8. 24, 32, 36, 39, 90

[78] Marey, E.-J. 1887, Le mécanisme du vol des oiseaux éclairé par la chronophotographie, *La nature : revue des sciences et de leurs applications aux arts et à l'industrie*, p. 8–14. 43

[79] Marszałek, M., I. Laptev et C. Schmid. 2009, Actions in context, dans *Conference on Computer Vision and Pattern Recognition*, p. 2929–2936. 9, 11, 12

[80] Masoud, O. et N. Papanikolopoulos. A method for human action recognition, *Image and Vision Computing*, vol. 21, n° 8, p. 729–743. 31

[81] McCann, J., N. S. Pollard et S. Srinivasa. 2006, Physics-based motion retiming, dans *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, p. 205–214. 71, 77, 78, 79, 80

[82] Mendoza, M. et N. Pérez de la Blanca. 2008, Applying space state models in human action recognition : A comparative study, dans *Articulated Motion and Deformable Objects*, p. 53–62. 37

[83] Michoud, B., E. Guillou, H. B. Pulido et S. Bouakaz. 2007, Real-Time Marker-free Motion Capture from multiple cameras, dans *Internationale Conference on Computer Vision*, p. 1–7. 46

[84] MoCap CMU. 2003, 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/. 46, 75, 76, 105, 138, 143, 144, 145

[85] Moeslund, T. B., A. Hilton, V. Krüger et L. Sigal, éd.. 2011, *Visual Analysis of Humans*, Springer. 10

[86] Müller, M. 2007, *Information Retrieval for Music and Motion*, Springer-Verlag New York, Inc., Secaucus, NJ, USA, ISBN 3540740473. 99

[87] Müller, M., A. Baak et H.-P. Seidel. 2009, Efficient and robust annotation of motion capture data, dans *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, p. 17–26. 28, 39, 83, 110, 143

[88] Müller, M., T. Röder, M. Clausen, B.Eberhardt, B. Krüger et A. Weber. 2007, Documentation mocap database hdm05, cahier de recherche, Universität Bonn. 105, 106, 108, 146

[89] Muybridge, E. 1887, *Animal Locomotion*. 43

[90] Natarajan, P. et R. Nevatia. 2008, Online, real-time tracking and recognition of human actions, dans *Workshop on Motion and video Computing*, p. 1–8. 37

[91] Niebles, J. C., C.-w. Chen et L. Fei-fei. 2010, Modeling temporal structure of decomposable motion segments for activity classification, dans *European Conference on Computer Vision*, p. 392–405. 15, 38

[92] Niebles, J. C., H. Wang et L. Fei-fei. 2006, Unsupervised learning of human action categories using spatial-temporal words, dans *British Machine Vision Conference*, p. 299–318. 15, 31, 38

[93] Ogale, A. S., A. Karapurkar et Y. Aloimonos. 2007, View-invariant modeling and recognition of human actions using grammars, dans *Proceedings of the International Conference on Dynamical Vision*, p. 115–126. 37

[94] Oikonomopoulos, A., I. Patras et M. Pantic. 2005, Spatiotemporal salient points for visual recognition of human actions, *IEEE Transactions on Systems, Man and Cybernetics - Part B*, p. 710–719. 15, 38

[95] Okada, R. et B. Stenger. 2008, A single camera motion capture system for human-computer interaction, *Transactions Information and Systems*, p. 1855–1862. 28, 30, 39, 83

[96] Parameswaran, V. et R. Chellappa. 2003, View invariants for human action recognition, dans *Conference on Computer Vision and Pattern Recognition*, p. 613–619. 22, 39, 143

[97] Patron-perez, A., M. Marszalek, A. Zisserman et I. Reid. 2010, High five : Recognising human interactions in tv shows, dans *British Machine Vision Conference*, p. 1–11. 9, 11, 12, 14, 38, 90

[98] Pearson, K. 1900, On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling, *Philosophical Magazine*. 96, 97

[99] Pearson, K. 1901, On lines and planes of closest fit to systems of points in space, *Philosophical Magazine*, p. 559–572. 92

[100] Pele, O. et M. Werman. 2009, Fast and robust earth mover's distances, dans *Internationale Conference on Computer Vision*, p. 460–467. 97

[101] Peursum, P., S. Venkatesh et G. West. 2007, Tracking-as-recognition for articulated full-body human motion analysis, dans *Conference on Computer Vision and Pattern Recognition*, p. 1–8. 37

[102] Picard, F. 2011, *Contextualisation & Capture de Gestuelles Utilisateur Contributions à l'Adaptativité des Applications Interactives Scénarisées*, thèse de doctorat, Université de La Rochelle. 47, 49

[103] Poppe, R. 2010, A survey on vision-based human action recognition, *Image and Vision Computing*, vol. 28, n° 6, p. 976–990. 7, 30, 37

[104] Ramanan, D. et D. A. Forsyth. 2003, Automatic annotation of everyday movements, dans *Nural Information Processing Systems*, p. 1547–1554. 22, 39

[105] Rao, C. et M. Shah. 2001, View-invariance in action recognition, dans *Conference on Computer Vision and Pattern Recognition*, p. 316–322. 24, 39

[106] Raptis, M., D. Kirovski et H. Hoppe. 2011, Real-time classification of dance gestures from skeleton animation, dans *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, p. 147–156. 25, 39, 92

[107] Rodriguez, M., J. Ahmed et M. Shah. 2008, Action mach : A spatio-temporal maximum average correlation height filter for action recognition, dans *Conference on Computer Vision and Pattern Recognition*, p. 1–8. 9, 11, 12

[108] Rosales, R. 1998, Recognition of human action using moment-based features, cahier de recherche, BU-1998-020, Boston University, Computer Science. 31

[109] Rousseeuw, P. J. et A. M. Leroy. 2003, *Robust Regression and Outliers Detection*, Wiley. 60

[110] Rubner, Y., C. Tomasi et L. J. Guibas. 2000, The earth mover's distance as a metric for image retrieval, *International Journal of Computer Vision*, p. 99–121. 97

[111] Ryoo, M. 2011, Human activity prediction : Early recognition of ongoing activities from streaming videos, dans *Internationale Conference on Computer Vision*, p. 1036–1043. 15, 38, 90

[112] Ryoo, M. S. et J. K. Aggarwal. 2009, Spatio-temporal relationship match : Video structure comparison for recognition of complex human activities, dans *Internationale Conference on Computer Vision*, p. 1593–1600. 15, 38

[113] Sakoe, H. et S. Chiba. 1978, Dynamic programming algorithm optimization for spoken word recognition, *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 26, p. 43–49. 63, 101

[114] Sakoe, H. et S. Chiba. 1978, Dynamic programming algorithm optimization for spoken word recognition, *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 26, p. 43–49. 99

[115] Salvador, S. et P. Chan. 2004, Fastdtw : Toward accurate dynamic time warping in linear time and space, dans *KDD Workshop on Mining Temporal and Sequential Data*, p. 561–580. 99

[116] Schuldt, C., I. Laptev et B. Caputo. 2004, Recognizing human actions : a local svm approach, dans *International Conference on Pattern Recognition*, p. 32–36. 8, 11, 12

[117] Schuldt, C., I. Laptev et B. Caputo. 2004, Recognizing human actions : A local svm approach, dans *International Conference on Pattern Recognition*, p. 32–36. 14, 33, 38

[118] Scovanner, P., S. Ali et M. Shah. 2007, A 3-dimensional sift descriptor and its application to action recognition, dans *Proceedings of the 15th international conference on Multimedia*, p. 357–360. 15, 38

[119] Shao, L. et X. Chen. 2010, Histogram of body poses and spectral regression discriminant analysis for human action categorization, dans *British Machine Vision Conference*, p. 88.1–11. 8, 20, 31, 38, 90

[120] Shechtman, E. et M. Irani. 2005, Space-time behavior based correlation, dans *Conference on Computer Vision and Pattern Recognition*, p. 405–412. 17, 18, 38

[121] Sheikh, Y., M. Sheikh et M. Shah. 2005, Exploring the space of a human action, dans *Internationale Conference on Computer Vision*, p. 144–149. 24, 39

[122] Shotton, J., A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman et A. Blake. 2011, Real-time human pose recognition in parts from a single depth image, dans *Conference on Computer Vision and Pattern Recognition*, p. 1297–1304. 10, 25, 46, 77, 82, 148

[123] Sminchisescu, C., A. Kanaujia, Z. Li et D. Metaxas. 2005, Conditional models for contextual human motion recognition, dans *Internationale Conference on Computer Vision*, vol. 2, p. 1808–1815. 37

[124] Sullivan, J. et S. Carlsson. 2002, Recognizing and tracking human action, dans *European Conference on Computer Vision*, p. 629–644. 35

[125] Ta, A.-P., C. Wolf, G. Lavoué, A. Baskurt et J.-M. Jolion. 2010, Pairwise features for human action recognition, dans *International Conference on Pattern Recognition*, p. 3224–3227. 31

[126] Tenorth, M., J. Bandouch et M. Beetz. 2009, The TUM Kitchen Data Set of Everyday Manipulation Activities for Motion Tracking and Action Recognition, dans *IEEE International Workshop on Tracking Humans for the Evaluation of their Motion in Image Sequences (THEMIS), in conjunction with ICCV*, p. 1089–1096. 10, 11, 12, 83, 108, 110, 113, 146

[127] Thevenet, Q., M. Lefevre, A. Cordier et M. Barnachon. 2012, Intelligent Interactions : Artificial Intelligence and Motion Capture for Negotiation of Gestural Interactions, dans *Workshop TRUE at ICCBR 2012*. 82

[128] Tran, D. et A. Sorokin. 2008, Human activity recognition with metric learning, dans *European Conference on Computer Vision*, p. 548–561. 8

[129] Tran, K. N., I. A. Kakadiaris et S. K. Shah. 2012, Part-based motion descriptor image for human action recognition, *Pattern Recognition*, vol. 45, n° 7, p. 2562–2572. 8, 22, 39, 90

[130] Tseng, C.-C., J.-C. Chen, C.-H. Fang et J.-J. J. Lien. 2012, Human action recognition based on graph-embedded spatio-temporal subspace, *Pattern Recognition*, vol. 45, n° 10, p. 3611 – 3624. 20, 38

[131] Turaga, P., R. Chellappa, V. S. Subrahmanian et O. Udrea. 2008, Machine Recognition of Human Activities : A Survey, *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 18, n° 11, p. 1473–1488. 7

[132] Turaga, P., A. Veeraraghavan et R. Chellappa. 2008, Statistical analysis on stiefel and grassmann manifolds with applications in computer vision, dans *Conference on Computer Vision and Pattern Recognition*, p. 1–8. 35

[133] Turaga, P., A. Veeraraghavan et R. Chellappa. 2009, Unsupervised view and rate invariant clustering of video sequences, *Computer Vision and Image Understanding*, vol. 113, n° 3, p. 353–371. 37

[134] Van den Bergh, M., W. Servaes, G. Caenen, S. De Roeck et L. Van Gool. 2005, Perceptive user interface, a generic approach, dans *Computer Vision in Human-Computer Interaction*, p. 60–69. 28, 29, 39

[135] Vapnik, V. N. 1995, *The nature of statistical learning theory*. 33

[136] Vicon, M. S. `http://www.vicon.com`. 44

[137] Viterbi, A. 1967, Error bounds for convolutional codes and an asymptotically optimum decoding algorithm, *Information Theory, IEEE Transactions on*, vol. 13, n° 2, p. 260–269. 36, 155

[138] Wang, J., Z. Liu, Y. Wu et J. Yuan. 2012, Mining actionlet ensemble for action recognition with depth cameras, dans *Conference on Computer Vision and Pattern Recognition*, p. 1290–1297. 10, 11, 12, 26, 39, 45, 48, 137, 138, 143, 148, 149

[139] Wang, L. et D. Suter. 2007, Learning and matching of dynamic shape manifolds for human action recognition, *IEEE Transactions on Image Processing*, vol. 16, n° 6, p. 1646–1661. 35

[140] Wang, L. et D. Suter. 2007, Recognizing human activities from silhouettes : Motion subspace and factorial discriminative graphical model, dans *Conference on Computer Vision and Pattern Recognition*, p. 1–8. 37

[141] Wang, L. et D. Suter. 2008, Visual learning and recognition of sequential data manifolds with applications to human movement analysis, *Computer Vision and Image Understanding*, vol. 110, n° 2, p. 153–172. 31

[142] Wang, Y. et G. Mori. 2009, Max-margin hidden conditional random fields for human action recognition, dans *Conference on Computer Vision and Pattern Recognition*, p. 872–879. 8

[143] Weinland, D. et E. Boyer. 2008, Action Recognition using Exemplar-based Embedding, dans *Conference on Computer Vision and Pattern Recognition*, p. 1–7. 18, 32, 33, 35

[144] Weinland, D., E. Boyer et R. Ronfard. 2007, Action recognition from arbitrary views using 3d exemplars, dans *Internationale Conference on Computer Vision*, p. 1–7. 19, 36, 38, 83

[145] Weinland, D., R. Ronfard et E. Boyer. 2006, Automatic discovery of action taxonomies from multiple views, dans *Conference on Computer Vision and Pattern Recognition*, vol. 2, p. 1639–1645. 35

[146] Weinland, D., R. Ronfard et E. Boyer. 2006, Free viewpoint action recognition using motion history volumes, *Computer Vision and Image Understanding*. 9, 11, 12

[147] Weinland, D., R. Ronfard et E. Boyer. 2011, A survey of vision-based methods for action representation, segmentation and recognition, *Computer Vision and Image Understanding*, vol. 115, p. 224–241. 7

[148] Welch, L. R. 2003, Hidden Markov Models and the Baum-Welch Algorithm, *IEEE Information Theory Society Newsletter*, vol. 53, n° 4. 36

[149] Willems, G., J. H. Becker, T. Tuytelaars et L. V. Gool. 2009, Exemplar-based action recognition in video, dans *British Machine Vision Conference*, p. 90.1 – 90.11. 14, 38

[150] Williams, R. 2002, *The Animator's Survival Kit*, Faber & Faber. 32

[151] Wong, S.-F., T.-K. Kim et R. Cipolla. 2007, Learning motion categories using both semantic and structural information, dans *Conference on Computer Vision and Pattern Recognition*, p. 1–6. 15, 38

[152] Xiong, J. et Z. Liu. 2007, Human motion recognition based on hidden markov models, dans *Advances in Computation and Intelligence*, p. 464–471. 20, 38

[153] Yacoob, Y. et M. Black. 1999, Parameterized modeling and recognition of activities, *Computer Vision and Image Understanding*, vol. 73, p. 232–247. 21, 39

[154] Yamato, J., J. Ohya et K. Ishii. 1992, Recognizing human action in time-sequential images using hidden markov model, dans *Conference on Computer Vision and Pattern Recognition*, p. 379–385. 36

[155] Yao, A., J. Gall, G. Fanelli et L. V. Gool. 2011, Does human action recognition benefit from pose estimation ?, dans *British Machine Vision Conference*, p. 67.1–67.11. 26, 39, 76, 83, 92, 108, 110, 147

[156] Yao, A., J. Gall, L. V. Gool et R. Urtasun. 2011, Learning probabilistic non-linear latent variable models for tracking complex activities, dans *Neural Information Processing Systems*, p. 1359–1367. 25, 39, 92, 93, 143

[157] Yeffet, L. et L. Wolf. 2009, Local trinary patterns for human action recognition, dans *Internationale Conference on Computer Vision*, p. 492–497. 8

[158] Yilmaz, A. et M. Shah. 2008, A differential geometric approach to representing the human actions, *Computer Vision and Image Understanding*, vol. 109, n° 3, p. 335–351. 20, 21, 38

[159] Zelnik-Manor, L. et M. Irani. 2001, Event-based analysis of video, dans *Conference on Computer Vision and Pattern Recognition*, p. 123–130. 13, 38

[160] Zhang, Z., Y. Wu, Y. Shan et S. Shafer. 2001, Visual panel : virtual mouse, keyboard and 3d controller with an ordinary piece of paper, dans *Proceedings of the 2001 workshop on Perceptive user interfaces*, p. 1–8. 28, 29, 39

[161] Ziaeefard, M. et H. Ebrahimnezhad. 2010, Hierarchical Human Action Recognition by Normalized-Polar Histogram, dans *International Conference on Pattern Recognition*, p. 3720–3723. 22, 23, 39, 90

Table des figures

Liste des tableaux

Cinquième partie

Annexes

Annexe A

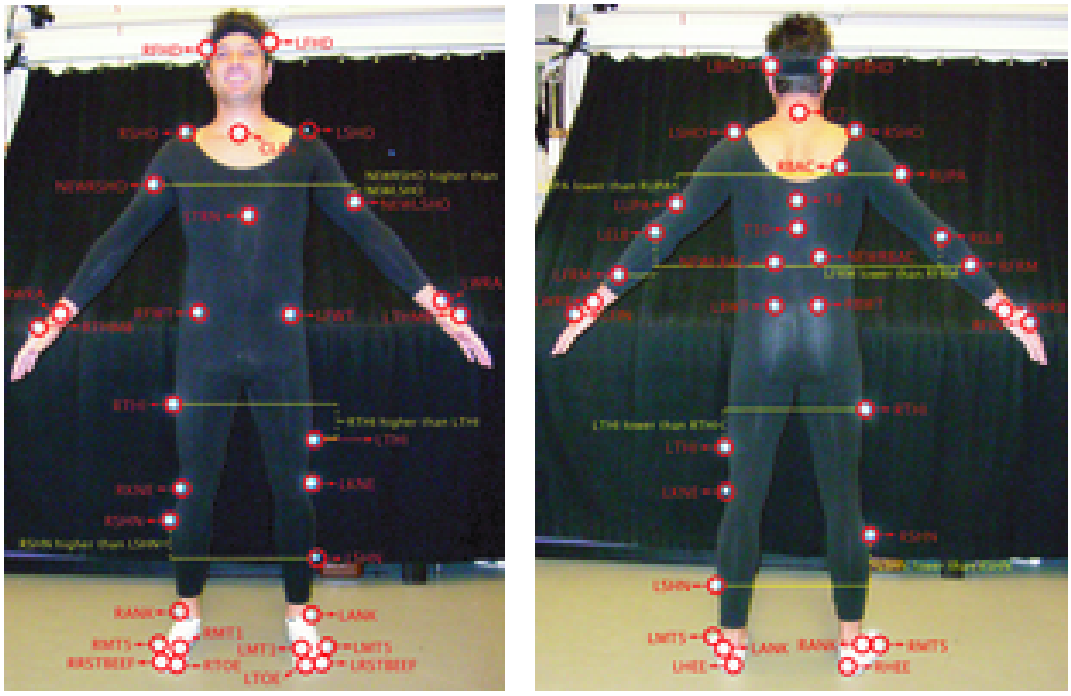
Jeux de données 3D

Sommaire

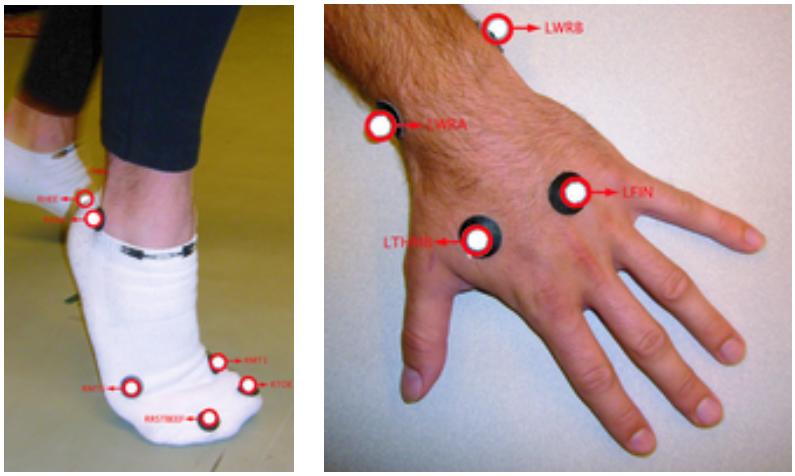
A.1 CMU

La capture de mouvement du jeu de données *Carnegie Mellon University (CMU)* [84] est une capture de mouvement avec marqueurs, tel que le présente la figure A.1. Il s'agit d'un jeux de données d'animations, constitué de 6 catégories distinctes : *Human Interaction*, *Interaction with Environment*, *Locomotion*, *Physical Activities & Sports*, *Situations & Scenarios* et *Test Motions*, pour un total de 144 acteurs.

Nous avons utilisé des sélections de jeu de données, voir tableau A.1. Cette sélection a été construite afin d'avoir des classes similaires aux jeux de données couramment utilisé en reconnaissance d'actions 2D. Cette sélection est composé de 9 classes : *Marche, Course, Coup de poing, Boxe, Saut, Agiter les mains, Rire, Boire* et *Manger*. Ce jeux est composé de 1 action par classe pour l'apprentissage et entre 3 et 6 actions pour la reconnaissance. La figure A.2 représente quelques actions de notre sélection. De nombreuses publications ont travaillé avec des restrictions comparables à la notre : Han et al. [47], Müller et al. [87], Parameswaran et Chellappa [96], Wang et al. [138], Yao et al. [156]. Les auteurs ne fournissant pas les informations nécessaires, nous n'avons pas été en mesure de tester nos algorithmes sur leurs restrictions respectives.



(a) Vue de face (b) Vue de dos

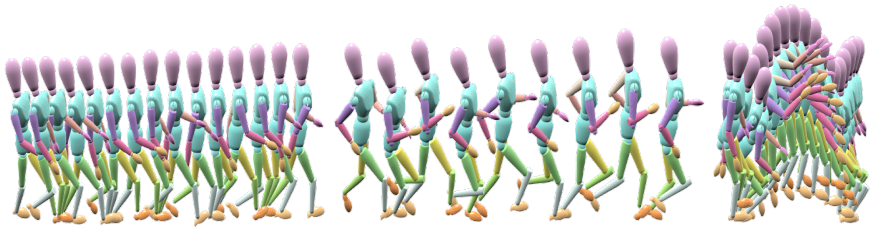


(c) Détail d'un pied (d) Détail d'une main

FIGURE A.1 – Capture de mouvement MoCap CMU [84].

Nom	Appr.	Reconnaissance
Walk	02_01	02_02 ;03_01 ;05_01 ;07_01 ;08_01
Run	02_03	09_01 ;17_01 ;35_22 ;77_10 ;141_02
Punching	143_23	02_05 ;111_19 ;113_13
Boxing	13_17	13_18 ;14_01 ;14_02 ;14_03 ;14_13 ;80_10
Jump	13_32	13_39 ;13_40 ;16_01 ;16_03 ;118_02
Shake hands	18_01	18_02 ;19_01 ;19_02 ;79_06 ;141_23 ;80_73
Laugh	13_14	13_15 ;13_16 ;14_17 ;14_18 ;14_19
Drink	13_09	14_04 ;14_37 ;23_13 ;79_38 ;79_40
Eat	79_12	79_15 ;79_42 ;80_11 ;80_33

TABLE A.1 – Description du jeu de données constitué à partir des captures de mouvement collectées par MoCap CMU [84].



(a) Walk (b) Run (c) Jump

FIGURE A.2 – Exemples d'acquisition du jeu de données CMU.

(a) Lie Down (b) Sit Chair (c) Elbow to Knee

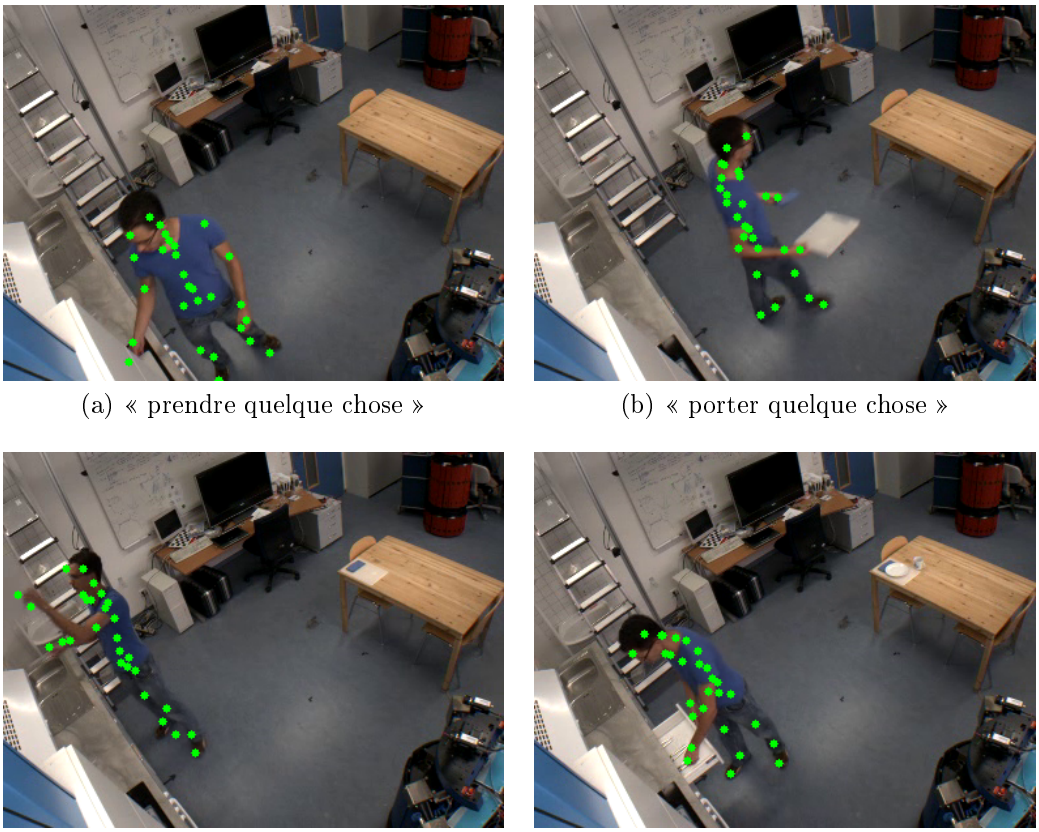
FIGURE A.3 – Exemples d'acquisition du jeu de données HDM.

A.2 HDM

Le jeu de données *Hochschule der Medien (HDM)* [88] est composé de plus de 3h de capture de mouvement. Il est composé de 130 classes distinctes, chacune composée de 10 à 50 réalisations, soit plus de 1500 clips. Ces données sont avant tout à destination des graphistes. La segmentation en clip est faite sur des critères facilitant la recherche de transition entre animations. Ainsi, lors des mouvements de marche, le premier pied utilisé est distingué. On retrouve des classes comme « marcher en commençant du pied gauche » et « marcher en commençant du pied droit ». Nous avons proposé un regroupement de ces clips en 33 classes uniquement, ne tenant pas compte du pied d'appui par exemple. Cela réduit le nombre de classes, mais rend la reconnaissance plus difficile, car certaines actions ne sont composées que de quelques poses parfois très différentes. La figure A.3 présente quelques exemples du jeux de données HDM.

A.3 TUM

Le jeu de données du *Technische Universität München (TUM)* [126] est une collection d'actions faites dans une cuisine. Il s'agit d'action de déplacement d'objet et d'interaction avec le mobilier de la cuisine. Il est à noter que les acteurs ne font pas la cuisine dans ces acquisitions. Cette cuisine dispose également de capteurs supplémentaires, de type RFID ou magnétique pour savoir quels objets sont déplacés



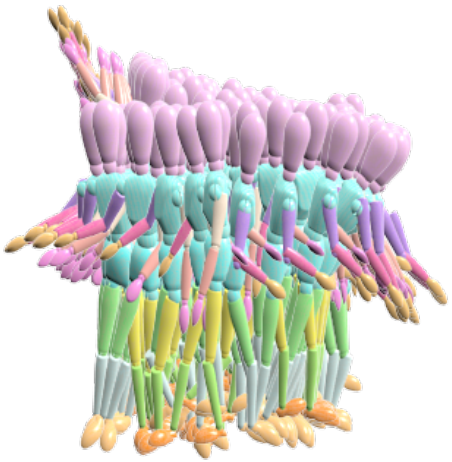
(a) « prendre quelque chose » (b) « porter quelque chose »

(c) « ouvrir un placard » (d) « ouvrir un tiroir »

FIGURE A.4 – Capture de mouvement Bandouch et Beetz [8].

ou si les portes de placard sont ouvertes. L'acquisition est composé de plusieurs séquences de mise en place d'une table pour un repas (d'une seule personne) et la capture de mouvement est faite par un système sans marqueur Bandouch et Beetz [8]. Le jeu de données est composé de 20 séquences d'action. Dans chacune des séquences un acteur met la table et répète donc plusieurs fois les 10 classes identifiées : *Marcher, Être debout, Tendre la main, Prendre, Se baisser, Lâcher, Ouvrir une porte, Ouvrir un tiroir, Fermer une porte* et *Fermer un tiroir*. La capture de mouvements étant non intrusive, il est possible d'utiliser des primitives vidéos en plus des primitives 3D de MoCap, comme l'on fait Yao et al. [155].

La figure A.4 présente des exemples d'actions présente dans le jeu de données TUM, avec le modèle interne du squelette superposé aux images des caméras. La figure A.5 présente la capture de mouvement pour des séquences d'acquisition.



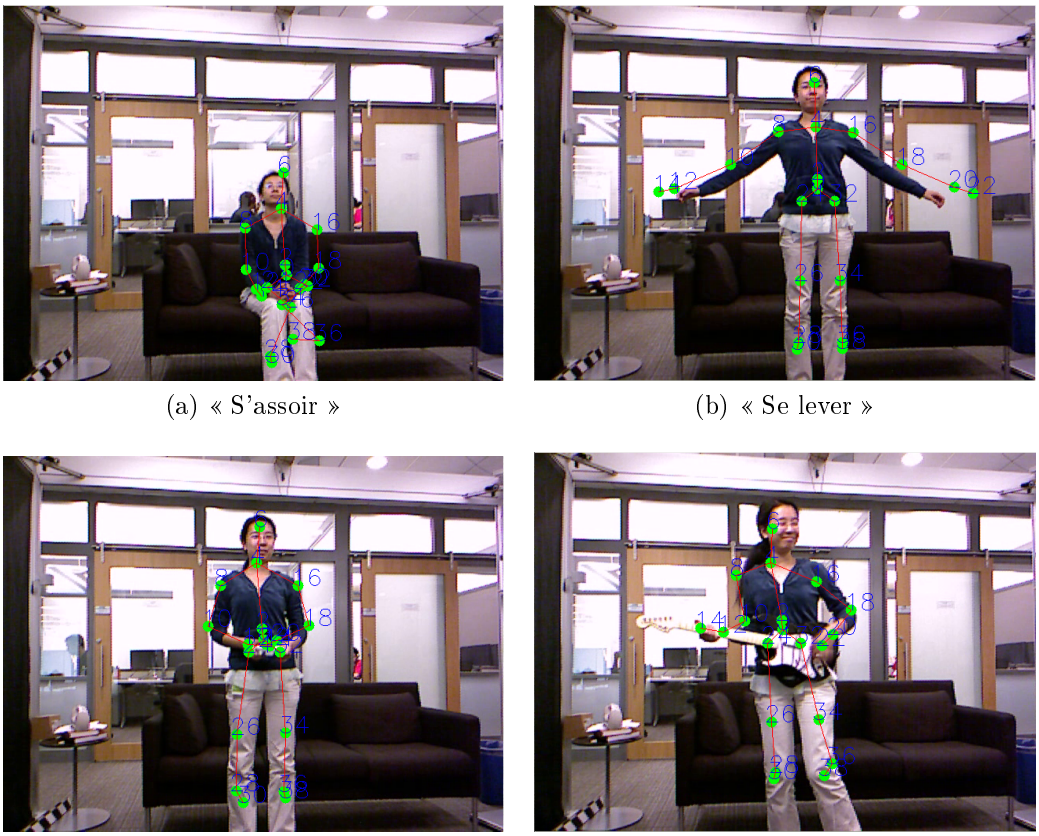
(a) Séquence 0-0

FIGURE A.5 – Exemples de séquences sur le jeu TUM.

A.4 MSR Daily Activity 3D

Le jeu de données *MSR Daily Activity 3D* [138] présente des actions faites dans un salon. Il est composé de 16 classes : *Boire, Manger, Lire un livre, Téléphoner, Écrire, Utiliser un portable, Utiliser un aspirateur, Encourager, Rester assis, Jeter un papier, Jouer à un jeu, S'allonger sur le canapé, Marcher, Jouer de la guitare, Se lever* et *S'assoir*. Chaque classe est faite par 10 acteurs deux fois : une fois assis et une fois debout. Les données sont composées d'une acquisition couleur, d'une acquisition de profondeur et d'une capture de mouvements. Elles ont été enregistrées avec un Kinect™. La capture de mouvements a été faite par la méthode de Shotton et al. [122]. Il est à noter que l'acquisition couleur et l'acquisition de profondeur ne sont pas synchronisées, car elles ont été enregistrées indépendamment.

La figure A.6 présente des exemples d'acquisition en couleur, avec la superposition du squelette.



(a) « S'assoir »

(b) « Se lever »

(c) « Jouer à un jeu »

(d) « Jouer de la guitare »

FIGURE A.6 – Exemple d'acquisition du « MSR Daily Activity 3D » [138].

Annexe B

Modèles de Markov

B.1 Chaînes et ordre

Les modèles de Markov sont une des solutions les plus naturelles pour traiter des données séquentielles. Nous cherchons à mettre en avant des motifs dans la séquence temporelle observée, en corrélant des observations proche de celle-ci. L'exemple commun est celui du temps. Supposons que nous disposons d'une variable binaire permettant de savoir s'il pleut aujourd'hui. À partir de cette observation, nous souhaitons savoir s'il va pleuvoir demain. Si nous nous contentons d'analyser statistiquement les données, nous serons en mesure d'extraire la fréquence relative de pluie. En pratique, nous savons que le temps ne change pas si rapidement, et que s'il a plu les jours précédents, il risque de pleuvoir demain. Dès lors, savoir qu'il pleut aujourd'hui, permet d'affiner notre prédiction du temps pour demain.

Savoir s'il va pleuvoir demain, à partir d'une observation de la météo d'aujourd'hui, peut être vu comme un processus de Markov. Nous relâchons la contrainte d'indépendance temporelle, afin d'exprimer notre modèle probabiliste.

Soit $x_1, x_2, \ldots, x_N$ une séquence d'observation. La probabilité d'observée cette séquence est :

$$p(x_1, x_2, \ldots, x_N) = \prod_{n=2}^{N} p(x_n | x_1, \ldots, x_{n-1}) \tag{B.1}$$

Supposons que la probabilité d'avoir de la pluie demain est indépendante du fait qu'il est plu hier et les jours précédent, et ne dépend que de la pluie de aujourd'hui : dans ce cas, nous avons une *chaîne de Markov du premier ordre.* C'est-à-dire que la distribution conditionnelle est indépendante des précédentes observations, à l'exception de la plus récente. La probabilité de cette séquence

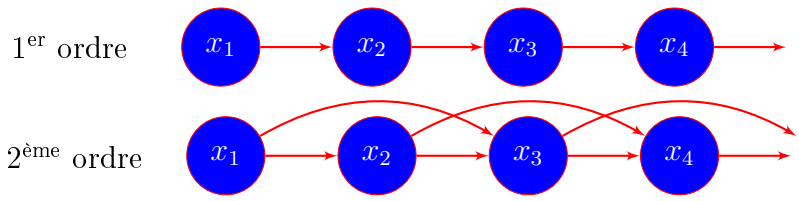


FIGURE B.1 – Exemple de représentation de chaînes de Markov.

devient :

$$p(x_1, x_2, \ldots, x_N) = p(x_1) \cdot \prod_{n=2}^{N} p(x_n|x_{n-1}) \tag{B.2}$$

$$\tag{B.3}$$

Nous déduisons la propriété suivante :

$$p(x_n|x_1, \ldots, x_{n-1}) = p(x_n|x_{n-1}) \tag{B.4}$$

Les chaînes de Markov du premier ordre ne sont pas les seules utilisées, bien qu'elles soient les plus courante. Les chaînes de Markov du second ordre, voir la figure B.1, ont aussi été mise en œuvre. La probabilité d'observé une séquence devient alors :

$$p(x_1, x_2, \ldots, x_N) = p(x_1) \cdot p(x_2|x_1) \cdot \prod_{n=3}^{N} p(x_n|x_{n-1}, x_{n-2}) \tag{B.5}$$

Nous pouvons facilement étendre cette approche à des ordres supérieurs. Néanmoins, le nombre de paramètres devient très important. Ainsi, dans une chaîne de Markov composée de K états. Chaque probabilité conditionnelle $p(x_n|x_{n-1})$ dépend de $K-1$ paramètres. Ceci nous donne $K \times (K-1)$ paramètres pour une chaîne d'ordre 1. Pour une chaîne d'ordre M, le nombre de paramètres devient $K^M \times (K-1)$, il est donc exponentiel avec l'ordre de la chaîne considérée. Ceci conduit en pratique à l'utilisation de système d'ordre faible — entre 1 et 3 — ou d'ordre variable afin de pouvoir évaluer le système en pratique.

Nous avons présenté les chaînes de Markov pour des variables discrètes. Il est possible d'étendre ce principe aux variables continues, notamment par des distributions Gaussiennes. Cette généralisation dépassant le cadre de ce chapitre, le lecteur intéressé est invité à se référer à Bishop [21, Chap. 13].

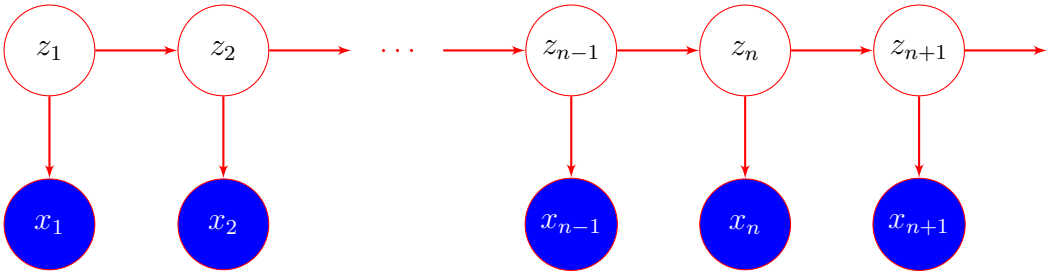


FIGURE B.2 – Exemple de représentation de chaînes de Markov par des états cachés. Les observations ne sont plus liées entre elles, mais les variables latentes le sont.

B.2 Modèle de Markov caché

Il est possible de construire un modèle de Markov avec un nombre limité de paramètres. Pour cela, on introduit une variable cachée ou latente. Pour chaque observation x_n, nous introduisons la variable z_n. Cette variable n'est ni nécessairement du même type que x_n, ni de la même dimension. Au lieu de construire la chaîne de Markov sur les observations, nous construisons une chaîne sur les variables latentes, comme présenté dans la figure B.2.

La probabilité conditionnelle de ce modèle est la suivante :

$$p(x_1, \ldots, x_N, z_1, \ldots, z_N) = p(z_1) \cdot \left[\prod_{n=2}^{N} p(z_n|z_{n-1}) \right] \cdot \prod_{n=1}^{N} p(x_n|z_n) \quad \text{(B.6)}$$

Notons que si les variables latentes sont discrètes, nous obtenons un système de Markov à états cachés ou *Hidden Markov Model* (HMM) (Elliott et al. [37]), et ce quelque soit la nature des observations (discrètes ou continues). Si les variable latentes sont continues, nous obtenons un *système dynamique linéaire*.

Notons $\mathbf{A}$ la matrice des probabilités de transitions entre états, définit par :

$$A_{jk} \equiv p(z_{nk} = 1|z_{n-1,j} = 1) \quad \text{(B.7)}$$

où : $0 \leq A_{jk} \leq 1$ et $\sum_k A_{jk} = 1$, du fait qu'il s'agit de probabilités. La matrice $\mathbf{A}$ est constituée de $K \times (K-1)$ paramètres indépendants. La distribution conditionnelle s'écrit donc :

$$p(z_n|z_{n-1}, \mathbf{A}) = \prod_{k=1}^{K} \prod_{j=1}^{K} A_{jk}^{z_{n-1,j} z_{nk}} \quad \text{(B.8)}$$

Notons que le premier état est spécial, en effet, celui-ci n'a pas de parent. Il dépend donc d'une distribution marginale $p(z_1)$ représenté par un vecteur de probabilité

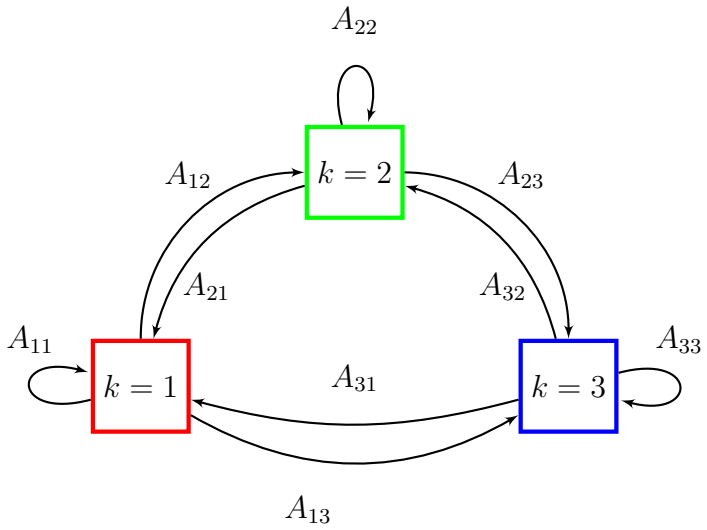


FIGURE B.3 – Exemple de diagramme de transition à 3 états.

π, avec $\pi_k \equiv p(z_{1k} = 1)$, tel que :

$$p(z_1|\pi) = \prod_{k=1}^{K} \pi_k^{z_{1k}} \tag{B.9}$$

où : $\sum_k \pi_k = 1$.

La matrice de transition est souvent représenté par un diagramme de transition, ou par un treillis si elle est développé au cours du temps, voir la figure B.3 pour la représentation en diagramme, et la figure B.4 pour la représentation en treillis.

Le modèle probabiliste est défini par les probabilités conditionnelles des observations $p(x_n|z_n, \phi)$, où ϕ représente les paramètres de la distribution : *les probabilités d'émission*. Dans le cas discret — présenté ici — il s'agit de probabilités conditionnelles. Les probabilités d'émissions se représentent sous la forme suivante :

$$p(x_n|z_n, \phi) = \prod_{k=1}^{K} p(x_n|\phi_k)^{z_{nk}} \tag{B.10}$$

Un modèle de Markov à état cachés est donc définit par :
- $X = \{x_1, \ldots, x_N\}$ les observations ;
- $Z = \{z_1, \ldots, z_N\}$ les variables latentes ;
- $\theta = \{\pi, \mathbf{A}, \phi\}$ les paramètres du modèle.

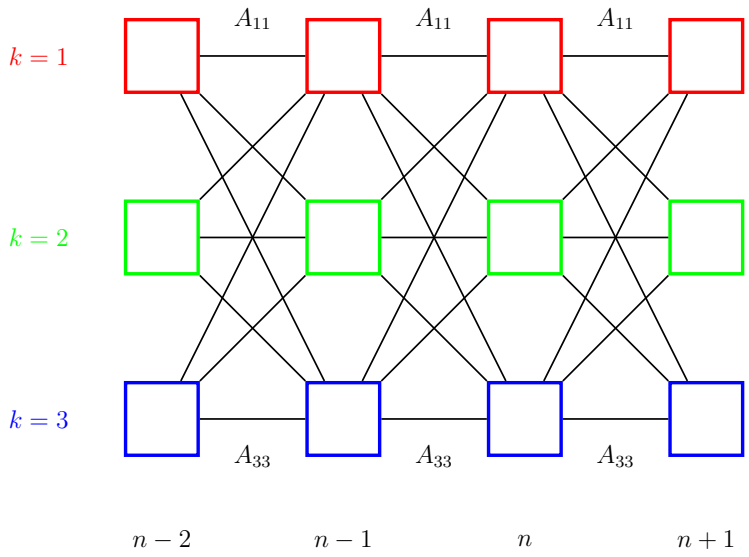


FIGURE B.4 – Exemple de treillis de transition à 3 états.

Nous obtenons la probabilité jointe suivante :

$$p(X, Z|\theta) \quad = \quad p(z_1|\pi) \cdot \left[\prod_{n=2}^{N} p(z_n|z_{n-1}, \mathbf{A})\right] \cdot \prod_{m=1}^{N} p(x_m|z_m, \phi) \qquad \text{(B.11)}$$

Pour modéliser une application par un modèle de Markov, il nous faut construire les états et les observations cachées associées, déterminer la table des transitions (les probabilités d'émissions) et résoudre la succession d'état la plus vraissemblable, à partir des mesures faite. Pour cette dernière état, l'algorithme de Viterbi [137] est généralement utilisé.