Reconnaissance de tâches par commande inverse

Sovannara Hak

THÈSE

présentée

devant l'Institut National des Sciences Appliquées de Toulouse

pour obtenir

le grade de : Docteur de l'Institut National des Sciences Appliquées de Toulouse
Spécialité Systèmes

par

Sovannara HAK

Équipe d'accueil : LAAS-CNRS - Équipe GEPETTO
École Doctorale : Edsys
Titre de la thèse :

Reconnaissance de tâches par commande inverse

Projet R-Blink Contrat ANR-08JCJC-0075-01

Soutenance prévue le 02/11/2011 devant la commission d'examen

Aude	BILLARD
Bernard	ESPIAU
Nicolas	MANSARD
Olivier	STASSE
Rachid	ALAMI
Jean-Paul	LAUMOND

Université de Toulouse

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par :
Institut National des Sciences Appliquées de Toulouse (INSA Toulouse)

Discipline ou spécialité :

Automatique et robotique

Présentée et soutenue par :

le : mercredi 2 novembre 2011

Titre :

Reconnaissance de tâches par commande inverse

Ecole doctorale :
Systèmes (EDSYS)

Unité de recherche :
LAAS-CNRS UPR 8001

Directeur(s) de Thèse :
Jean-Paul Laumond
Nicolas Mansard

Rapporteurs :

Aude Billard
Bernard Espiau

Membre(s) du jury :
Aude Billard
Bernard Espiau
Nicolas Mansard
Olivier Stasse
Rachid Alami
Jean-Paul Laumond

Remerciements

Je remercie tout d'abord Olivier Stasse, Nicolas Mansard et Jean-Paul Laumond mes directeurs de thèse pour m'avoir donné l'opportunité d'effectuer cette thèse dans le domaine de la robotique humanoïde au sein d'une excellente équipe, pour leurs conseils, disponibilité. Merci également à Abderrahmane Kheddar pour m'avoir donner la chance de faire mon stage de fin d'étude au Japon et ainsi de rencontrer mes futurs directeurs de thèse.

Je remercie également les membres de mon jury de thèse : Aude Billard et Bernard Espiau, pour avoir accepté d'être mes rapporteurs, d'avoir pris le temps d'étudier mon manuscrit et d'en tirer des commentaires intéressant. Olivier Stasse, et Rachid Alami, membres du jury, pour leurs intêrets pour mes travaux et la discussion qui en a découlée lors de ma soutenance.

Je remercie également les permanents de l'équipe Gepetto : Florent Lamiraux, Michel Taix, Philippe Souères. Ensuite je remercie (et souhaite un bon courage pour ceux qui n'ont pas encore soutenu) les doctorants du groupe RIA que j'ai pu cotoyer lors de mon séjour au LAAS : Naveed, Mathieu, Ali, Assia, Mokhtar, Diego, Redouane (my frienemy). Un autre merci revient aux personnes que j'ai pu voir et revoir au JRL ou au LAAS au cours de ma thèse : Claire, Diane, Mitsu, Adrien, Jean-Rémy, Paul et Amélie.

Les différentes personnes qui m'ont beaucoup aidé en effectuant les relectures des différents chapitres composant mon manuscrit de thèse afin de déceler fautes d'orthographes, formulations douteuses et erreurs : Antonio, Wassim, François et Roza [1].

Les personnes qui m'ont fait répéter ma soutenance, Antonio, Wassim et surtout Nicolas. Sans tes conseils et tes remarques, la présentation ne se serait pas aussi bien passée. Je remercie toutes les personnes qui ont assisté à ma soutenance, merci Marc et François pour avoir fait le déplacement, ma famille pour le soutien et la préparation du pot.

Je remercie chaleureusement tout les membres du gang Gepetto, dont les capacités de manipulations et d'intimidations ne sont plus à prouver, pour tout les moments partagés au labo et ailleurs : Manish, Oussama, David, Mathieu, Pancho, Minh, Anh, Alireza, Maxime, *los hombres valientes* Sébastien, Thomas, Nicolas P., Duong y señor Oscar, mes professeurs de libanais Antonio [2] et Layale [3] (grâce à vous et au vocabulaire que j'ai acquis, j'ai peut être une chance d'épouser Shakira) ainsi que bien evidemment, Wassim et Wassima les gepettistes par adoption.

Enfin, j'envoie mille merci à toutes les personnes extérieures au laboratoire dont le soutien, sous la forme de gestes ou de paroles, a été grandement aprécié : Roza, Mehdi, Marc, Benoit, François.

Les travaux présentés dans ce manuscrit font partie du projet R-Blink contrat ANR-08JCJC-0075-01.

1. Bonus dédication inconcevable
2. Bonus moustache
3. Bonus saucisse

Publications

[1] S. Hak. Identification de tâches pour le contrôle d'un robot humanoïde. In *Journées Nationales de la Robotique Humanoïde*, Poitiers, France, Juin 2010.

[2] S. Hak, N. Mansard, and O. Stasse. Humanoid robot task recognition from movement analysis. In *International Conference on Humanoid Robots (Humanoids)*, Nashville, États-Unis, Décembre 2010.

[3] S. Hak, N. Mansard, O. Stasse, and J.P. Laumond. Reverse control for a humanoid robot task recognition. *Soumis à IEEE Transactions on Systems, Man and Cybernetics, Part B*, 2011.

[4] O. Ramos, L. Saab, S. Hak, and N. Mansard. Dynamic motion capture and edition using a stack of tasks. In *International Conference on Humanoid Robots (Humanoids)*, Bled, Slovénie, Octobre 2011.

Table des matières

Notations mathématiques et conventions

Dans ce manuscrit, les scalaires sont notés en minuscules comme par exemple s. Les vecteurs sont notés en minuscule grasses : $\mathbf{v}$ et les matrices en majuscules grasses : $\mathbf{M}$. Sauf mention contraire, n représente le nombre de degrés de liberté d'un robot. $\mathbf{I}$ est une matrice identité. Une étoile en exposant dénote une valeur désirée : e^* et un accent circonflexe dénote une valeur observée : $\hat{p}$. Le point et le double point sont utilisés pour noter les vitesses et les accélérations : $\dot{\mathbf{v}}$, $\ddot{\mathbf{a}}$. Des notations supplémentaires seront présentées quand elles seront nécessaires.

Chapitre 1

Introduction

Le développement de la robotique de service est étroitement liée à une grande variété de problèmes scientifiques. L'objectif d'une grande partie de ce domaine de recherche est que l'intégration des robots dans l'environnement humain soit suffisamment fiable et efficace pour être utile. Les exemples de problèmes à résoudre concernent la navigation et la perception de l'environnement. C'est cette classe de problème qui a été tout d'abord résolue et des solutions ont été appliquées avec succès sous l'incarnation de robots aspirateurs, nettoyeurs de piscines ou robots explorateurs.

À ces fonctionnalités de navigations, des fonctionnalités de manipulations peuvent être ajoutées. Les robots mobiles manipulateurs tels que le PR2 [PR2, 2011] sont équipés de bras destinés aux actions de manipulations d'objets situés dans l'environnement (voir figure 1.1). Ces robots sont construits de manière à découpler les fonctionnalités de navigation et de manipulation, il est ainsi possible d'associer directement des espaces aux tâches à effectuer. Par exemple, pour la navigation, les roues du robot vont évoluer dans l'espace défini par le plan du sol. Tandis que pour la manipulation, les actions du bras vont être exprimées dans l'espace cartésien pour asservir la position de l'effecteur à une position désirée. Les deux domaines des actions sont bien découplés, et des opérations de programmation, debuggage, diagnostique et reconnaissance peuvent alors être traitées facilement en effectuant des observations dans ces deux espaces.

La même séparation de fonctionnalités peut aussi s'appliquer aux robots humanoïdes en considérant que les jambes remplissent la fonction de navigation et que le haut du corps s'occupe de la manipulation. La navigation se traduit alors par une planification d'empreintes de pas que le robot doit suivre dans l'espace à deux dimensions représentant le sol pour atteindre un objectif. À la manière des robots manipulateurs, les actions de manipulation seront exprimées dans l'espace cartésien. Cette approche est illustrée dans le scénario *Donne moi la balle rose* [Yoshida *et al.*, 2007], effectué par le robot humanoïde HRP-2 au LAAS-CNRS. La mission du robot est décomposée en sous-missions qui sont gérées par des modules de logiciels indépendants : localiser la balle, marcher vers la balle, attraper la balle, localiser l'opérateur, marcher vers l'opérateur et donner la balle (voir la figure 1.2).

D'une manière générale, le découpage en sous parties fonctionnelles n'est pas toujours possible à cause de la complexité du robot. C'est le cas par exemple d'un robot humanoïde où des couplages entre les différents membre, qui varient en fonction de l'action à réaliser, sont trop nombreux. Le scénario illustré par la figure 1.3 ressemble beaucoup au scénario précédent cependant, la méthodologie appliquée pour accomplir la mission est très différente. Ici, la mission du robot est d'attraper une balle se situant entre ses pieds. Pour accomplir la mission, le robot doit reculer puis tendre la main vers la balle. La mission est accomplie en considérant que les pas à effectuer représentent une

Figure *1.1 – Le robot PR2 de Willow garage.*

extension virtuelle du robot, ramenant la mission à un problème de cinématique inverse avec contraintes [Kanoun *et al.*, 2010]. La marche apparaît alors de façon implicite dans l'action *asservir la position de la main vers la position de la balle* qui décrit la mission.

1.1 Problématique

Dans ce manuscrit, nous cherchons à reconnaître les composantes du mouvement en cours lorsque le découpage fonctionnel n'est pas possible. Nous nous intéressons plus particulièrement aux mouvements faisant intervenir plusieurs objectifs à atteindre. Dans ce cadre, nous cherchons à montrer que la fonction de tâches, classiquement utilisée pour décrire un mouvement à générer, peut être utilisée comme une généralisation du découpage fonctionnel, pour reconnaître de manière explicite des parties couplées du mouvement.

La figure 1.4 illustre la problématique des travaux présentés dans ce manuscrit. Dans l'image de gauche, le robot effectue un mouvement d'atteinte avec sa main droite. Dans l'image de droite, le robot effectue deux mouvements d'atteintes en parallèle. Les deux mouvements sont visuellement très proches. Pourtant le rôle de la main gauche est discriminant. Dans le premier cas, la main gauche est couplée au mouvement de la main droite pour compenser le déséquilibre introduit par le déplacement de la main droite. Dans le second cas, la main gauche agit de manière indépendante pour accomplir l'action d'atteinte. Le problème soulevé est donc de trouver le moyen de différencier les deux mouvements en les observant. Nous montrons qu'en effectuant les observations dans les *bons* espaces, il est possible d'effectuer une reconnaissance de mouvements capable de désambiguïser ces mouvements visuellement proches.

Dans la suite de cette section, nous présentons différentes approches considérées dans le domaine de l'imitation et la reproduction de mouvements. Les travaux en imitation de mouvements sont reliés aux nôtres dans la mesure où le problème de l'imitation consiste à détecter les caractéristiques importantes ou communes d'un ou plusieurs mouvements observés afin de pouvoir les reproduire ou les généraliser.

1.2 État de l'art

La possibilité pour un robot d'apprendre à reproduire des mouvements à partir de démonstrations est intéressante car intuitive. Les démonstrations peuvent par exemple provenir d'un robot identique, ou sur ce même robot par kinestésie (un opérateur va manipuler à la main les membres du robot).

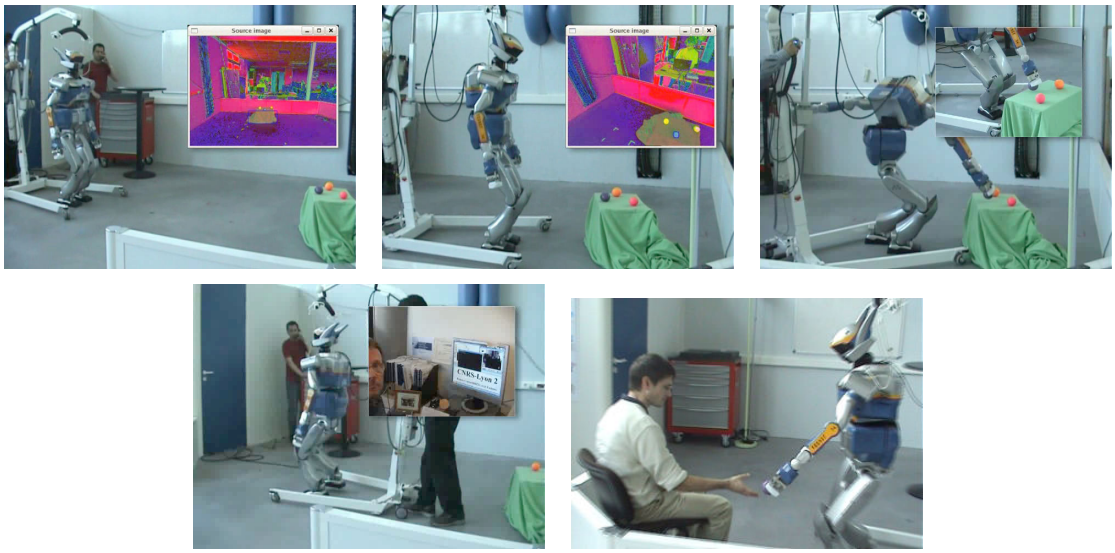


Figure *1.2 – La mission globale [Donne moi la balle]est décomposée en séquences de sous-missions [localiser la balle], [marcher vers la balle], [attraper la balle], [localiser l'opérateur], [marcher vers l'opérateur], et [donner la balle]. Les mouvements [marcher vers], [attraper], [donner] apparaissent comme une séquence structurant l'action (extrait de [Yoshida* et al*., 2007]).*

Dans ce cas, on parle de reproduction de mouvements. On parle d'imitation de mouvements si la démonstration provient d'un humain ou d'un autre type de robot. Le cas échéant, le mouvement doit être transformé pour pouvoir être exécuté au mieux du point de vue de la ressemblance ou des objectifs à atteindre. L'imitation ou la reproduction de mouvements peut porter sur des mouvements spécifiques comme la marche [Benallègue *et al.*, 2010], sur des mouvements du corps complet en respectant les contraintes d'équilibre du robot [Nakaoka *et al.*, 2003, Yamane et Hodgins, 2009, Miura *et al.*, 2009] ou sur les contraintes physiques par optimisation [Suleiman *et al.*, 2008]. Les méthodes d'apprentissage sont largement utilisées pour généraliser les mouvements appris et sont très populaires. Cette généralisation porte par exemple sur un changement dans l'environnement.

Le problème de l'imitation peut être formulé en sous-problèmes s'attaquant à la reconnaissance et à la génération de mouvements. La méthode de reconnaissance de tâches développée dans ces travaux s'appuie sur le modèle de génération de mouvements en se plaçant directement dans les espaces des tâches. Nous comparons les approches étudiées dans le domaine vis-à-vis des espaces utilisés pour représenter ou caractériser les informations véhiculées dans un mouvement.

1.3 Apprentissage et représentation du mouvement

Un aspect critique dans l'apprentissage de mouvement est le choix de l'espace dans lequel l'apprentissage s'effectue, ou autrement dit, des variables qui caractérisent la réalisation des objectifs. Ce choix détermine la représentation du mouvement reflétant l'expressivité des mouvements observés. Plusieurs solutions dans le choix de l'espace d'apprentissage ont été étudiées.

Dans [Shon *et al.*, 2005], il y a deux espaces d'observations : la configuration articulaire d'un humain acquise par capture de mouvements, et l'espace articulaire du robot. L'espace articulaire de l'humain est nécessaire car l'objectif de ces travaux est de réaliser une imitation. Un mouvement est généré sur un modèle du robot, et un humain imite ce mouvement. La trajectoire articulaire de l'humain est enregistrée grâce à un système de capture de mouvements. De cette façon, l'apprentissage porte sur un corpus parallèle de trajectoires de l'humain et du robot. L'association entre les trajectoires articulaires humaines et robotiques est donc manuelle.

Dans [Chalodhorn *et al.*, 2009], l'espace d'observation est l'espace articulaire du robot. L'objectif

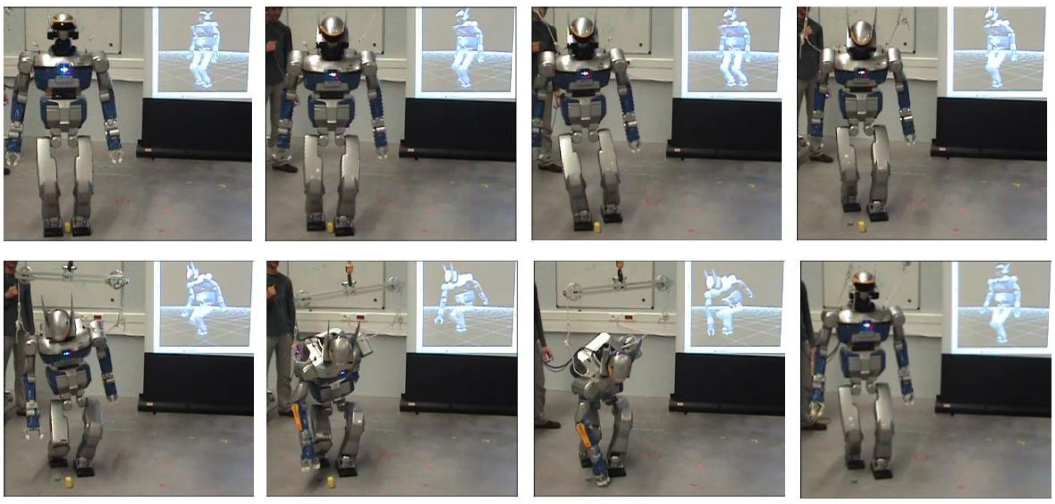


Figure *1.3 – Pour attraper la balle située entre ses pieds, le robot doit s'en éloigner en faisant quelques pas en arrière. Dans cette expérience, la partie* reculer *n'est pas gérée par un module de logiciel. Elle fait partie intégrale de l'action complexe* attraper la balle *(extrait de [Kanoun* et al.*, 2010]).*

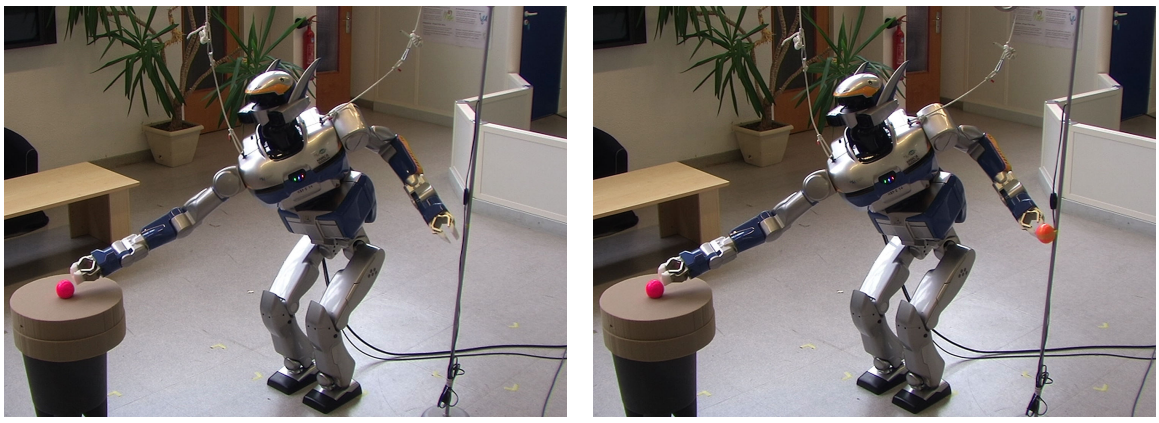


Figure *1.4 – Pour attraper la balle en face de lui (à gauche), le robot atteint une posture dans laquelle le bras gauche est utilisé pour garder son équilibre. Dans la figure de droite, le robot effectue deux actions en parallèle : attraper la balle devant lui tout en attrapant une autre balle se trouvant derrière lui (bien sûr, la balle qui se trouve derrière le robot a été volontairement placée à la position finale de la main gauche illustrée dans la figure de gauche). Il est impossible de différencier visuellement les deux postures. Cependant, est-il possible de discriminer ces deux mouvements ?*

de ces travaux étant d'effectuer des reconnaissances et des reproductions de mouvements. Pour pouvoir appliquer un algorithme d'apprentissage, une technique de réduction de dimension est appliquée : les auteurs proposent une analyse en composantes principales non linéaire avec contrainte circulaire. Il s'agit d'une méthode qui généralise l'analyse en composantes principales car elle permet d'obtenir les relations dans les deux sens entre l'espace de haute dimension et l'espace réduit. La contrainte circulaire permet d'adapter la technique aux courbes fermées. En ne considérant que l'espace articulaire, la sémantique du mouvement n'est pas explicite : l'exécution d'une tâche n'est que la conséquence de la reproduction de la trajectoire articulaire.

Le cas contraire est illustré dans [Herzog *et al.*, 2008]. Les travaux se focalisent sur les mouvements d'un bras droit pour des gestes d'atteintes d'objets. L'objectif étant d'effectuer la reconnaissance et l'imitation de mouvements d'atteintes effectués par un humain sur un robot humanoïde. L'espace d'observation est ici composé des trajectoires dans l'espace cartésiens de points associés à chaque articulation (épaule, coude, pouce, doigt, poing). Le choix de cet espace d'observation donne une interprétation aux mouvements considérés. Les déplacements de ces points sont mis à l'échelle du robot, puis sont utilisés pour le calcul des trajectoires articulaires que le robot doit exécuter pour imiter un mouvement. Ainsi l'algorithme d'apprentissage peut associer le mouvement du bras humain au mouvement du bras du robot. D'une manière similaire, les positions, vitesses et accélérations d'un effecteur sont observées dans [Hersch *et al.*, 2008] pour généraliser et reproduire des mouvements. Les mouvements de démonstrations sont appliqués directement sur le robot par un opérateur. Des informations de visions sont utilisées pour généraliser les mouvements appris à d'autres conditions initiales et gérer les perturbations.

Dans [Montecillo-Puente *et al.*, 2010], les positions et orientations de la tête, des mains, des pieds et du torse d'un humain sont observées. Ces positions sont transposées à un robot humanoïde pour reproduire le mouvement de l'humain en temps réel. Toutes les trajectoires de tâches sont imitées sans faire la distinction des mouvements pertinents ou non.

L'espace d'observation considéré dans [Billard *et al.*, 2003], est l'union de l'espace articulaire d'un bras humain, et des positions dans l'espace cartésien d'objets à manipuler. Dans [Billard *et al.*, 2006, Calinon *et al.*, 2007, Kwon et Park, 2008, Eppner *et al.*, 2009], l'espace articulaire et les positions absolues et relatives entre les effecteurs et les objets à manipuler dans l'environnement sont utilisés comme observations. Ces espaces permettent de représenter d'une manière plus explicite, les tâches effectuées puisque l'ajout des positions cartésiennes des éléments permet de spécifier qu'il n'y a pas que les trajectoires articulaires qui sont importantes dans le mouvement : les déplacements des objets et des effecteurs sont caractéristiques du mouvement. L'espace d'observation est donc spécialisé pour un certain type d'activité. Les variations de l'environnement seront prises en compte dans l'apprentissage pour extraire des descriptions génériques des mouvements démontrés. Dans [Gribovskaya *et al.*, 2011], l'apprentissage se base sur les vitesses et les forces afin de se spécialiser au contexte de l'interaction physique avec un humain.

Dans tous les travaux présentés, le choix des espaces d'observations est directement lié à la nature des mouvements. La spécialisation des espaces permet de décrire (et donc de reconnaître) plus efficacement les mouvements. Notre approche consiste à généraliser la spécialisation en considérant tous les espaces dans lesquels une tâche peut être exécutée.

1.4 Reconnaissance du mouvement

La reconnaissance du mouvement est un problème pouvant s'appliquer à de nombreux domaines comme la vision ou l'interaction homme-machine et s'intègre naturellement aux applications d'imitations. Dans le cas des mouvements humains, la reconnaissance peut porter sur plusieurs niveaux. Il peut s'agir de suivre un humain, d'estimer sa configuration articulaire, reconnaître l'identité d'un humain en se basant sur ses mouvements ou encore interpréter ses actions et ses comportements. Une revue complète et générale de l'analyse de mouvements humains est présentée dans [Moeslund *et al.*, 2006]. Une autre synthèse des travaux sur la reconnaissance de l'action exécutée dans un mouvement est

présentée dans [Krüger *et al.*, 2007].

L'étude des mouvements et des tâches en biomécanique a amené au développement de la théorie des *variétés non contrôlées* [Scholz et Schöner, 1999]. Pendant la réalisation d'une tâche, les degrés de liberté contribuant à la tâche d'un humain seraient classés dans deux catégories : d'une part les degrés de liberté qui sont contrôlés par le système nerveux et d'autre part ceux qui ne le sont pas. Les variables sont considérées comme étant contrôlées si elles sont stabilisées. Dans [Jacquier-Bret *et al.*, 2009], la théorie des variétés non contrôlées est appliquée pour l'étude de mouvements d'atteintes d'un humain en présence d'un obstacle qui va matérialiser des contraintes spatiales. L'objectif est de comprendre comment le système nerveux central gère la redondance du bras pour une tâche d'atteinte en fonction de la présence ou non d'un obstacle. Les différentes expérimentations menées renforcent l'hypothèse selon laquelle le système nerveux central adapte les synergies entre les différents degrés de liberté du bras en fonction des contraintes spatiales (ici des obstacles) et des objectifs. On peut noter la similitude entre les variétés non contrôlées et leurs sous-espaces orthogonals avec le formalisme de la pile de tâches présenté dans le chapitre 2.

1.4.1 Classification de mouvements

Le problème de reconnaissance d'action peut se formuler comme un problème de classification. Dans les travaux en vision présentés dans [Chan *et al.*, 2007], des mouvements humains, issus de séquences d'images, sont classés pour dégager des comportements distincts en utilisant des informations contextuelles aux mouvements du corps. Ces informations sont inspirées de la biomécanique et sont modélisées par un système de connaissance, à base de règles floues qui sont appliquées aux trajectoires articulaires d'un humain. Par exemple, pour un mouvement de marche, les deux bras ne doivent pas être en même temps devant le torse. Les classes de mouvements considérées sont la course et la marche. Elles correspondent à des comportements humains. L'inconvénient majeur de la méthode est qu'il est difficile de trouver et de formuler les règles qui définissent une classe de mouvements. Il est aussi difficile de gérer le grand nombre de règles si on accumule tous les comportements.

Peu de techniques sont développées pour la reconnaissance d'actions simultanées. La méthode présentée dans [Mori *et al.*, 2002] est dédiée aux actions humaines simultanées. Encore une fois, le défaut majeur de cette méthode est qu'une action est représentée par un ensemble d'heuristiques sélectionnées manuellement. Par exemple, pour qu'une action appartienne à la classe *levé de main*, le mouvement doit respecter deux règles : la position de la main doit être au dessus d'un certain seuil, et la main doit se déplacer de bas en haut. Toutes les heuristiques sont appliquées au mouvement observé pour dégager l'ensemble des actions exécutées.

D'autres types de classes peuvent être considérés comme dans [Drumwright et Matarić, 2003, Jenkins et Matarić, 2004]. Ces classes sont focalisées sur des classes sémantiques (un direct, un crochet, un uppercut, une garde). Chaque classe est représentée par des exemples de trajectoires articulaires qui vont créer un espace par balayage. Un classificateur bayesien est utilisé pour calculer la probabilité qu'un mouvement appartienne à une classe particulière sur des trajectoires articulaires observées, en se basant sur les espaces de chacune des classes sémantiques. Des sous-espaces articulaires sont utilisés comme critères de classification de pas de danse dans [Campbell et Bobick, 1995].

La classification présentée dans [Kulić et Nakamura, 2008a] est une classification adaptative en ligne. Les classes évoluent au fur et à mesure des observations. Chaque mouvement observé est encodé sous la forme d'une chaîne de Markov cachée pour être classé. Dans [Takano *et al.*, 2005], les classes de mouvements se basent sur les paramètres des modèles de Markov cachés encodant les mouvements.

1.4.2 Segmentation temporelle

La segmentation temporelle de mouvements est utilisée pour définir la succession de mouvements unitaires ou primitifs dans un mouvement observé. L'outil le plus utilisé dans ce domaine est le modèle de Markov caché. Cet outil a surtout été appliqué dans le domaine de la reconnaissance de

parole [Rabiner, 1989]. Mais ses propriétés mathématiques permettent de l'utiliser pour modéliser des données séquentielles génériques, dont le mouvement.

La segmentation de données de capture de mouvements est étudiée dans [Barbič *et al.*, 2004]. La segmentation est de haut niveau : chaque segment unitaire représente un comportement (marcher, courir, frapper...). Trois approches y sont évaluées en se basant sur les dimensions obtenues par analyse en composantes principales, puis en se basant sur une analyse en composantes principales associée à un modèle probabiliste [Tipping et Bishop, 1999] et enfin en utilisant des mélanges de modèles gaussiens.

Des techniques très simples sont considérées dans [Field *et al.*, 2008] pour la segmentation de mouvements. Celles-ci se basent sur les accélérations des corps en mouvement, les changements de directions (les points où la vitesse devient nulle) et les analyses en composantes principales probabilistes. En animation, dans le cadre des bases de données de mouvements, la segmentation automatique de mouvements peut être utilisée pour annoter automatiquement des mouvements et retrouver certains types de motifs de mouvements [So et Baciu, 2006, Beaudoin *et al.*, 2008]. Dans le même domaine, une segmentation de mouvements est réalisée en recherchant les postures clefs menant à des changements de direction significatifs [So et Baciu, 2005].

Dans [Kulić et Nakamura, 2008b], la segmentation d'un flux de données représentant des mouvements humains, se fait en ligne contrairement à la plupart des méthodes de segmentations. En effet, il est important que l'apprentissage puisse se faire en ligne afin de pouvoir tenir compte de la présence d'humains dans l'environnement du robot. De plus l'approche est incrémentale et permet de prendre en compte les mouvements déjà appris ou appris manuellement dans la segmentation. Cette segmentation se base sur des propriétés des vitesses angulaires des articulations. Dans [Chalodhorn *et al.*, 2009], la segmentation est également effectuée en ligne, et se base sur la projection des données observées dans un espace réduit. Un nouveau segment est créé lorsque les nouveaux points projetés ne correspondent plus au modèle de prédiction de l'espace courant. Les différents segments représentent des phases de marches (marche avant, pas sur le côté, virage). Cette approche ressemble à notre méthode de reconnaissance de tâche présentée dans le chapitre 4, dans le sens où les données observées sont comparées dans un espace approprié à des comportements attendus. La différence est que les espaces que l'on considère sont traités de manière parallèle et sont directement liés à la sémantique du mouvement.

1.5 Génération de mouvements

L'étape finale en imitation est la génération du mouvement à partir de la représentation des mouvements reconnus. Les primitives moteurs permettent d'exprimer le contrôle dans un espace de dimension inférieure à l'espace articulaire. En combinant les primitives, il est alors possible d'obtenir une grande expressivité pour les mouvements générés. Les primitives peuvent être tirées d'un système dynamique, et peuvent être modifiées et commutées en ligne [Degallier *et al.*, 2008]. Dans [Sugihara *et al.*, 2005], les mouvements sont générés en connectant par une fonction lisse et continue, une posture clef et la posture courante. Le mouvement généré tient compte des contraintes dynamiques du robot et la trajectoire du centre de gravité du robot est recalculée.

Dans [Drumwright et Matarić, 2003] à la manière de [Rose *et al.*, 1998], le mouvement est généré en interpolant des exemples appartenant à une classe de mouvements. Dans [Calinon *et al.*, 2007], des mouvements sont généralisés par mélanges de gaussiennes, puis un mouvement est généré par optimisation d'une fonction de mesure d'imitation. Cette phase d'optimisation permet d'adapter le mouvement au contexte courant.

Bien que la technique de réduction de dimension utilisée dans [Chalodhorn *et al.*, 2009] permette d'obtenir directement des trajectoires articulaires grâce à la connaissance des relations dans les deux sens entre l'espace de haute dimension et l'espace réduit, une perte de précision est possible à cause du processus d'encodage et décodage. Ainsi l'application du mouvement sur le robot ne garantit aucune stabilité. C'est pour cette raison que le mouvement généré est ajusté par optimisation d'une fonction de coût représentant des valeurs capteurs attendues en fonction de l'état du robot dans l'espace réduit.

1.6 Approche proposée

Notre approche pour la reconnaissance de mouvements s'inscrit dans la continuité des travaux réalisés au JRL et au LAAS-CNRS au sein de l'équipe Gepetto. Elle se focalise dans l'espace des tâches et par conséquent, à la reconnaissance de tâches. Plutôt que de construire un modèle discriminant ou génératif en utilisant les outils statistiques, nous exploitons le modèle de génération de mouvements. En effet, ce modèle permet d'exprimer directement les corrélations non linéaires entre les variables de mouvements qui sont inhérentes aux mouvements humanoïdes. Les approches statistiques et les techniques de réduction de dimensions ne considèrent généralement que les corrélations linéaires. Nous utilisons le formalisme de la fonction de tâche pour projeter le mouvement observé dans les espaces de tâches, caractérisant un mouvement, ainsi que dans les espaces nuls. Ce formalisme sera présenté dans le chapitre suivant. **Contrairement à la majorité des travaux antérieurs qui s'intéressent aux mouvements composés de séquences de primitives, nous explorons les mouvements complexes issus d'un empilement de contôleurs.**

Nous ne considérons pas les séquencements d'actions ni la généralisation de mouvements : aucun apprentissage n'est réalisé. En effet, nous supposons que les modèles de comportements sont connus puisque ce sont directement les modèles de générations de mouvements utilisés pour contrôler le robot. Notre méthode de reconnaissance de tâche est formulée comme un problème d'identification de tâches actives parmis un ensemble connus de tâches possible. Notre méthode est capable de détecter des tâches exécutées en parallèle en projetant les trajectoires observées dans des espaces caractéristiques. La méthode est aussi capable de gérer les éventuels couplages de tâches grâce à l'utilisation d'un opérateur de projection dans les espaces nuls des tâches.

1.7 Plan du manuscrit

Le chapitre 2 introduit les généralités sur la cinématique inverse et le concept de la fonction de tâches. Ces fonctions de tâches sont utilisées pour la génération de mouvements grâce à une hiérarchisation de contrôleurs. La méthode de génération de mouvements utilisée tout au long des travaux présentés est une implémentation de cette approche : la *pile de tâches*. La méthode de reconnaissance présentée dans ce manuscrit s'appuie sur les propriétés associées à la fonction de tâches. Ces propriétés sont utilisées pour manipuler les observations de mouvements.

Il est important de se placer dans un contexte proches des applications réelles. Ainsi, l'analyse des mouvements est précédée par une phase d'acquisition. Dans nos travaux, l'acquisition des mouvements s'effectue par le biais d'un système de capture de mouvements optique. Le chapitre 3 présente les différentes technologies permettant d'enregistrer un mouvement, le traitement des données collectées ainsi que des exemples d'applications orientées vers la fonction de tâches.

Les techniques présentées dans ces deux chapitres sont exploitées dans le chapitre 4 qui représente le cœur de la contribution apportée dans ces travaux : une méthode de reconnaissance de tâches pour un robot humanoïde, capable de reconnaître des tâches exécutées en parallèles et de manière précise. Les différentes expériences en simulation et sur un vrai robot validant la méthode y sont détaillées.

Chapitre 2

Pile de tâches

La fonction de tâche [Samson *et al.*, 1991] est une approche élégante pour décrire intuitivement des commandes pour un robot dans un espace approprié. Une tâche permet donc de relier l'état d'un robot à un espace choisi. On peut alors prédire les mouvements dans l'espace des tâches connaissant les mouvements du robot (fonction directe) ou bien trouver un mouvement (c'est-à-dire une commande) du robot réalisant un objectif donné dans l'espace des tâches. En se reposant sur la redondance du système, cette approche peut être étendue pour tenir compte d'un ensemble hiérarchisé de tâches [Siciliano et Slotine, 1991]. Il est possible de ranger ces tâches sous la forme d'une pile afin de construire efficacement une loi de commande générant des mouvements complexes [Baerlocher et Boulic, 2004, Mansard et Chaumette, 2007]. Cette organisation en pile autorise des opérations sur les tâches (ajouts, suppressions et permutations) qui préservent la continuité de la commande [Keith *et al.*, 2009]. Ce mode de composition offre une véritable versatilité : en effet, les tâches qui composent une pile sont réutilisables indépendamment, et peuvent être modifiées grâce à leurs paramétrages. De plus les tâches peuvent être exprimées dans les mêmes espaces que les données de capteurs, et il est par conséquent facile de suivre l'évolution des tâches.

Ce chapitre présente d'abord le problème de la géométrie inverse qui consiste à calculer une posture d'un corps poly-articulé en fonction d'un objectif dans l'espace cartésien à atteindre. Les principes et les outils permettant de résoudre ce problème ont un lien direct avec la fonction de tâche. Le formalisme de la fonction de tâche sera ensuite introduit, pour finir sur la présentation de la pile de tâches. Cette pile de tâches, grâce à ses propriétés, pourra être utilisée pour générer une loi de commande ou effectuer une reconnaissance de mouvements.

2.1 Cinématique inverse

Le problème de la cinématique inverse consiste à déterminer le mouvement à appliquer à un corps poly-articulé pour satisfaire un objectif désiré dans un espace de travail (par exemple l'espace cartésien). Le champ d'application de la cinématique inverse s'étend au domaine de la robotique pour le contrôle, en animation graphique et dans les jeux vidéo pour animer des corps poly-articulés tels que des personnages ou des animaux, mais aussi en bioinformatique pour modéliser les mouvements de protéines. La difficulté de ce problème réside dans l'explosion combinatoire due au grand nombre d'articulations à contrôler. Cette partie présente le problème de la cinématique inverse ainsi que les principales méthodes permettant de le résoudre.

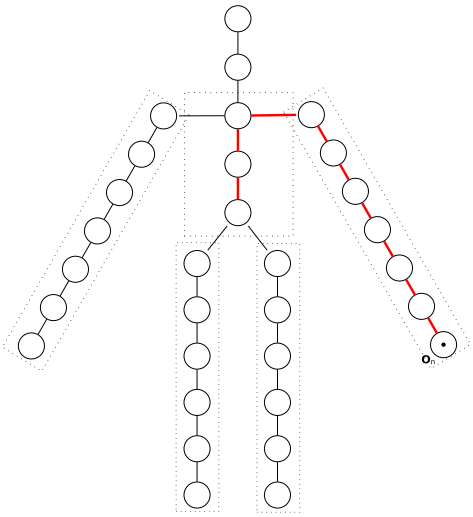


Figure *2.1 – Un exemple de système cinématique de type humanoïde. En fonction de la tâche, une sous-chaine cinématique peut être utilisé (par exemple une sous-chaine associé à un bras).*

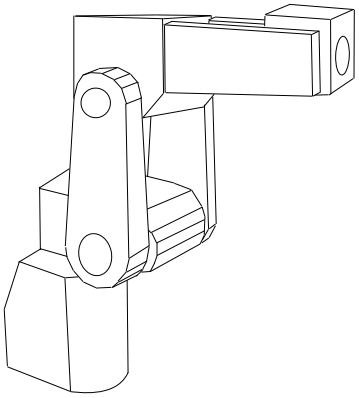


Figure *2.2 – Un exemple de robot de type bras articulé industriel à six degrés de liberté.*

2.1.1 Formulation du problème

La cinématique est l'étude du mouvement des corps d'un point de vue strictement géométrique et temporel. Les forces et les moments agissant sur ces corps ne sont pas considérés. Un système cinématique est composé de corps reliés entre eux par des articulations possédant un ou plusieurs degrés de liberté. Ces systèmes peuvent être représentés sous la forme d'un arbre dont les branches sont des sous-chaînes cinématique (voir la figure 2.1 pour un exemple de système cinématique). Un système cinématique peut être décrit par sa configuration articulaire :

$$\mathbf{q} = (q_0, \ldots, q_n) \in \mathbb{R}^n \tag{2.1}$$

Ces données articulaires peuvent par exemple correspondre à des valeurs angulaires dans le cas où l'articulation est un pivot. L'ensemble de ces données articulaires représente l'espace articulaire. Dans le cas d'un robot de type bras articulé industriel à six degrés de liberté , l'espace articulaire correspond au vecteur de dimension six dont les éléments sont les valeurs angulaires de chaque articulation (voir la figure 2.2).

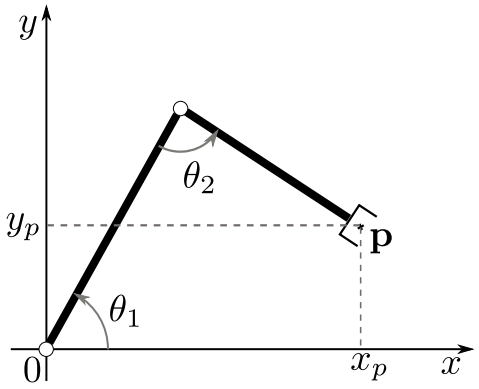


Figure *2.3 – Le problème de géométrie directe consiste à calculer la position du point* $\mathbf{p}$ *en fonction de la configuration du robot* $\mathbf{q} = [\theta_1, \theta_2]$.

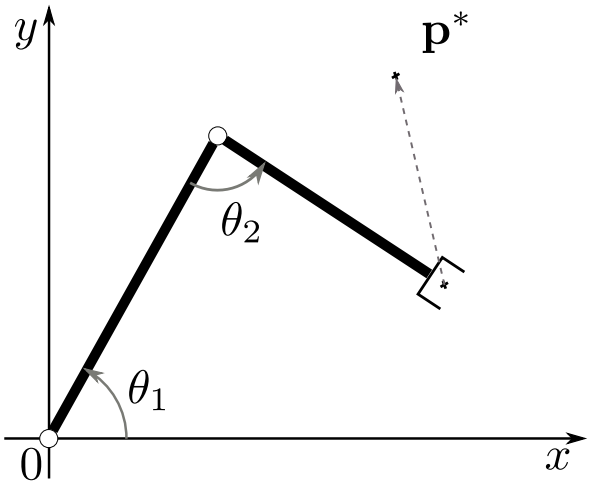


Figure *2.4 – Le problème de la cinématique inverse est de trouver les modifications à apporter à la configuration articulaire* $\mathbf{q}$ *pour qu'un l'effecteur atteigne une position désirée* $\mathbf{p}^*$.

Le problème de géométrie directe consiste à déterminer la position d'un point appartenant au robot à partir d'une configuration articulaire donnée (voir exemple dans la figure 2.3). Tandis que le problème de géométrie inverse consiste à déterminer une configuration articulaire correspondant à une position désirée. Le problème qui consiste à déterminer par la relation inverse $\begin{bmatrix} \mathbf{v} \\ \boldsymbol{\omega} \end{bmatrix} \rightarrow \dot{\mathbf{q}}$ une vitesse articulaire amenant un effecteur à une position précise dans l'espace cartésien en fonction de la configuration articulaire actuelle, est appelé la cinématique inverse (voir la figure 2.4).

2.1.2 Résolution du problème de cinématique inverse

Le problème de la cinématique inverse peut être résolu de plusieurs manières. Pour les robots simples (avec peu de degrés de liberté), le calcul analytique est envisageable. Elle consiste à résoudre un système équations souvent non linéaires. Une fois résolues, le résultat est alors obtenu directement en une seule étape. Il s'agit de l'approche classique en robotique pour le contrôle des bras articulés possédant jusqu'à six degrés de liberté car la solution analytique est rapide à calculer [Craig, 2004]. Malheureusement, dans le cas général d'une structure articulée avec beaucoup de degrés de liberté, il

n'existe pas de solutions analytique calculable.

Il est aussi possible de résoudre le problème de cinématique inverse par apprentissage, via des exemples, de la relation $\begin{bmatrix} \mathbf{v} \\ \boldsymbol{\omega} \end{bmatrix} \rightarrow \dot{\mathbf{q}}$ [D'Souza *et al.*, 2001]. Cette méthode est intéressante lorsque le modèle du robot n'est pas connu avec exactitude. Elle permet aussi de se restreindre aux configurations réalisables sur le robot et d'obtenir des mouvements dont l'allure n'est pas artificielle. C'est particulièrement important lorsqu'il s'agit d'animer des personnages par exemple. Les inconvénients de cette méthode sont hérités des méthodes par apprentissage, c'est-à-dire que la relation échantillonnée $\begin{bmatrix} \mathbf{v} \\ \boldsymbol{\omega} \end{bmatrix} \rightarrow \dot{\mathbf{q}}$ est spécifique au robot considéré et la qualité de la modélisation dépend de la qualité des données d'entrainement.

L'alternative la plus populaire est d'effectuer une résolution numérique. La méthode est itérative, et consiste à linéariser le système autour de l'état courant, puis d'effectuer une descente de gradient afin de converger vers la solution.

2.1.2.1 Formulation linéaire et non-linéaire

Une des méthodes pour résoudre le problème de cinématique inverse consiste à donner à un solveur non-linéaire une fonction de coût pour calculer des postures satisfaisant des contraintes représentant par exemple des contacts [Escande *et al.*, 2006]. Il est aussi possible de linéariser le problème et résoudre itérativement en calculant des petits incréments de la configuration [Nakamura, 1990]. Ces incréments font converger la configuration courante de la chaîne cinématique vers une configuration satisfaisant les objectifs. Ainsi, les itérations successives fournissent un chemin entre la posture initiale et la posture satisfaisant les objectifs, et donc ce chemin peut être utilisé comme une commande pour un robot.

2.1.2.2 Linéarisation

La linéarisation est effectuée en calculant la jacobienne de la fonction :

$$\mathbf{f} \rightarrow \mathbf{f}(\mathbf{q}) = \mathbf{p} \tag{2.2}$$

où $\mathbf{p}$ peut par exemple être la position de l'effecteur d'un robot. La jacobienne est une matrice qui généralise la notion de dérivée de fonction scalaire. Dans le cas de (2.2), la jacobienne $\mathbf{J}$ s'écrit :

$$\mathbf{J} = \begin{pmatrix} \frac{\partial f_1}{\partial q_1} & \cdots & \frac{\partial f_1}{\partial q_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial q_1} & \cdots & \frac{\partial f_m}{\partial q_n} \end{pmatrix} \tag{2.3}$$

La linéarisation permet l'approximation au premier ordre suivante :

$$\Delta\mathbf{p} = \mathbf{J}\Delta\mathbf{q} \tag{2.4}$$

La jacobienne est ensuite inversée pour obtenir une approximation locale de la fonction :

$$\mathbf{p} \rightarrow \mathbf{q} = \mathbf{f}^{-1}(\mathbf{p}) \tag{2.5}$$

On a alors une approximation de l'inverse :

$$\Delta\mathbf{q} = \mathbf{J}^{-1}\Delta\mathbf{p} + o(\|\Delta\mathbf{q}\|^2) \tag{2.6}$$

On pourra donc localement faire varier $\mathbf{q}$ dans la direction de la solution. Cette linéarisation pourra être utilisé pour résoudre de manière itérative le problème de cinématique inverse. Malheureusement, la jacobienne n'est pas toujours inversible. Entres autres, $\mathbf{J}$ n'est pas souvent carrée.

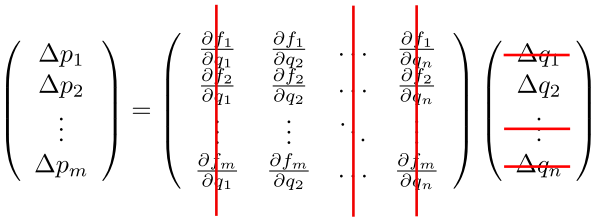


Figure 2.5 – *Illustration de la méthode* cyclic coordinate descent, *dans laquelle les articulations sont traités séquentiellement amenant ainsi à ne considérer qu'une colonne de la jacobienne par étape. L'inversion en est simplifiée.*

2.1.2.3 Simplification du problème d'inversion

Il est possible de contourner la difficulté de l'inversion de la jacobienne en simplifiant le problème. Une première idée est d'utiliser la transposée de la jacobienne plutôt que son inverse pour résoudre le problème de cinématique inverse [Wolovich et Elliott, 1984]. Ainsi contrairement à (2.6) on choisit le prochain $\Delta\mathbf{q}$ comme suit :

$$\alpha\mathbf{J}^T\Delta\mathbf{p} \tag{2.7}$$

L'intuition, derrière cette approximation, est de faire l'analogie avec la relation entre les couples moteurs et les forces dans l'espace cartésien. Si $\mathbf{f}$ est la force qui va amener le robot en direction de l'objectif, et $\boldsymbol{\tau}$ le moment articulaire interne à la chaine cinématique nécessaire, cette relation est :

$$\tau = \mathbf{J}^T\mathbf{f}$$

Ainsi une quantité représentant des déplacements angulaires est liée à une quantité représentant des déplacements dans l'espace par la transposée de la jacobienne.

Cette méthode est rapide car il n'y a pas d'opération d'inversion à effectuer pour calculer la variation de $\mathbf{q}$ à appliquer. Cependant, la convergence n'est pas rapide, et en particulier au voisinage de la solution, puisque les variations à appliquer de chaque articulations sont calculées sans tenir compte de l'influence des autres articulations comme le montre (2.8) pour un exemple de robots à deux degrés de liberté et un objectif de dimension deux.

$$\begin{pmatrix} \Delta\theta_1 \\ \Delta\theta_2 \end{pmatrix} = \alpha \begin{pmatrix} \frac{\partial x}{\partial\theta_1} & \frac{\partial y}{\partial\theta_1} \\ \frac{\partial x}{\partial\theta_2} & \frac{\partial y}{\partial\theta_2} \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} = \alpha \begin{pmatrix} \frac{\partial x}{\partial\theta_1}\Delta x + \frac{\partial y}{\partial\theta_1}\Delta y \\ \frac{\partial x}{\partial\theta_2}\Delta x + \frac{\partial y}{\partial\theta_2}\Delta y \end{pmatrix} \tag{2.8}$$

Une autre méthode itérative appelée *cyclic coordinate descent* consiste à calculer séquentiellement les variations à appliquer à chaque articulation [Luenberger, 1984, Wang et Chen, 1991]. Chaque itération est décomposée en n étapes, pour une chaine articulaire en série de n degrés de liberté. Chaque étape consiste à trouver la variation pour une articulation (en remontant de l'effecteur jusqu'à la base) qui va minimiser la distance entre l'effecteur et l'objectif. Ceci simplifie le calcul d'inversion puisqu'elle ne concerne plus qu'un vecteur de dérivées partielles (une colonne de la jacobienne, voir la figure 2.5). La méthode est rapide, mais converge lentement. De plus, à cause de l'approche séquentielle, les variations des premières articulations auront tendance à être plus grande que les suivantes : les postures calculées peuvent donc paraître non naturelles. Un autre problème est que la méthode est difficile à adapter pour des systèmes à plusieurs effecteurs.

Ces méthodes reposant sur la simplification de l'inversion de la jacobienne n'apportent pas de résultats satisfaisants pour la convergence et la généricité. C'est pour cela que la méthode la plus utilisée est l'utilisation de la pseudo inverse de la jacobienne décrite dans le paragraphe suivant.

2.1.2.4 Pseudo inverse de la jacobienne

La pseudo inverse notée $\mathbf{A}^+$, d'une matrice notée $\mathbf{A}$ est une généralisation de l'inverse d'une matrice [Ben-Israel et Greville, 2003, Moore, 1920, Penrose et Todd, 1955]. Elle peut être calculée pour

n'importe quelle matrice, même celles qui ne sont pas carrées ou de rang plein. Elle possède certaines propriétés de l'inverse d'une matrice. Le type de pseudo inverse le plus utilisé est la pseudo inverse de Moore-Penrose qui est définie par les conditions suivantes :

$$\mathbf{A}\mathbf{A}^{+}\mathbf{A} = \mathbf{A} \tag{2.9}$$

$$\mathbf{A}^{+}\mathbf{A}\mathbf{A}^{+} = \mathbf{A}^{+} \tag{2.10}$$

$$(\mathbf{A}\mathbf{A}^{+})^{T} = \mathbf{A}\mathbf{A}^{+} \tag{2.11}$$

$$(\mathbf{A}^{+}\mathbf{A})^{T} = \mathbf{A}^{+}\mathbf{A} \tag{2.12}$$

La pseudo inverse permet d'obtenir une solution minimale au sens des moindre carrés à la fois de la distance à la solution (0 si la solution est atteignable) et de la norme du vecteur solution d'un système d'équation linéaire. Ainsi pour l'équation (2.4), la pseudo inverse permet d'obtenir la solution homogène :

$$\Delta\mathbf{q}^{*} = \min \arg\min_{\Delta\mathbf{q}} \|\mathbf{J}\Delta\mathbf{q} - \Delta\mathbf{p}\|^{2} = \mathbf{J}^{+}\Delta\mathbf{p} \tag{2.13}$$

Une autre propriété intéressante de la pseudo inverse est que la matrice $(\mathbf{I} - \mathbf{J}^{+}\mathbf{J})$ effectue une projection dans le noyau de $\mathbf{J}$. Cela signifie que pour tout vecteurs $\mathbf{z}$, $\mathbf{J}(\mathbf{I} - \mathbf{J}^{+}\mathbf{J})\mathbf{z} = \mathbf{0}$ (direct d'après (2.10)). Cela permet d'obtenir d'autres solutions pour $\Delta\mathbf{q}$:

$$\Delta\mathbf{q} = \mathbf{J}^{+}\Delta\mathbf{p} + (\mathbf{I} - \mathbf{J}^{+}\mathbf{J})\mathbf{z} \tag{2.14}$$

$\Delta\mathbf{q}$ minimise encore $\|\mathbf{J}\Delta\mathbf{q} - \Delta\mathbf{p}\|^{2}$. La minimisation de $\|\Delta\mathbf{q}\|$ est alors perdue, sauf dans le cas $\mathbf{z} = 0$. Nous montrerons plus tard que cette propriété permet de gérer des commandes secondaires en utilisant $\mathbf{z}$ pour faire varier la solution selon des critères choisis.

2.1.2.5 Calcul de la pseudo inverse

Une méthode efficace pour calculer la pseudo inverse d'une matrice est d'effectuer une décomposition en valeurs singulières (singular value decomposition, SVD) sur cette matrice. Les propriétés de cette décomposition simplifient le calcul de la pseudo inverse. La décomposition en valeurs singulières d'une matrice $\mathbf{J}$ de rang r consiste à exprimer $\mathbf{J}$ sous la forme :

$$\mathbf{J} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{T}$$

où $\mathbf{U}$ et $\mathbf{V}$ sont des matrices orthogonales et $\mathbf{\Sigma}$ est de la forme :

$$\mathbf{\Sigma} = \begin{pmatrix} \mathbf{D} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix} \tag{2.15}$$

$\mathbf{D}$ est diagonale carrée de dimension r et $\mathbf{0}$ sont des matrices ne contenant que de éléments nuls. Si $\mathbf{J}$ est de dimension $m \times n$, alors $\mathbf{U}$ est de dimension $m \times m$, $\mathbf{\Sigma}$ est de dimension $m \times n$ et $\mathbf{V}$ est de dimension $n \times n$ (voir la figure 2.6). Les seuls membres non nuls de la matrice $\mathbf{D}$ sont les valeurs σ_i selon la diagonale avec $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r \geq 0$.

Soient $\mathbf{u}_i$ et $\mathbf{v}_i$ les $i^{ème}$ colonnes de $\mathbf{U}$ et $\mathbf{V}$. Les r premières colonnes de $\mathbf{U}$ forment une base orthonormée de l'image de $\mathbf{J}$, et les vecteurs $\mathbf{v}_{r+1}, \ldots, \mathbf{v}_n$ forment une base orthonormée du noyau de $\mathbf{J}$. La décomposition en valeurs singulières existe toujours. En ne considérant que les r premier vecteur, $\mathbf{J}$ peut être réduite à :

$$\mathbf{J} = \sum_{i=1}^{r} \sigma_i \mathbf{u}_i \mathbf{v}_i^{T} \tag{2.16}$$

Les matrices $\mathbf{U}$ et $\mathbf{V}$ sont orthogonales, et donc $\mathbf{U}^{-1} = \mathbf{U}^{T}$ et $\mathbf{V}^{-1} = \mathbf{V}^{T}$. $\mathbf{D}$ est diagonale, et donc son inverse est obtenu en remplaçant tout ses coefficients non nuls par leurs inverses. Ainsi, le calcul de la pseudo inverse de $\mathbf{J}$ est directe :

$$\mathbf{J}^{+} = \mathbf{V}\mathbf{\Sigma}^{+}\mathbf{U}^{T} = \mathbf{V}\begin{pmatrix} \mathbf{D}^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix}\mathbf{U}^{T} = \sum_{i=1}^{r} \sigma_i^{-1} \mathbf{v}_i \mathbf{u}_i^{T} \tag{2.17}$$

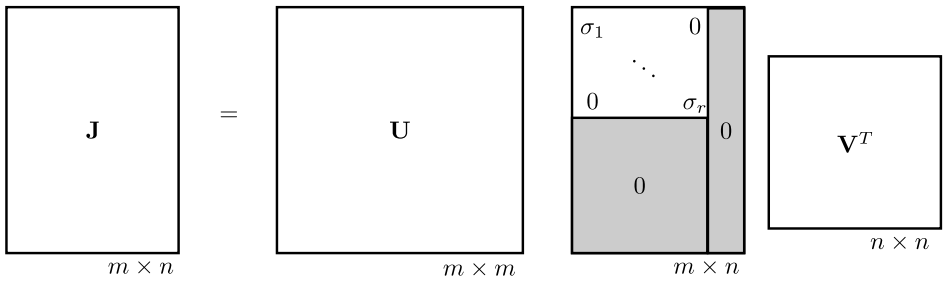
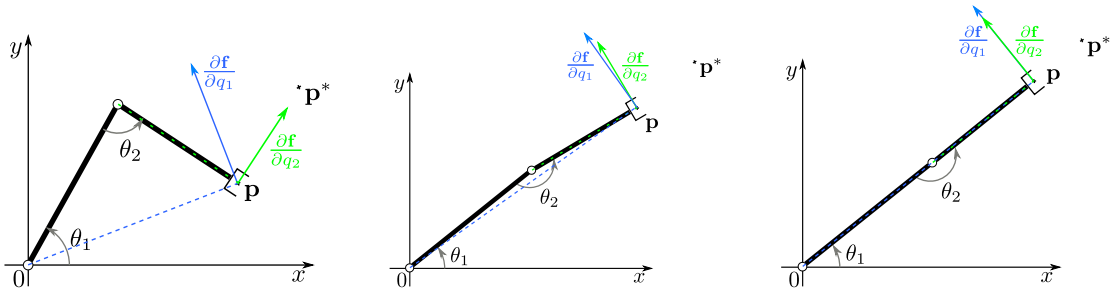


Figure 2.6 – *Décomposition en valeurs singulières d'une matrice* $\mathbf{J}$.



(a) La posture est non singulière. (b) Au voisinage de la singularité. (c) En singularité.

Figure 2.7 – *Illustration du problème de singularité pour un robot à deux degrés de liberté en rotation. Dans la figure 2.7(a), il n'y a pas de singularité. La figure 2.7(b) illustre le cas au voisinage de la singularité : les directions de déplacement dû aux variations des articulations* $\frac{\partial \mathbf{f}}{\partial q_1}$ *et* $\frac{\partial \mathbf{f}}{\partial q_2}$ *sont très proches. La figure 2.7(c) montre un exemple de singularité : l'objectifs* $\mathbf{p}^*$ *n'est plus accessible par combinaisons linéaire, alors que dans une autre position de départ,* $\mathbf{p}^*$ *aurait été accessible.*

2.1.2.6 Singularités

Les configurations singulières sont définies par l'ensemble :

$$\{q \in \mathbb{R}^n | \text{rank}(\mathbf{J}(\mathbf{q})) < \text{rank}(\mathbf{J}(\mathbf{q}_0))\} \tag{2.18}$$

où $\mathbf{q}_0$ est la configuration articulaire initiale. Pour ces configurations, la dimension de l'espace $\mathbb{I}(\mathbf{q}) = \text{Image}(\mathbf{J}(\mathbf{q}))$ diminue brusquement. Le problème de la pseudo inverse se pose à la frontière extérieure des singularités. En effet, la pseudo inverse est une fonction continue des matrices dans les ensembles de rang constant. Lors de la perte de rang, une discontinuité de la fonction pseudo inverse se produit. Elle s'accompagne d'une limite infinie lorsqu'on se rapproche de la frontière extérieure de la singularité. La décomposition en valeurs singulières montre plus clairement ce problème : lorsque les valeurs des coefficients σ_i sont très faibles, leurs inverses, nécessaires dans le calcul de la pseudo inverse, vont tendre vers l'infini et rendre les solutions arbitrairement grandes autour de ces configurations. Les problèmes apparaissent donc aux voisinages des singularités.

Dans l'exemple de la figure 2.7, le déplacement de l'effecteur pour un bras articulé à deux degrés de liberté en rotation est représenté par les bras de levier. En singularité, les coefficients σ_i sont nuls et donc $\frac{\partial \mathbf{f}}{\partial q_1}$ et $\frac{\partial \mathbf{f}}{\partial q_2}$ sont parallèles, $\mathbb{I}(q)$ perd brusquement une dimension et l'objectif $\dot{\mathbf{p}}^*$ n'est plus accessible par combinaisons linéaire, alors que dans une autre position, $\dot{\mathbf{p}}^*$ aurait été accessible.

2.1.2.7 Amortissement de la pseudo inverse

Pour gérer les singularités, il est possible de modifier la décomposition en valeurs singulières afin d'introduire un facteur d'amortissement [Nakamura et Hanafusa, 1986, Wampler, 1986]. Ce facteur va garantir que la norme de la solution reste sous une valeur prédéfinie dépendant uniquement de la

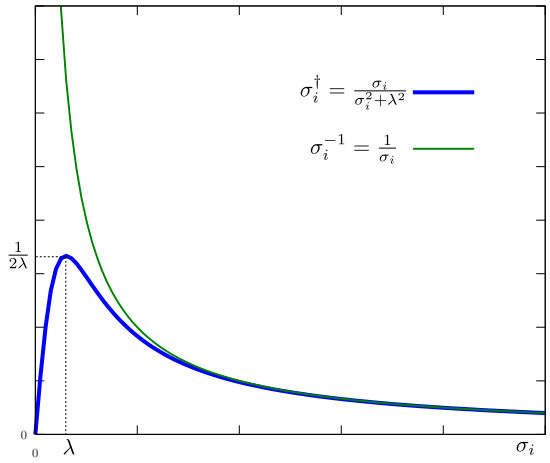


Figure 2.8 – *Illustration du comportement de σ_i^{-1} et de $\sigma_i^\dagger$.*

norme de $\mathbf{p}$ (et indépendante de $\mathbf{J}$). Le problème (2.13) est modifié pour limiter la norme de $\Delta\mathbf{q}$:

$$\Delta\mathbf{q}^* = \min \arg\min_{\Delta\mathbf{q}} \|\mathbf{J}\Delta\mathbf{q} - \Delta\mathbf{p}\|^2 - \eta^2\|\Delta\mathbf{q}\|^2 \tag{2.19}$$

η est le facteur d'amortissement qui va permettre de faire un compromis entre l'erreur de $\|\mathbf{J}\Delta\mathbf{q}-\Delta\mathbf{p}\|^2$ et la norme de la solution. Le cas $\eta = 0$ correspond à (2.13). Pour un $\eta > 0$, il est prouvé que la solution de ce problème est :

$$\Delta\mathbf{q}^* = \mathbf{J}^T(\mathbf{J}\mathbf{J}^T + \eta^2\mathbf{I})^+\Delta\mathbf{p} = \mathbf{J}^\dagger\Delta\mathbf{p}$$

Et finalement la décomposition en valeurs singulières de la pseudo inverse amortie $\mathbf{J}^\dagger$ amène au résultat suivant :

$$\mathbf{J}^\dagger = \mathbf{J}^T(\mathbf{J}\mathbf{J}^T + \eta^2\mathbf{I})^+ = \sum_{i=1}^{k} \frac{\sigma_i}{\sigma_i^2 + \eta^2}\mathbf{v}_i\mathbf{u}_i^T \tag{2.20}$$

On remarque que l'inverse des coefficients σ_i sont remplacés par :

$$\sigma_i^\dagger = \frac{\sigma_i}{\sigma^2 + \eta^2} \tag{2.21}$$

qui est continue et borné par $\frac{1}{2\eta}$ (voir la figure 2.8). Lorsque les σ_i sont grands $\mathbf{J}^\dagger \approx \mathbf{J}^+$. Donc la méthode de l'amortissement par les moindres carrés agit comme celle de la pseudo inverse loin des singularités, et adoucit les résultats de la pseudo inverse au voisinage des singularités.

2.2 La fonction de tâche

Dans la partie précédente, on ne s'est intéressé qu'aux relations entre des positions et vitesses de l'effecteur et l'espace des configurations. La fonction de tâche permet de généraliser ce concept à d'autres fonction que la position de l'effecteur. Une fonction de tâche permet de relier l'espace articulaire à un espace de tâche :

$$\begin{array}{rccc} \mathbf{e}: & \mathbb{R}^n & \rightarrow & \mathbb{R}^m \\ & \mathbf{q} & \mapsto & \mathbf{e}(\mathbf{q}) \end{array} \tag{2.22}$$

L'espace de la tâche est l'espace dans lequel est exprimé une tâche robotique. L'espace de la tâche est choisi en fonction de la nature de la tâche robotique à effectuer, et peut par exemple être l'espace

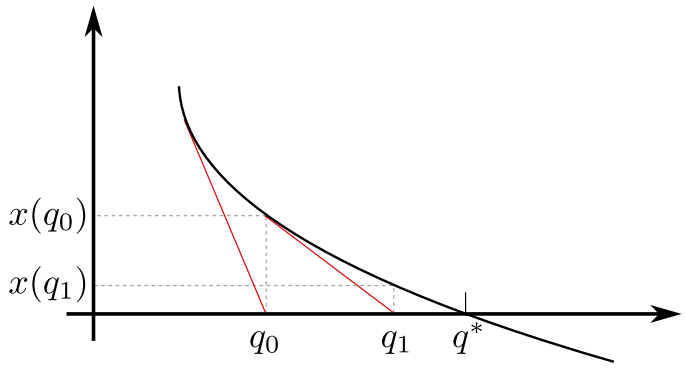


Figure *2.9 – La méthode de Newton-Raphson pour résoudre itérativement* $x(q) = 0$.

cartésien ou encore un sous espace de l'espace des configurations. Un cas particulier est bien sûr une paramétrisation de la position de l'effecteur, étudié dans le paragraphe 2.1.

L'exécution d'une tâche peut être considérée comme la régulation à zéro de la fonction de tâche associée. En prenant par exemple comme fonction de tâche $\mathbf{e}(\mathbf{q}) = \mathbf{s} - \mathbf{s}^*$ correspondant à l'erreur entre un signal $\mathbf{s}$ et sa valeur désirée $\mathbf{s}^*$. Ainsi, la jacobienne $\mathbf{J}$ relie le vecteur erreur au vecteur de configuration :

$$\dot{\mathbf{e}} = \frac{\partial \mathbf{e}}{\partial \mathbf{q}} \dot{\mathbf{q}} = \mathbf{J}\dot{\mathbf{q}} \tag{2.23}$$

Le comportement de la fonction de tâche doit être défini. Habituellement, en robotique, ce comportement est une décroissance exponentielle :

$$\dot{\mathbf{e}}^* = -\lambda \mathbf{e} \tag{2.24}$$

où λ est un paramètre positif permettant d'ajuster la vitesse de convergence. $\mathbf{J}$ n'étant pas toujours inversible, sa pseudo-inverse $\mathbf{J}^+$ est utilisée pour inverser (2.23) :

$$\dot{\mathbf{q}} = \mathbf{J}^+ \dot{\mathbf{e}}^* \tag{2.25}$$

La cinématique inverse est généralement utilisée pour résoudre le problème de géométrie inverse consistant à trouver une configuration $\mathbf{q}$ telle que $\mathbf{e}(\mathbf{q}) = 0$. Pour qu'un robot puisse exécuter une tâche, la résolution de (2.25) est itérative.

En choisissant (2.24) dans (2.25), on retrouve la méthode itérative de Newton-Raphson (voir figure 2.9) consistant à faire converger $\mathbf{e}(\mathbf{q})$ vers 0 de manière itérative. En utilisant des gains λ suffisamment petits, la suite de positions obtenue lors des itérations successives constitue de plus un chemin reliant la position initiale à la solution problème de géométrie inverse. Ce chemin peut ensuite être utilisé comme commande au robot.

2.3 Redondance et hiérarchie de tâches

La solution (2.25) peut être généralisée en utilisant le formalisme de la redondance présenté dans [Siciliano et Slotine, 1991]. Le robot devient redondant par rapport à une tâche donnée en un point $\mathbf{q}$ lorsque la dimension du tangeant à l'espace de configurations en $\mathbf{q}$ est supérieure à la dimension de l'espace $\mathbb{I}(\mathbf{q}) = \text{Image}(\mathbf{J}(\mathbf{q}))$ de la tâche considérée. Ainsi la généralisation de (2.25) donne :

$$\dot{\mathbf{q}} = \mathbf{J}^+ \dot{\mathbf{e}}^* + \mathbf{P}\mathbf{z} \tag{2.26}$$

où $\mathbf{P} = \mathbf{I} - \mathbf{J}^+\mathbf{J}$ est le projecteur orthogonal dans l'espace nul de $\mathbf{J}$, et $\mathbf{z}$ est un vecteur représentant une commande secondaire qui, grâce au projecteur, ne va pas perturber la tâche $\dot{\mathbf{e}}^*$ en exploitant par

exemple les degrés de liberté non utilisés. Une hierarchie de tâches est ainsi établie en définissant la tâche principale comme étant prioritaire par rapport à une tâche qui va se réaliser grâce à la commande secondaire.

2.3.1 Loi de commande pour deux tâches

La commande secondaire $\mathbf{z}$ introduite dans (2.26) peut être utilisée pour exécuter au mieux une seconde tâche en définissant $\mathbf{z}$ comme étant la commande qui va réaliser la relation de référence de la tâche : $\dot{\mathbf{e}}_\mathbf{2}^* = \mathbf{J}_2\dot{\mathbf{q}}$. En introduisant (2.26) dans cette dernière équation on obtient :

$$\dot{\mathbf{e}}_\mathbf{2}^* = \mathbf{J}_2\mathbf{J}^+\dot{\mathbf{e}}^* + \mathbf{J}_2\mathbf{P}\mathbf{z} \tag{2.27}$$

Cette dernière équation permet de calculer $\mathbf{z}$. La commande associée pour réaliser deux tâches $\dot{\mathbf{e}}_\mathbf{1}$ et $\dot{\mathbf{e}}_\mathbf{2}$ (qui sera de priorité inférieure à $\dot{\mathbf{e}}_\mathbf{1}$) ayant comme consigne $\dot{\mathbf{e}}_\mathbf{1}^*$ et $\dot{\mathbf{e}}_\mathbf{2}^*$, s'écrit :

$$\dot{\mathbf{q}} = \mathbf{J}_1^+\dot{\mathbf{e}}_\mathbf{1}^* + \mathbf{P}_1(\mathbf{J}_2\mathbf{P}_1)^+(\dot{\mathbf{e}}_2^* - \mathbf{J}_2\mathbf{J}_1^+\dot{\mathbf{e}}_\mathbf{1}^*) \tag{2.28}$$

Etant donné que $\mathbf{P}_1$, l'opérateur de projection, est hermitien et idempotent, (2.28) s'écrit plus simplement :

$$\dot{\mathbf{q}} = \mathbf{J}_1^+\dot{\mathbf{e}}_\mathbf{1}^* + (\mathbf{J}_2\mathbf{P}_1)^+(\dot{\mathbf{e}}_2^* - \mathbf{J}_2\mathbf{J}_1^+\dot{\mathbf{e}}_\mathbf{1}^*) \tag{2.29}$$

Un couplage apparait donc entre les deux tâches : la réalisation de $\dot{\mathbf{e}}_\mathbf{2}$ est modifiée pour ne pas perturber $\dot{\mathbf{e}}_\mathbf{1}$.

2.3.2 Loi de commande pour un nombre arbitraire de tâches

La loi de commande calculée pour deux tâches est étendue à un nombre arbitraire de tâches par récursion. Ces tâches sont ordonnées par priorités : une tâche de priorité 1 étant la tâche de plus haute priorité, et une tâche de priorité n la plus basse. Ce qui équivaut à dire qu'une tâche $\dot{\mathbf{e}}_i$ ne doit pas perturber une tâche $\dot{\mathbf{e}}_j$ si $i > j$. La formulation récursive s'écrit [Siciliano et Slotine, 1991] :

$$\begin{cases} \dot{\mathbf{q}}_0 & = 0 \\ \dot{\mathbf{q}}_i & = \dot{\mathbf{q}}_{i-1} + (\mathbf{J}_i\mathbf{P}_{i-1}^A)^+(\dot{\mathbf{e}}_i^* - \mathbf{J}_i\dot{\mathbf{q}}_{i-1}), \quad i = 1 \dots n \end{cases} \tag{2.30}$$

où $\mathbf{P}_{i-1}^A$ est le projecteur dans l'espace nul de la jacobienne augmentée $\mathbf{J}_i^A = (\mathbf{J}_1, \dots \mathbf{J}_i)$. La vitesse articulaire réalisant toutes les tâches est $\dot{\mathbf{q}}^* = \dot{\mathbf{q}}_n$. Une formulation récursive pour le calcul du projecteur $\mathbf{P}_{i-1}^A$ est proposée par [Baerlocher et Boulic, 1998].

$$\begin{cases} \mathbf{P}_0^A & = \mathbf{I} \\ \mathbf{P}_i^A & = \mathbf{P}_{i-1}^A - (\mathbf{J}_i\mathbf{P}_{i-1}^A)^+(\mathbf{J}_i\mathbf{P}_{i-1}^A), \quad i = 1 \dots n \end{cases} \tag{2.31}$$

où $\mathbf{I}$ est la matrice identité, $\mathbf{J}_i$ est la matrice jacobienne de la tâche i. Une implémentation complète de cette approche est proposée et détaillée dans [Mansard et Chaumette, 2007] sous le nom de *pile de tâches*. La représentation du mouvement par une pile de tâches permet d'ajouter des tâches, d'en supprimer ou de permuter les ordres de priorités entre deux tâches facilement. L'implémentation utilisée préserve la continuité de la loi de commande durant ces opérations. Les contraintes, comme les limites articulaires, peuvent être prises en compte localement.

2.4 Applications de la pile de tâches

Dans ce paragraphe les différents types de tâches robotiques utilisées dans la suite des travaux sont décrites. Puis un exemple d'exécution de tâches en parallèle est donné pour illustrer les effets des couplages des tâches. Dans le chapitre suivant nous présentons un exemple applicatif de génération de mouvement dans le cadre de l'imitation sans reconnaissance d'un mouvement humanoïde.

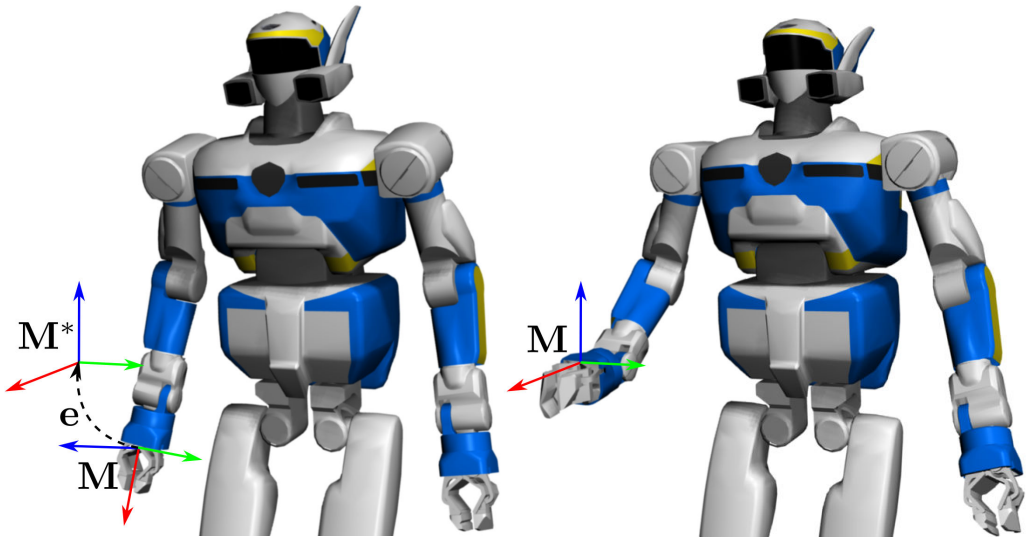


Figure *2.10 – Exemple d'une tâche de position 6D.*

Figure *2.11 – Représentation d'une rotation avec un vecteur.*

2.4.1 Exemples de tâches robotique

2.4.1.1 Tâche position 6D

Un repère qui est attaché à un corps du robot doit atteindre une position 6D désirée. L'erreur correspond à la transformation homogène $\mathbf{M}_e = \mathbf{M}(\mathbf{M}^*)^{-1}$ permettant de passer de la position actuelle à la position désirée avec $\mathbf{M} = \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & \mathbf{1} \end{pmatrix}$. La figure 2.10 illustre ce type de tâche. Cette représentation matricielle est redondante car il suffit de trois variables indépendante pour identifier une matrice de rotation dans le groupe des rotations SO(3). Une représentation vectorielle dans le groupe SE(3) (*Special Euclidian Group* qui est isomorphe à $\mathbb{R}^3 \times$SO(3)) est donc préférée. Le vecteur erreur $\mathbf{e}$ est défini par :

$$\mathbf{e} = \begin{pmatrix} \mathbf{t}_e \\ \mathbf{u}_{\theta,e} \end{pmatrix} \tag{2.32}$$

où $\mathbf{u}_\theta$ est le vecteur représentant une rotation équivalente à une matrice de rotation $\mathbf{R}$: sa direction correspond à l'axe de rotation et sa norme $\|\mathbf{u}_\theta\|$ correspond à la valeurs de l'angle de rotation par rapport à son axe (voir figure 2.11). On note que pour un suivi de trajectoire, l'erreur $\mathbf{e}(t) = \begin{pmatrix} \mathbf{t}_e(t) \\ \mathbf{u}_{\theta,e}(t) \end{pmatrix}$ doit être petit.

Dans la suite du document, on notera $\mathbf{e}_{rh}$ la tâche de prise droite, asservissant le point de contrôle

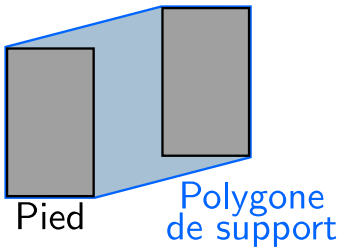


Figure *2.12 – Le polygone de support est défini par les frontières des empreintes de pieds.*

de la dernière articulation du poignet (telle que donnée dans le modèle classique du robot), voir figure 2.10. Cette tâche met en œuvre les chaînes articulaires du *free flyer*, du torse et du bras droit. Symétriquement, la tâche de prise main gauche sera noté $mbfe_{lh}$. On nommera les tâches de position 6D *prise orientée*.

2.4.1.2 Tâche de position 3D et de rotation 3D

Il est possible de ne sélectionner qu'une partie des dimensions d'une tâche de position 6D en supprimant les lignes de la jacobienne correspondant aux dimensions que l'on ne veut pas considérer. On peut ainsi définir des tâches de positions dans n'importe quel sous-espace de dimensions inférieures. Par convention dans la suite du manuscrit, sauf mention contraire, lorsque l'on parle de tâche de position 3D, nous ne considérons que les composantes de translations :

$$\mathbf{e} = \mathbf{t}_e \tag{2.33}$$

De même, lorsque l'on parle de tâche de rotation 3D, nous ne considérons que les composantes de rotations :

$$\mathbf{e} = \mathbf{u}_{\theta,e} \tag{2.34}$$

Dans la suite du document, les tâches nommées *prise* sont des tâches de position en 3D.

2.4.1.3 Tâche de position du centre de masse

L'équilibre statique d'un robot humanoïde est vérifiée lorsque la projection au sol du centre de masse (com) est à l'intérieur du polygone de support du robot (voir figure 2.12). Il est donc intéressant de contrôler la position du centre de masse. Sa position est calculée en fonction de la position du centre de masse de chacun des corps composant le robot :

$$\mathbf{p}_{\mathsf{com}} = \frac{1}{\sum_i m_i} \sum_i m_i \mathbf{p}_{\mathsf{com}_i} \tag{2.35}$$

La tâche de position du centre de masse correspond à un cas particulier de la tâche de position 3D : les colonnes de la jacobienne associée à cette tâche sont pondérées par les masses des corps du robot [Boulic *et al.*, 1996, Sugihara et Nakamura, 2002]. L'erreur est donc exprimé par :

$$e = \mathbf{p}_{com} - \mathbf{p}^*_{com} \tag{2.36}$$

2.4.1.4 Tâche regard

Cette tâche consiste à centrer un point dans le plan image d'une caméra situé au niveau de la tête du robot (voir figure 2.13). Nous décrivons brièvement le principe général dans ce paragraphe. Une description complète, avec notamment le calcul de la jacobienne est présentée dans [Espiau *et al.*, 1992,

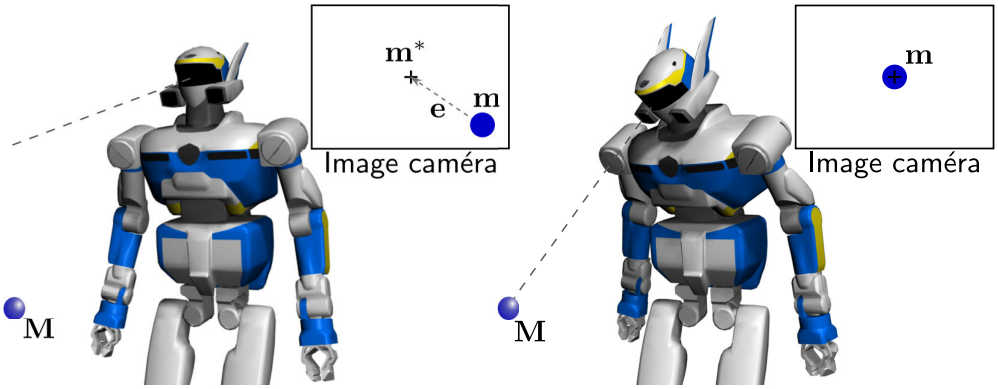


Figure *2.13 – La tâche de regard centre un point* $\mathbf{p}^*$ *de l'espace dans le plan image de la caméra situé au niveau de la tête du robot.*

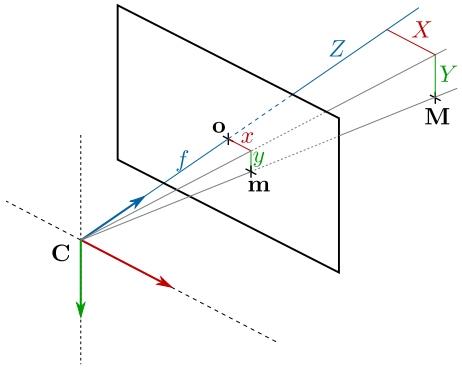


Figure *2.14 – Un point* $\mathbf{M}$ *dans l'espace se projète en un point* m *dans l'image de la caméra.*

Chaumette et Hutchinson, 2006, Chaumette et Hutchinson, 2007]. Un point $\mathbf{M} = \begin{pmatrix} X \\ Y \\ Z \end{pmatrix}$ dans l'espace est projeté dans l'image d'une caméra, ayant un centre optique $\mathbf{C}$ et une distance focale f, en un point m (voir figure 2.14). La position du point $m = \begin{pmatrix} x \\ y \end{pmatrix}$ est alors obtenue par la relation :

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} Xf/Z \\ Yf/Z \end{pmatrix} \tag{2.37}$$

Ainsi le vecteur erreur est défini par :

$$\mathbf{e} = \begin{pmatrix} x_e \\ y_e \end{pmatrix} = \begin{pmatrix} x - x^* \\ y - y^* \end{pmatrix} \tag{2.38}$$

Un exemple d'utilisation de cette tâche est présenté dans [Mansard *et al.*, 2007].

2.4.1.5 Tâche double support

En supposant que les deux pieds du robot sont posés au sol, la tâche de double support consiste à conserver une transformation géométrique constante entre le repère attaché au pied gauche, et le repère attaché au pied droit. Ainsi les deux pieds resteront en contact avec le sol (voir figure 2.15).

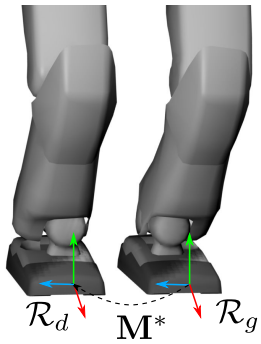


Figure *2.15 – La tâche double support permet de garder une transformation constante égale à* $\mathbf{M}^*$ *entre les repères attachés aux deux pieds du robot* $\mathcal{R}_d$ *et* $\mathcal{R}_g$.

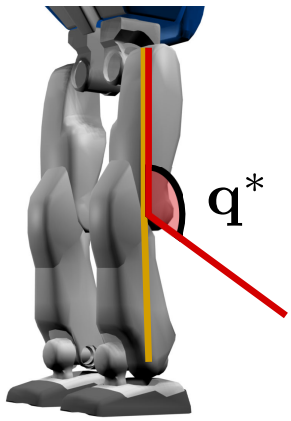


Figure *2.16 – Un exemple de tâche de posture, pour l'articulation d'une jambe.* $\mathbf{q}^*$ *étant la configuration désirée.*

Il s'agit en fait d'un autre cas particulier de la tâche de position 6D. La transformation entre deux repères attachés à deux corps du robot doit rester constante :

$$\mathbf{e} = \begin{pmatrix} {}^r\mathbf{t}_l \\ {}^r\mathbf{u}_{\theta,l} \end{pmatrix} \tag{2.39}$$

avec ${}^r\mathbf{t}_l$ la translation entre un repère attaché au pied droit et un repère attaché au pied gauche et ${}^r\mathbf{u}_{\theta,l}$ la rotation entre un repère attaché au pied droit et un repère attaché au pied gauche.

2.4.1.6 Tâche de posture

Une tâche de posture permet de définir un état référence dans l'espace de configuration :

$$\mathbf{e} = \mathbf{q} - \mathbf{q}^* \tag{2.40}$$

La figure 2.16 illustre une tâche de posture.

La correction de l'erreur agit directement sur $\mathbf{q}$ et la jacobienne est une matrice identité. Le contrôle de la posture pour un robot humanoïde est étudié en détail dans le chapitre 4 de [Sentis, 2007]. Les tâches de posture sont utilisées pour optimiser des critères de performances (comme les efforts des actionneurs) et de reproduire des comportements humain.

Main gauche
Main droite
Regard
Centre de masse
Double support

Figure *2.17 – La pile de tâches utilisé pour illustrer le couplage.*

2.4.2 Couplage des tâches

Grâce au formalisme de la fonction de tâche et de la représentation du mouvement sous forme de pile de tâches il est possible de combiner de manière efficace des tâches indépendantes à réaliser pour générer un mouvement pour un robot. Un couplage entre des tâches apparait si pour les réaliser il est nécessaire d'utiliser des sous-chaînes articulaires communes. En considérant la pile de tâches illustrée dans la figure 2.17, les tâches de regard, et de position de mains gauche et droite ont des influences mutuelles.

La figure 2.18 illustre le mouvement complet issu de cette pile de tâches. Pour illustrer les couplages, les différentes tâches de la piles sont isolées pour générer plusieurs mouvements.

La pile de tâches de base comporte les tâches de double support et de positionnement du centre de masse. La tâche de regard est ajoutée. Pour exécuter cette dernière tâches, le robot utilise son torse pour se pencher en avant. Les bras du robot sont donc entraînés par le torse. Un mouvement au niveau notamment de la main droite peut être observé alors qu'aucune tâche ne contrôle explicitement le bras droit (voir figure 2.19). En ajoutant la tâche de position de la main droite avec une position de référence à l'avant du robot, le mouvement de la main droite est naturellement modifié. Cependant, le mouvement de la main gauche est aussi modifié (figure 2.20). Le déplacement de la main gauche est causé par le couplage de la tâche main droite et la tâche de positionnement du centre de masse en effet, lorsque la main droite se déplace vers l'avant, le centre de masse se déplace aussi dans cette direction. Pour compenser le déplacement du centre de masse causé par le déplacement de la main droite, il est nécessaire de déplacer la main gauche. Le mouvement du torse est également modifié par rapport au mouvement ne faisant intervenir que le regard.

Conclusion

Dans ce chapitre nous avons présenté le formalisme sur lequel se base la suite des travaux présentés dans ce manuscrit : le formalisme de la fonction de tâches. Les méthodes utilisées pour la résolution de la cinématique inverse sont utilisés d'une manière analogue pour manipuler les fonctions de tâches. Nous avons présenté une méthode pour construire efficacement des lois de commandes, qui permettent de réaliser plusieurs tâches parallèles en respectant un ordre de priorité, en les combinant sous forme de pile de tâches. Les tâches de priorités inférieures sont réalisées aux mieux en exploitant l'espace nul des tâches plus prioritaires afin de ne pas perturber leurs réalisations. Chaque tâche est exprimée dans l'espace le plus adaptée à celle-ci. Cette caractéristique permet d'avoir une grande expressivité pour le contrôle d'un robot. Une tâche peut être définie dans un espace identique aux données issues d'un capteur. Ce qui permet par exemple de suivre facilement l'évolution de l'exécution de la tâche. Dans le chapitre suivant, nous présentons une application directe de la structure de la pile de tâches pour effectuer une imitation de mouvements non structurés sans reconnaissance.

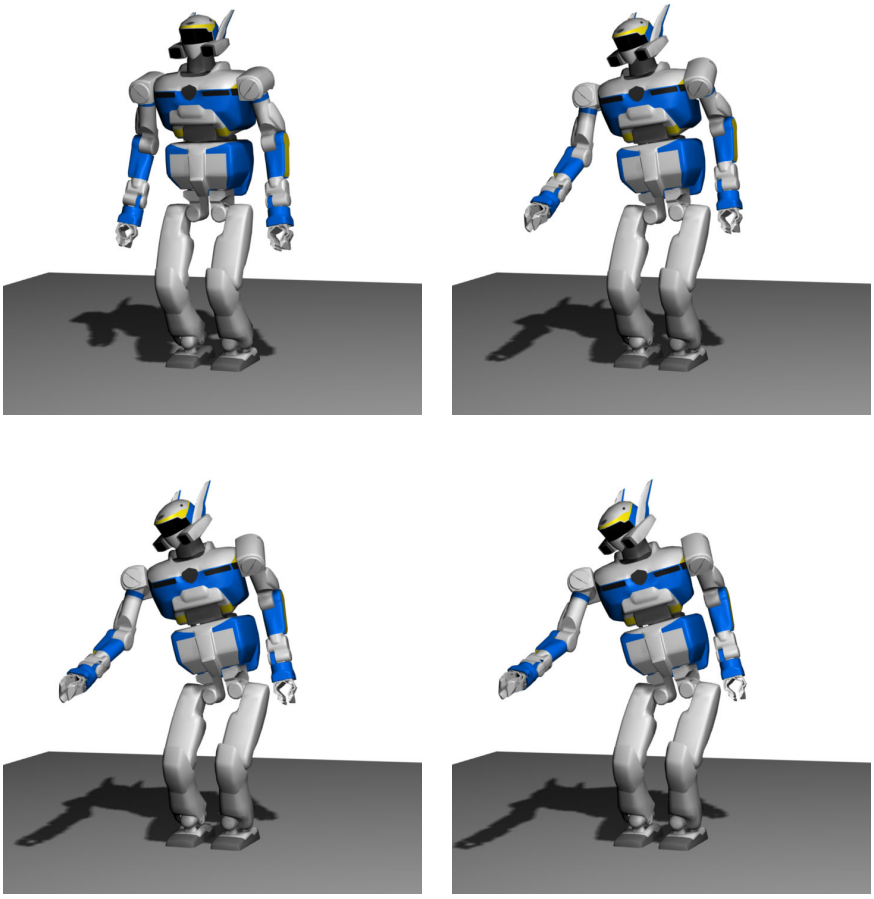


Figure *2.18 – Un exemple de mouvement comportant des tâches couplées. Les tâches exécutées sont la tâche de double support, le positionnement du centre de masse, l'orientation du regard, le positionnement de la main droite et gauche.*

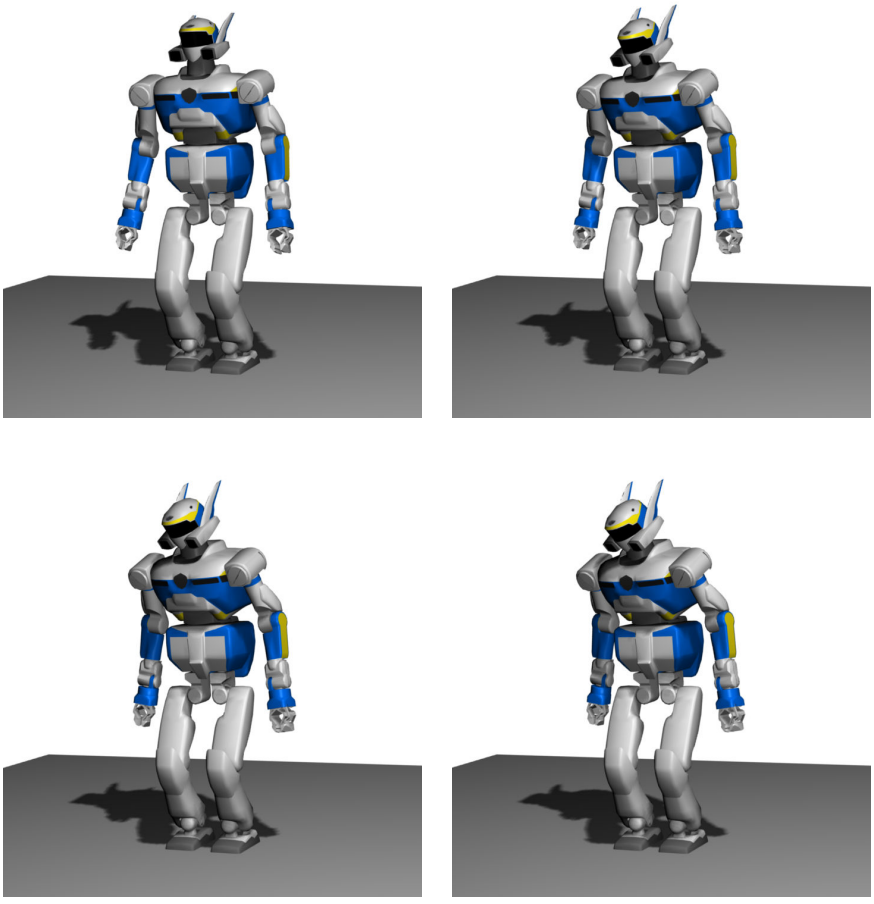


Figure *2.19 – Le mouvement est généré à partir des tâches de double support, de positionnement du centre de masse et de la tâche de regard.*

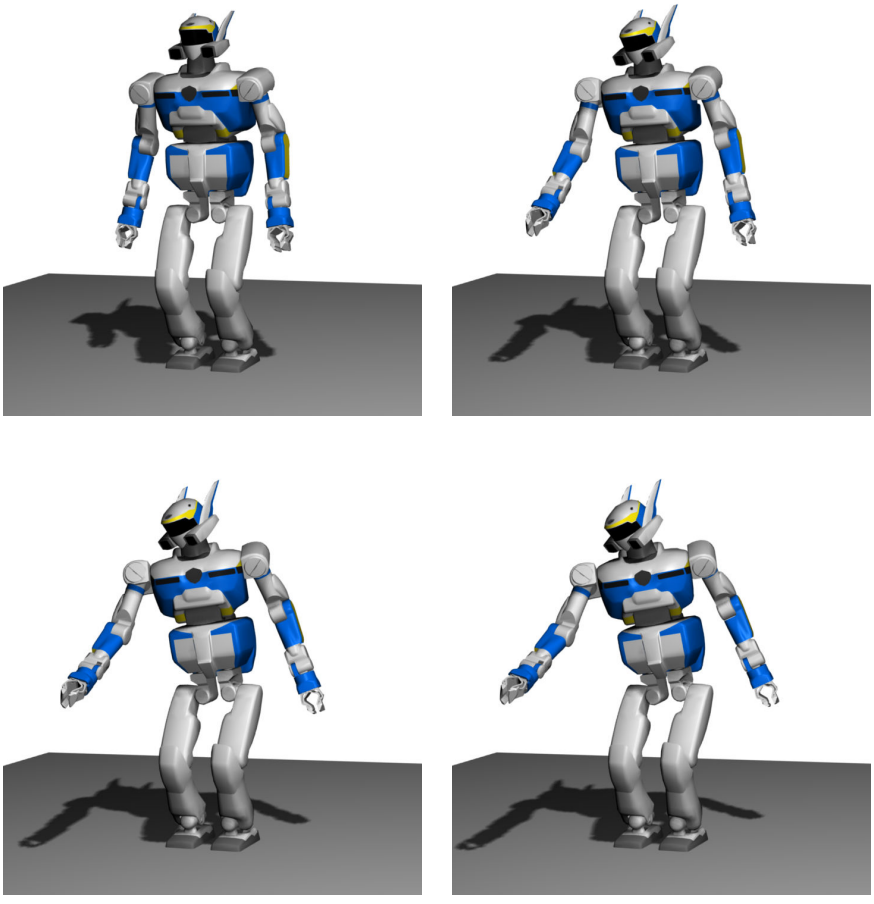


Figure *2.20 – La tâche de positionnement de la main droite perturbe la position du torse et de la main gauche.*

Chapitre 3

Imitation sans reconnaissance

L'imitation sans reconnaissance consiste à exécuter un mouvement en utilisant directement des données issues d'une démonstration. Aucune reconnaissance de ce qui doit être reproduit n'est effectuée. Pour obtenir les données à partir d'une démonstration, les techniques de capture de mouvements sont utilisées pour encoder un mouvement dans un format adapté aux besoins. Dans ce chapitre, nous présentons les principales technologies utilisées pour effectuer de la capture de mouvements, puis nous présenterons un peu plus en détail le système utilisé dans le cadre de nos travaux ainsi que les différentes étapes qui constituent l'utilisation de la capture de mouvements : calibration du système, définition de la structure à capturer, interprétation et phase de post-traitements des données acquises. Ensuite nous appliquons le principe de la pile de tâches, présenté dans le chapitre précédent, pour reproduire un mouvement difficilement reconnaissable car n'obéissant pas à une loi particulière. Nous prenons l'exemple du mouvement d'attente d'un humain, pour le reproduire sur un robot HRP-2. Ce type de mouvement peut se traduire par la réalisation de mouvements inconscients de l'humain lorsqu'il exécute l'action *ne rien faire*.

3.1 Capture de mouvements

La capture de mouvements consiste à enregistrer un mouvement indépendamment de l'aspect visuel, afin de l'encoder dans un format adapté aux besoins. Par exemple le format pourra faciliter une analyse ou une édition de mouvements. Le mouvement enregistré peut être issu de n'importe quel objet ou individu réel. Ainsi, les données acquises peuvent avoir des applications très variées. Les paragraphes suivants présentent quelques exemples d'applications de la capture de mouvements, puis les difficultés générales rencontrées. Ensuite nous présentons les différentes technologies développées pour effectuer de la capture de mouvement et détaillons le processus de capture pour le système disponible dans ces travaux.

3.1.1 Applications

La capture de mouvements est tout d'abord utilisée pour analyser des mouvements. Dans le domaine biomédical pour l'analyse de mouvements de marche par exemple ou pour l'analyse de performances sportives. La visualisation 3D de la performance de l'athlète lui permet de déceler les points perfectibles de sa technique plus facilement qu'avec une simple vidéo grâce à la possibilité de changement d'angle de vue. La capture de mouvements peut aussi servir de mécanisme d'entrée pour

l'interaction homme-machine. Un système de capture de mouvements peut aussi être utilisé comme un outil de localisation rapide et précise d'objets ou d'obstacles dans l'environnement dans le cadre d'une planification ou d'une replanification [Baudouin *et al.*, 2011, Stilman *et al.*, 2005].

Mais l'application la plus populaire des techniques de capture de mouvement est l'animation graphique dans le cinéma ou les jeux vidéo afin de doter des personnages virtuels de mouvements visuellement réalistes. En effet, il est très difficile de créer des mouvements réalistes et précis capable de traduire des caractéristiques subtiles qu'une personne produit. La technique d'animation manuelle (sans capture) la plus directe consiste à définir les positions et configurations d'objets à chaque instant de temps. Il est possible de réduire la quantité de travail de cette technique grâce à des outils informatique. Des configurations clefs sont définies, puis le mouvement entre deux configurations successives est calculé par interpolation [Burtnyk et Wein, 1976, Kovar et Gleicher, 2003]. Cependant, ces techniques nécessitent un animateur doté d'une grande expérience pour identifier et reproduire de façon convaincante les propriétés des mouvements. L'importance de l'expérience dans le domaine de l'animation est soulignée dans [Lasseter, 1987] où une analogie entre les principes d'animation traditionnelle dessinée à la main et l'animation 3D assistée par ordinateur est présentée. Le travail d'animation reste dans tous les cas considérable. C'est pour cette raison qu'il peut être intéressant de transférer des mouvements réels vers l'objet ou le personnage à animer en utilisant de la capture de mouvements. L'avantage direct est de pouvoir être capable de reproduire beaucoup de mouvements sans chercher à trouver un modèle mathématique permettant de générer un mouvement particulier et obtenir ainsi une grande expressivité théorique de mouvements.

3.1.2 Difficultés générales de la technique

Le transfert de mouvements n'est pas direct à cause, entres autres, des imperfections des systèmes de capture de mouvements. Les sessions de capture de mouvements complexes sont souvent suivies d'étapes de post-traitements. Les données collectées peuvent être bruitées de manière sensible ou présenter des discontinuités en fonction des technologies utilisées. Par conséquent, ces données doivent être nettoyées si nécessaire. Ensuite, ces mouvements peuvent être édités en considérant que le mouvement est un signal [Bruderlin et Williams, 1995]. L'édition de mouvement peut être motivée par plusieurs raisons [Gleicher, 2000]. Si on veut réutiliser un mouvement déjà enregistré en y apportant des petits changements comme adapter le mouvement à un personnage possédant des caractéristiques physiques différentes. Il s'agit du problème connu sous le nom de *motion retargeting* [Gleicher, 1998]. Un des problèmes récurrent du *motion retargeting* est que le mouvement recallé perd certaines propriétés. Notamment, lors des mouvements de marceh, les pieds du personnage perdent le contact avec le sol, ou glissent sur celui-ci. Une autre étape d'édition est donc nécessaire pour obtenir un mouvement cohérent. D'autres changements peuvent être apportés pour varier des mouvements dupliqués pour animer une foule par exemple. L'édition peut aussi être motivée par la volonté de créer des mouvements physiquement impossible ou corriger des imperfections issues de la prestation de l'acteur. Une autre étape délicate est l'association des données éditées aux éléments virtuels à animer. Cette dernière étape peut être très complexe. Par exemple l'animation de visage est considéré comme un problème à part entière et des techniques de réductions de dimensions sont même utilisées [Deng *et al.*, 2006].

Malheureusement, dans la plupart des cas, l'édition est un processus lourd de par la nature de la capture de mouvements. Les données correspondent aux configurations (ou postures) des éléments enregistrés, par conséquent, l'édition portera sur un important volume de données. Le volume de données cache souvent les problèmes liés à des erreurs de capteurs. Par exemple des erreurs de suivi ou des discontinuités trop importantes. Ainsi, la correction des erreurs nécessite un temps non négligeable de post-traitements. De plus ces données ne comportent aucune sémantique. Il s'agit de données bas niveau, et donc il n'y a pas directement d'informations sur les propriétés importantes du mouvement ni sur les motivations du mouvement. Il faut donc beaucoup de données pour avoir une grande couverture de type de mouvements (mouvement exécuté par un personnage avec un certain degré de fatigue, de blessures, de joie, à différentes vitesses...), et un problème d'explosion combinatoire peut ainsi apparaître [Gleicher, 2008].

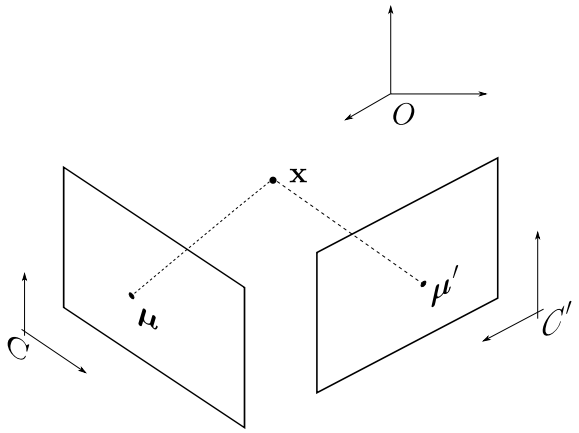


Figure *3.1 – Le point* $\mathbf{x}$ *se projète dans le plan image des caméras* C *et* C' *respectivement en* $\boldsymbol{\mu}$ *et* $\boldsymbol{\mu}'$.

3.1.3 Systèmes optiques avec marqueurs

Plusieurs caméras sont utilisées simultanément pour acquérir la position 3D de marqueurs spécifiques en émettant des rayons infrarouge se réfléchissant sur les marqueurs. Les marqueurs se présentent sous la forme de petites billes réfléchissantes légères, ainsi l'acteur n'est pas beaucoup gêné par le port de marqueurs. Le nombre de caméras nécessaire dépend du volume de capture souhaité et de la puissance de calcul disponible pour traiter les données des caméras. Les systèmes optiques nécessitent une calibration : les caméras suivent des objets de dimensions connues que le logiciel maître reconnait. Puis en combinant les informations des caméras, la position des caméras dans l'espace est calculée. Pour estimer la position des marqueurs dans l'espace, les images provenant des caméras sont filtrées afin d'en extraire les positions dans l'image de la caméra des marqueurs visibles. Ces images filtrées sont utilisées pour effectuer une correspondance spatiale des marqueurs (voir la figure 3.1). Un point dans l'espace 3D se projette en un point dans le plan image 2D d'une caméra C selon l'équation :

$$\boldsymbol{\mu} = \mathbf{P}\mathbf{x} \tag{3.1}$$

où $\boldsymbol{\mu}$ est le vecteur de dimension 3 des coordonnées homogènes du point dans l'image de la caméra, $\mathbf{P}$ est la matrice de projection de dimension 3×4 de la caméra et $\mathbf{x}$ le vecteur de dimension 4 des coordonnées homogènes du point dans l'espace cartésien. La matrice $\mathbf{P}$ est calculée lors de la phase de calibration de la caméra et traduit la projection, la mise à l'échelle de l'image, la translation à l'origine (dans le plan image) et la position et orientation de la caméra dans le repère du monde.

Cette équation de projection peut s'écrire à l'aide d'un produit vectoriel :

$$\boldsymbol{\mu} \times \mathbf{P}\mathbf{x} = \begin{pmatrix} \mu_1 \\ \mu_2 \\ 1 \end{pmatrix} \times \begin{pmatrix} \mathbf{P}_1^T \\ \mathbf{P}_2^T \\ \mathbf{P}_3^T \end{pmatrix} \mathbf{x} = \mathbf{0} \tag{3.2}$$

Le développement conduit à deux équations linéairement indépendantes :

$$(\mu_1 \mathbf{P}_3^T - \mathbf{P}_1^T)\mathbf{x} = 0 \tag{3.3}$$

$$(\mu_2 \mathbf{P}_3^T - \mathbf{P}_2^T)\mathbf{x} = 0 \tag{3.4}$$

$$\tag{3.5}$$

Les équations associées à chacune des caméras sont regroupées pour former un système $\mathbf{A}\mathbf{x} = \mathbf{0}$ dont la résolution sur $\mathbf{x}$ par les moindres carrés (à cause du bruit) fournit la position 3D du point observé.

Les données produites par la capture contiennent les positions 3D dans l'espace Cartésien de chacun des marqueurs utilisés. Cependant, bien que la position des marqueurs soit connue, le système n'a aucun autre indice que la continuité des positions pour les identifier, contrairement aux systèmes électromagnétiques par exemple dans lesquels chaque émetteur possède sa propre signature. Ceci peut poser problème en cas d'occlusion d'un marqueur qui pourrait être perdu dans la suite du mouvement. Des solutions matérielles peuvent être utilisées pour surmonter ce problème, comme l'utilisation de marqueurs actifs (avec des diodes électroluminescentes par exemple) pour distinguer les marqueurs. Cependant ces marqueurs sont plus encombrant, souvent câblés et nécessite des dispositifs d'alimentations embarqués.

Les données obtenues sont précises, il est relativement aisé de changer la configuration des marqueurs en cas de besoin (occlusions, changement d'acteur ayant une taille différente). La liberté des mouvements est importante par l'absence de câbles. L'espace de capture peut être grand (en fonction du nombre de caméra). Il est également possible de construire un squelette virtuel calculé à partir des points 3D correspondant aux marqueurs [Silaghi *et al.*, 1998]. Les données issues des transformations géométriques des os du squelette peuvent ensuite être exportées vers des logiciels d'animation graphique.

En revanche, l'environnement dans lequel se déroule la capture doit être contrôlé : la lumière du soleil, les objets réfléchissants apportent beaucoup de bruits. De plus la calibration doit être fréquente si les caméras ne sont pas fixe (utilisation de trépieds).

3.1.4 Systèmes optiques sans marqueurs

Il est également possible d'enregistrer des mouvements sous formes de données de positions avec de simples caméras via une phase de post-traitement sur le ou les flux vidéos. La précision de ces techniques est faible, en effet, il peut être difficile d'estimer les rotations d'un membre humain à cause de trop importantes ambiguïtés provenant de l'image ou du manque de points caractéristiques. L'étude présenté dans [Gleicher et Ferrier, 2002] souligne la nécessité d'utiliser des hypothèses supplémentaire sur ce qui est observé (dynamique des mouvements, limitations géométrique du monde...). Par exemple, une technique permettant de générer des mouvements physiquement réaliste à partir d'une séquence vidéo monoculaire est présentée dans [Wei et Chai, 2010]. L'utilisation d'un modèle physique contraint la reconstruction du mouvement aux lois de la physique. Mais elle permet aussi de calculer les couples et les forces de contacts correspondant au mouvement observé. Des informations de profondeurs, en plus des images 2D peuvent aussi être utilisées pour la reconnaissance de pose humaine [Shotton *et al.*, 2011].

3.1.5 Dispositifs de suivi électromagnétique

Pour ces systèmes, la mesure des distances et des orientations se base sur les champs magnétiques créés par un émetteur et captés par un recepteur [Raab *et al.*, 1979]. La figure 3.2 montre un exemple de placement de capteurs électromagnétiques sur une actrice. Lorsqu'un courant est appliqué à un bobinage, un champ magnétique est créé (voir figure 3.3).

Figure *3.2 – Un exemple de placement de capteurs issu d'un système de capture de mouvements électromagnétique (image tirée de [Bodenheimer* et al.*, 1997]).*

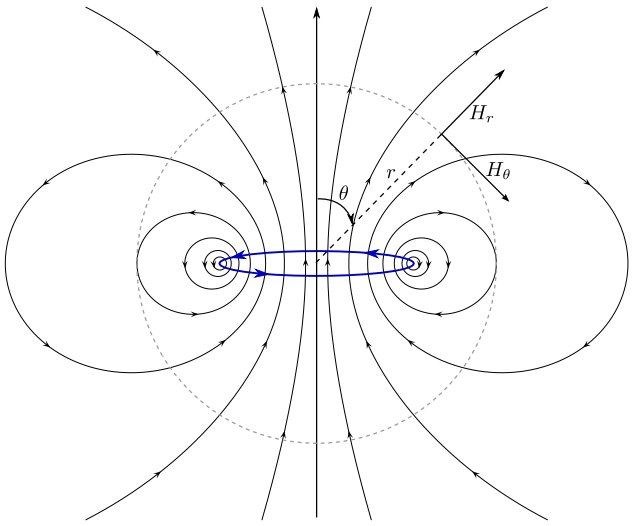


Figure *3.3 – Un courant appliqué à une spire crée un champ magnétique* $\mathbf{H}$*, de composante radiale* H_r*, et tangentielle* H_θ*.*

Les composantes radiale et tangentielle de ce champ magnétique sont décris par :

$$H_r = \frac{M}{2\pi r^3}\cos(\theta) \tag{3.6}$$

$$H_\theta = \frac{M}{4\pi r^3}\sin(\theta) \tag{3.7}$$

où $M = NIS$ est le moment magnétique, N étant le nombre de spires de la bobine, I l'intensité du courant et S la surface entouré par une spire. Le récepteur mesure les champs magnétiques créés par l'émetteur afin de déterminer leurs distances et leurs orientations relatives. Un couple émetteur/récepteur fournit donc des positions 6D. La précision des positions calculées décroit avec l'augmentation de la distance entre l'émetteur et le récepteur. Historiquement, les recepteurs, placés sur l'objet à suivre sont câblés. Ceci rendait l'utilisation de ce type de matériel difficile pour des mouvements amples ou complexes. Ce n'est que récemment que des systèmes sans fils sont apparus [Acht *et al.*, 2007]. Ce type de système est aussi sensible à l'environnement qui ne doit pas comporter d'éléments métalliques car ils causent des distorsions du champ magnétique qui faussent ainsi l'estimation de la position. Cependant, ils présentent l'avantage de ne pas avoir de problème d'occlusions. De plus, il est possible d'estimer les positions de l'épaule, du coude et du poignet ainsi que le vecteur de configuration de dimension sept associé au modèle d'un bras avec seulement deux capteurs 3D en effectuant une phase de calibration [Rezzoug *et al.*, 2010].

3.1.6 Dispositifs de suivi inertiel

Des capteurs inertiels peuvent être utilisés pour effectuer de la capture de mouvements comme par exemple dans [H. Liu et Ha, 2011]. La figure 3.4(a) montre un capteur inertiel. Ils peuvent être composés d'accéléromètres et de gyroscopes. Le suivi d'objets est effectué en déterminant les accélérations et les orientations de ces objets.

Les accélérations sont déterminées en suivant la seconde loi de Newton reliant la résultante des forces $\mathbf{F}$ exercées sur un corps de masse m à son accélération $\mathbf{a}$: $\mathbf{F} = m\mathbf{a}$. Les accéléromètres contiennent une masse connue attachée à un ressort. Lorsqu'une accélération est appliquée à l'accéléromètre, l'inertie va pousser la masse à compresser le ressort. Cette force est ensuite convertie en signal électrique grâce par exemple à des capteurs piézoélectriques (voir figure 3.4(b)). L'accélération est ensuite intégrée deux fois pour déterminer la position. De la même manière, les gyroscopes permettent de déterminer les positions angulaires d'un objet en mesurant les forces centrifuges d'une masse en rotation. Cette fois-ci, la force mesurée est proportionnelle à la vitesse angulaire. Une seule intégration est donc nécessaire pour avoir une mesure de la position angulaire.

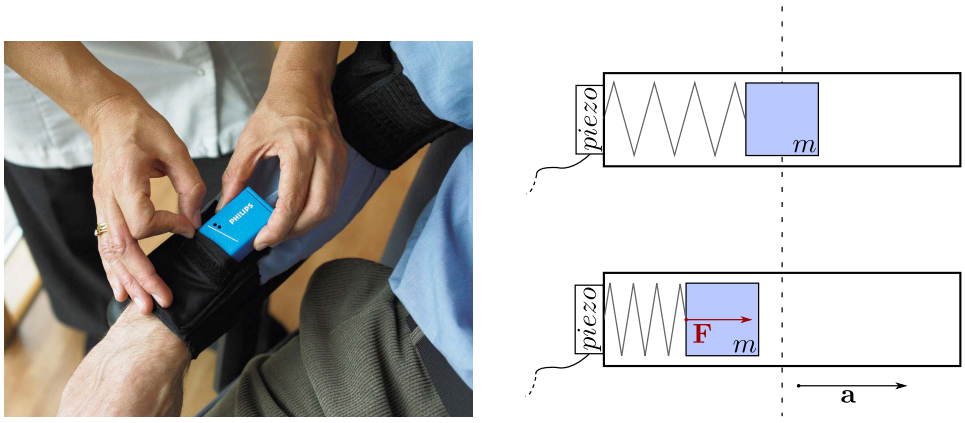
Les capteurs inertiels ont l'avantage d'être relativement facile à mettre en œuvre mais les mesures dérivent. Les erreurs de mesures s'accumulent et par conséquent, l'estimation de la position de l'objet diverge. Le suivi d'objet par capteurs inertiels n'est donc efficace que pendant de courtes périodes.

3.1.7 Systèmes de capture mécanique

Il s'agit de systèmes articulés que l'utilisateur manipule. Les angles des articulations sont directement mesurés grâce à des encodeurs. Ils peuvent prendre plusieurs formes en fonction du type de données considérés, comme par exemple un petit bras articulé pour obtenir une position 6D d'un objet manipulé avec la main, un gant pour obtenir les positions de mains ou de doigts ou encore un exosquelette si l'on s'intéresse aux mouvements complet du corps humain (voir figure 3.5). Ces systèmes sont limités par les contraintes mécaniques de ces systèmes et sont très encombrant.

3.2 Système utilisé

Le LAAS-CNRS est équipé du système de capture de mouvements de la société Motion Analysis. Le système installé est composé de dix caméras infrarouge (voir figure 3.6) : quatre caméras standard



(a) (b)

Figure *3.4 – (a) Un capteur inertiel placé pour mesurer les mouvements d'un bras. Photo tirée de [Acht* et al.*, 2007]. (b) L'accélération est déterminée à partir de la mesure de la force grâce à la relation* $\mathbf{F} = m\mathbf{a}$

(modèle *Hawk* 640x480@200Hz), et six caméras haute résolution (modèle *Eagle* 2352x1728@200Hz). Les données sont capturées à une fréquence de 200Hz. Les caméras sont pilotées par le logiciel nommé Cortex (voir figure 3.7). Les caméras sont réparties autour de la zone d'expérimentation suivant le schéma de la figure 3.8. Afin d'optimiser le placement des caméras, les caméras à haute résolution sont placés pour couvrir la longueur de la zone d'expérimentation, tandis que les caméras à basse résolution sont placées pour couvrir la largeur de la zone. La figure 3.9 illustre les zones de couvertures des caméras.

D'une manière générale, la capture de mouvements se décompose en plusieurs étapes. La qualité du résultat obtenu à chaque étape dépend directement de la précédente (voir la figure 3.10). Un exemple du processus complet de capture de mouvement est présenté dans [Bodenheimer *et al.*, 1997].

3.2.1 Calibration

La calibration permet d'associer les positions dans le monde réel et les positions dans l'image des caméras. C'est lors de cette phase que les paramètres des caméras sont calculés, ces paramètres dépendent des positions et orientations, des focales de l'objectif, des facteurs d'echelle suivant les deux axes de l'image et des translations d'origine de l'image des caméras. Les matrices de projections des caméras dépendent directement de ces paramètres. Dans le cas de notre système, celle-ci se décompose en deux parties.

3.2.1.1 Calibration statique

La calibration statique permet de définir un repère fixe au volume étudié et de déterminer une première estimation des paramètres d'au moins une partie des caméras. Une équerre de précision équipée de quatre marqueurs dont les positions relatives sont parfaitement connues est utilisée (voir la figure 3.11(a)). Les paramètres des caméras qui ont les quatre marqueurs dans leurs champs de vision ont alors une bonne première estimation qui sera raffiné par optimisation dans l'étape suivante. Les autres caméras auront une mauvaise estimation de leurs paramètres, mais ils seront corrigés lors de l'étape suivante.

(a) Phantom de SensAble.

(b) CyberGlove II de CyberGlove Systems.

(c) Gypsy 7 de Animazoo.

Figure *3.5 – (a) Un bras articulé à six degrés de liberté. La position et l'orientation du stylet est obtenue grâce aux mesures des angles articulaire. (b) Le gant articulé permet de mesurer la configuration de la main, mais permet également de fournir des positions 6D de la main. (c) L'exosquelette Gypsy 7 qui permet de mesurer les valeurs articulaires correspondant aux mouvements d'un humain.*

Figure *3.6 – Une caméra infrarouge.*

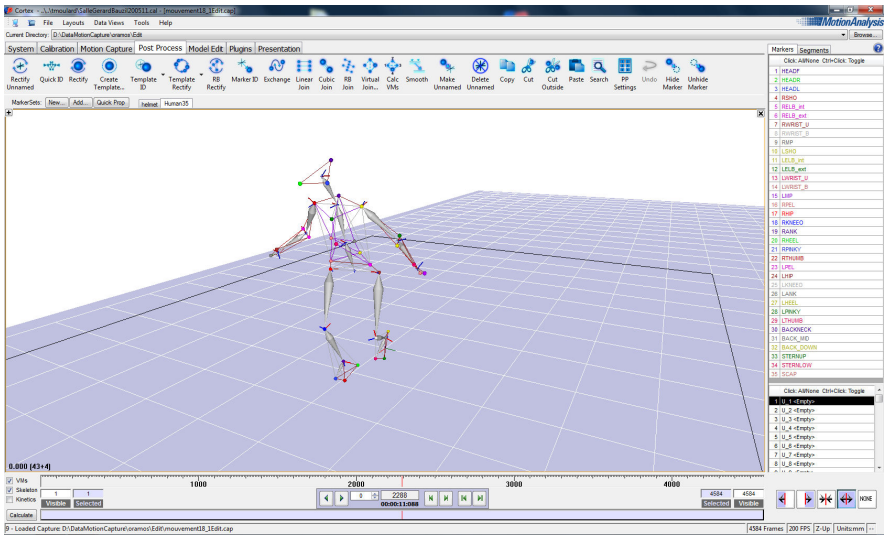


Figure *3.7 – Le logiciel de capture de mouvement utilisé au LAAS-CNRS : Cortex de Motion Analysis.*

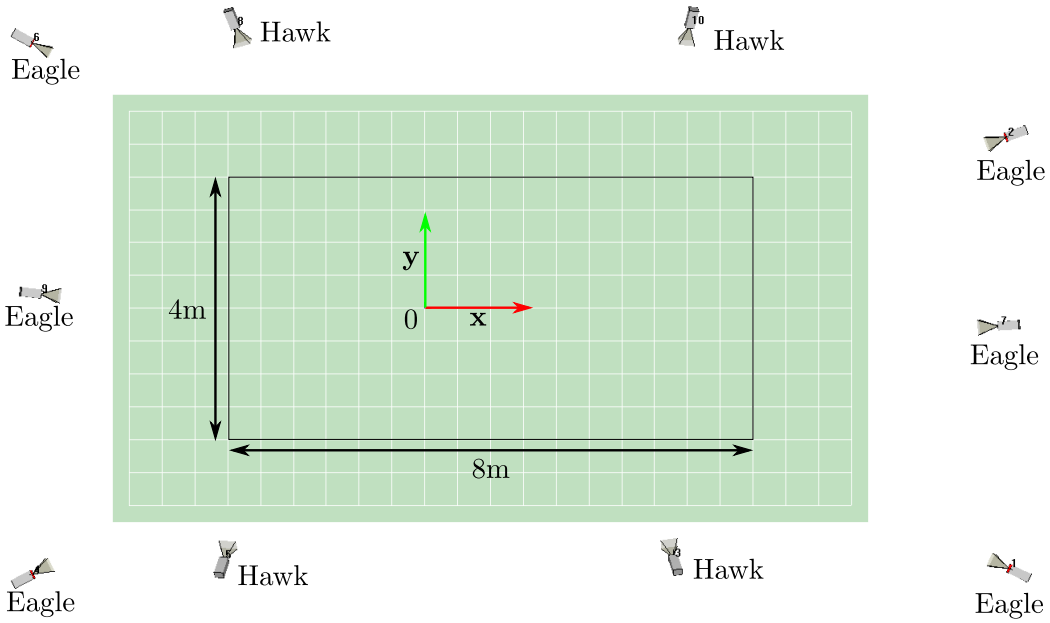


Figure *3.8 – Placement des caméras dans la zone d'expérimentation.*

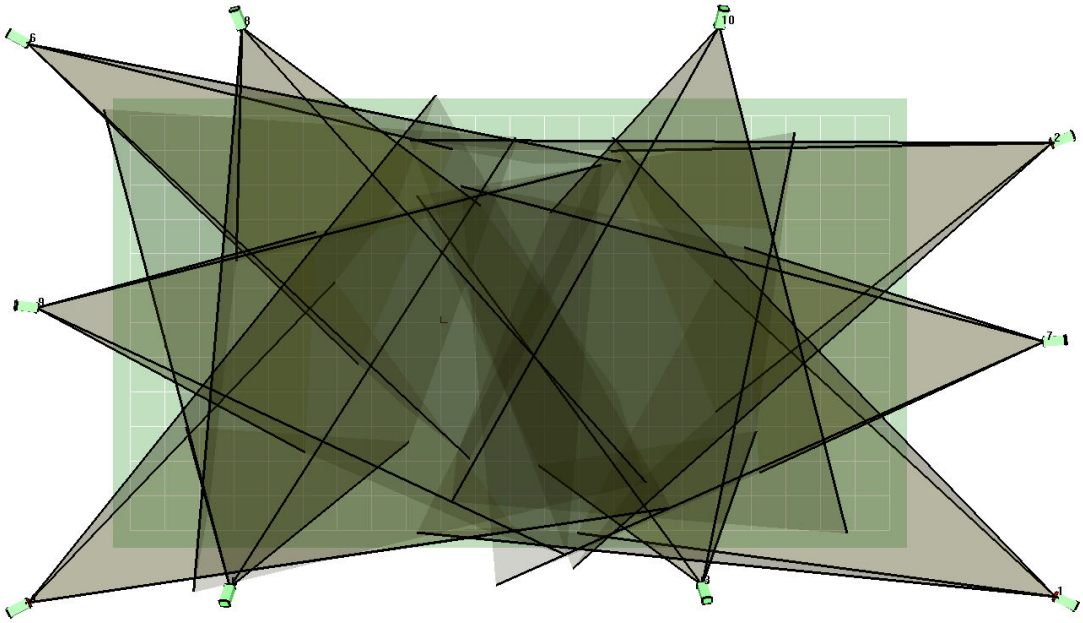


Figure *3.9 – Zone de couverture des caméras.*

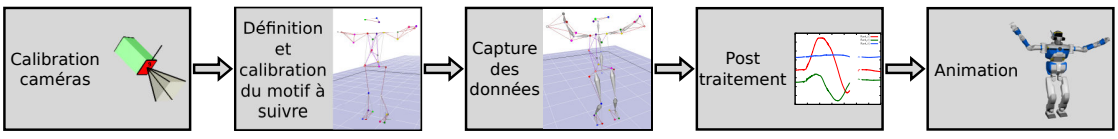
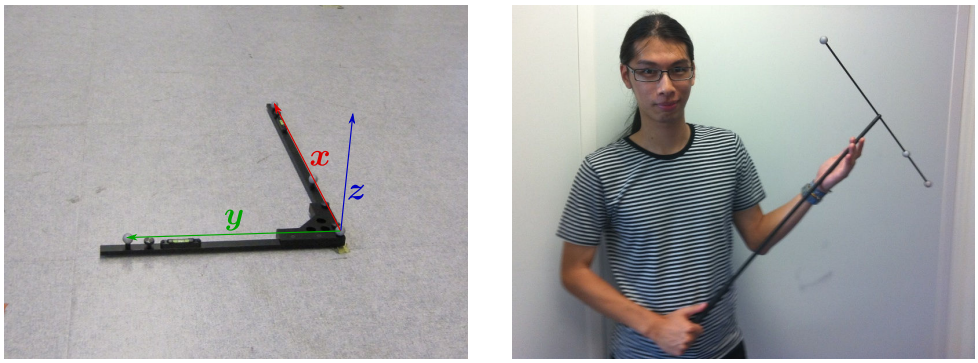


Figure *3.10 – Les différentes étapes pour capturer un mouvement.*



(a) (b)

Figure *3.11 – (a) L'équerre de calibration définit un repère fixe dans lequel les données seront exprimées. (b) La baguette de précision qui est utilisée pour balayer la zone d'expérimentation.*

3.2.1.2 Calibration dynamique

Elle consiste à balayer, de la manière la plus homogène possible, tout l'espace d'expérimentation avec une baguette équipée de trois marqueurs dont les positions relatives sont connues avec précision (voir figure 3.11(b)). Les données collectées sont utilisées pour calculer de manière plus précise les paramètres de toutes les caméras par optimisation. La couverture des données dans l'espace permet d'obtenir une homogénéisation sur la correspondance entre les points dans l'image des caméras et dans l'espace. C'est pour cette raison qu'il est important de balayer l'espace de manière homogène (ne pas se concentrer sur une zone ou en négliger).

3.2.2 Placement des marqueurs

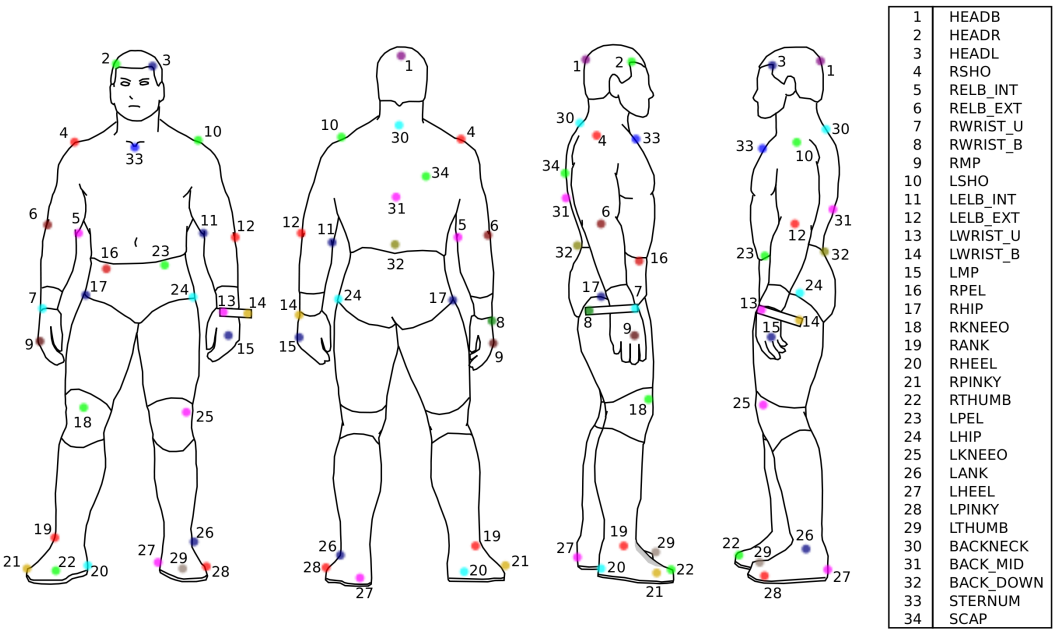
Idéalement pour capturer des mouvements humains, les marqueurs doivent être placés au centre des articulations, ou au plus proche possible pour éviter que lors d'une rotation d'une articulation, le marqueur associé à celle-ci translate ou bouge par rapport à ce centre. La configuration des marqueurs ne doit présenter aucune symétrie afin que le suivi temporel des marqueurs ne soit pas perturbé. En cas de symétrie, des permutations d'identifiant de marqueurs peuvent apparaitre.

Nous donnons un exemple de placement de marqueurs qui peut être modifié selon les besoins dans la figure 3.12. Dans cet exemple, des petites planches sont placées sur les poignets pour les alonger. De cette façon, les trois marqueurs placés sur les mains ne forment pas un triangle trop proche d'un triangle équilatéral, et sont plus facile à suivre par le logiciel de capture de mouvements. Le nombre de marqueurs peut être plus grand pour introduire des données redondantes pour compenser les données des marqueurs dont on sait à l'avance qu'ils vont être être occlus.

3.2.3 Squelette virtuel

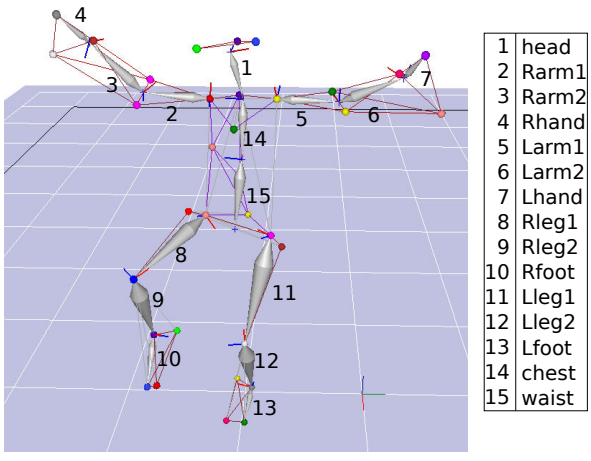
3.2.3.1 Définition du squelette

Le format de données calculées par défaut par le système est la liste des coordonnées 3D dans le repère de la zone de capture de tous les marqueurs. Pour des mouvements humains, il est plus intéressant de travailler en utilisant des corps et leurs transformations dans l'espace. Pour cela, un squelette est construit en hiérarchisant un ensemble de corps sous la forme d'un arbre. Chaque repère associé à un corps est défini par un ensemble de trois marqueurs, définissant l'origine, la direction de l'axe de la longueur et la direction de l'axe plan (voir figure 3.13). La figure 3.13 illustre un exemple de squelette associé à la configuration des marqueurs présentée plus tôt.



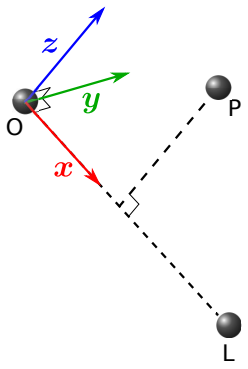
1	HEADB
2	HEADR
3	HEADL
4	RSHO
5	RELB_INT
6	RELB_EXT
7	RWRIST_U
8	RWRIST_B
9	RMP
10	LSHO
11	LELB_INT
12	LELB_EXT
13	LWRIST_U
14	LWRIST_B
15	LMP
16	RPEL
17	RHIP
18	RKNEEO
19	RANK
20	RHEEL
21	RPINKY
22	RTHUMB
23	LPEL
24	LHIP
25	LKNEEO
26	LANK
27	LHEEL
28	LPINKY
29	LTHUMB
30	BACKNECK
31	BACK_MID
32	BACK_DOWN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

Figure *3.12 – Exemple de configuration de marqueurs pour capturer des mouvements humains. Des petites planches sont utilisées pour allonger les poignets.*



1	head
2	Rarm1
3	Rarm2
4	Rhand
5	Larm1
6	Larm2
7	Lhand
8	Rleg1
9	Rleg2
10	Rfoot
11	Lleg1
12	Lleg2
13	Lfoot
14	chest
15	waist

(a)



(b)

Figure *3.13 – (a) Un squelette virtuel pour un humain. (b) Définition du repère associé à un corps à partir de trois marqueurs.*

3.2.3.2 Interprétation des données

Les données relatives au squelette sont exportées au format développé par Motion Analysis appelé HTR (hierarchical translation-rotation). Le fichier de sortie se décompose en quatre parties :

- *Header* : Informations générales.
- *Segment names & Hierarchy* : Graphe de hiérarchie des corps du squelette, avec pour nœud racine *GLOBAL*.
- *Base position* : Composantes de transformation initiale des corps relativement à son parent, qu'on note dans la suite $\mathbf{T}_{j,0}$ et $\mathbf{R}_{j,0}$.
- *Frame data* : Composantes de transformation à l'instant $Fr = i$ qu'on note $\mathbf{T}_j(i)$, $\mathbf{R}_j(i)$ et $\mathbf{S}_j(i)$.

La figure 3.14 illustre un extrait d'un fichier *htr*. La transformation $\mathbf{M}_j$ appliquée à un corps j par rapport à son parent à l'instant $Fr = i$ se calcule grâce à l'équation :

$$\mathbf{M}_j(i) = \mathbf{T}_{j,0}\mathbf{T}_j(i)\mathbf{R}_{j,0}\mathbf{R}_j(i)\mathbf{S}_j(i) \tag{3.8}$$

où $\mathbf{T}_j(i)$ représente la composition des transformations de translation $[tx_{j,0}\ ty_{j,0}\ tz_{j,0}]^T$ issues de la partie *Base position* et $[tx_j(i)\ ty_j(i)\ tz_j(i)]^T$ issues de la partie *Frame data*.

$$\mathbf{T}_{j,0} = \begin{pmatrix} 1 & 0 & 0 & tx_{j,0} \\ 0 & 1 & 0 & ty_{j,0} \\ 0 & 0 & 1 & tz_{j,0} \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.9}$$

$$\mathbf{T}_j(i) = \begin{pmatrix} 1 & 0 & 0 & tx_j(i) \\ 0 & 1 & 0 & ty_j(i) \\ 0 & 0 & 1 & tz_j(i) \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.10}$$

$\mathbf{R}_{j,0}$ est la composition de rotation calculée à partir des angles de la partie *Base position* selon l'ordre des angles d'Euler défini dans *Header*. Dans l'exemple choisi, l'ordre est ZYX donc $\mathbf{R}_{j,0} = \mathbf{R}_z\mathbf{R}_y\mathbf{R}_x$. De même $\mathbf{R}_j(i)$ est la composition de rotation à partir des données de la partie *Frame data*. Enfin $\mathbf{S}_j(i)$ est la matrice de mise à l'échelle :

$$\mathbf{S}_j(i) = \begin{pmatrix} SF_j(i) & 0 & 0 & 0 \\ 0 & SF_j(i) & 0 & 0 \\ 0 & 0 & SF_j(i) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.11}$$

où $SF_j(i)$ est le facteur d'échelle à l'instant $Fr = i$.

La transformation ${}^W\mathbf{M}_n$ dans le repère de la zone de capture (repère défini par l'équerre lors de la calibration) d'un corps n ayant comme parent un corps $n-1$ est obtenu grâce à l'équation :

$${}^W\mathbf{M}_n = \prod_{j=0}^{n} \mathbf{M}_j \tag{3.12}$$

```
# Hierarchical Translation and Rotation (.htr) file
# Generated by Cortex
[Header]
NumSegments 15
NumFrames 2648
DataFrameRate 200
EulerRotationOrder ZYX
CalibrationUnits mm
RotationUnits Degrees
[SegmentNames&Hierarchy]
#CHILD          PARENT
head            chest
Rarm1           chest
Rarm2           Rarm1
Rhand           Rarm2
Larm1           chest
Larm2           Larm1
Lhand           Larm2
chest           waist
waist           GLOBAL
[BasePosition]
#SegmentName     Tx   Ty   Tz   Rx   Ry   Rz   BoneLength
head            -20.152159 -202.713951 8.469059 -165.286789 80.587222 -171.153685 203.889140
Rarm1            25.661655 3.212979 157.173351 170.223032 -46.359516 168.283198 260.516899
Rarm2            0.000000 260.516899 0.000000 29.615291 32.173430 -67.932513 212.477333
Rhand            0.000001 212.477339 0.000002 28.352298 -26.382163 -20.534953 138.299515
Larm1           -25.661687 -3.213099 -157.173409 -168.288575 10.883513 171.092845 257.182587
Larm2            0.000000 257.182577 0.000000 -13.941330 -13.266533 -68.705961 214.109438
Lhand           -0.000001 214.109433 0.000001 -19.755832 -20.006212 -31.958093 137.284042
chest           12.985547 -206.958014 13.393044 -3.698000 -2.139499 3.728555 207.797058
waist           112.969803 -623.773804 1014.785828 -86.497264 3.754898 -167.585379 262.421536
#Beginning of Data.
[head]
#Fr Tx Ty Tz Rx Ry Rz SF
1 0.000000 0.000000 0.000000 -0.000000 0.000000 0.000000 1.000000
2 -0.040073 0.017102 0.084137 0.015299 0.004087 -0.021626 0.999953
[Rarm1]
#Fr Tx Ty Tz Rx Ry Rz SF
1 0.000000 0.000000 0.000000 -0.000000 -0.000000 0.000000 1.000000
2 -0.002744 -0.066199 -0.043759 0.007722 -0.018471 -0.051833 1.000110
[Rarm2]
#Fr Tx Ty Tz Rx Ry Rz SF
1 0.000000 0.000000 0.000000 0.000000 0.000000 -0.000000 -0.000000 1.000000
2 0.000000 0.028644 0.000000 -0.059715 -0.028338 0.077831 0.998560
[Rhand]
#Fr Tx Ty Tz Rx Ry Rz SF
1 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 1.000000
2 0.000000 -0.305908 0.000000 -0.055215 -0.000267 -0.179341 1.001190
[Larm1]
#Fr Tx Ty Tz Rx Ry Rz SF
1 0.000000 0.000000 0.000000 0.000000 0.000000 -0.000000 -0.000000 1.000000
2 0.002791 0.066316 0.043877 0.021400 -0.009270 0.006120 0.999618
[Larm2]
#Fr Tx Ty Tz Rx Ry Rz SF
1 0.000000 0.000000 0.000000 0.000000 -0.000000 -0.000000 -0.000000 1.000000
2 0.000000 -0.098207 0.000000 -0.040839 -0.042263 -0.047321 1.000178
[Lhand]
#Fr Tx Ty Tz Rx Ry Rz SF
1 0.000000 0.000000 0.000000 0.000000 -0.000000 -0.000000 -0.000000 1.000000
2 0.000001 0.038056 -0.000001 0.045240 0.003174 0.097901 0.998387
```

Figure *3.14 – Extrait d'un fichier HTR.*

3.2.4 Post-traitement des données

Bien que les caméras soient capables de fournir des données propres, les contraintes liées à l'environnement perturbent l'interprétation des données. Ces contraintes peuvent par exemple provenir des occlusions de marqueurs, d'un marqueur endommagé, des erreurs logiciels (mauvais étiquetage de marqueurs), du bruit numérique ou d'une mauvaise calibration. Les données collectées peuvent présenter des problèmes de discontinuités ou du bruit parasite. Cortex dispose d'outils qui permettent de corriger les intervalles de temps où un marqueur n'est plus étiqueté, en se basant sur la configuration géométrique des marqueurs (fonction *rectify*) et des outils d'interpolations. Les parties manquantes dans les trajectoires peuvent être interpolées linéairement (fonction *linear join*), à l'aide de splines cubiques (fonction *cubic join*) ou encore en générant des données en fonction de la configuration géométrique de trois autres marqueurs dont les trajectoires sont complètes (fonction *virtual join*). La figure 3.15(a) illustre les méthodes d'interpolation linéaire et par splines cubiques, et la figure 3.15(b) illustre la méthode *virtual join*. Cette dernière technique est intéressante : si on sait que le mouvement à capturer va entrainer des occlusions sur un ou plusieurs marqueurs particuliers, alors on peut ajouter des marqueurs redondants sur le corps en question pour pouvoir lors de la phase de post-traitement reconstruire les trajectoires des marqueurs masqués. La figure 3.16 illustre des données incomplètes d'un marqueur situé en bas du poignet droit. On remarque également que le signal est légèrement bruité. La figure 3.17 illustre l'interpolation de type *virtual join* de la trajectoire de ce marqueur en utilisant les positions des marqueurs situés au coude et en haut du poignet droit. L'étiquetage de marqueurs peut aussi être permuté manuellement si les marqueurs ont été mal identifiés. Il est possible de lisser les trajectoires des marqueurs selon deux méthodes : un filtre de moyenne glissante ou un filtre de Butterworth. Le filtre de Butterworth est un filtre passe bande paramétrable possédant de bonnes propriétés pour l'étude des mouvements biomécaniques. Ce filtre permet de retirer les composantes du mouvement qui ont des fréquences trop élevées pour avoir été réalisés par un humain.

3.3 Imitation : Reproduction de mouvements capturés

La pile de tâches permet de générer un mouvement à partir d'une référence générique. Les outils de capture de mouvements peuvent donc être utilisés pour fournir une référence n'étant pas issue d'un modèle mathématique. Il est ainsi possible de générer des mouvements qui ne représentent pas forcement l'exécution d'une tâche robotique et son donc plus expressif. Ce paragraphe décrit comment des données issues de la capture de mouvement d'un humain peuvent être utilisées comme trajectoire de référence pour une tâche qui doit être exécuté par un robot. Dans tout les cas, un changement de repère est nécessaire pour pouvoir calculer l'erreur du suivi. On pourra utiliser par exemple comme référentiel commun un repère dont l'axe $\boldsymbol{z}$ est normal au sol, la direction de l'axe $\boldsymbol{y}$ est définie par la direction du vecteur $(\mathbf{p}_{\mathsf{Rwaist}} - \mathbf{p}_{\mathsf{Lwaist}})$ où $\mathbf{p}_{\mathsf{Rwaist}}$ (respectivement $\mathbf{p}_{\mathsf{Lwaist}}$) est la position d'un point situé sur le côté droit (respectivement gauche) du bassin. Un deuxième changement de repère sera nécessaire si pour une posture *identique*, les repères attachés aux corps de l'humain et du robot n'ont pas des transformations identiques par rapport au référentiel d'origine (voir figure 3.18).

3.3.1 Tâches et suivi de trajectoires

D'une manière générale, les trajectoires des tâches sont directement reliées à leurs formulations mathématiques $\mathbf{e} = \mathbf{s}(t) - \mathbf{s}^*(t)$. Par exemple, pour attraper une balle, la main doit se déplacer en direction de la balle : $\mathbf{e}(t) = \mathbf{p}_{\mathsf{main}}(t) - \mathbf{p}^*_{\mathsf{balle}}$. En revanche, certaines tâches ont des trajectoires qui ne suivent pas un motif particulier. Si $\mathbf{s}^*$ est observable, alors des tâches de suivi de trajectoires peuvent être utilisées dans une pile de tâches pour reproduire les parties choisies d'un mouvement. Nous considérons qu'il y a une imitation lorsque le mouvement doit être adapté à une structure cinématique différente de la structure sur laquelle le mouvement a été observé.

Si nous possédons des connaissances sur l'action observée, nous pouvons choisir explicitement des tâches pertinentes à reproduire. Par opposition, des tâches apparaissent implicitement si par

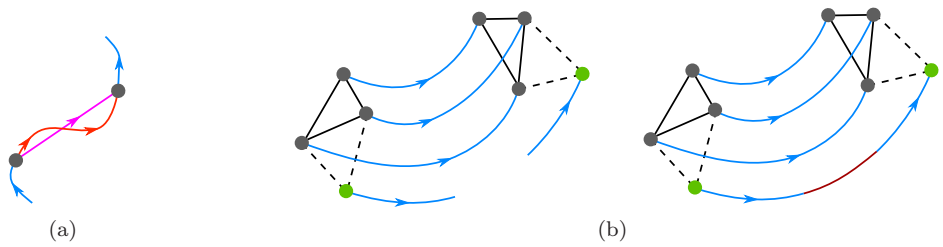


Figure *3.15 – (a) La trajectoire incomplète d'un marqueur est interpolé linéairement ou à l'aide d'une spline cubique. (b) La trajectoire incomplète d'un marqueur est reconstruite à partir de sa configuration géométrique par rapport à trois autres marqueurs.*

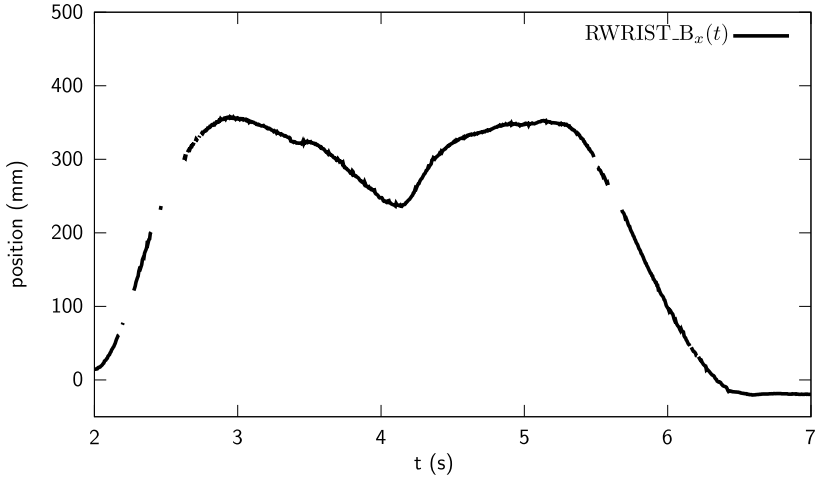


Figure *3.16 – La composante en x de la trajectoire du marqueur placé en bas du poignet droit présente des discontinuités et du bruit à hautes fréquences.*

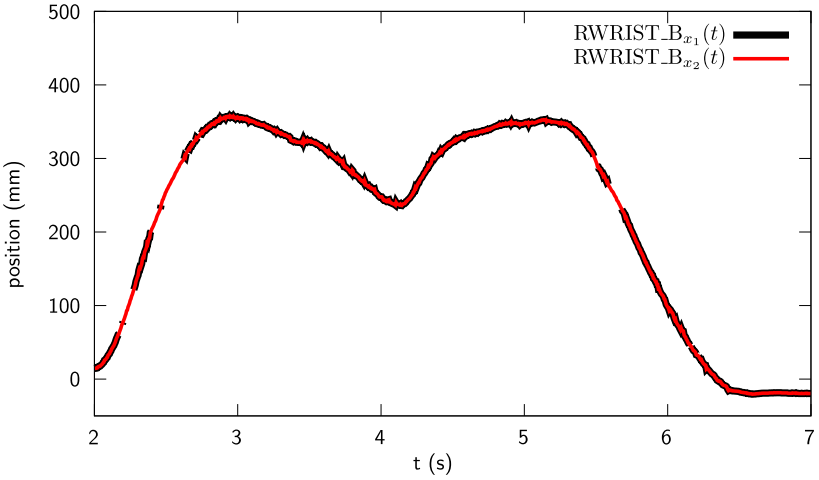


Figure *3.17 – Interpolation de la trajectoire du marqueur en bas du poignet droit.*

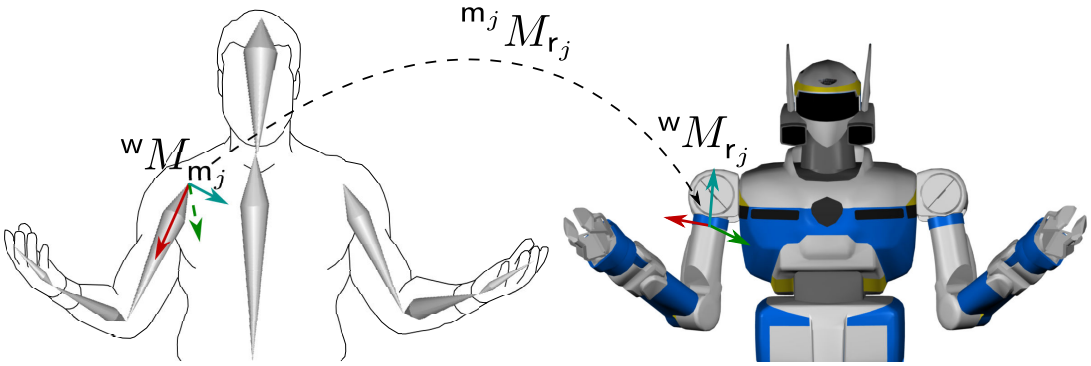


Figure *3.18 – Un changement de repère est nécessaire : les repères associés aux bras ne sont pas concordant. La transformation homogène $^{m_j}M_{r_j}$ est calculée en utilisant des postures concordantes sur le squelette de la capture de mouvements et le modèle du robot.*

exemple les trajectoires articulaires sont strictement recopiées. Par exemple considérons le cas du mouvement d'attente. Ce mouvement est défini comme étant le mouvement inconscient qu'un humain produit lorsqu'il *ne fait rien*. Il semble très difficile de définir simplement ce mouvement sous forme d'une combinaison de tâche, c'est-à-dire d'un modèle causal, sans donner une interprétation de ses caractéristiques, par exemple en remontant à la structure élastique de l'actionnement humain. Nous considérons que le mouvement d'attente d'un humain se traduit par les mouvements de ses bras, du torse et de la tête. Par conséquent le mouvement d'attente peut être représenté par une pile de tâches constituée de tâches de suivi de trajectoire des mains, et de la tête. Le torse n'est pas choisi dans la pile car les chaînes articulaires contrôlant les mains et la tête contiennent le torse. Le couplage des tâches entrainera le torse d'une manière cohérente dans le mouvement des autres membres. On note que l'utilisation de la capture de mouvements nous permet de générer un mouvement expressif où le robot *ne fait rien*. Alors qu'en ne considérant l'action *ne rien faire* d'un point de vue strictement robotique, la commande générée correspondrait à une vitesse nulle sur chaque articulation.

3.3.2 Tâches possibles et mesures des trajectoires de références

Pour une tâche d'atteinte d'une position 3D dans l'espace cartésien, la trajectoire de la position mesurée $\hat{\mathbf{p}}$ d'un marqueur peut être utilisée comme le signal de référence pour une tâche de suivi en position 3D (voir figure 3.19).

L'orientation du regard peut être approchée par l'orientation de la tête. En utilisant les positions de trois marqueurs placés sur la tête, on définit un repère et la matrice de transformation de rotation par rapport au repère du monde représentera l'orientation de la tête dans ce repère (voir figure 3.20). La position du centre de masse est utilisé comme un critère de stabilité statique : sa projection sur le sol doit rester à l'intérieur du polygone de support défini par les deux pieds d'un humanoïde. Une tâche de suivi de position du centre de masse nécessite la connaissance d'une trajectoire de référence. Le problème est que la position du centre de masse n'est pas directement observable avec les outils de capture de mouvements. Il est cependant possible de faire une approximation de sa trajectoire projetée au sol pour obtenir une trajectoire de référence satisfaisant la contrainte d'équilibre statique. Dans [Montecillo-Puente *et al.*, 2010], une approximation de la trajectoire du centre de masse projetée au sol se basant sur les mouvements de la tête est proposée afin d'imiter des mouvements de transfert de point d'appui :

- lors d'une phase de double support, la projection au sol du centre de masse évolue suivant un axe défini par la position des deux pieds, et son déplacement est proportionnel à la projection orthogonale du vecteur vitesse de la tête projeté au sol sur l'axe défini par les pieds,
- lors d'une phase de simple support, la projection du centre de masse se déplace suivant le vecteur

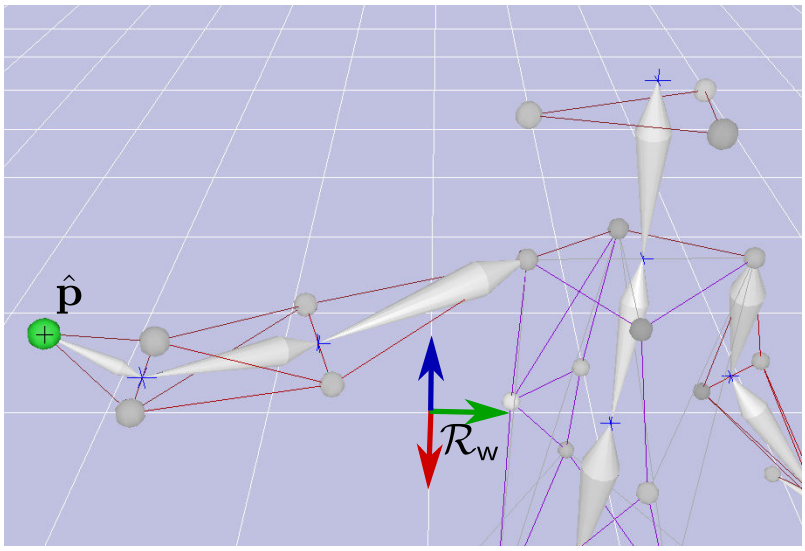


Figure *3.19 – La position mesurée $\hat{\mathbf{p}}$ d'un marqueur est utilisé comme référence pour une tâche de suivi de position 3D.*

vitesse de la tête projeté au sol.
Cette approximation est justifiée par l'importance du mouvement de la tête dans des mouvements humains qui est souligné dans les études en neurosciences [Sreenivasa *et al.*, 2009]. La figure 3.21 illustre cette approximation.

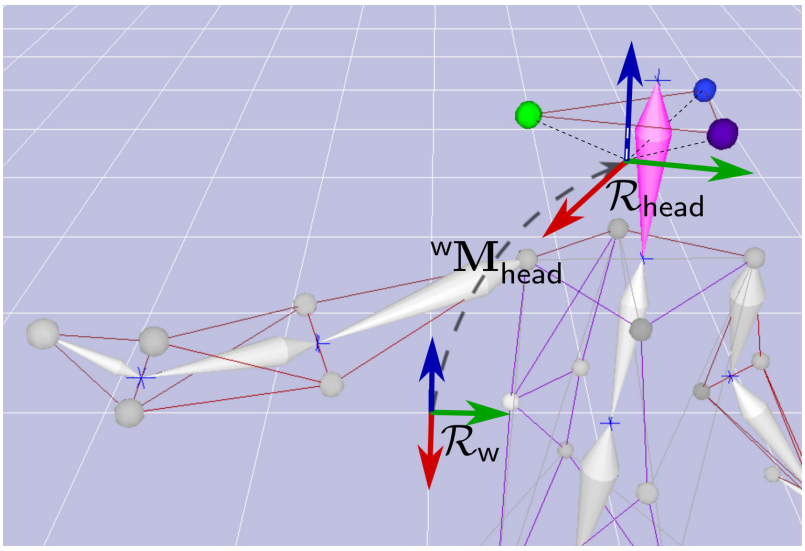


Figure *3.20 – Un repère est associé à la tête à partir de trois marqueurs. La matrice de transformation du repère d'origine $\mathcal{R}_w$ vers le repère associé à la tête $\mathcal{R}_{head}$ ${}^{w}\mathbf{M}_{head}$ donne l'orientation de la tête.*

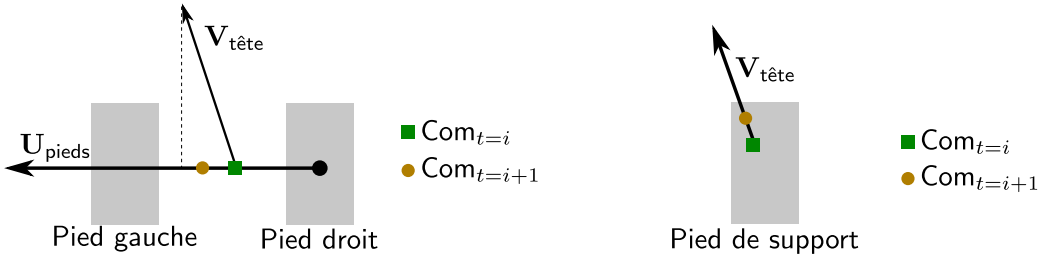


Figure *3.21 – Une approximation du déplacement du centre de masse lors d'un mouvement est effectuée en fonction de la vitesse de déplacement de la tête pour le cas d'un mouvement en double support et en simple support.*

3.3.3 Application au mouvement d'attente

Lorsqu'un humain attend, l'intégralité de son corps est en mouvement. Cependant, il n'est pas possible d'exprimer facilement une tâche d'attente (au sens fonction de tâche). C'est pour cette raison que pour reproduire ce mouvement, nous séléctionnons un ensemble de tâches qui vont, par combinaison, reproduire les caractéristiques du mouvement.

3.3.3.1 Réplication du mouvement

Les tâches de suivi de trajectoire des mains et de l'orientation de la tête, présentées dans le paragraphe précédent, sont choisies comme étant les tâches caractérisant un mouvement d'attente. Les trajectoires correspondantes sont donc mesurées sur un humain à l'aide du système de capture de mouvement, et serviront de trajectoires de références. Pour garantir la stabilité du robot, deux tâches sont ajoutées en haute priorité (*double support* et *centre de masse*) : les deux pieds sont contraints à rester au sol, et le centre de masse est fixe en x et en y (la hauteur n'est pas contrainte). Pour éviter les postures trop proches des singularités, une tâche de posture est utilisée en faible priorité : lors du mouvement, le robot gardera une posture proche de la configuration de repos (configuration *half-sitting*). La figure 3.22 illustre la pile de tâches utilisée pour réaliser un mouvement d'attente.

Posture *Half-sitting* (30D)
Main droite (3D)
Main gauche (3D)
Tête (6D)
Centre de masse (2D)
Double support (6D)

Figure *3.22 – La pile de tâches qui va permettre de reproduire le mouvement d'attente.*

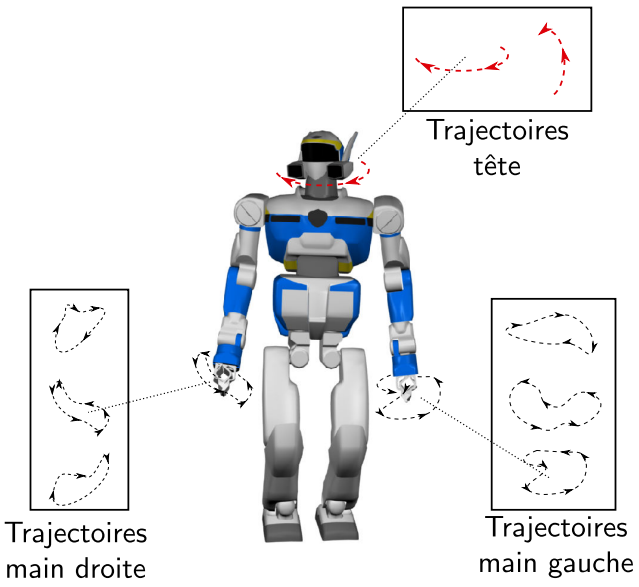


Figure *3.23 – Pour chaque tâche de suivi, des trajectoires sont choisies comme référence parmi un ensemble de trajectoires enregistrées.*

Cette pile de tâches sera exécutée par un robot HRP-2.

3.3.3.2 Edition du mouvement

Pour donner un aspect aléatoire au mouvement exécuté par le robot, plusieurs démonstrations de mouvements d'attente sont enregistrées. En ce qui concerne le mouvement d'attente, les positions atteintes par les différents membres ne sont pas importantes. Ce sont les déplacements de ces membres qui caractérisent ce mouvement. Il n'est donc pas capital que le suivi de trajectoire soit parfaitement accompli. Toutes les trajectoires sont éditées afin que les positions de début et de fin soient identiques pour chaque ensemble de trajectoires. Ainsi n'importe quel séquencement des trajectoires reste continu. Pour chaque tâche, une des trajectoires enregistrées correspondant à cette tâche est choisie aléatoirement puis est définie comme trajectoire de référence. Lorsque la trajectoire a été exécutée en totalité, une nouvelle trajectoire est choisie comme référence et ainsi de suite (voir figure 3.23). La figure 3.24 illustre le mouvement d'attente réalisé par le robot HRP-2 en utilisant la pile de tâches, et les vidéos associées sont disponibles [1].

1. `http://homepages.laas.fr/shak/videos/`

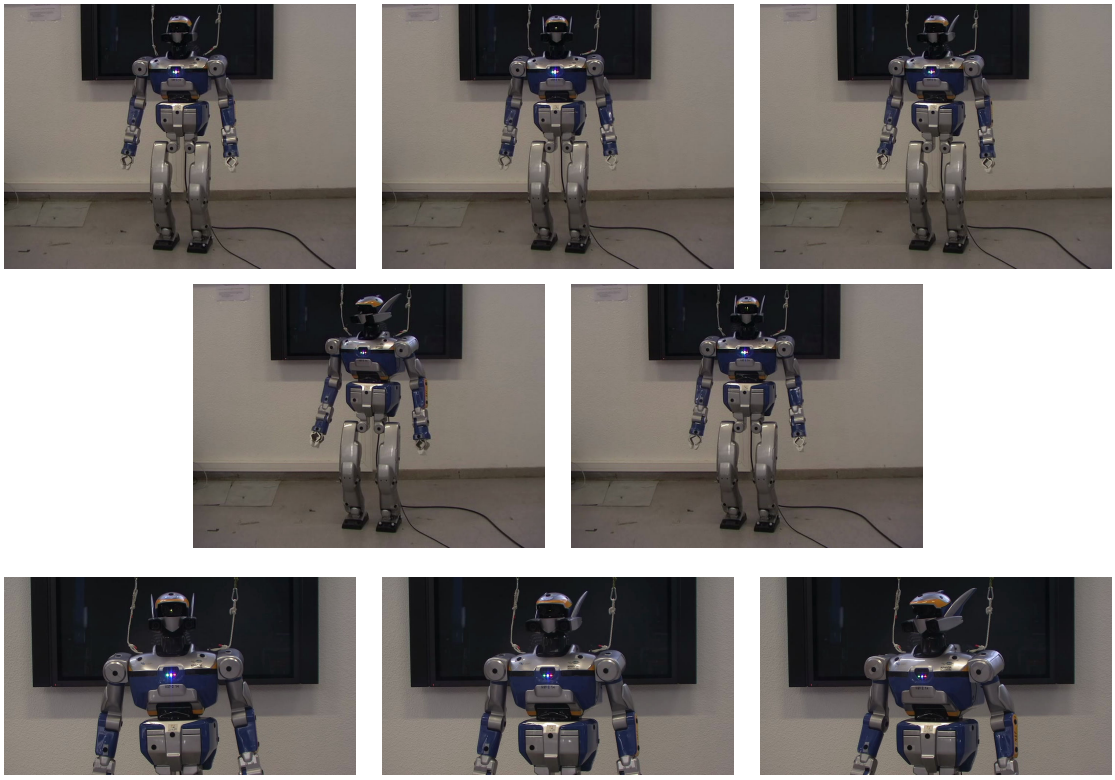


Figure *3.24 – Le mouvement d'attente réalisé par le robot HRP-2.*

3.3.4 Adaptation du mouvement au modèle

Les travaux présentés dans [Shin *et al.*, 2001] introduisent la notion d'*importance* d'un effecteur. Il s'agit d'un critère qui permet de déterminer si la position d'un effecteur est plus importante que la configuration articulaire associée à sa chaîne cinématique. L'importance des effecteurs est déterminée en analysant la posture d'un personnage en relation avec son environnement : lorsqu'un effecteur touche un objet (distance nulle), la position de l'effecteur est considérée comme étant plus importante que la configuration articulaire. Par exemple, lorsqu'une main touche un objet, il est important de conserver ce contact et donc la position de la main doit être recopiée en priorité. Par contre si la main est en l'air alors, la configuration articulaire est plus importante car il est possible que la position de la main ne soit pas à portée du personnage, à cause de la structure différente, qui reproduit le mouvement. La figure 3.25 illustre un exemple de mouvement où en fonction de l'effecteur, la configuration articulaire ou la position doit être respectée.

Ce paragraphe s'appuie sur le travail réalisé par [Ramos *et al.*, 2011] dans le cadre de l'édition de mouvement dynamique issu d'un mouvement humain pour l'adapter à la dynamique d'un robot. L'exemple présenté illustre également l'importance du choix des tâches à reproduire. L'utilisation de la dynamique dans la génération de mouvements a pour but de rendre le mouvement exécuté le plus réaliste possible.

Considérons un mouvement de yoga réalisé par un humain est enregistré par capture de mouvements (voir figure 3.26). Si le mouvement est converti en trajectoire articulaire et est appliqué directement sur le robot en cinématique, alors plusieurs problème liés à la différence de structures apparaissent. Le pied d'appui décolle du sol (problème connu en animation sous le nom de *foot skating* ou *foot sliding*), les mains rentrent en collision puis transpercent le torse (voir figure 3.27).

En rejouant la trajectoire mais cette fois ci en respectant la dynamique du robot (en utilisant une extension de la pile de tâches en dynamique [Saab *et al.*, 2011]), le pied d'appui est respecté mais le résultat n'est plus le mouvement de yoga puisque le robot perd l'équilibre à cause de ses propres contraintes dynamiques différente de l'humain (voir figure 3.28).

En analysant le mouvement initial, on peut déterminer les caractéristiques importantes du mouvement. Dans l'exemple du mouvement de yoga, la position relative d'une main par rapport à l'autre est importante, de même la position du centre de masse (et donc l'équilibre) doivent être contrôlées correctement. En choisissant ces caractéristiques comme étant les tâches à exécuter pour reproduire le mouvement, le mouvement généré devient correct (voir figure 3.29). Enfin la figure 3.30 illustre le mouvement appliqué sur le vrai robot.

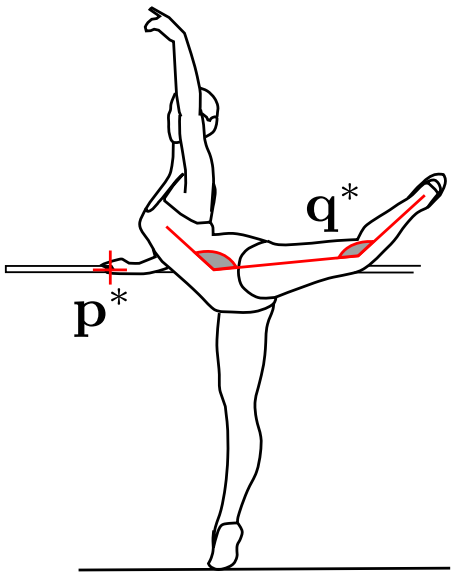


Figure *3.25 – Pour reproduire le mouvement de la danseuse, il est plus important de respecter la position de la main droite touchant la barre que la configuration articulaire associée. En ce qui concerne le pied gauche, il est plus important de respecter la configuration articulaire pour conserver l'allure du mouvement.*

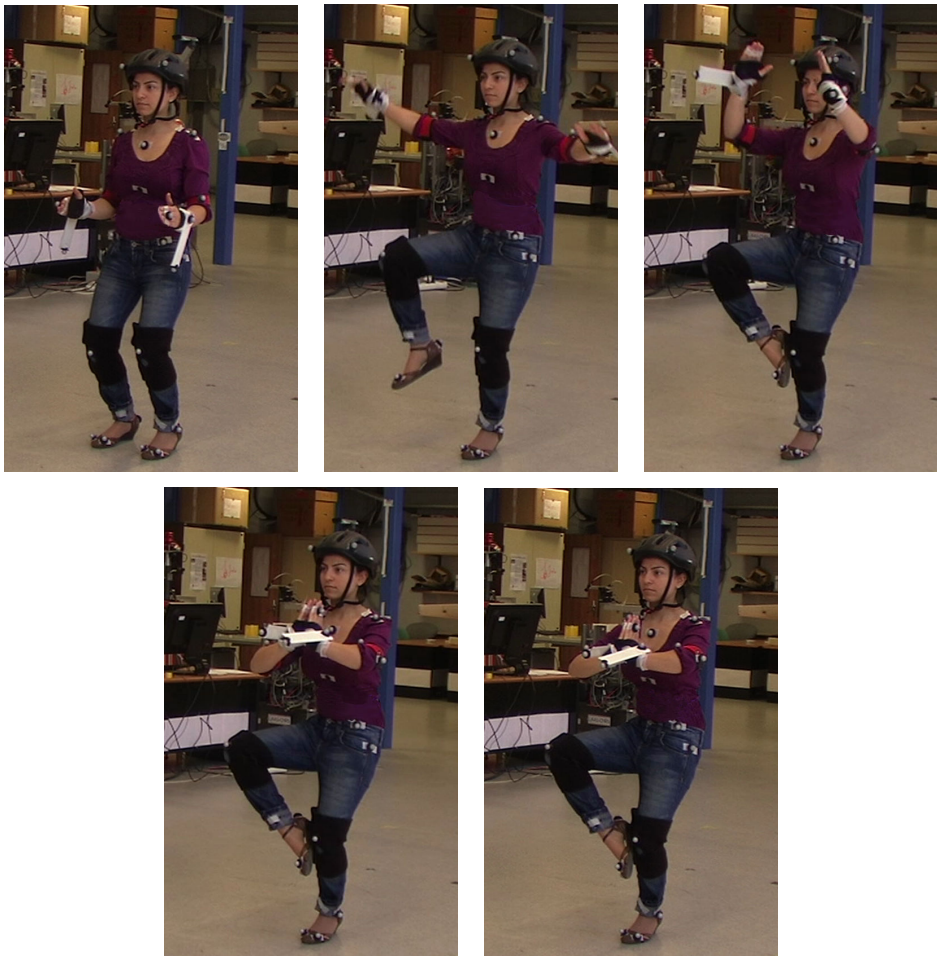


Figure *3.26 – Un mouvement de yoga enregistré grâce au système de capture de mouvements.*

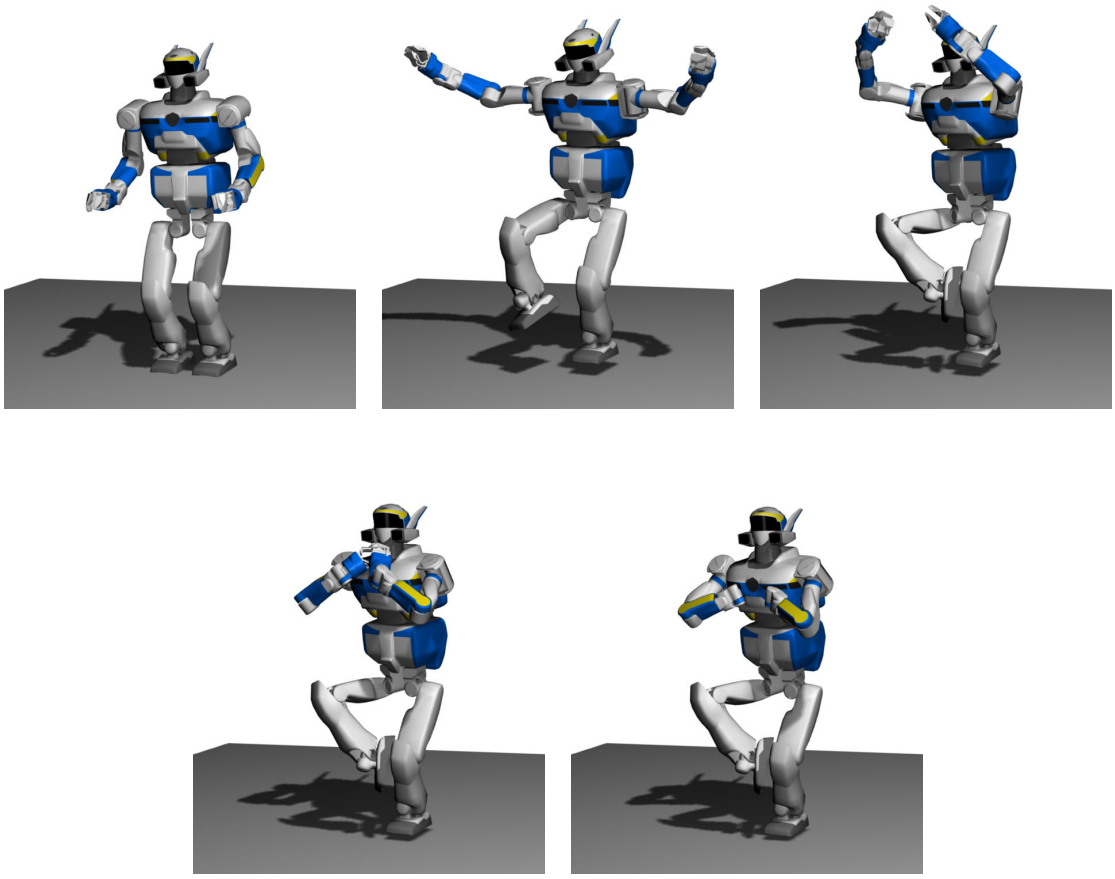


Figure *3.27 – Le mouvement de yoga reproduit en cinématique en suivant les trajectoires articulaires calculées à partir des données issues de la capture de mouvements. Plusieurs problèmes apparaissent : le pied d'appui perd son contact avec le sol, les mains rentrent en collision et transpercent le torse.*

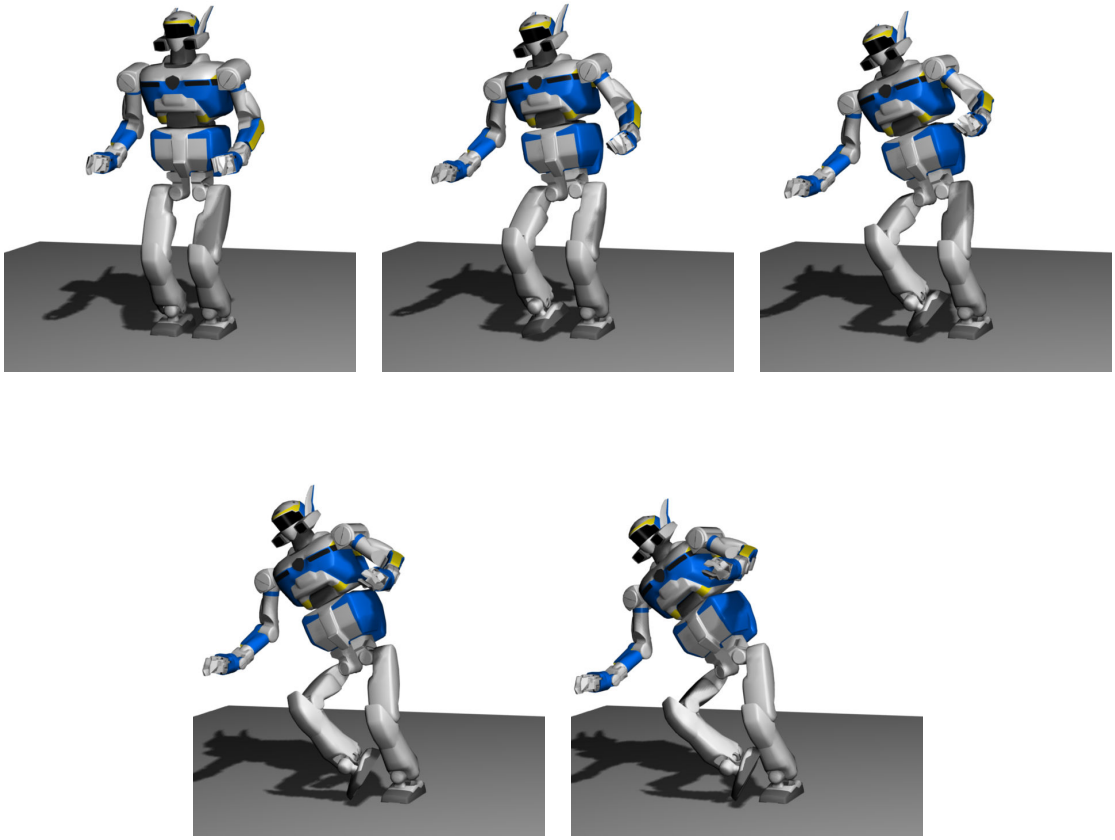


Figure *3.28 – Le mouvement de yoga reproduit en dynamique. Le robot perd l'équilibre à cause de ses contraintes dynamiques.*

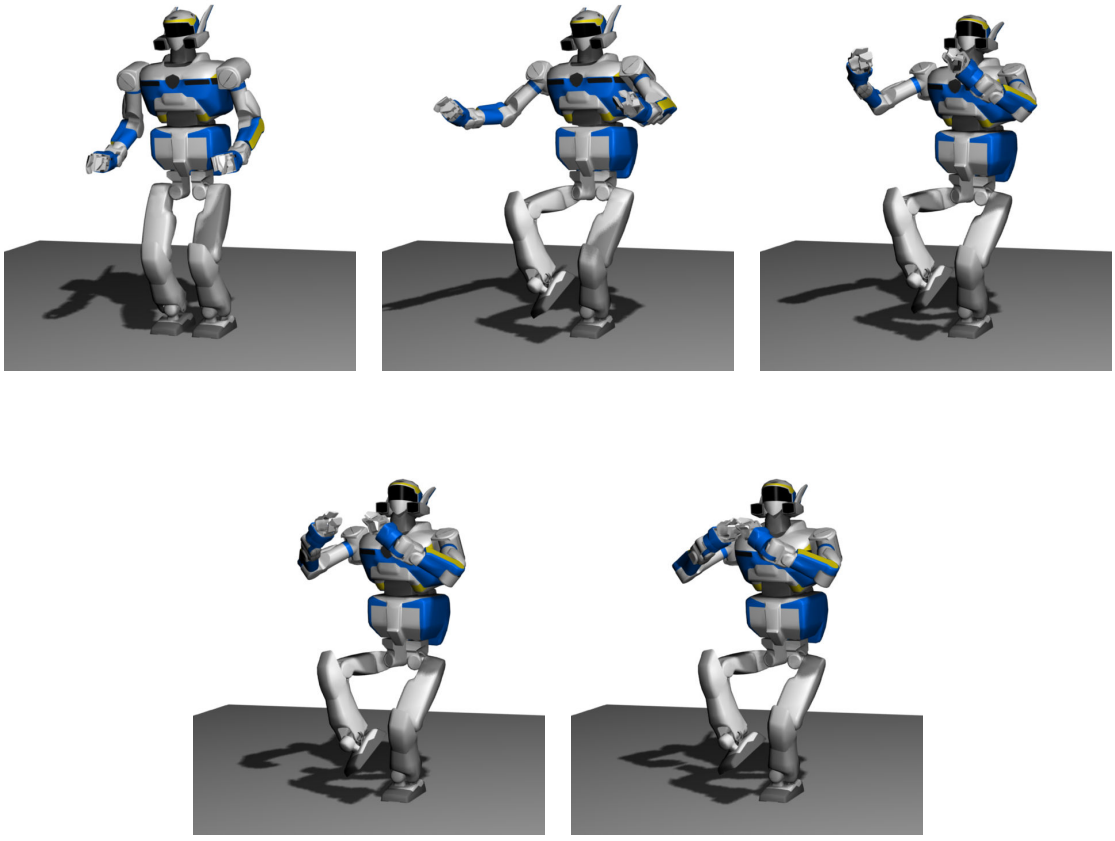


Figure *3.29 – Le mouvement de yoga reproduit en dynamique après édition du mouvement original pour définir correctement les tâches caractéristiques : la posture, la distance relative entre les deux mains, et le contrôle du centre de masse. Le mouvement généré est fidèle à la démonstration (sans collision ni perte d'équilibre).*

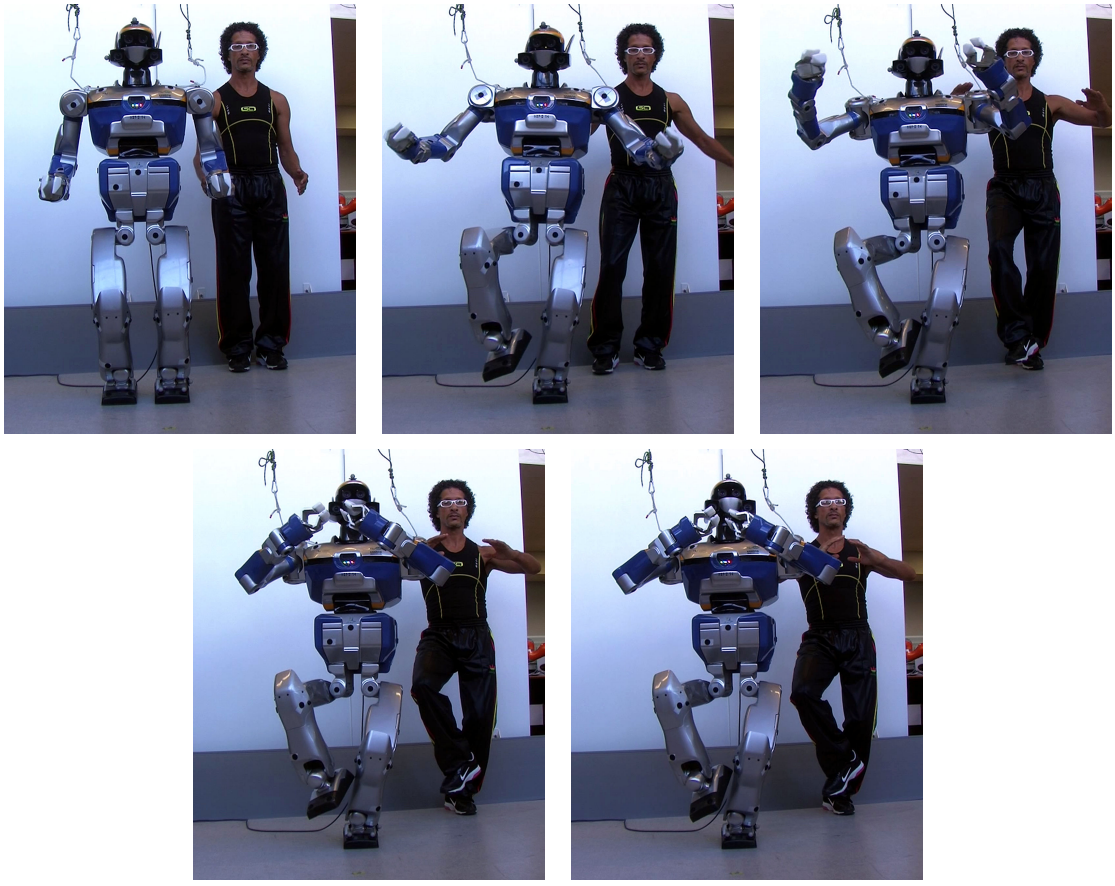


Figure *3.30 – Le mouvement de yoga reproduit sur le robot HRP-2.*

Conclusion

Dans ce chapitre, nous avons présenté différents outils existant pour acquérir des mouvements réels en détaillant les avantages et inconvénients des différentes techniques. Les techniques de capture de mouvements permettent d'analyser les données récoltées afin d'étudier un mouvement, mais aussi d'animer des objets ou des personnages virtuels en s'inspirant de mouvements réels. La possibilité d'éditer les données représentant les mouvements permet d'adapter un mouvement à différente morphologie ou de créer des variations de mouvements. Nous avons aussi souligné l'importance du choix des tâches qui une fois exécutées, vont permettre de conserver les caractéristiques du mouvement. Nous avons présenté comment un système de capture de mouvements a permit de fournir des trajectoires de références pour des tâches de suivi de trajectoires, fournissant ainsi un moyen d'élargir l'expressivité des mouvements générés pour le robot. Ces tâches peuvent ensuite être utilisées, via une pile de tâches, pour reproduire sur un robot HRP-2 un mouvement difficilement définissable par un modèle de génération : le mouvement d'attente. Dans le chapitre suivant le problème de reconnaissance de tâches est introduit, et les données issues de la capture de mouvements seront utilisées pour analyser des trajectoires associées à un mouvement.

Chapitre 4

Reconnaissance de tâche en robotique par commande inverse

Nous présentons dans ce chapitre une méthode de reconnaissance de tâches particulièrement adaptée à la reconnaissance de tâches exécutées en parallèle. L'originalité de la méthode présentée ici est d'utiliser les propriétés de la fonction de tâches, classiquement utilisée pour la génération de mouvements, pour effectuer la reconnaissance de tâches. L'idée principale est d'effectuer une rétro-ingénierie sur un mouvement observé, connaissant l'ensemble des tâches qui peuvent apparaître dans ce mouvement et en utilisant les trajectoires engendrées par les lois de commandes dans les espaces de tâches comme étant des trajectoires caractéristiques. Sous l'hypothèse que le mouvement a été généré en empilant un ensemble de contrôleurs, le mouvement est traité afin de rechercher les comportements connus dans chaque espace des tâches. Nous appelons cette approche de rétro-ingénierie la *commande inverse*. Alors que toutes les approches présentées dans le paragraphe suivant ne peuvent reconnaitre que des tâches non parallèles, nous nous appuyons sur les principes de la redondance en contrôle pour reconnaitre des ensembles de tâches parallèles. Lors de la phase de reconnaissance, le mouvement est projeté dans des espaces orthogonaux aux espaces des tâches déjà détectées. Cela assure un découplage efficace des tâches effectuées par le robot.

Nous avons introduit dans le chapitre 2 les fondements sur lesquels les travaux présentés s'appuient : le formalisme de la fonction de tâches. L'algorithme proposé pour effectuer l'analyse de mouvement est présenté dans les paragraphes suivants. Enfin, des expérimentations validant la méthode sont présentées en simulation et sur le robot HRP-2.

4.1 Etat de l'art

La reconnaissance de tâches est un problème qui est apparu assez tôt en robotique. Dans la communauté de la vision, ce problème est généralement considéré pour les mouvements non-structurés : aucune hypothèses ne sont faites, que ce soit sur la forme ou la rigidité des corps qui bougent.

Cependant, les informations sont issues de l'environnement et la reconnaissance est principalement faite à partir du contexte, par exemple, les points saillants en temps et en position (le flux de vidéo 2D est considéré comme étant une fonction 3D) [Laptev, 2005]. Ces points sont appris d'une base de données, puis associés pendant une démonstration. L'environnement est aussi utilisé pour effectuer une extraction d'arrière plan pour dégager une silhouette et effectuer, par exemple, une reconnaissance

de démarche [Liu et Sarkar, 2005].

La structure du corps poly-articulé est souvent supposée connue et est utilisée pour estimer sa pose. Par exemple l'estimation d'une trajectoire de posture d'un humanoïde dans [Zordan et Horst, 2003] est exécutée en utilisant des données issues d'un système de capture de mouvements pour guider un modèle physique connu. Dans le reste de ce chapitre, nous considérons que le modèle cinématique du corps poly-articulé est connu.

Les outils statistiques ont été appliqués avec succès à la reconnaissance d'actions et à l'analyse de mouvements [Schaal *et al.*, 2003]. Ces outils sont utilisés pour créer des symboles, et par extension, les détecter dans un mouvement. Par exemple, une méthode pour le contrôle basé comportement est présentée dans [Drumwright et Matarić, 2003, Drumwright *et al.*, 2004]. Les comportements sont définis comme étant des symboles de mouvements (par exemple un direct, un crochet, un uppercut, une garde). Les comportements sont modélisés par apprentissage à partir d'une série d'exemples. Une réduction de dimension est alors appliquée pour avoir une classification significative. La reconnaissance est gérée par un classifieur bayésien qui reconnait une trajectoire dans l'espace articulaire ou l'espace cartésien. L'imitation est une extension de la reconnaissance. Elle est effectuée en interpolant des exemples connus pour obtenir des trajectoires réalisables. L'introduction des processus de décision markoviens partiellement observables (partially observable Markov decision process, POMDP) ou des inférences bayésiennes [Pearl, 1988] ont relancé le domaine de la modelisation d'action [Kaelbling *et al.*, 1998] ces dernières années. De telles techniques sont appliquées à l'apprentissage de primitives motrices [Peters et Schaal, 2008] et à la segmentation de mouvements [Calinon *et al.*, 2010, Inamura *et al.*, 2004]. Les modèles de Markov cachés (HMMs) ont été largement utilisés dans des travaux antérieurs, par exemple, pour exécuter une reconnaissance d'action ou de démarche [Gu *et al.*, 2010] ou pour générer des mouvements proches du mouvement humain [Kwon et Park, 2008]. Dans ces travaux, des données issues de captures de mouvements humain sont utilisées pour construire une base dans l'espace articulaire pour chaque classe de mouvement en utilisant une technique de réduction de dimension. En parallèle, des HMMs sont entrainés pour capturer les caractéristiques relatives à la classe de mouvement dans l'espace des tâches. La génération est obtenue en trouvant la combinaison linéaire optimale des éléments de la base qui va maximiser la probabilité du HMM entrainé. Bien que l'espace des tâches soit considéré comme étant un espace où les caractéristiques doivent être extraites, la méthode est limitée à un espace de tâche spécifique par mouvement généré. Dans [Liang *et al.*, 2009] des modèles de Markov à tailles variable sont utilisés pour apprendre des actions atomiques d'humain. Une séquence d'actions atomiques représente un comportement complet. D'une manière générale, l'efficacité des techniques de reconnaissance basé sur les outils statistiques est dépendante de la qualité de la base de données construite lors de la phase d'apprentissage. Plusieurs démonstrations pour chaque cas particulier sont nécessaire afin d'extraire les invariants qui vont discriminer les tâches. La couverture des données d'apprentissage peut aussi limiter l'efficacité de la reconnaissance. Enfin, l'ensemble des démonstrations doivent être associées aux bons symboles, ce qui est généralement donné à l'algorithme d'apprentissage.

D'une autre façon, la reconnaissance peut se baser sur des critères spécifiques qui sont des a priori donnés au système. Dans [Nakaoka *et al.*, 2007], seules les trajectoires du robot sont utilisées pour distinguer les différentes phases de mouvement. Dans ces travaux là, une tâche est un mouvement corps complet du robot durant un segment temporel. Le mouvement global est une séquence de tâches. Chaque tâche possède ses propres paramètres appelés *skill parameters*. La méthode de reconnaissance de tâche est décomposée en deux étapes : d'abord, étiqueter tous les segments temporels du mouvement observé avec les noms des tâches. La seconde est l'estimation des *skill parameters* pour chaque segment. Chaque tâche est détectée par l'analyse de trajectoires projetées dans un espace spécifique. Par exemple, la tâche *faire un pas* est détectée en analysant la trajectoire d'un pied ; une tâche de *squat* est détectée en analysant la trajectoire verticale du bassin. Le critère utilisé pour la détection et le choix des espaces de projections sont ad-hoc, construits manuellement pour un mouvement particulier qui doit être imité par le robot. D'une manière similaire, [Mühlig *et al.*, 2009] utilise un ensemble d'espaces spécifiques dans lequel le mouvement observé est projeté. Des critères ad-hoc sont utilisés pour choisir automatiquement l'ensemble des espaces de tâches qui va décrire le mieux un

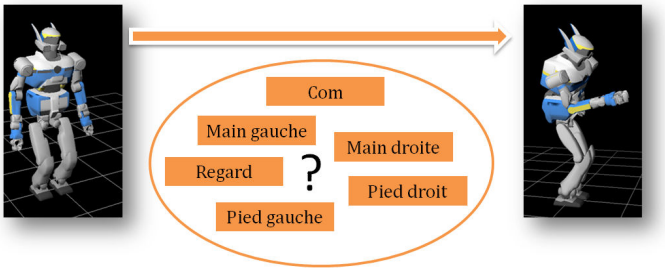


Figure *4.1 – Nous proposons de déterminer automatiquement quelles sont les tâches, parmi un ensemble connu, qui ont servi à générer le mouvement observé.*

mouvement donné, dans le but de focaliser la technique d'apprentissage dans ce nouvel espace. Le critère utilisé pour la sélection de l'espace de tâche est exprimé par des fonctions de coût inspirées de neurosciences : la saillance de l'objet manipulé, la variance de la dimension d'un espace à travers plusieurs démonstrations, et quelques heuristiques qui indiquent que les mouvements fatiguants ou inconfortables sont provoqués par l'exécution de tâches. Ces heuristiques règlent le problème de détection de tâches d'immobilité. Cette méthode présentée ajoute des informations de plus haut niveau que les analyses purement statistiques et s'appuie sur les espaces de tâches comme étant des espaces appropriés pour représenter le mouvement. Cependant, l'efficacité de la sélection de tâches dépend de la puissance des heuristiques.

L'approche commune de ces travaux est de projeter le mouvement observé dans un espace réduit particulier, dans lequel la reconnaissance est plus facile. Ces espaces sont choisis arbitrairement [Nakaoka *et al.*, 2007], séléctionnés automatiquement [Mühlig *et al.*, 2009], ou obtenus par apprentissage [Peters et Schaal, 2008]. L'analyse en composantes principales (PCA) est une technique de réduction de dimension qui permet de trouver des espaces réduits. Malheureusement cette technique ne tient pas compte des corrélations non linéaires entre les différents espaces réduits, et donc la technique est inadapté aux mouvements humanoïdes qui comporte de telles corrélations non linéaires. Il existe des techniques d'analyse en composantes principales non linéaires. Ces techniques sont plus adaptées aux mouvements humanoïdes [Chalodhorn *et al.*, 2009]. Cependant les espaces réduits sont dégagés automatiquement, or, en contrôle, des espaces de tailles réduites, présentant des corrélations non linéaires, sont construit pour définir des objectifs et moduler le comportement du robot. Par exemple l'approche de la fonction de tâches [Samson *et al.*, 1991] exprime des commandes génériques dans un espace de tâche donné de dimension n. L'approche a été étendue pour gérer un ensemble hiérarchique de tâches [Siciliano et Slotine, 1991, Nakamura *et al.*, 1987] en utilisant la redondance du système. Ces espaces de tâche représentent ainsi des espaces parfaitement appropriés pour effectuer des analyses et reconnaître des motifs particuliers. Nous proposons ainsi d'utiliser directement ces contrôleurs connus pour la génération de mouvement plutôt que d'utiliser des primitives déterminées automatiquement a posteriori.

Dans le chapitre 2, le formalisme de la fonction de tâches a été présenté. Chaque tâche peut être utilisée pour générer un modèle de mouvement dans diverses situations. Dans ce sens, une tâche est à la fois un contrôleur qui peut générer un mouvement, mais également un descripteur du mouvement exécuté. Ce descripteur est quasi symbolique, mais contient aussi des paramètres additionnels qui caractérisent la façon dont il est exécuté : par exemple, une tâche régie par une décroissance exponentielle est paramétrée par λ qui caractérise la vitesse de décroissance. Dans ce chapitre, nous proposons d'utiliser les tâches comme un ensemble de descripteurs pour reconnaitre un mouvement observé, en identifiant l'ensemble des tâches qui ont été utilisées pour générer ce mouvement (voir la figure 4.1).

L'ensemble des tâches identifiées est utilisé pour caractériser le mouvement observé, par exemple pour distinguer deux mouvements visuellement proches.

Nous présentons d'abord les hypothèses considérées pour l'application de la méthode de reconnais-

sance de tâches, ensuite, les différentes étapes constituant la reconnaissance de tâches sont présentées. Ensuite, des expérimentations sont effectuées en simulation pour illustrer le comportement de la méthode dans des conditions idéales, puis en utilisant un système capture de mouvement pour acquérir les mouvements à analyser joués par un vrai robot. Ces expérimentations illustrent la précision et la robustesse aux bruits de la reconnaissance de tâches en distinguant des paires de mouvements visuellement proches. Enfin, des expérimentations prospectives sur la validité de la méthode pour l'analyse du mouvement humain sont présentées.

4.2 Méthode

4.2.1 Hypothèses

On suppose que le modèle de comportement de toute les tâches potentiellement utilisées sont connus ($\mathbf{e}$, $\dot{\mathbf{e}}^*$, $\mathbf{J}$). Le modèle du robot ainsi que toutes les tâches susceptibles d'intervenir dans le mouvement sont connus. L'ensemble de ces tâches est appelé le *lot de tâches* et représente les capacités du robot. Le mouvement observé est supposé avoir été généré en utilisant un sous-ensemble inconnu du lot de tâches. On suppose aussi que les tâches intervenant dans le mouvement observé sont compatibles au sens des projections $\mathbf{P}$ définies dans (2.31), c'est-à-dire qu'il n'y a pas de singularité algorithmique [Chiaverini, 1997]. Enfin l'ensemble des tâches actives ne changent pas d'état pendant le mouvement (une pile de tâches fixe est valide pendant toute la durée du mouvement). Enfin, le mouvement observé est donné sous la forme de trajectoire articulaire : $\hat{\dot{\mathbf{q}}}(t)$. Dans notre cas, cette trajectoire articulaire sera reconstruite à partir de données issues d'un système de capture de mouvements comme décrit dans le paragraphe 4.4.1.

4.2.2 Présentation générale

L'idée générale est de reconstruire de manière itérative la pile de tâches qui aurait généré le mouvement observé. La trajectoire articulaire observée est projetée dans chacun des espaces de tâches issu d'un lot de tâches connus. Les trajectoires projetées sont ensuite comparé par optimisation à une trajectoire théorique calculée grâce au modèle de génération. Cette comparaison correspond à une fonction de coût dont le score va servir de critère de selection. La tâche qui donne la trajectoire projetée de plus faible score est sélectionnée comme étant une tâche intervenant dans le mouvement.

Les effets de la tâche sélectionnée sont annulés en projetant la trajectoire articulaire dans l'espace nul de la tâche. Le mouvement résultant est alors utilisé pour l'itération suivante.

Si à l'itération n le mouvement résultant de l'annulation d'une tâche est nul, alors tous les mouvements présent dans le mouvement initial ont été justifiés par l'exécution des tâches sélectionnées. Par conséquent, la pile de tâches qui aurait généré le mouvement observé est constituée des tâches sélectionnées.

L'algorithme est présenté dans l'algorithme 1. La trajectoire de vitesse articulaire du mouvement observé est notée $\hat{\dot{\mathbf{q}}}(t)$. $\mathbf{P}_A\dot{\mathbf{q}}(t)$ représente les résultats des mouvements projetés dans les espaces nuls des tâches sélectionnées. A est l'ensemble des tâches qui ont été selectionnées. Avant la première itération, $\mathbf{P}\dot{\mathbf{q}}(t)$ est initialisée avec la trajectoire à analyser. Ensuite, à chaque itération, cette trajectoire est projetée dans l'espace nul de la tâche sélectionnée. r_i représente le score de la fonction de coût de l'optimisation. *activePool* représente l'ensemble des tâches sélectionnées lors du déroulement de l'algorithme.

Dans le cas où le mouvement observé a été exactement généré par la pile de tâches, la trajectoire articulaire $\mathbf{P}\dot{\mathbf{q}}$ après les projections dans l'espace nul de toute les tâches est nulle. Cependant, en présence de bruits (comme par exemple un bruit issu de vrais capteurs lors de l'acquisition du mouvement), un résidu est systématiquement obtenu. Il faut donc utilisé une valeur seuil comme critère d'arrêt : la boucle s'arrête lorsque le résidu des projections passe en dessous du niveau du bruit de la chaine d'acquisition. ϵ représente ce seuil : lorsque la norme du mouvement est inférieure à ce seuil, l'algorithme s'arrête.

1: **Input :** $\hat{\dot{\mathbf{q}}}(t)$
2: **Output :** $activePool$
3: $\mathbf{P}_A\dot{\mathbf{q}}(t) \leftarrow \hat{\dot{\mathbf{q}}}(t)$
4: **while** $\int \|\mathbf{P}_A\dot{\mathbf{q}}(t)\|^2 dt > \epsilon$ **do**
5: **for** task $i = 1..n$ **do**
6: $r_i \leftarrow \text{taskFitting}(i, activePool)$
7: **end for**
8: $i_{select} \leftarrow \text{argmin}(r_i)$
9: $activePool.\text{push}(i_{select})$
10: $\mathbf{P}_A\dot{\mathbf{q}}(t) \leftarrow \text{projection}(i_{select}, \mathbf{P}_A\dot{\mathbf{q}}(t))$
11: **end while**

Algorithme 1 : Algorithme de sélection de tâches

La section suivante décrit les deux fonctions principales de l'algorithme :
- projection$(i, \dot{\mathbf{q}}(t))$ calcule la projection de la vitesse $\dot{\mathbf{q}}(t)$ dans l'espace nul de la tâche i.
- taskFitting$(i, activePool)$ s'occupe de l'ajustement de courbe du mouvement observé et du mouvement théorique et calcule donc un score. Ce processus est détaillé dans la section 4.2.4.

4.2.3 Projection du mouvement

Une reconstruction des trajectoires dans l'espace des tâches à partir des trajectoires angulaires peut être obtenue en multipliant cette trajectoire articulaire par la jacobienne de la tâche calculée en chaque point.

Afin d'annuler les effets d'une tâche détectée i, la trajectoire articulaire est projetée dans l'espace nul de cette tâche en la multipliant par le projecteur dans l'espace nul de toute les tâches à annuler. À chaque instant t du mouvement :

$$\mathbf{P}_A\dot{\mathbf{q}}(t) \leftarrow \mathbf{P}_{A+i}(t)\mathbf{P}_A\dot{\mathbf{q}}(t) \tag{4.1}$$

où A est l'ensemble des tâches déjà sélectionnées et $\mathbf{P}_A(t)$ est mis à jour :

$$\mathbf{P}_{\mathbf{A+i}}(t) = \mathbf{P}_A(t) - (\mathbf{J_i}(t)\mathbf{P}_A(t))^+(\mathbf{J_i}(t)\mathbf{P}_A(t))$$

Le projecteur $\mathbf{P}_A(t)$ est initialisé à $\mathbf{P}_A^0(t) = \mathbf{I}$. Le mouvement restant après projection $\mathbf{P}_A\dot{\mathbf{q}}(t)$ est analysé pour détecter les tâches potentiellement restantes.

L'opération de projection va annuler les effets de l'espace de la tâche sélectionnée dans le mouvement. Dans l'espace articulaire, ces effets sont les suivants : d'une part, la composante de mouvement qui est indépendante des autres tâches est complètement annulée, et d'autre part, la composante de mouvement qui est couplée avec d'autres tâches est modifiée. Le premier effet est bénéfique car il permet d'éviter de fausses détections causées par un mouvement présent dans l'espace de la tâche sélectionnée. Cependant, la partie couplée doit être traitée avec attention à cause de sa modification comme expliqué dans l'exemple suivant :

En considérant un mouvement composé de deux tâches arbitraires $\mathbf{e}_a$ et $\mathbf{e}_b$. La loi de commande est donnée par :

$$\dot{\mathbf{q}} = \mathbf{J}_a^+\dot{\mathbf{e}}_a^* - (\mathbf{J}_b\mathbf{P}_a)^+(\dot{\mathbf{e}}_b^* - \mathbf{J}_b\mathbf{J}_a^+\dot{\mathbf{e}}_a^*) \tag{4.2}$$

Si $\mathbf{e}_a$ est détectée en premier, (4.1) est appliquée avec $i_{select} = a$. Le produit de (4.2) par $\mathbf{P}_a$ annule le mouvement dans l'espace de la tâche a :

$$\mathbf{P}_a\dot{\mathbf{q}} = \underbrace{\mathbf{P}_a\mathbf{J}_a^+\dot{\mathbf{e}}_a^*}_{\mathbf{0}} + \mathbf{P}_a(\mathbf{J}_b\mathbf{P}_a)^+(\dot{\mathbf{e}}_b^* - \mathbf{J}_b\mathbf{J}_a^+\dot{\mathbf{e}}_a^*) \tag{4.3}$$

Le premier terme est nul par définition de $\mathbf{P}_a$. Le mouvement provoqué par $\mathbf{P}_a\dot{\mathbf{q}}$ dans l'espace de la tâche b est obtenue par son produit avec $\mathbf{J}_b$:

$$\mathbf{J}_b\mathbf{P}_a\dot{\mathbf{q}} = \dot{\mathbf{e}}_b^* - \mathbf{J}_b\mathbf{J}_a^+\dot{\mathbf{e}}_a^* \tag{4.4}$$

puisque $\mathbf{J}_b\mathbf{P}_a(\mathbf{J}_b\mathbf{P}_a)^+ = \mathbf{I}$ par hypothèse. Le premier terme est la composante indépendante de la seconde tâche, et le second terme est la composante couplée avec la tâche a. Par conséquent, la projection dans l'espace nul de la tâche découverte à la première itération va faire apparaitre le terme de couplage dont il faudra tenir compte pour la reconnaissance d'autres tâches. Ce couplage sera traité dans la prochaine section.

4.2.4 Ajustement de tâche par optimisation

$\widehat{\dot{\mathbf{e}}}(t)$ représente la trajectoire causée par le mouvement à analyser courant $\mathbf{P}_A\dot{\mathbf{q}}$ dans l'espace de la tâche observée $\mathbf{e}$. La quantification de la pertinence d'une tâche donnée $\mathbf{e}$, par rapport au mouvement, est obtenue en appliquant une optimisation par les moindre carrés entre le véritable mouvement observé projeté dans l'espace de la tâche et le comportement de référence d'une tâche dont les paramètres sont inconnus :

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} \frac{\int \|\widehat{\dot{\mathbf{e}}}(t) - \dot{\mathbf{e}}_{\mathbf{x}}(t)\|^2 \mathrm{dt}}{\int \|\widehat{\dot{\mathbf{e}}}(t)\|^2 \mathrm{dt}} \tag{4.5}$$

où $\widehat{\dot{\mathbf{e}}}(t)$ est la trajectoire observée et $\dot{\mathbf{e}}_{\mathbf{x}}(t)$ est la trajectoire générée par le modèle de tâche en utilisant les paramètres $\mathbf{x}$. Le score est donc le résidu de l'optimisation après avoir essayé d'obtenir la meilleure correspondance entre la trajectoire observée et le modèle théorique dans l'espace des tâches.

Dans quelques cas, la trajectoire $\widehat{\dot{\mathbf{e}}}$ peut être observée directement. Par exemple, si l'espace de tâche est la position 3D d'une main, une observation directe est possible. Cependant, en général, une observation directe n'est pas possible. Typiquement, le centre de masse du robot est difficile à observer avec précision dans le cadre d'une tâche de maintien du centre de masse dans le polygone de sustentation. De plus, il n'est pas possible d'observer directement l'effet d'une succession de projections. $\widehat{\dot{\mathbf{e}}}$ est alors obtenue grâce au modèle géométrique du robot et de sa trajectoire articulaires lors du mouvement : $\mathbf{J}_i\hat{\dot{\mathbf{q}}}$. Cependant, procéder de cette manière amène à (4.4), où le couplage des tâches précédemment sélectionnées apparait. Le mouvement projeté est alors augmenté par un terme qui va compenser ce couplage.

$$\widehat{\dot{\mathbf{e}}}(t) = \mathbf{J}_i\mathbf{P}_A\dot{\mathbf{q}}(t) + \mathbf{J}_i\mathbf{J}_A^+\widehat{\dot{\mathbf{e}}}_A(t) \tag{4.6}$$

où A est l'ensemble des tâches qui a déjà été détecté et i est la tâche candidate. $\mathbf{P}_A$ est le projecteur dans l'espace nul des tâches A, $\mathbf{J}_A$ est la jacobienne des tâches A. Le terme $\mathbf{J}_i\mathbf{J}_A^+\widehat{\dot{\mathbf{e}}}_A(t)$ représente la composante du mouvement de la tâche i couplé aux tâches de l'ensemble A. Tous les termes correspondant à A sont connus puisqu'ils ont été identifiés dans la précédente itération de l'algorithme.

En pratique, l'observation est échantillonnée et l'intégrale dans (4.5) est une somme. Le problème d'optimisation (4.5) est en général un problème non linéaire. Pour le résoudre numériquement, le solveur CFSQP a été utilisé [Lawrence *et al.*, 1997]. Le résultat de l'optimisation produit en même temps le résidu utilisé comme critère de sélection de la tâche la plus probable, et les valeurs numériques associées à la tâche (par exemple le gain et la position désirée, quand il s'agit de tâches proportionnelles).

4.2.5 Invariance à l'ordre d'annulation

La précédente formulation de détection-projection permet d'annuler les effets de bords des tâches précédemment détectées sans introduire de couplage venant des tâches non détectées. Nous montrons maintenant que l'ordre dans lequel les tâches sont détectées et les projections suivantes n'affectent pas la détection des tâches restantes.

Considérons un mouvement observé ayant été généré par une pile de tâches à deux étages où la tâche a est prioritaire par rapport à la tâche b (4.2). À partir du mouvement observé initiallement, noté $\dot{\mathbf{q}}$, on observe une trajectoire dans l'espace de la tâche i. La tâche est ensuite supprimé du mouvement. Le mouvement résultant est $\mathbf{P}_i\dot{\mathbf{q}}$. On observe alors une trajectoire dans l'espace de la tâche j. Nous montrons dans la suite que les observations dans les espaces des tâches sont invariablement les mêmes pour $i = a$ et $j = b$ ou pour $i = b$ et $j = a$. C'est-à-dire que les deux tâches ayant servies à générer le mouvement peuvent être détectées (et supprimées) dans n'importe quel ordre.

Le mouvement de référence de l'espace de tâche i est noté $\dot{\mathbf{e}}_i^*$. Le mouvement observé par la tâche i dans le mouvement initial $\dot{\mathbf{q}}$ est noté $\widehat{\dot{\mathbf{e}}}_i$, tandis que le mouvement observé par la tâche j après avoir supprimé le mouvement issu de la tâche i $\mathbf{P}_j\dot{\mathbf{q}}$ est noté $\widehat{\dot{\mathbf{e}}}_{j|i}$.

Proposition 4.1. *Soit un mouvement généré par une pile de tâches à deux étages. Alors* $\widehat{\dot{\mathbf{e}}}_i = \widehat{\dot{\mathbf{e}}}_{i|j} = \dot{\mathbf{e}}_i^*$*, pour* $i = a$ *et* $j = b$*, et réciproquement, pour* $i = b$ *et* $j = a$*.*

Preuve : La preuve se fait en deux parties, en considérant que a est annulée en premier ($i = a$ et $j = b$), puis que b est annulée en premier ($i = b$ et $j = a$).

Détection de la tâche a, puis de la tâche b :

Cette implication est immédiate. À la première itération, l'observation dans l'espace de tâche a est directement $\widehat{\dot{\mathbf{e}}}_a = \mathbf{J}_a\hat{\dot{\mathbf{q}}} = \dot{\mathbf{e}}_a^*$. Après la suppression de la tâche a, l'observation dans l'espace de la tâche b est obtenue à partir de (4.6) :

$$\begin{aligned}\widehat{\dot{\mathbf{e}}}_{b|a} &= \mathbf{J}_b\hat{\dot{\mathbf{q}}} + \mathbf{J}_b\mathbf{J}_a^+\widehat{\dot{\mathbf{e}}}_a \\ &= \mathbf{J}_b(\mathbf{J}_b\mathbf{P}_a)^+(\dot{\mathbf{e}}_b^* - \mathbf{J}_b\mathbf{J}_a^+\dot{\mathbf{e}}_a^*) + \mathbf{J}_b\mathbf{J}_a^+\dot{\mathbf{e}}_a^*\end{aligned}$$

impliquant directement $\widehat{\dot{\mathbf{e}}}_{b|a} = \dot{\mathbf{e}}_b^*$, puisque $\mathbf{J}_b(\mathbf{J}_b\mathbf{P}_a)^+ = \mathbf{J}_b\mathbf{P}_a(\mathbf{J}_b\mathbf{P}_a)^+ = \mathbf{I}$ et $\widehat{\dot{\mathbf{e}}}_a = \dot{\mathbf{e}}_a^*$.

L'observation obtenu dans l'espace de tâche b est indépendante de la projection dans l'espace nul de la tâche a.

Détection de la tâche b, puis de la tâche a :

À la première itération, l'observation dans l'espace de la tâche b est directement

$$\widehat{\dot{\mathbf{e}}}_b = \mathbf{J}_b\mathbf{J}_a^+\dot{\mathbf{e}}_a^* + \mathbf{J}_b(\mathbf{J}_b\mathbf{P}_a)^+(\dot{\mathbf{e}}_b^* - \mathbf{J}_b\mathbf{J}_a^+\dot{\mathbf{e}}_a^*)$$

ce qui implique $\widehat{\dot{\mathbf{e}}}_b = \dot{\mathbf{e}}_b^*$ puisque $\mathbf{J}_b(\mathbf{J}_b\mathbf{P}_a)^+ = \mathbf{I}$.

La projection du mouvement initial (4.2) dans l'espace nul de la tâche b à la fin de la première itération conduit au calcul de la trajectoire dans l'espace de la tâche a suivant :

$$\begin{aligned}\widehat{\dot{\mathbf{e}}}_{a|b} =& \mathbf{J}_a\mathbf{P}_b\mathbf{J}_a^+\dot{\mathbf{e}}_a^* + \mathbf{J}_a\mathbf{P}_b(\mathbf{J}_b\mathbf{P}_a)^+(\dot{\mathbf{e}}_b^* - \mathbf{J}_b\mathbf{J}_a^+\dot{\mathbf{e}}_a^*) + \mathbf{J}_a\mathbf{J}_b^+\widehat{\dot{\mathbf{e}}}_b \\ =& (\mathbf{J}_a\mathbf{P}_b\mathbf{J}_a^+ - \mathbf{J}_a\mathbf{P}_b(\mathbf{J}_b\mathbf{P}_a)^+\mathbf{J}_b\mathbf{J}_a^+)\dot{\mathbf{e}}_a^* \\ &+ (\mathbf{J}_a\mathbf{P}_b(\mathbf{J}_b\mathbf{P}_a)^+ + \mathbf{J}_a\mathbf{J}_b^+)\dot{\mathbf{e}}_b^*\end{aligned}$$

Nous montrons que cette somme est en réalité simplement égal à $\dot{\mathbf{e}}_a^*$ en montrant d'abord que le coefficient gauche du premier terme $(\mathbf{J}_a\mathbf{P}_b\mathbf{J}_a^+ - \mathbf{J}_a\mathbf{P}_b(\mathbf{J}_b\mathbf{P}_a)^+\mathbf{J}_b\mathbf{J}_a^+)$, qu'on note $\mathbf{C}_1$, est égal à l'identité. Puis en montrant que le second coefficient gauche $(\mathbf{J}_a\mathbf{P}_b(\mathbf{J}_b\mathbf{P}_a)^+ + \mathbf{J}_a\mathbf{J}_b^+)$, qu'on note $\mathbf{C}_2$, est nul. $\mathbf{C}_1$ peut s'écrire en développant l'operateur de projection $\mathbf{P}_b$ en utilisant $\mathbf{P} = \mathbf{I} - \mathbf{J}^+\mathbf{J}$:

$$\begin{aligned}\mathbf{C}_1 =& \mathbf{J}_a\mathbf{J}_a^+ - \mathbf{J}_a\mathbf{J}_b^+\mathbf{J}_b\mathbf{J}_a^+ \\ &- \mathbf{J}_a(\mathbf{J}_b\mathbf{P}_a)^+\mathbf{J}_b\mathbf{J}_a^+ + \mathbf{J}_a\mathbf{J}_b^+\mathbf{J}_b(\mathbf{J}_b\mathbf{P}_a)^+\mathbf{J}_b\mathbf{J}_a^+\end{aligned}$$

Le premier terme est l'identité par hypothèse sur la tâche a, car $(\mathbf{J}_b\mathbf{P}_a)^+ = \mathbf{P}_b(\mathbf{J}_b\mathbf{P}_a)^+$, le troisième terme est nul par définition. Par hypothèse sur les tâches a et b, $(\mathbf{J}_b\mathbf{P}_a)^+$ est l'inverse généralisé de $\mathbf{J}_b$, et donc $\mathbf{J}_b(\mathbf{J}_b\mathbf{P}_a)^+\mathbf{J}_b = \mathbf{J}_b$, le quatrième terme est égal au second et se simplifie. Donc $\mathbf{C}_1 = \mathbf{I}$.

De la même façon pour $\mathbf{C}_2$, le projecteur $\mathbf{P}_b$ est developpé :

$$\mathbf{C}_2 = \mathbf{J}_a(\mathbf{J}_b\mathbf{P}_a)^+ - \mathbf{J}_a\mathbf{J}_b^+\mathbf{J}_b(\mathbf{J}_b\mathbf{P}_a)^+ + \mathbf{J}_a\mathbf{J}_b^+$$

Comme précédemment, le premier terme est nul, $(\mathbf{J}_b\mathbf{P}_a)^+$ est l'inverse généralisé de $\mathbf{J}_b$, le second terme est égal au troisième et se simplifie.

Ceci prouve que le même ajustement est obtenu indépendamment de la tâche qui a été détectée la première. □

La preuve peut être étendue en utilisant les mêmes arguments pour un ensemble de n tâches. D'une façon similaire, il est trivial de prouver que quelque soit l'ordre de détection, le mouvement calculé après toutes les projections est nul.

Pour valider la méthode présentée plus haut, plusieurs expérimentations ont été effectuées sur le robot HRP-2 en simulation et sur le vrai robot. Ces expérimentations consistent à discriminer des paires de mouvements visuellement proches, mais faisant intervenir des tâches différentes : les sémantiques de ces mouvements sont différentes.

4.3 Résultats en simulation : reconnaissance de tâches effectuées par le robot HRP-2

Ce paragraphe détaille une série d'expérimentations effectuées en simulation pour valider l'algorithme de reconnaissance. La simulation nous permet d'observer le comportement nominal de l'algorithme en absence de bruits issus de capteurs. La première expérimentation valide simplement l'étape de projection du mouvement (paragraphe 4.3.2). La seconde partie réunit un ensemble d'expérimentations qui valide l'algorithme de reconnaissance de tâches (paragraphe 4.3.3). Pour chaque expérimentation de cet ensemble, l'algorithme de reconnaissance de tâches est appliqué à deux mouvement visuellement proches : ces deux mouvements ont été artificiellement construits pour présenter une ambiguïté visuelle lorsqu'on les compare entre eux. La présence de cette ambiguïté permet d'illustrer l'efficacité de notre algorithme vis à vis de la précision de la reconnaissance. Tous les mouvements qui ont servis aux expérimentations sont résumés dans le tableau 4.1. La description des tâches utilisées pour construire ces mouvements est détaillée ci-dessous.

4.3.1 Protocole d'expérimentations

Les mouvements de référence ont été générés en utilisant le modèle du robot humanoïde HRP-2. Celui-ci présente 30 degrés de liberté actionnés, plus six degrés de liberté pour la base flottante. Tous les mouvements démarrent de la configuration de repos *half-sitting* (comme illustrée dans la figure 4.2), et le centre de masse est situé à zéro dans le repère du bassin lorsque celui-ci est la racine de l'arbre cinématique. Cette configuration a pour particularité de ne pas présenter de singularité. En cinématique inverse, la position de la base flottante sous-actuée est résolue en contraignant le pied gauche à rester sur le sol. Les figure 4.3, 4.4 et 4.5 montrent les postures finales des mouvements utilisées dans les expérimentations.

L'ensemble des tâches considérées dans les expérimentations sur HRP-2 est :

- *Com* : tâche (2.36), le centre de masse du robot est contraint pour maintenir un équilibre statique (3 DDL)
- *Regard* : tâche (2.38), le robot regarde un point dans l'espace Cartésien, le point de contrôle se situe à 25cm au dessus du centre de la dernière articulation du cou (2 DDL)
- *Double support* : tâche (2.39), la transformation entre le repère attaché au pied droit et le repère attaché au pied gauche du robot doit être constante (6 DDL)
- *Prise gauche/droite* : tâche (2.33), un repère attaché à la main gauche ou droite du robot atteint un point défini dans l'espace Cartésien, les points de contrôle se situent aux centres des articulations terminales des bras (3 DDL)

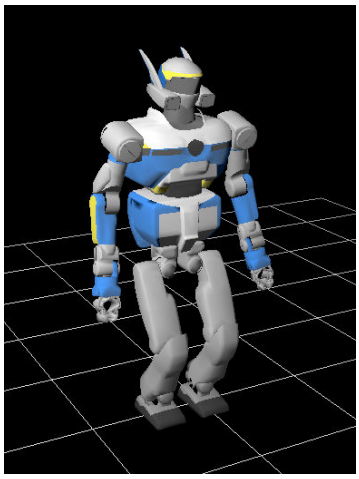


Figure *4.2 – Tout les mouvements de référence démarrent à partir de la posture de repos* half-sitting.

– *Prise orientée* : tâche (2.32), similaire à la tâche de *prise*, mais la position désirée doit être atteinte avec une orientation de la main définie (6 DDL)
– *Tête* : tâche (2.32), un repère attaché à la tête du robot est contrainte en position et en orientation, le point de contrôle se situe au centre de la dernière articulation de la tête (6 DDL)
– *Torse* : tâche (2.34), un repère attaché au torse du robot est contraint en orientation, le point de contrôle se situe au centre de la dernière articulation associée au torse (3 DDL)

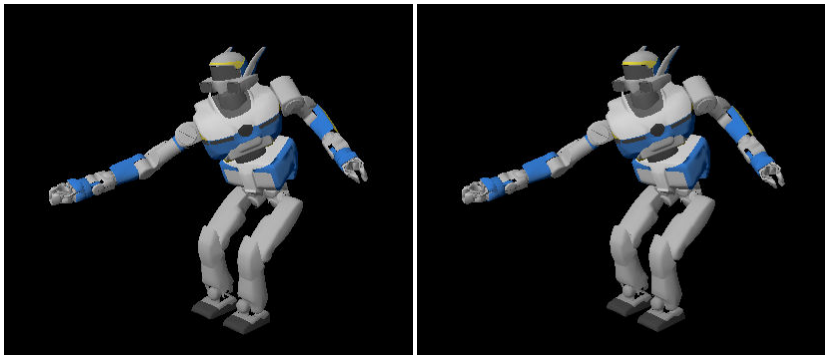


Figure *4.3 – Gauche : La posture finale du* mouvement 2.a *; Droite : La posture finale du* mouvement 2.b.

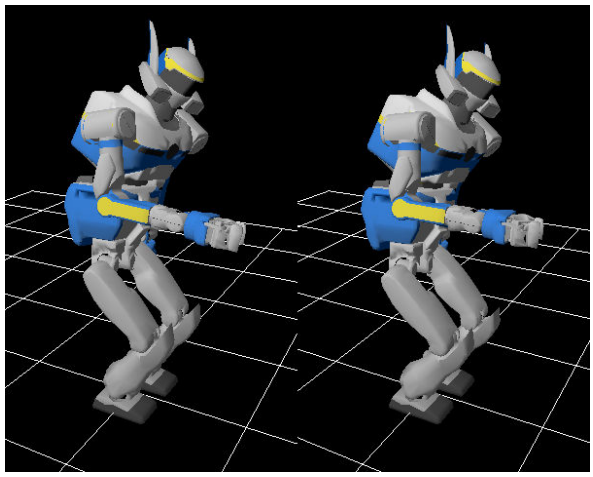


Figure *4.4 – Gauche : La posture finale du* mouvement 3.a *; Droite : La posture finale du* mouvement 3.b. *La différence entre ces deux mouvements est difficile à percevoir : l'orientation de la main droite est contrainte à être parallèle au sol dans le* mouvement 3.b.

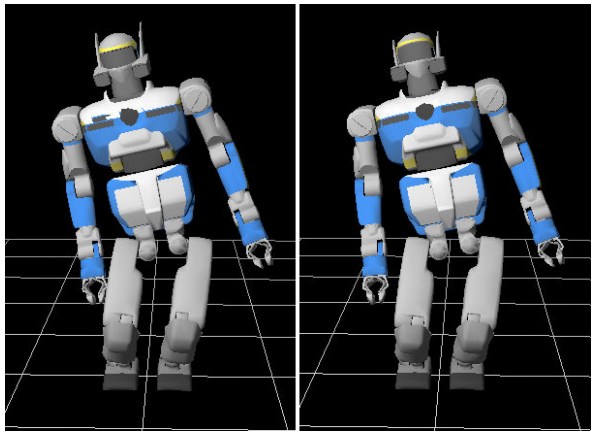


Figure *4.5 – Gauche : La posture finale du* mouvement 4.a *; Droite : La posture finale du* mouvement 4.b.

	(a)	(b)
Mouvement 1	*Prise droite* *Prise gauche* *Regard* *Com* *Double support*	
Mouvement 2	*Prise droite* *Com* *Double support*	*Prise gauche* *Prise droite* *Com* *Double support*
Mouvement 3	*Regard* *Prise droite* *Com* *Double support*	*Regard* *Prise orientée droite* *Com* *Double support*
Mouvement 4	*Regard* *Com* *Double support*	*Prise orienté gauche* *Com* *Double support*
Mouvement 5	*Prise droite* *Regard* *Com* *Double support*	*Prise droite* *Torse* *Com* *Double support*

Table *4.1 – Tableau des mouvements et des tâches considérés.*

4.3.2 Expérimentation 1 : Validation préliminaire

Dans cette expérimentation, un mouvement de base est généré. La reconnaissance de tâche par ajustement et le critère d'arrêt de l'algorithme sont validés ici. Le mouvement de référence est le *mouvement 1.a* (voir le tableau 4.1). Le mouvement est donné au programme de détection qui va sélectionner les tâches qui correspondent le mieux par optimisation.

La figure 4.6 montre des vignettes du mouvement original et du mouvement résultant des projections successives dans les espaces nuls des tâches détectées : *prise droite*, *com*, *regard*, *double support* et *prise gauche*. Le mouvement initial est visualisé en intégrant un champs de vecteurs. Les projections dans les espaces nuls vont modifier ces vecteurs que l'on intègre pour visualiser les mouvements projetés. Ces mouvements projetés n'ont donc pas de sens physique et seule la quantité de mouvements reste valide. Ces quantités de mouvements permettent d'avoir une bonne intuition sur les effets des projections dans les espaces nuls. Chaque projection annule une partie du mouvement, et le mouvement du robot devient nul quand toutes les projections sont appliquées, ce qui signifie que toutes les tâches intervenant dans le mouvement ont été détectées. Comme nous pouvons le constater dans la seconde ligne, le mouvement de la main droite est annulé. L'annulation de *com* à partir de la troisième ligne est plus difficile à percevoir. On peut noter la modification du mouvement des jambes.

Figure *4.6 – Le mouvement généré par une pile de tâches contenant :* Prise droite, Prise gauche, Com, Regard, Double support *est représenté dans la ligne (a). Les autres lignes représentent les projections successives du mouvement dans les espaces nuls des tâches : (b) Prise droite, (c) Com, (d) Regard, (e) Double support, (f) Prise gauche.*

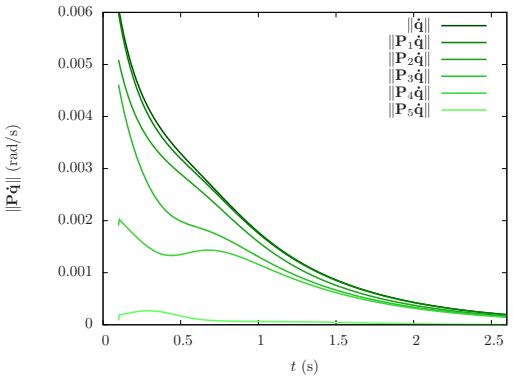


Figure *4.7 – L'évolution de la norme du mouvement après avoir projeté successivement le mouvement dans les espaces nuls des tâches.*

Cette modification correspond au fait que pour exécuter la tâche *Com*, le robot utilise ses jambes. À la quatrième ligne, l'annulation du mouvement de la tête est très claire. On note que le mouvement du torse est altéré. Le mouvement du torse est donc justifié par la tâche de regard. À la cinquième ligne, l'annulation de la tâche *double support* supprime la compensation faite avec le pied droit. Enfin, comme on peut le voir dans la dernière ligne, l'annulation de toutes les tâches mène à un mouvement nul. Tous les mouvements du robot ont donc été justifiés. La figure 4.7 montre l'évolution de la norme du mouvement, définie par la somme des normes au carré sur le temps de la trajectoire articulaire du robot. Chaque projection fait strictement décroître la norme de la vitesse (c'est-à-dire la quantité de mouvement). Après cinq projections, le mouvement est complètement annulé, ce qui confirme que toutes les tâches actives ont été découvertes. L'algorithme s'arrête donc.

4.3.3 Expérimentation 2 : Distinction entre deux mouvements proches

En ce qui concerne la détection de mouvement, être capable de différencier deux mouvements visuellement proches faisant tout de même intervenir des tâches différentes est un problème intéressant. Les algorithmes pour les systèmes anthropomorphes utilisent plutôt le contexte pour la desambigüisation. Le travail présenté ici montre que le critère d'ajustement mêlé à la représentation du mouvement sous forme de pile de tâches est suffisant pour distinguer des mouvements visuellement proches. Trois paires de mouvements ambigus vont être présentées pour illustrer la capacité de distinction de mouvement de notre méthode. Le modèle de comportement de tâche choisi est une décroissance exponentielle dont les paramètres seront les paramètres de l'optimisation du problème 4.5.

4.3.3.1 Mouvements d'atteintes

Dans ce paragraphe, deux mouvements sont considérés : *mouvement 2.a* et *mouvement 2.b*. Le premier est un mouvement d'atteinte lointain avec la main droite. Ce mouvement d'atteinte de la main droite a une influence sur la main gauche par le biais de la tâche *com* : pour retrouver son équilibre, le robot met sa main gauche en arrière. Le second mouvement est la même tâche d'atteinte de la main droite, mais une seconde tâche est ajoutée. Il s'agit d'une tâche d'atteinte de main gauche. La position désirée pour cette dernière tâche a été définie artificiellement comme étant la position finale de la main gauche obtenue lors du premier mouvement.

Les état finaux du robot pour les deux mouvements sont illustrés dans la figure 4.8. Une vidéo montrant les deux mouvements appliqués sur le robot HRP-2 est disponible [1]. Les deux mouvements se ressemblent, et il est très difficile à l'œil nu de dire quel mouvement implique les deux tâches

1. `http://homepages.laas.fr/shak/videos/`

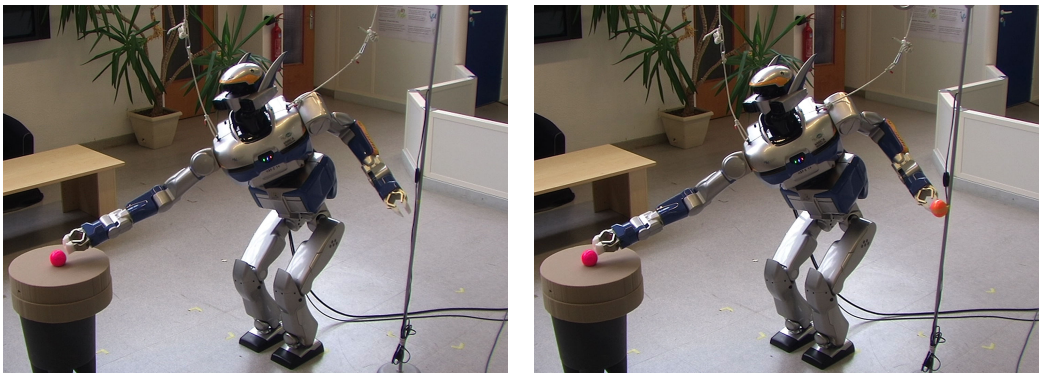


Figure 4.8 – *États finaux des le* mouvement 2.a *et le* mouvement 2.b.

Référence	Détectées	$\int \|\dot{q}(t)\|^2 dt$	$\int \|P\dot{q}(t)\|^2 dt$
Com Prise droite Double support	Com Prise droite Double support	0.364398	0.00159355
Com Prise gauche Prise droite Double support	Com Prise gauche Prise droite Double support	0.538329	0.0035343

Table 4.2 – *Résultats de l'algorithme de sélection de tâches lors de l'analyse du* mouvement 2.a *et du* mouvement 2.b.

d'atteinte sans le contexte. Dans le premier cas, le mouvement de la main gauche est dû à aux tâches *com* et *prise droite*, étant donné que c'est un effet secondaire pour compenser l'équilibre de la main droite. Dans le second cas, le mouvement de la main gauche est découplé, puisque la main gauche a été dirigée par son propre objectif. Cependant, dans l'espace de tâches approprié, ces mouvements apparaissent clairement différents. La figure 4.9 montre un exemple de résultats de l'ajustement de modèles de tâches pour la main droite et la main gauche appliqué au *mouvement 2.a*. Le résidu de l'optimisation est élevé puisque l'ajustement n'est pas possible sur une trajectoire qui ne respecte pas le modèle (en particulier, le résidu de la tâche *prise gauche* est bien plus élevé que celui associé à la tâche *prise droite*). Les résultats de l'algorithme de détection sont résumé dans le Tableau 4.2. La première colonne liste les tâches utilisées dans le mouvement de référence, la seconde colonne liste les tâches sélectionnées par l'algorithme, la troisième indique la norme du mouvement de référence (quantité de mouvement initialement observée), et la dernière colonne montre la norme du mouvement de référence projeté dans l'espace nul des tâches sélectionnées. La quantité de mouvement final est très faible pour les deux mouvements comparé au seuil défini comme critère d'arrêt $\int \|P\dot{q}(t)\|^2 dt > \epsilon$ qui est fixé à $\epsilon = 0.07$.

Enfin la figure 4.10 illustre comment les normes de la vitesse articulaire correspondant aux tâches *prise droite* et *prise gauche* évoluent après les projections dans les espaces nuls. La projection du *mouvement 2.a* dans l'espace nul de la tâche *prise droite* va faire décroître la norme de la vitesse articulaire théorique associée tout en laissant la norme de la vitesse articulaire théorique de la tâche *prise gauche* inchangée. La figure 4.10(a) montre que la vitesse articulaire théorique associée à la tâche *prise gauche* est diminuée après la projection du mouvement dans l'espace nul de la tâche *com*. Ceci explique que le bras gauche du robot a été déplacé par la tâche *com*. De plus, cette projection empêche la tâche *prise gauche* d'être détectée par l'algorithme dans des itérations futures à cause de mouvements parasites. Cependant, après la projection du *mouvement 2.b* dans l'espace nul de la tâche *prise droite* et la tâche *com*, la norme de la trajectoire de la vitesse articulaire associée à la tâche *prise gauche* reste significative (figure 4.10(b)). Ceci signifie que la tâche *com* n'a que peu d'influence

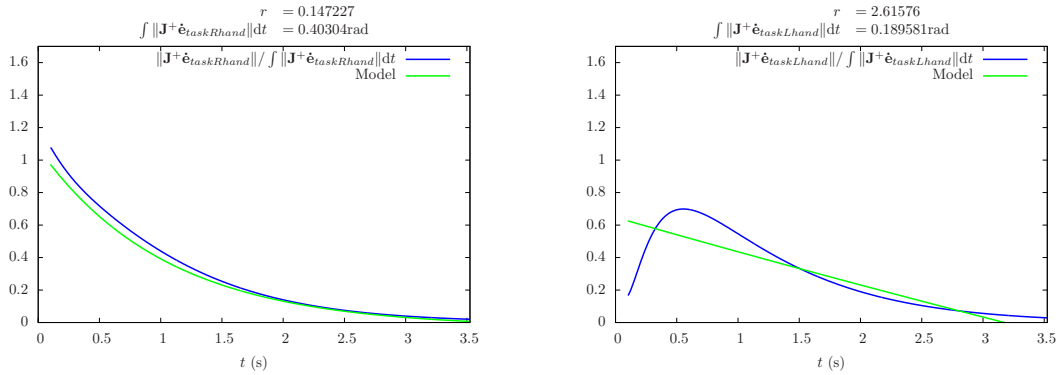
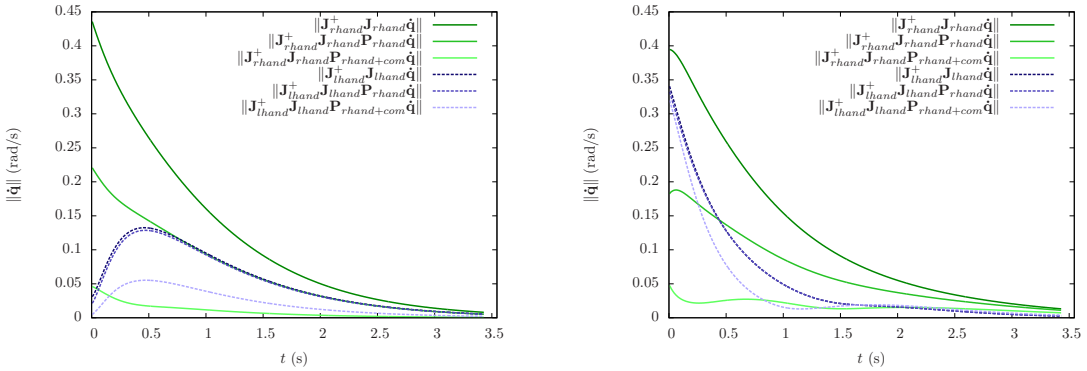


Figure *4.9 – L'ajustement de modèle de tâche sur le* mouvement 2.a *sur les tâches de la main droite et gauche. La variable r est le résidu, c'est-à-dire la distance entre deux courbes. Le mouvement de la main droite est correctement ajusté par le modèle de la tâche, montrant que la tâche est active. Le mouvement de la main gauche n'est pas ajusté correctement, puisque la tâche est inactive. Les deux cas sont facilement distinguables grâce à la valeur du résidu.*



(a) Mouvement 2.a (b) Mouvement 2.b

Figure *4.10 – Evolution de la trajectoire $\dot{\mathbf{q}}$ projetée dans les espaces des tâches* prise droite *(en ligne pleine) et* prise gauche *(en pointillés) après les projections successives du mouvement dans l'espace nul des tâches* prise droite *et l'espace nul de la tâche* com. *Dans le* mouvement 2.a, *une grande partie du mouvement du bras gauche est dû à la tâche* com *: la suppression de la tâche* com *va annuler presque tout le mouvement du bras gauche. Dans le* mouvement 2.b, *le mouvement du bras gauche n'est pas seulement issu de la tâche* com, *mais principalement de la tâche* prise gauche. *La suppression de la tâche* com *va seulement annuler une petite partie du mouvement du bras gauche.*

sur le mouvement du bras gauche, et que le mouvement de ce bras est dû à une autre tâche. Par conséquent, l'algorithme de sélection de tâche va continuer à chercher la tâche qui a contrôlée le bras gauche. Après la détection des deux tâches principales (pour le *mouvement 2.a*) et des trois tâches principales (pour le *mouvement 2.b*), la norme de la dernière trajectoire de la vitesse articulaire n'est pas nulle parce que l'algorithme de sélection de tâche n'est pas terminé et d'autres tâches n'ont pas encore été sélectionnées. La tâche de *double support* est alors détectée mais les courbes associées ne sont pas tracées par souci de clarté, car elles sont quasi-nulles. D'autre part, la figure 4.11 montre l'ajustement de modèle de tâche pour le *mouvement 2.b*.

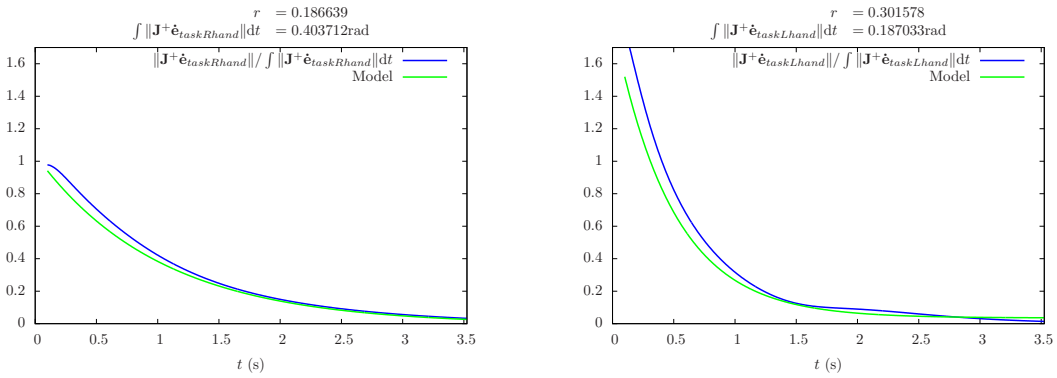


Figure *4.11 – L'ajustement de modèle de tâche pour le* mouvement 2.b *pour les tâches de* prise droite *et* prise gauche. *Les deux mouvements de la main gauche et de la main droite sont correctement ajustés par le modèle, avec un petit résidu : les tâches sont bien détectées.*

4.3.3.2 Prise VS Prise orientée

Dans ce paragraphe, le *mouvement 3.a* et le *mouvement 3.b* sont analysés. Les deux mouvements partagent la même position de but pour la main droite. La seule différence entre les deux démonstrations est la présence d'une contrainte d'orientation sur la main droite dans le *mouvement 3.b*. Les postures finales de ces mouvements sont illustrées dans la figure 4.4. Le tableau 4.3 résume les résultats de l'algorithme de sélection de tâche qui s'est déroulé avec succès. Lors de l'analyse du *mouvement 3.b*, la tâche de *prise* a également été sélectionnée. Ce qui est parfaitement logique car dans ce cas précis, la tâche *prise* est une sous-tâche de *prise orientée* : la tâche *prise* est identique à la tâche *prise orientée* sans la contrainte d'orientation. Par contre, lors de l'analyse du *mouvement 3.a*, la tâche *prise orientée* n'est pas sélectionnée. La figure 4.12 montre que l'ajustement de la tâche *prise orientée* échoue pour le *mouvement 3.a* mais réussi pour le *mouvement 3.b*. Comme précédemment, les projections annulent les mouvements résiduels qui pourraient mener à de futures détections erronées. Le mouvement résiduel après les projections successives sont finalement proches de zéro, ce qui prouve que toutes les tâches ont été détectées.

4.3.3.3 Prise orientée VS Regard

Les deux mouvements considérés sont le *mouvement 4.a* et le *mouvement 4.b*. Le *mouvement 4.a* peut être décrit par le scénario suivant : un objet se trouve devant le robot, et crée une occlusion dans le champ de vision du robot. Pour se débarrasser de cette occlusion, le robot se penche sur sa droite. Lorsque le robot se penche, la main gauche est entrainée par le torse par un effet secondaire involontaire. La position et l'orientation de cette main sont enregistrées comme étant l'état désiré pour la tâche de prise orientée dans le *mouvement 4.b*. Dans le *mouvement 4.b*, le regard n'est pas contrôlé. Le mouvement de la tête est cette fois-ci, un effet secondaire du mouvement de la main gauche. Les postures finales du robot pour ces deux mouvements sont illustrées dans la figure 4.5, et les résultats des analyses sont résumés dans le Tableau 4.4.

Comme précédemment, les bonnes tâches ont été détectées pour chaque mouvement. Le résidu après les projections de toutes les tâches détectées étant très bas, toutes les tâches ont été correctement détectées et soustraites au mouvement.

Pour conclure, nous venons de montrer expérimentalement que, sans bruits, l'algorithme de détection se comporte parfaitement, c'est-à-dire que toute les tâches actives dans le mouvement analysé sont détectées, sans faux positifs, et la suppression des effets de ces tâches, via les projections du mouvement original dans les espaces nuls, conduit à un mouvement quasi nul. En effet, le mouvement n'est pas totalement annulé à cause du bruit numérique. Dans le paragraphe suivant nous réalisons des expérimentations dans un cadre réaliste en utilisant un vrai robot et de véritables capteurs qui fourniront des signaux bruités, correspondant aux trajectoires articulaires du robot.

4.4 Expérimentations sur le robot

Dans ce paragraphe, nous démontrons la validité de l'algorithme de reconnaissance de tâche dans un cadre réaliste en analysant des signaux bruités. Cette fois, le mouvement de référence est joué sur un véritable robot HRP-2 et est observé en utilisant un système de capture de mouvement (figure 4.13). Comme dans les expérimentations en simulations, l'algorithme de sélection de tâches est appliqué sur des paires de mouvements visuellements proches. Dans la suite de ce paragraphe, nous détaillons le processus d'acquisition des trajectoires articulaires par le biais du système de capture de mouvements. Puis, deux paires de mouvements seront analysées par l'algorithme de reconnaissance de tâches.

Référence	Détectées	$\int \|\dot{q}(t)\|^2 dt$	$\int \|P\dot{q}(t)\|^2 dt$
Com Regard Prise Double support	Com Regard Prise Double support	0.619266	0.00245631
Com Regard Prise orientée Double support	Com Regard Prise Prise orientée Double support	0.717041	0.00344557

Table *4.3 – Résultats de l'algorithme de sélection de tâche pour l'analyse du* mouvement 3.a *et du* mouvement 3.b.

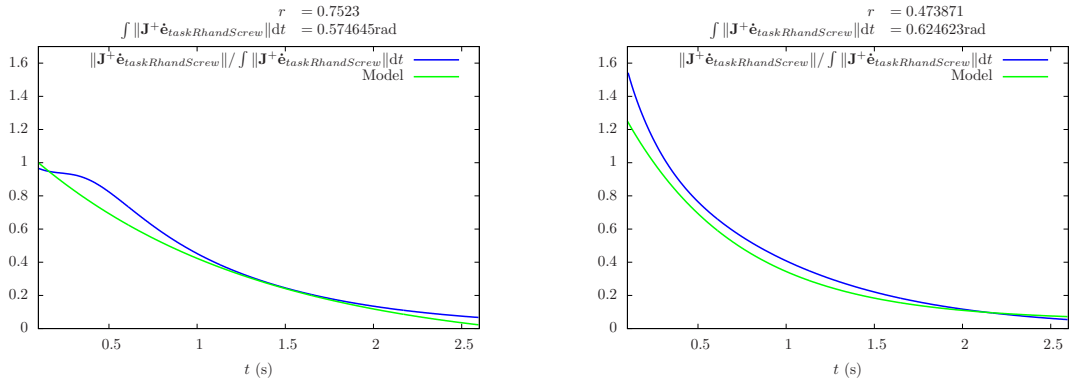


Figure *4.12 – L'ajustement de modèle de tâche* prise orientée *échoue, car le résidu est plus élevé, pour le* mouvement 3.a *et réussi pour le* mouvement 3.b.

4.4.1 Protocol expérimental

Un mouvement est généré par la pile de tâches, puis est exécuté sur le robot HRP-2 équipé de marqueurs sur chacun de ses corps (voir figure 4.13). Le système de capture de mouvements utilisé est composé de 10 caméras infrarouges, et enregistre les données à une fréquence de 200Hz. Les données collectées à partir de ces marqueurs sont utilisés pour construire un squelette virtuel qui est ajusté à la hiérarchie cinématique du robot (figure 4.13). Le système de capture de mouvements fournit une trajectoire pour chacun des corps du robot. Les données sont obtenues sous forme de trajectoires 6D de chaque corps du robot dans l'espace, sans aucune contrainte articulaire.

L'analyse du mouvement est effectuée sur la trajectoire articulaire. Par conséquent, les trajectoires articulaires doivent être calculées à partir des données issues de la capture de mouvements. Les trajectoires articulaires sont calculées de manière classique, par minimisation de la distance entre les matrices de transformations ${}^W\mathbf{R}_{r_j}(\mathbf{q})$ associées à l'origine des articulations du robot et des matrices de transformations mesurées ${}^W\hat{\mathbf{R}}_{m_j}(t)$ par le système de capture de mouvement, en tenant compte des limites articulaires du robot.

$$\hat{\mathbf{q}}(t) = \arg\min_{\mathbf{q}} \sum_{j}^{m} \| {}^W\hat{\mathbf{R}}_{r_j}(t) \ominus {}^W\mathbf{R}_{r_j}(\mathbf{q}) \|^2 \tag{4.7}$$

$$\text{s.t.} \quad q_{i\min} \leq q_i \leq q_{i\max},\ i = 1..n \tag{4.8}$$

où m est le nombre de corps, n le nombre d'articulations, $\mathbf{q}$ est le vecteur de configuration du robot, $\ominus$ est l'opérateur de distance dans SO(3), ${}^W\mathbf{R}_{r_j}(\mathbf{q})$ est calculée en utilisant le modèle cinématique du

Référence	Détectée	$\int \|\dot{q}(t)\|^2 dt$	$\int \|P\dot{q}(t)\|^2 dt$
Com Regard Double support	Com Regard Double support	0.534478	0.0545944
Com Prise orientée Double support	Com Prise orientée Double support	0.558084	0.0023297

Table *4.4 – Résultats de l'algorithme de sélection de tâches à partir de l'analyse du* mouvement 4.a *et du* mouvement 4.b.

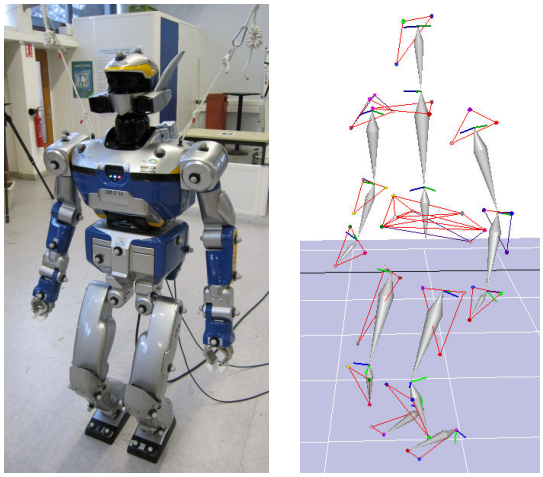


Figure *4.13 – Ensemble des marqueurs, et squelette virtuel pour le robot HRP-2.*

robot, et ${}^{W}\hat{\mathbf{R}}_{m_i}$ est obtenue par :

$$ {}^{W}\hat{\mathbf{R}}_{r_j}(t) = {}^{W}\mathbf{R}_{W_C} \times {}^{W_C}\hat{\mathbf{R}}_{m_j}(t) \times {}^{m_j}\mathbf{R}_{r_j} \tag{4.9} $$

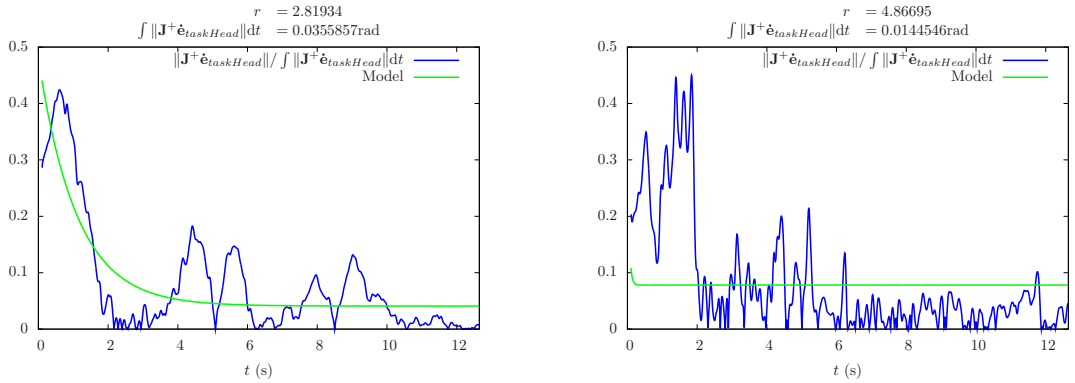
où ${}^{W}\mathbf{R}_{W_C}$ est la matrice de transformation entre le repère du monde et le repère de la zone d'expérimentation de la capture de mouvements. ${}^{m_j}\mathbf{R}_{r_j}$ sont des transformations constantes dû à la différence entre un modèle de squelette quelconque de la capture de mouvement et du modèle cinématique du robot utilisé lors de la phase de calibration. Cette phase de calibration consiste à calculer les transformations ${}^{W}\mathbf{R}_{W_C}$ et ${}^{m_j}\mathbf{R}_{r_j}$ à partir d'une posture connue. ${}^{W_C}\hat{\mathbf{R}}_{m_j}$ est la matrice de transformation mesurée associée à l'origine du corps virtuel i dans le repère de référence du système de capture de mouvements.

Les trajectoires articulaires obtenues sont utilisées pour effectuer la reconnaissance de tâches par retro-ingénierie.

Sur le véritable robot, une flexibilité est introduite après l'articulation de la cheville. Elle interfère avec le mouvement au tout début car l'accélération angulaire au niveau des articulations est importante. La flexibilité n'est pas modélisée, ni dans l'algorithme de contrôle, ni dans la méthode de détection. Pour ne pas être influencé par la flexibilité, les 100 premières millisecondes des mouvements analysés sont ignorées. C'est dans cette phase que les effets de la flexibilité sont les plus importants dûs à l'accélération initiale typique d'une tâche de régulation proportionnelle en cinématique inverse. Dans les deux prochains paragraphes, nous considérons que les mesures sont directement les trajectoires $\hat{\mathbf{q}}$ données par (4.8).

Référence	Détectée	$\int \|\|\dot{q}(t)\|\|^2 dt$	$\int \|\|P\dot{q}(t)\|\|^2 dt$
Com Prise droite Double support	Com Prise droite Double support	0.104835	0.0482885
Com Prise gauche Prise droite Double support	Com Prise gauche Prise droite Double support	0.142293	0.0541836

Table *4.5 – Résultats de l'algorithme de sélection de tâches pour l'analyse du* mouvement 2.a *et du* mouvement 2.b *joués sur le vrai robot.*



(a) Mouvement 2.a (b) Mouvement 2.b

Figure *4.14 – Ajustement des modèles de tâche à la première itération de l'algorithme de sélection de tâches pour les tâches* tête *dans le* mouvement 2.a *et* mouvement 2.b *Pour les deux mouvements, r est élevé : la tâche* tête*, qui n'est pas active, n'est pas sélectionnée.*

4.4.2 Prise VS Maintien de l'équilibre

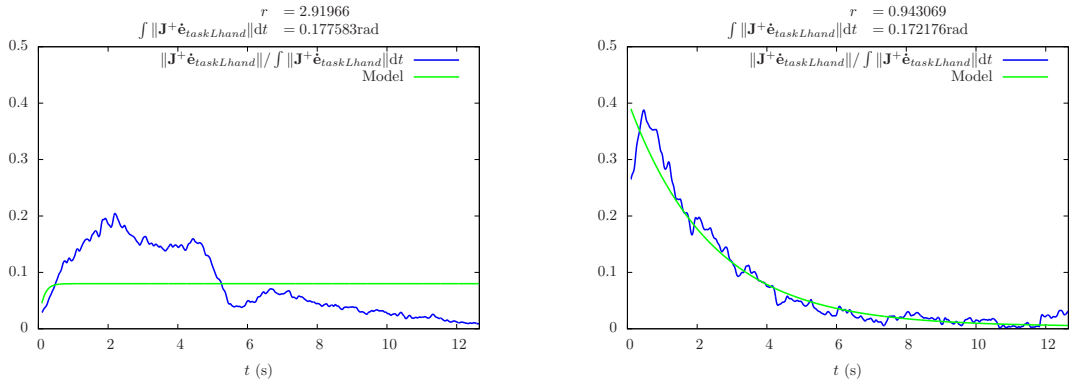
Cette expérience correspond à celle effectuée en simulation dans le paragraphe 4.3.3.1 : le premier mouvement est un mouvement de la main gauche produit par le couplage entre la prise lointaine de la main droite, et la tâche *com* (*mouvement 2.a*). Le second mouvement correspondant à une double prise, a été construit pour présenter une ambiguïté avec le premier mouvement (*mouvement 2.b* dans le tableau 4.1). Le tableau 4.5 présente les résultats de l'algorithme de reconnaissance pour le *mouvement 2.a* et le *mouvement 2.b* avec le critère d'arrêt égal à $\epsilon = 0.07$.

En ce qui concerne le *mouvement 2.a*, l'ordre d'extraction des tâches est : *prise droite*, *double support* et *com*. Tandis que pour le *mouvement 2.b*, les tâches extraites sont : *prise droite*, *com*, *double support* et *prise gauche*.

Les figure 4.14 et 4.15 montrent l'ajustement du modèle de tâche à la première itération de l'algorithme pour les tâches *tête* et *prise gauche*. Comme prévu, l'ajustement est correct seulement pour la tâche *prise gauche* dans le *mouvement 2.b* tandis que cette tâche est rejetée dans le *mouvement 2.a* à cause de son ajustement incorrect.

Bien que la tâche *tête* ne soit impliquée dans aucun des mouvements, la tête n'est pas fixe dans l'espace à cause du mouvement du torse et des jambes causés par les tâches actives. Dans le *mouvement 2.b*, la tâche candidate *tête* impliquerait moins de mouvement que dans le *mouvement 2.a*. La raison est que lorsque les deux mains opposées du robot sont contraintes, le torse n'a pas beaucoup de liberté. Cependant, la tâche *tête* n'est pas gardée pour aucun des deux mouvements (le résidu r est élevé).

L'évolution du mouvement projeté dans les espaces de la tâche *prise gauche* et *prise droite* après



(a) Mouvement 2.a (b) Mouvement 2.b

Figure *4.15 – Ajustement des modèles de tâche à la première itération de l'algorithme de sélection de tâches pour la tâche* prise gauche *dans le* mouvement 2.a *et* mouvement 2.b *La tâche* prise gauche *n'est pas pertinente dans le* mouvement 2.a*, mais est sélectionnée dans le* mouvement 2.b*.*

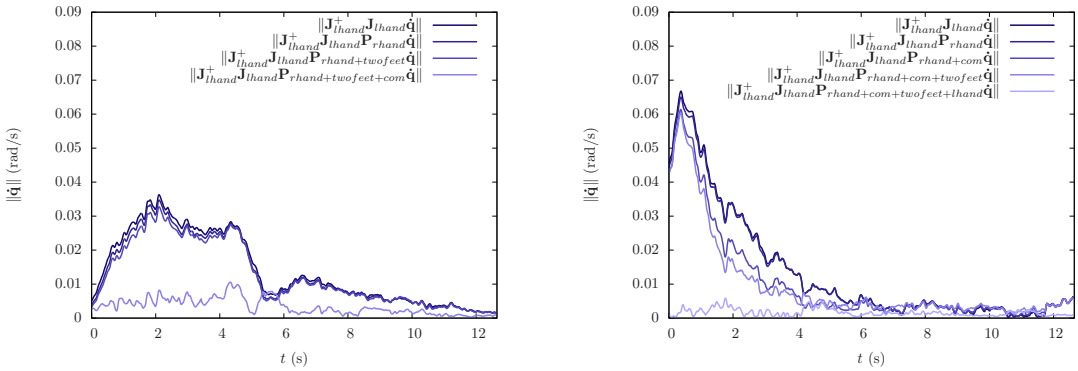
Référence	Détectée	$\int \|\dot{q}(t)\|^2 dt$	$\int \|P\dot{q}(t)\|^2 dt$
Com Regard Prise droite Double support	Com Regard Prise droite Double support	0.320001	0.0785168
Torse Com Prise droite Double support	Torse Com Prise droite Double support	0.216742	0.0516134

Table *4.6 – Résultats de l'algorithme de sélection de tâches à partir de l'analyse des* mouvement 5.a *et* mouvement 5.b *joués sur le vrai robot.*

les projections successives dans les espaces nuls automatiquement sélectionnées est illustrée dans la figure 4.16. Pour le *mouvement 2.a*, le mouvement du bras gauche est annulé après avoir annulé la tâche *com*. Dans le *mouvement 2.b*, le mouvement de la main gauche n'est pas annulé par la projection dans l'espace nul de la tâche *com*, mais devient nul après projection dans l'espace nul de la tâche *prise gauche*. La figure 4.17 montre comment évolue le mouvement reconstruit de la main droite lorsque le mouvement dû à une tâche est enlevé. On peut voir que le bras droit bouge à cause des tâches *prise droite* et *com*. La figure 4.18 montre l'évolution de la quantité de mouvement après chaque sélection de tâche dans les deux mouvements. La quantité décroit au fur et à mesure des projections, jusqu'à ce que le mouvement restant soit principalement du bruit issu des capteurs.

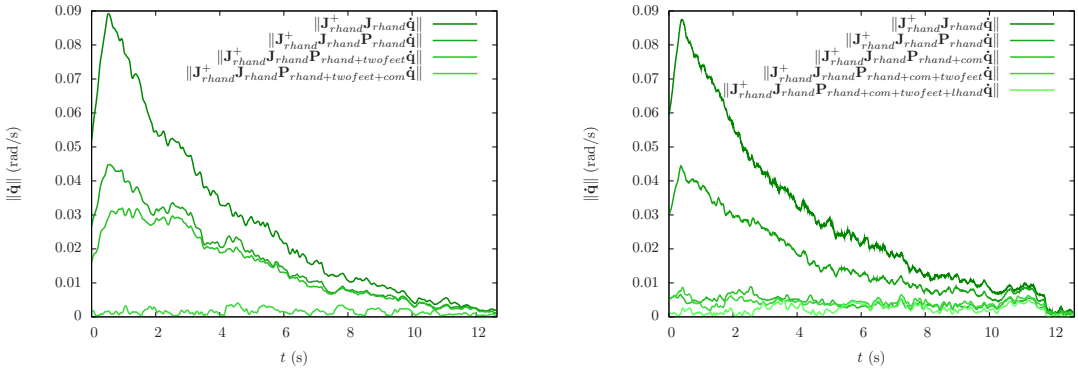
4.4.3 Regard VS torse

Les mouvements considérés sont les *mouvement 5.a* et *mouvement 5.b*. Dans le premier cas, le robot regarde sa main droite tout en effectuant une prise avec celle-ci. Dans le second cas, le robot effectue une prise identique, mais doit maintenir son torse dans la posture initiale. La figure 4.19 montre la posture finale du robot pour ces deux mouvements. Le tableau 4.6 résume les résultats de l'algorithme de sélection de tâches. Malgré le bruit, les tâches sont correctement détectées. Les deux mouvements sont différenciés correctement. Enfin, le résidu très faible après l'annulation de toutes les tâches indique qu'il n'y a pas d'autres tâches à reconnaitre. La figure 4.20 montre l'évolution de la norme du mouvement après les projections dans les espaces nuls des tâches : *Prise droite*, *regard*, *double support* and *com* pour le *mouvement 5.a*, et les tâches *prise droite*, *double support*, *com* et *torse*



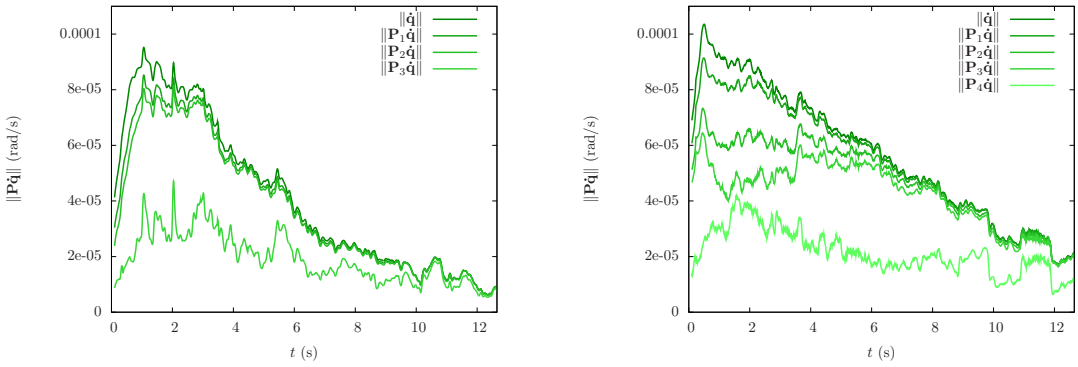
(a) Mouvement 2.a (b) Mouvement 2.b

Figure *4.16 – Norme du mouvement associés aux tâches* prise gauche *(c'est-à-diremouvement reconstruit de la main gauche). Le mouvement est nul après avoir retiré la tâche* com *dans le* mouvement 2.a. *Dans ce cas, cela signifie que le mouvement de la main gauche provenait de la tâche* com. *Tandis que dans le* mouvement 2.b, *le mouvement de la main gauche n'est pas annulé après avoir retiré la tâche* com. *Le mouvement devient nul uniquement après avoir retiré la tâche* prise gauche *: le mouvement de la main gauche provenait de la tâche* prise gauche.



(a) Mouvement 2.a (b) Mouvement 2.b

Figure *4.17 – Mouvement reconstruit de la main droite après les projections successives. Le mouvement de la main droite est une conséquence des tâches* prise droite *et* com, *comme c'est expliqué par le fait que le mouvement de la main droite est annulé après la suppression des tâches* prise droite *et* com.



(a) Mouvement 2.a (b) Mouvement 2.b

Figure *4.18 – Evolution de la norme du mouvement après projections successives du mouvement dans les espaces nuls des tâches pour le* mouvement 2.a *et* mouvement 2.b.

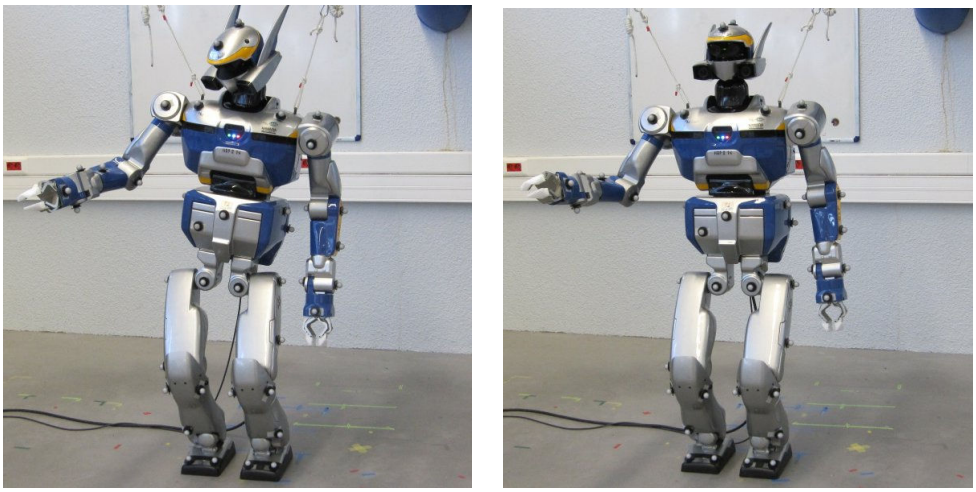
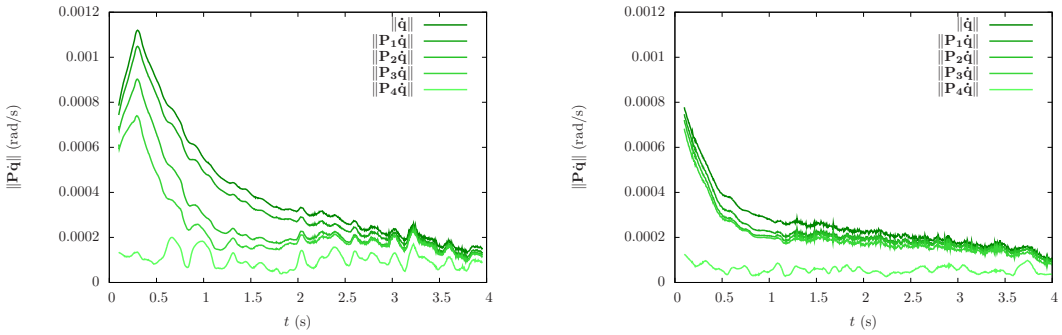


Figure *4.19 – Posture finale pour le* mouvement 5.a *et le* mouvement 5.b.



(a) Mouvement 5.a

(b) Mouvement 5.b

Figure *4.20 – Évolution de la norme du mouvement après avoir projeté avec succès le mouvement dans les espaces nuls des tâches pour les mouvements* mouvement 5.a *et* mouvement 5.b.

pour le *mouvement 5.b*. Comme dans les précédentes expérimentations, la projection fait strictement décroitre la quantité de mouvement à chaque itération de l'algorithme. La quantité de mouvement restant à la fin de l'algorithme est clairement dû au bruit.

4.5 Expérimentation sur l'humain

La méthode de reconnaissance de tâches présentée dans ce chapitre donne de très bon résultats lorsque l'on connaît à la fois le modèle des tâches mis en œuvre, et le modèle géométrique du robot pour calculer les différentes opération de projections. La reconnaissance de tâches pour le robot humanoïde se basait sur la connaissance parfaite du comportement du robot lors de l'exécution d'une tâche. Nous avions appliqué comme exemple de comportement, une décroissance exponentielle. On souhaite maintenant étudier la possibilité d'appliquer la méthode de reconnaisssance sur des mouvements humains en adaptant les modèles de comportement de tâche et en utilisant des modèles de génération de mouvement humains. En effet, la méthode n'est pas dépendante du comportement des tâches à retrouver. Deux points sont donc à vérifier, correspondant à la sélection de tâches d'une part (paragraphe 4.2.4), et d'autre part l'annulation du mouvement provoqué par l'exécution d'une

tâche (paragraphe 4.2.3). Dans le cadre de cette thèse, seul le premier point a été validé en se plaçant dans le cas particulier du mouvement d'atteinte, dont le modèle de comportement chez l'humain est bien connu. Il s'agit du modèle du minimum jerk qui sera utilisé pour caractériser la trajectoire de l'effecteur atteignant sa cible. La généralisation à d'autres types de tâches ainsi que le second point sont laissés en perspectives.

4.5.1 Comportement d'une tâche d'atteinte par un humain

Nous nous focalisons sur un type de tâche particulier : la tâche d'atteinte. En effet, des études ont mis en évidence le profil de trajectoires humaines pour les mouvement d'atteinte. Celles-ci respectent un profil de type *minimum jerk* [Flash et Hogan, 1985]. Cette connaissance de profil de trajectoire correspond bien à l'hypothèse requise pour l'application de notre méthode de reconnaissance, qui est la connaissance d'un modèle de comportement lors de l'exécution d'une tâche. La trajectoire de la partie du corps d'un humain effectuant ce mouvement d'atteinte suit un profil du type minimum jerk (de la même manière que dans la partie précédente, la trajectoire du robot correspondait à une décroissance exponentielle). La trajectoire en position reliant deux points A et B est dite de jerk minimum si elle correspond à l'optimum du problème :

$$\min \iiint_0^\infty \|\dddot{f}(t)\| dt \tag{4.10}$$

$$f(0) = A \tag{4.11}$$

$$f(\infty) = B \tag{4.12}$$

4.5.2 Modélisation du minimum jerk

Plutôt que de raisonner sur la forme implicite du problème (4.12), il est souvent choisi dans la littérature de représenter les trajectoires en minimum jerk comme une classe spécique de polynômes du troisième ordre, dont on sait que le jerk est faible. Nous choisissons de modéliser le jerk non pas comme un polynôme de degré 3, mais comme l'intégrale triple d'un signal à quatre créneaux correspondant respectivement aux phases d'accélération linéaires, de décélération, d'accélération et enfin d'accélération nulle. Ainsi notre modèle comportera 6 paramètres : les trois valeurs crêtes, et les trois instants relatifs de commutations. Ce choix est motivé par le fait que la méthode de reconnaissance utilise une méthode d'optimisation numérique pour la phase d'ajustement entre les trajectoires reconstruites (par observation) et le modèle de comportement. Utiliser les coefficients d'un polynôme comme paramètre d'optimisations ne permet pas de restreindre l'espace de recherche aux courbes de type minimum jerk, et par conséquent, modéliser une tâche par un polynôme n'est pas efficace. Au contraire, la modélisation en signal créneaux permet d'assurer la minimalité de la fonction de jerk sans passer par des contraintes artificielles sur les coefficients du ploynôme représentant la trajectoire. La trajectoire en position est alors obtenue par triple intégration du jerk.

Le jerk est alors défini ainsi :

$$\text{jerk}(t) = \begin{cases} K_1 & if \quad 0 < t < \Delta t_1 \\ K_2 & if \quad t_1 < t < \Delta t_1 + \Delta t_2 \\ K_3 & if \quad t_2 < t < \Delta t_1 + \Delta t_2 + \Delta t_3 \\ 0 & if \quad t > \Delta t_1 + \Delta t_2 + \Delta t_3 \end{cases} \tag{4.13}$$

La figure 4.21 illustre notre modélisation par un exemple de minimum jerk, et sa trajectoire en position correspondante.

Les mouvements considérés étant des trajectoires d'atteinte, le mouvement de la main se termine avec une position constante de celle-ci. Donc à $t = \Delta t_1 + \Delta t_2 + \Delta t_3$ la vitesse et l'accélération sont nulles. Ce qui amène à un système de deux équations avec comme inconnues Δt_3 et K_3, ce qui contraint la durée et le comportement de la phase finale pour atteindre une vitesse et accélération

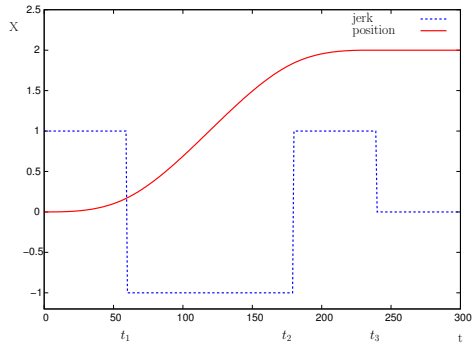


Figure *4.21 – Un exemple de minimum jerk et sa trajectoire en position correspondante.*

nulle. Le problème d'optimisation pour l'ajustement de courbe devient :

$$\begin{aligned} & \min_{\mathbf{x}} \int \|\hat{\mathbf{p}}(t) - \mathbf{p}_{\mathbf{x}}(t)\|^2 + K_1^2 + K_2^2 \mathrm{dt} \\ \text{s.t.} \quad & \mathbf{p}_{\mathbf{x}}(t) = \iiint \mathrm{jerk}(t) \mathrm{dt} \end{aligned} \tag{4.14}$$

où $\mathbf{x} = [dt_1\ dt_2\ K_1\ K_2]$, $\hat{\mathbf{p}}$ est la trajectoire mesurée. La seconde partie de la fonction de coût $K_1^2 + K_2^2$ est introduite pour limiter les paramètres K_1 et K_2, afin que les trajectoires obtenues par intégration du jerk soient d'un ordre de grandeur raisonnable. Comme dans la section 4.2.4, on utilise le solveur FSQP avec une forme discrétisée de (4.14), la distance entre $\mathbf{p}_{\mathbf{x}}$ et $\hat{\mathbf{p}}$.

4.5.3 Validation du modèle choisi

Dans cette expérimentation, nous validons l'algorithme présenté dans le paragraphe 4.2.4 pour la détection de trajectoire d'atteinte chez l'humain. Un humain est équipé d'un ensemble de 23 marqueurs qui vont permettre au système de capture de mouvement d'enregistrer des démonstrations. Le sujet doit atteindre le haut d'une bouteille en y posant son doigt, tout en gardant un pied au sol et en gardant la position de la main secondaire constante. La position finale de la main est donc clairement définie et constante, et son orientation n'est pas contrainte. La bouteille est suffisamment éloignée du sujet pour l'obliger à lever un pied pour atteindre la bouteille. Lors de cette expérimentation, la trajectoire de la main qui atteint la bouteille est étudiée. La figure 4.22 illustre la position finale du mouvement.

La trajectoire de la main dominante obtenue par capture de mouvement est donnée à un programme d'optimisation pour réaliser l'ajustement (4.14) de la trajectoire observée sur notre modèle théorique (4.13). La figure 4.23, montre que l'optimisation arrive à faire correspondre la trajectoire observée au modèle proposé.

4.5.4 Analogie avec la pile de tâches

L'hypothèse des *variétés non contrôlées* définit une séparation entre les espaces contrôlés et les espaces non-contrôlés [Scholz et Schöner, 1999]. Ainsi une grande variabilité dans l'espace non-contrôlé (*UCM*) ne va pas dégrader les performances d'une tâche, tandis qu'une grande variabilité dans son sous-espace orthogonal va dégrader ces performances.

Dans le cas particulier des tâches d'atteintes considérées dans les différentes expérimentations menées dans [Jacquier-Bret *et al.*, 2009], la présence de contraintes géométriques matérialisées par un obstacle entraîne une grande variabilité au niveau de l'espace non-contrôlé du coude : pour éviter l'obstacle, un humain va modifier la trajectoire du coude. Une approximation de cet espace non-contrôlé est réalisée en utilisant le noyau de la jacobienne. Ainsi, cet espace représente l'espace nul de la tâche d'atteinte. Les expérimentations menées dans ces travaux montrent qu'il est cohérent d'utiliser

Figure *4.22 – Position finale du mouvement d'atteinte humain.*

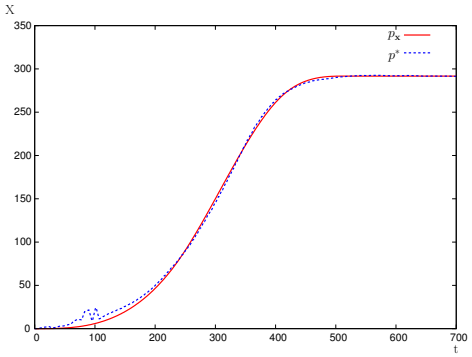


Figure *4.23 – Ajustement du modèle de minimum jerk sur une trajectoire d'atteinte réelle.*

les mécanismes liés à la redondance, bien connus dans le cas robotique, pour expliquer la gestion de la redondance dans les mouvements humains.

Conclusion

Ce chapitre a présenté une méthode pour identifier quelles tâches sont impliquées dans un mouvement observé, sans utiliser d'informations contextuelles : seule la trajectoire observée est analysée. L'analyse est conduite par la connaissance du comportement d'un robot lors de l'exécution d'une tâche (par exemple, une décroissance exponentielle). Le mouvement analysé est supposé être généré par un ensemble de contrôleurs appartenant à un lot de tâches connu. Le problème de la reconnaissance de tâches est traité en procédant par une rétro-ingénierie du mouvement. La trajectoire observée est analysée dans chaque espace de tâches connu pour décider quelles tâches sont actives en comparant ces trajectoires aux comportements théoriques attendu. La méthode est généralisable sur les comportements d'une tâche, et par conséquent, n'importe quelle loi de commande utilisée pour générer un mouvement peut être utilisée dans cette méthode pour caractériser une tâche.

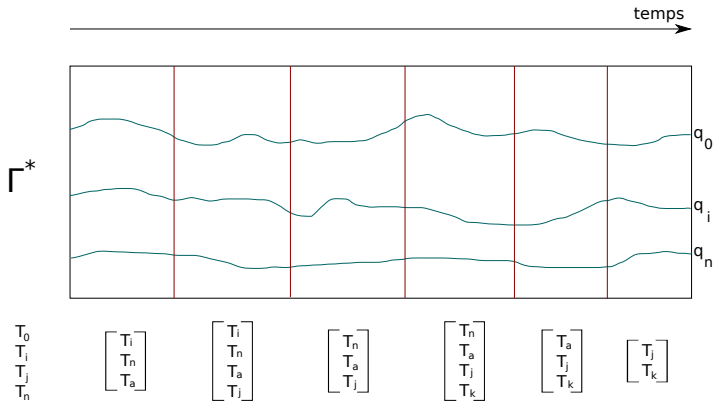


Figure *4.24 – Segmentation temporelle d'une trajectoire en séquence de piles de tâches.*

La méthode a été appliquée avec succès dans différent scénarios pour discriminer des mouvements visuellement proches, en simulation (paragraphe 4.3) et sur un véritable robot HRP-2 (paragraphe 4.4). Ces mouvements ont été construits spécialement pour présenter des ambiguïtés, dans le but d'illustrer l'efficacité de la méthode proposée.

Dans toutes ces expériences, une hypothèse forte considérée était que le mouvement analysé ne doit pas comporter de tâches qui changent d'état (active, non-active). Une solution intéressante à explorer serait d'ajouter une dimension temporelle à la reconnaissance de tâche : il s'agirait de déterminer la durée de validité d'une pile de tâches, ainsi un mouvement serait segmenté temporellement en succession d'instances de piles de tâches comme montré dans la Fig. 4.24. La segmentation temporelle s'effectuera en introduisant les temps de début et de fin de tâches dans les paramètres d'optimisation.

Dans l'expérience de le paragraphe 4.5, un modèle de tâche adapté aux mouvements humain pour des tâches d'atteintes est vérifié. Par extension, il serait ainsi possible d'appliquer notre méthode de reconnaissance de tâches pour ce type de mouvement. Le problème étant que les mouvements humains sont beaucoup plus complexes à décrire que les mouvements des robots. Les étapes futures s'attacheront à trouver dans la littérature du domaine des modèles pour d'autres tâches que le mouvement d'atteinte. D'autre part, pour les mouvements d'atteinte, nous chercherons à valider le modèle du minimum jerk comme étant un critère discriminant (c'est-à-dire dans le cas du scénario 2.a, si la main gauche ne suit pas une telle loi). Une solution pour généraliser à des modèles de tâches divers serait de s'appuyer sur des méthodes d'apprentissage. On pourrait ainsi extraire un modèle à partir d'une série de mouvement, par exemple en s'appuyant sur des processus gaussien. L'utilisation de ces techniques ont montrées des résultats prometteurs dans le domaine des mouvements humains [Wang *et al.*, 2008]. Les travaux présentés dans [Alvarez *et al.*, 2009] illustrent une approche hybride de modélisation de mouvement par apprentissage et modèle physique : les données de haute dimension représentant des mouvements humains sont résumé par des variables latentes ayant une interprétation physique. Le modèle serait ensuite re-utilisé en projetant le mouvement observé dans l'espace de la variable apprise. La distance au modèle appris serait alors utilisée comme une mesure d'activation de la tâche.

Dans notre méthode de reconnaissance, l'étape des annulations des tâches par projections dans les espaces nuls est primordiale. Outre le fait d'étudier la validité du modèle de minimum jerk pour caractériser l'exécution ou la non-exécution d'une tâche, une des voies à explorer est d'étudier si un modèle cinématique approché d'un humain, pour calculer les projections, peut être suffisant pour pouvoir déterminer si toute les tâches ont été détectées. L'utilisation d'un modèle plus réaliste et complet, tel un modèle de génération de mouvement dynamique [Saab *et al.*, 2011], pourra être étudié. En effet, en considérant un modèle dynamique, les inerties des corps seront prises en compte dans le mécanisme de projection. Enfin, la validité de la pile de tâches sera étudier pour des mouvements

humain, c'est-à-dire caractériser si les mécanismes de fusion d'objectifs multiples (ou comment est gérée la redondance) chez l'humain peut être représenter sous une forme de pile de tâches.

Conclusion et perspectives

Les travaux présentés dans cette thèse ont portés sur la reconnaissance de tâches qu'effectuent un système anthropomorphe. Nous avons d'abord présenté les fondements sur lesquels reposent nos travaux : le formalisme de la fonction de tâches. Ce formalisme permet d'exprimer intuitivement des commandes pour un robot en décrivant l'objectif dans un espace approprié. Nous avons également donné des exemples typiques de tâches robotiques exprimées dans ce formalisme. La génération de mouvements est effectuée en construisant une pile de tâches rangées par ordre de priorités. Les tâches de haute priorité ne sont pas perturbées par des tâches de priorités inférieures.

Les techniques utilisées pour acquérir des mouvements ont été présentées. Ces techniques sont applicables aussi bien sur un robot que sur un humain. Aussi nous avons donné un exemple d'une méthode générique pour reproduire un comportement humain sur un robot en utilisant les mécanismes de contrôle de robot présentés précedemment.

Enfin, nous avons montré qu'en se plaçant directement dans l'espace dans lequel une tâche est exprimée, il est possible d'utiliser le modèle de génération de mouvement comme un modèle de prédiction en résolvant un problème d'optimisation. Le résidu du problème d'optimisation produit une mesure d'adéquation entre un mouvement observé et une loi de génération de mouvements. Ainsi en supposant que les modèles de générations de mouvements et que les tâches pouvant intervenir dans un mouvement sont connus et qu'aucune tâche ne change d'état (active, inactive) durant le mouvement observé, il est possible de reconstruire la pile de tâches ayant généré ce mouvement.

Contributions

Capture de mouvements

Dans le cadre de l'acquisition de mouvements, il était nécessaire d'observer, de manière externe au robot, les trajectoires articulaires du robot à partir d'un système de capture de mouvements. Ainsi, un programme de recallage de modèle par optimisation a été développé. Le mouvement du squelette défini par la configuration géométrique des points suivis par la capture de mouvements est transféré au modèle du robot HRP-2. Ce programme a aussi été utilisé dans les travaux d'éditions de mouvements dynamiques de l'équipe dans lesquels les mouvements sont montrés par un humain. Les mouvements obtenus sont transposés au robot pour donner une première trajectoire articulaire. Cette trajectoire articulaire est ensuite modifiée en utilisant des tâches afin de corriger les erreurs introduites par le changement de modèle et d'adapter la dynamique du mouvement aux contraintes physiques du robot.

Génération de mouvements d'attente

Pour la génération de mouvements d'attente, un module logiciel a été créé pour assigner les lots de trajectoires mesurées par capture de mouvements aux trajectoires de références des tâches. Les

mouvements générés ont été utilisés pour *donner une impression de vie* au robot HRP-2 durant une démonstration publique du robot.

Reconnaissance de tâches

Une méthode de reconnaissance de mouvements capable de détecter des tâches effectuées en parallèle et possédant éventuellement des couplages a été développée. Cette méthode s'appuie sur des opérateurs de projection dans les espaces des tâches et leurs espaces orthogonaux pour d'une part projeter les mouvements à analyser dans des espaces caractéristiques d'une tâche et d'autre part pour annuler et découpler les tâches détectées. La méthode de la reconnaissance de tâches a été implémentée en utilisant les différentes librairies développées en collaboration par l'équipe Gepetto au LAAS-CNRS Toulouse et le JRL Tsukuba.

Perspectives

En ce qui concerne la reconnaissance de tâches, les perspectives envisagées sont réliées aux limites de la méthode. La reconnaissance ne s'applique que sur des segments temporels dans lesquels les tâches mises en jeu ne changent pas d'état. Ainsi il est envisageable d'ajouter des paramètres de temporels représentant le début et la fin d'une tâche dans le problème d'optimisation des paramètres du modèle de génération de mouvements pour détecter ces changements d'états. L'étape qui suit logiquement est la reconnaissance de séquence de pile de tâches. Un apprentissage des transitions entre les différentes instances de pile de tâches pourra être réalisé afin de construire des graphes de mouvements offrant une grande réactivité pour la reconnaissance et la reproduction.

D'autre part, le formalisme de la fonction de tâches s'étend très bien en dynamique : notre méthode de reconnaissance pourra être appliquée à des tâches dynamiques. Toutes les propriétés de la fonction de tâches sont conservées. Seuls les modèles de génération et les espaces d'observation diffèrent. Il semble donc immédiat d'adapter l'approche à la dynamique.

Nous avons donné une définition d'une tâche robotique. Cette définition est accompagnée d'outils et de propriétés puissantes qui nous ont permis de projeter des trajectoires dans des espaces spécialisés où la reconnaissance est plus facile. Mais est-il possible d'utiliser les mêmes concepts sur un humain ou de définir d'une manière similaire des tâches humaines ?

L'étude du mouvement d'attente nous a amené à réfléchir sur ce qui caractérise un mouvement ou un comportement. La méthode utilisée pour reproduire le mouvement d'attente est très pragmatique : les trajectoires associées à certains membres sont reproduites. L'évaluation du succès d'une imitation d'action est immédiate : si l'action à reproduire consiste à déplacer un objet à un endroit donné, alors la position finale de l'objet après l'imitation permet de valider le mouvement effectué. En revanche il n'y a aucun critère objectif permettant de valider le succès de l'imitation du mouvement d'attente. Lorsqu'un humain est debout et attend, des mouvements inconscients apparaissent : le regard est mobile, la respiration entraîne des mouvements au niveau du torse. Il n'est pas naturel de rester immobile. L'exemple des statues vivantes le montre bien, puisque de gros efforts physiques et de concentration sont nécessaires pour contrôler sa respiration, retirer toutes les tensions inutiles, maintenir une posture constante et se décaler subtilement lorsque l'attention du spectateur baisse pour se dégourdir. Dans les deux cas, on peut considérer que l'humain *ne fait rien*. Cette étude de ce qu'est un modèle générique d'action trouverait une simplification immédiate en introduisant des méthodes d'apprentissage automatique pour synthétiser plusieurs démonstrations successives en un seul modèle numérique.

Bibliographie

[PR2, 2011] (2011). Pr2, willow garage. `http://www.willowgarage.com/pages/pr2/overview`.

[Acht *et al.*, 2007] Acht, V. V., Bongers, E., Lambert, N. et Verbene, R. (2007). Miniature wireless inertial sensor for measuring human motions. *In IEEE International Conference on Engineering in Medicine and Biology Society*, Lyon, France.

[Alvarez *et al.*, 2009] Alvarez, M., Luengo, D. et Lawrence, N. D. (2009). Latent force models. *In International Conference on Artificial Intelligence and Statistics*, pages 9–16, Clearwater Beach, États Unis.

[Baerlocher et Boulic, 1998] Baerlocher, P. et Boulic, R. (1998). Task-priority formulations for the kinematic control of highly redundant articulated structures. *In IEEE/RAS Intelligent Robots and Systems (IROS)*, Victoria, Canada.

[Baerlocher et Boulic, 2004] Baerlocher, P. et Boulic, R. (2004). An inverse kinematic architecture enforcing an arbitrary number of strict priority levels. *The Visual Computer : International Journal of Computer Graphics*, 20(6):402–417.

[Barbič *et al.*, 2004] Barbič, J., Safonova, A., Pan, J.-Y., Faloutsos, C., Hodgins, J. K. et Pollard, N. S. (2004). Segmenting motion capture data into distinct behaviors. *In Graphics Interface*, pages 185–194, Londres, Canada.

[Baudouin *et al.*, 2011] Baudouin, L., Perrin, N., Moulard, T., Lamiraux, F., Stasse, O. et Yoshida, E. (2011). Real-time replanning using 3d environment for humanoid robot. *In International Conference on Humanoid Robots (Humanoids)*, Bled, Slovénie.

[Beaudoin *et al.*, 2008] Beaudoin, P., Coros, S., van de Panne, M. et Poulin, P. (2008). Motion-motif graphs. *In Symposium on Computer Animation*, pages 117–126, Dublin, Irlande.

[Ben-Israel et Greville, 2003] Ben-Israel, A. et Greville, T. N. E. (2003). *Generalized Inverses : Theory and Applications*. Springer Verlag, seconde édition.

[Benallègue *et al.*, 2010] Benallègue, M., Wieber, P. B., Kheddar, A. et Espiau, B. (2010). A computational model for synchronous motion imitation by humans : The mirror controller applied on stepping motions. *In International Conference on Humanoid Robots (Humanoids)*, Nashville, États-Unis.

[Billard *et al.*, 2006] Billard, A., Calinon, S. et Guenter, F. (2006). Discriminative and Adaptive Imitation in Uni-Manual and Bi-Manual Tasks. *Robotics and Autonomous Systems*, 54(5):370–384.

[Billard *et al.*, 2003] Billard, A., Epars, Y., Cheng, G. et Schaal, S. (2003). Discovering imitation strategies through categorization of multi-dimensional data. *In IEEE/RAS Intelligent Robots and Systems (IROS)*, Las Vegas, États-Unis.

[Bodenheimer *et al.*, 1997] Bodenheimer, B., Rose, C., Rosenthal, S. et Pella, J. (1997). The process of motion capture : Dealing with the data. *In Computer Animation and Simulation*, pages 3–18, Budapest, Hongrie.

[Boulic *et al.*, 1996] Boulic, R., Mas, R. et Thalmann, D. (1996). A robust approach for the control of the center of mass with inverse kinetics. *Computers and Graphics*, 20(5):693–701.

[Bruderlin et Williams, 1995] Bruderlin, A. et Williams, L. (1995). Motion signal processing. *In ACM Conference on Computer graphics and interactive techniques (SIGGRAPH)*, pages 97–104, Los Angeles, États-Unis.

[Burtnyk et Wein, 1976] Burtnyk, N. et Wein, M. (1976). Interactive skeleton techniques for enhancing motion dynamics in key frame animation. *Communications of the ACM*, 19(10):564–569.

[Calinon *et al.*, 2010] Calinon, S., D'Halluin, F., Sauser, E., Caldwell, D. et Billard, A. (2010). Learning and reproduction of gestures by imitation : An approach based on hidden markov model and gaussian mixture regression. *IEEE Robotics and Automation Magazine*, 17(2):44–54.

[Calinon *et al.*, 2007] Calinon, S., Guenter, F. et Billard, A. (2007). On learning, representing and generalizing a task in a humanoid robot. *IEEE Transactions on Systems, Man and Cybernetics, Part B*, 37(2):286–298.

[Campbell et Bobick, 1995] Campbell, L. W. et Bobick, A. F. (1995). Recognition of human body motion using phase space constraints. *In International Conference on Computer Vision*, pages 624–630, Cambridge, États-Unis.

[Chalodhorn *et al.*, 2009] Chalodhorn, R., MacDorman, K. et Asada, M. (2009). Humanoid robot motion recognition and reproduction. *Advanced Robotics*, 23:349–366.

[Chan *et al.*, 2007] Chan, C., Liu, H. et Brown, D. (2007). An effective human motion classification approach using knowledge representation in qualitative normalised templates. *In International Conference on Fuzzy Systems (FUZZ-IEEE)*, Orlando, États-Unis.

[Chaumette et Hutchinson, 2006] Chaumette, F. et Hutchinson, S. (2006). Visual servo control, part i : Basic approaches. *IEEE Robotics and Automation Magazine*, 13(4):82–90.

[Chaumette et Hutchinson, 2007] Chaumette, F. et Hutchinson, S. (2007). Visual servo control, part ii : Advanced approaches. *IEEE Robotics and Automation Magazine*, 14(1):109–118.

[Chiaverini, 1997] Chiaverini, S. (1997). Singularity-robust task-priority redundancy resolution for real-time kinematic control of robot manipulators. *IEEE Transactions on Robotics and Automation*, 13(3):398 – 410.

[Craig, 2004] Craig, J. J. (2004). *Introduction to Robotics : Mechanics and Control (3rd Edition)*. Prentice Hall.

[Degallier *et al.*, 2008] Degallier, S., Righetti, L., Natale, L., Nori, F., Metta, G. et Ijspeert, A. (2008). A modular bio-inspired architecture for movement generation for the infant-like robot icub. *In International Conference on Biomedical Robotics and Biomechatronics (BIOROB)*, Scottsdale, États-Unis.

[Deng *et al.*, 2006] Deng, Z., Chiang, P. Y., Fox, P. et Neumann, U. (2006). Animating blendshape faces by cross-mapping motion capture data. *In Symposium on Interactive 3D graphics and games*, pages 43–48, Redwood City, États-Unis.

[Drumwright *et al.*, 2004] Drumwright, E., Jenkins, O. C. et Matarić, M. (2004). Exemplar-based primitives for humanoid movement classification and control. *In International Conference on Robotics and Automation (ICRA)*, New Orleans, États-Unis.

[Drumwright et Matarić, 2003] Drumwright, E. et Matarić, M. (2003). Generating and recognizing free-space movements in humanoid robots. *In IEEE/RAS Intelligent Robots and Systems (IROS)*, Las Vegas, États-Unis.

[D'Souza *et al.*, 2001] D'Souza, A., Vijayakumar, S. et Schaal, S. (2001). Learning inverse kinematics. *In IEEE/RAS Intelligent Robots and Systems (IROS)*, Maui, États-Unis.

[Eppner *et al.*, 2009] Eppner, C., Sturm, J., Bennewitz, M., Stachniss, C. et Burgard, W. (2009). Imitation learning with generalized task descriptions. *In International Conference on Robotics and Automation (ICRA)*, pages 3968–3974, Kobe, Japan.

[Escande *et al.*, 2006] Escande, A., Kheddar, A. et Miossec, S. (2006). Planning support contact-points for humanoid robots and experiments on hrp-2. *In IEEE/RAS Intelligent Robots and Systems (IROS)*, pages 2974 – 2979, Pékin, Chine.

[Espiau *et al.*, 1992] Espiau, B., Chaumette, F. et Rives, P. (1992). A new approach to visual servoing in robotics. *IEEE Transactions on Robotics and Automation*, 8(3):313–326.

[Field *et al.*, 2008] Field, M., Stirling, D., Naghdy, F. et Pan, Z. (2008). Motion segmentation for humanoid control planning. *In Australasian Conference on Robotics and Automation (ACRA)*, Canberra, Australie.

[Flash et Hogan, 1985] Flash, T. et Hogan, N. (1985). The coordination of arm movements : An experimentally confirmed mathematical model. *Journal of Neuroscience*, 5(7):1688–1703.

[Gleicher, 1998] Gleicher, M. (1998). Retargeting motion to new characters. *In ACM Conference on Computer graphics and interactive techniques (SIGGRAPH)*, pages 33–42, New York, États-Unis.

[Gleicher, 2000] Gleicher, M. (2000). Animation from observation : Motion capture and motion editing. *Computers and Graphics*, 33(4):51–54.

[Gleicher, 2008] Gleicher, M. (2008). *More Motion Capture in Games : Can We Make Example-Based Approaches Scale ?* Lecture Notes in Computer Science. Springer.

[Gleicher et Ferrier, 2002] Gleicher, M. et Ferrier, N. (2002). Evaluating video-based motion capture. *In Computer Animation*, pages 75–80, Genève , Suisse. IEEE Computer Society.

[Gribovskaya *et al.*, 2011] Gribovskaya, E., Kheddar, A. et Billard, A. (2011). Motion Learning and Adaptive Impedance for Robot Control during Physical Interaction with Humans. *In International Conference on Robotics and Automation (ICRA)*, pages 4326–4332, Shangai, Chine.

[Gu *et al.*, 2010] Gu, J., Ding, X., Wang, S. et Wu, Y. (2010). Action and gait recognition from recovered 3-d human joints. *IEEE Transactions on Systems, Man and Cybernetics, Part B*, 40:1021–1033.

[H. Liu et Ha, 2011] H. Liu, X. W. et Ha, I. (2011). Realtime human motion control with a small number of inertial sensors. *In ACM Symposium on Interactive 3D Graphics and Games*, pages 133–140, San Francisco, États-Unis.

[Hersch *et al.*, 2008] Hersch, M., Guenter, F., Calinon, S. et Billard, A. (2008). Dynamical System Modulation for Robot Learning via Kinesthetic Demonstrations. *IEEE Transactions on Robotics*, 24(6):1463–1467.

[Herzog *et al.*, 2008] Herzog, D., Ude, A. et Krüger, V. (2008). Motion imitation and recognition using parametric hidden markov models. *In International Conference on Humanoid Robots (Humanoids)*, Daejeon, Corée.

[Inamura *et al.*, 2004] Inamura, T., Nakamura, Y. et Toshima, I. (2004). Embodied symbol emergence based on mimesis theory. *International Journal of Robotics Research*, 23(4):363–377.

[Jacquier-Bret *et al.*, 2009] Jacquier-Bret, J., Rezzoug, N. et Gorce, P. (2009). Adaptation of joint flexibility during a reach-to-grasp movement. *Motor Control*, 13(3):342–361.

[Jenkins et Matarić, 2004] Jenkins, O. C. et Matarić, M. (2004). Performance-derived behavior vocabularies : Data-driven acquisition of skills from motion. *International Journal of Humanoid Robotics*, 1(2):237–288.

[Kaelbling *et al.*, 1998] Kaelbling, L., Littman, M. et Cassandra, A. (1998). Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101:99–134.

[Kanoun *et al.*, 2010] Kanoun, O., Laumond, J. P. et Yoshida, E. (2010). Planning foot placements for a humanoid robot : a problem of inverse kinematics. *International Journal of Robotics Research*, 30(4):476–485.

[Keith *et al.*, 2009] Keith, F., Mansard, N., Miossec, S. et Kheddar, A. (2009). Optimization of tasks warping and scheduling for smooth sequencing of robotic actions. *In IEEE/RAS Intelligent Robots and Systems (IROS)*, Saint Louis, États-Unis.

[Kovar et Gleicher, 2003] Kovar, L. et Gleicher, M. (2003). Flexible automatic motion blending with registration curves. *In ACM Eurographics Symposium on Computer Animation*, Aire-la-Ville, Suisse.

[Krüger *et al.*, 2007] Krüger, V., Kragic, D., Ude, A. et Geib, C. (2007). The meaning of action : A review on action recognition and mapping. *Advanced Robotics*, 21(13):1473–1501.

[Kulić et Nakamura, 2008a] Kulić, D. et Nakamura, Y. (2008a). Incremental learning and memory consolidation of whole body motion patterns. *In International Conference on Epigenetic Robotics (EPIROB)*, Falmer, Royaume-Uni.

[Kulić et Nakamura, 2008b] Kulić, D. et Nakamura, Y. (2008b). Scaffolding on-line segmentation of fully body human motion patterns. *In IEEE/RAS Intelligent Robots and Systems (IROS)*, Nice, France.

[Kwon et Park, 2008] Kwon, J. et Park, F. C. (2008). Natural movement generation using hidden markov models and principal components. *IEEE Transactions on Systems, Man and Cybernetics, Part B*, 38(5):1184 – 1194.

[Laptev, 2005] Laptev, I. (2005). On space-time interest points. *International Journal of Computer Vision*, 64(2):107–123.

[Lasseter, 1987] Lasseter, J. (1987). Principles of traditional animation applied to 3d computer animation. *Computer Graphics*, 21(4):35–44.

[Lawrence *et al.*, 1997] Lawrence, C., Zhou, J. L. et Tits, A. L. (1997). User's guide for CFSQP version 2.5 : A C code for solving (large scale) constrained nonlinear (minimax) optimization problems, generating iterates satisfying all inequality constraints. Rapport technique TR-94-16RL, Institute for Systems Research, University of Maryland.

[Liang *et al.*, 2009] Liang, Y. M., Shih, S. W., Shih, A. C. C., Liao, H. Y. M. et Lin, C. C. (2009). Learning atomic human actions using variable-length markov models. *IEEE Transactions on Systems, Man and Cybernetics, Part B*, 39(1):268 – 280.

[Liu et Sarkar, 2005] Liu, Z. et Sarkar, S. (2005). Effect of silhouette quality on hard problems in gait recognition. *IEEE Transactions on Systems, Man and Cybernetics, Part B*, 35:170–183.

[Luenberger, 1984] Luenberger, D. G. (1984). *Linear and nonlinear programming*. Addison-Wesley.

[Mansard et Chaumette, 2007] Mansard, N. et Chaumette, F. (2007). Task sequencing for high level sensor-based control. *IEEE Transactions on Robotics*, 23(1):60–72.

[Mansard *et al.*, 2007] Mansard, N., Stasse, O., Chaumette, F. et Yokoi, K. (2007). Visually-guided grasping while walking on a humanoid robot. *In International Conference on Robotics and Automation (ICRA)*, pages 3041–3047, Rome, Italie.

[Miura *et al.*, 2009] Miura, K., Morisawa, M. et Nakaoka, S. (2009). Robot motion remix based on motion capture data towards human-like locomotion of humanoid robots. *In International Conference on Humanoid Robots (Humanoids)*, pages 596 – 603, Paris, France.

[Moeslund *et al.*, 2006] Moeslund, T. B., Hilton, A. et Krüger, V. (2006). A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding*, 104(2):90–126.

[Montecillo-Puente *et al.*, 2010] Montecillo-Puente, F. J., Sreenivasa, M. N. et Laumond, J. P. (2010). On real-time whole-body human to humanoid motion transfer. *In International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, pages 22–31.

[Moore, 1920] Moore, E. H. (1920). On the reciprocal of the general algebraic matrix. *Bulletin of the American Mathematical Society*, 26:394–395.

[Mori *et al.*, 2002] Mori, T., K.Tsujioka et Sato, M. S. T. (2002). Human-like action recognition system using features extracted by human. *In IEEE/RAS Intelligent Robots and Systems (IROS)*, pages 1214–1220, Lausanne, Suisse.

[Mühlig *et al.*, 2009] Mühlig, M., Gienger, M., Steil, J. et Goerick, C. (2009). Automatic selection of task spaces for imitation learning. *In IEEE/RAS Intelligent Robots and Systems (IROS)*, St Louis, États-Unis.

[Nakamura, 1990] Nakamura, Y. (1990). *Advanced Robotics : Redundancy and Optimization*. Addison-Wesley Longman Publishing, première édition.

[Nakamura et Hanafusa, 1986] Nakamura, Y. et Hanafusa, H. (1986). Inverse kinematic solutions with singularity robustness for robot manipulator control. *Journal of Dynamic Systems, Measurement and Control*, 108(3):163–171.

[Nakamura *et al.*, 1987] Nakamura, Y., Hanafusa, H. et Yoshikawa, T. (1987). Task-priority based re-dundancy control of robot manipulators. *International Journal of Robotics Research*, 6(2):3 – 15.

[Nakaoka *et al.*, 2007] Nakaoka, S., Nakazawa, A., Kanehiro, F., Kaneko, K., Morisawa, M., Hirukawa, H. et Ikeuchi, K. (2007). Learning from observation paradigm : Leg task models for enabling a biped humanoid robot to imitate human dances. *International Journal of Robotics Research*, 26(8):829–844.

[Nakaoka *et al.*, 2003] Nakaoka, S., Nakazawa, A., Yokoi, K., Hirukawa, H. et Ikeuchi, K. (2003). Generating whole body motions for a biped humanoid robot from captured human dances. *In International Conference on Robotics and Automation (ICRA)*, pages 3905 – 3910, Taipei, Taiwan.

[Pearl, 1988] Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann Publisher Inc.

[Penrose et Todd, 1955] Penrose, R. et Todd, J. A. (1955). A generalized inverse for matrices. *Mathematical Proceedings of The Cambridge Philosophical Society*, 51:406–413.

[Peters et Schaal, 2008] Peters, J. et Schaal, S. (2008). Reinforcement learning of motor skills with policy gradients. *Neural Networks*, 21(4):682–697.

[Raab *et al.*, 1979] Raab, F. H., Blood, E. B., Steiner, T. C. et Jones, H. R. (1979). Magnetic position and orientation tracking system. *IEEE Transactions on Aerospace Electronics Systems*, 15:7090–718.

[Rabiner, 1989] Rabiner, L. R. (1989). *A tutorial on hidden markov models and selected applications in speech recognition*. Morgan Kaufmann Publishers Inc.

[Ramos *et al.*, 2011] Ramos, O., Saab, L., Hak, S. et Mansard, N. (2011). Dynamic motion capture and edition using a stack of tasks. *In International Conference on Humanoid Robots (Humanoids)*, Bled, Slovénie.

[Rezzoug *et al.*, 2010] Rezzoug, N., Jacquier-Bret, J. et Gorce, P. (2010). A method for estimating three-dimensional human arm movement with two electromagnetic sensors. *Computer Methods in Biomechanics and Biomedical Engineering*, 13(6):663–668.

[Rose *et al.*, 1998] Rose, C., Bodenheimer, B. et Cohen, M. F. (1998). Verbs and adverbs : Multidimensional motion interpolation. *Computer Graphics and Applications*, 18(5):32–40.

[Saab *et al.*, 2011] Saab, L., Ramos, O., Mansard, N., Soueres, P. et Fourquet, J.-Y. (2011). Generic dynamic motion generation with multiple unilateral constraints. *In IEEE/RAS Intelligent Robots and Systems (IROS)*, San Fransisco, États Unis.

[Samson *et al.*, 1991] Samson, C., Le Borgne, M. et Espiau, B. (1991). *Robot Control : the Task Function Approach*. Clarendon Press, Oxford, UK.

[Schaal *et al.*, 2003] Schaal, S., Ijspeert, A. J. et Billard, A. (2003). Computational approaches to motor learning by imitation. *Philosophical Transaction of the Royal Society of London, series B*, 358(1431):537–547.

[Scholz et Schöner, 1999] Scholz, J. P. et Schöner, G. (1999). The uncontrolled manifold concept : identifying control variables for a functional task. *Experimental brain research*, 126(3):289–306.

[Sentis, 2007] Sentis, L. (2007). *Synthesis and Control of Whole-Body Behaviors in Humanoid Systems*. Thèse de doctorat, Stanford University.

[Shin *et al.*, 2001] Shin, H. J., Lee, J., Shin, S. Y. et Gleicher, M. (2001). Computer puppetry : An importance-based approach. *ACM Transactions on Graphics*, 20:67–94.

[Shon *et al.*, 2005] Shon, A. P., Grochow, K. et Rao, R. P. N. (2005). Robotic imitation from human motion capture using gaussian processes. *In International Conference on Humanoid Robots (Humanoids)*, pages 129–134, Tsukuba, Japon.

[Shotton *et al.*, 2011] Shotton, J., Fitzgibbon, A., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A. et Blake, A. (2011). Real-time human pose recognition in parts from a single depth image. *In Computer Vision and Pattern Recognition (CVPR)*, Colorado Springs, États-Unis.

[Siciliano et Slotine, 1991] Siciliano, B. et Slotine, J. J. (1991). A general framework for managing multiple tasks in highly redundant robotic systems. *In IEEE International Conference on Advanced Robotics (ICAR)*, Pise, Italie.

[Silaghi *et al.*, 1998] Silaghi, M. C., Plänkers, R., Silaghi, M. L., Boulic, R., Fua, P. et Thalmann, D. (1998). Local and global skeleton fitting techniques for optical motion capture. *In Workshop on Modelling and Motion Capture Techniques for Virtual Environments (CAPTECH)*, pages 26–40, Genève, Suisse. Springer.

[So et Baciu, 2005] So, C. et Baciu, G. (2005). Entropy-based motion extraction for motion capture animation : Motion capture and retrieval. *Computer Animation and Virtual Worlds*, 16:225–235.

[So et Baciu, 2006] So, C. K. F. et Baciu, G. (2006). Hypercube sweeping algorithm for subsequence motion matching in large motion databases. *In ACM International Conference on Virtual Reality Continuum and Its Applications*, pages 221–228, Hong Kong, Chine.

[Sreenivasa *et al.*, 2009] Sreenivasa, M., Soueres, P., Laumond, J. et Berthoz, A. (2009). Steering a humanoid robot by its head. *In IEEE/RAS Intelligent Robots and Systems (IROS)*, pages 4451–4456, St. Louis, États Unis.

[Stilman *et al.*, 2005] Stilman, M., Michel, P., Chestnutt, J., Nishiwaki, K., Kagami, S. et Kuffner, J. (2005). Augmented reality for robot development and experimentation. Technical Report CMU-RI-TR-05-55, Robotics Institute, Carnegie Mellon University.

[Sugihara et Nakamura, 2002] Sugihara, T. et Nakamura, Y. (2002). Whole-body cooperative balancing of humanoid robot using cog jacobian. *In JSME Annual Conference on Robotics and Mechatronics*, Matsue, Japon.

[Sugihara *et al.*, 2005] Sugihara, T., Takano, W., Yamane, K., Yamamoto, K. et Nakamura, Y. (2005). Online dynamical retouch of motion patterns towards animatronic humanoid robots. *In International Conference on Humanoid Robots (Humanoids)*, pages 117–122, Tsukuba, Japon.

[Suleiman *et al.*, 2008] Suleiman, W., Yoshida, E., Laumond, J. P. et Monin, A. (2008). On human motion imitation by humanoid robot. *In International Conference on Robotics and Automation (ICRA)*, pages 2697–2704, Pasadena, États-Unis.

[Takano *et al.*, 2005] Takano, W., Tanie, H. et Nakamura, Y. (2005). Key feature extraction for probabilistic categorization of human motion patterns. *In IEEE International Conference on Advanced Robotics (ICAR)*, Barcelone, Espagne.

[Tipping et Bishop, 1999] Tipping, M. E. et Bishop, C. M. (1999). Probabilistic principal component analysis. *Journal of the Royal Statistical Society, Series B*, 61(3):611–622.

[Wampler, 1986] Wampler, C. W. (1986). Manipulator inverse kinematic solutions based on vector formulations and damped least-squares methods. *IEEE Transactions on Systems, Man and Cybernetics*, 16(1):93 – 101.

[Wang *et al.*, 2008] Wang, J. M., Fleet, D. J. et Hertzmann, A. (2008). Gaussian process dynamical models for human motion. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, 30(2):283–298.

[Wang et Chen, 1991] Wang, L. C. T. et Chen, C. C. (1991). A combined optimization method for solving the inverse kinematics problems of mechanical manipulators. *IEEE Transactions on Robotics and Automation*, 7(4):489 – 499.

[Wei et Chai, 2010] Wei, X. et Chai, J. (2010). Videomocap : modeling physically realistic human motion from monocular video sequences. *ACM Transactions on Graphics*, 29(4):42 :1–42 :10.

[Wolovich et Elliott, 1984] Wolovich, W. A. et Elliott, H. (1984). A computational technique for inverse kinematics. *In IEEE Conference on Decision and Control*, Las Vegas, États-Unis.

[Yamane et Hodgins, 2009] Yamane, K. et Hodgins, J. (2009). Simultaneous tracking and balancing of humanoid robots for imitating human motion capture data. *In IEEE/RAS Intelligent Robots and Systems (IROS)*, pages 2510–2517, St Louis, États-Unis.

[Yoshida *et al.*, 2007] Yoshida, E., Mallet, A., Lamiraux, F., Kanoun, O., Stasse, O., Poirier, M., Dominey, P., Laumond, J. et Yokoi, K. (2007). "Give me the purple ball" - he said to HRP-2 N.14. *In International Conference on Humanoid Robots (Humanoids)*, Pittsburg, États-Unis.

[Zordan et Horst, 2003] Zordan, V. B. et Horst, N. C. V. D. (2003). Mapping optical motion capture data to skeletal motion using a physical model. *In ACM Eurographics Symposium on Computer Animation*, San Diego, États-Unis.

Table des figures

Reconnaissance de tâches par commande inverse

Des méthodes efficaces s'appuyant sur des outils statistiques pour réaliser de la reconnaissance de mouvement ont été développé. Ces méthodes reposent sur l'apprentissage de primitives situé dans des espaces approprié, par exemple l'espace latent de l'espace articulaire et/ou d'espace de tâches adéquat. Les primitives apprises sont souvent séquentielle : un mouvement est segmenté selon l'axe des temps. Dans le cas d'un robot humanoïde, le mouvement peut être décomposé en plusieurs sous-tâches simultanées. Par exemple dans un scénario de serveur, le robot doit placer une assiette sur la table avec une main tout en maintenant son plateau horizontal avec son autre main. La reconnaissance ne peut donc pas se limiter à une seule et unique tâche par segment de temps consécutif. La méthode présenté dans ces travaux utilise la connaissance des tâches que le robot est capable d'accomplir, ainsi que des contrôleurs qui génèreront les mouvements pour réaliser une rétro ingénierie sur un mouvement observé. Cette analyse est destiné à reconnaître des tâches qui ont été exécuté de manière simultanées. La méthode repose sur la fonction de tâche et les projection dans l'espace nul des tâches afin de découpler les contrôleurs. L'approche a été appliqué avec succès sur un vrai robot pour distinguer des mouvements visuellement très proches, mais sémantiquement différents.

Mots-clefs : Analyse de mouvements, pile de tâches, robotique

Task recognition by reverse control

Efficient methods to perform motion recognition have been developed using statistical tools. Those methods rely on primitives learning in a suitable space, for example the latent space of the joint angle and/or adequate task spaces. The learned primitives are often sequential : a motion is segmented according to the time axis. When working with a humanoid robot, a motion can be decomposed into simultaneous sub-tasks. For example in a waiter scenario, the robot has to keep some plates horizontal with one of his arms, while placing a plate on the table with its free hand. Recognition can thus not be limited to one task per consecutive segment of time. The method presented in this work takes advantage of the knowledge of what tasks the robot is able to do and how the motion is generated from this set of known controllers to perform a reverse engineering of an observed motion. This analysis is intended to recognize simultaneous tasks that have been used to generate a motion. The method relies on the task-function formalism and the projection operation into the null space of a task to decouple the controllers. The approach is successfully applied on a real robot to disambiguate motion in different scenarios where two motions look similar but have different purposes.

Keywords : Motion analysis, stack of tasks, robotics