Table des matières

Rapport-gratuit.com
LE NUMERO 1 MONDIAL DU MÉMOIRES

Définition des variables

Nous listons ici les variables utilisées tout au long de la thèse et rappelons leur signification. Le lecteur pourra s'y référer pour retrouver la signification d'une variable introduite avant la partie en cours de lecture.

A	concaténation de toutes les arêtes de tous les contacts
$a_{i,j}$	arête j du contact i
b	matrice des frottements visqueux articulaires vue après adjonction du couplage, $b = b_a + \sum J^t b_c J$ **OU** matrice d'un frottement visqueux quelconque
b_a	matrice des frottements visqueux articulaires de la dynamique du 1^{er} ordre
b_c	matrice des frottements visqueux d'un contrôleur de type *proportionnel-dérivé* exprimé dans l'espace opérationnel
b_{int}	matrice des frottements visqueux d'un contrôleur interne
b_q	matrice des frottements visqueux d'un contrôleur articulaire
b_x	matrice des frottements visqueux d'un contrôleur de type *computed torque* exprimé dans l'espace opérationnel
$C(q, \dot{q})$	matrice des forces de Coriolis et des forces centrifuges vue dans l'espace articulaire
com_i	centre de gravité du solide i
D	matrice associée à une décomposition SVD ($J = U \begin{bmatrix} D & 0 \end{bmatrix} V$)
D_r	restriction de D aux r valeurs singulières non-nulles
$d(q)$	distance à la violation de contrainte (modèle géométrique de la contrainte)
$d(q)^{free}$	distance à la contrainte sans compensation
d_s	distance à la contrainte sur le swing
d_t	distance à la contrainte sur le twist
η	X choisi pour faire respecter les contraintes, à projeter
f	fonction quelconque **OU** forces inertielles et de Coriolis en translation
f_{c_i}	force associée au contact i
$f_{c_i}^{dis}$	f_{c_i} discrétisé
g	gravité
G	matrice caractérisant les contacts et le poids, dans l'équation statique du système $GX = W$
G_f	restriction de G aux termes en force
G_x	restriction de G aux termes en position
$G(q)$	couples dérivant d'un potentiel de l'équation dynamique du $2^{\text{ème}}$ ordre exprimée dans l'espace articulaire
Γ	couples articulaires

Γ_c	couples de contrainte, générés pour faire respecter la contrainte
Γ_0	couples internes quelconques, destinés à être projetés
Γ_{int}	couples de commande issus du contrôle interne
$\Gamma_{prev(k)}$	$\Gamma_{prev(k)} = \Pi^t_{prev(k)\|b_a} \Gamma_k$
Γ_{task}	couples de commande
H	position quelconque de $SE(3)$, $H = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix}$
H_d	H désiré
${}^iH_j$	position dans $SE(3)$ de j par rapport à i exprimée dans la base i
${}^iH_j(t)$	position dans $SE(3)$ de j par rapport à i exprimée dans la base i à l'instant t
I_f	restriction de l'identité aux termes de force de X
I_q	restriction de l'identité aux termes de configuration de X
I_x	restriction de l'identité aux termes de position de X
J	jacobienne quelconque (associée à V)
J^+	pseudo inverse de J
J_a	jacobienne augmentée, correspondant à V_a **OU** jacobienne de v_a
$J_{A\in i/j(k)}$	jacobien associé $V_{A\in i/j(k)}$
J_c	jacobien associé à la contrainte
J_{c_i}	jacobien associée au contact i
J^g	inverse généralisée quelconque de J
J_i	jacobienne du repère de contrôle (ou de la tâche) i
$J_{prev(k)}$	$J_{prev(k)} = J_k \Pi_{prev(k)}$
J^r	jacobienne 3D, réduite aux composantes translationnelles
J_s	jacobien associé au swing
J_t	jacobien associé au twist
J_u	jacobienne interne, correspondant à V_u
K	matrice de poids quelconque (symétrique définie positive)
k	raideur quelconque
k_c	raideur d'un contrôleur de type *proportionnel-dérivé* exprimé dans l'espace opérationnel
k_{int}	raideur d'un contrôleur interne
k_q	raideur d'un contrôleur articulaire
k_x	raideur d'un contrôleur de type *computed torque* exprimé dans l'espace opérationnel
L	lagrangien quelconque
λ	multiplicateurs de Lagrange
$\Lambda(q)$	matrice d'inertie du système vue dans l'espace opérationnel
m	masse totale du système **OU** nombre de degrés de liberté (= nombre d'articulations pour simplifier)
m_a	nombre d'arêtes par cône de frottement
m_c	nombre de contacts
m_k	masse du solide k
M	matrice d'un LCP quelconque
$M(q)$	matrice d'inertie du système vue dans l'espace articulaire
$\mu(q)$	matrice des forces de Coriolis et des forces centrifuges vue dans l'espace opérationnel
N	base du noyau de J **OU** concaténation des normales de tous les contacts
n	normale quelconque **OU** nombre de tâches (ou de repères de contrôle) sur le système
n_i	normale au contact i
ν	G^+W
ν_f	restriction de ν aux termes de force
ν'_{f_i}	ν_f incluant la marge Δx_i
ν'_f	$\nu'_f = \min_i \left(\nu'_{f_i}\right)$
ν_x	restriction de ν aux termes de position

ω	une des inconnues d'un LCP quelconque **OU** forces inertielles et de Coriolis rotationnelles
p	position quelconque de $\mathbb{R}^3$
p_{a_i}	bras de levier entre le centre de a et le contact i
p_i	position du contact i
$p(q)$	couples dérivant d'un potentiel de l'équation dynamique du $2^{\text{ème}}$ ordre exprimée dans l'espace opérationnel
Π	projecteur quelconque
Π_i	projecteur dans le noyau de la tâche i
$\Pi_{prev(k)}$	projecteur dans le noyau des tâches 1 à $k-1$
$\Pi_{prev(k)\|b_a}$	$\Pi_{prev(k)}$ dont la pseudo-inverse est pondérée par b_a
Q	métrique **OU** ellipse
q	variété de configuration
q_c	correction articulaire calculée pour faire respecter la contrainte
q_d	variété de configuration désirée
q^{free}	configuration avant application de la contrainte
q_i	variables articulaires liées au solide i
R	rotation quelconque de $SO(3)$
r	$log(R) \in \mathfrak{se}(3)$ **OU** nombre de valeurs non singulières d'une matrice
${}^iR_j$	rotation dans $SO(3)$ de j par rapport à i exprimée dans i
R_s	rotation du swing dans $SO(3)$
R_t	rotation du twist dans $SO(3)$
r_s	$log(R_s) \in so(3)$
r_t	$log(R_t) \in so(3)$
ρ	composante vectorielle d'un LCP quelconque
Σ	matrice associée à une décomposition SVD ($J = U\Sigma V$)
t	temps
T	$\eta = T\xi$
θ_s	angle du swing
θ_t	angle du twist
U	matrice associée à une décomposition SVD ($J = U\Sigma V$) OU potentiel interne quelconque
U_r	restriction de U aux r valeurs singulières non-nulles
U_s	restriction de U aux valeurs singulières nulles
V	torseur cinématique quelconque **OU** matrice associée à une décomposition SVD ($J = U\Sigma V$) **OU** potentiel auto-projetant à projection constante
V_a	torseur cinématique (ou vitesse) augmentée
$V_{i/j(k)}$	torseur cinématique (ou vitesse) de i par rapport à j exprimé dans la base k
$V_{prev(k)}$	torseur cinématique (ou vitesse) associé à la jacobienne $J_{prev(k)}$
V_U	torseur cinématique (ou vitesse) interne
$V_{A\in i/j(k)}$	torseur cinématique (ou vitesse) de A fixe dans i par rapport à j exprimé dans k
v_a	vitesse 3D de a
v_{a_i}	vitesse 3D de a vue au contact i
$v_{i/j(k)}$	vitesse 3D de i par rapport à j exprimée dans k
W	torseur d'efforts (ou efforts) quelconque
W'	W incluant les marges de stabilité
W_{task}	efforts de commande
X	inconnues du problème d'équilibre non-coplanaire
x	position quelconque exprimée dans $\mathfrak{se}(3)$
x_c	composante de x_{com}
x_d	x désiré
x_i	x au repère de contrôle (ou à la tâche) i
x_k	x au pas k

Rapport-Gratuit.com

$x_{i(k)}$	x_i projeté dans la base k
$x^r_{O_{c_i} \in c_i/j(k)}$	position du centre du repère c_i fixe dans c_i par rapport à j projeté dans k
ξ	η exprimé dans une base différente
$\xi_{i/j(k)_{init}}$	axe de la liaison de i par rapport à j projeté dans la base k à l'initialisation
y_c	composante de x_{com}
Y	admittance quelconque
z	une des inconnues d'un LCP quelconque
z_c	composante de x_{com}
Z	impédance quelconque

Introduction

Le domaine de la simulation, lié à l'envol des technologies informatiques, permet d'envisager des cas de plus en plus complexes. Cette amélioration continue des performances des simulations est bien entendu fondée sur l'augmentation des vitesses de calcul des machines, mais également sur les raffinements successifs des algorithmes de simulation.

La *maquette numérique*, basée sur ces technologies de l'information, a permis de bouleverser la conception industrielle. Autrefois organisée autour de revues de projets sur maquettes physiques, les méthodes récentes de conception s'appuient dorénavant sur des *prototypes virtuels*. L'innovation en terme de simulation devient déterminante. Elle fait appel, entre autres, aux techniques émergentes dites de *réalité virtuelle*. Ces simulations doivent refléter au mieux les comportements des produits dans le monde auquel ils sont destinés. Et cette constatation est d'autant plus critique, que dans bien des cas le caractère temps-réel des simulations est déterminant.

Les fortes potentialités du prototypage virtuel sont claires aujourd'hui. Mais celui-ci ne deviendra une réalité pleinement tangible en entreprise que lorsqu'il aura été débarrassé de toutes les contraintes qui y sont liées pour le moment. Nous pouvons citer la préparation des données de l'environnement que nous souhaitons simuler [DLCG05], mais également les difficultés de mise en œuvre de tous les cas de tests. Ainsi la mise en œuvre de tests permettant de vérifier l'interaction entre l'utilisateur et le produit présente toujours des difficultés liées à la complexité des systèmes en jeu.

Les techniques autorisant à mettre en œuvre des mannequins virtuels permettant de tester la maquette numérique, sont principalement issues des domaines du *jeu vidéo* et des *effets spéciaux*. Leurs problématiques technologiques sont très proches de la notre et de manière générale, la réalité virtuelle bénéficie énormément des avancées de ces domaines très dynamiques. Cette thèse a pour objet d'y apporter une contribution. Nos travaux ont été envisagés principalement pour des applications liées à l'ingénierie mais les développements effectués pourront bénéficier à tous les domaines intéressés par l'humain virtuel.

Une des finalités de notre travail est de vérifier, de manière virtuelle et pendant sa conception, si le produit répond bien aux contraintes liées à son interaction avec l'utilisateur humain (ergonomie, accessibilité...). Notre étude portera donc sur le développement d'un système temps-réel d'animation prenant en compte l'**interaction physique avec l'environnement**, de manière naturelle et toutes ses implications en terme de stabilité, de contact et d'équilibre.

Le premier chapitre sera consacré à la description du contexte dans lequel nous avons étudié la simulation d'humains (ou mannequins) virtuels - modèles d'humains évoluant au sein de simulations informatiques. Nous y décrirons un état de l'art des techniques d'animation et de spécification du mou-

vement. Nous exposerons ensuite la problématique du prototypage virtuel, notre domaine d'application. Nous pourrons alors introduire un schéma fonctionnel global du système d'animation. Nous présenterons enfin les contributions de cette thèse.

Le deuxième chapitre nous permettra de jeter les bases d'un contrôle d'humains virtuels simple. Nous y présenterons d'abord la notion de passivité, qui nous servira de guide tout au long de l'étude. Nous identifierons alors les éléments constituants du contrôle. Ceci nous permettra de proposer une architecture pour implémenter les fonctionnalités de base de l'animation. Nous soulignerons enfin certains problèmes relatifs aux projections et proposerons des éléments de réponse.

Le chapitre 3 enrichit l'architecture donnée en 2 en permettant d'apporter une solution à la gestion des contraintes. Nous y décrirons d'abord un solveur de contraintes unilatérales, pour lequel nous formulerons le contact et les limites articulaires. Puis nous nous attacherons à décrire un contrôleur d'équilibre pour l'humain virtuel, dans le cas des contacts coplanaires d'abord, puis dans le cas multi-contacts non-coplanaires.

Le dernier chapitre est consacré à l'ordonnancement des modes de commande développés dans les chapitres précédents. Nous y décrirons la transition des modes de commande et les problèmes connexes. Après coup nous exposerons notre architecture informatique. Nous décrirons alors des cas d'application mis en œuvre de notre système d'animation. Enfin nous exposerons des perspectives envisageables pour continuer les travaux.

CHAPITRE 1

Contexte de la simulation des humains virtuels

Dans ce chapitre nous exposons l'environnement dans lequel s'insère notre étude. Nous décrivons donc dans un premier temps le système (ou modèle de mannequin) que nous souhaitons animer.

Il existe une littérature fournie liée aux humains virtuels. Nous nous attachons donc à décrire les travaux qui présentent un intérêt dans le cadre de notre étude.

Nous présenterons alors le prototypage virtuel *et les nouvelles méthodes de conception, contexte industriel dans lequel s'insèrent nos travaux.*

Après avoir exprimé notre besoin, nous jetterons les premières bases d'une architecture d'animation. Le détail et l'implémentation de celle-ci sera l'objet des autres chapitres de notre étude.

Enfin nous présenterons les contributions qui seront apportées dans les chapitres suivants.

1.1 Modèles d'humains virtuels

Une des représentations, parmi les plus simples, de l'humain virtuel est dictée par les techniques robotiques utilisées pour l'animation (voir 1.2). C'est-à-dire que nous pouvons voir l'humain virtuel comme une structure polyarticulée arborescente de solides, que nous appelons assez naturellement *squelette*. Une géométrie en relation avec la topologie du corps humain [AMH02], est attachée aux solides, ou *os*. Deux os sont séparés par une articulation. Cette représentation est, comme nous pouvons le voir figure 1.1, assez "rigide" et diffère quelque peu de l'idée que nous nous faisons d'un véritable humain :

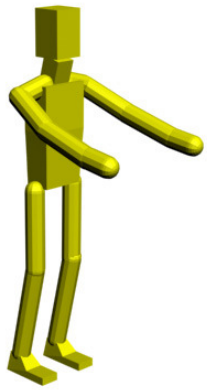

Fig. 1.1 – Représentation d'un mannequin par un ensemble de solides.

Pour obtenir un meilleur aspect visuel (rendu 3D), nous pouvons modéliser l'enveloppe extérieure, ou *peau* de l'humain. Les techniques à base de mailles, de NURBS, de subdivisions de surfaces [Rat03], [Kel98], sont particulièrement adaptées à la modélisation de surfaces organiques. Avec ce type de modèles, des milliers de points paramètrent la surface qui représente la peau du mannequin. La dimension du problème est à l'heure actuelle beaucoup trop élevée pour être résolue de manière directe. L'idée est donc de reparamétrer la surface, grâce au modèle de squelette décrit auparavant, problème de dimension réduite. Le mouvement est alors résolu sur le squelette et les mouvements du squelette sont appliqués à l'enveloppe [AMH02]. Cette technique est appelée *skinning* et peut maintenant être accélérée par les cartes graphiques et leurs langages de programmation de *shaders*[1]. Le skinning est beaucoup utilisé dans les jeux vidéo, il est d'ailleurs mis à profit sur les derniers démonstrateurs de nVidia® [Pha05]. C'est la méthode que nous utiliserons pour le rendu - nous utilisons à ce titre le logiciel Virtools® qui implémente l'algorithme.

Cette méthode donne de bons résultats, mais ne conserve pas les volumes. Aussi, quand on s'éloigne de la configuration initiale, des déformations peu convaincantes apparaissent, comme montré dans [LCF00], figure 1.3.

Cette observation a mené certains chercheurs, comme Aubel [Aub02], à se rapprocher de l'architecture musculo-squelettique humaine. Il associe une géométrie réaliste des os, des muscles et une couche sous-cutanée de graisse, auxquels il adjoint un comportement jugé réaliste. Les résultats sont assez probants, comme le montre la figure 1.4.

Le réalisme peut être poussé encore plus avant. Les mannequins virtuels peuvent servir lors de crash tests virtuels. Pour obtenir des données intéressantes les modèles doivent être particulièrement détaillés [fSH05], leur topologie sera dictée par les méthodes de calcul utilisées. Ces modèles sont beaucoup plus complexes que ceux cités précédemment et sont loin d'être compatibles avec les contraintes temps-réel évoquées par la suite.

Nous ne détaillons pas la cinématique du mannequin, celle-ci étant ajustée en rapport au cas d'étude et

[1] Petits programmes destinés à décrire la méthode de rendu utilisée.

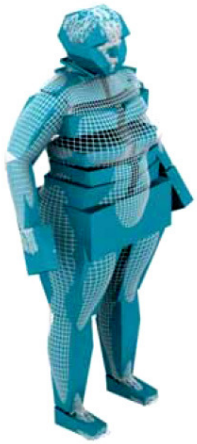

FIG. 1.2 – Maille et squelette sous jacent, d'après [SMT04].

FIG. 1.3 – La déformation sur le bras est due au skinning, d'après [VRL].

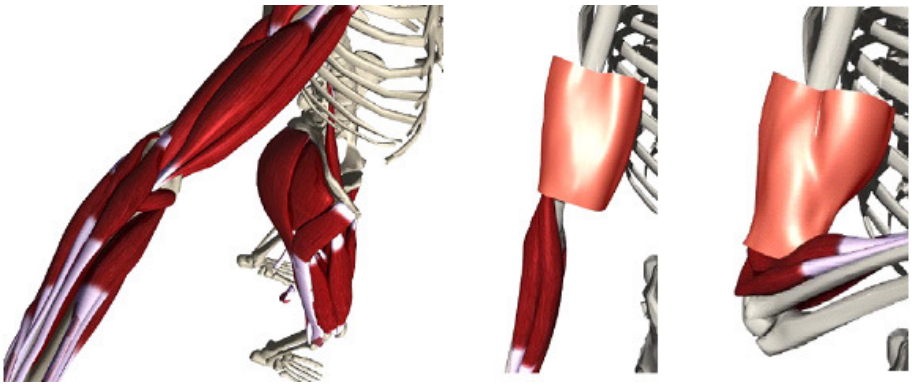

FIG. 1.4 – Architecture de déformation musculo-squelettique selon A. Aubel.

aux capacités des machines à disposition. Elle reste très classique [FVdPT98], [HW98], l'illustration d'un exemple, à 36 degrés de liberté, est donnée figure 1.5. Nous souhaitons cependant insister sur un point critique : l'épaule présente une fermeture de la chaîne cinématique[2] [LS03], notre modèle n'implémentera pas cette particularité (à l'heure actuelle les boucles cinématiques ne sont pas résolues de manière exacte dans le simulateur que nous allons mettre en œuvre). De même nous simplifions les articulations de type rotule par trois pivots à angles droits (structure équivalente aux singularités près).

[2]C'est-à-dire qu'en parcourant la chaîne cinématique de proche en proche, il est possible de revenir à une articulation déjà rencontrée précédemment.

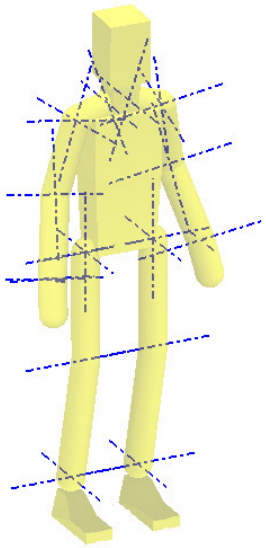

Fig. 1.5 – Exemple de modèle cinématique mis en œuvre pour les besoins d'un démonstrateur. Les axes articulaires sont représentés en bleu.

1.2 L'animation de personnages virtuels dans la littérature

1.2.1 Contrôle du mouvement

Nous présentons ici des techniques d'animation d'humains virtuels (vus comme des structures poly-articulées arborescentes de solides), déjà données dans la littérature. Nous décrivons d'abord les bases mathématiques utiles à notre étude. Nous abordons alors des méthodes de contrôle très bas niveau, puis des méthodes s'appuyant sur ces couches basses.

1.2.1.1 Formalisme mathématique : groupes de Lie

Les considérations qui suivent peuvent être retrouvées dans [PC04], ou [MLSS94]. Un cadre mathématique naturel pour l'étude des déplacements des corps rigides (ou solides) est celui du *groupe spécial Euclidien* $SE(3)$. Le *groupe spécial Euclidien* $SE(3)$ est le produit semi-direct de $\mathbb{R}^3$ et du *groupe spécial orthogonal* $SO(3)$. Un élément de $SE(3)$ est représenté par une matrice de transformation homogène.

Aparté - $SE(3)$ en deux mots

Concrètement l'espace $SE(3)$ est une modélisation de l'espace dans lequel nous vivons. Pour repérer les positions des solides, on leur adjoint un repère d'espace (voir figure 1.6), dont la position est exprimée dans $SE(3)$ espace 6D. Trois des paramètres sont liés aux translations, les trois autres étant liés à l'*attitude*, ou orientation du solide. $SE(3)$ est également appelé *espace opérationnel*, ou *espace de la tâche*.

L'ensemble des transformations rigides (notées H) sur $\mathbb{R}^3$, tel que $H(p_0) = Rp_0 + p$, avec $R \in SO(3)$ et $p \in \mathbb{R}^3$, forme le groupe spécial Euclidien $SE(3)$. $SO(3)$ est formé des matrices de rotation, il est défini par :

$$SO(3) = \left\{ R \in \mathbb{R}^{3\times 3} | RR^t = I, det(R) = 1 \right\}, \tag{1.1}$$

où I est l'identité. Chaque élément H de $SE(3)$, peut être écrit sous la forme d'une matrice homogène

de dimensions 4×4, de la forme :

$$H = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix}. \tag{1.2}$$

On peut montrer que le groupe $SE(3)$ est un groupe de Lie de dimension 6 pour la multiplication matricielle. Un groupe de Lie est un groupe G, qui est également une variété différentiable pour laquelle le produit est une application différentiable $G \times G \to G$. $SE(3)$ est un sous-groupe de $GL(n)$, groupe des matrices inversibles de dimensions $n \times n$ pour la multiplication matricielle.

L'algèbre de Lie $\mathfrak{se}(3)$

L'algèbre de Lie $\mathfrak{g}$ associée à un groupe de Lie G, est définie par $\mathfrak{g} = T_I G$, espace tangent à G à l'identité. L'algèbre de Lie de $SE(3)$, notée $\mathfrak{se}(3)$ est isomorphe à $\mathbb{R}^6$, elle est formée des vecteurs $V = \begin{bmatrix} \omega \\ v \end{bmatrix}$, avec $\omega, v \in \mathbb{R}^3$. V peut être interprété comme un torseur cinématique. Chaque élément de $\mathfrak{se}(3)$ peut être associé à une matrice 4×4, notée $[V]$, de la forme :

$$[V] = \begin{bmatrix} [\omega] & v \\ 0 & 0 \end{bmatrix}, \tag{1.3}$$

où $[\omega]$ est la représentation par une matrice 3×3 antisymétrique d'un élément de $so(3)$ (l'algèbre de Lie associée à $SO(3)$) :

$$\text{si on a } \omega = \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \in \mathbb{R}^3, \text{ alors } [\omega] = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}. \tag{1.4}$$

Notons que $[\omega_1]\omega_2 = \omega_1 \wedge \omega_2$, avec $\wedge$ le produit vectoriel.

Deux conventions sont envisageables, $[V] = T^{-1}\dot{T}$, ou $[V] = \dot{T}T^{-1}$. Nous utiliserons l'une ou l'autre suivant les besoins, en notant $V_{i/j(k)}$ le torseur cinématique du repère i dans son mouvement par rapport au repère j, projeté dans la base associée à k et réduit au centre du repère k.

L'opération appelée *crochets de Lie* sur $\mathfrak{se}(3)$, notée $[\cdot,\cdot] : \mathfrak{se}(3) \times \mathfrak{se}(3) \to \mathfrak{se}(3)$ est le commutateur défini par $[[V_1], [V_2]] = [V_1][V_2] - [V_2][V_1]$. En utilisant l'opérateur *adjoint* défini par

$$ad_V = \begin{bmatrix} [\omega] & 0 \\ [v] & [\omega] \end{bmatrix}, \text{ avec } V = \begin{bmatrix} \omega \\ v \end{bmatrix}, \tag{1.5}$$

on peut vérifier que

$$[[V_1], [V_2]] = [ad_{V_1} V_2], \tag{1.6}$$

puisque

$$ad_{V_1} V_2 = \begin{bmatrix} \omega_1 \wedge v_2 - \omega_2 \wedge v_1 \\ \omega_1 \wedge \omega_2 \end{bmatrix}. \tag{1.7}$$

Le changement de coordonnées de $V \in \mathfrak{se}(3)$ est effectué par l'intermédiaire de la transformation adjointe induite par $H \in SE(3)$ par la relation suivante

$$[Ad_H V] = H[V]H^{-1}. \tag{1.8}$$

On peut vérifier que

$$Ad_H = \begin{bmatrix} R & 0 \\ [p]\,R & R \end{bmatrix}, \tag{1.9}$$

pour H donné en (1.2).

Application exponentielle

Pour un groupe de Lie G et son algèbre $\mathfrak{g}$, l'application exponentielle, $\exp : \mathfrak{g} \to G$, est définie comme l'exponentielle matricielle. Pour $SE(3)$, par conséquent, l'exponentielle de $V \in \mathfrak{se}(3)$ est définie comme l'exponentielle matricielle de $[V]$, c'est-à-dire $\exp(V) = \text{expm}([V]) \in SE(3)$, où $\text{expm}(A) = \sum_{i=0}^{\infty} \left(\frac{1}{i!} A^i\right)$. Une forme close peut être trouvée sur $\mathfrak{se}(3)$ ([PC04]) :

$$\begin{cases} \omega = 0, \;\; \exp(V) = \begin{bmatrix} I & v \\ 0 & 1 \end{bmatrix} \\ \\ \omega \neq 0, \;\; \exp(V) = \begin{bmatrix} \Omega & \frac{1}{\|\omega\|^2} \left(I - \Omega\right) \left(\omega \wedge v + \omega\omega^t v\right) \\ 0 & 1 \end{bmatrix} \end{cases}, \tag{1.10}$$

avec $\Omega = \exp(\omega) = \text{expm}([\omega]) \in SO(3)$, défini par

$$\exp(\omega) = I + \frac{\sin\left(\|\omega\|\right)}{\|\omega\|}[\omega] + \frac{1 - \cos(\|\omega\|)}{\|\omega\|^2}[\omega]^2, \tag{1.11}$$

cette dernière équation est connue sous le nom de formule de Rodrigues. Pour simplifier les notations, nous confondrons les opérations exp et expm.

En posant ${}^jH_i$, position $6D$ du repère i par rapport au repère j projetée dans la base associée à j, on a $[V_{i/j(i)}] = {}^iH_j {}^j\dot{H}_i$ ([MLSS94]). S. Stramigioli [SB01] nous indique que la solution à l'équation ${}^j\dot{H}_i = {}^jH_i[V_{i/j(i)}]$ est donnée par :

$${}^jH_i(t) = \exp\left([V_{i/j(i)}]t\right) {}^jH_i(0), \tag{1.12}$$

avec t le temps. Imaginons que le repère i soit relié au repère j par une liaison pivot d'axe $\xi_{i/j(i)}$ (axe de la liaison liant i à j exprimé dans la base associée à i), d'angle q_i par rapport à la position initiale, comme indiqué figure 1.6. On peut alors écrire (1.12) sous la forme :

$$\begin{aligned} {}^jH_i(t) &= \exp\left([V_{i/j(i)}]t\right) {}^jH_i(0) \\ &\triangleq \exp\left([\xi_{i/j(i)}]q_i\right) {}^jH_i(0). \end{aligned} \tag{1.13}$$

Cette manipulation nous permet de paramétrer une liaison de type pivot, par un axe $\xi_{i/j(i)}$ et un paramètre $q_i \in \mathbb{R}$. Nous avons essentiellement mis en œuvre des liaisons de type pivot, le cas des rotules est détaillé en annexe D.1.

1.2.1.2 Contrôle bas niveau

Deux espaces sont couramment manipulés pour exprimer la position d'un robot, ou squelette d'humain virtuel. L'espace de configuration Q, qui est la variété des paramètres articulaires, et l'espace opérationnel, qui permet de spécifier la position d'un des solides du squelette. Tout l'intérêt des méthodes bas-niveau est de passer d'un espace à l'autre de la manière la plus adaptée.

Cinématique

La plus simple des techniques d'animation est la *cinématique directe*. Elle consiste à spécifier chaque paramètre articulaire et à observer la position résultante de l'*effecteur* dans $SE(3)$.

Nous appellerons plus volontiers les effecteurs, *repères de contrôle*, ou *points de contrôle*, car les points d'intérêt ne sont pas toujours terminaux sur la chaîne. Nous pouvons passer de la variété de configuration

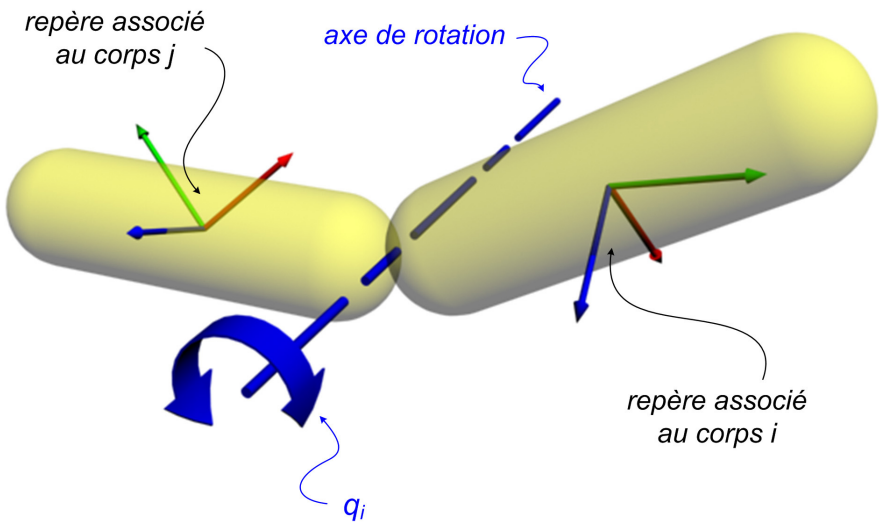

FIG. 1.6 – Paramétrage d'une liaison de type pivot, par un axe $\xi_{i/j(i)}$ et un angle de rotation q_i.

à l'espace opérationnel en reprenant le formalisme des groupes de Lie et la formule de Brockett [SB01] :

$$ {}^0H_i(t) = \prod_{k=1}^{i} \exp\left(\xi_{k/k-1(0)_{init}} q_k\right) {}^0H_i(0), \tag{1.14} $$

avec 0 indice du repère de base et i indice d'un repère d'intérêt.

Cette technique est simple et son utilisation est fastidieuse. Il est plus facile de spécifier une tâche dans l'espace opérationnel et de laisser un algorithme retrouver les variables articulaires. C'est le problème inverse.

La *cinématique inverse* est une des premières techniques intéressantes pour l'animation : elle est d'ailleurs très utilisée dans les studios de production. Les grands logiciels d'infographie proposent tous des solutions très élaborées dans ce domaine [Auta], [Ali], [Has]. Nous la détaillons pour souligner l'intérêt qu'elle présente.

Comme son nom l'indique, la cinématique inverse est basée sur l'inversion du problème cinématique direct. Le problème direct pose les équations permettant d'exprimer la position d'un point du squelette à partir de ses variables articulaires. Cette relation $H = f(q)$ entre variables articulaires q (c'est l'ensemble des q_i concaténés en un vecteur) et position H n'est pas linéaire dans le cas général. Aussi, si dans des cas simples, l'inversion du problème est assez directe, dans la majorité des cas, nous ne connaissons pas de solution analytique au problème inverse. Pour pallier cette situation, nous linéarisons le problème et comme cette linéarisation ne peut être que locale, nous itérons sur des pas suffisamment petits pour rester dans l'hypothèse de linéarité.

Nous introduisons le préfixe δ signifiant *petit*, il permet de souligner que nous respectons l'hypothèse des petits déplacements. Pour écrire la linéarisation nous travaillons dans $\mathfrak{se}(3)$, algèbre de Lie associée à $SE(3)$ qui paramètre $SE(3)$ (les deux espaces sont isomorphes à l'exception d'une singularité en π). Nous écrivons une relation entre espace de configuration et espace opérationnel qui s'exprime sous forme de matrice (nous supposerons f lisse) :

$$ \delta x = \log(\delta H) = \log\left(f(\delta q)\right) \tag{1.15} $$

$$ \simeq J\delta q, \tag{1.16} $$

avec J, l'application tangente de $f : J = T_f = \dfrac{\partial f}{\partial q}$. Elle est appelée *matrice jacobienne* du système et représente la relation locale entre la variété de configuration et la variété opérationnelle.

Rem : δH est un abus de notation destiné à rappeler que l'on travaille en petits déplacements.

Nous préférons utiliser une autre expression plus facile à calculer de J. Pour ne pas compliquer les notations, nous supposons le nombre d'articulations égal au nombre de degrés de liberté - l'extension est directe. Alors pour une structure polyarticulée à m articulations (donc $q \in \mathbb{R}^m$), le théorème de composition des vitesses [ABD86] donne :

$$V_{m/0(0)} = \sum_{i=1}^{m} V_{k/k-1(0)} \tag{1.17}$$

$$= \sum_{i=1}^{m} \xi_{k/k-1(0)} \dot{q} \tag{1.18}$$

$$= \begin{bmatrix} \xi_{1/0(0)} & \cdots & \xi_{m/m-1(0)} \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \vdots \\ \dot{q}_m \end{bmatrix}, \tag{1.19}$$

avec q_i paramètres articulaires associés à l'articulation i et $V_{i/j(k)}$, torseur cinématique de i dans son mouvement par rapport à j, projeté et réduit dans k. Nous intègrons alors (1.19) sur δt :

$$V_{m/0(0)} \delta t = \begin{bmatrix} \xi_{1/0(0)} & \cdots & \xi_{m/m-1(0)} \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \vdots \\ \dot{q}_m \end{bmatrix} \delta t \tag{1.20}$$

$$\delta x_{m/0(0)} = \begin{bmatrix} \xi_{1/0(0)} & \cdots & \xi_{m/m-1(0)} \end{bmatrix} \begin{bmatrix} \delta q_1 \\ \vdots \\ \delta q_m \end{bmatrix}, \tag{1.21}$$

en identifiant l'équation précédente à (1.16), il vient :

$$J_{m/0(0)} = \begin{bmatrix} \xi_{1/0(0)} & \cdots & \xi_{m/m-1(0)} \end{bmatrix} \tag{1.22}$$

Pour inverser le problème il suffit alors d'inverser J, nous trouvons alors nos δq à intégrer (l'annexe C montre que l'intégration ne porte pas forcément sur δq). Cette inversion n'est pas toujours possible, deux cas peuvent se présenter :

- l'espace de configuration et l'espace opérationnel n'ont pas la même dimension. Dans ce cas le nombre d'équations et le nombre d'inconnues différent. Le problème peut être sur-contraint : il n'y a pas toujours de solution au problème. Ou sous-contraint : il y a une infinité de solutions. Dans le dernier cas, nous parlons de problème *redondant*, nous reviendrons sur cette notion. Un mannequin possédant de nombreux degrés de liberté, il faudra gérer sa redondance.
- la matrice est singulière, c'est-à-dire qu'elle n'est plus de rang plein. Sur une décomposition en valeurs singulières, ce cas se traduit par la présence de valeurs singulières nulles[3].

Ces deux problèmes peuvent être résolus par l'introduction d'inverses généralisés de J et notamment de la pseudo-inverse J^+.

Supposons J de rang plein et plaçons nous dans le cas redondant. Nous savons qu'il existe une infinité de solutions. La méthode courante d'inversion ne peut donner la moindre information quant à ces

[3]Du fait des approximations dues à la quantification des méthodes numériques, ce mauvais conditionnement donne un comportement hiératique du système non seulement quand les valeurs singulières sont nulles, mais aussi quand nous approchons de ces configurations singulières. Les problèmes liés aux singularités ne sont donc pas ponctuels, mais locaux.

solutions. Pour en sélectionner une parmi cette infinité, nous utilisons la *pseudo-inverse*. Elle est solution d'un problème d'optimisation, sous contrainte de satisfaction de l'équation (1.16). Le critère choisi est la minimisation quadratique du déplacement local $\|\delta q\|$, comme spécifié dans les équations suivantes. La pseudo-inverse possède des propriétés intéressantes, décrites dans [BO71]. Nous obtenons d'autres inverses généralisées en changeant de critère.

$$\begin{cases} \min\limits_{\delta q} \dfrac{1}{2}\|\delta q\|^2 \\ \quad \text{sachant } \delta x = J\delta q. \end{cases} \tag{1.23}$$

Aux valeurs stationnaires, le Lagrangien L vérifie ([Min83]) :

$$\frac{\partial L}{\partial \delta q} = \delta q^t + \lambda^t J = 0 \tag{1.24}$$

d'où nous tirons

$$\begin{cases} \delta q = -J^t\lambda \\ \lambda = -(JJ^t)^{-1}\delta x \end{cases} \tag{1.25}$$

donc

$$\delta q = J^t(JJ^t)^{-1}\delta x \tag{1.26}$$

$$\boxed{\delta q = J^+\delta x, \text{ avec } J^+ = J^t(JJ^t)^{-1}.} \tag{1.27}$$

Notons que nous avons supposé (JJ^t) non singulière, c'est-à-dire que l'on n'a pas pris en compte le problème des singularités. Afin d'analyser ces singularités, nous allons utiliser une *décomposition en valeurs singulières* donnée dans [Wil03], [PTVF02].

Pour simplifier les explications qui vont suivre, nous supposerons les valeurs singulières classées dans l'ordre croissant dans D (cette hypothèse est faite sans perte de généralité, il suffit de réorganiser les matrices U, Σ et V). Nous pouvons tirer énormément d'informations de cette décomposition, ainsi le noyau V_N vient par le réarrangement qui suit

$$J = U\Sigma V^t \tag{1.28}$$
$$= U \begin{bmatrix} D & 0 \end{bmatrix} V^t \tag{1.29}$$
$$JV = \begin{bmatrix} UD & 0 \end{bmatrix} \tag{1.30}$$
$$J \begin{bmatrix} V_r & V_N \end{bmatrix} = \begin{bmatrix} UD & 0 \end{bmatrix}, \tag{1.31}$$

nous avons décomposé V en $\begin{bmatrix} V_r & V_N \end{bmatrix}$, avec V_r les r premières colonnes de V et nous voyons sur (1.31) que V_N est le noyau de J. Naturellement la dimension du noyau augmente quand le rang de J diminue .

Voyons maintenant l'intérêt de modifier quelque peu la définition de la pseudo-inverse donnée jusqu'à présent. Toujours dans le cas redondant, J^+ nous est donnée par

$$J^+ = J^t(JJ^t)^{-1} \tag{1.32}$$
$$= V \begin{bmatrix} D \\ 0 \end{bmatrix} D^{-2}U^t. \tag{1.33}$$

Dans le cas de valeurs singulières, nous posons D_r restriction de D aux r valeurs singulières non nulles

$$D^2 = \begin{bmatrix} D_r^2 & 0 \\ 0 & 0 \end{bmatrix}, \tag{1.34}$$

le caractère non-inversible de D^2 apparaît clairement. La pseudo-inverse telle qu'elle a été définie ne nous est donc pas d'une grande utilité dans le cas de singularités. Notons que dans les cas non-singuliers la pseudo-inversion se résume à

$$J^+ = V \begin{bmatrix} D^{-1} \\ 0 \end{bmatrix} U^t. \tag{1.35}$$

Sur $J = U\Sigma V^t$, nous voyons que les colonnes de U génèrent l'image de J, l'apparition de singularités correspond à des directions de l'espace opérationnel dans lesquelles nous ne pouvons plus déplacer le robot, une commande articulaire n'a pas d'effet dans ces directions :

$$\begin{aligned} J &= U\Sigma V^t && (1.36)\\ &= U \begin{bmatrix} D & 0 \end{bmatrix} V^t && (1.37)\\ &= \begin{bmatrix} U_r & U_s \end{bmatrix} \begin{bmatrix} D_r & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} V^t, && (1.38) \end{aligned}$$

avec U_r les r premières colonnes de U. Nous voyons sur l'équation précédente que les directions U_s de l'espace opérationnel, correspondant aux valeurs singulières nulles, sont des directions dans lesquelles une commande articulaire n'a pas d'effet. En restreignant la pseudo-inverse aux directions U_r, nous pouvons parvenir à une pseudo-inverse modifiée :

$$\begin{aligned} J &= \begin{bmatrix} U_r & U_s \end{bmatrix} \begin{bmatrix} D_r & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} V^t && (1.39)\\ &= U_r \begin{bmatrix} D_r & 0 & 0 \end{bmatrix} V^t, && (1.40) \end{aligned}$$

donc

$$\boxed{J^+ = V \begin{bmatrix} D_r^{-1} \\ 0 \\ 0 \end{bmatrix} U_r^t.} \tag{1.41}$$

La méthode est exposée dans le pseudo-code donné algorithme 1.

Algorithme 1 : Cinématique inverse

Données : H_d : position désirée du repère de contrôle
$\xi_{i/0(i)}$: les axes de la structure

Résultat : H : position courante du repère de contrôle

$q = 0$: variété de configuration
H : position à l'initialisation (par cinématique directe)
répéter
  $\delta x = \log(H_d H^{-1})$
  pour *les m articulations* **faire**
    $\xi_{i/0(0)} = Ad_{0_{H_i}} \xi_{i/0(i)}$
  fin
  $J = \left[\begin{array}{ccc} \xi_{1/0(0)} & \dots & \xi_{m/0(0)} \end{array} \right]$
  $[U, \Sigma, V] = svd(J)$
  D_r : restriction de Σ aux r valeurs singulières non nulles
  U_r : restriction de U aux r valeurs singulières non nulles
  $J^+ = V \left[\begin{array}{c} D_r^{-1} \\ 0 \\ 0 \end{array} \right] U_r^t$
  $\delta q = J^+ \delta x$
  $H = \int \delta q$, cette étape d'intégration sera détaillée en 2.2.1.2.
jusqu'à *ce que l'utilisateur arrête la simulation*;

Commentaire : *Nous avons envisagé les $\xi_{i/0(i)}$ comme étant les axes de liaisons unitaires, nous élargissons cette hypothèse réductrice en annexe D - pour le cas des rotules. Dans le cas général, $\xi_{i/0(i)}$ dépend du type de liaison et de son paramétrage.*

Dans le cas sur-contraint, il n'y a pas forcément de solution au problème. L'optimisation ne peut donc être posée avec la contrainte de (1.16). Le but est de satisfaire au mieux cette équation, mais il n'y a pas forcément de solution exacte. Pour tenter de s'en rapprocher le plus possible, nous allons résoudre le problème suivant (on se place dans le cas J de rang plein pour simplifier) :

$$\min_{\delta q} = \frac{1}{2} \| J\delta q - \delta x \|^2, \tag{1.42}$$

par Lagrange, à l'optimum nous avons :

$$(J\delta q - \delta x)^t J = 0 \tag{1.43}$$

$$J^t J \delta q = J^t \delta x \tag{1.44}$$

$$\delta q = (J^t J)^{-1} J^t \delta x \tag{1.45}$$

$$\boxed{\delta q = J^+ \delta x, \text{ avec } J^+ = (J^t J)^{-1} J^t.} \tag{1.46}$$

Les cas des singularités n'est pas détaillé, car le mannequin est fortement redondant. Il se pose de toutes façons dans les mêmes termes.

Utilisée hors ligne *(c'est-à-dire principalement avec les techniques de* keyframing *décrites en 1.2.2), la cinématique inverse demande des compétences d'artiste. Elle est très utilisée dans l'industrie du film, nous pourrons citer* Shrek®, *ou* Les Indestructibles®, *dont nous pouvons voir des images illustration 1.7. Mais également dans l'industrie du jeu, avec des logiciels* comme *Motion Builder*®, spécialisé dans l'animation

temps-réel pour le jeu vidéo. Nous verrons en 1.2.2 que suivant la manière avec laquelle nous mettons en œuvre cette technique de cinématique inverse, nous pouvons arriver à des contextes d'utilisation très différents.

FIG. 1.7 – A gauche le film *Shrek*® de DreamWorks®. A droite *Les Indestructibles*® de Pixar®. Les protagonistes y sont animés principalement en cinématique inverse.

Dynamique

Pour copier le comportement du monde réel, nous pouvons mettre en œuvre des modèles plus proches du monde réel. Pour établir le comportement des structures polyarticulées, nous passons, comme indiqué dans [SB01], par une minimisation dont le critère nous est dicté par le *principe du minimum d'énergie* [P.03]. En exprimant l'énergie (cinétique et potentielle) dans l'espace de configuration, nous pouvons définir le lagrangien du problème d'optimisation, qui nous donne une condition vérifiée à l'optimum. Après quelques manipulations algébriques nous obtenons :

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = \Gamma, \tag{1.47}$$

avec $M(q)$ matrice d'inertie du robot vue dans l'espace de configuration, $C(q,\dot{q})\dot{q}$, couples dus aux forces de Coriolis et aux forces centrifuges, $G(q)$ les couples dérivant d'un potentiel (comme les forces gravitationnelles) et Γ les couples dus aux efforts appliqués sur le robot (forces extérieures, frottement visqueux et frottement sec articulaire, moteurs...).

Une des techniques les plus simples de contrôle en dynamique est celle dite du *computed torque* [Kha02]. Les couples de commande Γ_{task} appliqués au robot sont donnés par :

$$\Gamma_{task} = M(q)\Big(\ddot{q}_d + b_q(\dot{q}_d - \dot{q}) + k_q(q_d - q)\Big) + C(q,\dot{q})\dot{q} + G(q) \tag{1.48}$$

$$= \underbrace{M(q)\ddot{q}_d + C(q,\dot{q})\dot{q} + G(q)}_{boucle\ ouverte\ (terme\ direct)} + \underbrace{M(q)\Big(b_q(\dot{q}_d - \dot{q}) + k_q(q_d - q)\Big)}_{correcteur}, \tag{1.49}$$

où q_d est la trajectoire articulaire désirée. Si nous souhaitons exprimer la trajectoire dans l'espace de la tâche (comme c'est souvent le cas), nous mettrons en œuvre une cinématique inverse au préalable.

Nous pouvons aussi utiliser un correcteur proportionnel dérivé seul sans terme direct [MLSS94], toujours exprimé dans l'espace articulaire :

$$\Gamma_{task} = b_q\Big(\dot{q}_d - \dot{q}\Big) + k_q\Big(q_d - q\Big), \tag{1.50}$$

comme ce contrôleur ne possède pas de terme direct, il ne peut pas assurer un positionnement exact. Nous pourrions lui adjoindre un terme intégral, mais cela peut entraîner une perte de passivité qui peut rendre le système instable. Plus souvent, une composante issue de la boucle ouverte est donc ajoutée aux Γ_{task} :

$$\Gamma_{task} = M(q)\ddot{q}_d + C(q,\dot{q})\dot{q}_d + G(q) + b_q\Big(\dot{q}_d - \dot{q}\Big) + k_q\Big(q_d - q\Big). \tag{1.51}$$

Jusqu'ici, le contrôle est exprimé dans l'espace articulaire. Si nous souhaitons exprimer la consigne dans l'espace de la tâche, différentes méthodes sont possibles :

- effectuer une passe préliminaire de cinématique inverse,
- écrire le contrôle directement dans l'espace de la tâche. Khatib [Kha95] a montré qu'il était possible d'exprimer la relation (1.47) dans l'espace de la tâche :

$$\Lambda(q)\ddot{x} + \mu(q,\dot{q}) + p(q) = W, \tag{1.52}$$

avec $\Lambda(q)$, $\mu(q,\dot{q})$ et $p(q)$ équivalents respectifs de $M(q)$, $C(q,\dot{q})\dot{q}$ et $G(q)$, mais vus dans l'espace de la tâche, W représentant les efforts extérieurs appliqués sur le robot.

Pour le contrôle, nous pouvons écrire dans l'espace opérationnel les contrôleurs exprimés auparavant dans l'espace articulaire [MLSS94]. Ainsi la méthode du *computed torque* à l'espace opérationnel s'exprime par l'équation suivante :

$$W_{task} = \Lambda(q)\Big(\ddot{x_d} + b_x(\dot{x_d} - \dot{x}) + k_x(x_d - x)\Big) + \mu(q,\dot{q}) + p(q) \tag{1.53}$$

$$\Gamma_{task} = J^t W_{task}, \tag{1.54}$$

avec W_{task}, les efforts de commande.

ATTENTION : l'écriture $(x_d - x)$ est un abus de notation destiné à faciliter la compréhension. De manière plus rigoureuse nous écririons $\delta x = \log(H^{-1}H_d)$.

L'extension du contrôleur proportionnel dérivé est également envisageable [Sal80] :

$$\boxed{W_{task} = b_c(\dot{x_d} - \dot{x}) + k_c(x_d - x).} \tag{1.55}$$

Cette loi de commande est appliquée au contrôle du mannequin dans [ZVDH03]. Nous l'utilisons d'ailleurs en 2.2.1 dans notre propre contrôleur. Elle est à rapprocher de la commande en impédance de Hogan [Hog87].

1.2.1.3 Contrôle haut niveau

Edition de mouvement

Les méthodes d'édition de mouvement (*motion editing* en anglais) permettent de modifier des mouvements obtenus au préalable. Ils pourront ainsi être réutilisés dans des conditions différentes ce celles dans lesquelles ils ont été obtenus, ou transformés dans le but de leur ajouter des mouvements secondaires... Nous distinguons deux grandes familles d'utilisation : l'adaptation de mouvements (*motion adaptation*) et le *motion retargetting* (terme anglais difficilement traduisible). La première permet de modifier le mouvement pour l'adapter à de nouvelles conditions, d'environnement notamment. La seconde permet de modifier le squelette sur lequel le mouvement est appliqué.

Les premières techniques d'adaptation de mouvement utilisent des méthodes issues du traitement du signal, sur les signaux des paramètres articulaires [BW95], [WP95] : filtrage, décomposition fréquentielle,

décomposition temps-fréquence, interpolation, look-up tables... Elles permettent de modifier le mouvement directement sur les signaux articulaires. On peut accentuer la *nervosité* du mouvement (en accentuant les hautes fréquences), au contraire rendre les mouvements posés, ou mélanger deux mouvements par interpolation...

Ces méthodes sont pour la plupart articulaires, aussi nous ne maîtrisons pas toujours parfaitement le sens de nos modifications et celles-ci peuvent amener à violer des contraintes. Aussi [GL98] introduit un algorithme permettant d'effectuer des modifications sur le mouvement, tout en respectant certaines contraintes opérationnelles. Grâce à cette méthode, nous pouvons, par exemple, modifier la position des mains alors que les pieds sont contraints à leur position initiale. La différence avec la cinématique inverse est que l'optimisation est faite sur toute la séquence de mouvement (on parle d'*optimisation espace-temps*). La différence entre le mouvement initial et le mouvement résultant est minimisée, tout en respectant les contraintes, ce qui permet de conserver les caractéristiques du mouvement initial (comme la démarche par exemple).

Le même auteur utilise la même technique pour le *motion retargetting* [Gle98]. Dans ce cas on ne modifie plus les contraintes opérationnelles identifiées, mais le squelette sur lequel le mouvement est appliqué. C'est-à-dire que nous modifions le squelette tout en respectant des contraintes en position de repères de contrôle. Nous voyons l'utilité d'une telle approche sur la figure 1.8. Ces deux dernières méthodes fonctionnent bien et trouvent leur utilité dans un contexte infographique. Elles ont été implémentées au sein du logiciel Maya® [Ali].

Fig. 1.8 – *Retargetting* en cours de mouvement, d'après [Gle98].

Les méthodes vues jusqu'à présent sont cinématiques, il existe également des méthodes dynamiques d'édition de mouvement [PW99], [SHP04]. Mieux, certaines permettent de rendre physiquement réaliste un mouvement qui ne l'était pas. Le principe du premier article [PW99] est de trouver le mouvement physique le plus proche du mouvement initial, les paramètres peuvent alors être modifiés et le mouvement final enfin reconstruit. Le problème est lourd d'un point de vue calculs. Le mouvement initial est donc d'abord appliqué sur un squelette à la cinématique simplifiée. Le mouvement physique le plus proche est ensuite recherché par un problème d'optimisation (espace-temps) dont le critère est une distance entre le mouvement résultant et le mouvement initial, sous contrainte de respect de la dynamique. Le deuxième article ([SHP04]) réduit aussi la dimension de l'espace des solutions, mais en faisant l'observation que les mouvements sont souvent coordonnés. Il postule grâce à des observations biomécaniques que quand une jambe bouge, le mouvement du bras opposé est connu. Le mouvement est donc reparamétré.

Compromis attitude / contrainte

Comme indiqué sur la figure 1.9, il n'est pas possible de conserver tous les attributs du mouvement lors d'un *retargetting*. Les morphologies différentes de l'acteur et de l'humain virtuel nous amènent à donner la priorité à la conservation de *l'attitude*, ou à la conservation des *contraintes* principalement cartésiennes [SLGS01] (les positions désirées des repères de contrôle dans notre cas).

S. Ménardais [MMA01] propose une méthode donnant lieu à un compromis intéressant entre attitude et contraintes. Il met d'abord en œuvre un facteur d'échelle sur le mouvement, fonction des différences relatives entre morphologie de l'acteur et morphologie du mannequin - c'est dans ce cas un *retargetting*

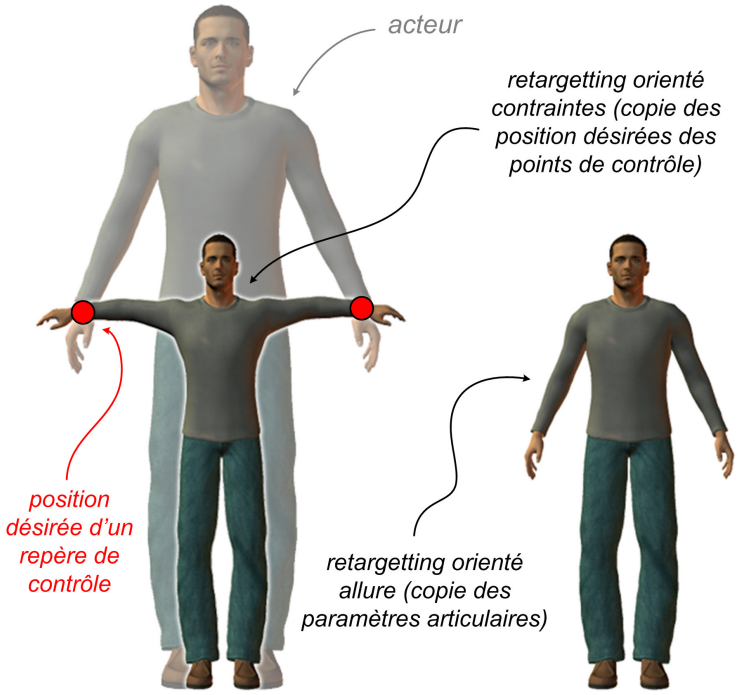

FIG. 1.9 – Comparaison du *retargetting* orienté contraintes (à gauche) et du *retargetting* orienté allure (à droite). Les repères de contrôle sont symbolisés par des points, par souci de simplification de l'illustration.

orienté attitude. Puis il adapte les trajectoires de manière à respecter au mieux les contraintes cartésiennes initiales (position des pieds, des mains...). Les résultats probants sont illustrés dans l'application *MKM* [MDAK].

Notons que pour le *retargetting* sur un squelette de morphologie différente, la position d'initialisation est importante également. C'est le cas par exemple, si nous souhaitons que le mannequin ait les mains jointes lorsque l'acteur a les siennes jointes également. Dans le cas où on prend une position d'initialisation classique (exemple, figure 1.10a), le *retargetting* du mouvement peut mener à des cas où, l'acteur ayant les mains jointes, celles du mannequin sont disjointes. Dans ce cas, la mise en correspondance entre monde virtuel et monde réel, à l'init, devra tenir compte de cette contrainte supplémentaire. On peut alors décider de mettre en œuvre une position d'init comme indiqué illustration 1.10b.

Méthodes procédurales

Les *méthodes procédurales* permettent de générer entièrement un mouvement. Il s'agit souvent de mouvements répétitifs (comme des cycles de marche), sur lesquels l'opérateur n'a que peu de moyens de contrôle - cela rend la spécification d'autant plus simple. Ces méthodes proviennent en général d'observations de mouvements d'humains réels, desquelles des lois (souvent empiriques) sont extraites. Elles sont alors appliquées aux humains virtuels. Le terme *procédural* fait référence à ces lois ou procédures. Malgré son intérêt, il semble que ce sujet soit quelque peu tombé en désuétude, peu d'études récentes lui sont consacrées.

Boulic et les Thalmann ont étudié la biomécanique des marches humaines, afin d'en retirer des paramètres et relations permettant de contraindre leur générateur de marche [BMTT90]. Aidés du schéma qu'ils introduisent (figure 1.11), ils génèrent une marche anthropomorphe par l'entremise de cinématique inverse.

Laszlo et Van de Panne [LVdPF96] introduisent un algorithme permettant de générer une séquence de marche tout en contrôlant l'équilibre par le biais d'une notion qu'ils appellent *limit cycle control*. Bien que

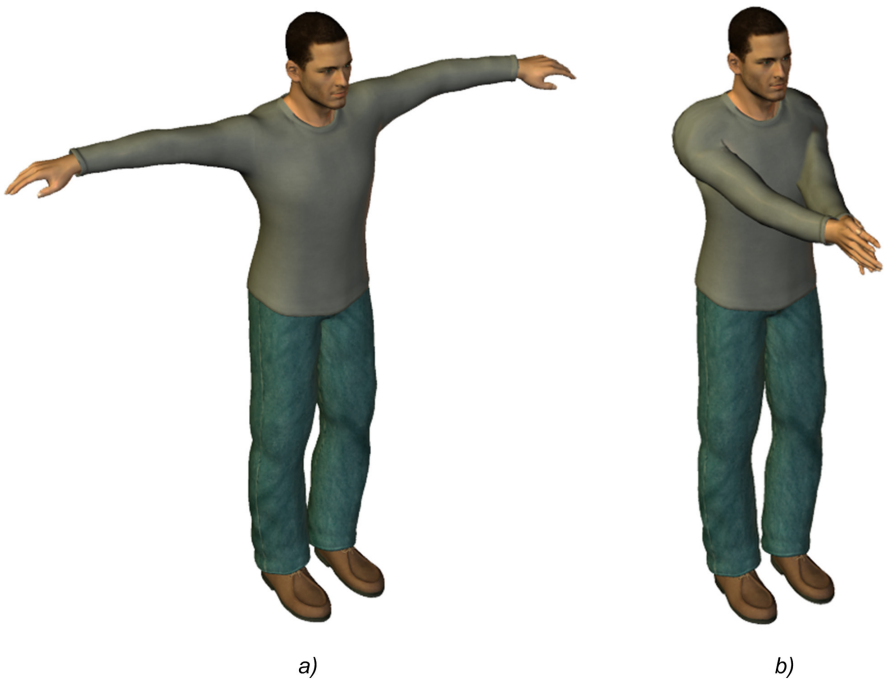

Fig. 1.10 – Différentes positions d'initialisation.

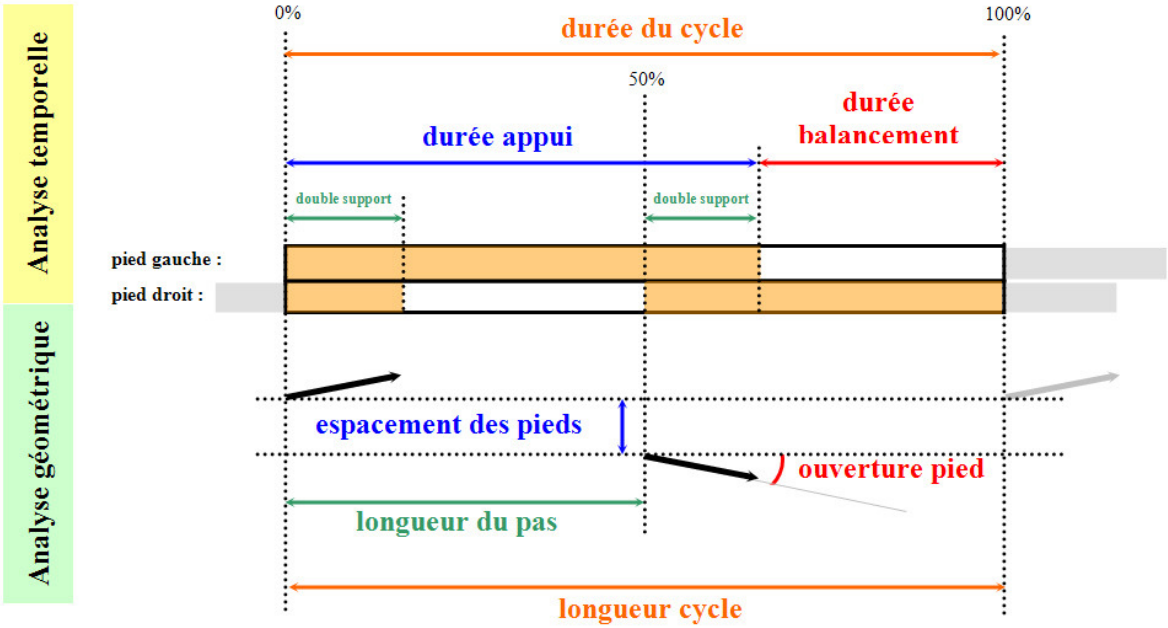

Fig. 1.11 – Analyse d'un cycle de marche, d'après [BMTT90].

la marche générée montre quelques côtés "automatiques", les résultats sont convaincants. L'algorithme se comporte bien face aux perturbations : les auteurs simulent l'action du vent, le contrôleur de cycle limite penche alors le mannequin pour conserver l'équilibre.

Une technique intéressante en matière de génération de marche est apportée par [VDP97]. Sa méthode permet de générer des séquences de marche à partir de marques de pas. L'opérateur place des marques de pas, ainsi que des indications temporelles, sur un sol virtuel, où il veut que le personnage numérique pose les pieds (figure 1.12). Ces marques de pas sont utilisées pour générer la trajectoire du bassin (car c'est souvent la racine du squelette), dont la trajectoire des hanches est déduite. Les marques de pas peuvent également être générées de manière procédurale à partir de la trajectoire du bassin spécifiée par l'opérateur. La configuration de la jambe en appui est donnée par cinématique inverse. La trajectoire de l'autre jambe est générée suivant une courbe de forme pré-établie dont les paramètres sont adaptés au pas courant. Des mouvements secondaires peuvent être ajoutés aux jambes pour personnaliser la démarche, mais aussi aux bras (de manière là aussi procédurale). Cette technique est utilisée dans le module *Character Studio*® de *3DSmax*®.

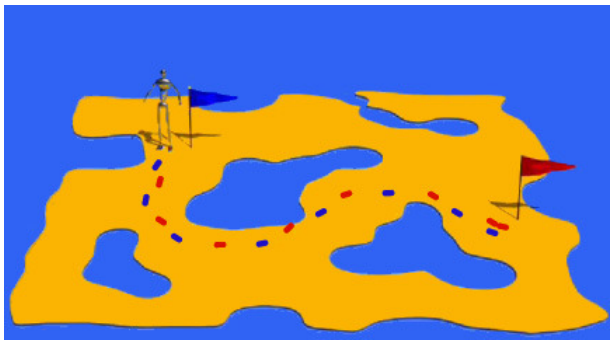

FIG. 1.12 – Génération automatique d'une marche à partir de marques de pas, d'après [VDP97].

Il existe aussi des techniques dynamiques. Leurs auteurs ne revendiquent pas la qualification *procédurales*, car leur caractère dynamique est prépondérant, mais elles sont bien du même ordre. Hodgins [HW98] s'appuie sur le même type d'observations biomécaniques que dans les cas cinématiques précédents, mais une fois les trajectoires articulaires générées, une couche dynamique est ajoutée. Cette couche supplémentaire calcule les couples qui permettent de générer le mouvement correct du mannequin. Les mouvements qu'elle simule sont complexes (course à pied, vélo, ou cheval d'arceau...), aussi ceux-ci sont découpés en sous-mouvements plus simples, dont elle gère les transitions par une machine d'état. Les résultats, en relation avec les Jeux Olympiques d'Atlanta de 1998, sont probants.

Planification de trajectoire

Les méthodes de recherche de chemin (*path planning* en anglais) permettent de générer automatiquement une trajectoire dans un environnement encombré. La notion d'encombrement est vue au sens large, l'environnement peut être encombré d'autres humains virtuels. Ces techniques sont souvent de nature probabiliste [HLK05] et sont exprimées dans l'espace de configuration. Beaucoup de ces méthodes sont issues des travaux pionniers de J.C. Latombe [Lat91]. Dans un premier temps une carte (appelée *probabilistic roadmap*) montrant les régions de l'espace de configuration respectant les contraintes, est construite, puis un chemin est recherché dans cette carte. Le principe général est le suivant :

- la variété de configuration est échantillonnée au hasard, c'est de cette étape que découle la nature probabiliste de la méthode
- pour chaque échantillon, on détermine s'il respecte les contraintes environnementales
- un chemin est recherché d'échantillon en échantillon, entre une configuration de initiale et une configuration finale spécifiée par l'opérateur. C'est souvent un plus court chemin suivant un critère donné
- une étape permettant de raffiner le mouvement obtenu peut être ajoutée [KAAT03]

Les travaux suivants [SLCPar], [EAL05] améliorent cette méthode de base, permettant d'organiser la construction de la roadmap, ou la recherche du chemin, de manière à les accélérer. Ainsi [SLCPar] facilite la recherche de solutions dans les cas très contraints, en amincissant le robot de manière artificielle,

une seconde étape permet d'éliminer les chemins non-réalisables en remettant sa véritable géométrie au robot. [LS02] introduit une technique permettant de gérer la roadmap dans les cas d'environnement mobile.

[KAAT03] applique les méthodes de ce type au cas du mannequin, pour des tâches de saisie d'objets. Il met en œuvre une méthode de réorganisation de la carte dans le cas des environnements mobiles. Leur méthode est temps-réel sur les vingt deux degrés de liberté de leur mannequin. C'est une réussite, comme nous pouvons en juger sur la figure .

Les travaux de l'équipe de J.P. Laumond ont mené à la possibilité de planifier des trajectoires de marche tout en évitant les obstacles [EAL05], [PLS03], [BELF05]. Il observe les mouvements d'un véritable humain et les modifie (voir section sur l'*Edition de mouvement*) de manière à prendre en compte les obstacles. Le personnage virtuel peut ainsi se baisser quand il passe sous une arche, éventuellement modifier la position d'un objet qu'il saisit... Ses résultats sont très concluants, ce que montre la figure 1.13, ils ont d'ailleurs fait l'objet d'une commercialisation dans le domaine de l'ingénierie (logiciel Kineo® [Kin]). J.J. Kuffner [SK] parvient à des résultats semblables.

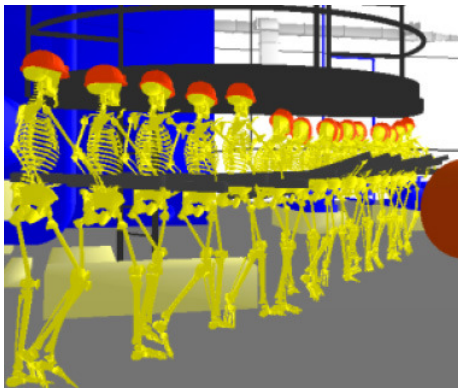

Fig. 1.13 – Un humain virtuel incline la tête, pour passer sous un obstacle, d'après [AEJ04].

P. Chedmail [CCC03], [Ler00] a appliqué, avec succès, les méthodes multi-agents au *path planning* pour le cas de pièces et de mannequins. Différents agents indépendants sont mis en œuvre, qui gèrent chacun une fonctionnalité (asservissement d'un repère de contrôle sur une position de consigne, évitement d'obstacles, orientation de la tête...). L'activité de chaque agent est pondérée par un taux d'activité, spécifié par l'opérateur. Un équilibre s'établit alors, entre l'activité de tous les agents [MCR02], [Mai03]. Il applique la méthode à la simulation de tâches de maintenance, notamment à des systèmes aussi complexes que des réacteurs d'avions.

[SKG05], du laboratoire de Mike Gleicher, élargit cette recherche de chemin au cas des foules. Ils peuvent générer des mouvements de foule évitant les collisions entre personnages et les collisions avec l'environnement. La méthode peut traiter jusqu'à trois cent mannequins en temps-réel pourvu que l'espace entre les personnages soit suffisamment large. Les mouvements générés sont ensuite raffinés par le biais d'une recherche dans un graphe de mouvement [KGP02].

1.2.1.4 Humains virtuels interactifs

Dans notre contexte, nous entendons par le terme *interactif*, la possibilité laissée à l'humain virtuel d'interagir avec l'environnement, le manipuler, ou modifier son état. Le simple fait d'ouvrir une porte, d'appuyer sur un bouton, ou de saisir un objet, amène à mettre en œuvre des techniques d'interaction.

En matière d'interaction virtuelle deux écoles s'opposent [Fuc03], [Che05] :

– Le première qui consiste à mettre en oeuvre les techniques virtuelles de manière à reproduire au

mieux les comportement du monde réel. Nous parlerons d'*interaction naturelle*.
– La seconde qui consiste à utiliser les potentialités du virtuel pour réaliser les mêmes interactions que précédemment mais d'une manière différente de celle réalisée dans le monde réel. Cette approche peut permettre de simplifier l'interaction, ou de pallier un manque de réalisme relatif aux techniques utilisées. Quand la nouvelle méthode d'interaction requière un apprentissage, on parle de *métaphore d'interaction*([Fuc03]).

Exemple de métaphore d'interaction

M. Chevaldonné [Che05] propose des métaphores d'interaction adaptées au cas de la simulation de cockpits virtuels. Les applications sont effectuées sur des avions de marque Airbus®. Ainsi, [Che05] remarque que la manipulation des boutons d'un cockpit virtuel peut être malaisée du fait de la précision requise, et du manque de méthodes adaptées à l'*interaction naturelle*. Aussi il propose, entre autres, une *métaphore d'interaction* facilitant la spécification du bouton à manipuler, d'une part, et sa manipulation d'autre part.

Pour spécifier le bouton à manipuler, [Che05] propose d'adjoindre à l'index droit un *faisceau de pointage virtuel*, dont l'intersection avec des éléments sensibles de l'environnement (les boutons manipulables) permet de sélectionner l'objet d'interaction désiré comme indiqué figure 1.14. La manipulation du bouton est faite à distance, en tournant la main dans un sens ou dans l'autre, le bouton tourne alors automatiquement dans le sens indiqué.

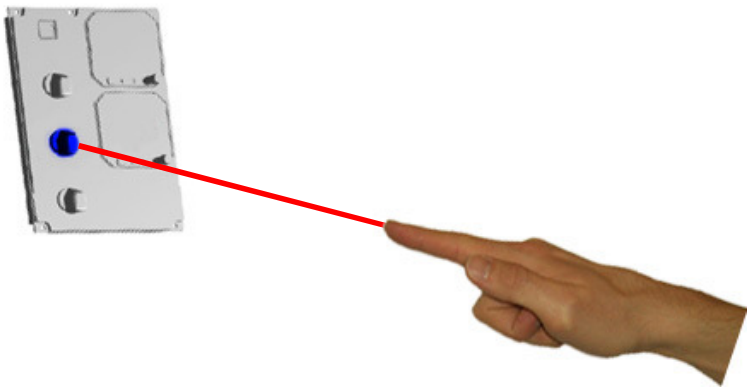

FIG. 1.14 – Exemple de *métaphore d'interaction* : adjonction d'un faisceau de pointage à distance à l'index, pour la sélection du bouton à manipuler (qui apparaît ici en bleu).

Interaction naturelle

On pourra noter le peu de méthodes répondant aux besoins en terme interaction des humains virtuels avec l'environnement.

Schmidl et Lin [SL04], de l'Université de Caroline du Nord, mettent en œuvre un modèle hybride utilisant la cinématique inverse pour déterminer la réaction du mannequin au contact et un modèle de physique basée sur des impulsions aux contacts, pour l'environnement. La figure 1.15 illustre les possibilités ainsi offertes en matière d'interaction. Le caractère hybride de leur approche implique la perte de la nature physique de la simulation.

Zordan et Hodgins [ZH02], également de l'université de Californie, proposent un algorithme permettant de donner des coups à un humain virtuel et d'obtenir la réaction du système. Pour cela, ils modifient les gains du contrôle durant la simulation. Comme le disent les auteurs, la méthode, qu'ils appliquent à un combat de boxe virtuel (voir figure 1.16), est malheureusement sujette aux instabilités, et n'est pas temps-réel.

Sentis et Khatib [SK04], de l'université de Stanford, apportent une application de l'humain virtuel

Fig. 1.15 – Un humain virtuel s'appuie sur un mur : les paramétres articulaires du bras sont ajustés par cinématique inverse pour éviter l'interpénétration. D'après [SL04].

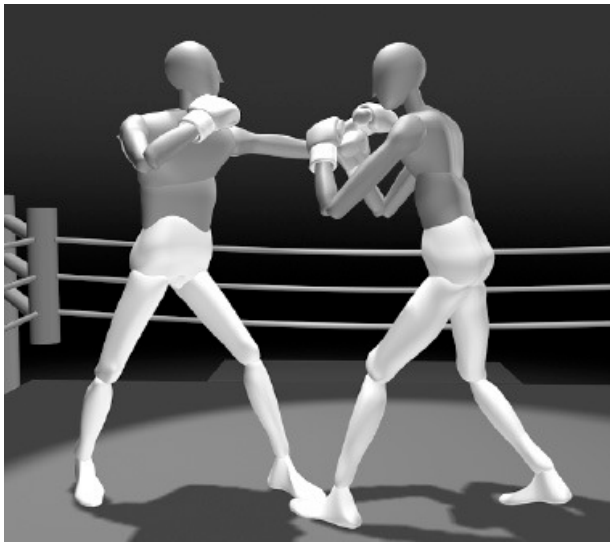

Fig. 1.16 – Exemple d'interaction entre deux mannequins. D'après [ZH02].

interactif, des plus intéressantes. Elle est d'ailleurs celle qui nous semble la plus prometteuse des trois citées ici. L'interaction se fait de manière naturelle par le biais de forces de contact (vecteur naturel de l'interaction). Les exemples d'application ne sont pas très complexes pour l'heure, mais la méthode semble extensible à des cas plus intéressants. Une illustration des possibilités laissées par le système d'animation est donnée illustration 1.17.

L'interaction des humains virtuels avec leur environnement est un sujet d'intérêt dans la communauté scientifique, comme en témoignent les trois applications précédentes. Il semble cependant qu'aucune ne soit utilisable dans des cas complexes à l'heure actuelle.

1.2.2 Spécification du mouvement

Keyframing

Le *keyframing* est souvent utilisé conjointement avec de la cinématique inverse. Elle est très utilisée

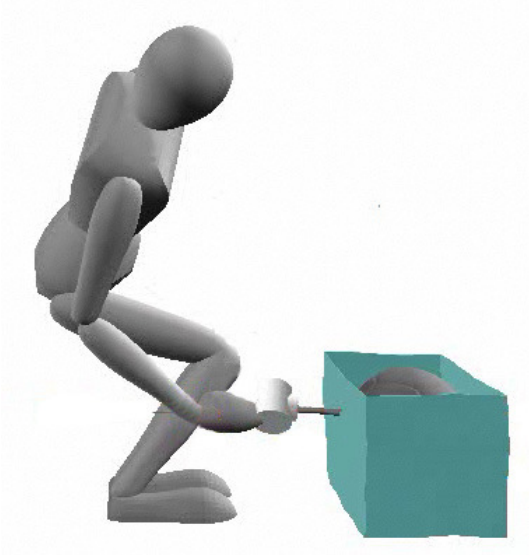

FIG. 1.17 – Exemple d'interaction d'un mannequin avec l'environnement. Le mannequin pousse sur une caisse fixe avec un outil. D'après [SK04], et [Sta].

en infographie [Autb], [Chab]. Beaucoup des animations du cinéma ou du jeu sont obtenues de cette manière. Elle est trouvée également dans les modules de mannequin à caractère industriel [Dasa]. La méthode comporte deux étapes :

- **la mise en place de poses clés :** les poses clés sont des postures de mannequin à un instant donné. Elles sont spécifiées par l'animateur, qui détermine la position des repères de contrôle, ainsi que d'autres paramètres internes (voir 2.2.1) de manière à obtenir l'attitude désirée du squelette. Dans les grands studios, les paramètres de contrôle du mannequin sont rassemblés dans un *character setup*, ou *character rig* défini par le *character TD* (directeur technique spécialiste des personnages). Ces paramètres peuvent avoir plusieurs degrés de liberté (six pour la position d'un repère de contrôle) et ils peuvent être nombreux. Or le dispositif privilégié pour la saisie de la posture est la souris, ou la tablette graphique, dispositifs 2D. Les animateurs ne peuvent donc gérer tous les degrés de liberté de leur setup de manière simultanée, ils le font donc les uns après les autres. Le processus est relativement laborieux et demande du savoir-faire.
- **le timing :** le timing est la gestion de l'espacement temporel entre les poses clés. C'est cette notion qui va donner son rythme au mouvement. Nous comprenons aisément que la gestion du timing requière une bonne dose de savoir-faire, plus généralement du ressort des artistes plutôt que des ingénieurs. Les postures intermédiaires entre deux poses clés sont générées de manière automatique par interpolation[4].

Cette technique du *keyframing* , si elle est fastidieuse et demande du savoir-faire, n'en donne pas moins des résultats corrects dans les mains de béotiens[5] et de très bons résultats dans celles d'experts.

Afin d'augmenter le nombre de degrés de liberté spécifiables en simultané et par là même de diminuer la charge de travail des animateurs, des *trackballs*, ou *spacemouses* sont introduits (termes pour lesquels nous ne connaissons pas d'équivalent français). Ces dispositifs dont nous pouvons voir quelques exemples figure 1.19, permettent de spécifier le mouvement dans les trois translations et les trois rotations de l'espace. Une trackball permet donc de contrôler les six degrés de liberté d'un repère de contrôle en même temps. Il est possible de lui adjoindre du retour d'effort, pour augmenter la richesse des informations

[4]Dans l'animation traditionnelle cette notion de poses clés et de poses intermédiaires existe également (sous un autre nom), les poses clés sont dessinées par un animateur principal et les poses intermédiaires par des animateurs subalternes.

[5]Les personnes en charge de l'animation dans les bureaux d'étude n'ont pas souvent la formation artistique nécessaire pour pouvoir être qualifiées autrement.

Fig. 1.18 – Keyframing : les postures bien marquées sont des poses clé du mouvement de marche, les postures transparentes sont des poses intermédiaires.

retournées à l'utilisateur (voir 4.4.4). Des exemples de dispositifs à retour d'efforts sont donnés figure 1.19.

Fig. 1.19 – La *Space Pilot* ® de 3D Connexion® (à gauche). Le *Phantom Omni* ®, dispositif haptique de Sensable® (à droite).

Mais un setup de personnage comporte bien souvent beaucoup plus de degrés de liberté, alors même si les algorithmes de contrôle sont temps-réel, la spécification ne l'est pas, à moins de multiplier les dispositifs d'entrée. Leur mise en œuvre pratique peut cependant être difficile quand leur nombre augmente. Aussi nous utilisons la solution décrite au point suivant.

Capture de mouvement

La *motion capture*, ou *capture de mouvement* est un processus d'observation du mouvement d'humains réels (on parle d'*acteurs*). Dans notre cas ils seront appliqués à des personnages virtuels dans le but de les animer. Il existe plusieurs technologies de dispositifs de capture de mouvements [FMP01] : optique, électromagnétique, électromécanique (exosquelettes qui peuvent également conjuguer du retour d'effort), fibres optiques, ultrasoniques...

Nous utiliserons un dispositif optique - simple à mettre en œuvre et fiable - comme le système visible figure 1.20. Des marqueurs réfléchissants sont positionnés sur un acteur observé dans l'infrarouge par des caméras. Un pré-traitement est fait à la sortie des caméras pour localiser les marqueurs sur les images, puis des étapes de mise en correspondance et de reconstruction vont permettre de déterminer la position

6D de chaque marqueur.

FIG. 1.20 – Dispositif de capture de mouvement conçu par *Vicon*®.

Les techniques optiques souffrent d'avoir une portée dite, elle aussi, "optique", c'est-à-dire que s'ils sont occultés, les marqueurs ne sont plus visibles. Pour pallier ce problème nous pouvons envisager d'élaborer des modèles de comportement, des prédictions, ou de faire de la fusion entre données optiques et données issues d'une autre technologie.

Il existe également des déclinaisons sans marqueur des systèmes optiques, mais ceux-ci sont aujourd'hui encore à l'état de prototypes, [DC01].

Notons que la capture de mouvement peut être utilisée :

- soit en apposant un marqueur par membre à animer pour ne pas avoir de problème inverse à résoudre
- soit avec une cinématique inverse temps-réel. Ce dernier processus est beaucoup plus flexible comme nous le verrons
- ou en enregistreant des mouvements à retoucher par la suite en production.

Bibliothèques de mouvements

L'utilisation des bibliothèques de mouvements est fondée sur la composition de mouvements simples (ou échantillons, ou micro-mouvements, ou clips) contenus dans une base de données (ou bibliothèque) en vue de créer des mouvements plus complexes [CMU]. Il est envisageable de générer des transitions entre les échantillons de mouvements, de manière à pouvoir les assembler sans qu'apparaisse d'artefact. Le principe de la construction de la bibliothèque est d'échantillonner l'espace des *micro-mouvements* possibles. Bien souvent le micro-mouvement exact souhaité n'est pas disponible dans la bibliothèque. Si la bibliothèque est bien construite, un échantillon qui lui est proche peut être trouvé et dans ce cas, les techniques d'édition de mouvement prennent tout leur sens.

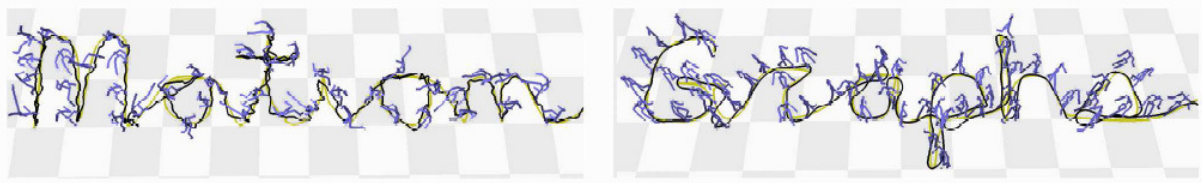

FIG. 1.21 – Exemple de mouvement obtenu en parcourant un graphe de mouvement, d'après [KGP02]. L'humain virtuel accomplissant le mouvement est représenté en bleu.

L'échantillonnage des micro-mouvements possibles est une tâche longue et jamais parfaitement accomplie. La plupart du temps, des données d'animations précédentes, ou des données de capture de mouvement sont récupérées et les micro-mouvements jugés opportuns en sont extraits. La création de la bibliothèque et ensuite la recherche des échantillons pour la reconstruction d'un mouvement, obligent donc à gérer d'énormes quantités de données. Les développements principaux quant aux bibliothèques

de mouvements sont donc orientés vers l'élaboration de techniques qui vont permettre de simplifier ces tâches. [BSP+04] et [KG04] proposent des schémas permettant de segmenter de manière automatique des mouvements complexes en micro-mouvements pour les inclure dans les bibliothèques. Pour ce qui est de la recherche dans la base de données, [RP04], and [KGP02] introduisent les graphes de mouvement (*motion graphs* en anglais), technique de recherche d'un chemin dans un graphe dont les nœuds sont les échantillons de mouvement. Le chemin dans le graphe détermine le mouvement complexe final. La technique permet de spécifier un mouvement en indiquant simplement la trajectoire désirée du bassin. Le graphe est alors traversé de manière à générer la trajectoire la plus proche, comme nous pouvons le voir figure 1.21. L'algorithme est surtout adapté à des mouvements de marche, le cas de la manipulation d'un environnement est plus complexe.

Langages comportementaux

Le plus simple des moyens de spécification d'un mouvement pour un opérateur, est sa description en langage naturel. ACUMEN® [AKA+02], [KT99], l'AML [AKM+02], ou HPTS++ [DDLT02] apportent des solutions dans ce sens. Ils introduisent des langages de description du comportement d'humains virtuels. Certains d'entre-eux, comme le CML® [AKM+02], ou EMOTE® [BAZB03] permettant même de gérer l'influence de l'état émotionnel du personnage sur ses mouvements.

Deux points méritent une attention particulière [AKM+02] :

- **la puissance d'expression du langage :** le langage doit pouvoir recenser tous les comportements humains et les traduire dans une représentation non-ambiguë. C'est le cœur du problème et étant donné la richesse des comportements humains, il est très complexe.
- **la simulation :** elle doit être capable de reproduire les comportements descriptibles dans le langage. Cette problématique est d'un autre ordre (elle relève du contrôle bas niveau, avec lequel le langage doit s'interfacer (voir 1.4.1.) et est plus proche de nos préoccupations.

Ces langages peuvent même être mis en œuvre dans des contextes réactifs, ou l'humain alors *autonome*, réagit aux sollicitations extérieures, d'un utilisateur. C'est ce type d'agents qui est utilisé par Microsoft® pour ses systèmes de chat dans des mondes 3D. Les participants à la discussion y sont représentés par leurs *avatars* (nom spécifique donné à leurs humains virtuels) [Cor98]. Ces modèles *réactifs*, ou *comportementaux*, aux perspectives prometteuses, semblent avoir été introduits un peu tôt. Ils ne parviennent pas encore à capter toute les subtilités du comportement humain et semblent quelque peu délaissés par les utilisateurs, après l'intérêt qui leur a été porté lors de leur introduction. Des modèles expérimentaux proposent des solutions plus avancées [KT99], [AKM+02].

A la vue de la description que nous avons faite de la spécification du mouvement désiré, nous pouvons distinguer les méthodes temps-réel*, qui permettent de spécifier le mouvement alors qu'il est généré, des méthodes* hors ligne *qui demandent une phase de préparation. Cette vue du problème est celle des acteurs du monde des effets spéciaux, du film et des jeux vidéo. Nous lui préférons le point de vue des roboticiens, basé sur le* degré d'automatisation *de la méthode. Le nombre de degrés de liberté de l'humain virtuel étant de manière générale, supérieur au nombre de degrés de liberté spécifiables simultanément, nous envisageons trois réponses au problème :*

- *plus le dispositif d'entrée est rudimentaire (ex : la souris, dispositif 2D), plus le travail de l'animateur est de bas niveau et fastidieux. Les consignes des degrés de liberté sont spécifiées les unes à la suite des autres.*
- *nous pouvons également choisir d'augmenter le nombre de degrés de liberté spécifiables en parallèle, en utilisant des dispositifs d'acquisition élaborés. Nous avons déjà abordé cette méthode, en proposant de substituer à la souris, des dispositifs d'acquisition de type* trackball*, ou capture de mouvement.*
- *les deux réponses précédentes imposent à l'opérateur de spécifier beaucoup des degrés de liberté du mannequin. Nous pouvons imaginer de gérer automatiquement certains d'entre-eux (par des couplages entre degrés de liberté, des contraintes, de la génération de trajectoires...). Alors plus le dispositif d'entrée sera rudimentaire, plus le nombre de degrés de liberté à gérer de manière automatique devra être important : le processus est automatisé. Cette constatation a d'ailleurs amené*

certains chercheurs à introduire des méthodes novatrices en matière de spécification de mouvement [AFO03], [IMH05]. Leur intérêt est leur processus d'acquisition du mouvement : simple et naturel, comme le dessin à main levée d'une esquisse de posture.

*Les méthodes très automatisées laissent moins de possibilités de création et d'*expressivité*, aussi elles sont peu appréciées des artistes. Dans un bureau d'étude, la problématique est différente. Si l'*expressivité *du mouvement est d'une importance non négligeable, elle n'est pas prépondérante. En ingénierie nous chercherons avant tout à vérifier qu'un mouvement est faisable, dans des conditions acceptables pour l'opérateur. Ces tests doivent pouvoir être réalisés de manière simple et rapide, par des non-spécialistes de l'animation de mannequins. Face à ce constat, nous chercherons à automatiser le plus de degrés de liberté possible, de manière à faciliter le travail de l'utilisateur du système d'animation.*

1.3 Prototypage virtuel

Le travail que nous présentons ici s'insère dans le contexte particulier de la *conception industrielle*. L'humain virtuel est l'extension naturelle d'une révolution commencée il y a une vingtaine d'années : le *prototypage virtuel*. Comme indiqué dans [Wat01], il se situe à la convergence de la simulation, de la CAO et de la réalité virtuelle. Il a permis d'introduire de nouvelles méthodes de conception, comme celle dite de l'*ingénierie concourante*. Tim Hodgson de *Comptek Federal Systems Inc* nous donne la définition suivante du prototypage virtuel [Hod98] (traduit de l'anglais) :

Le prototypage virtuel est une discipline d'ingénierie basée sur des technologies informatiques qui met en œuvre une modélisation mécanique du système étudié, sa simulation et la visualisation 3D de son comportement, dans les conditions d'opération du monde réel. Des optimisations et améliorations de la conception peuvent alors être étudiées et testées de manière itérative et ce avant même d'avoir construit le premier prototype physique.

Il existe un adage dans la communauté de la conception mécanique qui met en lumière de manière concrète l'intérêt des prototypes immatériels, elle est appelée *la règle des 10* : *le coût de résolution d'un problème qui aurait pu être évité dans la phase de conception est multiplié par 10 s'il est décelé sur maquette physique, par 100 s'il est trouvé après fabrication et par 1000, s'il est révélé par le client.* L'intérêt du prototypage virtuel est de minimiser les erreurs de conception dès que possible dans la phase de développement du produit. Traditionnellement, les ingénieurs concevaient le produit, réalisaient une maquette, l'analysaient et modifiaient le modèle. Les revues maquettes étaient rares du fait du temps de fabrication de la maquette et du coût de celle-ci, les modifications à faire pouvaient donc être importantes. Grâce au prototypage virtuel, les ingénieurs peuvent réaliser autant de tests qu'ils le souhaitent, les modifications ont donc une portée et des implications beaucoup plus réduites.

L'humain virtuel que nous mettons ici en œuvre doit s'insérer dans l'approche de l'*ingénierie concourante*. En première approximation, nous pouvons dire que les anciennes méthodes de conception de produits étaient séquentielles, alors que l'ingénierie concourante est simultanée [Bro99]. C'est-à-dire que les différents corps de métier devant intervenir dans la conception d'un objet intervenaient auparavant les uns après les autres, alors que l'approche concourante permet de paralléliser leurs interventions. Dans ce contexte, on ne dispose plus du produit réel, mais de son prototype virtuel. Le lecteur intéressé par cette problématique pourra se reporter à [Dur04a]. On pourra citer les projets *Cedix* [Ced] et *EnHance* [EnH] (portés entre autres, par de grandes entreprises du domaine aéronautique : EADS®, SNECMA®, Dassault Aviation®...), pour montrer l'intérêt suscité par ces méthodes. Cette ingénierie concourante, est d'ailleurs mise à profit dans des projets de large échelle (comme ce fut le cas pour la conception de l'A380 d'Airbus® par exemple).

Cette approche de conception est rendue possible par les innovations dans le domaine des technologies de l'information et de la communication de ces dernières décennies, comme le prototypage virtuel décrit ci-avant. Elle repose sur de nouvelles méthodes d'organisation et de management. Des méthodes rationnelles

doivent être mise en œuvre pour en tirer parti, comme les revues de maquette virtuelle[6], ou le déploiement de systèmes de PDM (Product Data Management)[7]...

Dans un tel contexte, le concept de maquette virtuelle n'est plus un simple assemblage dans un outil de CAO, mais un objet géré par le système de PDM, qui vérifie et assure l'intégrité de la maquette dans cet environnement de travail collaboratif et simultané.

Ces aspects collaboratifs impliquent de contrôler les flux de données, pour les mettre à disposition des parties intéressées. On distingue les données échangées entre des personnes et des machines et les données produites à partir d'autres données, grâce notamment à des bases de connaissances métier[8].

L'innovation dans les technologies de l'information et de la communication est destinée à supporter et développer le savoir-faire collectif, que l'on cherche à capitaliser. Elle porte, sur les domaines suivants :

- la simulation et l'analyse continue du produit virtuel, des premières esquisses jusqu'au modèle prêt à être fabriqué
- les flux de données techniques, des premières investigations, jusqu'aux utilisateurs finaux. On compte parmi ces données, les plans, nomenclatures...
- les cycles de connaissance, des premiers concepts, jusqu'au support de l'utilisateur final. Connaissance dont les concepteurs doivent pouvoir bénéficier, mais qu'ils doivent aussi pouvoir enrichir.

Cette vision des choses reste pour l'instant à l'état embryonnaire, de nombreux problèmes restant à résoudre. Nous allons nous concentrer sur un de ceux-ci : la simulation du produit et son analyse dans le cadre de la conception centrée sur l'humain (*human centric design*). Un produit est conçu pour satisfaire les besoins d'un utilisateur humain et ce dans toutes les étapes de son cycle de vie.

Puisque l'on utilise maintenant des prototypes virtuels, la relation directe entre humains et produit, qui avait anciennement lieu lors de la manipulation de la maquette physique, est perdue. Pour prendre en compte les contraintes du *Human Centric Design* dans le cadre de l'utilisation de maquettes virtuelles, une idée est de simuler les humains par le biais de leur pendant virtuel. Initialement les outils destinés à manipuler la maquette numérique (outils CAO...) ne permettaient pas de manipuler ces humains virtuels. Nous verrons qu'il existe maintenant des solutions commerciales pour leur simulation, mais que celles-ci ne sont pas pleinement opportunes. Du fait de la complexité du système constitué par des humains virtuels, ceux-ci sont difficiles et longs à mettre en oeuvre à l'heure actuelle. Nous souhaitons développer de nouvelles méthodes de contrôle destinées à leur utilisation.

Nous avons identifié le besoin quant aux humains virtuels industriels dans [RMA+04]. Ceci en passant en revue les différents cas d'utilisation d'un produit et en exprimant le besoin quant aux mannequins pour chacun d'eux - en restreignant notre étude au contrôle du mouvement. L'analyse des cas d'utilisation a été basée sur la vision du cycle de vie d'un produit (ou PLM) de l'entreprise EADS®, comme indiqué sur la figure 1.22.

Expression du besoin

En premier lieu apparaît l'intérêt des humains virtuels autonomes, dont le contrôle demande un minimum d'interaction avec l'opérateur. Il requiert seulement la spécification de buts complexes que nous nommerons *consignes de haut niveau*. C'est le type de consignes que l'on pourrait donner à un humain réel. Leur caractère complexe vient du fait qu'elles sont influencées par des paramètres non définis explicitement et relatifs aux us et coutumes et au contexte de l'opération à accomplir. Ainsi, si on demande à un ouvrier de visser un écrou sur une tige filetée, celui-ci saura, sans que qui que ce soit ne

[6] On parle plus aisément de DMU (Digital Mock-Up).

[7] Le PDM sert dans les cas suivants. Si différentes personnes travaillent en même temps sur la maquette, il faut surveiller qu'il n'y ait pas de modifications conflictuelles sur la maquette. Cette tâche de surveillance, particulièrement fastidieuse pour un humain, est le domaine d'activité privilégié des ordinateurs.

[8] Elles proposent par exemple, de manière interactive, des solutions techniques à certains problèmes, ex : comment assurer l'étanchéité au contact de deux produits, dans des conditions données... Ce petit exemple permet d'ailleurs de se rendre compte que l'on ne travaille plus en terme de formes géométriques comme c'était le cas avec le dessin technique sur planche, ou la CAO, mais en terme de fonctionnalités (étanchéité, liaisons...).

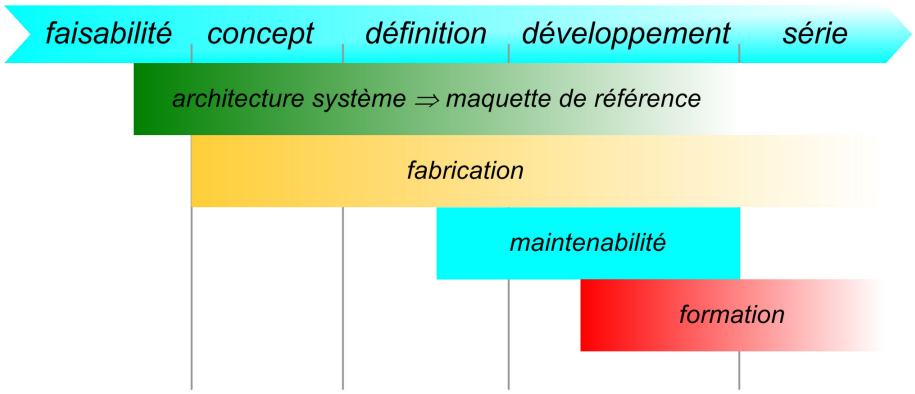

FIG. 1.22 – PLM simplifié, dans le contexte de l'ingénierie concourante, comme il est vu à EADS®.

le lui ait spécifié, qu'il faudra dans un premier temps qu'il aille chercher ses outils. La décomposition des consignes de haut niveau en consignes élémentaires plus simples (aller chercher les outils, tenir l'écrou sur la tige filetée, visser...) est d'autant plus difficile qu'aucun recensement de ces buts n'est envisageable, du fait de leur diversité.

Grâce à leur nature adaptative aux changements de l'environnement, les mannequins autonomes s'intègrent parfaitement dans le schéma flexible de l'ingénierie concourante. Grâce à cette méthode de conception, analyses de la maintenance, de l'ergonomie et autres processus peuvent être menées en parallèle. Si une modification de la maquette est proposée par un service, les autres services doivent être prévenus du caractère conflictuel de la modification, s'il y a lieu. Sans ce caractère autonome, remarquer qu'une modification peut-être conflictuelle n'est pas une tâche évidente et requiert beaucoup de discipline et d'organisation (surtout quand des produits aussi complexes qu'un avion, un hélicoptère, ou une voiture sont considérés, en interaction avec les non moins complexes humains virtuels). Grâce aux mannequins autonomes, ces tests peuvent être réalisés de manière automatique. Imaginons un service qui souhaite analyser la faisabilité d'opérations de maintenance, notamment l'accessibilité d'une pièce. La personne en charge du test pourrait simplement réaliser le test et analyser les résultats. Mais nous pouvons également imaginer de sauvegarder la gamme de maintenance. Et suivant les résultats d'une étude d'impact, faire générer automatiquement au PDM, les mouvements du mannequin relatifs à la maquette modifiée - de manière à vérifier son intégrité. Cette fonctionnalité nous paraît essentielle, mais les techniques manquent de maturité pour une autonomie complète.

Un autre enjeu des prototypes virtuels est qu'ils sont reconfigurables très rapidement. Lors des revues d'un projet, les problèmes à résoudre sont présentés et les différentes solutions analysées. Grâce à la maquette virtuelle, on peut tester différentes solutions les unes à la suite des autres, de manière à ce que les décideurs puissent trancher. Dans certains cas, on pourra même tester les solutions proposées au cours de la revue quelques instants après leur énonciations.

Les méthodes à base de capture de mouvement présentent un intérêt particulier. Les solutions trouvées de manière automatique sont étiquetées *virtuelles* dans nos esprits tant qu'elles ne sont pas testées. Il est possible d'être plus convaincant auprès des analystes ou preneurs de décision assistant à la revue : on peut leur offrir de tester le produit par eux-même par le biais de leur équivalent virtuel. Cette solution sera plus persuasive qu'aucune autre, surtout si elle laisse la possibilité à l'opérateur d'interagir avec la maquette virtuelle, c'est-à-dire de la manipuler. **L'interaction est au cœur de nos développements**.

Notons que jusqu'à présent, nous avons uniquement abordé les systèmes permettant de piloter un humain virtuel. Certaines tâches ne peuvent être accomplies par un humain seul, notre système doit donc être capable de piloter plusieurs humains virtuels. Ce caractère pluriel peut être envisagé de deux manières.

Dans le premier cas, le bon accomplissement d'une tâche nécessite plusieurs personnes. Imaginons par exemple que nous ayons à soulever une charge lourde requérant l'intervention de deux personnes. Dans ce cas, le travail et les mouvements doivent être bien organisés de manière à être efficace, nous l'appellerons *travail collaboratif* [Kha95]. C'est-à-dire que plusieurs personnes ont un but commun, qu'elles doivent atteindre grâce aux compétences et capacités de chacun. Ce groupe de travail cherche la synergie des efforts de chacun.

Fig. 1.23 – Déplacement d'un piano : deux robots et un humain interviennent, d'après [EAL05].

D'un autre côté, nous pouvons également rencontrer des situations complètement chaotiques dans lesquelles les personnes agissent de manière isolée, menant à des mouvements, désorganisés, désordonnés et inefficaces [THO+04]. C'est typiquement le cas des foules, dont l'intérêt est évident pour tester des situations d'urgence. Nous n'aborderons pas cette problématique ici.

Insistons également sur l'intérêt d'avoir des outils intégrés aux outils déjà mis en place dans les bureaux d'étude. Ceci afin de minimiser le nombre d'opérations nécessaires pour mener le travail à bien. On peut ainsi envisager d'intégrer le contrôle des humains virtuels dans des modules additionnels de gestion de PLM, de la même manière que la solution apportée par DELMIA® [Dasb].

Solutions utilisées dans les environnements industriels

D'un point de vue général, des solutions intéressantes existent déjà en réalité virtuelle pour résoudre certains problèmes. On pourra citer l'application Samira®, développée par le centre commun de recherche de la société EADS® [DGLC05], [Gui]. Elle permet grâce à un Virtuose®, dispositif de retour d'effort, de vérifier la montabilité de pièces, sur un produit en cours de développement, comme indiqué figure 1.24. L'opérateur ressent les efforts de contact par le biais d'un dispositif à retour d'effort non représenté ici.

L'extension directe de cette application est l'introduction de mannequins pour manipuler la pièce bleue. Cependant en matière d'humains virtuels et malgré les enjeux, les solutions adoptées dans les logiciels du commerce ne sont pas entièrement satisfaisantes. Les plus simples des solutions sont à base de cinématique directe et keyframing, qui obligent l'opérateur à piloter tous les degrés de liberté à la main, rendant le processus d'animation long et pénible. La cinématique inverse est donc utilisée pour simplifier quelque peu le processus. C'est la solution adoptée dans eM-Human de Tecnomatix [Teca], Safeworks [Saf] et Man3D [Ver03] (développé par le constructeur automobile Renault®). L'algorithme est souvent couplé à un dispositif de capture de mouvement pour rendre plus aisé l'acquisition des positions des cibles [Ver03], [Fig], [Erg].

La solution à base de bibliothèques de mouvements est très utilisée pour sa simplicité d'utilisation et la qualité des mouvements obtenus. Les mouvements sont alors adaptés, grâce aux algorithmes présentés en 1.2.1 à tout le spectre imaginable des morphologies de mannequins. En mélangeant des mouvements et calculant des transitions entre *clips*, on parvient à des résultats satisfaisants. La technique est utilisée

FIG. 1.24 – L'application Samira® à l'essai. Le cas d'application mis en œuvre ici permet de vérifier que la pièce manipulée (en bleu sur la figure de gauche) par le Virtuose®, est bien démontable du bras maître (dispositif auquel est accroché un réacteur - en jaune sur la figure de droite) d'un Airbus® A380® en situation.

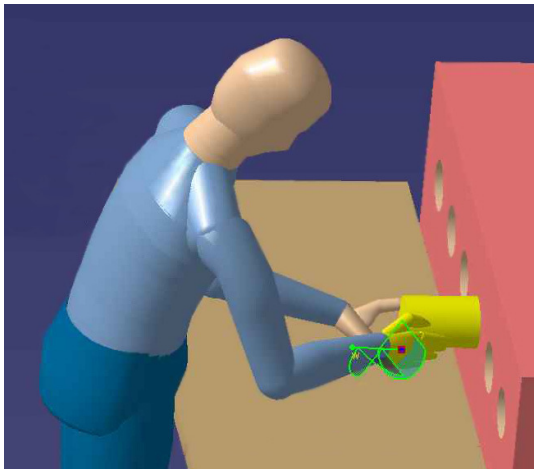

FIG. 1.25 – Contrôle d'un humain virtuel en cinématique inverse sous Catia®.

dans DI-Guy [DIG], Ergo [Erg] et Jack [Jac].

Ces techniques soit ne permettent pas de spécifier un large spectre de mouvements, soit sont difficiles à mettre en œuvre. Et surtout, elles ne permettent pas d'interagir directement et de manière naturelle avec la maquette. Ce constat nous amène à proposer de nouveaux algorithmes de contrôle destinés à l'interaction. Ils sont l'objet de notre étude, qui s'insère dans les développements de Samira®.

1.4 Mise en place des prémices d'une architecture

Nous posons ici les bases d'une organisation, par le biais d'un schéma fonctionnel et d'une proposition d'architecture, regroupant les besoins exprimés. Le détail de cette architecture sera l'objet des chapitres suivants de la thèse.

1.4.1 Schéma fonctionnel de la simulation d'un humain virtuel

Nous partons du comportement le plus complexe qui soit, c'est-à-dire l'humain virtuel autonome. Nous détaillons alors les éléments nécessaires à son bon contrôle à partir de la description haut niveau d'un mouvement complexe (une description synthétique d'une tâche à accomplir, comme nous pourrions la donner à un humain réel).

Ex : Un responsable donne l'ordre d'accomplir une tâche de maintenance donnée. Le technicien déduit d'un ensemble de règles non exprimées explicitement et du contexte dans lequel il se trouve, que la première tâche à accomplir est d'aller chercher ses outils, de se placer ensuite devant le poste de travail...

- Il faut disposer d'un langage de description des tâches complexes (on parle de *langage de scénarisation*, ou de *langage comportemental* [AB03], [AKM+02]).
- Comme sur l'exemple du technicien, un algorithme de décomposition du mouvement en mouvements plus simples est nécessaire. Cette décomposition est très complexe. Elle est conditionnée par le contexte dans lequel l'humain virtuel se trouve, la logique de la situation et également par un ensemble de règles non exprimées explicitement (ayant trait aux us et coutumes et à la culture de l'humain simulé). Nous appellerons cette étape la *segmentation de mouvement*, car c'est bien du passage d'un mouvement complexe à une succession de mouvements plus élémentaires dont il s'agit.
- Une fois le mouvement segmenté, il faut le générer - tout en s'assurant que le chemin obtenu ne rentre pas en conflit avec l'environnement dans lequel notre humain évolue. Se dessine alors un module de *génération de trajectoire / pathfinding* qui nous permettra de générer des mouvements en respectant cette contrainte.
- Pour asservir un système réel le bon sens et la pratique nous amènent à introduire une **boucle de rétroaction** dans le système par le biais d'un *contrôleur*. Ce contrôleur plus bas niveau permettra de s'assurer du respect de la consigne.
- Dans notre cas, cette rétroaction peut de nouveau être mise en oeuvre, à condition de remplacer le système réel par un bloc de *simulation temps-réel* de la physique de notre monde (qui se bornera au domaine de la dynamique dans notre cas d'étude).

Ces blocs de *Comportement/scénarisation*, de *Segmentation de mouvement*, de *Pathfinding*, de *Contrôle bas niveau* et enfin de *Simulation*, peuvent être retrouvés dans la figure 1.26.

D'autres cas d'utilisation de l'humain virtuel nécessitent d'autres fonctionnalités. Pour ce qui est du travail collaboratif, l'opérateur spécifie la trajectoire désirée de l'objet manipulé par le groupe d'humains virtuels. Le bloc fonctionnel *Travail collaboratif* se charge ensuite de trouver le mouvement de chaque humain permettant d'y parvenir. Ce bloc apparaît dans la figure 1.26.

Nous souhaitons insister sur la distinction (déjà notée, mais de manière informelle) qui peut être faite sur le schéma 1.26 entre deux parties distinctes. Nous y trouvons :

- des couches basses qui permettront le contrôle et la simulation du mannequin
- et des couches de plus haut niveau qui permettront de piloter le mannequin de manière plus automatique, voir autonome.

Pour la gestion du contact (utile à l'interaction) et le pathfinding, l'agencement du monde dans lequel on évolue doit être connu, d'où la présence du bloc *environnement*. Enfin la sortie de l'animation est un rendu 3D (avec en plus, mise à disposition des données chiffrées de l'animation), comme indiqué sur le schéma 1.26.

Dans la section précédente, nous avons mis en valeur l'utilité de la capture de mouvement. Nous la retrouvons en entrée de notre architecture. Nous nous concentrons dans la suite de l'étude sur l'animation d'un seul humain virtuel, en interaction avec l'environnement - par le biais de capture de mouvement. Ceci nous permettra de tester les couches basses de notre architecture. Nous élargirons cependant ce cadre à d'autres méthodes de spécification des consignes, plus riches et apportant de premiers éléments

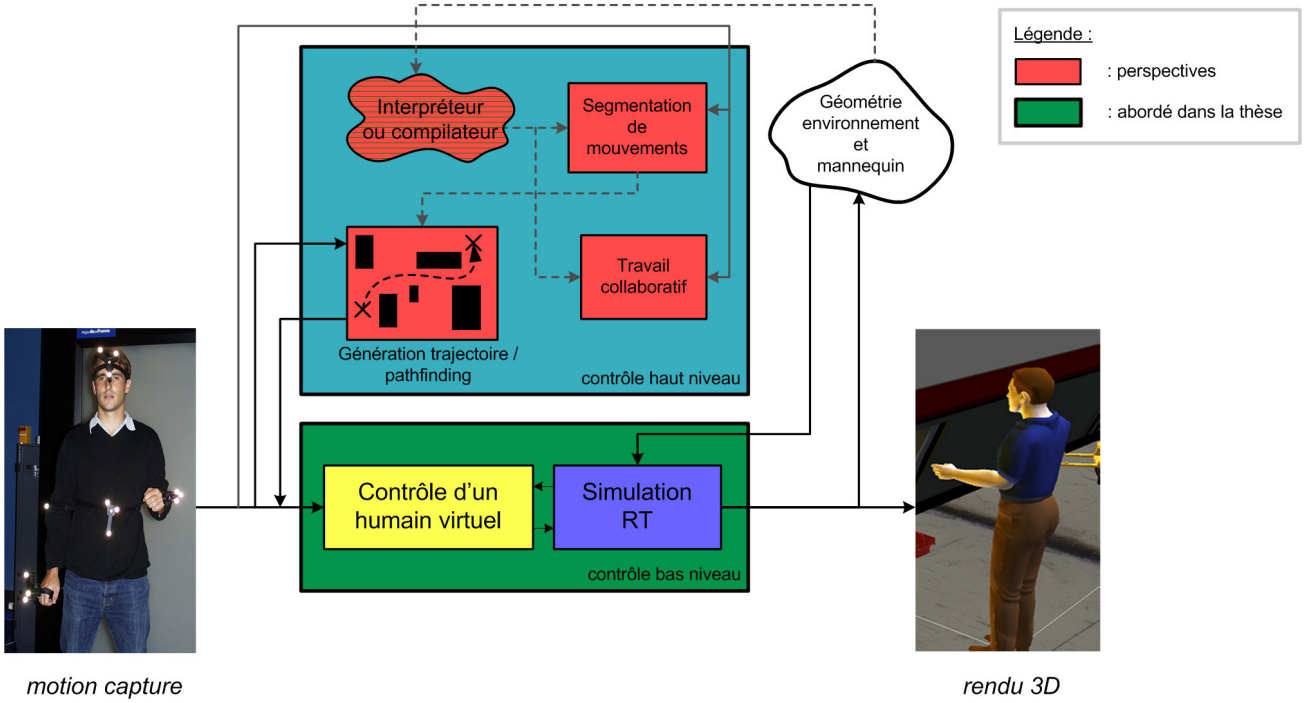

FIG. 1.26 – Schéma fonctionnel global de l'architecture d'animation mise en place dans la thèse.

de réponse à l'autonomie de nos mannequins[9], dans le dernier chapitre du présent rapport.

Nous souhaitons insister sur le caractère temps-réel de notre application. Celui-ci permet de donner un retour visuel à l'acteur accomplissant sa tâche, alors qu'il est observé par le dispositif de capture de mouvement. Il pourra alors compenser d'éventuelles erreurs d'asservissement. Une boucle de rétroaction est ainsi mise en œuvre naturellement par l'acteur, par le biais de ce retour visuel sur le monde virtuel.

Partage du contrôle-commande des degrés de liberté

Nous mettrons en œuvre une technique à base de capture de mouvement pour spécifier les consignes du mannequin. Il pourrait sembler opportun d'acquérir la position de tous les membres de l'acteur et de les appliquer tel quel sur le squelette de l'humain virtuel. Cette technique directe n'est envisageable que dans des cas simples. Dans le cas général, nous devons prendre en compte le fait que le monde réel et sa contrepartie simulée peuvent être différents. Nous devons principalement prendre en compte des différences morphologiques et des différences d'environnement :

- l'acteur observé et l'humain virtuel peuvent avoir des dimensions et même des proportions différentes. Si les positions désirées sont appliquées directement aux membres de l'humain virtuel, les mouvements du squelette ne correspondent pas à ce qui est attendu d'eux : limites articulaires dépassées, équilibre de l'humain virtuel non respecté...
- l'environnement dans lequel évolue l'humain virtuel est différent de l'environnement dans lequel évolue l'acteur. Puisque l'intérêt est de simuler un prototype qui n'a pas d'équivalent physique, l'environnement de l'acteur sera libre, alors que celui du mannequin sera encombré par la maquette à tester. L'acteur peut donc se placer dans des positions qui ne sont pas atteignables par le mannequin. Si aucune contrainte n'est imposée, le mannequin va pénétrer son environnement : ce comportement n'est pas réaliste.

[9]Nos mannequins ne seront pas autonomes à proprement parler, mais nous augmenterons sensiblement le degré d'automatisation des méthodes, pour diminuer la charge de travail de l'opérateur.

Les mouvements de l'acteur doivent donc être modifiés de manière à prendre en compte ces contraintes. Ainsi, une partie du mouvement du mannequin virtuel sera dictée par les données de la capture de mouvement et une seconde partie sera calculée de manière automatique, en considérant les différences de morphologie et d'environnement. Il y a donc un véritable compromis à trouver entre capture de mouvement complète et autonomie complète qui permet d'accomplir les mouvements désirés en prenant en compte les différences exprimées ci-avant.

Le premier des problèmes, portant sur les différences morphologiques est en partie résolu par les méthodes de *retargetting*, décrites en 1.2.1.3. Le choix qui doit être effectué entre priorité à la conservation de l'*attitude* et priorité à la conservation des contraintes (voir 1.2.1.3), nous est dicté par la finalité de notre application. Notre problématique portant sur l'interaction avec la maquette. La position des objets que nous souhaitons manipuler est imposée par la maquette. Aussi la méthode de *retargetting* choisie est de conserver les positions des membres servant à l'interaction (mains et pieds notamment), c'est-à-dire les contraintes.

Ce partage du contrôle des degrés de liberté trouve un autre intérêt, en ce sens qu'il permet d'apporter une assistance à l'opérateur. On peut l'imaginer sous forme de coordination des mouvements, de guidage... La finalité de l'approche étant de contrôler l'humain virtuel de la manière la plus simple et la plus efficace qui soit, en automatisant ce qui peut l'être tout en laissant des possibilités de commande à l'opérateur.

La gestion de ce partage du contrôle-commande des degrés de liberté sera l'objet de notre étude.

1.4.2 Identification des éléments constituants du contrôle

Dans cette section, nous précisons la partie *bas niveau* (blocs jaune et bleu du schéma 1.26), objet principal de notre attention dans cette thèse. L'architecture du simulateur est conditionnée (du moins partiellement) par le besoin en terme d'animation. Aussi, nous identifierons dans un premier temps les sources du mouvement humain, c'est-à-dire les facteurs qui l'influencent. Puis dans un second temps, nous introduirons une à une les sources de mouvement identifiées dans un contrôleur destiné à animer l'humain virtuel en temps réel. Notre approche sera donc d'émuler chaque facteur influent dans le mouvement d'un humain réel, de manière à animer un humain virtuel, de la manière la plus réaliste qui soit (tant d'un point de vue visuel que physique). Une architecture d'animation simplifiée sera alors proposée.

1.4.2.1 Contrôle d'une tâche dans l'espace opérationnel

La première des fonctions à implémenter dans notre système est le contrôle d'une tâche dans l'espace opérationnel (ou tâche externe). Le personnage virtuel doit être capable de positionner un repère de contrôle, à un emplacement désiré, spécifié dans l'espace opérationnel[10].

Concrètement, ce besoin peut être révélé par la situation suivante : un opérateur désire ouvrir une porte. Le premier des mouvements à faire est alors de placer la main de l'humain virtuel (ce sera notre repère de contrôle dans ce cas) sur la poignée, figure 1.27.

Nous devons également être capables de gérer plusieurs repères de contrôle simultanément. En effet, imaginons l'exemple d'un humain virtuel cycliste. Le mouvement de base de ce système pourra être obtenu en gérant un repère de contrôle au niveau de chaque pied - pour le mouvement du pédalier-, ainsi qu'un repère de contrôle à chaque main - pour le mouvement du guidon.

Voyons maintenant le problème d'un point de vue mathématique. Le mannequin se trouve à une position donnée et dans une configuration donnée. La position de ses repères de contrôle est connue. L'opérateur modifie la consigne, engendrant une erreur dans l'espace opérationnel (entre la position

[10]Cette position de consigne étant spécifiée par un dispositif adéquat. Il pourra s'agir de capture de mouvement, de génération de trajectoire...

FIG. 1.27 – La main doit atteindre la poignée de porte. Une erreur dans l'espace opérationnel est donc à compenser.

courante et la position désirée des repères de contrôle). Le contrôle à implémenter devra alors modifier les paramètres articulaires de l'humain virtuel, de manière à compenser l'erreur dans l'espace opérationnel. Nous voyons donc qu'un contrôle de tâche doit gérer deux espaces différents :

- l'espace opérationnel, qui est l'espace le plus intuitivement manipulé par le cerveau humain (puisque c'est un modèle de l'espace dans lequel nous vivons)
- et l'espace des paramètres articulaires (ou espace de configuration), qui est celui qui permet de manipuler le plus aisément possible, une structure robotique, comme celle qui permet de modéliser un squelette humain

Toute la difficulté étant de passer d'une erreur exprimée dans le premier espace, à une commande exprimée dans le second.

1.4.2.2 Contrôle d'une tâche interne

Imaginons un bras série à sept degrés de liberté. Si nous spécifions une tâche de dimension six (comme c'est le cas du groupe de Lie $SE(3)$), il restera un degré de liberté à gérer. Dans ce cas le bras est dit *redondant*. Il existe alors plusieurs solutions au problème (souvent une infinité). Le contrôle de la tâche en sélectionne une, mais ça n'est pas forcément celle qui est attendue. Pour le cas d'un bras humain, ce problème peut se traduire comme sur l'illustration 1.28.

Le problème est bien sûr accentué dans les cas, comme celui de l'humain virtuel, où le nombre de degrés de liberté est plus important. Nous serons donc amené à contrôler la cinématique (ou la dynamique s'il y a lieu) interne.

1.4.2.3 Contraintes physiques et biomécaniques

Le dernier objectif identifié, à accomplir de manière à obtenir un mouvement réaliste, est le respect de certaines contraintes. Notamment les contraintes liées à la structure biomécanique sous-jacente des humains et les contraintes physiques.

Ces contraintes peuvent se décliner sous trois formes essentiellement : le contact, les limites articulaires et le respect de l'équilibre.

Fig. 1.28 – A des positions de main et de buste identiques la configuration du bras n'est pas entièrement définie dans le cas redondant.

Contact

Un humain virtuel ne doit pas pénétrer son environnement (ni les parties de son propre corps). Ces contraintes doivent être respectées, de manière automatique, lors de l'interaction du mannequin virtuel avec son environnement.

La non-pénétration est bien évidemment essentielle, mais une notion connexe, déjà évoquée ci-avant l'est bien plus : celle de l'interaction avec l'environnement. L'interaction nous amène à considérer le processus de gestion du contact de manière plus fine. En effet, il va non seulement falloir assurer la non-pénétration, mais également laisser la possibilité de manipuler, d'interagir avec son environnement. C'est-à-dire que les mouvements du mannequin peuvent avoir un impact sur l'état du monde virtuel qui l'entoure. C'est le cas dans bien des situations de la vie courante : lorsque nous devons pousser une porte, appuyer sur un bouton...

Limites articulaires

Les limites articulaires sont une contrainte du même ordre que la gestion du contact. De manière générale, on pourra noter que les limites articulaires sont plus contraignantes dans le cas d'un humain (virtuel ou non), que dans le cas d'un robot pour lequel il n'est pas rare, par exemple, de trouver des articulations capables d'effectuer un tour complet autour de leur axe.

Equilibre

Les mouvements des humains réels sont équilibrés de manière naturelle (du moins après un ou deux ans d'apprentissage). Mais si nous n'y prenons pas garde, les mouvements produits par un algorithme de contrôle quelconque ne respectent pas cette contrainte d'équilibre. C'est-à-dire que des mouvements qui ne sont pas faisables par un humain réel dans les mêmes conditions sont générés. C'est ce que nous constatons sur la figure 1.29 :

Nos personnages digitaux, qui rappelons-le s'insèrent dans le contexte du prototypage virtuel, doivent permettre de démontrer la faisabilité d'opérations complexes sur un produit en cours de conception. Il apparaît donc indispensable que les mouvements générés par notre système soient faisables. Un contrôleur d'équilibre nous paraît indispensable si on ne veut pas parvenir au type de cas illustré figure 1.30.

En capture de mouvement, les mouvements d'un acteur sont observés pour les appliquer au personnage simulé. De prime abord nous pourrions penser que si les contraintes sont respectées dans le monde réel, elles seront respectées dans le monde simulé. Si nous souhaitons disposer d'un système qui puisse prendre en compte des différences entre monde réel et monde simulé, comme indiqué en 1.4.1, un mécanisme visant à faire respecter les contraintes physiques et biomécaniques est indispensable.

FIG. 1.29 – Posture déséquilibrée d'un humain virtuel.

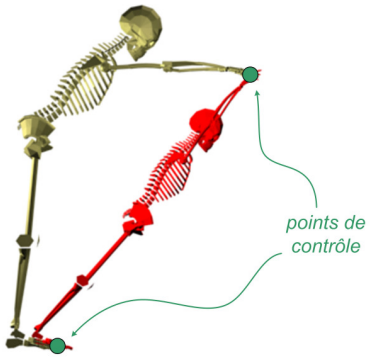

FIG. 1.30 – En appliquant les mouvements d'un acteur à un humain virtuel de morphologie différente, on peut violer certaines contraintes, si on n'y prend pas garde - ici l'équilibre.

1.4.2.4 Proposition d'une architecture pour le bas niveau

En 1.4.1, nous avons proposé de mettre en oeuvre un bloc de *simulation* et une boucle de rétroaction par le biais d'un bloc *contrôleur*. Nous pouvons maintenant raffiner quelque peu cette architecture. Le besoin en terme d'interaction avec l'environnement, évoqué en 1.3, influencera le choix des modèles implémentés, plus que l'architecture en elle-même.

Simulation

D'un point de vue de la simulation, le comportement physique est donné sous la forme d'une équation différentielle signifiée par un bloc *physique* (Φ). Il convient d'intégrer cette équation différentielle, nous ajouterons donc un bloc d'*intégration* à l'architecture ($\int$).

Contrôleur

Nous avons vu que les facteurs influent le mouvement peuvent être groupés en 3 catégories :

– le contrôle de la tâche dans l'espace opérationnel, permettant à un repère de contrôle d'atteindre une position de consigne

– le contrôle d'une tâche interne, gérant les degrés de liberté restants
– le respect des contraintes, de manière à obtenir des mouvements physiquement et biomécaniquement corrects

Ces trois types de contrôle constituent le cœur du contrôle bas niveau. Nous les étayerons d'un bloc de *guides virtuels* visant à aider l'acteur immergé dans la capture de mouvement à faire réaliser les mouvements désirés au mannequin.

Nous voyons poindre ici, les prémices d'une architecture pour la simulation et le contrôle d'un humain virtuel. Nous y trouverons :

– un bloc de **simulation temps-réel** comportant :
  – un étage de *physique*
  – un étage *d'intégration*
  – un étage de *résolution de contraintes*
– un bloc de **contrôle de l'humain virtuel** comportant :
  – un étage de *contrôle de la tâche*
  – un étage de *contrôle interne*
  – un étage de *contrôle de l'équilibre*
  – un étage de *guidage*

Cette architecture simplifiée, dont nous voyons un schéma de principe figure 1.31, sera détaillée en 2.2 et 3.

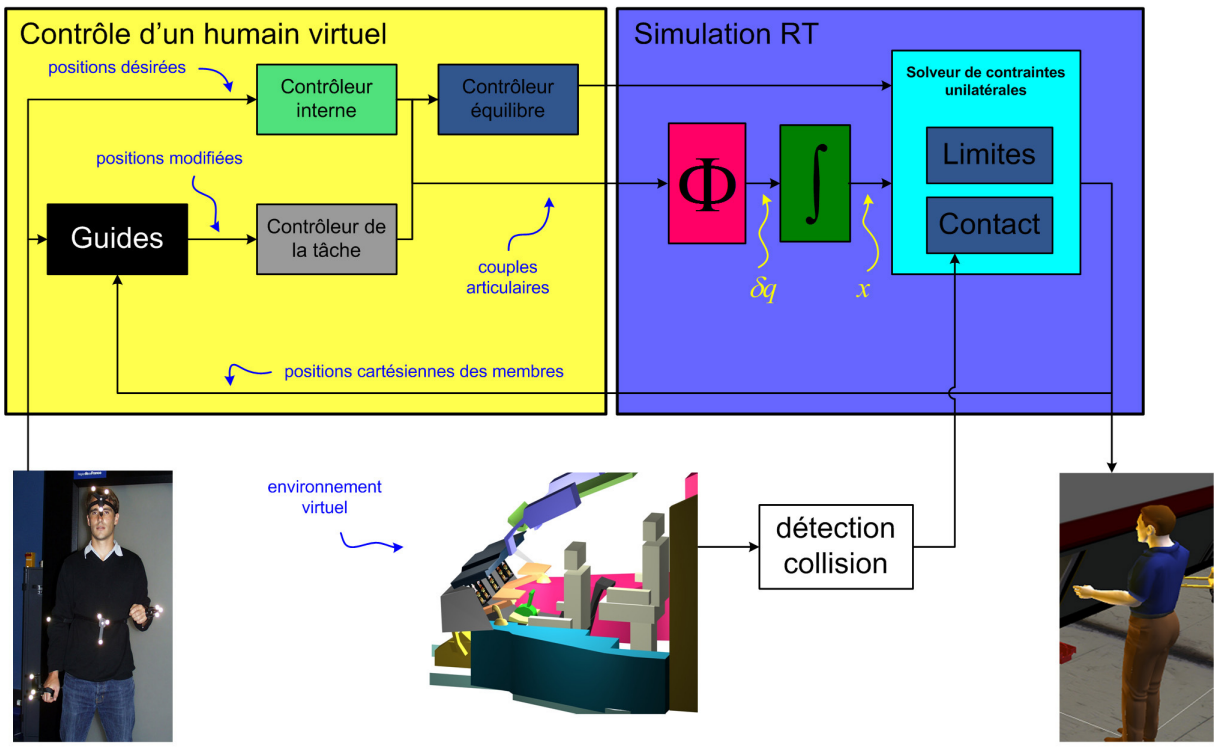

Fig. 1.31 – Schéma de principe du bas niveau (notre centre d'intérêt pour cette thèse) de l'architecture d'animation.

L'architecture proposée, que nous avons introduite dans [RMM+05b], est innovante dans le domaine de la réalité virtuelle et a été empruntée au domaine de la robotique.

Rem : L'architecture complète et détaillée du système est donnée figure 1.32. Le lecteur pourra

s'y référer à tout moment. L'objet de la suite de ce manuscrit est d'argumenter et de motiver les choix effectués qui ont permis d'aboutir à ce schéma. Notons qu'une équivalence dans la couleur des blocs fonctionnels permet de mettre en correspondance les schémas 1.26, 1.31 et 1.32.

1.5 Contributions scientifiques

Le schéma de la figure n°1.31 est l'architecture vers laquelle nous souhaitons tendre. Notre contribution porte sur les points suivants :

- Nous avons proposé une architecture de système en vue de l'animation d'humains virtuels en temps-réel. Toute la conception a été pensée de manière à pouvoir **interagir de manière naturelle avec un environnement complexe** par le biais de forces d'interaction - c'est là un des principaux intérêts du travail effectué. L'architecture est modulaire d'un point de vue fonctionnel, comme on peut le voir figure 1.31, mais également du point de vue de la stabilité (c'est-à-dire que grâce à la propriété de passivité décrite en 2.1, **l'ajout d'une fonctionnalité ne perturbe pas la stabilité** du système). Le système développé a également permis de formaliser (pour la première fois à notre connaissance) une architecture de système d'animation mettant en oeuvre un simulateur temps-réel pour l'animation d'humains virtuels. Le schéma traditionnel *système + contrôleur* est ici remplacé par un schéma *simulateur + contrôleur*. Notons de surcroît que s'il n'est pas implémenté à l'heure actuelle, le système permet naturellement de **coupler un dispositif haptique** (dispositif à retour d'effort) au mannequin.
- Lorsque plusieurs repères sont contrôlés sur un humain virtuel (ex : les repères associés à chaque main), leur consigne peut être conflictuelle (ex : le mannequin ne peut pas atteindre la consigne donnée sur la main droite et celle donnée sur la main gauche simultanément) - voir 2.2.2. Pour gérer ce type de situations, des travaux récents issus de la littérature proposent de donner une priorité à chaque repère contrôlé. L'outil mathématique pour les faire appliquer est basé sur des *projecteurs*. Dans cette thèse il est mis en évidence que des problèmes peuvent survenir à l'utilisation de ces projecteurs si on n'y porte pas une attention soutenue(les détails sont donnés en 2.2.2.2). Nous montrons en effet que **la passivité n'est pas assurée dans le cas général lors de l'utilisation de *projecteurs*** dans un contrôleur.
- Face à cette constatation de la perte de passivité lors de l'utilisation de projecteurs (dans le cas général), il nous est apparu intéressant de caractériser un sous-ensemble de projections passives. Nous proposons donc une **méthodologie de construction de projecteurs passifs**, empruntée au domaine de la télémanipulation. Cette approche nous permet d'augmenter la robustesse du système. Les projecteurs passifs ainsi créés, sont basés sur une analogie avec des mécanismes (les mêmes que ceux qui sont simulés), pour cette raison, on les appelle des *mécanismes virtuels*(voir 2.2.2.2). Ces *mécanismes virtuels* permettent d'aider l'opérateur à animer le mannequin, en automatisant, ou contraignant certains degrés de liberté du mannequin, ils feront donc office de **guides de mouvement**.
- Les mouvements de l'humain virtuel produits par une système d'animation brut, ne sont pas contraints à être équilibrés (c'est-à-dire que dans le cas général ils ne le sont pas). C'est-à-dire que les mouvements issus d'un système d'animation ne sont pas forcément reproductibles par un humain réel (voir 1.4.2). Pour s'assurer que les mouvements produits par le système d'animation soient équilibrés, nous avons développé un **contrôleur d'équilibre**. Lorsque les mouvements produits par le système sont jugés déséquilibrés, le contrôleur d'équilibre les corrige. Plusieurs versions sont proposées. La première ne prend en compte que les cas où les contacts de l'humain virtuel avec l'environnement sont coplanaires (formulation donnée en 3.3.2.1). Nous avons alors étendu ce contrôleur aux **cas des contacts non-coplanaires frottants** (comme c'est souvent le cas) dans une formulation originale, sous la forme d'un problème d'optimisation (détaillé dans 3.3.2.2).
- Dans les contributions qui précédent, nous avons introduit différents *modes de commande* :
  - un mode de commande normal, ou *libre*, qui correspond à un repère de contrôle non contraint
  - un mode de commande *contraint*, qui correspond aux *guides* de mouvements apportés par les mécanismes virtuels

– un mode de commande *autonome*, ou la trajectoire de consigne du repère contrôlé est générée automatiquement

Nous proposons alors un contrôleur plus haut niveau permettant d'**ordonnancer ces différents modes de commande**. Il s'appuie sur une machine d'état (la méthode est précisée en 4.1.2). Les transitions entre états (ou modes de commande), sont conditionnées par l'état de la simulation, celles-ci étant prévues par avance, on parle de **scénario de comportement**.

Conclusion

Dans ce chapitre, nous nous sommes d'abord livrés à un état de l'art des techniques d'animation d'humains virtuels, en distinguant les algorithmes de contrôle et les méthodes de spécification du mouvement désiré.

Nous avons ensuite détaillé le contexte industriel dans lequel s'insère notre étude : le prototypage virtuel et les techniques d'ingénierie concourante. Nous avons alors relevé le besoin relatif à cet environnement, en constatant que les techniques mises en œuvre jusqu'à présent n'y répondaient pas parfaitement.

Nous avons alors décrit une architecture globale permettant de répondre au besoin et cerné les points précis de cette architecture qui seront approfondis dans la suite de la thèse.

Les chapitres 2 et 3 porterons sur la description du bas niveau *de l'architecture (bloc vert figure 1.26, précisé figure 1.31). Le chapitre 4 apportera certaines fonctionnalités plus* haut niveau *(bloc bleu figure 1.26).*

Enfin nous avons exposé les points sur lesquels vont porter notre contribution dans cette thèse.

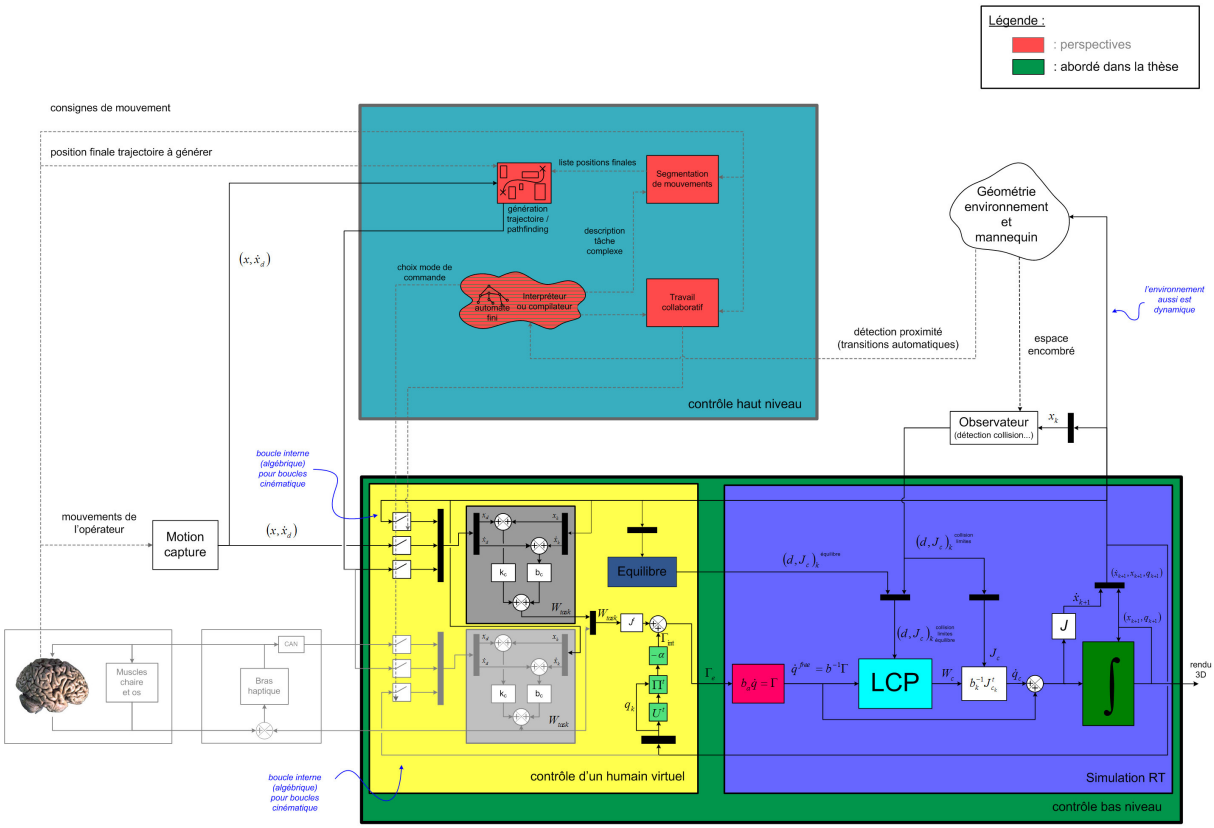

FIG. 1.32 – Schéma global d'architecture. C'est l'architecture mise en œuvre dans cette thèse. Attention le couplage haptique (en grisé clair) est donné à titre informatif seulement (pour montrer que l'haptique s'insère naturellement dans l'architecture développée), il n'a pas été réalisé. SCHEMA PREVU EN A3 DANS LE DOCUMENT FINAL.

CHAPITRE 2

Génération des mouvements d'un humain virtuel

Du fait principalement de la cinématique du squelette humain, les mouvements de l'Homme sont complexes, caractéristiques et variés. De plus, notre cerveau est tellement habitué à analyser et à interpréter des mouvements humains, qu'une petite erreur dans l'animation est instantanément perçue, d'où la difficulté de notre travail.

Dans ce chapitre nous allons dans un premier temps introduire la notion de passivité, *propriété du monde réel qui nous servira de guide lors de la mise en place de tous les éléments de l'architecture.*

Nous détaillerons ensuite les méthodes utilisées pour l'implémentation de chaque fonctionnalité des blocs Contrôle d'un humain virtuel *(jaune) et* Simulation temps-réel *(bleu) du schéma 1.31.*

Nous aborderons dans un premier temps la problématique de la simulation et discuterons du modèle de comportement physique choisi. Nous mettrons en lumière l'attention qui doit être portée à l'intégration de l'équation de comportement du fait de l'utilisation de méthodes discrètes. Nous décrirons ensuite les contrôleurs mis en œuvre.

Ces contrôleurs incitent, dans certaines situations, à utiliser un outil mathématique appelé projection. *Nous montrons dans un second temps que ces projections peuvent entraîner une perte de passivité si l'on ne prend pas garde à leur mise en œuvre. Nous introduirons à cet effet un autre outil nommé* mécanisme virtuel *qui permet, sous hypothèses fixées, de réaliser des projections passives.*

Notons que l'architecture donnée figure 1.31 sera entièrement détaillée à l'issue de ce chapitre, à l'exception du bloc Solveur de contraintes unilatérales *sur lequel nous porterons une attention particulière au chapitre 3.*

2.1 La passivité : un guide de synthèse pour la commande

L'architecture fonctionnelle telle qu'elle a été mise en place en 1.4.1 est modulaire d'un point de vue fonctionnel, mais l'ajout de contre-réactions au système peut perturber la stabilité. C'est-à-dire qu'en modifiant les bouclages du système (en ajoutant ou en retirant des blocs fonctionnels au schéma), sa stabilité peut être complètement perturbée. La modularité n'est donc qu'une vue conceptuelle. Dans les faits il faut toujours envisager le système dans son intégrité pour le faire fonctionner correctement.

Certains de nos blocs sont fortement non-linéaires, voir non continus, non-invariants et discrétisés (ce sera le cas de la simulation). La notion de *passivité* nous permet d'assurer la stabilité alors que des boucles de contre-réaction sont ajoutées ou retirées au système. La passivité est une propriété des systèmes du monde réel et est à relier à leur énergie. Nous présenterons dans un premier temps cette notion de passivité, puis montrerons en quoi elle peut nous être utile dans la construction de notre architecture d'animation, en introduisant les théorèmes qui font son intérêt.

Aparté - la passivité en deux mots

La passivité est une propriété du monde réel. Elle spécifie qu'un système ne peut créer d'énergie sans source extérieure. Cette propriété vérifiée implique la stabilité du système étudié. Deux système passifs en contre-réaction forment un système passif. C'est là l'intérêt principal de la passivité : des boucles de contre-réaction (ou fonctionnalités supplémentaires) peuvent être ajoutées au système sans perturber la stabilité. C'est-à-dire qu'à chaque modification de l'architecture, on ne doit plus étudier la stabilité de tout le système, mais uniquement s'assurer de la passivité du sous-système modifié. Pour un système d'animation flexible comme celui que nous souhaitons construire, cette modularité nous semble intéressante. Nous chercherons la passivité dans les développements détaillés dans la suite de l'étude.

L'intérêt de la passivité peut être montré en quelques théorèmes fondamentaux, mais avant de les exposer, décrivons les notions de *port d'interaction* et de passivité. Un port d'interaction est le lieu d'échange de l'énergie entre systèmes. Pour un mannequin en contact avec l'environnement, un exemple de port sera donné simplement par un des contacts. La passivité est une condition de stabilité spécialisant les fonctions de Lyapunov [Kha02]. Elle est relative à l'énergie échangée aux ports d'interaction du système considéré avec l'environnement. [Kha02] nous donne les définitions suivantes :

Définition 2.1.1 *Un système d'entrée u, d'état x et de sortie y est dit* passif, *s'il existe une fonction $V(x)$ définie positive continuement différentiable (appelée fonction de stockage) telle que :*

$$u^t y \geq \dot{V},\ \forall (x,u) \in \mathbb{R}^n \times \mathbb{R}^p. \tag{2.1}$$

Définition 2.1.2 *Il est dit* passif strictement par rapport à sa sortie *si :*

$$u^t y \geq \dot{V} + y^t \rho(y),\ \textit{avec}\ y^t \rho(y) > 0,\ \forall\ y \neq 0,\ \textit{et pour}\ \rho\ \textit{tel que}\ \dim(y) = \dim(\rho(y)). \tag{2.2}$$

[And90], précise la définition dans le cas de systèmes mécaniques :

Définition 2.1.3 *Un port d'interaction (W,V), constitué d'une force $W(t)$ et d'une vitesse $V(t)$ est passif si et seulement si il existe un réel β tel que :*

$$\forall t > 0, \int_0^t W^t(t) V(t) dt \geq -\beta^2. \tag{2.3}$$

C'est à dire que le système ne peut créer *d'énergie. C'est une propriété des systèmes du monde réel.*

Le théorème suivant nous permet de passer de la notion de passivité à la notion de stabilité, en nous assurant qu'un système passif strictement par rapport à sa sortie est stable [Kha02] :

Théorème 2.1.1 *Un système passif strictement par rapport à sa sortie est stable au sens du gain fini en norme L_2.*

La définition de la stabilité au sens du gain fini L_2 est donnée dans [Kha02], nous retiendrons essentiellement son caractère stable.

La passivité est une condition suffisante, mais pas nécessaire à la stabilité, ce qui veut dire qu'il existe des systèmes stables qui ne sont pas passifs. La passivité nous apporte donc un ensemble restreint de solutions au problème de la stabilité. En revanche elle permet grâce aux théorèmes suivants, d'assurer la stabilité des systèmes couplés, uniquement par des conditions sur ces mêmes systèmes découplés ([Kha02], preuve du Lemme 6.8.) :

Théorème 2.1.2 *Deux systèmes passifs strictement par rapport à leur sortie, en contre-réaction forment un système passif strictement par rapport à sa sortie.*

Théorème 2.1.3 *Un système H_1 passif en contre-réaction avec un système H_2 passif strictement par rapport à sa sortie, forme un système passif strictement par rapport à la sortie de H_2.*

C'est-à-dire que des boucles de contre-réaction peuvent être ajoutées à un système passif, sans perturber la stabilité de celui-ci, tant que les blocs ajoutés sont eux-mêmes passifs. Ce résultat est étendu dans [Col94] aux systèmes à n ports (c'est-à-dire aux systèmes couplés à plus d'un autre système). Ce sont exactement les résultats nécessaires au découplage complet du système (fonctionnalités et stabilité comprises). Nous allons maintenant les étendre à un cas moins restrictif grâce à la notion de *stabilité inconditionnelle* introduite dans [Bol57] :

Définition 2.1.4 *Un système est inconditionnellement stable si et seulement si il n'existe pas de système additionnel passif avec lequel la contre-réaction donne un système instable.*

La notion de stabilité inconditionnelle permet d'assurer la stabilité dans des conditions moins contraintes, mais encore faut-il pouvoir disposer de critères permettant de la prouver. Ceux-ci existent pour certaines catégories de systèmes (ex : Llewellyn [Lle52] pour le cas d'un quadripôle et de deux contre-réactions scalaires, ou [Col94] qui apporte certains éléments de réponse pour le cas multivariables). Nous disposons dès à présent des outils nécessaires à la poursuite de notre étude.

2.2 Architecture

Dans cette section, nous raffinons peu à peu la structure proposée du contrôleur, en incluant le contrôle de la tâche et le contrôle interne, facteurs influents identifiés en 1.4.2. La gestion des contraintes qui constitue une partie volumineuse, sera détaillée chapitre 3.

2.2.1 Détails de la simulation et des contrôles externes et internes

Avant de nous intéresser à la partie contrôle, il nous semble primordial de détailler les choix effectués au niveau du bloc *simulation temps-réel*. C'est en effet ce bloc qui va conditionner d'une part les similitudes de comportement entre le monde réel et le monde virtuel et d'autres part les méthodes de contrôle implémentées.

2.2.1.1 Simulation physique

Sur l'architecture du système donnée schéma 1.32, cette fonctionnalité correspond au bloc donné illustration 2.1.

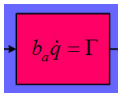

Fig. 2.1 – Bloc du schéma d'architecture 1.32, correspondant à la fonctionnalité décrite dans cette partie.

Nous avons décrit un modèle de comportement dynamique du second ordre en 1.2.1. Dans les cas complexes (la complexité peut être liée à l'ordre, à la dimension du système...), il peut être difficile de trouver des méthodes analytiques d'intégration de la dynamique du deuxième ordre en temps-réel. Comme dans le cas de la cinématique inverse (vue en 1.2.1), nous allons faire l'hypothèse des petits déplacements de manière à linéariser nos équations et à les intégrer de manière discrète.

Ces méthodes numériques discrétisant les équations différentielles sont efficaces, mais elles nécessitent du réglage. Réglage d'autant plus fin et difficile que l'ordre des équations différentielles est élevé. Ceci suggère l'utilisation d'une physique d'ordre moins élevé. Cette dynamique d'ordre faible, moins sensible que les comportements d'ordre plus élevés, nous permet, de tester nos algorithmes de contrôle, en ayant moins à nous soucier des problématiques d'ordre numérique.

Deux possibilités sont envisageables :

- le comportement différentiel d'ordre zéro : c'est un abus de langage qui correspond au comportement géométrique $\delta x = J\delta q$ donné équation (1.16).
- le comportement d'ordre un, dit *quasi-statique*, où l'intégration est rendue possible par l'intégration d'un frottement visqueux articulaire b_a symétrique défini positif (propriété que nous noterons *sdp*) :

$$\boxed{b_a\dot{q} = \Gamma,} \tag{2.4}$$

avec Γ les couples moteurs. La matrice des frottements visqueux b_a est choisie sdp pour que le système soit inversible. On écrit $b_a\dot{q} = \Gamma + \Gamma_{gravite}$, avec $\Gamma_{gravite}$ les couples articulaires dus à la gravité s'il y a lieu.
Au regard des problèmes énoncés précédemment, nous pourrions penser que la dynamique la plus simple est la plus indiquée. Mais la stabilité n'est pas le seul critère à considérer (d'autant plus qu'à l'ordre zéro une inversion de la matrice jacobienne J est à effectuer, dont les singularités peuvent également poser des problèmes). En effet, l'élévation de l'ordre des équations différentielles, apporte une richesse de comportement, qu'il est impossible de retrouver dans des physiques d'ordre moins élevé, plus précisément :

Comparaison des comportements

Passage d'ordre deux à ordre un

Les termes d'ordre deux correspondent au comportement inertiel. Les comportements inertiels sont impossibles à simuler avec un comportement différentiel du premier ordre. En ordre un, si nous lançons un objet (uniquement soumis à l'action du lanceur) celui-ci s'arrête à l'endroit où on le lâche.

De la même manière, les rebonds (liés à la transformation partielle de l'énergie cinétique en énergie potentielle et inversement) nécessitent un comportement inertiel pour être simulés. Ils sont impossibles à mettre en œuvre de manière naturelle en ordre un.

Passage d'ordre un à ordre zéro

Dans le cas de l'ordre zéro (nous rappelons que c'est un abus de langage, à l'ordre 0, il n'y a plus de comportement différentiel), l'équation de comportement est donnée par la relation géométrique $\delta x = J\delta q$ - décrite en (1.16). Cela signifie que les efforts ne sont pas pris en compte.

Récapitulatif du choix du modèle

Comportement différentiel d'ordre un

Avantages - inconvénients :

+ intègre la notion d'effort (elle nous est nécessaire pour obtenir une interaction naturelle du mannequin avec l'environnement)
+ moins sensible aux problèmes numériques qu'un comportement d'ordre deux (facilité d'implémentation et de mise en œuvre)
− pas de comportement inertiel (donc pas de rebonds)
− trajectoire linéaires et non quadratiques (le mouvement des objets manipulés semble parfois artificiel)

Passivité du système en boucle ouverte

Analysons maintenant le comportement du système en boucle ouverte du point de vue de la passivité. Le manipulateur en boucle ouverte est passif strictement en sortie au port $(\Gamma, \dot{q})$. Ce résultat s'appréhende aisément en vérifiant l'énergie échangée à notre port :

$$\int_0^t \Gamma^t \dot{q} dt = \int_0^t \dot{q}^t b_a \dot{q} dt, \tag{2.5}$$

b_a étant positive, l'intégrale l'est aussi, le manipulateur est donc bien passif strictement en sortie au port considéré[1].

2.2.1.2 Intégration numérique

Sur l'architecture du système donnée schéma 1.32, cette fonctionnalité correspond au bloc donné illustration 2.2.

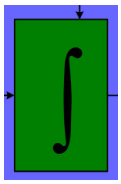

FIG. 2.2 – Bloc du schéma d'architecture 1.32, correspondant à la fonctionnalité décrite dans cette partie.

Nous présentons ici les problèmes liés à l'intégration numérique sur un exemple simple d'instabilité. Des schémas d'intégration très utilisés (explicites, implicites et leur extension sur groupes de Lie) sont décrits en annexe C.

[1] Dans le cas de la dynamique du deuxième ordre, cette propriété est assurée par le caractère antisymétrique de $\dot{M} - 2C$, comme indiqué dans [MLSS94].

Intégration numérique et passivité

Dans la section 2.1, nous avons rappelé le principe de conservation de l'énergie exprimé sous la forme de la notion de passivité. Nous allons voir ici que l'intégration numérique peut, si elle est mal réglée, provoquer une perte de la passivité du système simulé.

Pour cela prenons un exemple simple, celui d'un ressort compressé, puis détendu auquel on impose la position x d'une extrémité, comme illustré figure 2.3. Nous donnons d'une part le comportement continu et d'autre part le comportement discret.

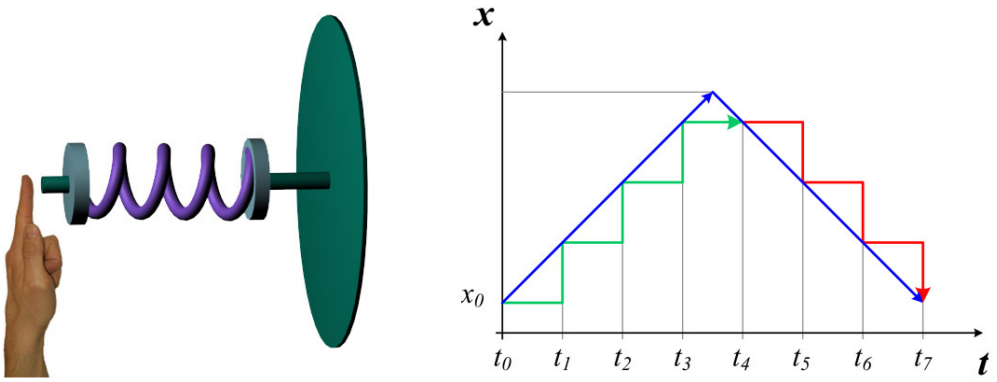

Fig. 2.3 – Le système considéré et le comportement qui lui est imposé (cas continu et discret).

Le mouvement x engendre une force $f = k(x - x_0)$, dérivant dans le cas continu d'une énergie potentielle $\varepsilon_p = \frac{1}{2}k(x - x_0)^2$, l'énergie ε_p^{dis} dans le cas discret étant intégrée à partir de f par Euler-explicite (voir annexe C). Nous quantifions ces résultats pour chaque pas de temps dans le tableau 2.1.

temps	**force**	ε_p	ε_p^{dis}
t_0	0	0	0
t_1	$k\delta x$	$\frac{1}{2}k\delta x^2$	0
t_2	$2k\delta x$	$2k\delta x^2$	$k\delta x^2$
t_3	$3k\delta x$	$\frac{9}{2}k\delta x^2$	$3k\delta x^2$
t_4	$3k\delta x$	$\frac{9}{2}k\delta x^2$	$3k\delta x^2$
t_5	$2k\delta x$	$2k\delta x^2$	0
t_6	$k\delta x$	$\frac{1}{2}k\delta x^2$	$-2k\delta x^2$
t_7	0	0	$-3k\delta x^2$

Tab. 2.1 – Force et énergie pour chaque pas de temps dans les cas continu puis discret.

La figure suivante 2.4 donne les courbes associées au tableau 2.1.

Nous voyons que le mouvement de compression apporte moins d'énergie dans le cas discret que dans le cas continu et que la détente relâche plus d'énergie dans le cas discret. La différence d'énergie entre la compression et la détente dans le cas discret va venir s'ajouter à l'énergie propre du système. **La discrétisation crée artificiellement de l'énergie**, il y a donc un risque de perdre la passivité. Des erreurs de ce type, sont retrouvées dans tous les schémas d'intégration (de manière, cependant, plus ou moins marquée).

Sur le schéma 2.4, nous pouvons constater que si le pas d'intégration δt (ou période d'échantillonnage) diminue, l'énergie créée diminue également. Une solution pour réduire les problèmes de passivité dus à la discrétisation est donc de réduire ce pas. Cependant si nous souhaitons une simulation temps-réel, nous nous heurtons à un problème. En effet, dans ce cas, la durée du calcul doit être inférieure au pas de temps simulé. C'est-à-dire qu'en raison des ressources limitées de la machine qui effectue le calcul, nous ne pouvons pas descendre en deçà d'un seuil de pas de temps.

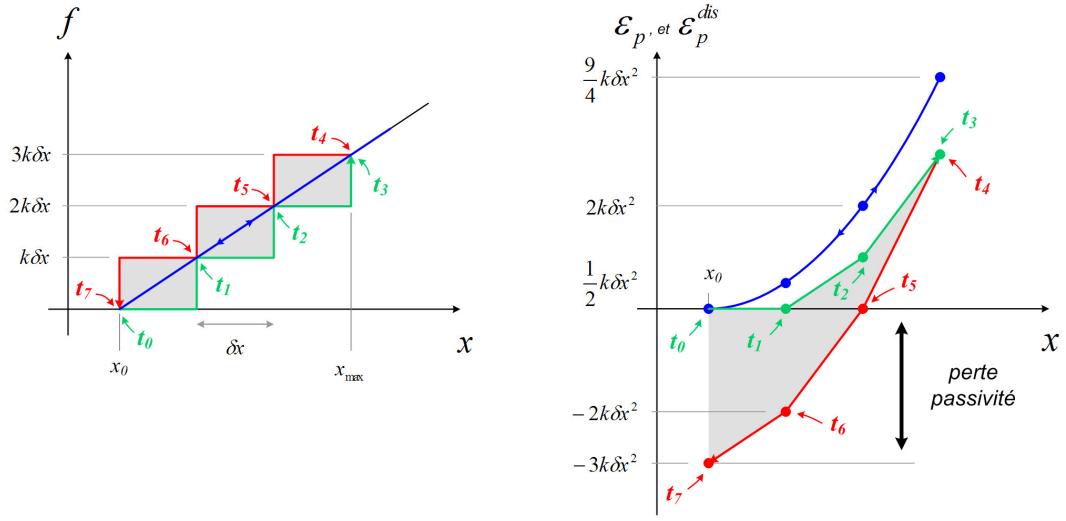

FIG. 2.4 – Evolution de la force et de l'énergie potentielle associée au ressort (cas continu et discret).

De la même manière, si la raideur k (donc la force mise en jeu), augmente, l'énergie créée de manière artificielle augmente également. Cela signifie que pour rester stable, notre simulation ne peut mettre en œuvre que des efforts limités (dépendant entre autres du pas d'intégration).

Nous pouvons introduire des termes dissipatifs dans les équations, de manière à contrôler l'énergie du système, ce que nous ferons au point suivant 2.2.1.3. Le dimensionnement de ces éléments dissipatifs n'est pas toujours aisé, aussi la quantité d'énergie retirée est souvent "arbitraire". Nous constatons simplement qu'il n'y a pas de divergence dans la simulation. Cette solution empirique, souvent mise en œuvre, mène bien entendu à des résultats eux aussi quelque peu "arbitraires". C'est pourtant elle qui est utilisée (voir terme dérivé du contrôleur en 2.2.1.3).

Dans le cas simple du ressort amorti, les conditions de stabilité liant la raideur k, la matrice des frottements visqueux b et la période d'échantillonnage δt sont données par les équations suivantes. Si aucune force n'est appliquée sur le ressort, on a :

$$b\dot{x} = -kx. \tag{2.6}$$

Dans le cas d'une fonction f quelconque, l'intégration de type Euler-explicite est donnée par (annexes, équation (22))

$$\frac{x_{k+1} - x_k}{\delta t} \simeq \dot{x_k} = f(x_k), \tag{2.7}$$

alors

$$b\,(x_{k+1} - x_k) = -k\delta t x_k, \tag{2.8}$$

donc

$$x_{k+1} = \left(1 - \frac{k\delta t}{b}\right) x_k. \tag{2.9}$$

On reconnaît la forme d'une suite géométrique de raison $\left(1 - \frac{k}{b}\delta t\right)$. Pour qu'elle converge, il nous faut donc

$$\left|1 - \frac{k\delta t}{b}\right| < 1, \tag{2.10}$$

k et b sont choisis positifs, donc la condition de convergence du système (2.6) est donnée par :

$$0 < \frac{k\delta t}{b} < 2. \tag{2.11}$$

Une deuxième approche est d'implémenter un schéma d'intégration plus performant que celui d'Euler-explicite que nous avons principalement utilisé dans notre étude. Cette méthode n'apporte pas une réponse complète au problème, mais permettrait de le réduire. Ainsi [And96] utilise une intégration selon le schéma de Tustin qui conserve la passivité - elle est implicite. L'intégration sur groupes de Lie de Runge-Kutta-Munthe-Kass (voir annexe C), permet aussi de réduire les erreurs.

Récapitulatif du choix du modèle

Intégration par Euler explicite

Avantages - inconvénients :

\+ calculs rapides
\+ implémentation facile
− passivité conditionnelle (restreint les possibilités du temps-réel, puisque le pas doit être *petit*)
− manque de précision

2.2.1.3 Contrôle de tâches externes

Sur l'architecture du système donnée schéma 1.32, cette fonctionnalité correspond aux blocs donnés illustration 2.5.

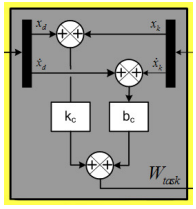

Fig. 2.5 – Blocs du schéma d'architecture 1.32, correspondant à la fonctionnalité décrite dans cette partie.

Grâce à l'introduction de la physique et à l'étage d'intégration, nous sommes maintenant en mesure de simuler un système soumis à des efforts extérieurs.

Comme nous l'avons évoqué précédemment, une tâche externe peut être exprimée sous la forme de l'asservissement à une position de consigne d'un repère de contrôle donné du mannequin. Un état de l'art du contrôle des tâches externes a été effectué en 1.2.1.2.

Nous construisons un correcteur qui permet de compenser l'erreur en position du repère de contrôle considéré, exprimée dans l'espace opérationnel. Ce correcteur génère une force qui, appliquée au repère de contrôle, permet à l'étage de simulation temps-réel de modifier la configuration du squelette.

Pour compenser l'erreur en position, le correcteur sera composé d'un terme proportionnel et pour *contrôler* l'énergie créée par la discrétisation, nous ajouterons un frottement visqueux assuré par un terme dérivé traditionnel [AH99]. C'est donc un asservissement classique de type proportionnel-dérivé

(PD), qui sera mis en œuvre dans notre architecture et ce de la même manière que dans [ZVDH03]. Le correcteur déjà donné en (1.55) s'exprime par

$$W_{task} = k_c(x_d - x) + b_c(\dot{x}_d - \dot{x}) \tag{2.12}$$

grâce à (2.4) nous écrivons le comportement du système bouclé

$$\begin{aligned} b_a\dot{q} &= \Gamma && (2.13)\\ &= J^t W_{task} && (2.14)\\ &= J^t\Big(k_c(x_d - x) + b_c(\dot{x}_d - \dot{x})\Big) && (2.15)\\ &= J^t\Big(k_c(x_d - x) + b_c(\dot{x}_d - J\dot{q})\Big), && (2.16) \end{aligned}$$

après quelques manipulations sommaires, il vient

$$(b_a + J^t b_c J)\dot{q} = J^t\Big(k_c(x_d - x) + b_c\dot{x}_d\Big), \tag{2.17}$$

que nous notons

$$b\dot{q} = \Gamma, \tag{2.18}$$

avec $b = b_a + J^t b_c J$ et $\Gamma = J^t(k_c(x_d - x) + b_c\dot{x}_d)$. La matrice $b_a + J^t b_c J$ est inversible si b_a symétrique définie, ou b_c symétrique définie et $\ker(J) = \{0\}$, or nous avons choisi b_a sdp en (2.4), donc

$$\begin{aligned} \dot{q} &= (b_a + J^t b_c J)^{-1} J^t\Big(k_c(x_d - x) + b_c\dot{x}_d\Big) && (2.19)\\ &= b^{-1}\Gamma && (2.20) \end{aligned}$$

Notons qu'un terme intégral supplémentaire perturberait la passivité. Notre système est du premier ordre, mais si un champ de force lui est adjoint (ex : poids), on observera une erreur statique. Faute de terme intégral, on peut imaginer ajouter un terme en boucle ouverte dans la commande pour compenser l'erreur statique.

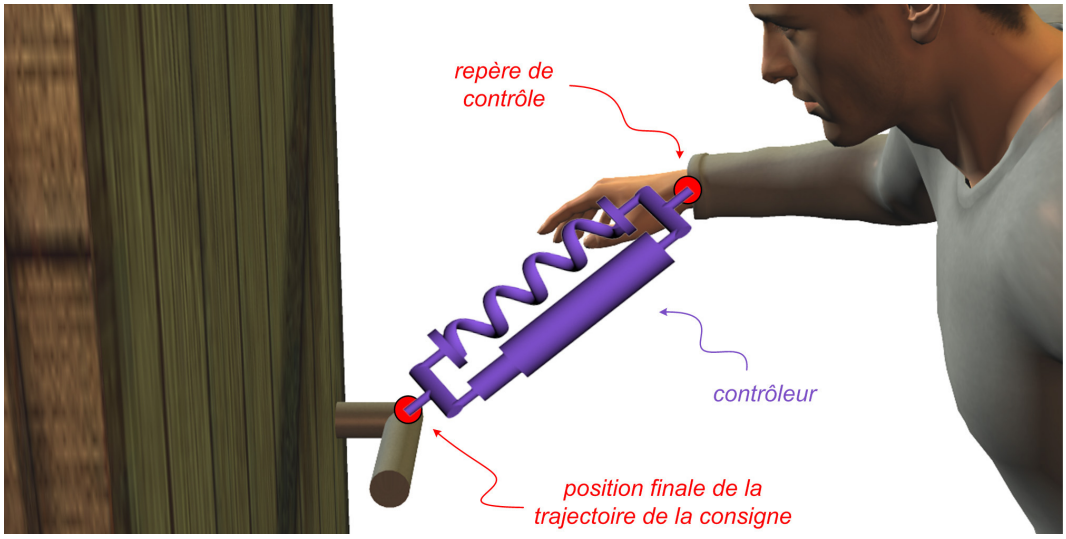

FIG. 2.6 – La consigne suit une trajectoire qui la mène à la poignée de porte, le contrôleur proportionnel-dérivé (exprimé dans l'espace de la tâche) permet au repère de contrôle de suivre la consigne. Ce repère contrôlé est ici lié à la main, et symbolisé par un disque rouge.

Rem : Si le repère de contrôle était relié à sa position cible, par un ressort amorti, la force générée aurait la même forme que l'asservissement proportionnel-dérivé (figure 2.6). Cela permet de représenter le contrôleur comme un simple ressort amorti reliant repère de contrôle et position de consigne (nous parlerons de couplage externe). Mais surtout cette analogie mécanique permet d'affirmer que le contrôleur est passif, puisqu'il est analogue à un ressort amorti du monde réel.

ATTENTION : la passivité est assurée ici dans le cas continu ! Dans le cas discret le problème est plus complexe. Le comportement du système peut bien sûr être étendu au cas où plusieurs repères de contrôle sont considérés.

Récapitulatif du choix du modèle

Correcteur proportionnel-dérivé

Avantages - inconvénients :

\+ système bien conditionné (pas de singularité)
\+ comportement intuitif : analogie mécanique au ressort amorti
− un soin particulier doit être apporté au choix des gains qui conditionne la passivité
− pas de terme intégral, ou en boucle ouverte (ils permettraient de compenser les erreurs statiques)

2.2.1.4 Contrôle interne

Sur l'architecture du système donnée schéma 1.32, cette fonctionnalité correspond aux blocs donnés illustration 2.7.

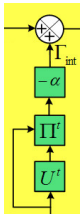

Fig. 2.7 – Blocs du schéma d'architecture 1.32, correspondant à la fonctionnalité décrite dans cette partie.

Si nous souhaitons à présent appliquer le poids à un mannequin possédant 35 degrés de liberté et 4 repères de contrôle (un à chaque main et pied). La jacobienne augmentée du système est de dimensions $(4 \cdot 6) \times 35$, le système est donc redondant. Aussi lorsque nous appliquons le poids, les pieds et les mains restent à leur position de consigne (car ils sont asservis), mais le reste du corps s'avachit.

Pour redonner une attitude naturelle au mannequin, nous pouvons lui appliquer un potentiel énergétique additionnel, destiné à contrôler les degrés de liberté jusqu'alors laissés en boucle ouverte. Il va ajouter une composante aux couples articulaires, dont la valeur sera déterminée par l'optimisation d'un critère $U(q, \dot{q})$ appelé *potentiel énergétique interne*. Son caractère interne fait référence au fait que les couples additionnels ne doivent pas perturber les tâches externes jugées prépondérantes. Prenons un exemple simple, un humain virtuel a pour consigne d'atteindre un interrupteur placé à côté de lui. Il n'y a aucun obstacle entre la position courante de la main et sa position de consigne. Dans un premier temps, nous souhaitons seulement qu'il atteigne l'interrupteur : le proportionnel-dérivé précédemment décrit, auquel on indique une trajectoire de consigne adéquate, s'acquitte de sa tâche de manière satisfaisante.

Imposons maintenant une seconde tâche, interne celle-ci, destinée à minimiser le mouvement. Si nous n'y prenons pas garde, les deux tâches vont rentrer en conflit - c'est-à-dire que le minimum de mouvement est réalisé au détriment de la tâche. Un compromis s'établira alors entre ces deux tâches et aucune des deux ne sera complètement réalisée : l'humain virtuel n'atteindra pas sa cible, alors qu'elle est tout à fait accessible.

Commande basée sur un projecteur dynamique

Pour éviter toute possibilité de perturbation de ce type, l'influence du potentiel est projetée dans le noyau de la tâche externe, par le biais de la projection Π (dont l'expression est donnée ci-dessous). Les couples internes sont alors donnés par :

$$\Gamma_{int} = -\alpha \Pi^t \frac{\partial U^t(q,\dot{q})}{\partial q}, \text{ avec } \alpha > 0. \tag{2.21}$$

Si l'on souhaite contrôler une tâche simultanément au contrôle interne, la vitesse V du repère de contrôle associé à cette tâche, de jacobienne J, est donnée par

$$V = J\dot{q} = Jb_a^{-1}\Gamma = Jb_a^{-1}\left(\Gamma_{task} + \Gamma_{int}\right) \tag{2.22}$$

$$= Jb_a^{-1}\left(J^t W_{task} - \alpha \Pi^t \nabla U\right), \tag{2.23}$$

donc pour éviter que le contrôle interne ne perturbe la tâche, Π^t est choisie comme la projection dans $\ker(Jb_a^{-1})$.

Le choix du potentiel va déterminer la manière avec laquelle la redondance est gérée. Les tâches internes sont exprimées de manière à optimiser un critère du type *minimisation des déplacements articulaires*, ou *minimisation des efforts articulaires*... Certains critères sont issus d'observations biomécaniques plus poussées, comme dans [KWDSS04], qui prend en compte des observations d'ordre biomécanique. Si le critère que nous souhaitons minimiser est la distance quadratique $U(q) = \frac{1}{2}\|q_d - q\|^2$ à une configuration désirée q_d, le correcteur interne s'écrit

$$\Gamma_{int} = \Pi^t\Big(k_{int}(q_d - q)\Big), \text{ avec } k_{int} = \alpha, \tag{2.24}$$

auquel pour les mêmes raisons liées à la passivité, que dans le cas externe, un terme de frottements visqueux peut être ajouté :

$$\boxed{\Gamma_{int} = \Pi^t\Big(k_{int}(q_d - q) + b_{int}(\dot{q_d} - \dot{q})\Big).} \tag{2.25}$$

Nous avons là un véritable outil de contrôle dans l'espace articulaire, ayant la même forme que son équivalent de l'espace opérationnel et qui grâce à Π ne le perturbe pas.

Notons qu'une intégration implicite, plus stable, peut être envisagée, mais ce type d'intégration implique un surcoût de calculs. La dynamique (2.4), alliée au correcteur interne (2.21), nous donne la dynamique interne suivante, réécrite en introduisant $f(q)$ comme indiqué ci-dessous :

$$\dot{q} = b_a^{-1}\Gamma_{int} = -\alpha b_a^{-1}\Pi^t \nabla U \tag{2.26}$$

$$= f(q). \tag{2.27}$$

Imaginons alors une intégration d'Euler implicite linéarisée (voir annexe C) :

$$q_{k+1} \simeq \left(f(q_k) + \frac{\partial f}{\partial q}(q_k)\dot{q}_k \delta t\right)\delta t + q_k, \tag{2.28}$$

où $f(q)$ est une matrice. Pour l'intégration, le gradient de $f(q)$ doit être calculé, c'est un tenseur d'ordre trois. Nous laissons son implémentation pour perspective.

En pseudo-code l'algorithme décrit précédemment peut s'exprimer ainsi :

Algorithme 2 : Bases du contrôle et de la simulation

Données :

H_d, q_d	:	position opérationnelle (resp. configuration) désirée
$\dot{x}_d$, $\dot{q}_d$	:	vitesse opérationnelle (resp. de configuration) désirée
b_a	:	matrice du frottement visqueux articulaire
k_c, k_{int}	:	raideur du couplage (resp. du contrôle interne)
b_c, b_{int}	:	matrice du frottement visqueux du couplage (resp. du contrôle interne)
δt	:	le pas d'intégration

Résultat : H : position courante du repère de contrôle

H : position à l'initialisation (par cinématique directe)
répéter
  pour *les m articulations* **faire**
    $\xi_{i/0(0)} = Ad_{0_{H_i}} \xi_{i/0(i)}$
  fin
  $J_i = \begin{bmatrix} \xi_{1/0(0)} & \cdots & \xi_{m/0(0)} \end{bmatrix}$
  $b = b_a + J^t b_c J$
  $\Gamma = J^t \Big(k_c \left(x_d - x \right) + b_c \dot{x}_d \Big)$
  Π : projection dont l'obtention est détaillée en 2.2.2.2
  $\Gamma_{int} = \Pi^t \Big(k_{int}(q_d - q) + b_{int}(\dot{q}_d - \dot{q}) \Big)$
  $\delta q = \delta t b^{-1} \Big(\Gamma + \Gamma_{int} \Big)$
  $\int q$ (Euler explicite)
  H : position courante du repère de contrôle (par cinématique directe)
jusqu'à *jusqu'à ce que l'utilisateur arrête la simulation*;

Commentaire : *Pour écrire cet algorithme nous avons fait trois choix restrictifs, pour simplifier sa compréhension :*
- *le choix du potentiel interne*
- *le choix de la méthode d'intégration*
- *nous n'avons envisagé qu'un seul repère de contrôle.*

[Roy99] propose une méthode alternative pour l'introduction d'un potentiel interne garantissant la passivité dans le cadre des hypothèses pour laquelle elle a été écrite.

Commande d'impédance augmentée

Cette méthode introduite par Laurent Royer dans [Roy99], étend la commande d'impédance de Hogan [Hog87]. Elle est appliquée à un bras redondant et permet de remplir deux exigences :

- la passivité à un port d'interaction avec l'environnement (supposé passif), pour assurer la stabilité du système
- le découplage de la dynamique interne et du mouvement de l'effecteur.

Soit un bras redondant, dont les m variables articulaires seront notées q, la vitesse et la jacobienne de l'effecteur étant données respectivement par V et J. Le modèle cinématique est donné par la relation $V = J\dot{q}$. La vitesse est augmentée en ajoutant r vitesses secondaires aux m vitesses cartésiennes, regroupées en un vecteur V_{int}, auquel on associe une jacobienne J_{int}. De la même manière la force opérationnelle W_{task}, appliquée à l'effecteur, sera complétée par une force secondaire W_{int}. Nous pouvons donc écrire

les relations suivantes dans notre système augmenté :

$$V_a = \begin{bmatrix} V \\ V_{int} \end{bmatrix}, \quad J_a = \begin{bmatrix} J \\ J_{int} \end{bmatrix} \quad W_a = \begin{bmatrix} W_{task} \\ W_{int} \end{bmatrix} \tag{2.29}$$

$$V_a = J_a \dot{q} \tag{2.30}$$

$$\Gamma = J_a^t W_a = J^t W_{task} + J_{int}^t W_{int}. \tag{2.31}$$

Afin d'introduire la notion de priorité, nous utilisons la loi de commande permettant de mettre en œuvre un contrôle externe et un contrôle interne découplé :

$$\Gamma = J^t W_{task} + \Pi^t \Gamma_0, \tag{2.32}$$

avec W_{task} force désirée à l'effecteur et

$$\Pi = I - J^g J \tag{2.33}$$

$$J^g = K^{-1} J^t \left(J K^{-1} J^t \right)^+, \tag{2.34}$$

où J^g est une inverse généralisée et K^{-1} une matrice de pondération symétrique définie positive, définie ci-dessous.

A ce stade les expressions de J_{int} et W_{int} ne sont pas encore connues. Nous allons reformuler Π, ceci nous permettra d'identifier J_{int} et W_{int}. En suivant [OCY97] nous écrivons :

$$\Pi = I - K^{-1} J^t \left(J K^{-1} J^t \right)^+ J \tag{2.35}$$

$$= N \left(N^t K N \right)^{-1} N^t K, \tag{2.36}$$

où N est une base du noyau de J. Intégrons (2.35) à la loi de commande (2.32), nous obtenons :

$$\Gamma = J^t W_{task} + K N \left(N^t K N \right)^{-1} N^t \Gamma_0. \tag{2.37}$$

Nous pouvons maintenant identifier W_{int} et J_{int}, en comparant (2.31) et (2.37). Laurent Royer propose de prendre pour effort de commande W_{int} :

$$W_{int} = N^t \Gamma_0 \tag{2.38}$$

$$J_{int} = \left(N^t K N \right)^{-1} N^t K, \tag{2.39}$$

W_{task} et W_{int} sont générés par des contrôleurs passifs (ex : proportionnels-dérivés).

Le système s'écrit alors[2] :

$$V_a = J_a \dot{q} \qquad \text{d'après (2.30)}$$

$$\dot{q} = b_a^{-1} \Gamma \qquad \text{d'après (2.4)}$$

alors

$$V_a = J_a b_a^{-1} \Gamma \tag{2.40}$$

$$= J_a b_a^{-1} J_a^t W_a, \tag{2.41}$$

avec

$$J_a b_a^{-1} J_a = \begin{bmatrix} J b_a^{-1} J^t & J b_a^{-1} J_{int}^t \\ J_{int} b_a^{-1} J^t & J_{int} b_a^{-1} J_{int}^t \end{bmatrix}, \tag{2.42}$$

[2] Royer réalise son découplage sur une dynamique du deuxième ordre, nous l'appliquons à notre dynamique du premier ordre.

pour découpler dynamique de l'espace opérationnel et dynamique interne (i.e. annuler les termes non-diagonaux de (2.42)), nous cherchons donc K tel que $Jb_a^{-1}J_{int}^t = 0$. C'est le cas si $K = b_a$:

$$Jb_a^{-1}J_{int}^t = Jb_a^{-1}KN\left(N^tKN\right) = 0,$$

car N est une base du noyau de J, donc $JN = 0$.

Le cas de la dynamique d'ordre deux est similaire à ceci près qu'il faut utiliser la relation suivante, en lieu et place de (2.30) :

$$V_a = J_a\dot{q} \;\Rightarrow\; \ddot{q} = J_a^{-1}\left(\dot{V}_a - \dot{J}_a\dot{q}\right), \tag{2.43}$$

dans ce cas, le découplage est uniquement inertiel.

Nous savons par (2.5) que le manipulateur en boucle ouverte est passif strictement par rapport à sa sortie. Il est ensuite bouclé par la rétroaction d'un correcteur générant les efforts de commande. L'environnement est supposé passif. Ainsi, si un correcteur passif est choisi, notre système complet sera passif. C'est le cas, comme nous l'avons vu précédemment, des contrôleurs de type proportionnel dérivé.

Le système en contre-réaction selon la méthode décrite ici est donné par l'illustration 2.8. On notera que les correcteurs sont encadrés de J, J^t et de N, N^t. Cette particularité permet à la méthode de conserver la passivité.

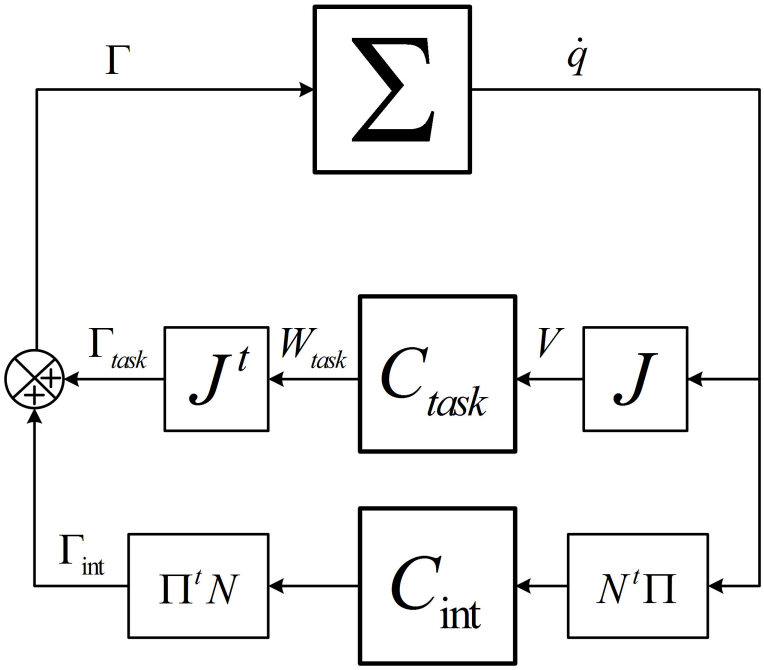

Fig. 2.8 – Commande d'impédance augmentée selon [Roy99]. Les contrôleurs sont encadrés de *transformateurs*.

[Roy99] montre que la commande basée sur une projecteur dynamique, première méthode décrite pour introduire un potentiel interne, est passive si l'inertie est correctement estimée (dans notre cas l'estimation est toujours correcte, puisqu'en virtuel, nous maîtrisons parfaitement les caractéristiques de notre environnement). La commande en impédance augmentée, dernière méthode décrite, propose une autre solution passive à l'introduction d'un potentiel interne. Cependant on peut distinguer deux types de ports d'interaction externes :

– les repères de contrôles : dont les caractéristiques sont déterminées dans une phase préliminaire à la simulation

– les contacts : qui apparaissent en cours de simulation.

Or si un port d'interaction externe apparaît en cours de simulation (comme c'est le cas lors d'interactions avec l'environnement), les deux méthodes précédentes sont incapables d'assurer la passivité.

Perte de passivité

Nous apportons à présent la preuve de la perte de passivité dans le cas de la commande basée sur un projecteur dynamique, si des ports externes additionnels interviennent de manière inattendue. Le choix de ce cas de figure n'est, bien sûr, pas anodin. Si nous souhaitons contrôler un mannequin, les repères de contrôle sont choisis. Ils représentent autant de ports externes. En revanche des ports externes, déterminés uniquement pendant la simulation, peuvent se créer en cas de contact avec l'environnement. Rappelons que l'interaction avec l'environnement est un des points clé de notre simulateur, nous nous devons donc de maîtriser les échanges d'énergie qu'elle implique.

Imaginons donc une structure redondante sur laquelle nous placerons un repère de contrôle dont la matrice jacobienne, le torseur des efforts, la vitesse opérationnelle et les efforts articulaires qu'il génère sont donnés respectivement par J_1, W_1, V_1 et Γ_1. Une tâche interne caractérisée par le potentiel $U(q)$, générant des couples articulaires Γ_{int}, est adjointe à ce système. Pour respecter les priorités naturelles, nous avons :

$$\Gamma_{int} = -\alpha \Pi_1^t \frac{\partial U^t}{\partial q},$$

la vitesse au port 1 est donnée par :

$$V_1 = J_1 \dot{q} = J_1 b_a^{-1} \Gamma = J_1 b_a^{-1} \left(J_1^t W_1 - \alpha \Pi_1^t \frac{\partial U^t}{\partial q} \right) \tag{2.44}$$

pour assurer le découplage, nous choisissons Π_1^t comme étant la projection dans $\ker\left(J_1 b_a^{-1}\right)$. Vérifions la passivité de ce système au port externe :

$$W_1^t V_1 = W_1^t J_1 b_a^{-1} J_1^t W_1 \geq 0, \text{ car } b_a \text{ positive.} \tag{2.45}$$

donc le système composé d'une tâche externe auquel une tâche interne est adjointe, est passif au port externe, si ce port était passif avant l'adjonction du potentiel interne.

Des problèmes surviennent cependant si de nouveaux ports externes sont ajoutés. Imaginons que notre structure entre en collision avec l'environnement à un port 2, de grandeurs associées J_2, W_2, V_2 et Γ_2. Dans un souci de simplification des équations, nous considérerons $W_1 = 0$, alors

$$W_2^t V_2 = W_2^t J_2 b_a^{-1} \Gamma \tag{2.46}$$

$$= W_2^t J_2 b_a^{-1} J_2^t W_2 - \alpha W_2 J_2 b_a^{-1} \Pi_1^t \frac{\partial U^t}{\partial q} \tag{2.47}$$

l'influence du potentiel interne ne disparaît pas ici comme en (2.45). Ainsi, projeter un potentiel interne peut conduire à la perte de passivité.

Nous pouvons également montrer la perte de passivité sur une simulation. Le système considéré est plan à trois degrés de liberté. Il possède deux ports d'interaction commandés en position suivant les deux directions de l'espace et un potentiel interne U, comme indiqué figure 2.9a. On impose un mouvement cyclique articulaire dont le cycle sur les deux premières articulations est visualisé figure 2.9b.

Dans un premier temps une seule tâche est mise en œuvre, la commande est donnée par $\Gamma = J_1^t W_1 - \alpha \Pi_1^t \nabla U^t$, avec $U = \|q_d - q\|^2$ construit pour ramener à une position articulaire q_d. On calcule W_1 et W_2 de telle manière que le système réalise le cycle articulaire fixé. On calcule $E = \int W_1 V_1 + W_2 V_2 dt$, son évolution donnée figure 2.10a permet de constater la passivité du système. Si un contact apparaît, un deuxième port est introduit, on observe alors le comportement suivant $\Gamma = J_1^t W_1 + J_2^t W_2 - \alpha \nabla U^t$, si on ne

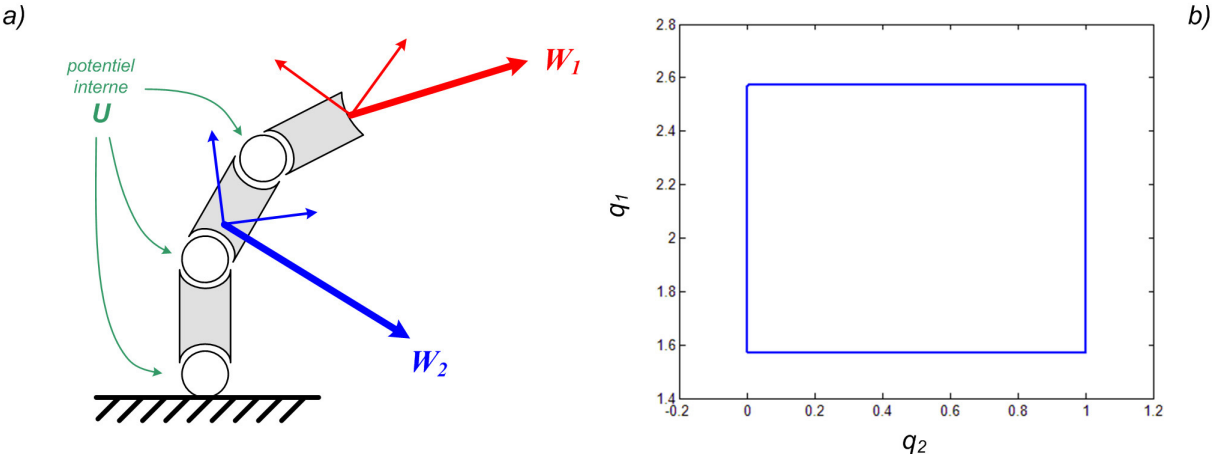

Fig. 2.9 – Système simulé et cycle articulaire de consigne sur les deux premières articulations.

projette pas l'influence du potentiel interne. L'évolution de E pour ce système est donnée figure 2.10b et permet de constater la passivité du système aux ports considérés. Nous projetons maintenant l'influence du potentiel interne dans le noyau de la tâche 1 grâce à la loi de commande $\Gamma = J_1^t W_1 + J_2^t W_2 - \alpha \Pi_1^t \nabla U^t$, avec Π_1^t choisi pour projeter dans $\ker\left(J_1 b_a^{-1}\right)$. 2.10c et permet d'observer une perte de passivité.

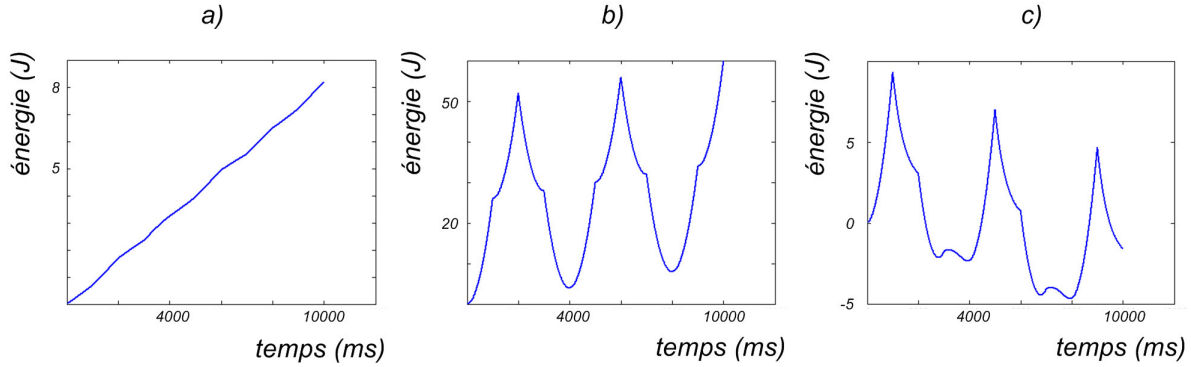

Fig. 2.10 – a) Seule la tâche 1 et le potentiel interne, dont l'influence est projetée, sont mis en œuvre. Le système est passif aux ports observés. b) Deux ports et prise en compte du potentiel interne, mais son influence n'est pas projetée. Le système est passif aux ports observés. c) Deux tâches et projection de l'influence du potentiel interne dans le noyau de la tâche 1. L'énergie diminue sans comportement asymptotique, ce comportement caractérise une perte de passivité.

L'interaction étant un de nos centres d'intérêt principaux, nous devons trouver des solutions au problème, il en existe quelques-unes :

- nous pouvons réduire l'influence du potentiel interne par le biais de α, de manière à se ramener dans le cas passif, c'est la méthode utilisée à présent. Un observateur énergétique peut être introduit afin de veiller à ce que la projection de $\frac{\partial U^t}{\partial q}$ n'impose pas un apport d'énergie trop grand.
- nous pouvons nous restreindre au cas des potentiels internes que nous appelerons *auto-projetants* (c'est-à-dire qu'il n'est pas nécessaire de projeter leur influence), qui sont de la forme :

$$\Pi_1^t \frac{\partial U^t}{\partial q} = \frac{\partial U^t}{\partial q}. \tag{2.48}$$

Dans ce cas (2.47) devient

$$W_2^t V_2 = W_2^t J_2 b_a^{-1} J_2^t W_2 - \alpha W_2 J_2 b_a^{-1} \frac{\partial U^t}{\partial q} \tag{2.49}$$

$$= \left\| \sqrt{(b_a^{-1})} \left(J_2^t W_2 - \alpha \frac{\partial U}{\partial q} \right) \right\|^2 + \alpha \dot{U} \tag{2.50}$$

donc

$$\int W_2^t V_2 dt = \int \left\| \sqrt{(b_a^{-1})} \left(J_2^t W_2 - \alpha \frac{\partial U}{\partial q} \right) \right\|^2 dt + \alpha U \geq -\beta^2, \tag{2.51}$$

la méthode est donc passive, si comme dans notre cas, $U(q, \dot{q}) \geq 0$. Cette méthode est à rapprocher de la notion d'intégrabilité du gradient d'un potentiel projeté [MIH91]. Pour trouver un approximation de ces potentiels auto-projetant, nous pouvons discrétiser la variété de configuration et travailler alors sur des projecteurs constants dont l'influence est pondérée. Dans ce cas le potentiel $V(q)$ s'exprime :

$$V(q) = U(q_0 + \Pi_1(q - q_0)) \tag{2.52}$$

$$= U(r), \tag{2.53}$$

alors

$$\frac{\partial V}{\partial q} = \frac{\partial U}{\partial r} \Pi_1. \tag{2.54}$$

– des projections étendues Π_E^t peuvent également être utilisées, elles projettent dans le noyau de tous les ports externes. Pour le cas de (2.47), si Π_E projette également dans le noyau de $J_2 b_a^{-1}$, l'influence du potentiel disparaît (notons que cela revient également à diminuer l'influence de la tâche interne, mais de manière dirigée cette fois). Cette méthode nécessite la connaissance de la position des contacts, ce qui ne pose pas de problème dans une simulation, où l'environnement est maîtrisé. Elle peut cependant s'avérer lourde en temps de calcul, si le nombre de contacts est élevé, comme ce sera le cas pour notre mannequin dans des endroits encombrés.

Dans l'architecture développée, le problème est géré par la première méthode.

Remarquons également que nous pouvons gérer la dynamique interne du système par le biais des coefficients de la matrice b_a. $b_a \dot{q}$ représentant un frottement visqueux, plus les coefficients sont faibles, plus les articulations sont mobiles (une force peu importante sera nécessaire pour les mettre en mouvement). Nous ne pouvons cependant atteindre la finesse de comportement donnée par l'introduction d'un potentiel interne.

Récapitulatif du choix du modèle

Correcteur proportionnel-dérivé articulaire

Avantages - inconvénients :

\+ implémentation aisée pour une intégration de type Euler explicite
\+ autorise plusieurs ports externes simultanément, de manière passive
− des précautions doivent être prises lors de l'implémentation pour assurer la passivité
− ne tient pas compte des efforts (donc pas de critère de type *minimisation des efforts articulaires* possible)

2.2.2 Multi-tâches conflictuel

Nous avons vu comment prendre en charge une tâche externe, ainsi qu'une tâche interne. Le problème dans l'espace opérationnel est sous-contraint et la redondance du mannequin est donc entièrement gérée par le contrôle interne. Pour manipuler une partie de ces degrés de liberté laissés à la charge du contrôle interne, il est également possible d'ajouter d'autres tâches externes. La redondance du système final sera donc moins importante.

Ajouter d'autres tâches externes, est dans une application comme celle de la gestion d'un mannequin, de première importance. Le contrôle mis en place, permet de gérer plusieurs cibles en même temps : il suffit d'ajouter des contrôleurs au système, qui généreront les forces de commande adéquates aux repères de contrôle.

Un problème se pose cependant dans le cas où nous spécifions des positions de consigne conflictuelles à deux repères de contrôle différents. En effet, si nous demandons à un mannequin d'atteindre un point avec la main droite et un autre point très distant du premier avec la main gauche, le mannequin ne pourra pas accomplir les deux tâches qui lui sont attribuées. Ce problème n'est pas spécifique aux univers virtuels et se retrouve également dans le monde réel.

Le problème peut être remarqué sur la figure 2.11, où dans le cas de gauche, il ne pleut pas et le squelette parvient à atteindre la poignée de la porte, sans aucun mal. En revanche, dans le second cas, il pleut mais l'opérateur spécifie une position de consigne du parapluie trop éloignée et le mannequin n'arrive pas à l'atteindre. Les deux tâches entrent en conflit et le système converge vers une position d'équilibre qui ne satisfait réellement aucune des deux tâches. Ce problème est similaire à celui qui nous invite à projeter l'influence de notre potentiel interne pour ne pas perturber la tâche principale, à la section précédente.

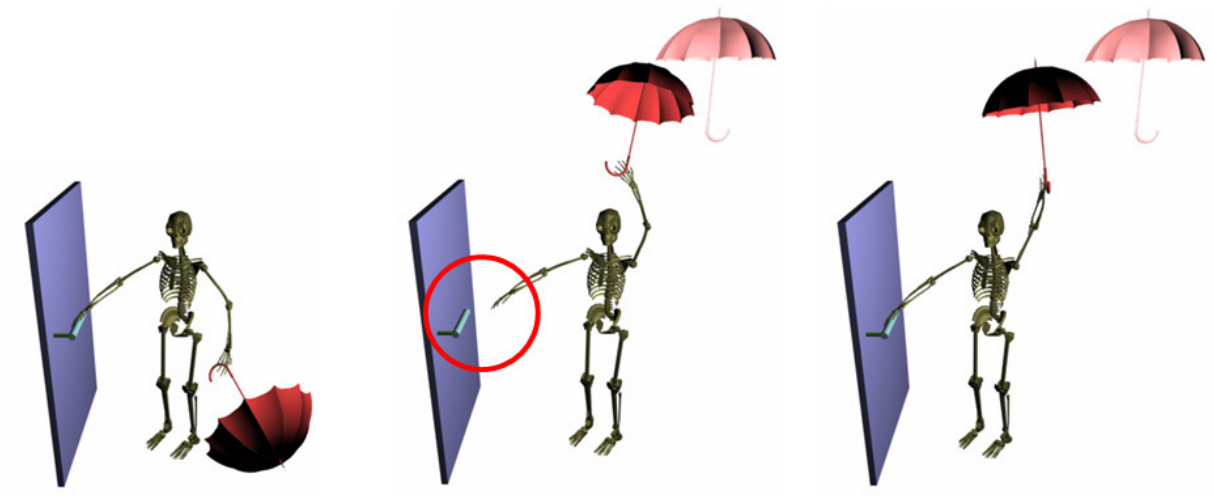

Fig. 2.11 – Cas de conflit entre deux tâches.

Ce problème des tâches conflictuelles est très fréquent dans notre cas d'application. Reprenons à ce titre notre exemple de la figure 1.30. Nous avons un acteur géant, observé par un dispositif de capture de mouvement, cherchant à animer un humain virtuel de petite taille. L'acteur peut être amené à faire des gestes amples, dans ces cas, si l'humain virtuel a les pieds et les mains pour respères de contrôle, il trouvera une position d'équilibre entre toutes ses cibles lors des mouvements larges de l'acteur. C'est-à-dire que dans ces cas, il sera soulevé du sol, figure 2.12. Ce type de comportement doit être évité.

Le problème passe d'un cas sous-contraint, à un cas sur-contraint, où il n'est plus possible de résoudre toutes les contraintes de manière simultanée. Dans la littérature, on trouve deux grandes manières d'y apporter une solution : la pondération et la mise en œuvre de priorités.

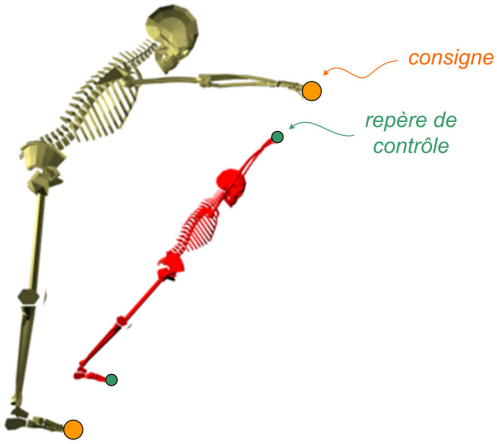

FIG. 2.12 – En appliquant les mouvements d'un acteur à un humain virtuel de morphologie différente. Certaines tâches peuvent entrer en conflit : elles ne peuvent pas être accomplies simultanément. Ici il est impossible à l'humain virtuel rouge d'atteindre à la fois les positions de consigne sur les pieds et les mains, données par le squelette blanc.

2.2.2.1 Résolution de tâches conflictuelles par pondération

Dans le cas cinématique où il y a plusieurs cibles, les différentes grandeurs sont concaténées de la manière suivante (on parle de problème *augmenté*) [Bae01] :

$$\delta x = \begin{bmatrix} \delta x_1 \\ \vdots \\ \delta x_n \end{bmatrix} \qquad J = \begin{bmatrix} J_1 \\ \vdots \\ J_n \end{bmatrix}. \tag{2.55}$$

En cas de conflit, le problème est sur-contraint, nous avons vu que l'optimisation quadratique à effectuer pour trouver notre solution était donnée par (1.42) : $\min_{\delta q} \frac{1}{2}\|J\delta q - \delta x\|^2$. Il est possible d'appliquer une métrique Q au calcul de cette norme quadratique, qui permettra de pondérer l'influence des différents repères de contrôle, le problème d'optimisation est alors donné par

$$\min_{\delta q} \frac{1}{2}\|J\delta q - \delta x\|_Q^2, \tag{2.56}$$

dont la résolution nous donne la pseudo-inverse pondérée dans le cas sur-contraint

$$\delta q = (J^t Q J)^{-1} J^t Q \delta x, \tag{2.57}$$

ce qui équivaut à travailler avec (1.42) avec les variables modifiées $\sqrt{Q}J$ et $\sqrt{Q}\delta x$. L'approche est illustrée figure 2.13.

Pour le cas dynamique (que ce soit en ordre un ou en ordre deux), cette approche par pondération est effectuée en donnant des raideurs différentes aux différents couplages.

2.2.2.2 Résolution de tâches conflictuelles par priorités

La mise en œuvre de priorités permet d'établir une hiérarchie, au sens strict entre les tâches. C'est-à-dire qu'une tâche de niveau inférieur est accomplie dans la mesure où elle ne perturbe pas les tâches jugées plus importantes. Elle est basée sur la notion de découplage de tâches, opération rendue possible par l'introduction de projecteurs. Nous avons déjà rencontré cet outil mathématique sous une forme simple lors de l'introduction du contrôle interne. Cette section, nous permettra d'étendre cette forme de

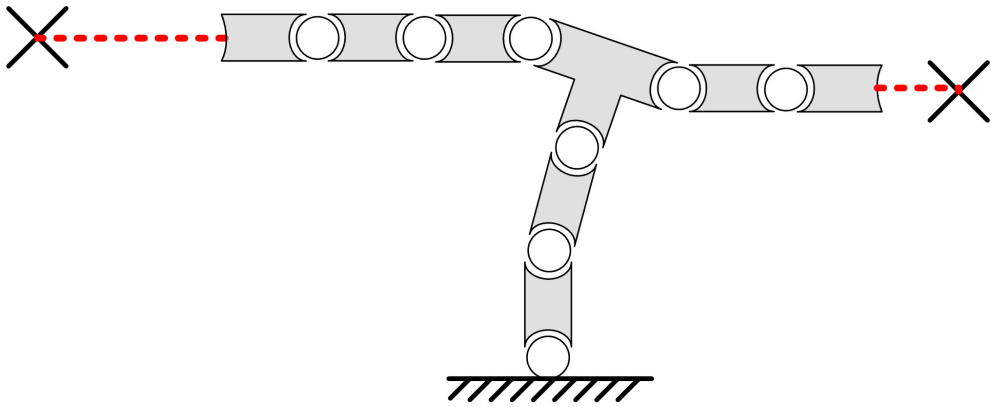

Fig. 2.13 – Résolution de conflit par pondération : aucune des tâches n'est parfaitement accomplie.

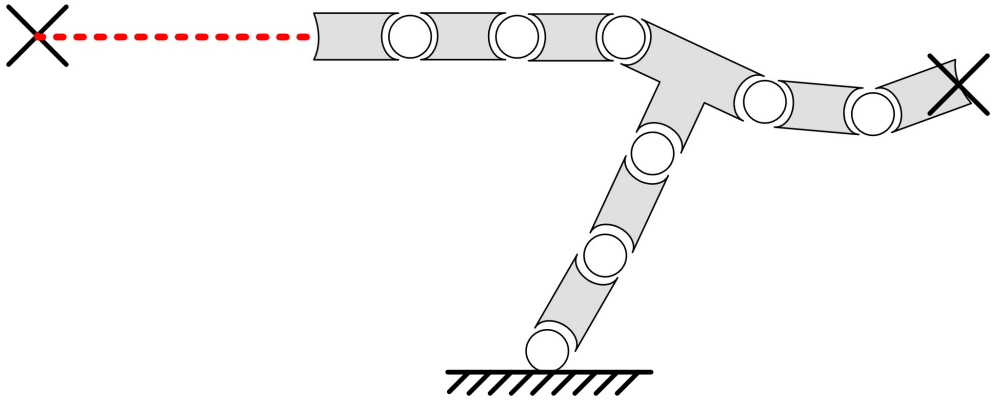

Fig. 2.14 – Résolution de conflit par priorité : la tâche la plus prioritaire est accomplie, les autres le sont dans la mesure du possible.

projection à des cas plus complexes, autorisant une hiérarchisation entre tâches externes. L'exemple de 2.13, résolu par les priorités est donné figure 2.14.

Nous présentons dans un premier temps une méthode de découplage de tâches externes dans le cas géométrique, donné dans [BB04], puis montrons son extension au cas dynamique donnée dans [SK04]. Nous montrons ensuite que les projecteurs qui permettent de mettre en œuvre ce découplage peuvent entraîner une perte de passivité du système.

Découplage de tâches externes

Cas géométrique

Le découplage de tâches externes, décrit dans [Nak91], a été appliqué au mannequin virtuel dans un contexte cinématique par Paolo Baerlocher [BB04], il a été étendu au cas dynamique par Luis Sentis [SK04]. Nous décrivons ici ces algorithmes, en suivant le fil de la chronologie.

La méthode de Baerlocher permet d'étendre les équations de la cinématique inverse décrites en 1.2.1, de manière à prendre en compte les priorités de tâches.

Imaginons dans un premier temps un système redondant à m paramètres articulaires notés q. Il possède un repère de contrôle dont la matrice jacobienne est donnée par J_1 et dont l'erreur de consigne

en position est notée δx_1. La correction des variables articulaires permettant de compenser l'erreur δx_1 est connue et est donnée par :

$$\delta q_1 = J_1^+ \delta x_1.$$

Introduisons maintenant un seconde tâche opérationnelle de grandeurs associées J_2 et δx_2. La résolution de la première tâche, induit un déplacement au niveau du second repère de contrôle. L'erreur à compenser n'est alors plus donnée par δx_2, mais par $\delta x_{prev(2)} = \delta x_2 - J_2 q_1$. Une première approche pour résoudre la tâche 2, dans le noyau de la tâche 1, est de résoudre la tâche 2 et de projeter le résultat dans le noyau de la tâche 1 :

$$\delta q = J_1^+ \delta x_1 + \Pi_1 J_2^+ \delta x_2, \tag{2.58}$$

où Π_1 est la projection dans le noyau de J_1.

Dans ce cas, la tâche 2 ne trouble pas la tâche 1, mais une partie de l'espace des solutions est occultée, nous lui préférerons donc la solution donnée par l'optimisation suivante ([MK85]) :

$$\begin{cases} \min\limits_{\delta q} \dfrac{1}{2} \|J_2 \delta q - \delta x_2\|^2 \\ \quad \text{sachant } \delta x_1 = J_1 \delta q \end{cases} \tag{2.59}$$

on a $\delta x_1 = J_1 \delta q$, donc

$$\delta q = J_1^+ \delta x_1 + \Pi_1 z \tag{2.60}$$

avec z choisi pour minimiser le terme $\frac{1}{2}\|J_2 \delta q - \delta x_2\|^2$. Grâce à (2.59) et (2.60) on peut écrire :

$$\min_{\delta q} \frac{1}{2} \|J_2 J_1^+ \delta x_1 + J_2 \Pi_1 z - \delta x_2\|^2, \tag{2.61}$$

les valeurs stationnaires sont données par

$$\frac{\partial \|J_2 J_1^+ \delta x_1 + J_2 \Pi_1 z - \delta x_2\|^2}{\partial z} = 0 \tag{2.62}$$

$$\left(J_2 J_1^+ \delta x_1 + J_2 \Pi_1 z - \delta x_2\right)^t J_2 \Pi_1 = 0 \tag{2.63}$$

on peut alors écrire

$$z = \left(\left(J_2 \Pi_1\right)^t \left(J_2 \Pi_1\right) \right)^+ \left(J_2 \Pi_1\right)^t \left(\delta x_2 - J_2 J_1^+ \delta x_1\right) \tag{2.64}$$

$$= \left(J_2 \Pi_1\right)^+ \left(\delta x_2 - J_2 J_1^+ \delta x_1\right), \tag{2.65}$$

car on peut vérifier que pour une matrice quelconque A, $(A^t A)^+ A^t = A^+$. Le résultat de l'optimisation est donc donné par

$$\delta q = J_1^+ \delta x_1 + \Pi_1 \left(J_2 \Pi_1\right)^+ \left(\delta x_2 - J_2 J_1^+ \delta x_1\right) \tag{2.66}$$

$$\boxed{\delta q = J_1^+ \delta x_1 + \left(J_2 \Pi_1\right)^+ \left(\delta x_2 - J_2 J_1^+ \delta x_1\right).} \tag{2.67}$$

L'extension à n tâches (celle d'indice 1 étant la plus prioritaire) est alors relativement directe, en prenant garde aux deux points suivants :

- la compensation de l'erreur de la tâche k doit être réalisée en prenant en compte les contributions des $k-1$ premières tâches

– une tâche de priorité k, ne devant troubler aucune des tâches 1 à $k-1$ de priorité supérieure, la projection devra être effectuée dans le noyau de toutes les jacobiennes des tâches de priorité supérieure. Pour ce faire nous introduirons $\Pi_{prev(k)}$, la projection dans le noyau des tâches de priorité supérieure à la tâche k (ex : $\Pi_{prev(2)} = \Pi_1$). Nous avons :

$$\Pi_{prev(k)} = I - J^+ J, \text{ avec } J = \begin{bmatrix} J_1 \\ \vdots \\ J_{k-1} \end{bmatrix}, \text{ appelé jacobienne augmentée.} \tag{2.68}$$

[Cli64] nous prouve que $\Pi_{prev(k)}$ peut également se mettre sous une forme incrémentale, plus propice au calcul :

$$\Pi_{prev(k)} = \Pi_{prev(k-1)} - J_{k-1}^+ J_{k-1}, \text{ avec } \Pi_{prev(1)} = I. \tag{2.69}$$

Une fois ces considérations prises en compte, il suffira de boucler sur chaque niveau de priorité, comme indiqué dans l'algorithme 3, issu de [BB04], pour obtenir un résultat comme ceux de P. Baelocher, dont un échantillon est donné figure .

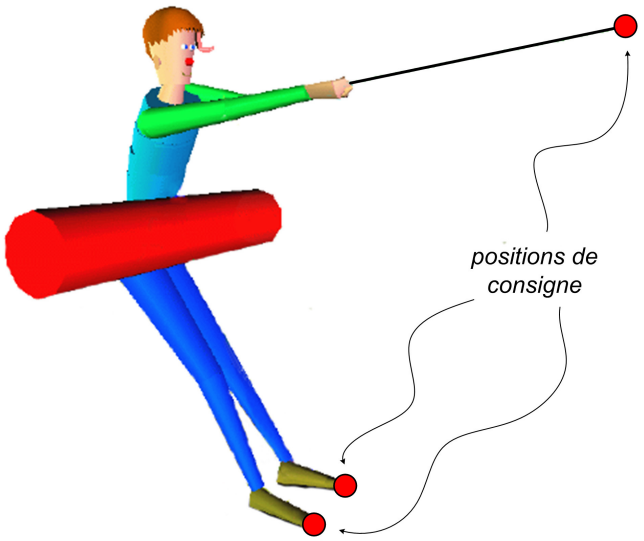

Fig. 2.15 – Les pieds du mannequin sont contraints avec une priorité haute à leur position, ils atteignent leur position de consigne. Un priorité plus basse est donnée à la consigne sur les mains, la consigne est satisfaite dans la mesure où elle ne trouble pas la contrainte sur les pieds. D'après [Bae01].

Algorithme 3 : Découplage de n tâches, dans un contexte cinématique

Données : H_{k_d} : position désirée de chaque repère de contrôle
$\xi_{i/0(i)}$: les axes de la structure

Résultat : H_k : position courante de chaque repère de contrôle

$q = 0$: variété de configuration
H_k : position des repères de contrôle à l'initialisation
répéter
  pour *toutes les tâches de priorité* $k \in [1, n]$ **faire**
    J_k : jacobienne de chaque tâche
    δx_k : erreur de position de chaque tâche
  fin
  $\delta q_0 = 0$
  $\Pi_{prev(1)} = I$
  pour *toutes les tâches de priorité* $k \in [1, n]$ **faire**
    $\delta x_{prev(k)} = \delta x_k - J_k \delta q_{k-1}$
    $\delta q_k = \delta q_{k-1} + \left(J_k \Pi_{prev(k)}\right)^+ \delta x_{prev(k)}$
    $\Pi_{prev(k+1)} = \Pi_{prev(k)} - J_k^+ J_k$
  fin
  $\delta q = \delta q_n$
  $H_k = \int \delta q$
jusqu'à *ce que l'utilisateur arrête la simulation*;

Commentaire : *Il est possible de mettre en œuvre des tâches de même niveau de priorité [Bae01].*

Cas dynamique

La version dynamique du découplage fonctionne suivant un principe similaire, mais requiert quelques précisions que nous allons maintenant donner. Illustrons cette approche par l'exemple suivant.

Le système à deux tâches externes a la loi de commande $\Gamma = \Gamma_1 + \Pi^t_{prev(2)}\Gamma_2$. La vitesse à la tâche 1, jugée la plus importante, s'écrit :

$$V_1 = J_1\dot{q} \tag{2.70}$$
$$= J_1 b_a^{-1}\Gamma, \text{ d'après l'équation (2.4)} \tag{2.71}$$
$$= J_1 b_a^{-1}\left(\Gamma_1 + \Pi^t_{prev(2)}\Gamma_2\right) \tag{2.72}$$
$$= J_1 b_a^{-1}\Gamma_1 + J_1 b_a^{-1}\Pi^t_{prev(2)}\Gamma_2. \tag{2.73}$$

Donc si $\Pi^t_{prev(2)}$ projette dans $\ker\left(J_1 b_a^{-1}\right)$, le découplage est assuré. Si on prend $\Pi_{prev(2)}$ avec la pseudo-inverse de J_1 pondérée par b_a (on note $\Pi_{prev(2)|b_a} = I - b_a^{-1}J_1^t\left(J_1 b_a^{-1}J_1^t\right)^+ J_1$), on a $\Pi^t_{prev(2)}$ qui projette dans $\ker\left(J_1 b_a^{-1}\right)$:

$$J_1 b_a^{-1}\Pi^t_{prev(2)|b_a} = J_1 b_a^{-1}\left(I - b_a^{-1}J_1^t(J_1 b_a^{-1}J_1^t)^+ J_1\right)^t \tag{2.74}$$
$$= J_1 b_a^{-1} - J_1 b_a^{-1}J_1^t(J_1 b_a^{-1}J_1^t)^+ J_1 b_a^{-1} \tag{2.75}$$
$$= 0, \tag{2.76}$$

donc $b_a^{-1}\Pi^t_{prev(2)}\Gamma_1 = 0$ et

$$V_1 = J_1 b_a^{-1}\Gamma_1, \tag{2.77}$$

la tâche 1 ne dépend donc pas de la tâche 2.

[SK04] propose d'utiliser la loi de commande suivante, pour le cas à n tâches :

$$\Gamma = \Gamma_1 + \Pi^t_{1|b_a}\left(\Gamma_2 + \Pi^t_{2|b_a}\left(\Gamma_3 + \ldots\right)\right), \tag{2.78}$$

en réutilisant la notation établie dans le cas purement cinématique et en posant $\Gamma_{prev(k)} = \Pi^t_{prev(k)}\Gamma_k$, nous obtenons :

$$\Gamma = \Gamma_1 + \Pi^t_{prev(2)|b_a}\Gamma_2 + \Pi^t_{prev(3)|b_a}\Gamma_3 + \ldots \tag{2.79}$$

$$\boxed{\Gamma = \Gamma_{prev(1)} + \Gamma_{prev(2)} + \Gamma_{prev(3)} + \ldots.} \tag{2.80}$$

Une jacobienne consistante avec les tâches de priorité supérieure, obtenue en projetant les vitesses articulaires, est alors définie :

$$J_{prev(k)} = J_k\Pi_{prev(k)}, \tag{2.81}$$

ceci permet de projeter, la dynamique articulaire du système dans l'espace de la tâche $prev(k)$ considérée. A cette fin, L. Sentis introduit, dans [SK04], la variable $V_{prev(k)} = J_{prev(k)}\dot{q}$, alors :

$$\dot{q} = b_a^{-1}\Gamma_{prev(k)}, \tag{2.82}$$

et

$$J_{prev(k)}\dot{q} = J_{prev(k)}b_a^{-1}\Gamma_{prev(k)} \tag{2.83}$$

$$V_{prev(k)} = J_{prev(k)}b_a^{-1}J^t_{prev(k)}W_{prev(k)}, \tag{2.84}$$

avec $W_{prev(k)}$, torseur d'effort de commande pour la tâche k.

Nous avons décrit les équations dans le cas de la dynamique du premier ordre, le principe est le même pour le deuxième ordre (voir section *Commande d'impédance augmentée* en 2.2.1.4).

A ce stade de l'étude, analysons la grandeur $V_{prev(k)} = \dot{x}_{prev(k)}$, introduite comme la vitesse calculée à partir de $J_{prev(k)}$. Cette vitesse n'est pas celle du point x_k, puisque celle-ci est donnée par J_k et que dans le cas général (notamment en cas de conflit) $J_{prev(k)} \neq J_k$. Cette vitesse ne correspond peut-être pas à celle d'un point fixe $x_{prev(k)}$ dans un des solides du système, comme indiqué figure 2.17, c'est-à-dire que $x_{prev(k)}$ n'est pas paramétrée par q. Pour que ce soit le cas, il faut que $\dot{x}_{prev(k)}$ soit intégrable [MIH91], ce qui n'est pas acquis sans condition. [MLSS94] nous donne une condition nécessaire et suffisante d'intégrabilité :

Théorème 2.2.1

$$\exists x,\ \frac{\partial x}{\partial q} = J_{prev(k)} \Longleftrightarrow \frac{\partial J_{prev(k)_{\alpha\beta}}}{\partial q_\gamma} = \frac{\partial J_{prev(k)_{\alpha\gamma}}}{\partial q_\beta}.$$

Ces conditions ne sont pas vérifiées de manière évidente, nous ne pouvons donc conclure de manière directe à l'existence d'un point $x_{prev(k)}$ appartenant à un des solides du système dont $V_{prev(k)}$ serait la vitesse.

Nous montrons maintenant que de manière générale, les projections ne permettent pas de conserver la propriété de passivité [RMM+05c].

Perte de passivité

Prenons l'exemple d'un système redondant dont nous étudions deux ports d'interaction. Chaque port est caractérisé par sa jacobienne, son torseur d'effort, sa vitesse opérationnelle et son influence sur les

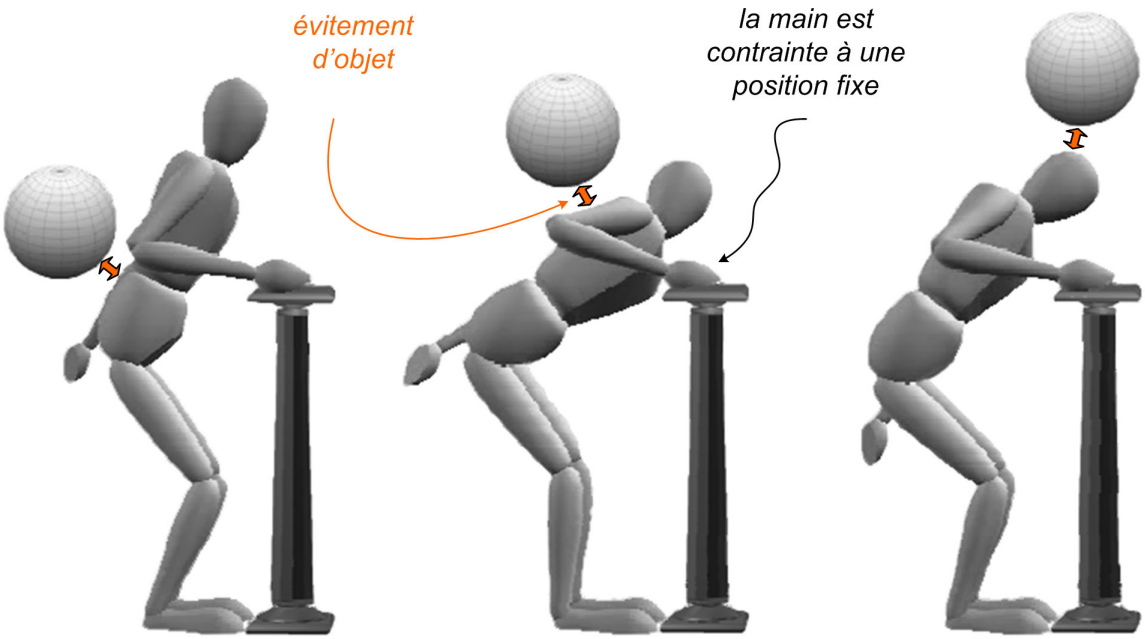

FIG. 2.16 – Evitement de collision alors que la main est contrainte à sa position et que l'équilibre est contrôlé. D'après [SK05].

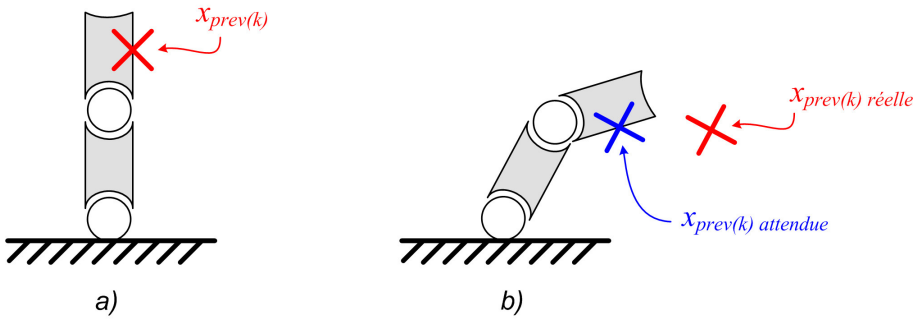

FIG. 2.17 – Si le jacobien n'est pas intégrable, $x_{prev(k)}$ peut varier de manière inattendue.

couples articulaires du système - notés respectivement J_1, W_1, V_1 et Γ_1, pour le port 1 et J_2, W_2, V_2 et Γ_2, pour le port 2.

Le système est donné figure 2.18. On se place dans les hypothèses suivantes :

$$\begin{cases} J_1 = \begin{bmatrix} A_1 & A_2 \end{bmatrix} \\ J_2 = \begin{bmatrix} A_1 & 0 \end{bmatrix} \\ A_1 \text{ carrée et de rang plein} \\ \ker(A_2) \text{ non restreint à } 0 \\ W_1 \neq 0,\ W_1 \in \ker(A_2^t) \\ W_2 = -2W_1 \end{cases} . \tag{2.85}$$

Nous donnons à la tâche 1 la priorité la plus grande, pour ce faire, nous introduirons Π_1, projecteur

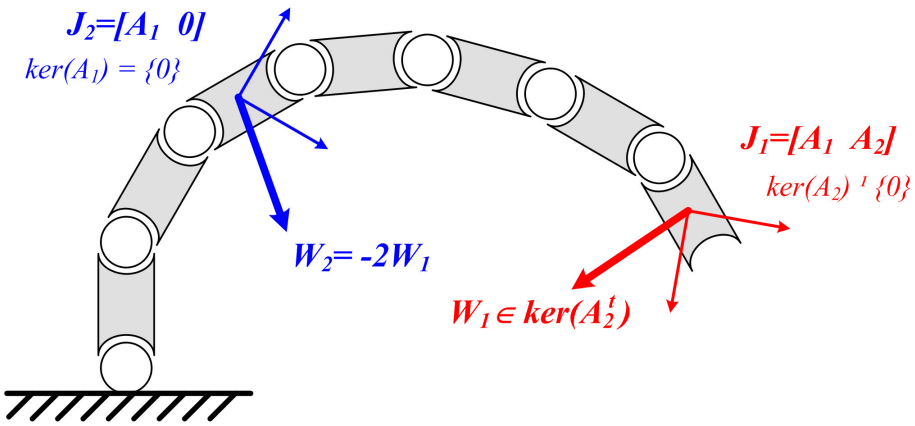

Fig. 2.18 – Exemple de perte de passivité par les projections.

dans le noyau de J_1. Nous pouvons donc écrire :

$$\Gamma_1 = J_1^t W_1 \tag{2.86}$$
$$\Gamma_2 = \Pi_1^t J_2^t W_2, \tag{2.87}$$

et

$$V_1 = J_1 \dot{q} = J_1 b_a^{-1}(\Gamma_1 + \Gamma_2) \tag{2.88}$$
$$= J_1 b_a^{-1} J_1^t W_1 + J_1 b_a^{-1} \Pi_1^t J_2^t W_2, \tag{2.89}$$

avec $\Pi_1 = \Pi_{1|b_a}$.

Analysons la passivité de ce système. Pour assurer cette propriété, nous savons d'après 2.1, que l'équation suivante doit être vérifiée :

$$\forall (W_1, W_2),\ \int_0^t \left(W_1^t V_1 + W_2^t V_2\right) dt \geq -\beta^2,\ \beta \in \mathbb{R}$$

développons alors $W_1^t V_1 + W_2^t V_2$:

$$W_1^t V_1 + W_2^t V_2 = (W_1^t J_1 + W_2^t J_2)\dot{q} \tag{2.90}$$
$$= (W_1^t J_1 + W_2^t J_2) b_a^{-1} (J_1^t W_1 + \Pi_1^t J_2^t W_2) \tag{2.91}$$
$$= W_1^t J_1 b_a^{-1} J_1^t W_1 + W_2^t J_2 b_a^{-1} J_1^t W_1 + W_2^t J_2 b_a^{-1} \Pi_1^t J_2^t W_2 \tag{2.92}$$
$$= \left(W_1^t J_1 + \frac{1}{2} W_2^t J_2\right) b_a^{-1} \left(J_1^t W_1 + \frac{1}{2} J_2^t W_2\right) - \frac{1}{4}\left(W_2^t J_2 b_a^{-1} J_2^t W_2\right) + \left(W_2^t J_2 b_a^{-1} \Pi_1^t J_2^t W_2\right). \tag{2.93}$$

On a

$$J_1^t W_1 + \frac{1}{2} J_2^t W_2 = \begin{bmatrix} A_1^t \\ A_2^t \end{bmatrix} W_1 + \frac{1}{2} \begin{bmatrix} A_1^t \\ 0 \end{bmatrix} W_2 \tag{2.94}$$
$$= 0 \tag{2.95}$$

donc

$$W_2^t J_2 b_a^{-1} \Pi_1^t J_2^t W_2 = 4 W_1^t J_1 b_a^{-1} \Pi_1^t J_1^t W_1 \tag{2.96}$$
$$= 0, \tag{2.97}$$

puisque Π_1^t projette dans $\ker(J_1 b_a^{-1})$, donc

$$W_1^t V_1 + W_2^t V_2 = -\frac{1}{4}\left(W_2^t J_2 b_a^{-1} J_2^t W_2\right), \tag{2.98}$$

or on a

$$J_2^t W_2 = -\begin{bmatrix} 2A_1^t W_1 \\ 0 \end{bmatrix} \neq 0, \text{ car } A_1 \text{ de rang plein.} \tag{2.99}$$

Donc, dans cet exemple le système (dans le cas continu), crée de l'énergie aux ports étudiés du fait de la projection.

Nous pouvons également montrer la perte de passivité sur un exemple plus simple. Le système considéré est plan à trois degrés de liberté. Il possède deux ports d'interaction commandés en position suivant les deux directions de l'espace, comme indiqué figure 2.19a.

On impose un mouvement cyclique articulaire dont le cycle sur les deux premières articulations est visualisé figure 2.19b. Dans un premier cas, la tâche 2 n'est pas projetée, la commande est donnée par $\Gamma = \Gamma_{task} = J_1^t W_1 + J_2^t W_2$ (on calcule W_1 et W_2 de telle manière que le système réalise le cycle articulaire fixé). On calcule $E = \int W_1 V_1 + W_2 V_2 dt$. Son évolution donnée figure 2.19c permet de constater la passivité du système.

Nous projetons alors la tâche 2 dans le noyau de la tâche 1 grâce à la loi de commande $\Gamma = J_1^t W_1 + \Pi_1^t J_2^t W_2$, avec Π_1^t choisi pour projeter dans $\ker(J_1 b_a^{-1})$. L'énergie observée est cette fois donnée figure 2.19d et permet d'observer une perte de passivité : **dans le cas général, les projections impliquent une perte de passivité**.

La perte de passivité a été envisagée dans le cas de la dynamique continue du premier ordre. Au deuxième ordre, nous observons bien sûr le même phénomène, qui sera encore accru, s'il y a lieu, par le caractère discret des équations.

Il convient cependant de tempérer ces résultats théoriques à la vue des observations de l'exemple donné dans [SK04]. Ils indiquent que le système semble "utilisable" dans des conditions correctes même si d'un point de vue théorique la passivité n'est pas assurée.

Mécanismes virtuels : mode de commande contraint

Nous avons vu que dans le cas général les projections n'étaient pas passives. Nous présentons maintenant une méthode alternative permettant de construire les projections différemment. Ces projecteurs permettent de conserver la propriété de passivité en assurant un découplage partiel. Nous décrivons comment les construire et les mettre en œuvre. Nous montrons qu'il est possible de les assimiler à des mécanismes à part entière, appelés *mécanismes virtuels*, du fait de leur analogie avec de vrais systèmes mécaniques.

Ces mécanismes virtuels ont été introduits dans le cadre de la téléopération par [Jol97], ils sont à rapprocher des fonctions de tâche et des techniques d'asservissement visuel [ECR92]. Ils permettent de guider (ou contraindre) le bras maître lors de l'exécution d'une tâche de précision. Pour nous replacer dans le contexte de la téléopération, nous pouvons imaginer un dispositif à base de capture de mouvement comme étant le maître, qui commande un humain virtuel esclave du système téléopéré. Nous voyons maintenant leur fonctionnement dans le cadre du contrôle d'humains virtuels.

Prenons pour exemple un mannequin possédant un repère de contrôle sur une main d'une part et un mécanisme virtuel constitué d'une simple liaison prismatique d'autre part, comme indiqué figure 2.20. Le repère de contrôle du mannequin peut atteindre n'importe quel point de $SE(3)$, alors que l'effecteur du mécanisme virtuel est contraint sur une droite. Nous *attachons* alors le repère de contrôle à l'effecteur du mécanisme virtuel, il est alors contraint sur l'espace atteignable du mécanisme virtuel. Les équations sont fidèles à ce schéma de principe.

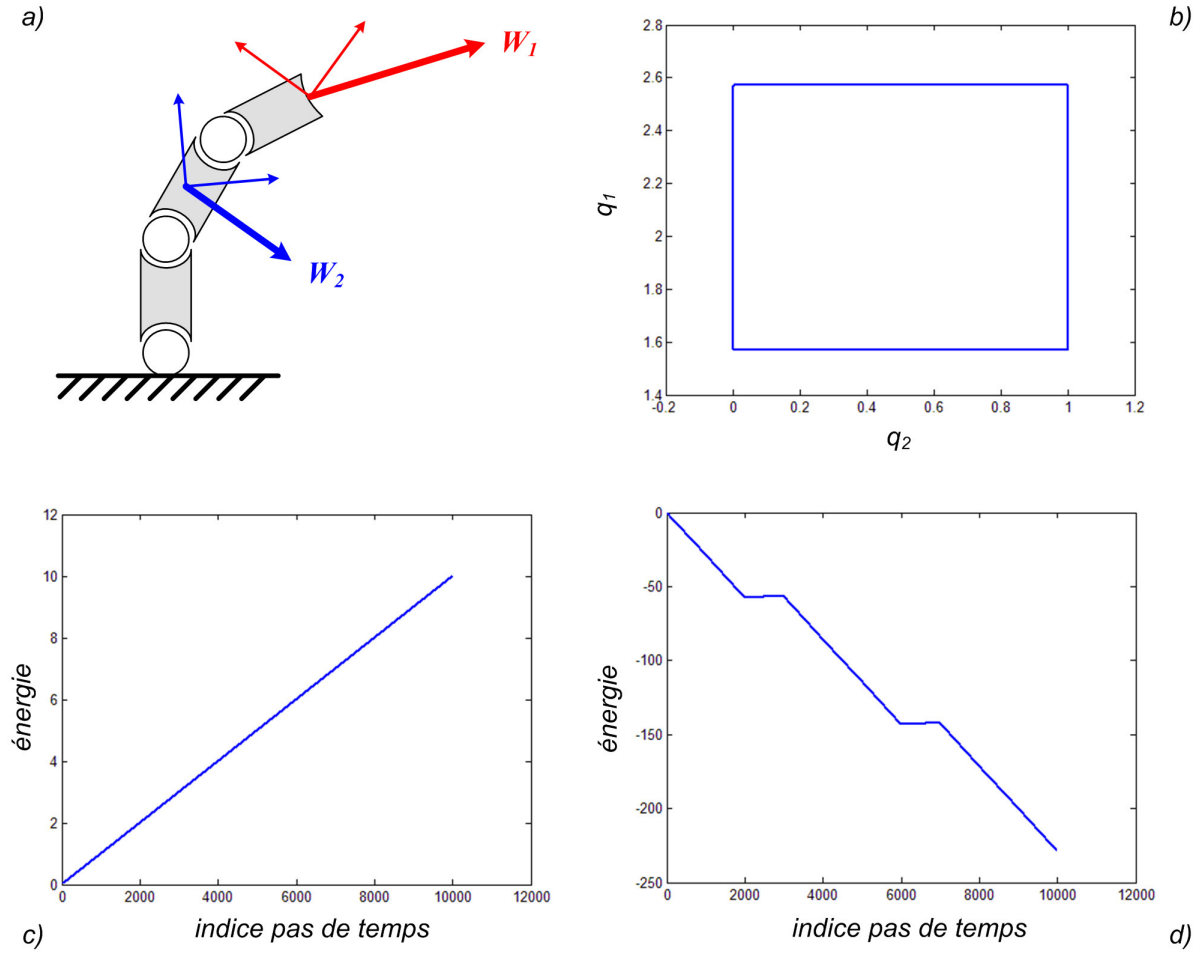

Fig. 2.19 – Perte de passivité du fait d'une projection. En haut à gauche on voit le système simulé. En haut à droite on visualise le cycle réalisé par les deux premières variables articulaires lors du mouvement. En bas à gauche on a l'énergie du système sans projection, le système est passif. En bas à droite l'énergie du système avec projection de la tâche 2 : l'énergie diminue sans comportement asymptotique, ce comportement caractérise une perte de passivité.

Le caractère passif des mécanismes virtuels est montré dans [Jol97]. Il y est également montré que les mécanismes virtuels sont localement équivalents à des projecteurs. Ils peuvent permettre le découplage localement dans $\mathbb{R}^n$.

Les mécanismes virtuels peuvent être vus comme des contraintes bilatérales, qui permettent de contraindre le mouvement du repère de contrôle considéré. Nous pourrions obtenir un comportement similaire en projetant l'erreur de consigne ([Jol97] prouve d'ailleurs que les mécanismes virtuels tendent vers les projecteurs dans ce cas), mais les mécanismes virtuels garantissent la passivité en sus.

Si ces méthodes permettent de distinguer dans l'ensemble des projections, un sous-ensemble passif, celui-ci se borne, a priori, aux projections servant de guides et ne contient pas en général, celles qui auraient permis le découplage.

L'utilité du découplage apparaît lorsque toutes les tâches ne peuvent être satisfaites simultanément. C'est-à-dire que les mouvements désirés ne sont pas faisables. Une question se pose alors dans ces cas conflictuels : souhaitons-nous contrôler tant bien que mal l'humain virtuel dans des cas où la tâche

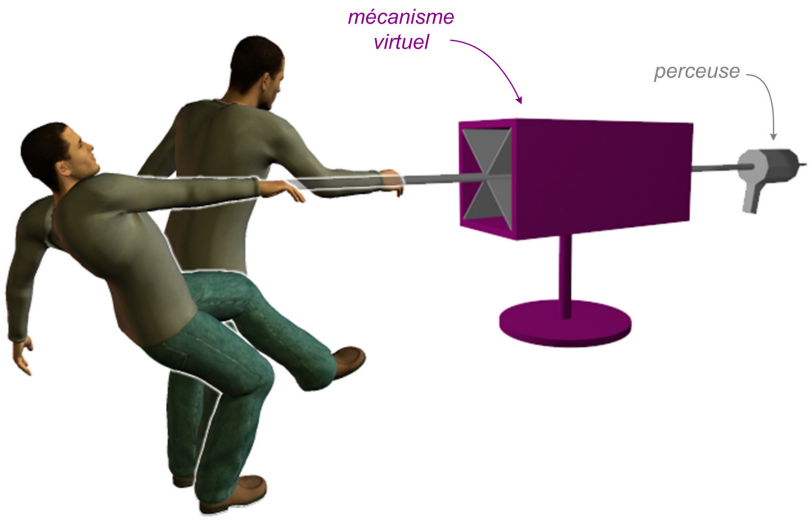

FIG. 2.20 – Principe d'un mécanisme virtuel : une fois la main attachée à l'effecteur du mécanisme virtuel (en rouge), la main de l'humain virtuel est contrainte sur l'espace atteignable du mécanisme virtuel. Dans le cas de l'exemple, la perceuse reste alignée sur son axe même en cas de perturbation.

impartie n'est de toutes façons pas réalisable ? Le cas de l'humain virtuel à but industriel est cerné. Nous souhaitons connaître si, avec un humain virtuel de morphologie donnée, les mouvements désirés sont faisables ou non et être averti en cas d'impossibilité d'atteindre les cibles. Cette fonctionnalités est d'ores et déjà envisageable, sans découplage.

Les considérations qui suivent laissent cependant penser qu'il peut exister des méthodes de découplage par les mécanismes virtuels :

- les guides mis en œuvre de manière adéquate, peuvent nous permettre de nous passer partiellement du découplage, c'est une de leurs raisons d'être. C'est ce que nous pouvons vérifier sur l'exemple 2.21.

- l'une des différences principales entre les projecteurs pour le découplage et les projecteurs servant de guide est que les premiers dépendent de la configuration du système, alors que les derniers varient avec la position du repère de contrôle. Nous avons montré que pour le cas des potentiels internes auto-projetants, nous pouvions approximer localement les projecteurs par des projecteurs constants pondérés (2.52), ces projecteurs sont invariants. On peut donc espérer trouver une méthodologie d'assistance à la construction de mécanismes virtuels, permettant le découplage par le biais des mécanismes virtuels. Nous laissons ces travaux en perspective.

Cette méthode offre, grâce à l'introduction de nouveaux modes de commande contraints, un large panel de possibilités, dont nous pourrons apprécier la richesse en 4.1.

Conclusion

Dans ce chapitre nous avons d'abord identifié les éléments influents sur le mouvement humain. Nous avons relevé trois sources principales, que sont les tâches de l'espace opérationnel, la manière avec laquelle la redondance est gérée et enfin les contraintes physiques et biomécaniques.

Ceci nous a permis de mettre en place une première architecture d'animation. Elle comporte un bloc de simulation bien identifié, permettant d'émuler un comportement physique, que nous avons bouclé sur un bloc de contrôle permettant à notre humain virtuel d'atteindre sa consigne.

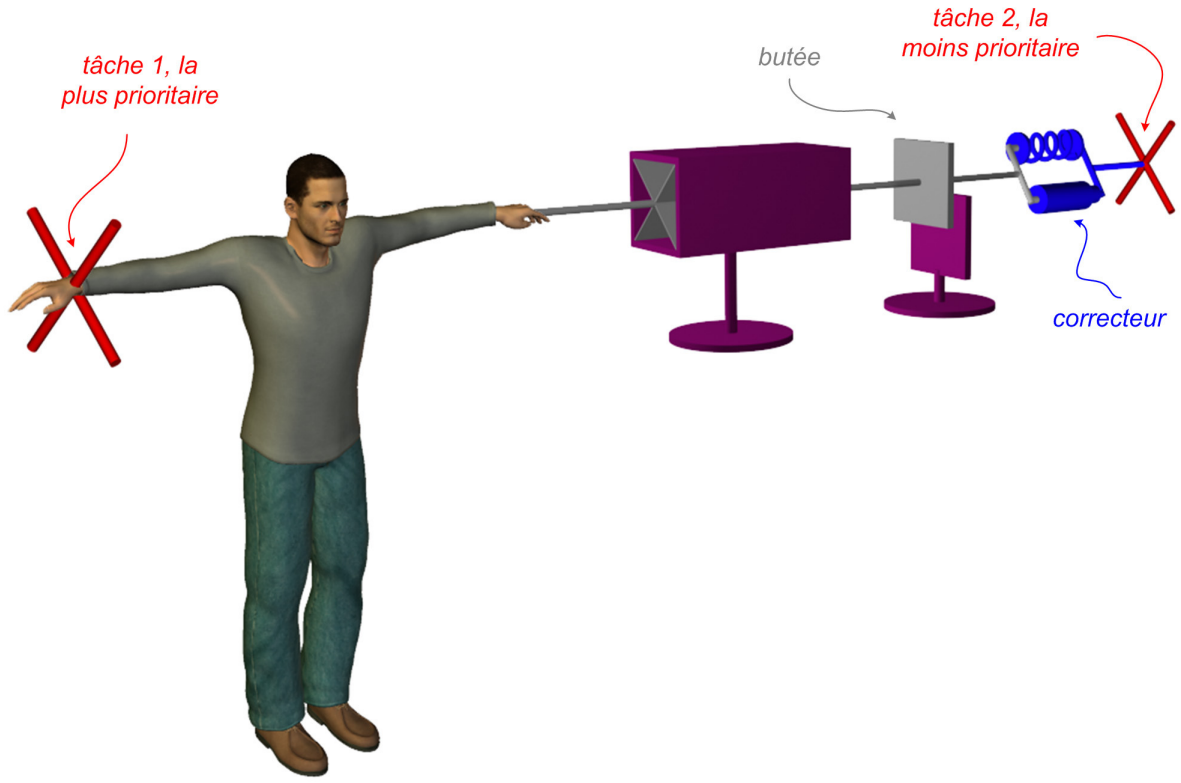

Fig. 2.21 – La butée placée sur le mécanisme virtuel positionné de manière adéquate permet de limiter l'influence de la tâche 2 sur le mannequin. C'est une méthode envisageable pour le découplage.

La prise en compte des contraintes est traitée dans le chapitre qui suit, en raffinant l'architecture déjà en place.

CHAPITRE 3

Gestion des contraintes unilatérales

Dans le chapitre précédent, nous avons identifié trois facteurs influençant les mouvements humains. Nous avons décrit une implémentation de deux d'entre-eux (le contrôle de tâches, et le contrôle interne). Ce chapitre fait l'objet de l'implémentation du dernier facteur d'influence : la gestion des contraintes unilatérales.

Un dénominateur commun sous tend toutes les contraintes que nous allons évoquer ici. Ainsi, si les contraintes sur la position des repères de contrôle évoquées dans la section précédente étaient du type bilatéral *(ce sont des égalités), celles que nous allons traiter ici sont toutes* unilatérales *(ce sont des inégalités). Ces dernières, comme indiqué en 1.4.2), concernent plus exactement le contact, les limites articulaires et l'équilibre.*

Nous décrivons dans un premier temps l'architecture du solveur de contraintes unilatérales mis en œuvre. Puis détaillons l'implémentation particulière des limites articulaires et du contact.

Nous proposons ensuite des méthodes de gestion de l'équilibre. Dans un premier cas simple d'abord. Puis nous élargissons les hypothèses d'étude à des cas plus complexes.

Cette fonctionnalité est retrouvée dans le sous-bloc bleu ciel (marqué LCP*, et* Solveur de contraintes unilatérales*) des schémas 1.32, et 1.31.*

3.1 Solveur de contraintes unilatérales

Aparté - les contraintes en deux mots

Les mécanismes virtuels, introduits au chapitre précédent, permettent de mettre en œuvre des *contraintes bilatérales*. Les contraintes dont il s'agit ici sont *unilatérales*. De manière générale, pour un état x appartenant à un espace d'état X donné, la différence entre ces deux types de contraintes est donnée figure 3.1, et exprimée de la manière suivante :

- **les contraintes bilatérales :** permettent de restreindre le mouvement à un sous-espace de X. Elles s'expriment en donnant l'équation de l'hypersurface sur laquelle doit se déplacer x.
- **les contraintes unilatérales :** permettent de restreindre le mouvement à une restriction de X. Elles s'expriment en donnant une inéquation, que x doit vérifier.

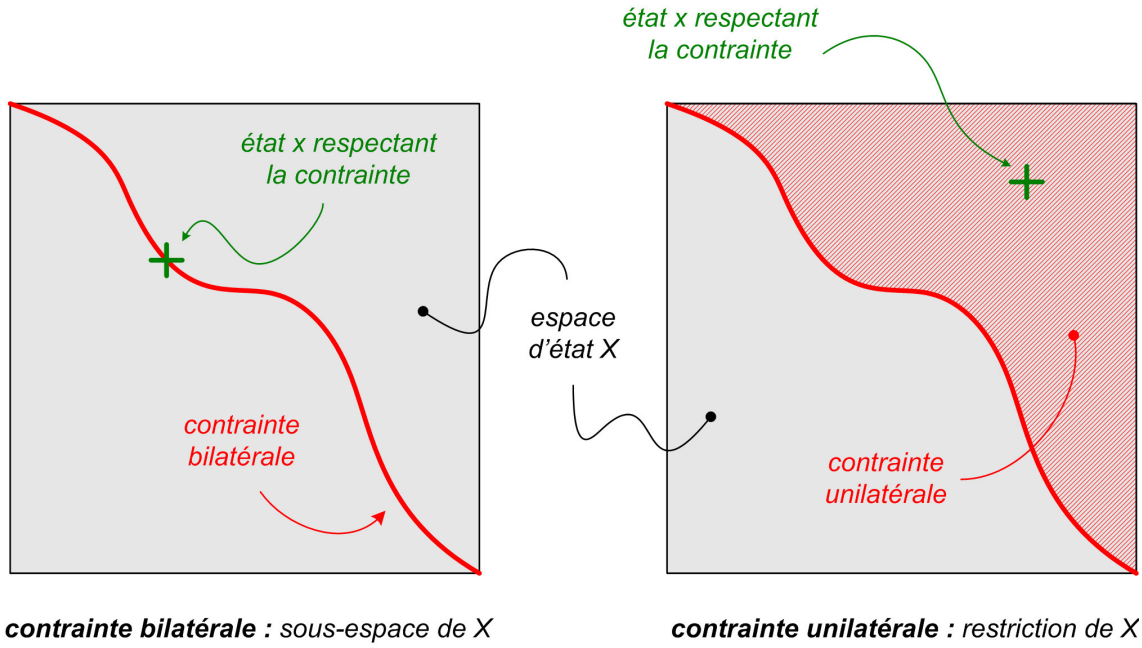

Fig. 3.1 – Comparaison des contraintes bilatérales et unilatérales.

3.1.1 Gestion des contacts : état de l'art

Le choix de la méthode de résolution des contraintes sera largement conditionné par les possibilités d'interaction souhaitées pour le système. Nous avons décrit en 1.2.1.4 des applications autorisant l'interaction des humains virtuels avec l'environnement. Celles-ci s'appuient sur des méthodes de gestion du contact*, que nous allons décrire brièvement. Une méthode de* gestion de contacts *génère la réaction (c'est-à-dire les efforts, les impulsions...) qui, une fois appliquée aux solides en contact, permet d'empêcher que ces solides ne s'interpénètrent.*

Différentes méthodes ont été développées dans un contexte non spécifique à l'animation de personnages. Mirtich [MC94], [Mir95] introduit ainsi une dynamique impulsionnelle, qui corrige les violations de contrainte grâce à des impulsions de force. Une impulsion est générée de manière à garder le solide en dehors de la contrainte, ainsi le solide est en permanence en mouvement balistique. Par définition, la notion d'impulsion n'existe pas en quasi-statique.

Il existe également des méthodes permettant d'exprimer la contrainte du contact sous la forme d'un

LCP (notion présentée au chapitre 3.1.2). Ainsi Baraff [Bar94], propose une formulation en accélération de ce problème de complémentarité (c'est-à-dire que l'accélération intervient comme contrainte du problème). Ruspini [RK97] propose également une formulation en accélération et introduit la jacobienne de contact, qui permet de projeter le comportement du système dans l'espace du contact. Nous avons choisi une approche similaire à cette dernière que nous détaillons en 3.1.2 et 3.2.1.

3.1.2 Solveur de contraintes unilatérales par LCP

Sur l'architecture du système donnée schéma 1.32, cette fonctionnalité correspond aux blocs donnés illustration 3.2.

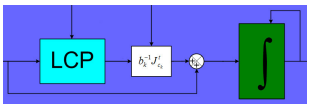

FIG. 3.2 – Blocs du schéma d'architecture 1.32, correspondant à la fonctionnalité décrite dans cette partie.

Toutes les contraintes que nous allons exposer s'expriment sous la forme de *problèmes de complémentarité linéaires* (*Linear Complementarity Problems en anglais* ou *LCP*). Notre solveur de contraintes unilatérales est de ce type.

Le choix d'exprimer les contraintes unilatérales sous forme de LCP a été motivé par l'existence d'algorithmes efficaces de résolution des LCP qui fournissent des réponses *quasi-exactes*, et la possibilité de résoudre toutes les contraintes simultanément. Ce dernier point est important car toutes les contraintes unilatérales sont de même priorité. On ne peut donc se permettre d'en résoudre une au détriment d'une autre - comme ce serait le cas si la résolution n'était pas simultanée.

Nous décrivons ici les problèmes de complémentarité linéaire (LCP) dont la résolution nous permettra d'apporter une réponse aux contraintes unilatérales.

Définition 3.1.1 *Un problème de complémentarité linéaire* (ρ, M) *s'exprime de la manière suivante [CPR92] :*

Soit une matrice M *carrée,* ρ *et* w *des vecteurs de dimension idoine, trouver* z *tel que :*

$$w = Mz + \rho \qquad z \geq 0 \qquad w \geq 0 \qquad w^t z = 0 \tag{3.1}$$

Une version abrégée notée $0 \leq Mz + \rho \perp z \geq 0$, est souvent utilisée. Notons que z peut être vue comme une variable de commande et w comme la distance à une contrainte à respecter.

Aparté - la complémentarité en deux mots

La relation de complémentarité ou orthogonalité $w^t z = 0$ de (3.1) signifie simplement qu'à tout moment au moins une des deux inconnues du LCP (w et z) est nulle.

Dans les différents cas que nous étudions, w exprime une distance $d(q)$ au contact, une distance aux limites articulaires, ou une distance à la perte d'équilibre. Cette distance est appelée *modèle géométrique de la contrainte*. z représente les couples articulaires Γ_c de compensation, générés de manière à faire

respecter la contrainte. Le LCP s'écrit donc :

$$d(q) = fonction(\Gamma_c), \quad \begin{cases} d(q) \geq 0 \\ \Gamma_c \geq 0 \\ \Gamma^t d(q) = 0 \end{cases} . \tag{3.2}$$

Récapitulatif du choix du modèle

Résolution de contraintes unilatérales par LCP (par l'algorithme de Lemke)

Avantages - inconvénients :

\+ résolution quasi-exacte des contraintes unilatérales
\+ le formalisme LCP permet de décrire de nombreuses contraintes unilatérales différentes (contact, limites articulaires, équilibre...)
− la matrice du LCP doit être copositive pour que l'algorithme de Lemke utilisé pour résoudre le LCP trouve une solution à coup sûr si elle existe (ça n'est pas toujours le cas)
− impose de linéariser les problèmes non-linéaires

Architecture du solveur de contraintes unilatérales

$d(q)$ n'est malheureusement connue, ou observable qu'au pas courant k, or nous voulons faire respecter la contrainte au pas $k+1$: $d(q_{k+1}) \geq 0$. Aussi, comme dans le cas de l'intégration, une approche implicite et une approche explicite sont envisageables, ainsi que de nombreuses possibilités intermédiaires (le choix de la méthode d'implémentation conditionne le type de solveur mis en place) :

– [DDKA05] et [Dur04b], mettent en œuvre un schéma à deux étages, où la physique et l'intégration sont réalisées deux fois par pas de temps, comme indiqué sur le schéma 3.3. Physique et intégration sont calculées une première fois pour connaître un $d(q_{k+1})^{free}$, distance à la contrainte sans compensation. Elle permet de calculer la compensation Γ_c, par un LCP. Un deuxième étage de physique et d'intégration est alors réalisé pour prendre en compte les efforts de contrainte. Cette méthode ne pose pas de problème mathématique particulier. Nous ne l'utilisons pas dans notre architecture.

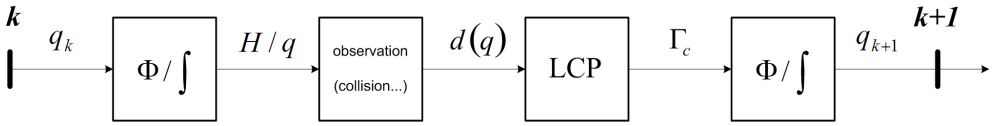

Fig. 3.3 – Schéma d'intégration à deux étages de *Physique et d'intégration* , notés $\Phi / \int$.

– une deuxième approche nous est proposée dans [Mer05] et met en œuvre une prédiction $\hat{d}(q_{k+1})$ de la distance à la contrainte, illustration 3.4. Cette méthode, moins exacte que la précédente, est pourtant plus adaptée à notre contexte temps-réel, notamment parce qu'elle permet de se passer d'un étage complet de physique et d'intégration. C'est celle qui a été retenue et nous la décrivons brièvement ici.

A chaque pas, nous observons, au pas courant k, les distances aux contraintes $d(q_k)$ (ex : détection de collisions pour le contact). Notons que ces distances ne sont pas toutes nulles, car d'une part une contrainte satisfaite est strictement positive ou nulle et d'autre part l'algorithme mis en œuvre peut laisser des contraintes légèrement violées (du fait de l'approximation décrite en (3.6)).

En l'absence de contraintes, le mouvement libre $\dot{q}^{free}$ serait calculé et donnerait l'évolution suivante

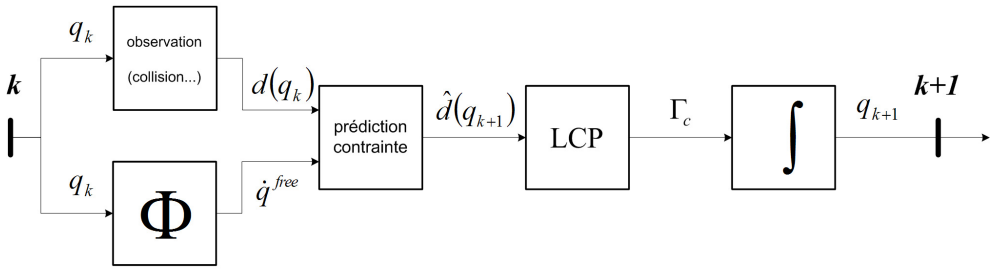

FIG. 3.4 – Schéma d'intégration à prédiction du modéle gémétrique de contrainte $\hat{d}(q_{k+1})$.

du système[1] :

$$q_{k+1} = q_k + \delta t \dot{q}^{free}, \tag{3.3}$$

or l'équation de comportement intégrant les forces de contrainte est donnée par :

$$b(q)\dot{q} = \Gamma + \Gamma_c, \tag{3.4}$$

on ajoute un terme correctif $\dot{q}_c$ à (3.3), pour prendre en compte les contraintes :

$$\begin{cases} q_{k+1} = q_k + \delta t \dot{q} \\ \dot{q} = \dot{q}^{free} + \dot{q}_c \\ \dot{q}_c = b^{-1}\Gamma_c. \end{cases} \tag{3.5}$$

Γ_c sera engendré de manière à respecter les contraintes $d(q_{k+1})$. Une approximation de $d(q_{k+1})$ est donnée par la linéarisation de $d(q_{k+1})$ autour de q_k, formulée grâce à J_c, jacobienne de la contrainte :

$$\hat{d}(q_{k+1}) = d(q_k) + \delta t J_c(q_k)\dot{q} \tag{3.6}$$

$$\boxed{\hat{d}(q_{k+1}) = d(q_k) + \delta t J_c(q_k)\dot{q}^{free} + \delta t J_c(q_k)b^{-1}(q_k)\Gamma_c,} \tag{3.7}$$

où $d(q_k)$ est observé. La relation de complémentarité associée est : $0 \leq \hat{d}(q_{k+1}) \perp \Gamma_c \geq 0$. Le choix de l'intégrateur et la résolution de la contrainte sont liés puisque l'expression de la contrainte linéarisée dépend du schéma d'intégration choisi. Notons que l'expression de J_c dépend de la contrainte à résoudre, et sera donnée dans les sections suivantes pour les contraintes qui nous intéressent ici.

Notons que nous exprimerons le LCP non pas sur Γ_c, mais sur W_c, avec $\Gamma_c = J_c^t W_c$, de telle manière que la matrice $J_c(q_k)b^{-1}(q_k)J_c(q_k)^t$ du LCP soit symétrique positive. Dans les cas où la matrice du LCP est copositive, le LCP possède au moins une solution et l'algorithme de Lemke la trouve [CPR92]. Nous sommes donc assurés de toujours obtenir une solution pour le contact.

Nous parvenons à la configuration finale q_{k+1} grâce à (3.5).

Les entrées du solveur de contraintes sont constituée de $d(q_k)$, distance à la contrainte à l'instant k et de $J_c(q_k)$, sa matrice jacobienne. L'algorithme suivant donne le mode opératoire du solveur de contraintes unilatérales.

[1] Notons que l'intégration donnée ici est écrite sur un espace vectoriel, dans un souci de simplification, mais son extension aux groupes de Lie est directe.

Algorithme 4 : Solveur de contraintes unilatérales et intégration

Données :

$d(q_k)$	:	distance à la contrainte mesurée au pas courant k
$\dot{q}^{free}$	:	vitesse en l'absence de contraintes (calculée par l'étage de physique)
$J_c(q_k)$	:	jacobienne de la contrainte
b	:	matrice des frottements visqueux apparents

Résultat : q_{k+1} : position au pas suivant

répéter

Observation de $d(q_k)$
Physique (voir 2.2) $\rightarrow \dot{q}^{free}$

$M = \delta t J_c(q_k) b^{-1}(q_k)$, matrice du LCP
$\rho = d(q_k) + \delta t J_c(q_k)\dot{q}^{free}$, vecteur du LCP
$\Gamma_c = Lemke(M,\ \rho)$
$\dot{q}_c = b^{-1}\Gamma_c$
$\dot{q} = \dot{q}^{free} + \dot{q}_c$
$q_{k+1} = q_k + \delta t \dot{q}$

jusqu'à *ce que l'utilisateur arrête la simulation*;

Rem n°1 : pour écrire cet algorithme, nous avons envisagé le cas d'une intégration sur des espaces vectoriels. Pour l'intégration sur groupes de Lie, il suffira de changer $q_{k+1} = q_k + \delta t \dot{q}$ par une des formules vues en Annexe C. Cet algorithme est à mettre en œuvre à chaque pas.

Rem n°2 : nous noterons que sur le schéma de description de l'architecture donné figure 1.31, une distinction très nette est faite entre le bloc d'intégration et le solveur de contraintes unilatérales, alors que dans le schéma 3.4, ces deux fonctionnalités sont très imbriquées. En toute rigueur, le LCP des contraintes se fait même avant l'intégration, comme indiqué dans l'algorithme 4.

Récapitulatif du choix du modèle

Choix de l'architecture du solveur de contraintes unilatérales

Avantages - inconvénients :

\+ la résolution d'une contrainte unilatérale ne nécessite que la connaissance de sa jacobienne
\+ n'implique le calcul que d'un étage de *physique* (Φ), et d'*intégration* ($\int$)
− la linéarisation implique une perte de précision (elle est cependant vraiment très faible)
− la jacobienne de la contrainte peut être ardue à calculer

3.2 Contact et limites articulaires

3.2.1 Formulation du contact

Spécialisons $d(q)$, comme étant la distance entre objets à un contact (on parle de distances minimales locales [JW04], elles sont observées par un algorithme de détection de collisions qui dépasse le cadre de notre étude) et intéressons-nous à la cinématique du contact [Mon89], [MLSS94], [JC01]. Au niveau de chaque contact i des m_c contacts totaux, la contrainte est exprimée par :

$$d_i(q) \geq 0, \forall i \in [1; m_c], \tag{3.8}$$

notons que cette contrainte exprime les conditions de contact dites de Signorini [Sig33] : $0 \leq d(q) \perp \Gamma_c \geq 0$. Afin d'exprimer la jacobienne associée à $d_i(q)$, nous posons c_{a_i} et c_{b_i}, les points du contact i sur les solides a et b [RK97] (obtenus grâce à un algorithme de détection de collision, dont le détail dépasse le cadre de notre étude), comme indiqué sur la figure 3.5.

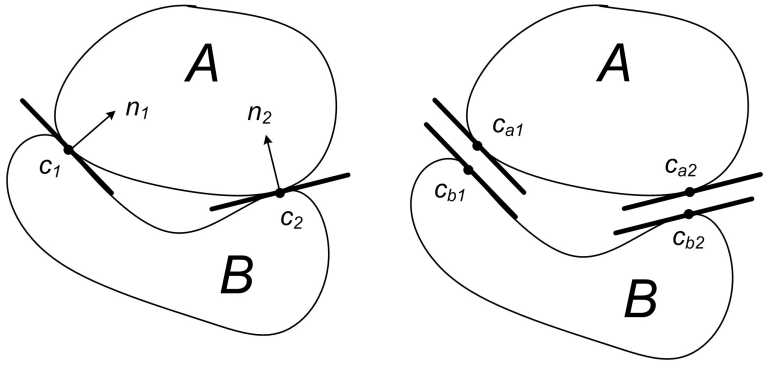

FIG. 3.5 – Contact de deux solides, d'après [RK97].

La vitesse v_{a_i}, de c_{a_i} est obtenue par :

$$v_{a_i} = v_a + \omega_a \wedge p_{a_i}, \tag{3.9}$$

avec v_a et ω_a, les vitesses linéaire et rotationnelle du repère associé au solide a et p_{a_i} le bras de levier de c_{a_i}. Cette relation peut être reformulée pour faire intervenir la jacobienne J_a de v_a :

$$v_{ai} = \begin{bmatrix} -[p_{a_i}] & I \end{bmatrix} V_a, \quad \text{avec } V_a = \begin{bmatrix} \omega_a \\ v_a \end{bmatrix} \tag{3.10}$$

$$= \begin{bmatrix} -[p_{a_i}] & I \end{bmatrix} J_a \dot{q} \tag{3.11}$$

où $[p_{a_i}]$ est le passage à l'expression par matrice antisymétrique de p_{a_i}. Soit :

$$p_{a_i} = \begin{bmatrix} p_{a_{ix}} \\ p_{a_{iy}} \\ p_{a_{iz}} \end{bmatrix}, \quad \text{alors } [p_{a_i}] = \begin{bmatrix} 0 & -p_{a_{iz}} & p_{a_{iy}} \\ p_{a_{iz}} & 0 & -p_{a_{ix}} \\ -p_{a_{iy}} & p_{a_{ix}} & 0 \end{bmatrix}. \tag{3.12}$$

La vitesse v_i (ou $\dot{d}_i(q)$) du contact i, projetée sur sa normale n_i est donnée par :

$$v_i = n_i^t (v_{a_i} - v_{b_i}) \tag{3.13}$$

$$= n_i^t \Big(\begin{bmatrix} -[p_{a_i}] & I \end{bmatrix} J_a - \begin{bmatrix} -[p_{b_i}] & I \end{bmatrix} J_b \Big) \dot{q} \tag{3.14}$$

$$\boxed{v_i = J_{c_i} \dot{q},} \tag{3.15}$$

où J_{c_i} est la jacobienne du contact i. Cette jacobienne est calculée pour chaque contact et donnée en entrée du solveur de contraintes unilatérales. Ce dernier donnant pour résultat les forces de contact, comme visualisé figure 3.6.

Le passage au contact frottant est problématique dans le cas de la mécanique du solide. Le modèle de Coulomb, largement adopté, présente une non-linéarité, résolue la plupart du temps en linéarisant la contrainte [APS99]. La matrice M du LCP qui en résulte est alors mal conditionnée, car elle n'est pas copositive [CPR92]. Pour obtenir une matrice M copositive, nous pouvons la régulariser au prix de la perte du comportement physique (ce qui correspond à une compliance au niveau du contact)[2]. Les

[2]Notons que *la nature n'a aucun problème à résoudre ses contacts*, puisque tous les objets y sont compliants ou déformables : la notion de solide n'est bien sûr qu'une modélisation, une approximation du monde réel...

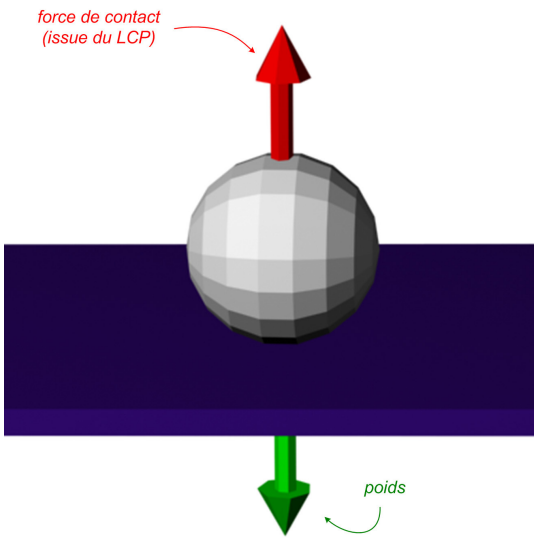

Fig. 3.6 – Force générée par le solveur de contraintes, pour le contact entre une balle et une table. Elle compense naturellement le poids.

méthodes impulsionnelles (nous avons vu qu'elles n'existent pas en quasi-statique), ou l'introduction de déformations visco-élastiques permettent d'éviter ce problème.

Le glissement des pieds sur le sol, problème difficile à résoudre, retire de son réalisme à une simulation. Le frottement permettrait d'y apporter une réponse - partielle du moins. Pour remédier au problème et également asservir les pieds quand c'est nécessaire, nous plaçons des couplages sur les pieds.

3.2.2 Formulation des limites articulaires

Cette contrainte articulaire doit être exprimée pour chaque type de liaison. Nous utilisons principalement des liaisons de type pivot, mais proposons également une formulation pour les rotules. Il suffit encore une fois de trouver une observation de la contrainte, ainsi que sa jacobienne.

Liaisons pivot

Dans le cas d'une liaison pivot, les limites articulaires sont exprimées par $q_{i_{min}} \leq q_i \leq q_{i_{max}}$, ce qui donne une double contrainte :

$$\begin{bmatrix} q_{i_{max}} - q_i \\ q_i - q_{i_{min}} \end{bmatrix} \geq 0 \tag{3.16}$$

dont la jacobienne est très simple :

$$J_c = \begin{bmatrix} 0_1 & \dots & -1_i & \dots & O_m \\ 0_1 & \dots & 1_i & \dots & O_m \end{bmatrix} \tag{3.17}$$

Le cas des liaisons rotules, testé mais non introduit dans le système, est présenté en annexe D.

3.3 Humains virtuels en équilibre

Sur l'architecture du système donnée schéma 1.32, cette fonctionnalité correspond au bloc donné illustration 3.7.

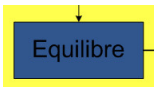

FIG. 3.7 – Bloc du schéma d'architecture 1.32, correspondant à la fonctionnalité décrite dans cette partie.

Pour l'équilibre, une démarche similaire aux précédentes est suivie :

- *construction d'une distance*
- *calcul de la jacobienne.*

Le problème étant redondant, nous allons également appliquer un critère d'optimisation, nous ne nous limitons donc pas à la seule gestion de l'équilibre. Nous verrons d'ailleurs que l'expression du problème de l'équilibre permet même de contrôler des tâches externes.

3.3.1 Analyse de l'existant

Les caractérisations les plus simples de l'équilibre sont basées sur l'appartenance d'un point d'intérêt particulier à une zone, à définir, de l'espace opérationnel. Nous retenons principalement, comme points caractéristiques, la projection du centre de gravité et le Zero Moment Point (ZMP).

Projection du centre de gravité

Imaginons un système polyarticulé statique, soumis à la gravité. Chaque solide est de masse m_k et de position x_k. Il est en contact avec un environnement donné aux points de position p_i. Ces contacts génèrent les forces $f_{c_i}\boldsymbol{n}$, avec $\boldsymbol{n}$ normale unitaire au contact. Si tous les contacts sont sur un même plan horizontal, alors les projections horizontales et verticales de la statique sont découplées et nous avons [Wie02] :

$$\begin{cases} -\sum m_k g\boldsymbol{n} + \sum f_{c_i}\boldsymbol{n} = 0 \\ -\sum x_k \wedge m_k g\boldsymbol{n} + \sum p_i \wedge f_{c_i}\boldsymbol{n} = 0. \end{cases} \tag{3.18}$$

Posons alors $m = \sum m_k$, la masse totale du système et $x_{com} = \frac{\sum m_k x_k}{m}$, nous pouvons alors écrire :

$$x_{com} \wedge \boldsymbol{n} = \frac{\sum f_{c_i} p_i \wedge \boldsymbol{n}}{\sum f_{c_i}}. \tag{3.19}$$

En prenant maintenant en compte le fait qu'un contact est unilatéral, c'est-à-dire $\forall i, f_{c_i} \geq 0$, [Wie02] énonce le théorème suivant :

Théorème 3.3.1 *Un système marcheur peut rester statique (c'est-à-dire équilibré) ssi la projection verticale de son centre de masse est située à l'intérieur de la coque convexe de ses points de contact (appelé* polygone de support*).*

La figure 3.8 illustre ce cas.

Rapport-gratuit.com
LE NUMERO 1 MONDIAL DU MÉMOIRES

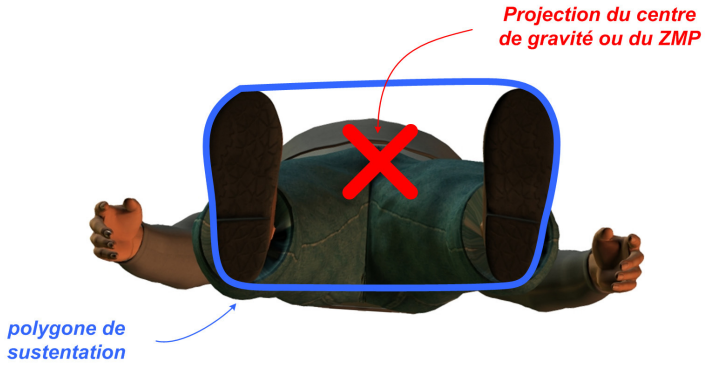

Fig. 3.8 – Humain virtuel vu de dessous, polygone de sustentation et projection du centre de gravité, ou du ZMP.

Cette caractérisation géométrique de l'équilibre est efficace uniquement dans les cas statiques, où les contacts sont coplanaires.

Zero Moment Point (ZMP)

Nous nous plaçons maintenant dans les mêmes conditions que précédemment, mais dans un cas dynamique [Wie02], [VB04]. Posons f et ω, les efforts inertiels et de Coriolis, respectivement translationnels et rotationnels. Si les contacts sont planaires :

$$\begin{cases} -\sum m_k g\boldsymbol{n} + \sum f_{c_i}\boldsymbol{n} = (f \cdot \boldsymbol{n})\boldsymbol{n} \\ -\sum x_k \wedge m_k g\boldsymbol{n} + \sum p_i \wedge f_{c_i}\boldsymbol{n} = \boldsymbol{n} \wedge \omega \wedge \boldsymbol{n}, \end{cases} \tag{3.20}$$

avec $\boldsymbol{n} \wedge \omega \wedge \boldsymbol{n}$, composante horizontale de ω. De la même manière qu'en (3.19), nous pouvons écrire :

$$\frac{mgx_{com} \wedge \boldsymbol{n} + \boldsymbol{n} \wedge \omega \wedge \boldsymbol{n}}{mg + f \cdot \boldsymbol{n}} = \frac{f_{c_i} p_i \wedge \boldsymbol{n}}{\sum f_{c_i}}, \tag{3.21}$$

et donc de la même manière qu'en 3.3.1 :

Théorème 3.3.2 *Un système marcheur est équilibré d'un point de vue dynamique, si la projection verticale du point défini par* $\frac{mgx_{com}+\boldsymbol{n}\wedge\omega}{mg+f\cdot\boldsymbol{n}}$*, appelé Zero Moment Point (ZMP), est située à l'intérieur du polygone de sustentation (théorème* (3.3.1)*). On peut montrer que la projection de ce point sur le plan de contact est le point où le moment horizontal des forces gravitationnelles et des efforts dynamiques* f *et* ω *s'annulent.*

Ces deux types de caractérisations, des plus utilisées, ne permettent pas de prendre en compte des situations aux contacts non-coplanaires. Ces cas sont pourtant nominaux lorsque l'on adresse la problématique du mannequin en interaction avec son environnement.

Contrôle de l'équilibre spécifique aux mannequins virtuels

Différents chercheurs ont apporté leur contribution à l'intégration de la gestion de l'équilibre dans le contrôle d'humains virtuels.

Laszlo, Van de Panne et Fiume [LVdPF96] ont introduit le *contrôle de cycle limite*, qui permet de compenser de petites perturbations dans des mouvements cycliques (vus dans des espaces d'état réduits). Ils utilisent une variable, appelée *up-vector*, à réguler par l'intermédiaire de variables de contrôle, dont le choix s'arrête sur les angles articulaires du bassin. Cette approche est intéressante pour des mouvements cycliques comme la marche, mais nous voulons pouvoir réaliser des mouvements quelconques.

Hodgins en Wooten [HW98] proposent de contrôler les variables articulaires des chevilles et du bassin de manière à assurer l'équilibre statique. Leur approche est là encore spécifique à un type de mouvement (le cheval d'arceau dans leur exemple).

Faloutsos, Van de Panne et Terzopoulos [FVdPT98] prennent en compte les perturbations de l'équilibre grâce à la notion de polygone de support établie ci-avant. Ils corrigent la position du mannequin, en contrôlant uniquement la raideur de la cheville. Leur approche se comporte bien dans les situations de perte d'équilibre, mais leur contrôle est plutôt restrictif.

En effet, lorsqu'un être humain perd son équilibre, toutes ses articulations sont sollicitées pour retrouver une situation stable. Il ne se limite pas au seul contrôle de la cheville ou du bassins, comme dans les méthodes ci-avant. A cette fin, Liu et Popovic [LP02] proposent d'ajouter une distance à la rupture d'équilibre dans l'objectif de leur optimisation espace-temps, ceci assurant une réponse aux perturbations distribuée sur toutes les articulations. Leur méthode permet par ailleurs d'obtenir des résultats intéressants à partir d'esquisses de mouvements. Elle souffre cependant des problèmes communs à toutes les méthodes espace-temps, déjà soulignés en 1.2.1. Fang et Pollard [FP03], mettent quant à eux en œuvre un *filtre physique* (algorithme qui permet de rendre physiquement correct un mouvement qui ne l'était pas). Il n'autorise pas le temps-réel - bien qu'il soit plus rapide que les tentatives précédentes de ce type.

Approches par fonctions de Lyapunov

Une autre approche est mise en œuvre dans [Wie00], qui propose d'utiliser des mouvements de référence calculés au préalable et connus pour être équilibrés. Ces échantillons mis bout à bout permettent de réaliser des trajectoires plus complexes, qui dans un environnement parfaitement maîtrisé seraient équilibrées (cette méthode est à rapprocher des bibliothèques de mouvements décrites en 1.2.2).

Les mouvements obtenus peuvent être perturbés par des événements du monde environnant le robot. [Wie00] fait alors appel à la notion de *viabilité* d'un état $(q, \dot{q})$ introduite dans [Aub91].

Considérons l'ensemble Ω des configurations d'un robot où nous estimons qu'il a chuté :

Définition 3.3.1 *Un état $(q, \dot{q})$ est viable si, à partir de cet état, le robot est en mesure de réaliser un mouvement $q(t)$ qui ne pénètre pas dans l'ensemble Ω.*

Les états du robot forment alors deux groupes distincts :
- les états viables à partir desquels le robot peut éviter de tomber, qui forment ce qui est appelé le *noyau de viabilité*
- et les états où la chute est inéluctable.

Le robot doit donc coordonner ses mouvements de manière à rester "suffisamment proche" d'un ensemble d'états qui lui ont été prescrits (des trajectoires de marche par exemple). [Wie00] nous propose de considérer que la configuration est suffisamment proche à un instant donné de la trajectoire de consigne, elle doit le rester par la suite. Ceci l'amène à considérer que l'ensemble d'états doit être *stable au sens de Lyapunov* (voir [Kha02] pour un définition de la stabilité au sens de Lyapunov).

La *stabilité asymptotique* [Kha02] nous permet d'aller plus loin, en exprimant le souhait que l'ensemble D soit attractif, pour se rapprocher du mouvement de référence après la perturbation (voir [Wie00]).

Dans [Wie02], l'auteur construit une marge de stabilité au sens de Lyapunov qui apporte une vue dont il dit lui-même qu'elle est conservatrice. En effet l'ensemble des états qu'il considère comme viables est inclus dans l'ensemble des états réellement viables. Mais même si elle n'est appliquée qu'au cas des robots marcheurs, elle doit pouvoir être mise en œuvre dans des cas différents, comme ceux qui nous intéressent. Cette approche est surtout adaptée au cas des bibliothèques de mouvements, où un ensemble de mouvements de référence équilibrés est connu.

Protective step

Les cas de contrôle d'équilibre évoqués auparavant permettent uniquement de gérer les situations où

l'équilibre est peu perturbé. En effet, un humain réel soumis à de fortes perturbations a deux réactions envisageables : soit il tombe, soit il se déplace dans le sens de la perturbation en effectuant un *pas dit de protection*, qui lui permet de retrouver une situation plus stable. Les deux situations sont gérées par la plateforme DANCE® de Faloutsos [FVdPT98], [SFNTH05], grâce à une composition temporelle de contrôleurs spécialisés.

Arikan, Forsyth et O'Brien présentent dans [AFO05] une méthode permettant à un humain virtuel de réagir à des sollicitations extérieures. Ils construisent une bibliothèque de mouvements d'humains poussés, dans laquelle il viennent piocher de manière raisonnée. A cette fin, ils mettent en œuvre un algorithme de décision auquel ils auront au préalable appris à bien se comporter. Leur méthode est temps-réel et donne des résultats très convaincants.

Vers le multi-contacts non-coplanaire

Parmi les approches analysées, les suivantes sont plus proches de nos préoccupations.

Takubo [TI05] et Harada [HHK+04], [HKKH03b], [HKKH03a], apportent des éléments de réponse à la problématique de l'équilibre dans le cas de contacts non coplanaires - comme c'est le cas dès que l'on interagit avec l'environnement. Ils caractérisent l'équilibre de manière géométrique, par le biais de variantes étendues du ZMP.

La première contribution [TI05], permet de pousser sur un environnement mobile connu (fauteuil roulant, chariot, caisse pesante...). La méthode donne des résultats intéressants dans les cas où le système étudié n'est en contact avec l'environnement qu'aux endroits où il pousse l'objet. Elle ne permet pas de prendre en compte des contacts non prévus initialement.

Fig. 3.9 – Un robot virtuel pousse sur un environnement connu, d'après [TI05].

La seconde approche [HHK+04] permet de prendre en compte les cas où le robot (un humanoïde) agrippe son environnement. Le contrôle est exprimé sous forme d'un problème de programmation non-linéaire et ramené à une forme linéaire pour sa résolution. Il n'est pas limité à un type donné d'interactions comme c'était le cas précédemment. L'approche est restreinte aux cas où le centre de gravité se déplace sur un plan horizontal : un mouvement simple comme ramasser un objet par terre, ne peut-être résolu par la méthode proposée.

3.3.2 Intégration de l'équilibre dans notre architecture

Nous pouvons distinguer trois familles de méthodes de résolution d'équilibre, qui interviennent dans des situations complémentaires :

- *les cas où l'équilibre est recherché en modifiant la configuration interne de l'humain virtuel. Dans ce cas on peut encore atteindre les consignes de nos repères de contrôle.*
- *les cas où un pas de protection est nécessaire (on doit alors s'écarter de nos consignes).*
- *les cas où l'équilibre est perdu.*

Notre étude est restreinte au premier cas. Nous introduisons en plusieurs étapes, un contrôleur d'équilibre qui nous permet d'apporter certains éléments de réponse, visant à dépasser les limitations évoquées ci-dessus.

3.3.2.1 Résolution de l'équilibre par LCP : une première approche

Nous présentons une première méthode de contrôle de l'équilibre dans les cas statiques, aux contacts coplanaires. Nous exprimons le problème sous forme de LCP et ce pour les raisons suivantes :

- il assure un respect strict des contraintes (donc un équilibre parfaitement respecté), dans les conditions pour lesquelles le contrôleur a été écrit
- cette méthode fournit une réponse répartie sur toutes les articulations, elle est donc plus naturelle
- écrite sous forme de LCP, la contrainte peut être résolue en même temps que le contact, ou les limites articulaires

Aparté - Principe du contrôleur d'équilibre

Nous construisons une distance d à la perte d'équilibre. L'équilibre sera caractérisé par le respect de la contrainte $d \geq 0$. Dans les cas où la contrainte est violée, des couples articulaires Γ_c destinés à retrouver une configuration équilibrée seront générés. Nous avons décrit en 3.1 un algorithme permettant de générer ces couples Γ_c, connaissant la jacobienne de d. Dès lors, il nous suffit de calculer cette jacobienne.

Dans les conditions d'un support plan, une caractérisation de l'équilibre est que la projection du centre de masse soit à l'intérieur du polygone de sustentation. Nous approximons ce polygone de sustentation par une ellipse englobant les deux pieds en contact avec le sol. Cela est fait sans perte de généralité, car nous pourrions exprimer tout polygone de sustentation par des contraintes linéaires. Le mannequin est donc considéré en équilibre lorsque la projection du centre de gravité se situe à l'intérieur de l'ellipse limite.

Nous construisons $d(q)$, la distance quadratique à la limite, de telle manière qu'elle soit positive ou nulle si la contrainte est respectée et négative sinon. Si cette contrainte est violée, des couples Γ_c de compensation sont générés, par l'intermédiaire de l'algorithme de résolution du LCP, pour rejoindre un état admissible $d(q) = 0$. Pour résoudre cette situation, le solveur requiert la connaissance de $d(q)$ et de sa matrice jacobienne J_c :

$$\boxed{d(q) = \delta^2 - \|\Pi\left(x_{com} - x_c\right)\|_Q^2,} \tag{3.22}$$

avec δ la distance quadratique pondérée maximale, Π une projection verticale, x_{com} la position du centre de masse, x_c le centre de l'ellipse Q et Q la métrique lui correspondant. La jacobienne J_c se calcule de la manière suivante :

$$J_c = \frac{\partial d(q)}{\partial q}, \tag{3.23}$$

d'après (3.22)

$$\partial d(q) = -\partial \|\Pi\left(x_{com} - x_c\right)\|_Q^2, \tag{3.24}$$

en considérant que le polygone de support ne varie pas, nous avons :

$$\partial d(q) = -(x_{com} - x_c)^t \Pi^t (Q + Q^t) \partial \big(\Pi (x_{com} - x_c)\big) \tag{3.25}$$
$$= -(x_{com} - x_c)^t \Pi^t (Q + Q^t) \Pi \partial x_{com} \tag{3.26}$$
$$= -(x_{com} - x_c)^t \Pi^t (Q + Q^t) \Pi J^r_{com} \partial q, \tag{3.27}$$

où J^r_{com} est la jacobienne réduite associée à la position du centre de gravité. Elle est réduite car elle ne fait pas intervenir de composante rotationnelle, uniquement la position 3D est considérée ici.

La position $x_{com(0)}$, du centre de masse, exprimée dans le repère de base d'indice 0 est donnée par :

$$x_{com(0)} = \frac{\sum m_i x_{com_i(0)}}{\sum m_i}, \tag{3.28}$$

avec $x_{com_i(0)}$ la position du centre de masse du solide i et m_i sa masse. Alors la vitesse $v_{com/0(0)}$ du centre de masse du système complet vue dans le repère 0 et exprimée dans ce même repère est :

$$v_{com/0(0)} = \frac{\partial x_{com(0)}}{\partial t} \tag{3.29}$$
$$= \frac{\sum m_i \frac{\partial x_{com_i(0)}}{\partial t}}{\sum m_i} \tag{3.30}$$
$$= \frac{\sum m_i v_{com_i/0(0)}}{\sum m_i} \tag{3.31}$$

ainsi $J^r_{com/0(0)}$ (simplement notée J^r_{com}, dans (3.27)), la matrice jacobienne de $v_{com/0(0)}$, est donnée par :

$$J^r_{com/0(0)} = \frac{\sum m_i J^r_{com_i/0(0)}}{\sum m_i}, \tag{3.32}$$

avec $J^r_{com_i/0(0)}$ jacobienne réduite associée à la vitesse du centre de gravité du solide i. Posons $J_{A\in i/j(k)}$ la matrice jacobienne associée à la vitesse d'un point A fixe dans le repère i, dans son mouvement par rapport au repère j et exprimée dans la base k. La relation entre $J_{A\in i/j(k)}$ et $J^r_{A\in i/j(k)}$, est donnée par :

$$J^r_{A\in i/j(k)} = \begin{bmatrix} 0_{3*3} & I_3 \end{bmatrix} J_{A\in i/j(k)} \tag{3.33}$$
$$= S J_{A\in i/j(k)}, \tag{3.34}$$

sachant cela, exprimons $J^r_{com/0(0)}$, grâce aux matrices jacobiennes $J_{com_i/0(0)}$ des centres de masse de chaque solide du système articulé, qui sont connues. (3.32) nous donne :

$$J^r_{com/0(0)} = \frac{\sum m_i J^r_{com_i/0(0)}}{\sum m_i} \tag{3.35}$$
$$= \frac{\sum m_i S J_{com_i/0(0)}}{\sum m_i}. \tag{3.36}$$

D'où

$$\boxed{\frac{\partial d(q)}{\partial q} = -(x_{com} - x_c)^t \Pi^t (Q + Q^t) \Pi \frac{\sum m_i S J_{com_i/0(0)}}{\sum m_i}.} \tag{3.37}$$

Ce contrôleur d'équilibre est mis en œuvre figure 3.10 et a été publié dans [RMM+05b] et [RMM+05a].

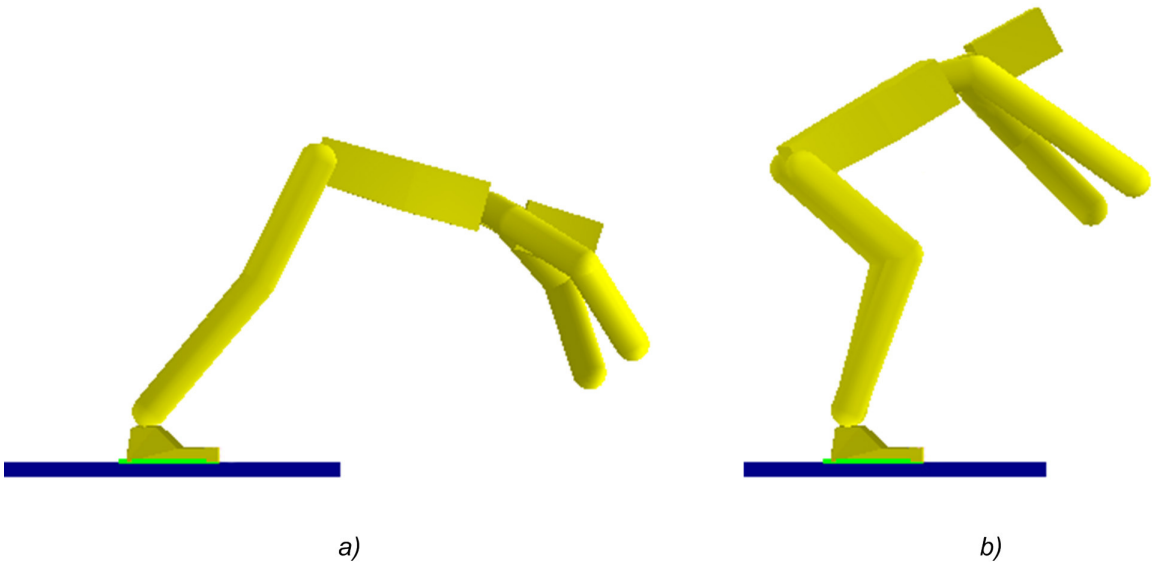

FIG. 3.10 – a) Sans contrôleur d'équilibre l'humain virtuel peut atteindre des configurations dans lesquelles un humain réel chuterait. b) Avec contrôleur d'équilibre, pour la même consigne, ces configurations sont évitées.

3.3.2.2 Equilibre multi-contacts non-coplanaires

Aparté - Principe du contrôleur d'équilibre

Nous construisons ici un contrôleur d'équilibre, en le posant sous la forme d'un problème d'optimisation. Les inconnues sont la position du centre de gravité du système et les forces de contact. Le principe de la méthode est fondé sur la constatation qu'un système est équilibré s'il respecte l'équation d'équilibre dynamique, et que ses contacts avec l'environnement sont non-glissants. Nous nous restreindrons au cas statique.

- le non-glissement des contacts sera exprimé sous forme de contraintes unilatérales
- le respect de l'équation d'équilibre statique sera exprimé sous forme de contrainte bilatérale
- le critère optimisé permettra de minimiser les efforts en jeu, ainsi que d'assurer une marge de stabilité sur le glissement des contacts

Le problème d'optimisation consiste donc à trouver la position du centre de gravité et les forces de contact telles que :

$$\begin{cases} min(\text{efforts}) \text{ et } max(\text{marges de stabilité}) \\ \quad \text{sachant} \begin{cases} \text{respect de l'équilibre statique} \\ \text{respect du non-glissement des contacts,} \end{cases} \end{cases} \tag{3.38}$$

Des raffinements seront ensuite apportés à ce premier contrôleur.

Premier contrôleur

Les approches précédentes [HHK+04], [TI05] caractérisent l'équilibre de manière géométrique par des variantes du ZMP. Ainsi l'état d'un système possédant une trentaine de degrés de liberté est caractérisé par un point dans un espace 3D : ces *ZMP étendus* présentent donc une vue réduite du système. La prise compte de tous les degrés de liberté du système permettra d'enrichir l'ensemble des solutions.

L'équilibre du mannequin est à mettre en relation avec le problème de la stabilité de prise (apparaissant lors de la préhension manuelle d'un objet) [XLL03], [LL04].

Comme souligné dans [Wie02], pour être dans une situation dite *stable*, un humain (virtuel ou réel) doit avoir des contacts non-glissants et doit respecter l'équation d'équilibre dynamique.

Nous nous placerons dans la restriction au cas de l'équilibre statique. L'hypothèse est peu réductrice, si on considère les mouvements d'un technicien réalisant une opération de maintenance, ou d'un pilote aux commandes de son avion...

Trois types de forces agissent sur le système :

- le poids P
- les forces de contact W_{c_i}
- des éventuels efforts extérieurs W_{ext_k}, potentiellement perturbateurs

En suivant la démarche de [BLR05], la contrainte d'équilibre statique s'exprime donc :

$$\sum W_{ext_k} + P + \sum W_{c_i} = 0. \tag{3.39}$$

Posons 0 l'indice du repère dans lequel l'équilibre sera calculé (à cette fin, nous pourrons choisir le repère de base par exemple...), alors (3.39) devient :

$$\sum Ad^*_{{}^0H_{ext_k}} W_{ext_k} + Ad^*_{{}^0H_{com}} P + \sum Ad^*_{{}^0H_{c_i}} W_{c_i} = 0, \tag{3.40}$$

avec

- ${}^0H_{ext_k}$, matrice de passage du repère 0 au repère ext_k (repère dans lequel est exprimé W_{ext_k})
- ${}^0H_{com}$, matrice de passage de 0 à com, repère associé au centre de gravité
- ${}^0H_{c_i}$, matrice de passage de 0 à c_i, repère associé au contact i
- Ad^*_H, matrice coadjointe de H.

Dans notre approche nous considérerons les contacts ponctuels avec frottement [MLSS94]. Dans ce cas, les efforts de contact W_{c_i} se réduisent à la résultante f_{c_i}, exprimée dans le repère associé au contact.

Soit R et p, les parties rotationnelles et translationnelles d'une matrice

$$H = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix}. \tag{3.41}$$

alors :

$$Ad^*_H = \begin{bmatrix} R & [p\,]R \\ 0 & R \end{bmatrix}, \tag{3.42}$$

et

$$Ad^*_H W_{c_i} = Ad^*_H \begin{bmatrix} 0 \\ f_{c_i} \end{bmatrix} \tag{3.43}$$

$$= \begin{bmatrix} R & [p\,]R \\ 0 & R \end{bmatrix} \begin{bmatrix} 0 \\ f_{c_i} \end{bmatrix} \tag{3.44}$$

$$= \begin{bmatrix} [p\,]R \\ R \end{bmatrix} f_{c_i}. \tag{3.45}$$

Insérons maintenant (3.45) dans (3.40), après quelques manipulations algébriques nous obtenons :

$$\sum \begin{bmatrix} [p_{i(0)}\,]^0R_i \\ {}^0R_i \end{bmatrix} f_{c_i} = -\Big(\sum Ad^*_{{}^0H_{ext_k}} W_{ext_k} + Ad^*_{{}^0H_{com}} P \Big). \tag{3.46}$$

Nous avons aussi :

$$P = \begin{bmatrix} 0 \\ m\vec{g} \end{bmatrix}, \text{ avec } \vec{g} \text{ la gravité,} \tag{3.47}$$

et pour simplifier les équations (mais sans perte de généralité), nous prenons

$$ {}^0H_{com} = \begin{bmatrix} I_3 & x_{com} \\ 0 & 1 \end{bmatrix}, \tag{3.48} $$

avec x_{com}, position du centre de gravité. Alors

$$ Ad^*_{0_{H_{com}}} P = \begin{bmatrix} I_3 & [x_{com}\,] \\ 0_{3*3} & I_3 \end{bmatrix} \begin{bmatrix} 0 \\ m\vec{g} \end{bmatrix} \tag{3.49} $$

$$ = \begin{bmatrix} 0 & -mg & 0 \\ mg & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ -mg \end{bmatrix} \tag{3.50} $$

Intégrons (3.50), dans (3.46), en ayant au préalable posé m_c, le nombre de contacts :

$$ \begin{bmatrix} [p_{1(0)}\,]^0R_1 & \dots & [p_{m_c(0)}\,]^0R_{m_c} \\ {}^0R_1 & \dots & {}^0R_{m_c} \end{bmatrix} \begin{bmatrix} f_{c_1} \\ \vdots \\ f_{c_{m_c}} \end{bmatrix} + \begin{bmatrix} 0 & -mg & 0 \\ mg & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ mg \end{bmatrix} - \sum Ad^*_{0_{H_{ext_k}}} W_{ext_k}, \tag{3.51} $$

$$ \boxed{ \begin{bmatrix} & & & 0 & -mg & 0 \\ [p_{1(0)}\,]^0R_1 & \dots & [p_{m_c(0)}\,]^0R_{m_c} & mg & 0 & 0 \\ & & & 0 & 0 & 0 \\ {}^0R_1 & \dots & {}^0R_{m_c} & 0 & 0 & 0 \\ & & & 0 & 0 & 0 \\ & & & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} f_{c_1} \\ \vdots \\ f_{c_{m_c}} \\ x_c \\ y_c \\ z_c \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ mg \end{bmatrix} - \sum Ad^*_{0_{H_{ext_k}}} W_{ext_k}, } \tag{3.52} $$

qui est l'équation statique qu'il faut respecter, c'est-à-dire à intégrer comme contrainte bilatérale dans notre algorithme.

Pour la contrainte d'adhérence, nous nous appuierons sur le modèle de contact de type Coulomb qui relie les forces normales f_n et tangentielles f_t. Elle exprime une relation de complémentarité (voir figure 3.11) entre f_t et v_t, vitesse tangentielle au contact :

$$ \|f_t\| \leq \mu f_n, \tag{3.53} $$

que nous pouvons décomposer en

$$ \|f_t\| < \mu f_n \implies v_t = 0 \quad \text{(adhérence)} \tag{3.54} $$

$$ f_t = -\mu f_n \frac{v_t}{\|v_t\|} \quad \text{(glissement)}, \tag{3.55} $$

avec μ coefficient de frottement dépendant principalement des matériaux en contact. Le contact est adhérent si la force appartient à l'intérieur du cône. Si elle est sur son bord il y a glissement.

La contrainte (3.54) est non-linéaire. Nous allons *linéariser le contact*, comme dans [APS99], de manière à pouvoir incorporer ces contraintes à l'algorithme de résolution des problèmes de complémentarité

linéaires. Le cône est discrétisé, comme indiqué sur la figure. Pour un contact i, l'approximation $\hat{f}_{c_i}$, de f_{c_i}, sur un cône à m_a arêtes a_i s'écrira donc :

$$\hat{f}_{c_i} = f_{c_{i,1}} a_{i,1} + \ldots + f_{c_{i,n}} a_{i,m_a} \tag{3.56}$$

$$= \begin{bmatrix} a_{i,1} & \ldots & a_{i,m_a} \end{bmatrix} \begin{bmatrix} f_{c_{i,1}} \\ \vdots \\ f_{c_{i,m_a}} \end{bmatrix} \tag{3.57}$$

$$\boxed{\hat{f}_{c_i} = a_i f_{c_i}^{dis},} \tag{3.58}$$

l'indice *dis* étant employé pour abréviation du mot *discret*. La linéarisation se traduit par le cône *facetisé*, figure 3.11.

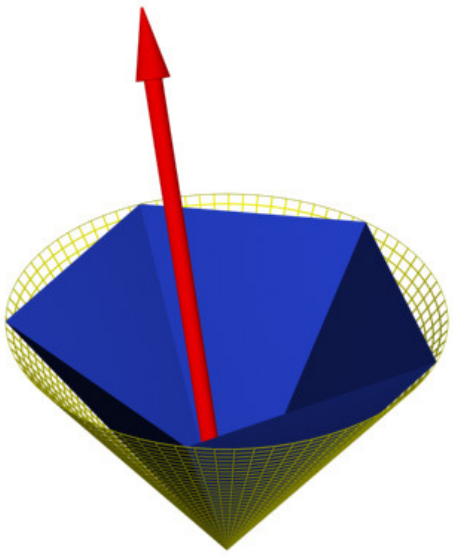

Fig. 3.11 – Selon le modèle de Coulomb, la force de frottement doit se trouver à l'intérieur du cône (jaune), ou à défaut de sa version linéarisée (bleue), pour que le contact soit considéré adhérant.

En remplaçant les f_{c_i} par leur approximation (3.58), dans (3.52), l'approximation suivante est obtenue :

$$\begin{bmatrix} & & & 0 & -mg & 0 \\ [p_{1(0)}\,]^0R_1a_1 & \ldots & [p_{m_c(0)}\,]^0R_{m_c}a_{m_c} & mg & 0 & 0 \\ & & & 0 & 0 & 0 \\ {}^0R_1a_1 & \ldots & {}^0R_{m_c}a_{m_c} & 0 & 0 & 0 \\ & & & 0 & 0 & 0 \\ & & & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} f_{c_1}^{dis} \\ \vdots \\ f_{c_{m_c}}^{dis} \\ x_c \\ y_c \\ z_c \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ mg \end{bmatrix} - \sum Ad^*_{{}^0H_{ext_k}} W_{ext_k}, \tag{3.59}$$

que nous noterons

$$GX = W, \tag{3.60}$$

avec

- G : la matrice caractérisant les contacts et le poids du membre de gauche de (3.59)
- X : les inconnues (efforts de contact et position du centre de masse)
- W : le membre de droite de (3.59).

Posons f_c^{dis}, I_f et I_x tels que :

$$X = \begin{bmatrix} f_c^{dis} \\ x_{com} \end{bmatrix} = \begin{bmatrix} I_{m_c m_a * m_c m_a} & 0 \\ 0 & I_{3*3} \end{bmatrix} X = \begin{bmatrix} I_f \\ I_x \end{bmatrix} X \tag{3.61}$$

Ainsi linéarisée, la contrainte de non-glissement s'exprime par $f_c^{dis} \geq 0$, donc de la manière suivante :

$$I_f X \geq 0. \tag{3.62}$$

Le problème est donc défini par le système d'équations/inéquations (3.60) et (3.61) :

$$\begin{cases} GX & = W \\ I_f X & \geq 0. \end{cases} \tag{3.63}$$

Dans le cas hyper-redondant du mannequin, le problème est généralement sous-contraint. Pour résoudre la redondance, nous lui adjoignons un critère quadratique à optimiser, par souci de simplicité d'une part et parce que les optimisations quadratiques sous contrainte unilatérale peuvent-être exprimées sous forme de LCP d'autre part. Nous avons choisi ici de minimiser les efforts de contact, le déplacement du centre de masse, ainsi que les efforts tangentiels de contact et pour ce faire, il nous faut les exprimer en fonction de X.

Pour un contact i, de normale n_i et de force tangentielle $f_{t_{c_i}}$ nous avons :

$$f_{t_{c_i}} = (I - n_i n_i^t) f_{c_i} \tag{3.64}$$

aussi pour l'ensemble des contacts, nous pouvons écrire

$$f_{t_c} = (I - NN^t) f_c, \tag{3.65}$$

avec

$$f_c = \begin{bmatrix} f_{c_1} \\ \vdots \\ f_{c_{m_c}} \end{bmatrix} \qquad f_{t_c} = \begin{bmatrix} f_{t_{c_1}} \\ \vdots \\ f_{t_{c_{m_c}}} \end{bmatrix} \qquad N = \begin{bmatrix} n_1 & 0 & 0_{3*1} \\ 0_{3*1} & \ddots & 0_{3*1} \\ 0_{3*1} & 0 & n_{m_c}. \end{bmatrix}. \tag{3.66}$$

En remplaçant les f_{c_i} par leur approximation (3.58), dans (3.65), l'approximation suivante est obtenue :

$$f_{t_c} = (I - NN^t) AX, \tag{3.67}$$

avec

$$A = \begin{bmatrix} a_1 & 0 & 0_{3*m_a} & 0_{3*3} \\ 0 & \ddots & 0 & 0 \\ 0_{3*m_a} & 0 & a_{m_c} & 0_{3*3} \end{bmatrix}. \tag{3.68}$$

Nous choisissons de minimiser les forces de contacts et de positionner le centre de gravité à une position x_{com_d} désirée, dans la mesure du possible. Afin d'assurer une marge de stabilité sur le glissement nous minimisons également les efforts tangentiels. Pour interagir avec l'environnement, en lui appliquant des efforts, nous introduisons les efforts de contact désirés $f_{c_d}^{dis}$ dans le critère d'optimisation (bien sûr discrétisés au préalable). Le problème d'optimisation s'écrit donc de la manière suivante :

$$\boxed{\begin{cases} \min\limits_X \dfrac{1}{2} \left(\left\| X - \begin{bmatrix} f_{c_d}^{dis} \\ x_{com_d} \end{bmatrix} \right\|_{Q_1}^2 + \alpha \left\| \left(I - NN^t \right) AX \right\|_{Q_2}^2 \right) \\ \quad \text{sachant } \begin{cases} GX = W \\ I_f X \geq 0, \end{cases} \end{cases}} \tag{3.69}$$

avec $\alpha \geq 0$ permettant de moduler l'importance relative des deux objectifs.

Il est possible d'exprimer un problème d'optimisation quadratique sous forme de LCP (Annexe A) et le LCP suivant est obtenu :

$$0 \leq I_f \Pi_G Q_A^{-1} I_f^t z + I_f \Pi_G Q_A^{-1} \begin{bmatrix} I \\ (I - NN^t)A \end{bmatrix}^t Q \begin{bmatrix} f_{c_d}^{dis} \\ x_{com_d} \\ 0 \end{bmatrix} + I_f Q_A^{-1} G^t (G Q_A^{-1} G^t)^{-1} W \perp z \geq 0, \quad (3.70)$$

avec

$$Q = \begin{bmatrix} Q_1 & 0 \\ 0 & \alpha Q_2 \end{bmatrix} \quad (3.71)$$

$$Q_A^{-1} = \left(\begin{bmatrix} I \\ (I - NN^t)A \end{bmatrix}^t Q \begin{bmatrix} I \\ (I - NN^t)A \end{bmatrix} \right)^{-1}$$

$$\Pi_G = I - Q_A^{-1} G^t \left(G Q_A^{-1} G^t \right)^{-1} G.$$

Ces paramètres d'entrée, fournis à un algorithme de résolution de LCP, nous permettent d'obtenir X. Les forces de contact et position du centre de gravité composant X assureront l'équilibre du mannequin au cours de la simulation, une fois appliquées au système.

Notons $M = I_f \Pi_G Q_A^{-1} I_f^t$, nous avons $\Pi_G Q_A^{-1}$ symétrique positive car Π_G est pondérée par Q_A^{-1}, donc M est aussi symétrique positive. C'est une condition suffisante pour affirmer que la méthode de Lemke utilisée pour résoudre le LCP trouvera une solution au problème s'il en existe une et échouera s'il n'y en a pas [CPR92].

Le bon comportement du système peut être vérifié sur les figures 3.12, 3.13 et 3.14. On y constate un comportement naturel et attendu de l'humain virtuel.

ATTENTION : d'après l'annexe A, la transformation d'une optimisation quadratique en LCP nécessite que deux conditions soient remplies :

- la contrainte $GX = W$ ne doit pas sur-contraindre le problème (i.e. $G \in G(n, m), m \geq n, n \in \mathbb{R}, m \in \mathbb{R}$) : cette condition est toujours respectée dans le cas présent (mais on verra qu'il faudra y porter une attention particulière par la suite)
- G doit être de rang plein : sans l'avoir vérifiée de manière théorique, cette condition ne nous a posé aucun problème sur les cas pratiques mis en oeuvre

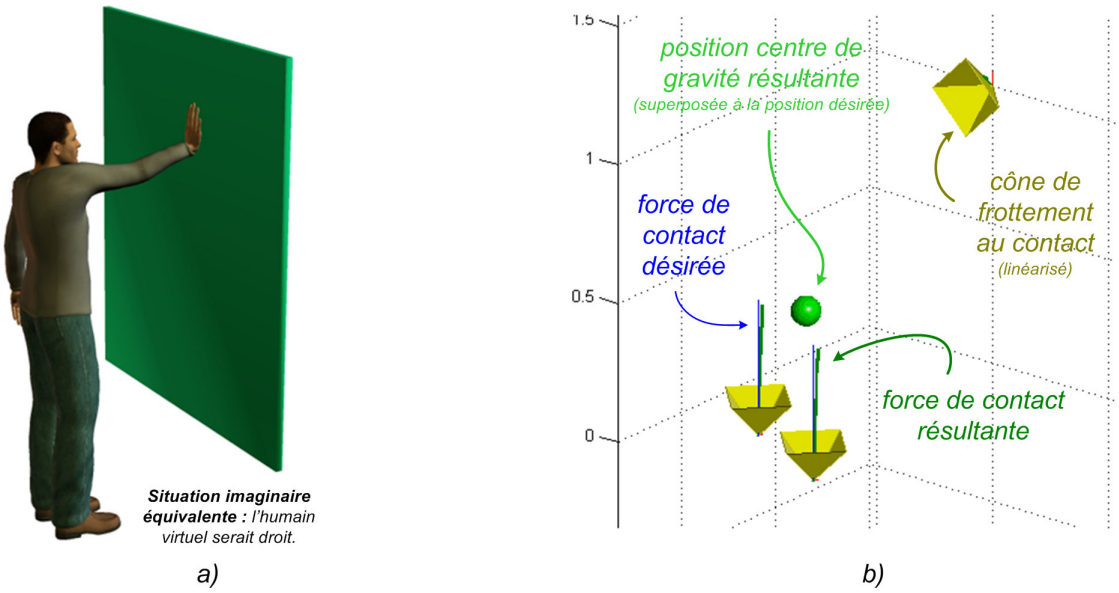

FIG. 3.12 – Mise en œuvre de l'optimisation (3.69). La position désirée du centre de gravité est entre les pieds. Comme indiqué en b), le système parvient à trouver des forces de contact et une position du centre de gravité proche de sa position désirée satisfaisant le problème. Ce cas de figure correspondrait à un mannequin dans la configuration donnée en a). Nous avons fait des simulations Matlab®, l'implémentation de l'algorithme sur un humain virtuel est laissée en perspectives. La force de contact au niveau de la main est quasi-nulle, ce qui correspond à la solution qu'un véritable humain adopterait.

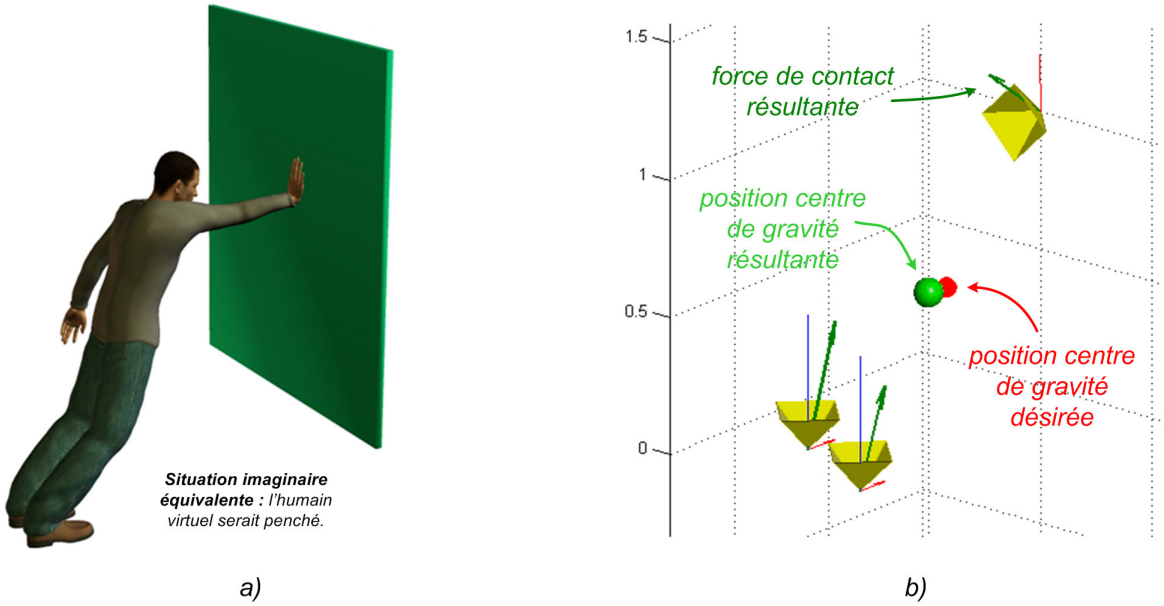

FIG. 3.13 – Mise en œuvre de l'optimisation (3.69). La situation équivalente serait maintenant celle d'un humain virtuel penché, appuyé sur le mur, comme indiqué en a). Son centre de gravité se trouverait donc légèrement en avant, comme vu en b). La force de contact désirée au niveau de la main est nulle, mais puisque l'humain virtuel est penché, appuyé sur le mur, la force résultant de l'optimisation est relativement importante.

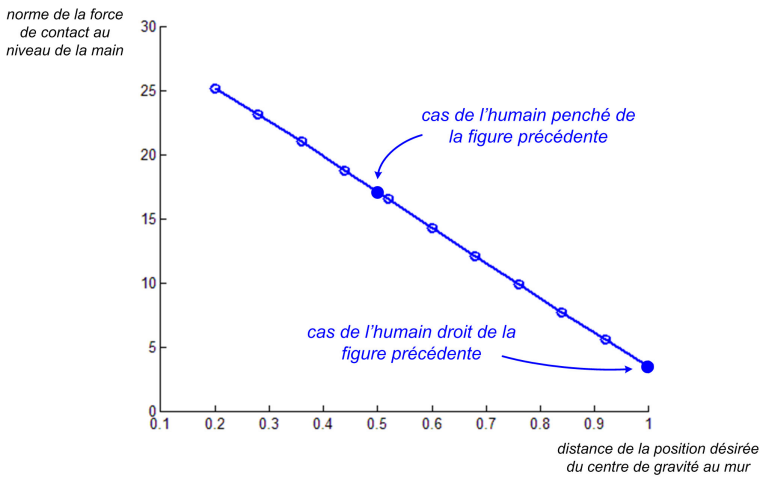

Fig. 3.14 – Evolution de la force de contact au niveau de la main, fonction de la distance de la position désirée du centre de gravité au mur : plus l'humain se penche et plus la force de contact est importante. C'est le comportement attendu.

Notons que cette algorithme permet de poser les principes de la méthode, mais ne tient pas compte des limitations réelles du mannequin, ainsi x_{com} est laissé non contraint. Nous inclurons certaines contraintes (ex : limites articulaires...) au fur et à mesure du développement.

Introduction de marges de stabilité

L'algorithme présenté ici permet de trouver des configurations qui respectent l'équilibre statique du mannequin, ainsi que le non glissement des contacts. A son issue, si une configuration stable existe, elle sera trouvée et l'humain virtuel ne tombera pas. Cependant est-elle robuste aux perturbations ? A cette fin nous introduisons la notion de *marge de stabilité* à notre contrôleur d'équilibre.

Pour être stable, les contraintes $GX = W$ et $I_f X \geq 0$ doivent être respectées, c'est-à-dire que l'optimisation mise en place ci-avant s'exprime par :

$$X = G^{+}W + \Pi\eta \tag{3.72}$$
$$= \Pi\eta + \nu, \tag{3.73}$$

avec Π, projection dans le noyau de G. η est choisi pour faire respecter $I_f X \geq 0$ et optimiser le critère dont nous avons parlé ci-avant. En décomposant, de la même manière que dans [BLR05], nous avons

$$X = \begin{bmatrix} f_c^{dis} \\ x_{com} \end{bmatrix} = \begin{bmatrix} \Pi_f\eta + \nu_f \\ \Pi_x\eta + \nu_x \end{bmatrix}, \tag{3.74}$$

la contrainte d'adhérence des contacts s'exprime donc par le polytope[3] :

$$\Pi_f\eta + \nu_f \geq 0. \tag{3.75}$$

C'est-à-dire que pour que les contacts soient non glissants, il faut que η se situe à l'intérieur du polytope exprimé par (3.75). Une condition supplémentaire portant sur l'addition des marges de stabilité s'exprimera en restreignant ce polytope.

[3]Figure géométrique délimitée par des portions de droites, de plans, ou d'hyperplans.

Marges exprimées en terme de position du centre de gravité

Effectuons un changement de variable η de (3.74) de manière à identifier les dimensions de l'espace de η ayant une influence sur x_{com}. Effectuons pour cela une décomposition SVD de Π_x :

$$\Pi_x = U\Sigma V^t \tag{3.76}$$
$$= U \begin{bmatrix} \Sigma_r & 0 \end{bmatrix} V^t, \tag{3.77}$$

avec, U et V orthogonales de dimension respective $m_u = 3$ et $m_v = m_c m_a + 3$ et Σ_r diagonale. Posons

$$\Sigma_{r_{ext}} = \begin{bmatrix} \Sigma_r & 0 \\ 0 & I_{m_v - m_u} \end{bmatrix} \qquad U_{ext} = \begin{bmatrix} U & 0 \\ 0 & I_{m_v - m_u} \end{bmatrix} \qquad T = V\Sigma_{r_{ext}}^{-1} U_{ext}^t. \tag{3.78}$$

Nous avons

$$\Pi_x T = U \begin{bmatrix} \Sigma_r & 0 \end{bmatrix} V^t V \Sigma_{r_{ext}}^{-1} U_{ext}^t \tag{3.79}$$
$$= U \begin{bmatrix} \Sigma_r & 0 \end{bmatrix} V^t V \begin{bmatrix} \Sigma_r^{-1} & 0 \\ 0 & I_{m_v - m_u} \end{bmatrix} \begin{bmatrix} U^t & 0 \\ 0 & I_{m_v - m_u} \end{bmatrix} \tag{3.80}$$
$$= \begin{bmatrix} I_{m_u} & 0 \end{bmatrix}, \tag{3.81}$$

x_{com} est donc influencé uniquement par les $m_u = 3$ premières dimensions de ξ, où ξ représente les coordonnées de η dans la nouvelle base donnée par la transformation T : $\eta = T\xi$. D'ailleurs ces 3 premières coordonnées de ξ, sont aussi les coordonnées de x_{com} à ν_x près.

Exprimer une marge de stabilité en terme de position du centre de gravité revient à trouver une configuration d'équilibre dans laquelle la stabilité est assurée, même en perturbant quelque peu la position du centre de gravité du personnage virtuel. Nous devons donc ajouter des contraintes qui nous assureront que même en cas de perturbation extrême désirée nous sommes en situation stable. C'est-à-dire que le mannequin doit être stable pour la configuration ξ, mais aussi pour les configurations $\xi + marge$. Ces m_p marges Δx_i sont données par l'utilisateur. Ainsi si nous souhaitons que le système soit suffisamment robuste pour résister à une perturbation de $10cm$ suivant la direction x, on prendra $\Delta x = \begin{bmatrix} 0.1 \\ 0 \\ 0 \end{bmatrix}$. Elles définissent chacune un nouveau polytope, exprimant les contraintes unilatérales. Il est donné par :

$$\Pi_f T \left(\xi + \begin{bmatrix} \Delta x_i \\ 0 \end{bmatrix} \right) + \nu_f \geq 0 \tag{3.82}$$
$$\Pi_f T \xi + \nu'_{f_i} \geq 0, \tag{3.83}$$

avec $\nu'_{f_i} = \Pi_f T \begin{bmatrix} \Delta x_i \\ 0 \end{bmatrix} + \nu_f$.

Nous devrons respecter toutes les contraintes des nouveaux polytopes, ainsi que celles du polytope initial. C'est-à-dire que la nouvelle contrainte unilatérale est l'intersection du polytope initial et des m_p polytopes calculés pour chaque marge extrême définie par l'utilisateur :

$$\begin{bmatrix} \Pi_f T \xi + \nu_f \\ \Pi_f T \xi + \nu'_{f_1} \\ \vdots \\ \Pi_f T \xi + \nu'_{f_{m_p}} \end{bmatrix} \geq 0. \tag{3.84}$$

Ceci augmente énormément le nombre de ces contraintes. Sur (3.83), nous voyons que ces contraintes ne font que rapprocher les facettes du polytope (3.75) de son centre. Donc l'intersection de tous les polytopes sera donnée par les facettes les plus contraignantes (les plus proches du centre). Aussi pour

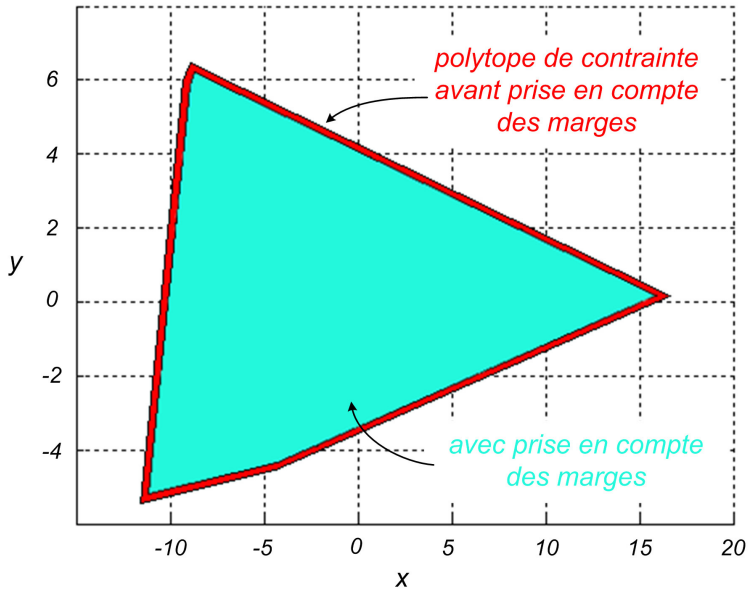

Fig. 3.15 – Polytope des contraintes unilatérales projeté suivant les dimensions correspondant au centre de gravité avant et après introduction des marges de stabilité (il n'est pas borné sur z_c).

éliminer les facettes inutiles et nous ramener au même nombre de contraintes que dans le cas initial, nous prendrons le ν'_f tel que :

$$\Pi_f T\xi + \nu'_f \geq 0, \tag{3.85}$$

avec

$$\nu'_f = \min_i \left(\nu'_{f_i}\right), \tag{3.86}$$

Les polytopes de contrainte avant et après introduction des marges de stabilité peuvent être visualisés figure 3.15. La contrainte bilatérale d'équilibre est alors modifiée pour prendre en compte les marges, elle est donnée par :

$$X = \begin{bmatrix} f_c^{dis} \\ x_{com} \end{bmatrix} = \begin{bmatrix} \Pi_f \eta + \nu'_f \\ \Pi_x \eta + \nu_x \end{bmatrix} = \Pi T\xi + \nu', \tag{3.87}$$

donc

$$GX = G\nu' \tag{3.88}$$

$$= W'. \tag{3.89}$$

Nous avons établi en (3.89) la contrainte bilatérale permettant de mettre en œuvre des marges de stabilité. La contrainte unilatérale étant la même qu'en (3.69). En mettant également le même critère en œuvre qu'en (3.69), le problème à résoudre se traduit alors de manière mathématique par :

$$\left\{ \begin{array}{l} \min\limits_X \left(\left\| X - \begin{bmatrix} f_{c_d}^{dis} \\ x_{com_d} \end{bmatrix} \right\|_{Q_1}^2 + \alpha \left\| \left(I - NN^t\right) AX \right\|_{Q_2}^2 \right) \\ \quad \text{sachant } \left\{ \begin{array}{l} GX = W' \\ I_f X \geq 0. \end{array} \right. \end{array} \right. \tag{3.90}$$

La contrainte statique est donnée par $GX = W$, or on résout $GX = W'$, c'est-à-dire que l'adjonction de marges de stabilité se traduit par des efforts supplémentaires *artificiels* sur le système, donnés par :

$$G_f(\nu'_f - \nu_f), \text{ avec } G = \left[\begin{array}{cc} G_f & G_x \end{array} \right]. \tag{3.91}$$

Ces efforts biaisent l'équilibre, aussi il nous faut retirer leur influence, ce qui, au regard des équations (3.87) et (3.74), se fait en prenant :

$$X^{final} = X - \left[\begin{array}{c} \nu'_f - \nu_f \\ 0 \end{array} \right]. \tag{3.92}$$

Une comparaison des résultats avec et sans marge de stabilité en position est donnée figure 3.16.

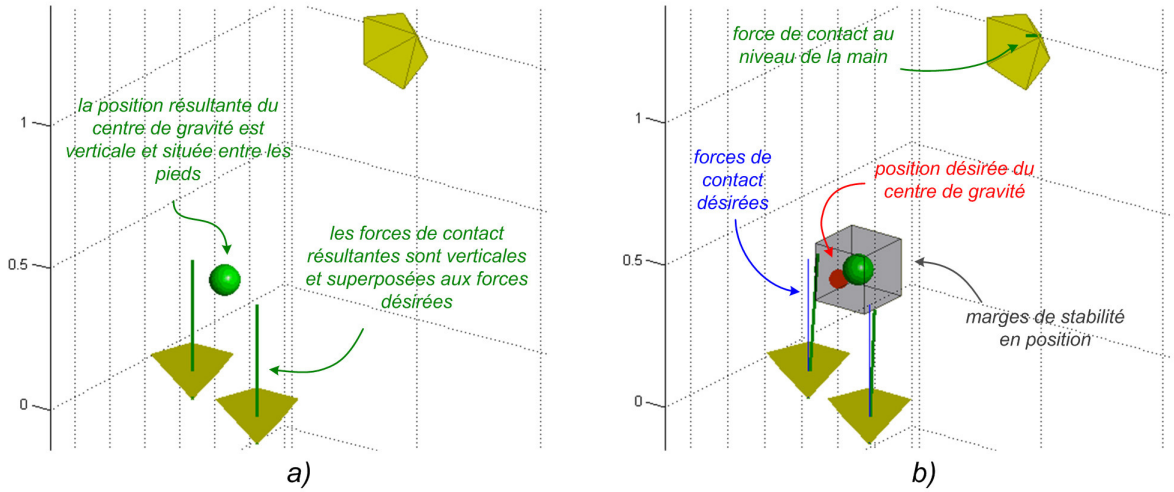

FIG. 3.16 – a) Configuration sans marge de stabilité : le centre de gravité est situé entre les pieds et la force de contact au niveau de la main est nulle. C'est-à-dire que l'humain virtuel est en équilibre debout, comme en 3.12a. b) La spécification du problème inclut une marge de stabilité en position : le centre de gravité s'avance et une force de contact apparaît sur le mur. C'est-à-dire que l'humain virtuel s'appuye sur le mur et se penche, comme en 3.13a - c'est la solution qui serait adoptée de manière intuitive par un véritable humain.

Marges exprimées en terme de forces perturbatrices

Dans cette partie, au lieu d'exprimer les marges en terme de position du centre de gravité, nous les formulons en terme de forces perturbatrices. Dans ce cas il faut que dans la *configuration solution*, l'humain virtuel reste stable même en ajoutant les forces de perturbation maximales ΔW_i (choisies au préalable par l'utilisateur), au système. Dans la formulation initiale il fallait respecter (3.75) :

$$\begin{aligned} f_c^{dis} &= \Pi_f \eta + \nu_f \\ &= \Pi_f \eta + (G^+)_f W \geq 0, , \text{ avec } G^+ = \left[\begin{array}{c} (G^+)_f \\ (G^+)_x \end{array} \right]. \end{aligned} \tag{3.93}$$

Dans le cas avec marges, il faut respecter la même contrainte avec les efforts extérieurs supplémentaires

$$f_c^{dis} = \Pi_f \eta + (G^+)_f (W + \Delta W_i) \geq 0. \tag{3.94}$$

ATTENTION : $(G^+)_f \neq (G_f)^+$

Il suffit donc de remplacer l'expression de ν'_{f_i} de (3.83) par :

$$\nu'_{f_i} = G_f^+ \Delta W_i + \nu_f, \text{ avec } G^+ = \begin{bmatrix} G_f^+ \\ G_x^+ \end{bmatrix}. \tag{3.95}$$

Nous pouvons alors procéder à la même sélection $\nu'_f = \min_i \left(\nu'_{f_i}\right)$ qu'en (3.86) et mettre en œuvre la même optimisation qu'en (3.90). Les résultats d'un problème avec prise en compte de marges de stabilité en effort est donné figure 3.17.

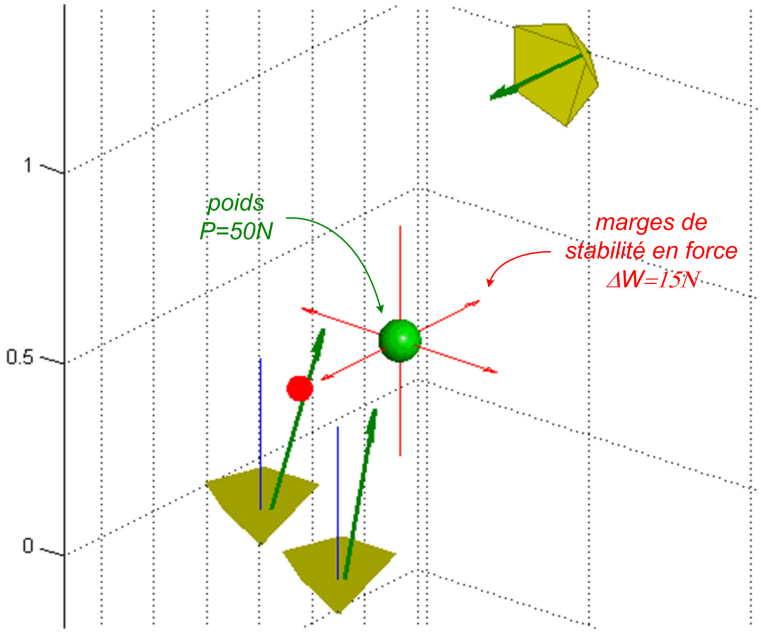

Fig. 3.17 – Dans ce cas des marges de stabilité en effort, le comportement est très marqué : le centre de gravité s'avance plus qu'en 3.16b (c'est-à-dire que l'humain virtuel est plus penché) et la force de contact au niveau de la main est plus importante. C'est bien le comportement auquel on s'attend.

Jusqu'à présent le problème était résolu en prenant uniquement en compte les trois degrés de liberté de x_{com} pour définir la configuration du mannequin. Nous envisageons maintenant la résolution d'un problème plus complexe intégrant les degrés de liberté articulaires du mannequin.

Configuration de l'humain virtuel

Dans le cas où on prend en compte les degrés de liberté articulaires, le problème devient non-linéaire. Pour le résoudre, nous le formulons de manière variationnelle, en linéarisant x_{com} à l'instant k. Cette approche offre une richesse de solutions intéressante.

$$\boxed{x_{com_{k+1}} \simeq x_{com_k} + J^r_{com}\delta q,} \tag{3.96}$$

avec J^r_{com} déjà exprimée en (3.36). Nous insérons ce résultat dans (3.89) :

$$\begin{bmatrix} G_f & G_x \end{bmatrix} \begin{bmatrix} f_c^{dis} \\ x_{com_k} + J^r_{com}\delta q \end{bmatrix} = W' \tag{3.97}$$

$$\begin{bmatrix} G_f & G_x J^r_{com} \end{bmatrix} \begin{bmatrix} f_c^{dis} \\ \delta q \end{bmatrix} = W' - G_x x_{com_k}. \tag{3.98}$$

Nous souhaitons que l'humain virtuel reste en contact avec l'environnement après résolution de l'équilibre par l'algorithme. Comme l'algorithme gère la configuration du robot, nous devons lui adjoindre des contraintes de *conservation des contacts*. Nous savons que $V_{O_{c_i}\in c_i/0(0)}$ (torseur cinématique du repère associé au point de contact c_i dans son mouvement par rapport à 0, exprimé dans la base 0 au point O_{c_i}, centre du repère associé à c_i) est donné par :

$$V_{O_{c_i}\in c_i/0(0)} = J_{O_{c_i}\in c_i/0(0)}\dot{q}, \tag{3.99}$$

exprimé en petits déplacements, on a :

$$\delta x_{O_{c_i}\in c_i/0(0)} = J_{O_{c_i}\in c_i/0(0)}\delta q, \tag{3.100}$$

avec $x_{O_{c_i}\in c_i/0(0)}$ position de O_{c_i} (centre du repère c_i), fixe dans c_i par rapport au repère 0, projeté dans 0, exprimée dans $\mathfrak{se}(3)$. $\delta x_{O_{c_i}\in c_i/0(0)}$ est une position 6D, nous souhaitons uniquement assurer la position 3D du contact, nous ne gérerons pas l'orientation ici :

$$\delta x^r_{O_{c_i}\in c_i/0(0)} = \begin{bmatrix} 0 & {}^0R_{c_i} \end{bmatrix} J_{c_i/0(c_i)}\delta q, \tag{3.101}$$

cette relation permet de modifier la configuration au cas où les contacts s'écartent de leur position souhaitée. Nous reécrivons donc la contrainte bilatérale :

$$\begin{bmatrix} G_f & G_x J^r_{com} \\ 0 & \begin{bmatrix} 0 & {}^0R_{c_1} \end{bmatrix} J_{c_1/0(c_1)} \\ \vdots & \vdots \\ 0 & \begin{bmatrix} 0 & {}^0R_{c_{m_c}} \end{bmatrix} J_{c_{m_c}/0(c_{m_c})} \end{bmatrix} \begin{bmatrix} f_c^{dis} \\ \\ \delta q \end{bmatrix} = \begin{bmatrix} W' - G_x x_{com_k} \\ \delta x^r_{O_{c_1}\in c_1/0(0)} \\ \vdots \\ \delta x^r_{O_{c_{m_c}}\in c_{m_c}/0(0)} \end{bmatrix}, \tag{3.102}$$

qu'en redéfinissant X nous noterons :

$$G''X = W''. \tag{3.103}$$

Comme nous avons linéarisé le problème (3.96), nous avons fait l'hypothèse des petits déplacements, il nous faut donc nous assurer que l'on n'en sorte pas :

$$\begin{bmatrix} I_q \\ \\ -I_q \end{bmatrix} X \geq \begin{bmatrix} \delta q^{dep}_{min} \\ \\ -\delta q^{dep}_{max} \end{bmatrix}, \tag{3.104}$$

avec $I = \begin{bmatrix} I_f \\ I_q \end{bmatrix}$. Il est également possible de s'assurer de ne pas dépasser les limites articulaires :

$$\begin{bmatrix} I_q \\ \\ -I_q \end{bmatrix} X \geq \begin{bmatrix} \delta q^{lim}_{min} \\ \\ -\delta q^{lim}_{max} \end{bmatrix}. \tag{3.105}$$

Hypothèse des petits déplacements et limites articulaires s'exprimant de la même manière, nous prenons les limites les plus contraignantes.

Nous pouvons également rajouter des tâches à effectuer (consignes de position de points de contrôle), en rajoutant des contraintes à (3.102), du type $\begin{bmatrix} 0 & {}^0R_i \end{bmatrix} J_{i/0(i)}\delta q = \delta x^r_{O_i\in i/0(0)}$, avec i indice du point de contrôle et n, nombre de tâches. L'appelation *contrôleur d'équilibre* est donc réductrice puisque l'on peut également envisager de contrôler des tâches :

$$\begin{bmatrix} G_f & G_x J^r_{com} \\ 0 & \begin{bmatrix} 0 & {}^0R_{c_1} \end{bmatrix} J_{c_1/0(c_1)} \\ \vdots & \vdots \\ 0 & \begin{bmatrix} 0 & {}^0R_{c_{m_c}} \end{bmatrix} J_{c_{m_c}/0(c_{m_c})} \\ 0 & \begin{bmatrix} 0 & {}^0R_1 \end{bmatrix} J_{1/0(1)} \\ \vdots & \vdots \\ 0 & \begin{bmatrix} 0 & {}^0R_n \end{bmatrix} J_{n/0(n)} \end{bmatrix} \begin{bmatrix} f_c^{dis} \\ \\ \delta q \end{bmatrix} = \begin{bmatrix} W' - G_x x_{com_k} \\ \delta x^r_{O_{c_1}\in c_1/0(0)} \\ \vdots \\ \delta x^r_{O_{c_{m_c}}\in c_{m_c}/0(0)} \\ \delta x^r_{O_1\in 1/0(0)} \\ \vdots \\ \delta x^r_{O_n\in n/0(0)} \end{bmatrix}. \tag{3.106}$$

Nous choisissons de minimiser le déplacement articulaire. A l'issue de toutes ces observations, nous pouvons poser le problème d'optimisation, que nous résoudrons, comme en (3.70), sous forme de LCP :

$$\left\{ \begin{array}{l} \min\limits_{X} \left(\left\| X - \begin{bmatrix} f_{c_d}^{dis} \\ 0 \end{bmatrix} \right\|_{Q_1}^2 + \alpha \left\| \left(I - NN^t \right) AX \right\|_{Q_2}^2 \right) \\ \quad \left\{ \begin{array}{c} G'' X = W'' \\ \text{sachant} \quad \left\{ \begin{bmatrix} I_f \\ I_q \\ -I_q \end{bmatrix} X \geq \begin{bmatrix} 0 \\ \delta q_{min} \\ -\delta q_{max} \end{bmatrix} . \end{array} \right. \end{array} \right. \tag{3.107}$$

La méthodologie adoptée est donnée en pseudo-code dans l'algorithme 5 page 111.

Nous pourrons noter que la matrice entrée dans l'algorithme de Lemke, choisi pour la résolution du LCP, étant symétrique positive, si Lemke ne trouve pas de solution, c'est qu'il n'en existe pas satisfaisant le problème.

ATTENTION : pour passer d'un problème exprimé sous la forme d'une optimisation quadratique à un problème exprimé sous la forme d'un LCP, il est nécessaire que $G'' \in G(n,m), m \geq n, n \in \mathbb{R}, m \in \mathbb{R}$. Cela signifie que le nombre de tâches que l'on peut ajouter est limité. Il faut y prendre garde lorsque l'on pose le problème.

Une implémentation Matlab® a permis de tester ces méthodes pour le cas multi-contacts non-coplanaires, comme on peut le voir figure 3.18. L'intégration à la plate-forme temps-réel est envisagée comme suite des travaux.

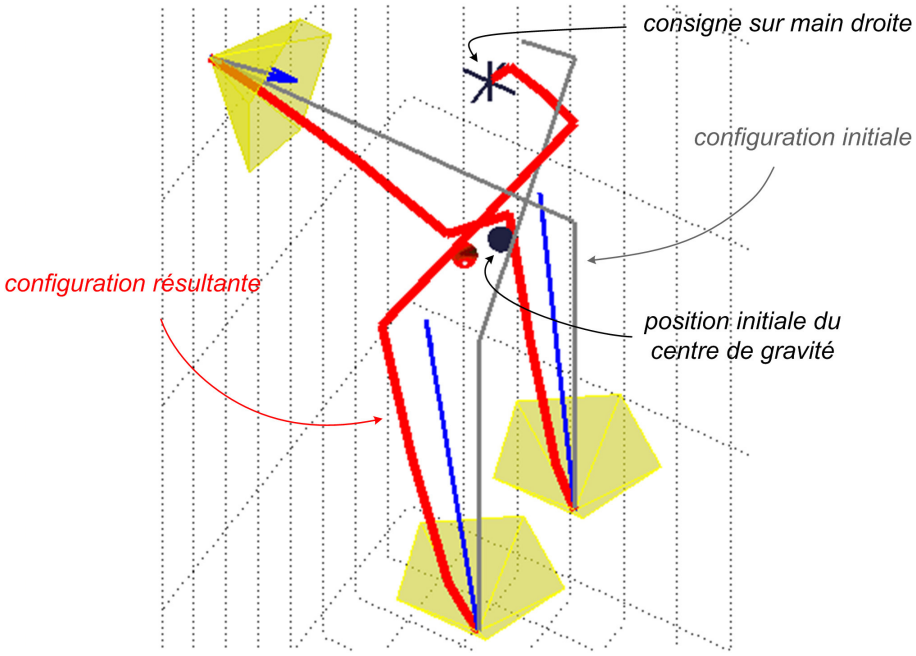

Fig. 3.18 – Résolution de l'équilibre avec marges de stabilité sur le squelette d'un mannequin. Là encore le mannequin se penche en avant pour assurer la marge de stabilité. Les points de contact sont bien immobiles et la consigne sur la main droite est bien atteinte.

Conclusion

Dans ce chapitre, nous avons d'abord établi un schéma de résolution de contraintes unilatérales, avec pour cœur l'algorithme de Lemke permettant de résoudre des LCP. Nous avons montré que dans ce cas l'étage de gestion des contraintes et celui d'intégration étaient fortement imbriqués (la vue fonctionnelle bien découplée qu'apporte le schéma n° 1.31 n'est donc qu'une première approximation).

Nous avons ensuite montré comment poser les problèmes du contact sans frottement et des limites articulaires, pour pouvoir les résoudre par notre approche.

Nous nous sommes ensuite attachés à la problématique de la conservation de l'équilibre. Nous avons proposé une première approche dans le cas de contacts coplanaires basée sur une caractérisation géométrique de l'équilibre. Elle s'appuie elle-même sur la projection du centre de gravité. Nous avons alors proposé des pistes de recherche prometteuses pour le non-coplanaire. En posant le problème sous la forme d'une optimisation quadratique, nous sommes parvenus à l'aborder de front, gérant la complexité inhérente à tous les degrés de liberté du système.

Nous allons maintenant aborder des comportements plus complexes, de plus haut niveau, en élargissant quelque peu notre vision du contrôle.

Algorithme 5 : Equilibre dans le cas multi-contacts non-coplanaires (marges en effort).

Données :

${}^0H_{c_i}$	:	position associée au repère du contact i
a_i	:	arêtes associées au cône du contact i
$m,\ g$	:	respectivement masse et gravité
${}^0H_{ext_k},\ W_{ext_k}$	:	repères et torseurs d'effort associés

Résultat : H : position courante des os du squelette

pour *tous les contacts* **faire**
| ${}^0H_{c_{i_d}} = {}^0H_{c_i}$: position désirée des contacts

fin

répéter

$G = \begin{bmatrix} G_f & G_x \end{bmatrix}$ (expression de G donnée en (3.59))

W (expression de W donnée en (3.59))

$G^+ = \begin{bmatrix} (G^+)_f & (G^+)_x \end{bmatrix} = G^t(GG^t)^{-1}$

$\nu = \begin{bmatrix} \nu_f \\ \nu_x \end{bmatrix} = G^+W$

$\nu'_f = \min_i \left(G_f^+ W_i^{disturb} \right)$

$W' = G\nu' = G \begin{bmatrix} \nu'_f \\ \nu_x \end{bmatrix}$

J^r_{com} : jacobienne réduite du centre de gravité (expression donnée en (3.36))

pour *tous les contacts* **faire**
| $J_{c_i/0(c_i)}$: jacobiennes associés aux contacts (cinématique directe)
| $\delta x^r_{O_{c_i} \in c_i/0(0)} = \begin{bmatrix} O & I \end{bmatrix} \log \left({}^0H_{c_{i_d}} {}^{c_i}H_0 \right)$

fin

$G'' \left(G_f, G_x, J^r_{com}, {}^0R_{c_i}, J_{c_i/0(c_i)} \right)$ (expression donnée en (3.102))

$W'' \left(W', G_x, x_{com_k}, \delta x^r_{O_{c_i} \in c_i/0(0)} \right)$ (expression donnée en (3.102))

$$\min_X \left(\left\| X - \begin{bmatrix} f^{dis}_{c_d} \\ 0 \end{bmatrix} \right\|^2_{Q_1} + \alpha \left\| \left(I - NN^t \right) AX \right\|^2_{Q_2} \right) \text{ sachant } \begin{cases} G''X = W'' \\ I_f X \geq 0 \end{cases}$$

$\int q \Rightarrow H$: position opérationnelle des os du mannequin

pour *tous les contacts* **faire**
| calcul de ${}^0H_{c_i}$

fin

jusqu'à *ce que l'utilisateur arrête la simulation*;

Commentaire : *Cet algorithme est envisagé pour une seule configuration de contacts. Pour simplifier son écriture, nous n'avons délibérément pas inclus de limite articulaire, ou de tâche additionnelle.*

CHAPITRE 4

Ordonnancement de modes de commande

Grâce aux chapitres 2 et 3, nous avons maintenant un contrôle bas niveau robuste, qui nous autorise à interagir de manière naturelle avec l'environnement par le biais de forces. Il nous permet de prendre en compte et de gérer de manière cohérente les différences entre monde réel et monde virtuel et ce en suivant au mieux *les mouvements spécifiés par la capture de mouvement. Nous pouvons donc maintenant nous atteler à la résolution de tâches de plus haut niveau. C'est-à-dire que les travaux effectués dans ce chapitre se situent à un niveau plus amont dans l'architecture globale donnée figure 1.26 (bloc bleu ciel marqué "Contrôle haut niveau").*

Pour l'instant l'opérateur contrôle les m degrés de liberté de sa variété de configuration par le biais de cibles 6D de capture de mouvement. Nous chercherons ici à restreindre le nombre de degrés de liberté à contrôler par l'opérateur, c'est-à-dire à pousser l'automatisation plus avant, de manière à réduire la charge de travail de l'opérateur.

Nous avons introduit, sans les nommer, deux modes de contrôle : le mode libre *(mode de commande usuel) et le mode* contraint *(qui intervient dès qu'un repère de contrôle est lié à un mécanisme virtuel). Nous allons introduire un troisième mode : le* mode automatique*, où la trajectoire désirée du repère de contrôle est générée automatiquement. Nous discuterons dans un premier temps des problématiques liées à leur implémentation, puis nous aborderons leur ordonnancement : comment et pourquoi nous passons de l'un à l'autre. L'ordonnancement des modes de commande marquant le paroxysme de nos possibilités actuelles, nous décrirons des résultats obtenus grâce à l'architecture complète mise en œuvre dans notre étude. Enfin nous exposerons certaines perspectives envisageables.*

4.1 Transition de modes de commande

Les chapitres 2 et 3 étaient focalisés sur le bloc vert du schéma d'architecture figure 1.26, notre attention se situe maintenant sur son bloc bleu plus haut niveau. Dans cette section nous nous appuyons sur les résultats déjà obtenus sur notre contrôleur bas niveau, pour les étendre à des cas plus complexes.

Grâce à la capture de mouvement, les repères de contrôle peuvent être commandés dans un mode que nous qualifierons de *libre*. C'est-à-dire que dans la mesure du possible, le mouvement du repère de contrôle virtuel reproduit le mouvement de son équivalent réel. En 2.2.2, nous avons introduit un mode de commande différent, qui permet de contraindre les mouvements de manière partielle, nous parlerons de mode *contraint*, *semi-automatique*, ou *guidé*. Enfin, il est bien évidemment envisageable de contraindre complètement le repère de contrôle en générant sa trajectoire de manière automatique, ce sera le mode *autonome*, encore dit *automatique*.

Dans une première section nous débattrons de l'intérêt, des enjeux et des difficultés liées à chaque mode de commande. Nous pourrons alors envisager de passer de l'un à l'autre en cours de simulation, suivant les situations imposées par l'exercice mené.

4.1.1 Mise en place des différents modes

Nous envisageons donc trois modes de commande différents pour les repères de contrôle. Le mode libre a fait l'objet de la section 2.2. Les modes contraint et automatique présentent des points qu'il nous semble intéressant d'éclaircir, au regard de leur utilisation ultérieure.

Mode de commande contraint

Nous avons vu en 2.2.2, que les modes de commande contraints peuvent être mis en œuvre de manière passive, par le biais de la notion de *mécanisme virtuel*. C'est cette méthode que nous allons utiliser ici.

Lors d'interactions avec un environnement réel, le sens du toucher est une des sources d'information traitées par le cerveau, qui influe notamment sur la précision de la tâche accomplie, sur sa vitesse... Or lorsqu'un acteur est immergé dans un dispositif de capture de mouvement, celui-ci ne possède pas d'équivalent de l'environnement virtuel. Il ne dispose donc pas de l'information haptique. Les mouvements de l'humain virtuel peuvent alors sembler gauches et imprécis, ce que nous souhaitons éviter. D'où l'utilité des guides virtuels qui vont aider l'humain virtuel à accomplir son mouvement, dans les situations de *carence perceptive*.

Comme il a été relevé précédemment, ce mode de commande semi-autonome permet également de gérer des degrés de liberté de manière automatique. L'opérateur aura donc moins de degrés de liberté à gérer. Il devient même envisageable de spécifier la commande avec des dispositifs d'entrée possédant peu de degrés de liberté, comme un trackball, ou une simple souris, de manière simple et aisée.

Le mécanisme virtuel, destiné à aider l'acteur, doit être spécifié en fonction de la tâche à accomplir. Ainsi guider une perceuse sur son axe se fera par l'intermédiaire d'un mécanisme virtuel de type pivot glissant lié à la perceuse et d'axe aligné avec l'axe désiré du trou, voir figure 2.20.

L'analogie mécanique permet d'appréhender facilement les mécanismes sous-jacents à l'emploi des mécanismes virtuels. Elle et autorise la mise en œuvre de contraintes simples exprimables par des mécanismes tels que nous avons l'habitude d'envisager. Mais qu'en est-il si nous souhaitons guider un repère de contrôle sur une spline, une NURBS, ou une autre surface gauche couramment utilisée dans les logiciels de CAO, pour faire poncer un capot de voiture au mannequin virtuel par exemple ?

Des travaux issus du domaine de la téléopération [Jol97] nous éclairent sur ce point. D'un point de vue mathématique, les mécanismes virtuels peuvent être vus comme des contraintes bilatérales holonomes. Nous devons connaître leur jacobienne, en vue du calcul de leur évolution (ces calculs ont fait l'objets de la

section 2.2.1). Cette vision des mécanismes virtuels permet de considérer des contraintes plus complexes et intéressants : nous pouvons maintenant envisager d'avoir pour mécanisme de contrainte, des splines, voir même des fonctions plus exotiques... L'analogie mécanique peut donc être dépassée pour peu que le modèle géométrique soit C^1 et que sa jacobienne soit de rang plein, pour éviter les problèmes numériques.

Il convient d'ajouter que nous avons abordé les guidages en position, mais comme spécifié dans [Jol94], ces contraintes peuvent être étendues à des cas de guidages en effort, ou hybrides position/force. C'est-à-dire qu'il est possible de commander en position dans certaines directions et en force dans d'autres, voir de laisser libre dans un troisième ensemble de directions. Ceci peut être réalisé en construisant un mécanisme virtuel dont certaines articulations seront commandées en force, d'autres en position (pour laisser une direction libre, son articulation associée sera commandée en force, avec une consigne nulle). Nous voyons donc que nous pouvons dans ce cas classer les degrés de liberté du mécanisme virtuel en deux ensembles :

- le premier dont les articulations sont commandées en position, ou en force avec une consigne nulle, permet de se déplacer sur la contrainte de position
- le second commandé en efforts permet de plaquer l'humain virtuel sur la contrainte et d'appliquer des forces désirées à l'environnement.

En liant le repère de contrôle considéré de l'humain virtuel à l'effecteur du mécanisme virtuel, on obtient la commande désirée - elle est illustrée figure 4.1.

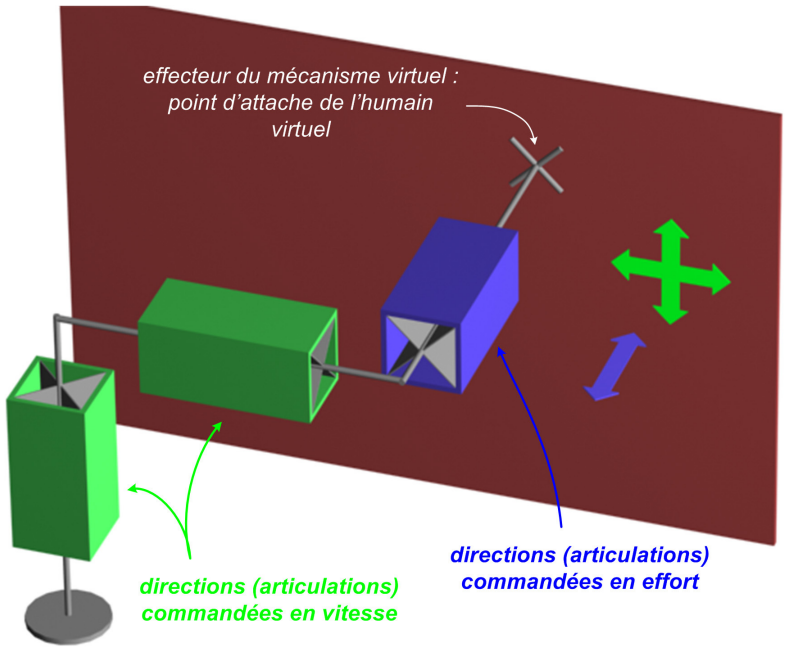

FIG. 4.1 – Contrôle hybride position/force par les mécanismes virtuels. Le repère de contrôle, attaché à l'effecteur du mécanisme virtuel, pourra se déplacer sur le plan rouge et exercer une force perpendiculairement.

Mode de commande autonome

Ce mode de commande correspond à de la génération de trajectoire, mais nous tenons à signaler que nous n'envisageons pas le problème dans toute sa complexité. En effet le domaine de la génération de trajectoire fait souvent référence à des méthodes d'évitement d'obstacle mettant en œuvre toute la configuration du robot (voir 1.2.1). Notre génération de trajectoire est une simple recherche de chemin entre deux points de l'espace opérationnel, de manière à pouvoir fournir automatiquement une consigne

à un repère de contrôle de notre mannequin.

Il existe des méthodes pour interpoler dans l'espace opérationnel : le groupe spécial euclidien $SE(3)$. Le principal obstacle à l'interpolation dans ce groupe de Lie, est sa courbure[1]. Nous pourrions envisager de mettre en œuvre des méthodes variationnelles pour trouver des géodésiques de $SE(3)$, méthode lourde et coûteuse. [Moa02] propose de passer par son algèbre associée $\mathfrak{se}(3)$ (voir 1.2.1.1). $SE(3)$ et $\mathfrak{se}(3)$ sont isomorphes sur $[0;\pi[$, de plus les géodésique de $\mathfrak{se}(3)$ sont les géodésiques de $SE(3)$ [Moa02]. $\mathfrak{se}(3)$ étant plan, ses géodésiques sont des segments de droites. Cette technique est décrite sur l'algorithme 6.

Algorithme 6 : Interpolation géodésique dans SE(3)

Données :
– $\{H_0; H_1\} \in SE(3)^2$: points extrêmes de la géodésique.
– $\gamma \in [0;1]$: abscisse curviligne normalisée (paramétrant la géodésique).
Résultat : H_γ : position interpolée

$h_0 = \log(H_0)$
$h_1 = \log(H_1)$
$h_\gamma = (1-\gamma)h_0 + \gamma h_1$
$H_\gamma = \exp(h_\gamma)$

Malheureusement, rien ne nous indique que la projection des géodésiques de $SE(3)$ sur $\mathbb{R}^3$ soient les géodésiques de $\mathbb{R}^3$. Et de fait, de manière générale ça n'est pas le cas, c'est-à-dire que le centre d'un repère décrivant une géodésique de $SE(3)$ se déplace en fait sur une courbe (non droite) de $\mathbb{R}^3$. Ce comportement a beau être exact d'un point de vue mathématique, il n'est pas ergonomique. Aussi, nous n'effectuons pas les interpolations dans $SE(3)$, mais dans $\mathbb{R}^3 \times SO(3)$ ($SO(3)$ défini en 1.2.1.1), de manière à interpoler sur des droites de $\mathbb{R}^3$. L'interpolation sur $\mathbb{R}^3$ est évidente, celle sur le groupe des rotations $SO(3)$ se fera elle de la même manière que dans le cas de l'algorithme 6, en remplaçant le log défini sur $SE(3)$ par sa version exprimée dans $SO(3)$.

Réglage des gains

Le couplage au mécanisme virtuel se fait par un couplage interne que nous avons déjà évoqué en 2.2.2. Le mannequin est de caractéristiques physiques (morphologie, poids...) connues et les paramètres de la simulation sont les suivants :

– le pas de temps d'intégration : δt
– les frottements visqueux articulaires : b_a
– les gains des couplages externes : k_c, b_c
– les gains des couplages internes : $k_{c_{int}}$, $b_{c_{int}}$
– la masse m et la gravité g.

Si nous souhaitons coupler un mannequin déjà paramétré à un mécanisme virtuel, il faut mettre en œuvre un couplage interne. Pour qu'il soit jugé *suffisamment raide*, ses gains devront permettre de contrecarrer les efforts générés par tous les correcteurs agissant sur le mannequin, ils doivent donc être relativement élevés.

Des gains élevés vont générer des efforts élevés. Or à δt donné, les efforts applicables sont limités pour conserver la propriété de stabilité (voir 2.2.1). Bien souvent les $k_{c_{int}}$, $b_{c_{int}}$ nécessaires à l'obtention d'un couplage raide sont trop élevés pour la simulation.

Du fait de nos contraintes temps-réel, nous ne pouvons diminuer δt. Aussi la méthode que nous proposons est de conserver des gains de couplage internes nous assurant la stabilité. Le couplage n'étant plus suffisamment raide, nous diminuerons en proportion les gains des couplages externes et pour conserver le

[1] Qui provient des contraintes d'orthogonalité de la matrice de rotation et de son déterminant qui doit être unitaire, voir 1.2.1.1

même comportement du mannequin, il nous faut également diminuer la matrice des frottements visqueux articulaires b_a d'un même ordre de grandeur.

Nous avons un problème similaire de réglage en ce qui concerne le poids. En effet, lorsque le poids est appliqué, notre mannequin s'affaisse. Pour équilibrer cette réaction, un contrôle interne est appliqué au mannequin destiné à lui redonner une attitude naturelle. Si le poids est trop important, les couples articulaires générés par le contrôle interne sont trop élevés pour garantir une simulation stable. Dans ces cas, nous pouvons le réduire de manière à se ramener à un cas stable. Pour conserver un comportement similaire, nous pouvons là encore diminuer les frottements visqueux b_a.

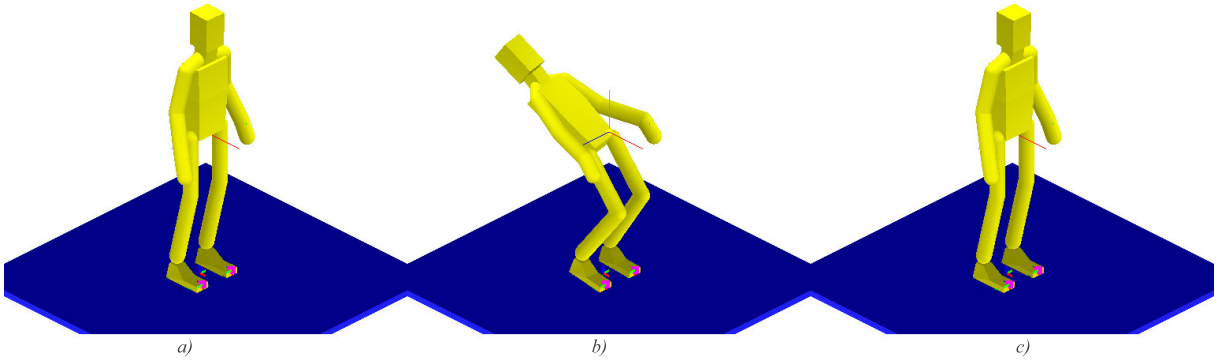

FIG. 4.2 – a) Mannequin sans poids. b) Avec du poids, le mannequin est affaissé. c) Un potentiel interne (voir 2.2.1.4) - s'il est bien dimensionné - permet de lui redonner une attitude naturelle.

Nous pouvons nous aider de relations du type (2.11) pour régler les gains.

4.1.2 Ordonnancement

Sur l'architecture du système donnée schéma 1.32, cette fonctionnalité correspond au bloc donné illustration 4.3.

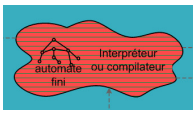

FIG. 4.3 – Bloc du schéma d'architecture 1.32, correspondant à la fonctionnalité décrite dans cette partie.

Nous avons trois modes de commande différents à disposition. Il faut maintenant pouvoir passer de l'un à l'autre en cours de simulation. Idéalement, le système passerait d'un mode à un autre aux moments jugés opportuns dans le contexte de la simulation en cours. Nous parlerons d'*ordonnancement des modes de commande*.

Dans ce contexte l'opérateur d'indique les conditions de changement de mode du système, ainsi que le mode (ou état) suivant et ses paramètres associés (ce pourra être la position finale de notre générateur de trajectoire, la description du mécanisme virtuel à mettre en œuvre...). Ces transitions pouvent être conditionnées par la réalisation d'une action, la proximité d'une zone sensible... Cette méthodologie est à rapprocher des travaux liés au système DANCE® de P. Faloutsous [Fal]. Donnons maintenant quelques exemples de manière à clarifier la méthode :

- le mannequin visse un écrou en mode libre, quand l'écrou est suffisamment serré (ce dont nous jugerons d'une manière idoine), nous décidons de passer en mode automatique et d'aller reposer automatiquement la clé de serrage.
- nous pouvons également décider que la pénétration du mannequin dans une zone sensible enclenche un mode guidé. Ainsi quand la main d'un humain virtuel, tenant une perceuse, s'approche d'une zone sensible située à proximité d'un objet à percer, une contrainte se met en place pour aider la perceuse à rester perpendiculaire à la surface dans laquelle le trou doit être effectué.
- ...

Donner ces informations au système permet de spécifier en partie son comportement, on dira alors que l'on spécifie un scénario de simulation [GCL05].

Nous sommes donc amenés à mettre en œuvre une méthodologie de spécification de ces comportements, c'est-à-dire un langage qui permettra de le décrire. Nous abordons alors la théorie des automates, des langages formels et de la calculabilité, domaines qui dépassent largement le cadre de notre étude. Nous nous bornons à étudier les aspects *automatiques* et limitons les points informatiques à la partie congrue, laissant la complexité des langages aux spécialistes du domaine ([Ste04]).

Comme mis en évidence dans la section 1.2.2, des tentatives d'élaboration de langages de spécification de comportement d'humain virtuels ont déjà eu lieu. Les langages de *Parameterized Action Representation* (PAR) de Badler [AKA+02], [AB03], les *interfaces comportementales* de Thalmann [AKM+02], [KT99], ou le langage HPTS++® [DDLT02] développé par S. Donikian, en sont des exemples. Ce dernier permet par exemple, par le biais d'un management des ressources disponibles, de gérer des tâches conflictuelles (dites mutuellement exclusives) comme fumer, boire et lire en même temps [LD02]. Le mannequin, pose ainsi automatiquement son journal avant de reprendre sa tasse. Ces langages sont de type formel, ils ont donc une puissance d'expression importante[2], mais sont principalement dédiés à donner des règles de comportement à des humains autonomes. Dans le cadre des hypothèses que nous nous sommes fixés, cette problématique est hors étude. Nous souhaitons cependant pouvoir transiter d'un mode de commande à un autre.

Il est possible, avec des connaissances de base en informatique industrielle, d'apporter des éléments de réponse simples à notre problème. Ils nous permettront d'obtenir un premier système d'ordonnancement des modes de commande, surtout utile pour mener des tests sur la transition et valider d'autres notions fondamentales utiles quel que soit le langage de spécification des comportements utilisé.

L'opérateur décrit le scénario qu'il souhaite élaborer en définissant successivement les états, les événements attendus et les actions à effectuer. Le comportement ainsi défini sera alors stocké dans une base de donnée traduite sous forme d'automate lors de l'exécution du scénario. L'automate choisi est de type *fini* (il est représenté par le bloc *automate fini* du schéma 1.32). Nous nous gardons de donner une définition formelle et adoptons une vision plus empirique donnée par [Wik05] :

Définition 4.1.1 *Un* automate fini *(parfois appelée* machine à états finis*), en anglais* finite state automaton*, ou* finite state machine *(FSA, FSM), est une machine abstraite utilisée en théorie de la calculabilité et des langages formels. Un automate est constitué d'*états *et de* transitions*. Son comportement est dirigé par un mot fourni en entrée : l'automate passe d'état en état, suivant les transitions, à la lecture de chaque lettre de l'entrée. Un automate possède un nombre fini d'états distincts : il ne dispose donc que d'une mémoire bornée.*

Un automate fini forme naturellement un graphe orienté étiqueté, dont les états sont les sommets et les transitions les arêtes étiquetées.

Cette vision *graphique* (dans les sens à la fois mathématique et usuel de ce terme) de l'ordonnancement permet de spécifier de manière simple et rapide un scénario, voir illustration 4.4. Cependant cette manière

[2] C'est-à-dire qu'ils sont capables de décrire beaucoup de cas

de spécifier les comportements, si elle présente les avantages déjà évoqués n'en est pas moins limitée et la puissance d'expression d'un langage tel que les automates finis est assez limitée.

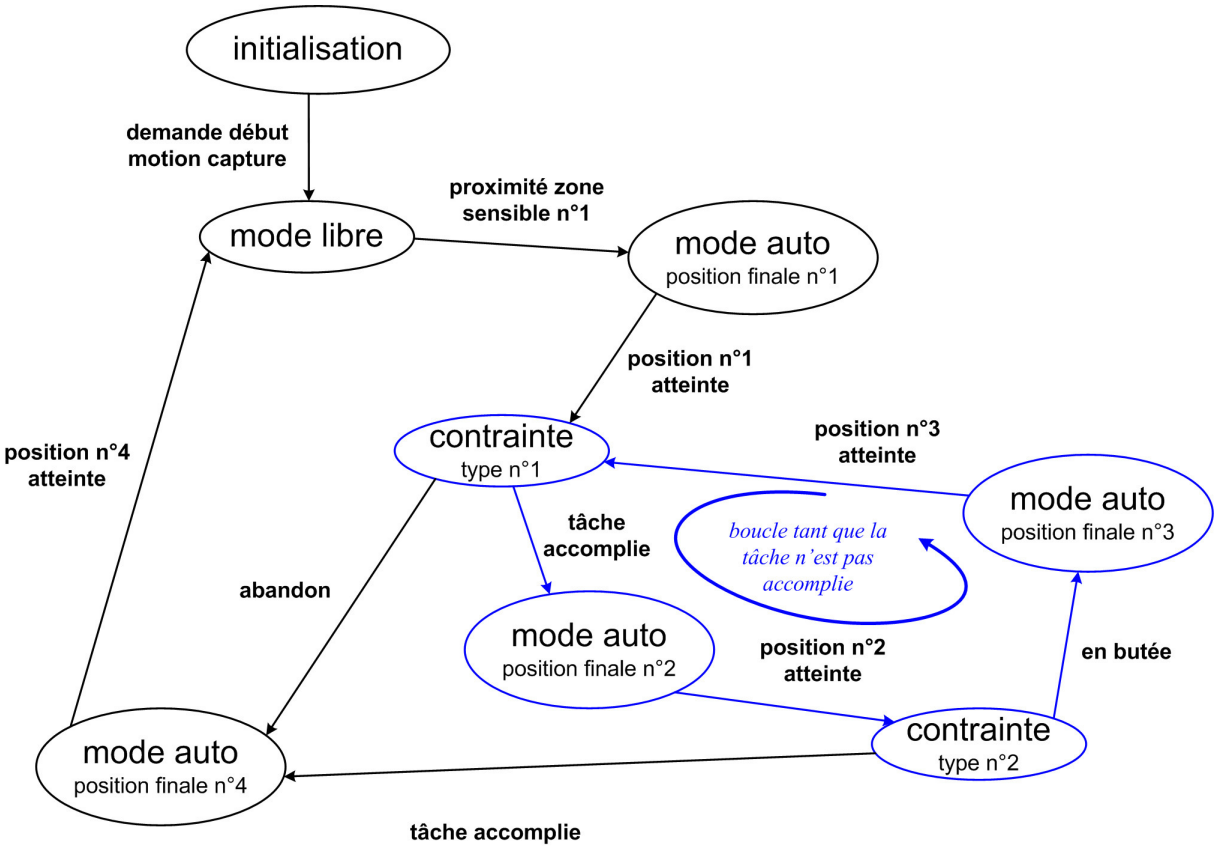

FIG. 4.4 – Exemple de machine d'état pour un mouvement cyclique générique.

Imaginons, par exemple, que notre humain virtuel soit occupé à réaliser une opération donnée et que pendant la simulation, l'opérateur le perturbe en lui demandant de réaliser un autre travail jugé urgent. Si, par la suite, nous souhaitons que l'humain virtuel revienne dans la situation dans laquelle il se trouvait, il faut garder son état en mémoire (on parle de contexte), pendant que l'action plus urgente est réalisée. Pour mettre en place un tel comportement, la mémoire bornée de l'automate fini ne suffit pas. Il faut lui ajouter une mémoire de type *pile*, cette même mémoire que nous trouvons dans les langages évolués comme le *C*. Elle permet de sauvegarder le contexte d'un programme quand on fait appel à une sous-fonction, pour retrouver ensuite le même contexte au retour de la sous-fonction.

Nous savons donc notre langage de scénarisation limité et nous laissons aux spécialistes des langages le soin de l'étendre. Il convient d'ailleurs de noter que ce besoin ayant été identifié, il fait l'objet d'une étude au CEA/LIST [GCL05], qui doit mener à un moteur de scénarisation interactive apte à gérer la majorité de nos besoins.

Récapitulatif du choix du modèle

Ordonnancement des modes de commande par machine d'état

Avantages - inconvénients :

+ simple à coder
+ très simple à utiliser (du fait de son caractère graphique)
− puissance d'expression limitée (limite les possibilités de scénarisation)

Transition de modes et continuité

Les mouvements humains sont fluides. [FH85] insiste sur le caractère lisse des trajectoires des membres des humains. Or pour réaliser complètement une tâche, plusieurs modes de commande sont enchaînés, ordonnancés comme décrit ci-avant. La commutation entre ces différents modes doit donc faire l'objet d'une attention particulière pour obtenir un comportement représentatif des mouvements humains. Nous allons assurer la continuité en vitesse de la position de consigne.

Ce même type de problèmes est rencontré en téléopération. Dans ce cas, [Jol97] propose de bloquer *progressivement* les bras maître et esclave sur une position fixe. Un mode automatique ayant ramené maître et esclave sur la contrainte au préalable. La transition de modes est effectuée seulement après blocage.

Cette approche n'est pas envisageable dans un dispositif de capture de mouvement seul, où aucun appareillage nous permet de bloquer les mouvements de l'acteur sur une position fixe lors de la transition. Nous nous devons de trouver une solution en adéquation avec notre problème spécifique.

Nous avons trois modes de commande, neufs cas de transition peuvent donc se poser. Le mode le moins contraint est le mode libre et le mode le plus contraint, le mode autonome. Nous détaillons maintenant les problèmes qui peuvent se poser.

Des problèmes peuvent surgir lorsque le point de sortie du premier mode est différent du point d'entrée du second. Ce peut être le cas pour des raisons diverses :

– la position de ces points peut être imposée
– le second mode est plus contraint que le premier, donc la position finale du premier mode ne respecte pas la contrainte du second, cet exemple est illustré figure 4.5.
– l'intersection des deux contraintes peut-être vide, donc par construction les positions des deux points sont différentes

On peut alors imaginer de mettre en œuvre une génération de trajectoire qui mènera de la position finale du premier mode à une position d'entrée satisfaisant la contrainte du second mode.

Un autre problème peut être relevé : si une direction précédemment contrainte devient libre, la consigne peut de nouveau être respectée sur cette direction. Un génération de trajectoire sur cette direction permettra alors au système de rejoindre sa position de consigne sur cette direction.

De manière générale, il ne paraît pas envisageable de demander à un opérateur de positionner précisément de manière manuelle (quel que soit le dispositif de spécification du mouvement) le repère de contrôle sur la nouvelle contrainte avant d'effectuer la transition (du fait principalement du manque de précision du contrôle humain, de tremblements...). Cette constatation justifie d'autant plus l'utilisation de générations de trajectoires pour la transition entre modes de commande.

Remarquons que les correcteurs (de type proportionnel-dérivé) ne comportent pas de terme intégral, mais comme le système est du premier ordre, il n'y aura pas d'erreur statique. Information à tempérer en cas contraint, ou si du poids est ajouté.

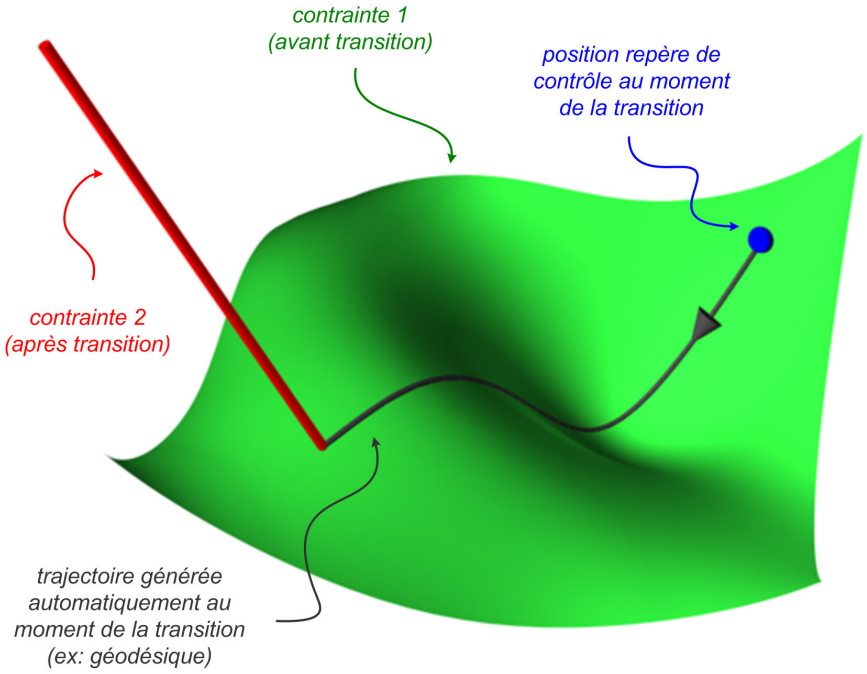

FIG. 4.5 – Au moment où la transition est décidée, le repère de contrôle se situe sur la contrainte n°1, mais ne respecte pas la contrainte n°2. Une trajectoire est générée qui amène le repère de contrôle de la position courante à une position sur la contrainte n°2.

Dans cette section nous avons d'abord mentionné que nous sortions du cadre strict du contrôle de l'humain au sens où nous l'avions envisagé en 2 et 3. Ce contrôle bas niveau *nous autorise trois modes de commande. Nous avons mis en œuvre un* superviseur *plus haut niveau, permettant de les ordonnancer en fonction des besoins d'un scénario de simulation. Nous avons ensuite évoqué les problèmes de continuité que pose la transition en elle-même et proposé des méthodes pour les réduire. Voyons maintenant l'architecture matérielle déployée.*

4.2 Intégration

D'un point de vue matériel, nous avons donc plusieurs machines à gérer, la capture de mouvement (1 PC), le rendu 3D (n PC, avec 1 PC et 1 carte graphique par vue, mais des clusters de cartes graphiques commencent à apparaître, comme la solution développée par TechViz® [Tecb]), la simulation (1 PC, mais les premiers PPU, cartes accélératrices pour la physique, doivent faire leur apparition commerciale sous peu [Age05], [BFG]).

Dans ces conditions, l'idée de clusters distribués de PC s'impose naturellement. Nous connaissons d'ailleurs leurs caractéristiques de flexibilité, modularité, extensibilité largement débattues dans la littérature [TCBV05], [LIO]. Le principe de base est d'ajouter des PC au cluster, pour prendre en charge de nouvelles fonctionnalités au fur et à mesure du développement de la plateforme de réalité virtuelle.

Dans la mesure du possible, on tend à ne pas faire de développement dans ces architectures. C'est-à-dire que la réutilisation de pièces logicielles préexistantes et leur intégration est à la base de la mise en œuvre des clusters. Les technologies du jeu et de la réalité virtuelle (le terme dédié est d'ailleurs *serious game* en anglais [Chaa], [Cha06]) étant relativement proches, nous pouvons utiliser nombre d'outils du premier domaine, très dynamique. En revanche, certaines fonctionnalités sont à apporter à l'existant, ce sont ces cas qui justifient les études du type de celle que nous avons réalisée.

Le schéma suivant permet de se donner une idée concrète du cluster que nous avons mis en œuvre, figure 4.6.

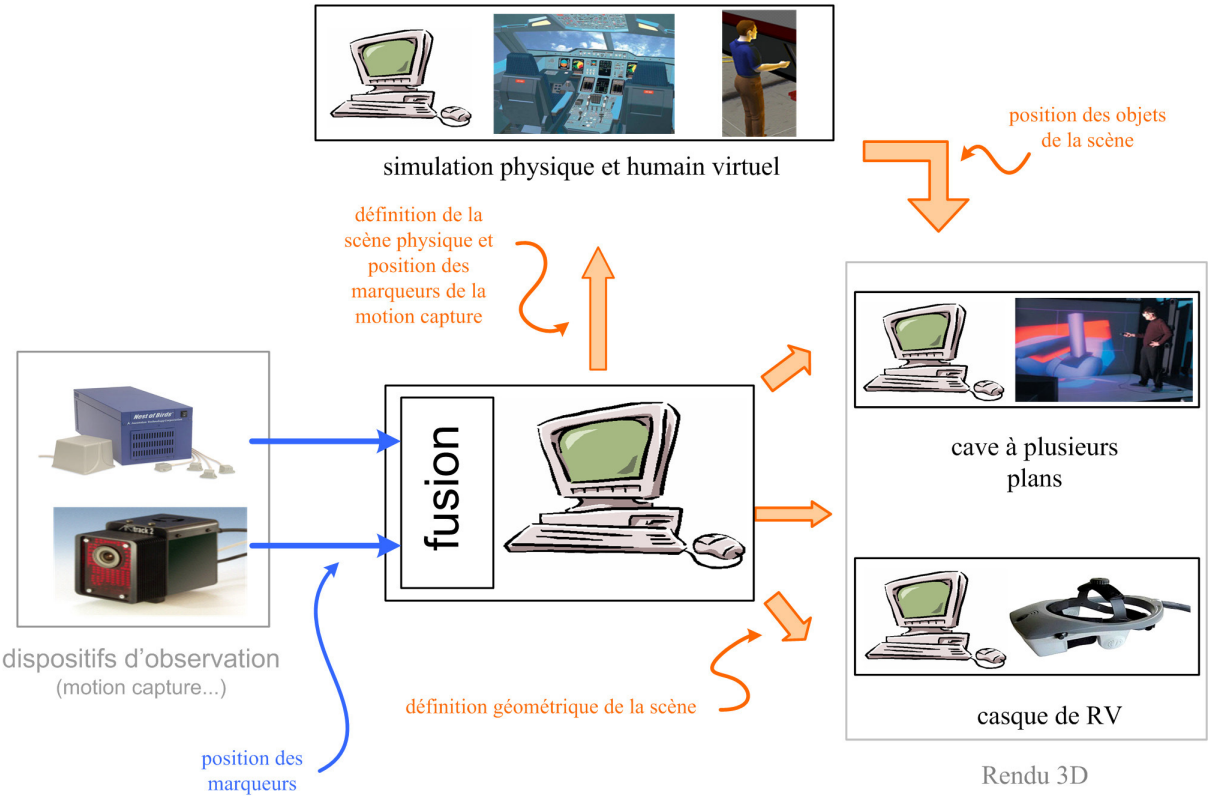

Fig. 4.6 – Schéma du cluster mis en place.

4.3 Applications et performances

4.3.1 Applications

Nous présentons ici des résultats obtenus sur le simulateur temps réel mis en œuvre comme décrit dans cette thèse. Comme indiqué en 1.4.1, l'entrée du simulateur est un dispositif de capture de mouvement. Nous plaçons donc des marqueurs sur un acteur, dont on observe les mouvements. Une photo de capture de mouvement en situation est donnée illustration 4.7.

FIG. 4.7 – Marqueurs destinés à observer les mouvements de l'acteur par capture de mouvement optique.

Notons que dans la majorité des cas, nous avons représenté la géométrie du mannequin servant à la détection de collision - le rendu 3D dit "photoréaliste" (voir figure 4.12) n'étant pas toujours utile à nos tests.

4.3.1.1 Mouvements simples

L'image 4.8, illustre le cas de figure d'un mouvement simple de saut. Comme indiqué, six repères de contrôle sont mis en œuvre ici sur les pieds, les mains, le bassin et la tête. L'acteur s'accroupit puis s'étend vers le haut pour sauter. Ses mouvements sont reproduits sur le mannequin. On visualise sur l'image les positions extrêmes du saut en plein et des positions intermédiaires en transparence.

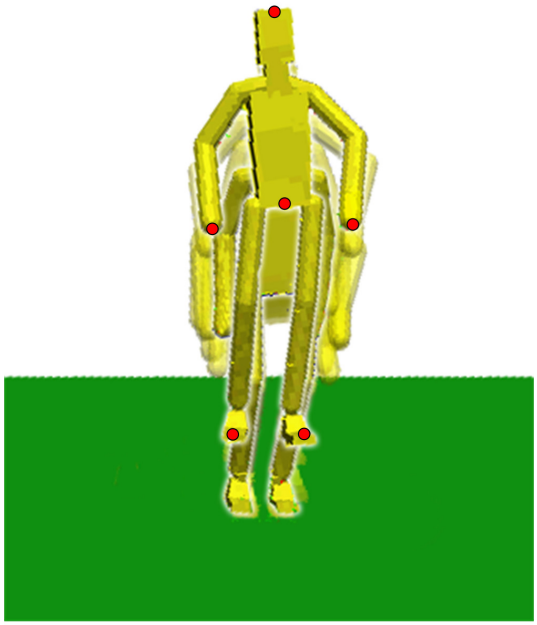

Fig. 4.8 – L'acteur effectue un saut dans le dispositif de capture de mouvement et l'humain virtuel reproduit le saut correspondant. Les repères de contrôle sont indiqués en rouge.

4.3.1.2 Interaction avec une balle

Nous illustrons maintenant l'interaction avec un objet simple. Sur l'image 4.9, une balle tombe du ciel, l'humain virtuel la saisit. Les efforts de contact sont indiquées sous la forme de flèches, dont la taille et la couleur dépendent de la norme de la force. Le mannequin pose ensuite la balle sur la table et joue avec elle. Il la pousse alors par terre pour jouer au pied avec elle.

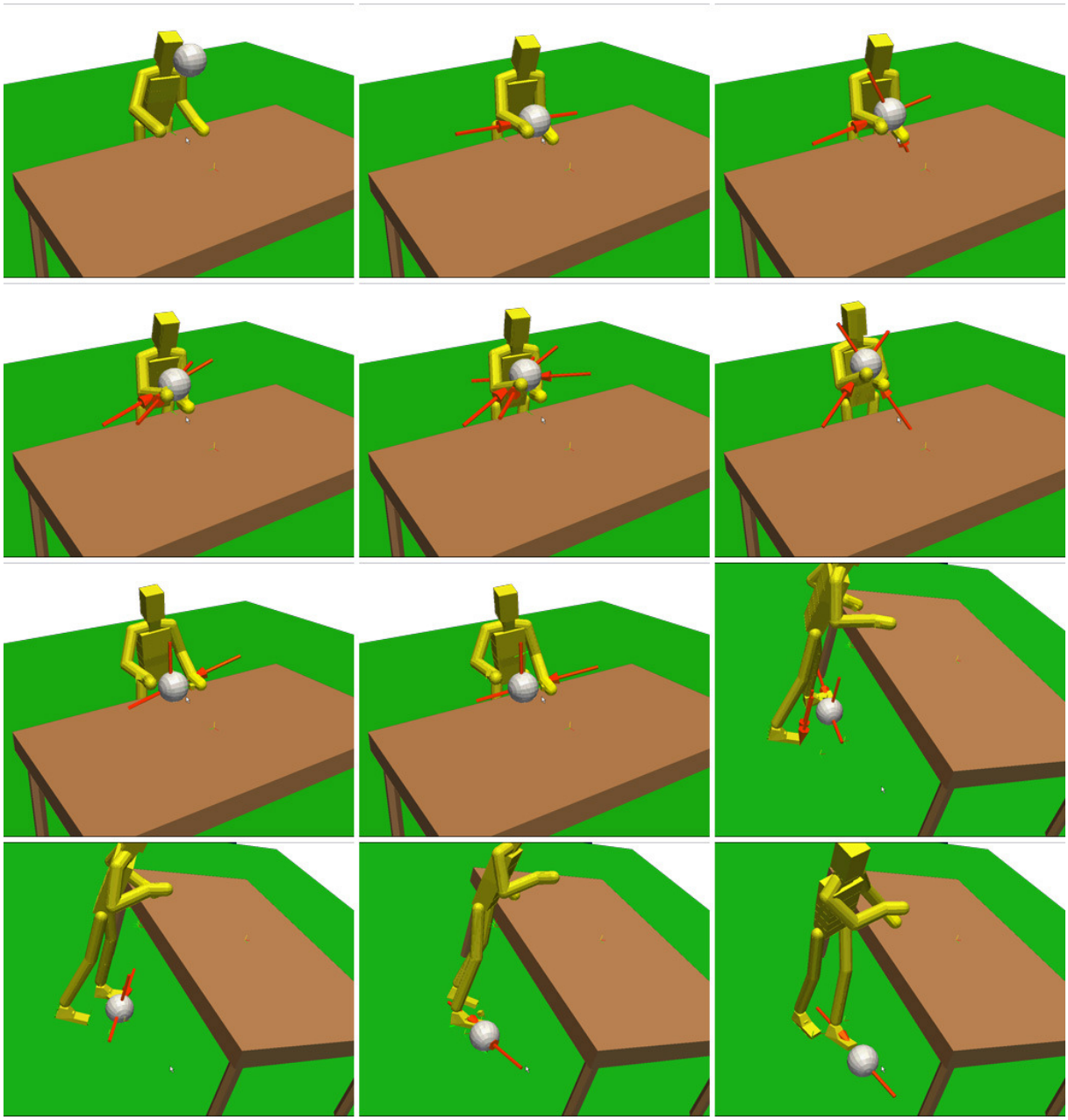

FIG. 4.9 – L'humain virtuel joue avec une balle.

Sur la figure 4.10, la balle est rattrapée en vol sur le pied droit du mannequin qui profite de l'occasion pour jouer au ballon.

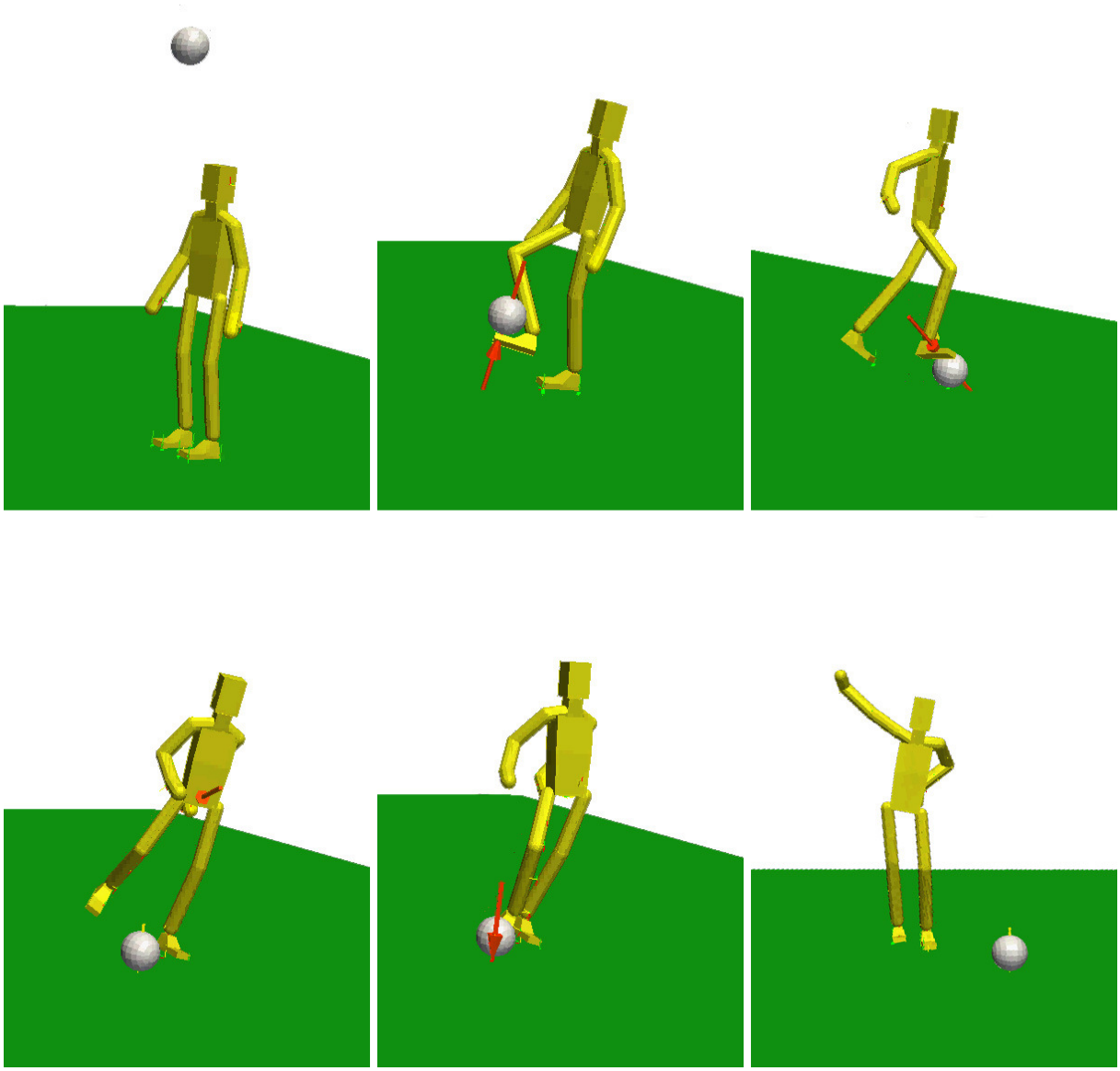

Fig. 4.10 – L'humain virtuel joue au pied avec une balle.

4.3.1.3 Deux mannequins

L'illustration n°4.11 montre deux humains virtuels soulevant un piano. Chaque mannequin est piloté par le biais de capture de mouvement, dispositif dans lequel deux acteurs sont immergés. Quatre repères de contrôle sont mis en place, un sur chaque main des mannequins.

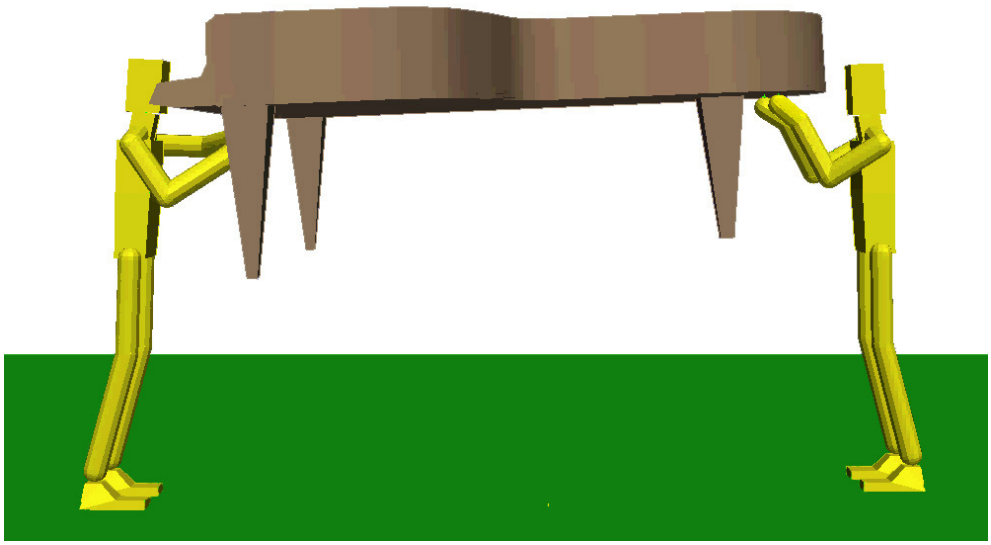

FIG. 4.11 – Deux mannequins manipulant un piano.

4.3.1.4 Fonctionnalités composées

Dans cette sous-section, nous complexifions les tâches. Un test de perçage est réalisé figure 4.12. Après avoir saisi la perceuse dans la main droite et une lampe torche dans la main gauche. L'humain virtuel effectue la tâche de perçage qui lui est assignée. La position du trou est indiquée précisément par le disque noir figure de gauche. Pour percer exactement à l'endroit indiqué, une assistance est mise en œuvre par le biais d'un mécanisme virtuel, qui guide l'axe de la perceuse sur l'axe désiré du trou (le mécanisme virtuel utilisé est décrit figure 2.20).

Le mannequin s'éclaire grâce à la lampe torche. Pour s'assurer que son faisceau pointe toujours vers le trou, une seconde assistance est mise en œuvre. Deux mécanismes virtuels sont donc utilisés simultanément.

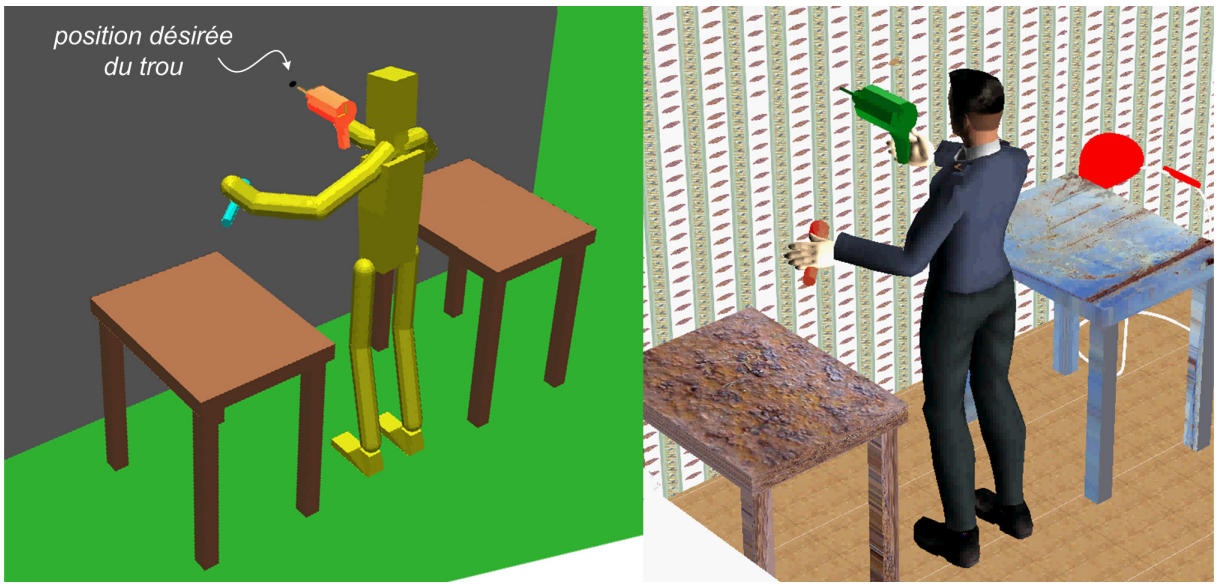

Fig. 4.12 – Humain virtuel réalisant une tâche de perçage, avec assistance. Modèle physique à gauche et couche additionnelle de visualisation à droite.

L'illustration donnée figure 4.13, montre un garçon de café virtuel tenant un plateau dont la bonne orientation est, là encore, assurée par une assistance. Il ouvre la porte de sa main libre, passe au travers, puis nous salue une fois sa tâche accomplie.

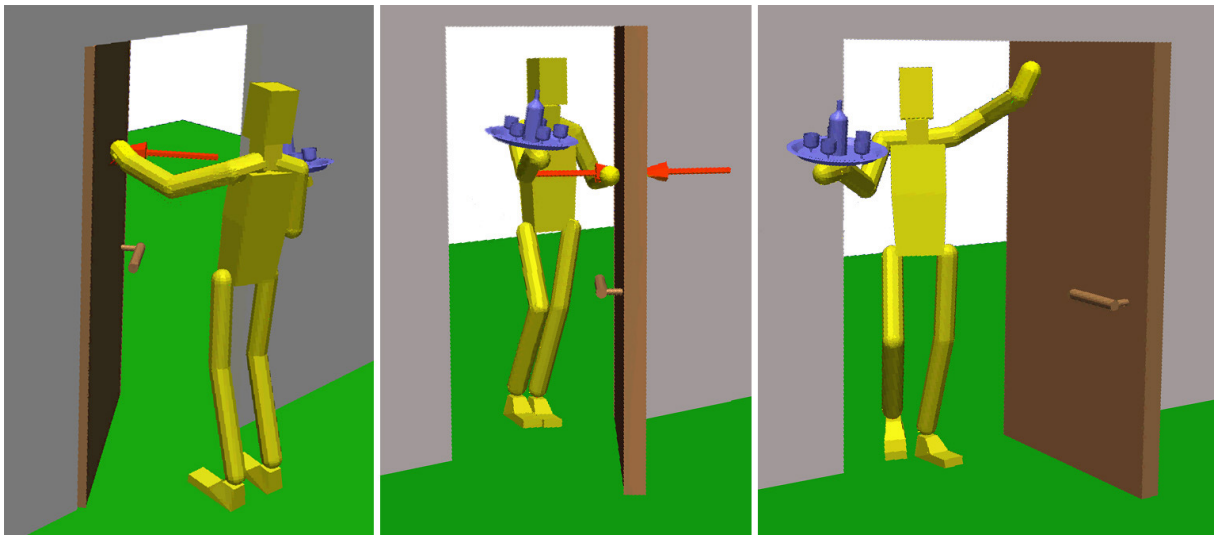

Fig. 4.13 – Garçon de café virtuel.

4.3.1.5 Cas industriel

Les cas illustrés dans cette section sont à caractère industriel.

L'application de la figure 4.14 a pour finalité d'analyser l'ergonomie d'un cockpit d'A400M®, avion de transport militaire d'Airbus®, en phase de développement[3]. La maquette visualisée ici a été fragmentée

[3]Les premières livraisons sont prévues pour 2007.

et simplifiée, de manière à satisfaire les contraintes de la détection de collision utilisée. On voit à droite les positions initiales et finales de la manette des gaz. A gauche, on visualise la tâche en cours de visualisation et les efforts de contact calculés. Notons la complexité du démonstrateur : les efforts d'interaction principaux sont visualisés par la flèche rouge, mais énormément de forces de contact secondaires sont calculées, elles sont visualisées sous la forme de petites flèches vertes, figure de gauche.

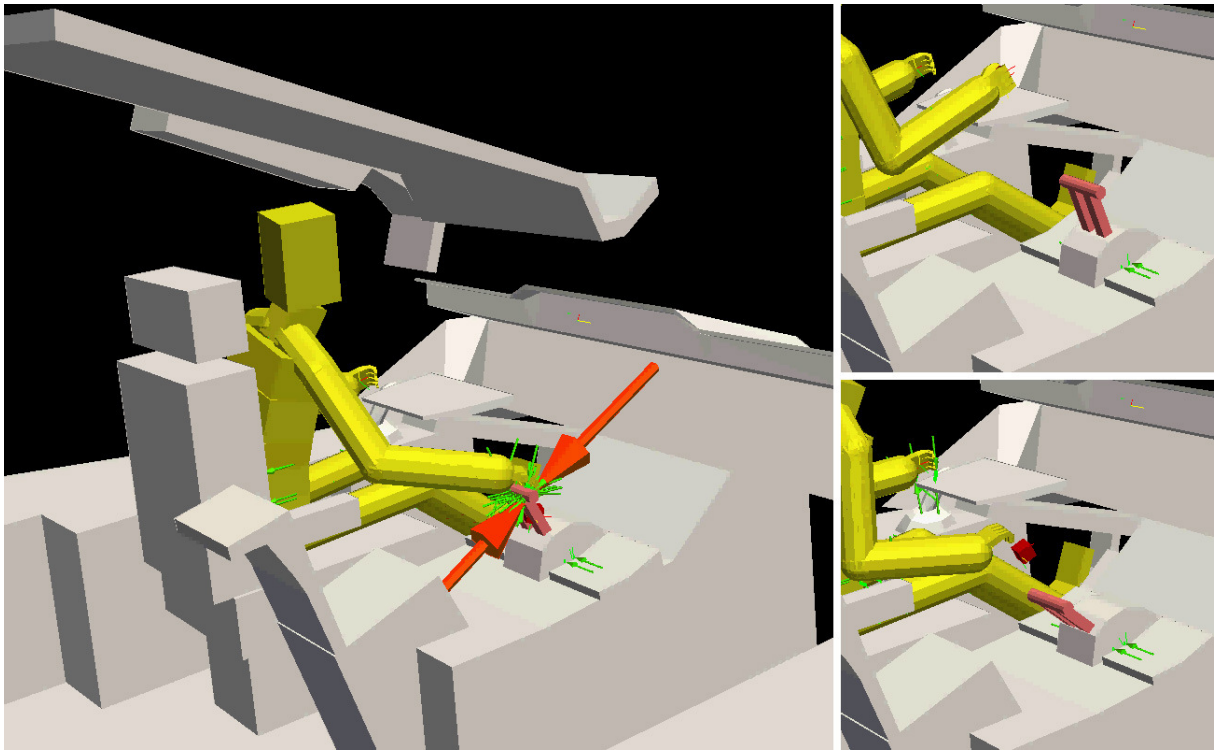

FIG. 4.14 – Pilote virtuel manipulant la manette des gaz d'un Airbus A400M®.

L'application figures 4.15 et 4.16 met en oeuvre un rotor de Super Puma®, hélicoptère développé par la société Eurocopter®. La tâche impartie est le positionnement de la pièce blanche (illustration 4.16) entre deux pâles de rotor. Cette tâche requière une certaine précision pour l'alignement des axes des trous d'une part, et de la goupille de fixation d'autre part (cerclés de rouge figure 4.15). Cette précision est difficilement accessible sans information kinesthésique, comme c'est le cas sans système haptique (voir 4.1.1). Pour palier ce problème, à l'approche de la configuration désirée, un premier mécanisme virtuel est mis en oeuvre de manière à guider la pièce blanche, imposant l'alignement des axes. Un second mécanisme virtuel guide simultanément la goupille en translation sur l'axe du trou.

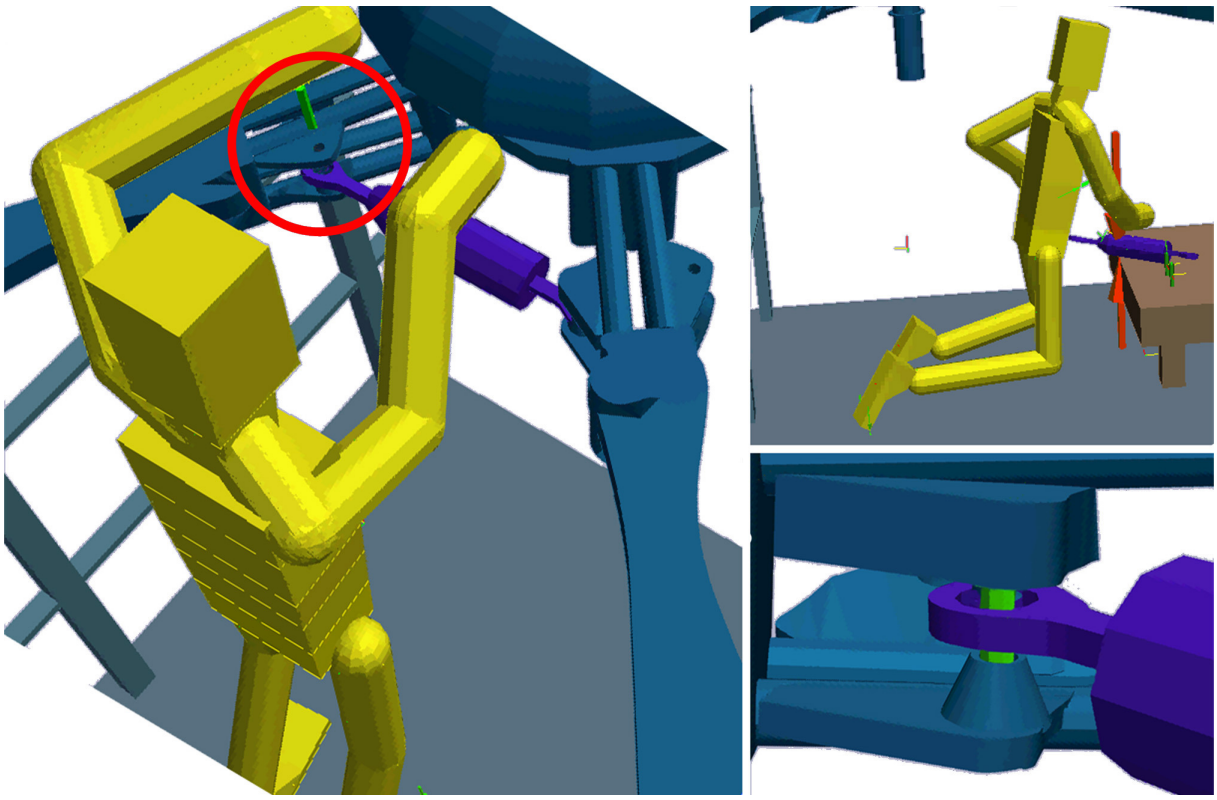

Fig. 4.15 – Technicien virtuel effectuant une opération de maintenance sur un Rotor de Super Puma®. Visualisation de la géométrie utilisée pour la détection de collision.

Fig. 4.16 – Technicien virtuel effectuant une opération de maintenance sur un Rotor de Super Puma®. Visualisation “photoréaliste”.

4.3.2 Performances

Sur la courbe 4.17, nous montrons l’efficacité du solveur de contraintes dans le cas d’un mannequin posant la main sur une table. Dans un premier temps, le mannequin évolue dans l’espace libre. Ensuite la consigne amène le mannequin a poser la main sur la table. Dans ce cas le solveur de contraintes

unilatérales développé en 3 intervient de manière à faire respecter la contrainte de non pénétration de l'environnement.

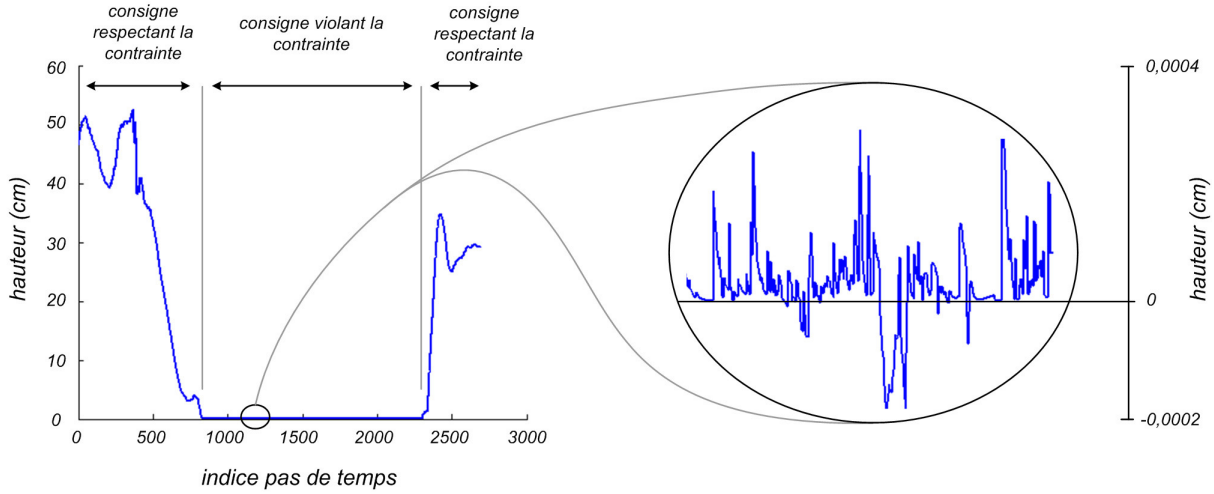

FIG. 4.17 – Hauteur de la main. La main évolue d'abord dans l'espace libre, puis vient se poser sur la table de hauteur 0. Dans un premier temps, la consigne respecte la contrainte. On spécifie alors une consigne violant la contrainte : dans ce cas le solveur de contraintes empêche la main de pénétrer la table.

La courbe 4.17 indique que le solveur de contraintes fonctionne de manière satisfaisante. On constate cependant sur le grossissement (à droite), des micro-pénétrations. Celles-ci peuvent être attribuées à la linéarisation intervenant dans l'équation (3.7). La méthode de détection de collision utilisée, basée sur les travaux de D.E. Johnson [JW04] est inefficace dans le cas où les objets s'interpénètrent. Nous sommes donc amenés à mettre en œuvre une marge entre la maille de l'objet et la *coque* qui va effectivement servir à la collision. La détection de collision renvoie une distance $d_{collision}$ entre objets, la distance effectivement fournie est $d^{modifiee}_{collision} = d_{collision} - 2 * marges$. Ainsi, comme illustré sur la figure 4.18, s'il y a interpénétration sur la géométrie avec les marges bien dimensionnées, il n'y a pas interpénétration sur la géométrie manipulée par la détection de collision. En pratique la valeur de la marge dépendra de la scène simulée et des vitesses de calcul - elle peut être petite au point d'être imperceptible.

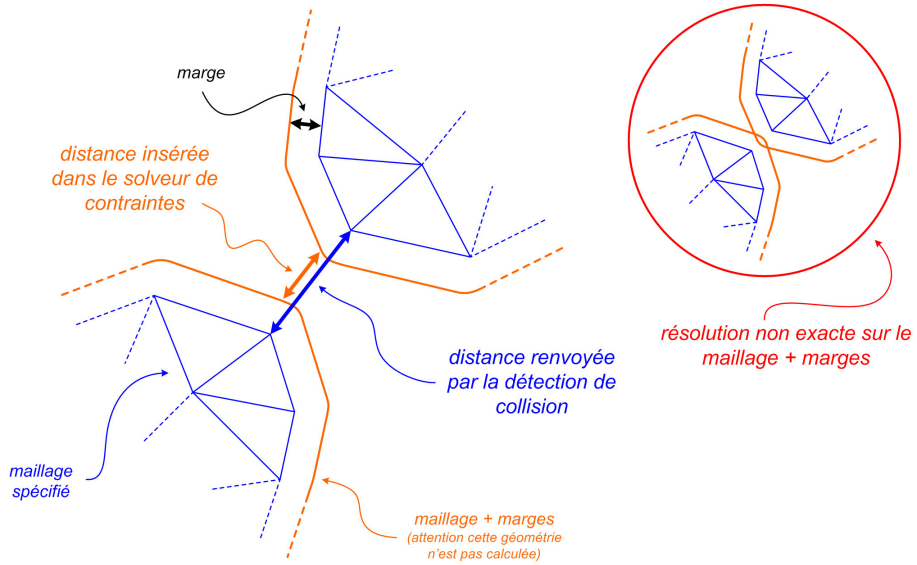

Fig. 4.18 – Ajout de marges de manière à gérer les cas de résolution non exacte du contact.

Nous testons maintenant les guides virtuels, mis en œuvre par la notion de mécanismes virtuels décrite en 2.2.2.2. L'expérience effectuée est décrite schéma 2.20 et consiste à guider une perceuse sur l'axe de la position désirée d'un trou. La contrainte laisse un seul degré de liberté en translation à l'opérateur. Sur la figure 4.19, on a représenté l'angle entre l'axe idéal et l'axe courant, avec et sans assistance.

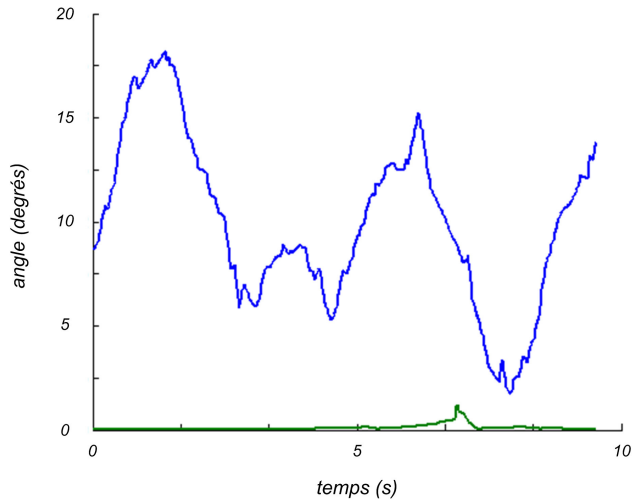

Fig. 4.19 – Angle entre l'axe idéal du trou et l'axe courant de la perceuse, en vert avec assistance, en bleu sans.

Les courbes suivantes 4.20 et 4.21 montrent les performances temporelles de l'architecture développée. Un saut dans le nombre de contacts montre que les performances sont extrêmement dépendantes du nombre de contact dans la simulation. Sur 4.20, on voit que le temps correspondant aux blocs *contrôle et simulation* (c'est le temps de calcul du bloc vert, marqué *contrôle bas niveau*, du schéma 1.26) varie énormément avec le contact. Après le saut, environ 280 contacts sont à gérer, la courbe verte montre que la simulation et le contrôle nécessitent environ 100ms par cycle, auxquels il faut ajouter le temps de la détection de collision (en bleu) : on s'éloigne du temps-réel. Ce cas est choisi pour son caractère extrême,

dans les tests réalisés en 4.3.1, le caractère temps-réel des simulation est conservé.

En comparant les courbes 4.21 et 4.20, on s'aperçoit que le temps de calcul du bloc *contrôle et simulation* est presque entièrement dévolu à la résolution du contact après le saut. C'est la réaction attendue dans ce cas.

Notons également sur 4.20 que la détection de collision prend un temps non négligeable. Sur des scènes complexes, comme c'est le cas des géométries à caractère industriel (voir illustration 4.14), elle peut complètement conditionner les performances de la simulation. Dans ces cas, il faut porter une attention particulière à la géométrie donnée en entrée de la détection de collision. En effet, les performances peuvent dépendre énormément de la manière avec laquelle la géométrie est fragmentée. Ainsi, si elles sont "mieux maillées" (ex : facettes régulières...), des géométries complexes peuvent parfois prendre moins de temps à gérer que des géométries plus simples. Il peut alors être intéressant de mettre en œuvre des méthodologies de gestion de la fragmentation (notion plus connue sous son appellation anglaise *tessellation*) [DLCG05].

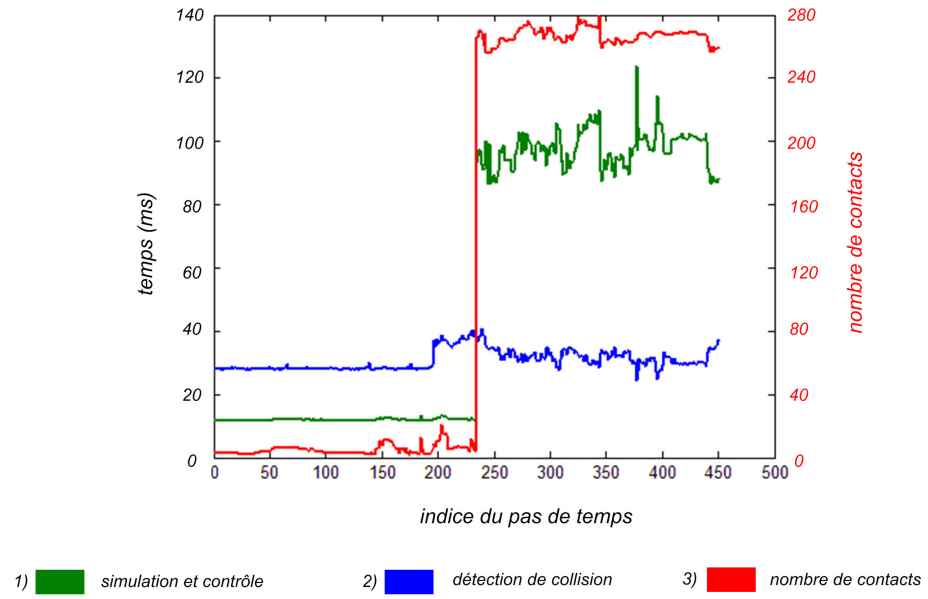

FIG. 4.20 – Courbes indiquant les temps de calcul par cycle pour les blocs *contrôle et simulation* de l'architecture. On impose un saut dans le nombre de contact lors du test.

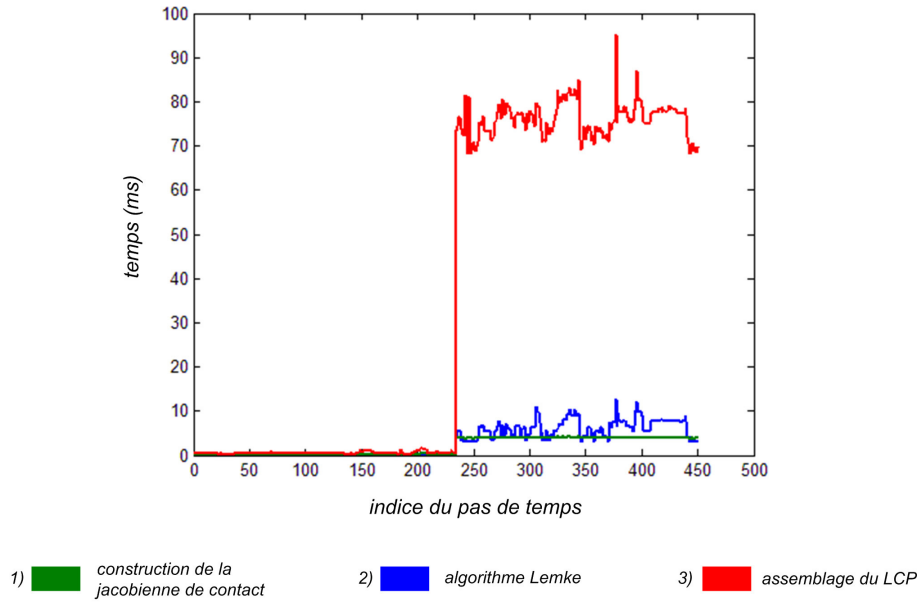

Fig. 4.21 – Courbes indiquant les temps de calcul par cycle des algorithmes liés au contact. Ces temps de calculs sont inclus dans la courbe verte *contrôle et simulation* de la figure 4.20 .

La courbe 4.22, montre les performances du contrôleur d'équilibre dans le cas des contacts coplanaires, développé en 3.3.2.1. Dans le cas de la courbe verte, le mannequin, sans contrôleur, est parfois amené à violer la contraintes (cas où $d(q) < 0$, avec $d(q)$ défini en (3.22)). Dans le cas de la courbe bleue, l'équilibre est toujours respecté grâce au contrôleur d'équilibre.

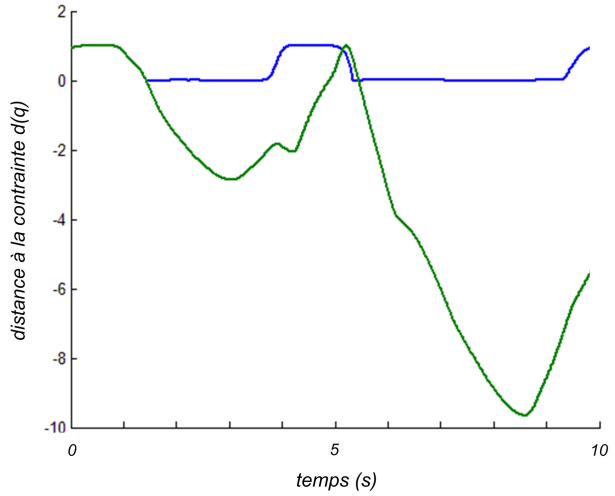

Fig. 4.22 – En vert, distance à la contrainte sans contrôleur d'équilibre, $d(q) < 0$ marque la perte d'équilibre. En bleu, distance à la contrainte avec le contrôleur d'équilibre, $d(q) \geq 0$.

D'un point de vue performances temporelles, le contrôleur d'équilibre dans le cas des contacts coplanaires, correspond à l'ajout d'une dimension au LCP. Il est donc léger en temps de calculs.

Les performances temporelles du contrôleur d'équilibre dans le cas des contacts non coplanaires, développé en 3.3.2.2 n'ont pas été testées. Rien n'assure donc pour le moment qu'il puisse être implémenté

en temps réel. En revanche, comme indiqué équation (3.70), la taille du LCP est donnée par $m_c * m_a$, nombre de contacts multiplié par le nombre d'arêtes pas contact. Pour un nombre "raisonnable" d'arêtes par contact (3 ou 4), on peut espérer une résolution temps réel pour quelques dizaines de contacts. Ce qui nous autorisera à mettre en œuvre des cas relativement intéressants.

4.4 Perspectives

Nous discutons ici des perspectives de travaux envisageables.

4.4.1 Formulation d'un contrôle interne dans le contrôleur d'équilibre

Dans la formulation variationnelle du contrôleur d'équilibre (3.107), nous avons choisi un critère d'optimisation qui minimise les déplacements articulaires locaux. C'est-à-dire que nous avons fait un choix pour la résolution de la redondance du mannequin. Il est possible de poser l'optimisation de manière à formuler un véritable contrôle interne dans l'optimisation.

Reprenons le principe du contrôle interne décrit en 2.2.1.4. L'erreur entre la configuration q du mannequin et une configuration q_d désirée, est minimisée. La correction désirée est donc donnée par $\delta q_d = q_d - q$. Dans le cadre du contrôleur donné (3.107), nous pouvons changer le critère de minimisation des déplacements articulaires, en un critère minimisant l'erreur δq_d de la manière suivante :

$$\boxed{\left\{ \begin{array}{l} \min_X \left(\left\| X - \begin{bmatrix} f_{c_d}^{dis} \\ \beta \delta q_d \end{bmatrix} \right\|_{Q_1}^2 + \alpha \left\| \left(I - NN^t\right) AX \right\|_{Q_2}^2 \right) \\ \quad \text{sachant} \left\{ \begin{array}{c} G'' X = W'' \\ \begin{bmatrix} I_f \\ I_q \\ -I_q \end{bmatrix} X \geq \begin{bmatrix} 0 \\ \delta q_{min} \\ -\delta q_{max} \end{bmatrix} \end{array} \right. \end{array} \right.}, \tag{4.1}$$

avec β paramètre de réglage du contrôle interne.

Avec cette possibilité de contrôle interne additionnelle, le problème résolu par (4.1) est plus complexe que la simple résolution de l'équilibre, puisque l'on peut accomplir du contrôle de tâches (voir 3.3.2.2) et du contrôle interne.

4.4.2 Intégration du contrôleur d'équilibre à notre architecture

Le résultat de l'optimisation (3.107) est un δq. Or la simulation (bloc bleu du schéma 1.31) prend des couples articulaires en entrée. Il pourrait donc être intéressant de reformuler l'optimisation de manière à obtenir ces couples articulaires, ou de mettre en œuvre un contrôleur articulaire permettant de générer les couples articulaires permettant d'obtenir les δq. Ce type de contrôleur a été décrit en 1.4.2.1.

4.4.3 Paradoxe contact - contrôle

Les ports d'interaction externes envisagés dans cette thèse sont de deux types différents. On rencontre :

- **les repères de contrôle :** dont les caractéristiques (position du couplage, comportement...) sont parfaitement maîtrisées et déterminées dans une phase préalable à la simulation

– **les contacts :** qui apparaissent en cours de simulation

Ces ports donnent naissance à des forces de natures différentes. L'une va servir au contrôle et l'autre au respect des contraintes. Dans une simulation, ces types de forces peuvent rentre en conflit, si un solide est à la fois en contact avec l'environnement virtuel et asservi en position. Ces conflits apparaissent notamment en cas d'erreur de mise en correspondance entre le monde réel et le monde virtuel. Or ces erreurs, aussi infimes soient-elles, sont inévitables, même en mettant en œuvre une étape de calibration de bonne qualité.

Prenons l'exemple d'un pied asservi en position pour être capable de reproduire des mouvements de marche et dont les contacts avec le sol sont pris en compte dans le contrôleur d'équilibre. Du fait des erreurs de calibration, le contrôle peut positionner le pied virtuel légèrement en retrait du sol dans le monde virtuel, alors qu'il le touche dans le monde réel, comme indiqué figure 4.23. Dans ce cas, la détection de collision ne relève pas de contact au niveau du pied, aussi l'appui au sol sur le pied en question n'est pas pris en compte dans le contrôleur d'équilibre, alors que les appuis au sol sont primordiaux (on peut voir sur les figures 3.16 et 3.17 que les forces de contact au sol sont importantes).

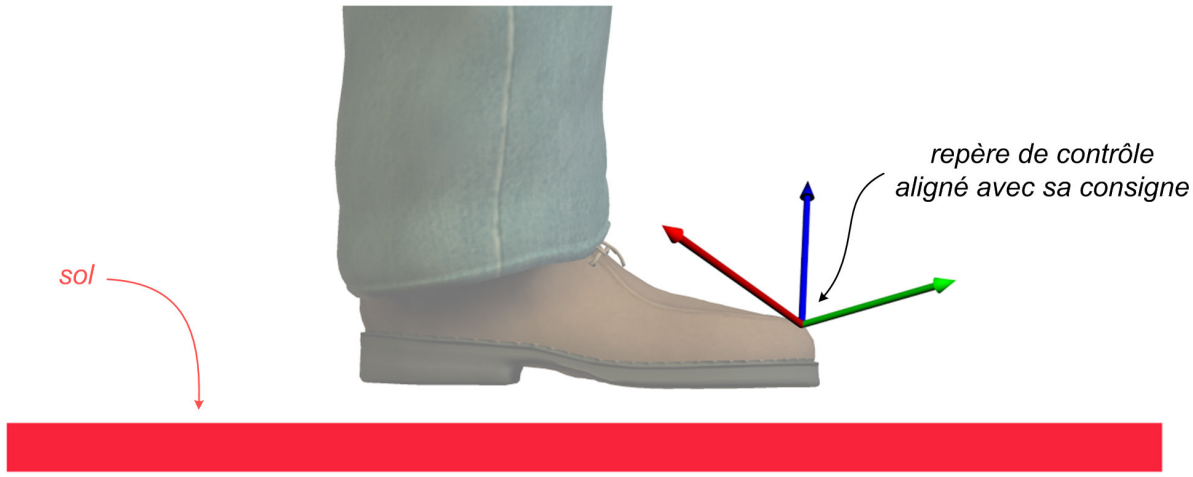

Fig. 4.23 – Le pied est asservi en position avec une erreur nulle. Des erreurs de calibration impliquent le non contact entre le pied et le sol (alors que le pied réel est bien en contact avec le sol).

Cette remarque est un des obstacles majeurs à l'implémentation du contrôleur tel qu'il est exprimé équation (3.107).

4.4.4 Enrichissement haptique

L'introduction de l'haptique va permettre à l'utilisateur de ressentir les contraintes d'un point de vue kinesthésique : les forces générées devront empêcher l'acteur de **violer ces contraintes dans le monde réel**. Cela vaut pour les guides, mais également pour les limites articulaires, ou le contact : c'est une information très riche qui est restituée par la sensation kinesthésique. L'intérêt de l'haptique, notamment dans les tâches de maintenance qui nous préoccupent, a fait l'objet de publications de l'équipe de Norman Badler [BDR+03], ou d'autres laboratoires [AKH01].

L'haptique peut se superposer à tous les modes de commande. Mais pour ce faire il nous faut coupler l'organe haptique à la simulation. Notre architecture mettant déjà en œuvre la notion de force, cet interfaçage se fera de manière naturelle, les forces de consigne à appliquer à l'organe haptique étant déjà calculées par la simulation.

Le couplage à l'haptique présente deux problèmes majeurs :
- le Virtuose® - dispositif haptique que nous souhaitons mettre en œuvre - peut être vu comme une inertie m soumise à un frottement visqueux de paramètre b. Il a donc un comportement dit en *admittance* Y_h : c'est-à-dire que des forces W_h sont appliquées en entrée du dispositif qui fournit un mouvement V_h en correspondance [AH99]. Il doit être couplé au simulateur qui a également un comportement d'admittance, comme indiqué figure 4.24. Le lien rouge de la figure est problématique car le dispositif haptique et le monde virtuel prennent tous les deux en entrée des efforts W_d et W_e et renvoient des vitesses V_d et V_e et renvoient des déplacements, sans pour autant qu'il y ait de sous-système générant ces efforts.
- une fois couplé, la discrétisation peut impliquer des problèmes de stabilité. Elle peut, dans certains cas, introduire de manière artificielle, de l'énergie dans le système. Or nous savons (d'après 2.1) que ce comportement nous amène à perdre la passivité.

FIG. 4.24 – Schéma simpliste du couplage d'un dispositif haptique à la simulation.

Ces deux problèmes sont résolus dans [AH99], en introduisant une impédance de couplage Z_{cp}, comme indiqué sur le schéma 4.25.

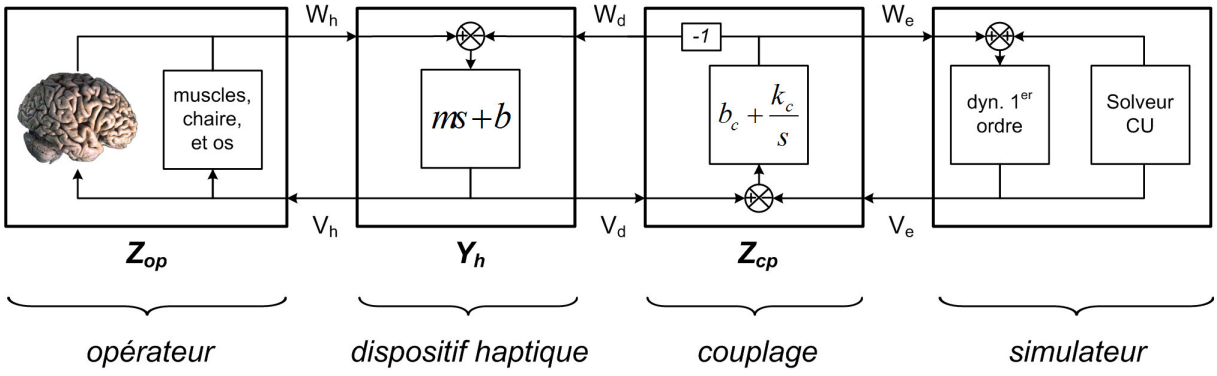

FIG. 4.25 – Schéma de fonctionnement du couplage haptique.

Notons que cette impédance de couplage Z_{cp} va perturber quelque peu le comportement désiré du bras, car le couplage peut être vu comme une compliance locale.

4.4.5 Points divers laissés en suspens

Des points ont été laissés en suspens au cours de la thèse, nous les regroupons ici.

- Les rotules ont été remplacées par trois pivot concourantes et orthogonales. Nous proposons une paramétrisation et une formulation des limites sur les rotules. Celle-ci est placée en annexe D, du fait de son caractère très calculatoire.

- La méthode d'intégration mise en œuvre est de type Euler-explicite, il existe d'autres méthodes plus performantes, elles pourraient être mise à profit dans le cas de l'humain virtuel. Nous décrivons brièvement quelques méthodes d'intégration sur espaces vectoriels et leur extension aux groupes de Lie en annexe C.
- En 2.2.1.4, nous avons proposé des méthodes permettant le contrôle interne de manière passive. Nous avons notamment décrit l'introduction de projecteurs constants, permettant un découplage local à un point q_0 de l'espace de configuration. Nous estimons envisageable d'étendre la méthode, en calculant des projecteurs locaux en différents points de l'espace de configuration.
- Nous avons souligné en 2.2.2.2, la probabilité de l'existence d'une possibilité d'assistance à la création de mécanismes virtuels en vue du découplage passif de tâches. Cette voie reste entièrement à explorer.
- Les possibilités du contrôle haut niveau ont été envisagées en 1.3. Il n'a été qu'effleuré en 4.
- ...

Conclusion

Dans cette thèse nous avons posé les fondations d'une architecture d'animation d'humains virtuels, dont les performances autorisent une interaction temps-réel. Elle pourra donc être utilisée de manière à immerger un acteur dans un monde virtuel. L'intérêt majeur de notre approche est qu'elle autorise l'interaction avec l'environnement. Cette interaction est réalisée de manière naturelle par le biais du même phénomène rencontré dans la nature : les forces.

L'architecture est basée sur un comportement différentiel du premier ordre, où un frottement visqueux articulaire remplace la notion d'inertie du deuxième ordre. Ceci nous permet de minimiser les problèmes de réglage de la simulation et autorise quand même l'introduction de la notion d'effort - vecteur naturel de l'interaction. Nous plaçons ensuite un étage d'intégration sur la variété de configuration qui permet de parvenir à la position opérationnelle des membres du squelette. Les correcteurs sont de simples proportionnels-dérivés et sont exprimés dans l'espace de la tâche. L'introduction de contrôleurs plus complexes doit être considérée avec attention pour ne pas entraver la bonne marche du système, notamment en terme de stabilité. Nous adjoignons enfin un correcteur dit interne pour contrôler les degrés de libertés de la redondance laissés en boucle ouverte par le contrôle externe.

Les capacités d'interaction recherchées nous ont amené à analyser le comportement énergétique aux contacts avec l'environnement, par le biais de la notion de passivité. Leur analyse a montré que des précautions devaient être prises quand des opérations artificielles comme les projections, ou les discrétisations sont introduites. Dans le cas général, les projections violent la propriété de passivité. Ceci nous a amené a envisager des modes de commande contraints par des projections passives grâce à la notion de mécanisme virtuel. Ceux-ci permettent de restreindre les possibilités de mouvement du mannequin de manière à ce que les projections telles qu'elles étaient envisagées auparavant deviennent moins pertinentes. Nous avons noté que l'absence de certaines sensations, comme le sens kinesthésique, pouvait entraver la bonne conduite d'une simulation. Ces mécanismes virtuels vont également pouvoir servir de guides lorsque la perception de l'opérateur est dégradée.

Nous avons également abordé le domaine des contrôleurs d'équilibre, dans le cas des contacts coplanaires, dans un premier temps - où ils sont envisagés dans la majorité des travaux précédents. Puis nous avons souligné que dans le cas de l'interaction la position des contacts n'était pas prédéterminée, aussi nous avons établi les lignes directrices d'un contrôleur d'équilibre dans le cas non-coplanaire. Nous avons remarqué que tous les degrés de liberté du système pouvaient avoir une influence sur son équilibre, aussi nous avons abordé le problème statique dans toute sa complexité. C'est un problème redondant, nous l'avons posé sous la forme d'un problème d'optimisation quadratique, résolu sous la forme d'un LCP, propice à une implémentation temps-réel.

La composante bas niveau de notre contrôleur en place, nous avons relevé trois modes de commande possibles des repères de contrôle : libre, contraint et autonome. Nous avons proposé une méthode simple permettant leur ordonnancement basée sur un automate fini. Nous nous sommes alors concentrés sur les problèmes de continuité liés à la transition entre les différents modes de commande.

Toute la construction de notre architecture ayant été envisagée sous l'angle de la passivité (du moins en temps continu), nous savons donc notre système modulaire tant d'un point de vue fonctionnel, que du point de vue de la stabilité. Malheureusement, cette passivité est conditionnelle, les conditions dépendent du pas de temps d'intégration et ce pas de temps est contraint pas le caractère temps-réel de nos applications. Cela se traduit par le fait que des efforts limités doivent être appliqués pour rester passif. Pour obtenir une simulation réaliste il faut alors régler tous les paramètres en conséquence et c'est là une des limitations de notre système. Pour augmenter les efforts maximaux admissibles nous pouvons attendre l'augmentation des vitesses de calcul des ordinateurs pour diminuer le pas de temps d'intégration. De manière plus active, nous pouvons également mettre en œuvre des schémas d'intégration qui préservent mieux l'énergie. Nous avons testé certains de ces schémas sans pousser leur implémentation jusqu'à leur accomplissement complet dans notre architecture.

L'architecture proposée apporte une première solution à notre besoin, mais de nombreuses améliorations pourraient lui être apportées, nous n'avons d'ailleurs fait qu'effleurer certains sujets. Il ne faut donc pas envisager notre travail comme une étude aboutie et close, mais plutôt comme une opportunité de pousser plus avant encore les recherches. Sur le schéma d'architecture, nous voyons que nos préoccupations pour cette thèse ont surtout concerné le contrôle bas niveau. Du contrôle plus haut niveau a été envisagé, avec la génération de trajectoire et l'ordonnancement. Tous deux présentent des approches simplistes plus destinées à valider nos résultats qu'à réellement apporter des solutions pertinentes. Il reste bien sûr énormément de points à améliorer dans notre contrôleur bas niveau, nous pourrons citer l'intégration, l'amélioration de la gestion de la passivité dans le cas de la projection de l'influence du potentiel interne, l'équilibre uniquement géré dans le cas statique... Mais l'architecture implémentée apporte une première solution au problème, les développements à venir porteront aussi sur le haut niveau, délaissé jusqu'ici et pour cause, il aurait été de peu d'utilité sans un bas niveau relativement fiable.

Annexes

A Reformulation d'une optimisation quadratique en LCP

Le problème d'optimisation quadratique suivant peut être réécrit sous forme d'un LCP [CPR92]. Le problème d'optimisation générique est donné par

$$\min_X \frac{1}{2}\|AX+b\|_Q^2, \text{ avec } \ker(A)=\{0\} \quad (2)$$
$$\text{sachant } \begin{cases} CX=d, \text{ avec } C\in G(n,m),\ m\geq n, \text{ et } C \text{ de rang plein} \\ EX\leq f \end{cases}$$

Exprimons le Lagrangien du problème d'optimisation

$$L=\frac{1}{2}\|AX+b\|_Q^2+\lambda^t(CX-d)+\mu^t(EX-f),\ \mu\geq 0 \quad (3)$$

alors on sait que

$$\frac{\partial L}{\partial X}=\frac{1}{2}(AX+b)^t(Q+Q^t)A+\lambda^t C+\mu^t E=0 \quad (4)$$

donc

$$A^tQ(AX+b)+C^t\lambda+E^t\mu=0, \text{ car Q symétrique} \quad (5)$$

ainsi

$$X=-Q_A^{-1}(C^t\lambda+E^t\mu+A^tQb), \text{ avec } Q_A=A^tQA,\ A \text{ de rang plein} \quad (6)$$

alors

$$CX=-CQ_A^{-1}C^t\lambda-CQ_A^{-1}E^t\mu-CQ_A^{-1}A^tQb \quad (7)$$

et comme $C\in G(n,m)$, $m\geq n$, et C de rang plein, $(CQ_A^{-1}C^t)^{-1}$ est définie, donc

$$\lambda=-(CQ_A^{-1}C^t)^{-1}(CX+CQ_A^{-1}E^t\mu+CQ_A^{-1}A^tQb) \quad (8)$$

(8), dans (6) :

$$X - Q_A^{-1}C^t(CQ_A^{-1}C^t)^{-1}(CX + CQ_A^{-1}E^t\mu + CQ_A^{-1}A^tQb) + Q_A^{-1}E^t\mu + Q_A^{-1}A^tQb = 0 \tag{9}$$

en regroupant les termes :

$$\left(I - Q_A^{-1}C^t(CQ_A^{-1}C^t)^{-1}C\right)X + \left(I - Q_A^{-1}C^t(CQ_A^{-1}C^t)^{-1}C\right)Q_A^{-1}E^t\mu + \left(I - Q_A^{-1}C^t(CQ_A^{-1}C^t)^{-1}C\right)Q_A^{-1}A^tQb = 0 \tag{10}$$

posons $\Pi = \left(I - Q_A^{-1}C^t(CQ_A^{-1}C^t)^{-1}C\right)$:

$$\Pi X + \Pi E^t\mu + \Pi Q_A^{-1}A^tQb = 0 \tag{11}$$

or $CX = d$, donc :

$$\begin{bmatrix} \Pi \\ C \end{bmatrix} X + \begin{bmatrix} \Pi Q_A^{-1}E^t \\ 0 \end{bmatrix} \mu = \begin{bmatrix} -\Pi Q_A^{-1}A^tQb \\ d \end{bmatrix} X \tag{12}$$

que l'on multiplie à droite et à gauche par $\begin{bmatrix} \Pi & Q_A^{-1}C^t(CQ_A^{-1}C^t)^{-1} \end{bmatrix}$:

$$(\Pi^2 + Q_A^{-1}C^t(CQ_A^{-1}C^t)^{-1}C)X + \Pi^2Q_A^{-1}E^t\mu = -\Pi^2Q_A^{-1}A^tQb + Q_A^{-1}C^t(CQ_A^{-1}C^t)^{-1}d \tag{13}$$

or Π étant un projecteur, $\Pi^2 = \Pi$:

$$X = -\Pi Q_A^{-1}E^t\mu - \Pi Q_A^{-1}A^tQb + Q_A^{-1}C^t(CQ_A^{-1}C^t)^{-1}d \tag{14}$$

donc

$$f - EX = E\Pi Q_A^{-1}E^t\mu + E\Pi Q_A^{-1}A^tQb - EQ_A^{-1}C^t(CQ_A^{-1}C^t)^{-1}d + f \tag{15}$$

le LCP s'exprimera donc de la manière suivante :

$$0 \leq E\Pi Q_A^{-1}E^t\mu + E\Pi Q_A^{-1}A^tQb - EQ_A^{-1}C^t(CQ_A^{-1}C^t)^{-1}d + f \perp \mu \geq 0. \tag{16}$$

B Réécriture de la somme de deux normes quadratiques

$$\|A_1X + b_1\|^2_{Q_1} + \alpha\|A_2X + b_2\|^2_{Q_2} = (A_1X + b_1)^tQ_1(A_1X + b_1) + \alpha(A_2X + b_2)^tQ_2(A_2X + b_2) \tag{17}$$

$$= \left[\begin{bmatrix} A_1 \\ A_2 \end{bmatrix} X + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}\right]^t \begin{bmatrix} Q_1 & 0 \\ 0 & \alpha Q_2 \end{bmatrix} \left[\begin{bmatrix} A_1 \\ A_2 \end{bmatrix} X + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}\right] \tag{18}$$

$$= (AX + b)^tQ(AX + b) \tag{19}$$

avec

$$A = \begin{bmatrix} A_1 \\ A_2 \end{bmatrix} \qquad b = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \qquad Q = \begin{bmatrix} Q_1 & 0 \\ 0 & \alpha Q_2 \end{bmatrix} \tag{20}$$

C Méthodes d'intégration

Le problèmes à résoudre ici est d'intégrer l'équation

$$\dot{x} = f(x). \tag{21}$$

On a déjà vu l'intérêt de la discrétisation. On distingue deux grandes familles de schémas d'intégration numériques listées ci-dessous :

– les **schémas explicites :** qui calculent l'état au pas suivant uniquement à partie d'informations disponibles au pas courant. On pourra donner l'exemple de *Euler explicite* :

$$\frac{x_{k+1} - x_k}{\delta t} \simeq \dot{x}_k = f(x_k), \tag{22}$$

d'*Adams*, ou de *Runge-Kutta* [Hil02]. Ces méthodes sont les plus utilisées, car elles sont les plus intuitives et les plus faciles à mettre en œuvre. Elles peuvent cependant conduire à des instabilités en cas de dynamiques rapides.

– les **schémas implicites :** qui calculent l'état au pas suivant à partir des informations disponibles au pas courant, mais également d'informations qui ne seront disponibles qu'au pas suivant. On pourra donner l'exemple de *Euler implicite* :

$$\frac{x_{k+1} - x_k}{\delta t} \simeq \dot{x}_{k+1} = f(x_{k+1}), \tag{23}$$

de la *θ-méthode*, ou du schéma de *Newmark* [Hil02]. De manière générale, un schéma implicite ne peut être résolu au sens strict, car on perd la causalité[4]. On peut alors mettre en place une prédiction de x_{k+1}, puis une linéarisation de f autour de x_k, dans un schéma appelé *Euler implicite linéarisé* :

$$\frac{x_{k+1} - x_k}{\delta t} \simeq \dot{x}_{k+1} \simeq f(x_k + \dot{x}_k \delta t) \tag{24}$$

$$\simeq f(x_k) + \frac{\partial f}{\partial x}(x_k)\dot{x}_k \delta t. \tag{25}$$

Les méthodes implicites sont bien souvent non-linéaires, donc plus complexes à mettre en œuvre, mais sont en contrepartie plus stables.

Les méthodes citées précédemment permettent d'intégrer sur des espaces vectoriels. Si on souhaite intégrer sur des espaces courbes (comme c'est le cas de $SE(3)$, variété dans laquelle évoluent les positions de nos solides), on utilise des méthodes d'intégration sur groupes de Lie. La plus évidente est celle de Crouch-Grossman [CG93] qui remplace l'accroissement infinitésimal sur les espaces vectoriels $x_{k+1} = x_k + \dot{x}\delta t$, par son équivalent sur les groupes de Lie $x_{k+1} = \exp^{(\dot{x}\delta t)} x_k$. Munthe-Kaas [MK98] transforme le problème courbe en un problème plan, en paramétrant le groupe de Lie par son algèbre associée, sur laquelle il intègre, la méthode est étendue aux ordres plus élevés dans [CO03].

D Jacobiennes des limites sur les rotules

Nous décrivons ici la méthode de paramétrage et de limitation des rotules, mise en œuvre.

D.1 Paramétrage d'une rotule

Pour le paramétrage des rotules, nous passons par le formalisme des groupes de Lie, en utilisant l'exponentielle définie sur $SO(3)$. L'intérêt, le principe et les défauts de l'approche sont décrits dans [Gra98]. La rotation ${}^{j}R_i$ du repère associé au solide j au repère associé à i est donc paramétrée par $\omega_{i/j(i)} \in so(3)$, vitesse de i par rapport à j projetée et réduite dans i :

$${}^{j}R_i = {}^{j}R_i(t) \tag{26}$$

$$= \exp\left(\omega_{i/j(i)}\right){}^{j}R_i(0) \tag{27}$$

Notons que le paramétrage mis en œuvre ne correspond pas à une contrainte bilatérale.

[4]Les informations à l'instant $k+1$ ne peuvent être disponibles à l'instant k puisqu'on est en train de les calculer.

D.2 Formulation des limites sur les rotules

Nous avons montré en 3.2.2, comment la gestion des limites sur les liaisons pivot était effectuée. Le cas des liaisons rotules est plus complexe. Le modèle des limites que nous avons développé est basé sur la décomposition *swing/twist* issue de [Bae01]. Le repère d'indice 0 est lié au solide parent et celui d'indice 1 au solide dont nous voulons contraindre le mouvement. Nous choisissons d'exprimer les limites par rapport à l'axe z. Pour toutes les grandeurs, si la base de projection n'est pas précisée, il s'agit de 0. Comme indiqué illustration 26, la décomposition de la rotation ${}^0R_1$ est alors la suivante :

- le *swing* ${}^0Rs$ est la rotation géodésique sur la sphère S_2 qui mène z_0 en z_1
- le *twist* ${}^sR_1$ est la rotation complémentaire autour de z_1

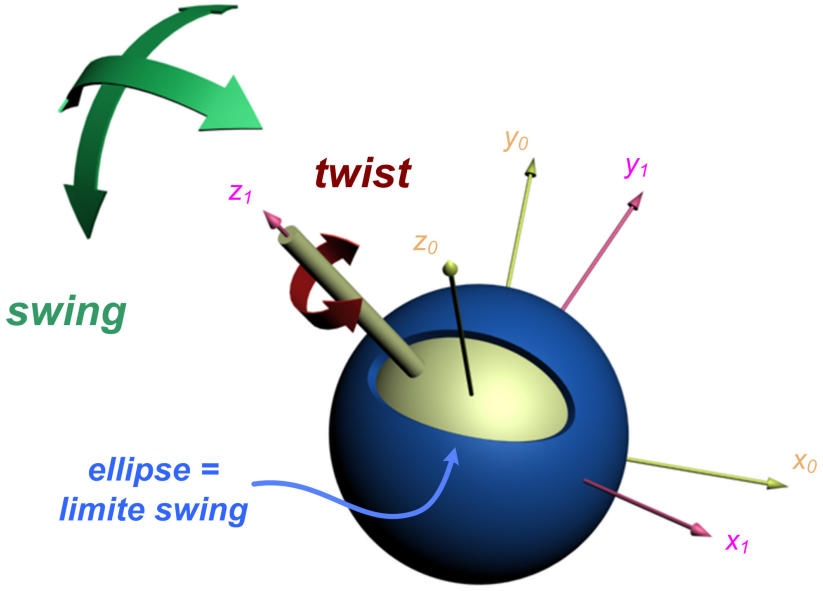

FIG. 26 – Décomposition de la rotation d'une rotule en *swing* et *twist*, de manière à pouvoir exprimer les limites de manière intuitive.

Nous écrivons

$$
\begin{aligned}
{}^0R_1 &= R && (28)\\
&= {}^0R_s\,{}^sR_1 && (29)\\
&= R_sR_t, \text{ avec } R_s = {}^0R_s \text{ et } R_t = {}^sR_1 && (30)\\
&= \exp([r]), \text{ avec } [r] = \log(R) && (31)\\
&= \exp([r_s])\exp([r_t]), \text{ avec } [r_s] = \log(R_s) \text{ et } [r_t] = \log(R_t) && (32)\\
&\neq \exp([r_s] + [r_t]) && (33)
\end{aligned}
$$

car $\exp(A)\exp(B) = \exp(A+B)$ ssi A et B commutent, c'est-à-dire $[r_s][r_t] - [r_t][r_s] = 0$, ce qui est vrai ssi $r_s // r_t$, or par construction $r_s \perp r_t$. Cette constatation a des implications importantes, cela signifie que le twist dépend du swing, c'est-à-dire que swing et twist ne commutent pas. Cette décomposition ne fonctionne donc pas de manière complètement intuitive. En revanche c'est une méthode simple et facile à manipuler pour spécifier les limites sur les rotules et c'est là son intérêt.

Nous donnons ici les distances aux limites pour les swing et twist, ainsi que leur jacobienne.

Limite du swing

${}^0R_s$ est la rotation géodésique amenant z_0 à z_1, les grandeurs sont exprimées dans la base d'indice 0, aussi

$$r_s = \frac{z_0 \wedge z_1}{\|z_0 \wedge z_1\|} \arccos\left(z_0 \cdot z_1\right) \tag{34}$$

$$= \frac{\theta_s}{\sin\left(\theta_s\right)} z_0 \wedge z_1. \tag{35}$$

Nous allons contraindre r_s à rester à l'intérieur d'une ellipse Q, c'est-à-dire

$$\|r_s\|_Q^2 \leq d_0^2, \tag{36}$$

pour cela nous posons la distance quadratique d_s, à la violation de la limite, définie comme suit :

$$\boxed{d_s = d_0^2 - \|r_s\|_Q^2} \tag{37}$$

nous imposerons alors $d_s \geq 0$. Sa jacobienne J_s, est définie de telle manière que :

$$\delta d_s = J_s \delta\omega_{1/0(1)}, \tag{38}$$

nous rappelons que le préfixe δ signifie "petite quantité" au sens physique et que remplacer une différentielle totale d par un δ revient à effectuer une approximation linéaire motivée par l'hypothèse des petits déplacements. Exprimons δd_s, sachant $d_s = d_0^2 - \|r_s\|_Q^2$:

$$\delta d_s = -\delta\|r_s\|_Q^2 \tag{39}$$

$$= -\delta r_s^t Q r_s \tag{40}$$

$$= -r_s^t (Q + Q^t)\delta r_s \tag{41}$$

en prenant en compte (35), on a

$$\delta r_{s(0)} = \delta\left(\frac{\theta_s}{\sin\left(\theta_s\right)}\right)[z_0]z_1 + \frac{\theta_s}{\sin\left(\theta_s\right)}\delta\left(z_0 \wedge z_1\right) \tag{42}$$

or

$$\delta\left(\frac{\theta_s}{\sin\left(\theta_s\right)}\right) = \frac{\delta(\theta_s)\sin\left(\theta_s\right) - \theta_s\delta\sin\left(\theta_s\right)}{\sin\left(\theta_s\right)^2} \tag{43}$$

avec

$$\delta\theta_s = -\frac{\delta\cos\theta_s}{\sin\left(\theta_s\right)} \tag{44}$$

$$= -\frac{\delta(z_0 \cdot z_1)}{\sin\left(\theta_s\right)} \tag{45}$$

$$= -\frac{z_0 \cdot \delta z_1}{\sin\left(\theta_s\right)} \tag{46}$$

$$\delta z_1 = \delta\Big({}^0R_1 z_{1(1)}\Big) = \delta\Big({}^0R_1 z_{0(0)}\Big) \tag{47}$$

$$= \delta\Big({}^0R_1\Big) z_0 \tag{48}$$

$$= {}^0R_1\left[\delta\omega_{1/0(1)}\right] z_0 \tag{49}$$

$$= -{}^0R_1[z_0]\delta\omega_{1/0(1)} \tag{50}$$

$$\tag{51}$$

donc

$$\delta\theta_s = \frac{z_0 \cdot {}^0R_1[z_0]\delta\omega_{1/0(1)}}{\sin(\theta_s)} \tag{52}$$

et

$$\delta\sin(\theta_s) = \cos(\theta_s)\delta\theta_s \tag{53}$$
$$= \text{cotan}(\theta_s)\Big(z_0 \cdot {}^0R_1[z_0]\Big)\delta\omega_{1/0(1)} \tag{54}$$

ainsi on a

$$\delta\left(\frac{\theta_s}{\sin(\theta_s)}\right) = \frac{z_0^{t\,0}R_1[z_0]\delta\omega_{1/0(1)}\big(1-\theta_s\text{cotan}(\theta_s)\big)}{\sin(\theta_s)^2} \tag{55}$$
$$= \frac{\sin(\theta_s)-\theta_s\cos(\theta_s)}{\sin(\theta_s)^3}z_0^{t\,0}R_1[z_0]\delta\omega_{1/0(1)} \tag{56}$$

et pour $\delta\big(z_0 \wedge z_1\big)$

$$\delta\big(z_0 \wedge z_1\big) = \delta\Big([z_0]z_1\Big) = [z_0]\delta(z_1) \tag{57}$$
$$= -[z_0]^0R_1[z_0]\delta\omega_{1/0(1)} \tag{58}$$

donc

$$\delta r_s = \delta\left(\frac{\theta_s}{\sin(\theta_s)}\right)[z_0]z_1 + \frac{\theta_s}{\sin(\theta_s)}\delta\big(z_0 \wedge z_1\big) \tag{59}$$
$$= \frac{\sin(\theta_s)-\theta_s\cos(\theta_s)}{\sin(\theta_s)^3}z_0^{t\,0}R_1[z_0]\delta\omega_{1/0(1)}[z_0]z_1 - \frac{\theta_s}{\sin(\theta_s)}[z_0]^0R_1[z_0]\delta\omega_{1/0(1)} \tag{60}$$
$$= [z_0]\left(\frac{\sin(\theta_s)-\theta_s\cos(\theta_s)}{\sin(\theta_s)^3}z_1z_0^t - \frac{\theta_s}{\sin(\theta_s)}I_3\right)^0R_1[z_0]\delta\omega_{1/0(1)} \tag{61}$$

que l'on écrit $\delta r_s = J_{r_s}\delta\omega_{1/0(0)}$, en posant $J_{r_s} = [z_0]\left(\frac{\sin(\theta_s)-\theta_s\cos(\theta_s)}{\sin(\theta_s)^3}z_1z_0^t - \frac{\theta_s}{\sin(\theta_s)}I_3\right)^0R_1[z_0]$. On identifie J_s grâce à (38) :

$$\boxed{J_s = -2r_s^tQ[z_0]\left(\frac{\sin(\theta_s)-\theta_s\cos(\theta_s)}{\sin(\theta_s)^3}z_1z_0^t - \frac{\theta_s}{\sin(\theta_s)}I_3\right)^0R_1[z_0].} \tag{62}$$

Quand $\sin(\theta_s) \to 0$ on est en cas singulier, pour $\theta_s \in]-\pi,\pi]$, on a donc une singularité en 0 et une en π.

Singularité en 0

Quand $\theta_s \to 0$ on peut écrire

$$r_s = \frac{\theta_s}{\sin(\theta_s)}z_0 \wedge z_1 \tag{63}$$
$$\simeq \frac{\theta_s}{\theta_s}z_0 \wedge z_1 \tag{64}$$
$$\simeq z_0 \wedge z_1 \tag{65}$$

ainsi

$$\begin{aligned}\delta r_s &\simeq \delta(z_0 \wedge z_1) && (66)\\ &\simeq [z_0]\delta z_1 && (67)\\ &\simeq -[z_0]^0 R_1 [z_0]\delta\omega_{1/0(1)} && (68)\\ & && (69)\end{aligned}$$

donc

$$J_s = 2r_s^t Q[z_0]^0 R_1[z_0] \tag{70}$$

Singularité en π

On ne peut rien faire, la représentation *axe-angle* des rotations admet une singularité en π [Moa02]. Et c'est une vraie singularité.

Limite du twist

Pour le cas du twist, nous exprimons les limites de la même manière que pour le cas de l'articulation de type pivot :

$$\begin{aligned}R_t &= \exp([r_t]) && (71)\\ &= \exp\Big(\frac{r_t}{\|r_t\|}\theta_t\Big), \text{ avec } \theta_t = \|r_t\|, && (72)\end{aligned}$$

θ_t est contraint par

$$\theta_t \in [\theta_{min}, \theta_{max}], \tag{73}$$

la distance $d_t \geq 0$ à la violation de la contrainte est donc définie par

$$\boxed{d_t = \begin{bmatrix}\theta_{max} - \theta_t \\ \theta_t - \theta_{min}\end{bmatrix},} \tag{74}$$

sa jacobienne est définie de telle manière que :

$$\delta d_t = J_t \delta\omega_{1/0(1)}, \tag{75}$$

et grâce à (74) on écrit

$$\delta d_t = \begin{bmatrix}-\delta\theta_t \\ \\ \delta\theta_t\end{bmatrix} \tag{76}$$

or

$$\begin{aligned}\delta\theta_t &= \delta\|r_t\| && (77)\\ &= \frac{r_t^t}{\|r_t\|}\delta r_t && (78)\end{aligned}$$

On remarque que $\|r_t\|$ n'est pas différentiable en 0. Mais on peut faire l'approximation

$$\|r_t\| \simeq \frac{\|r_t\|^2}{\sqrt{\|r_t\|^2 + \epsilon^2}}, \tag{79}$$

ce qui correspond à "arrondir la singularité" pour la rendre différentiable, l'arrondi étant réglé par la variable infinitésimale ϵ. Dans ce cas

$$\delta\theta_t = \delta\|r_t\| \simeq \frac{\|r_t\|^2}{\sqrt{\|r_t\|^2+\epsilon^2}} \tag{80}$$

$$\simeq \frac{\delta\Big(\|r_t\|^2\Big)\sqrt{\|r_t\|^2+\epsilon^2} - \|r_t\|^2\delta\sqrt{\|r_t\|^2+\epsilon^2}}{\|r_t\|^2+\epsilon^2} \tag{81}$$

$$\simeq \frac{2\|r_t\|\delta\big(\|r_t\|\big)\sqrt{\|r_t\|^2+\epsilon^2} - \|r_t\|^2\delta\Big(\|r_t\|^2+\epsilon^2\Big)\frac{1}{2\sqrt{\|r_t\|^2+\epsilon^2}}}{\|r_t\|^2+\epsilon^2} \tag{82}$$

$$\simeq \frac{2\|r_t\|\frac{r_t^t}{\|r_t\|}\sqrt{\|r_t\|^2+\epsilon^2}\delta r_t - 2\|r_t\|^2\|r_t\|\frac{r_t^t}{\|r_t\|}\delta r_t\frac{1}{2\sqrt{\|r_t\|^2+\epsilon^2}}}{\|r_t\|^2+\epsilon^2} \tag{83}$$

$$\simeq \frac{2r_t^t\Big(\|r_t\|^2+\epsilon^2\Big)\delta r_t - \|r_t\|^2 r_t^t\delta r_t}{\big(\|r_t\|^2+\epsilon^2\big)^{\frac{3}{2}}} \tag{84}$$

$$\simeq \frac{\big(\|r_t\|^2+\epsilon^2\big)r_t^t}{\big(\|r_t\|^2+\epsilon^2\big)^{\frac{3}{2}}}\delta r_t. \tag{85}$$

D'après [IMKNZ00], on a

$$\dot{R}_t = [dexp(r_t)\dot{r_t}]\,R_t \tag{86}$$

$$= R_t\,[dexp(-r_t)\dot{r_t}]\,, \tag{87}$$

avec $dexp(r)$, la différentielle de l'exponentielle, dont une forme close est exprimée dans [IMKNZ00]. En posant $r = \overline{[r]}$ (la barre est l'opération de passage de la notation matricielle à la notation vectorielle des éléments de $\mathfrak{se}(3)$) :

$$\delta r_t = dexp\,(-r_t)^{-1}\,\overline{R_t^t\delta R_t} \tag{88}$$

$$= dexp\,(-r_t)^{-1}\,\overline{{}^sR_1{}^t\delta{}^sR_1} \tag{89}$$

$$= dexp\,(-r_t)^{-1}\,\overline{{}^1R_0{}^0R_s\delta\big({}^sR_0{}^0R_1\big)} \tag{90}$$

$$= dexp\,(-r_t)^{-1}\,\overline{{}^1R_0{}^0R_s\big(\delta({}^sR_0)^0R_1 + {}^sR_0\delta({}^0R_1)\big)} \tag{91}$$

$$= dexp\,(-r_t)^{-1}\,\overline{{}^1R_0{}^0R_s\delta({}^sR_0)^0R_1 + {}^1R_0{}^0R_s{}^sR_0\delta({}^0R_1)} \tag{92}$$

$$= dexp\,(-r_t)^{-1}\,\overline{{}^1R_0\left[\delta\omega_{0/s(0)}\right]{}^0R_1 + \left[\delta\omega_{1/0(1)}\right]} \tag{93}$$

$$= dexp\,(-r_t)^{-1}\,\overline{\left[\delta\omega_{1/0(1)}\right] - \left[{}^1R_0\delta\omega_{s/0(0)}\right]} \tag{94}$$

$$= dexp\,(-r_t)^{-1}\left(\delta\omega_{1/0(1)} - {}^1R_0dexp\,(r_s)\,\delta r_s\right) \tag{95}$$

$$= dexp\,(-r_t)^{-1}\left(\delta\omega_{1/0(1)} - {}^1R_0dexp\,(r_s)\,J_{r_s}\delta\omega_{1/0(1)}\right) \tag{96}$$

$$= dexp\,(-r_t)^{-1}\left(I - {}^1R_0dexp\,(r_s)\,J_{r_s}\right)\delta\omega_{1/0(1)}, \tag{97}$$

donc

$$\boxed{J_t = \begin{bmatrix} -\frac{\left(\|r_t\|^2+\epsilon^2\right)r_t^t}{\left(\|r_t\|^2+\epsilon^2\right)^{\frac{3}{2}}}dexp\,(-r_t)^{-1}\left(I - {}^1R_0dexp\,(r_s)\,J_{r_s}\right) \\ \\ \frac{\left(\|r_t\|^2+\epsilon^2\right)r_t^t}{\left(\|r_t\|^2+\epsilon^2\right)^{\frac{3}{2}}}dexp\,(-r_t)^{-1}\left(I - {}^1R_0dexp\,(r_s)\,J_{r_s}\right) \end{bmatrix}}. \tag{98}$$

L'algorithme a été testé, mais nous laissons son implémentation sur le mannequin complet en perspective.

Bibliographie

[AB03] J. Allbeck and N. Badler. Representing and parameterizing agent behaviors. *Life-like Characters : Tools, Affective Functions and Applications*, 2003.

[ABD86] P. Agati, Y. Bremont, and G. Delville. *Mécanique du Solide, Applications Industrielles.* 1986.

[AEJ04] G. Arechavaleta, C. Esteves, and Laumond J.P. Planning fine motions for a digital factotum. In *IEEE / RSJ International Conference on Intelligent Robots and Systems*, 2004.

[AFO03] O. Arikan, D.A. Forsyth, and J.F. O'Brien. Motion synthesis from annotations. In *Proceedings of ACM SIGGRAPH*, San Diego, USA, 2003.

[AFO05] O. Arikan, D.A. Forsyth, and J.F. O'Brien. Pushing people around. In P. Faloutsos K. Anjyo, editor, *Eurographics/ACM SIGGRAPH Symposium on Computer Animation*, 2005.

[Age05] Ageia. A whipe paper : Physics, gameplay and the physics processing unit. Technical report, Ageia Technologies, Inc, `http://www.ageia.com/pdf/wp_2005_3_physics_gameplay.pdf`, 2005.

[AH99] R.J. Adams and B. Hannaford. Stable haptic interaction with virtual environments. *IEEE Transactions on Robotics and Automation*, 15(3), 1999.

[AKA+02] J. Allbeck, K. Kipper, C. Adams, W. Schuler, E. Zoubanova, N. Badler, M. Palmer, and A. Joshi. Acumen : Amplifying control and understanding of multiple entities. In *Workshop AAMAS 2002 : Embodied Conversationnal Agents-let's specify and evaluate them.*, Bologna, Italy, July 2002.

[AKH01] R.J. Adams, D. Klowen, and B. Hannaford. Virtual training for a manual assembly task. *Haptics-e*, 2(2), 2001.

[AKM+02] Y. Arafa, K. Kamyab, E. Mamdani, S. Kshirsagar, N. Magnenat-Thalmann, A. Guye-Vuillème, and D. Thalmann. Two approaches to scripting character animation. In *Workshop AAMAS 2002 : Embodied Conversationnal Agents-let's specify and evaluate them.*, Bologna, Italy, July 2002.

[Ali] Alias, `http://www.alias.com/glb/eng/products-services/family_details.jsp?familyId=3900009`. *Maya.*

[AMH02] T. Akenine-Möller and E. Haines. *Real-Time Rendering, second edition.* A. K. Peters, September 2002.

[And90] R.J. Andersen. Dynamic damping control : Implementation issues and simulation results. In *IEEE International Conference on Robotics and Automation*, May 1990.

[And96] R.J. Anderson. Building a modular robot control system using passivity and scattering theory. In *Proceedings of International Conference on Robotics and Automation*, Minneapolis, USA, April 1996.

[APS99] M. Anitescu, F. Potra, and D. Stewart. Time-stepping for three-dimensional rigid body dynamics. *Computer Methods in Applied Mechanics and Engineering*, 177, 1999.

[Aub91] J.P. Aubin. *Viability Theory*. Birkhäuser, 1991.

[Aub02] A. Aubel. *Anatomically-Based Human Body Deformation*. PhD thesis, Ecole Polytechnique Federale de Lausanne, Lausanne, 2002.

[Auta] Autodesk, Inc, www.autodesk.com/3dsmax. *3DSmax*.

[Autb] Autodesk, Inc, `http://usa.autodesk.com/adsk/servlet/item?siteID=123112&id=5659402`. *Character Studio*.

[Bae01] P. Baerlocher. *Inverse Kinematics Techniques for the Interactive Posture Control of Articulated Figures*. PhD thesis, Ecole Polytechnique Fédérale de Lausanne, 2001.

[Bar94] D. Baraff. Fast contact force computation for nonpenetrating rigid bodies. In *SIGGRAPH, Computer Graphics Proceedings*, July 1994.

[BAZB03] N. Badler, J. Allbeck, L. Zhao, and Byun. Representing and parameterizing agent behaviors. In *Lifelike Characters : Tools Affective Functions and Applications*, Germany, 2003.

[BB04] P. Baerlocher and R. Boulic. An inverse kinematic architecture enforcing an arbitrary number of strict priority levels. *The Visual Computer*, 20(6), 2004.

[BDR+03] A. Bloomfield, Y. Deng, P. Rondot, J. Wampler, D. Harth, M. McManus, and N. Badler. A taxonomy and comparison of haptics for disassembly tasks. In *Proceedings of IEEE Virtual Reality Conference*, pages 225–231, Los Angeles, USA, March 2003.

[BELF05] I. Belousov, C. Esteves, J.P. Laumond, and E. Ferré. Motion planning for large space manipulators with complicated dynamics. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Edmonton, Canada, 2005.

[BFG] *BFG technologies*. `http://www.bfgtech.com/`.

[BLR05] T. Bretl, J.C. Latombe, and S. Rock. Toward autonomous free-climbing robots. *Robotics Research, Springer*, pages 6–15, 2005.

[BMTT90] R. Boulic, N. Magnenat-Thalmann, and D. Thalmann. A global human walking model with real-time kinematic personnification. *The Visual Computer*, 6(6), 1990.

[BO71] T.L. Boullion and P.L. Odell. *Generalized Inverse Matrices*. Wiley-Interscience, 1971.

[Bol57] E.F. Bolinder. Survey of some properties of linear networks. *IRE Transactions on Circuit Theory, CT-4*, pages 70–78, 1957.

[Bro99] A.S. Brown. Role models. *Mechanical engineering*, 1999.

[BSP+04] J. Barbic, A. Safonova, J.Y. Pan, C. Faloutsos, J.K. Hodgins, and N.S. Pollard. Segmenting motion capture data into distinct behaviors. In *Proceedings of Graphics Interface*, May 2004.

[BW95] A. Bruderlin and L. Williams. Motion signal processing. In *Proceedings of ACM SIGGRAPH*, 1995.

[CCC03] P. Chedmail, D. Chablat, and Le Roy C. A distributed approach for access and visibility task with a manikin and a robot in a virtual reality environment. *IEEE Transactions on Industrial Electronics*, 50(4) :692–698, August 2003.

[Ced] *Projet Européen CEDIX : Concurrent Engineering based design for Integrated X-methodologies*. `http://www.irccyn.ec-nantes.fr/Equipes/CMAO/CMAO_3_2.html`.

[CG93] P.E. Crouch and R. Grossman. Numerical integration of ordinary differential equations on manifolds. *Journal of Nonlinear Science*, 3 :1–33, 1993.

[Chaa] *The Serious Games Initiative*. `http://www.seriousgames.org/`.

[Chab] Character Animation Technologies, Inc, `http://www.catoolkit.com/`. *CAT*.

[Cha06] *Serious Games Summit*. Game Developers Conference, San Jose, USA, March 2006.

[Che05] M. Chevaldonné. *Représentation et Interaction pour l'Immersion Virtuelle*. PhD thesis, Université de Bourgogne, 2005.

[Cli64] R.E. Cline. Representation for the generalized inverse of a partitioned matrix. *Journal of Soc. Industrial Applied Mathematics*, XII :588–600, 1964.

[CMU] *CMU Graphics Lab Motion Capture Database*.

[CO03] E. Celledoni and B. Owren. Lie group methods for rigid body dynamics and time integration on manifolds. *Computer Methods in Applied Mechanics and Engineering*, 192 :421–438, 2003.

[Col94] J.E. Colgate. Coupled stability of multiport systems - theory and experiments. *Journal of Dynamic Systems, Measurement, and Control*, 116 :419–428, September 1994.

[Cor98] Microsoft Corporation. *Microsoft Programming Interface Overview*. `http://www.microsoft.com`, 1998.

[CPR92] E.W. Cottle, J.S. Pang, and Stone R.E. *The Linear Complementarity Problem*. 1992.

[Dasa] Dassault Systèmes, `http://www.3ds.com/products-solutions/plm-solutions/catia/overview/`. *Catia*.

[Dasb] Dassault Systèmes, `http://www.delmia.com/`. *Delmia*.

[DC01] T. Drummond and R. Cipolla. Real-time tracking of highly articulated structures in the presence of noisy measurements. In *Proceedings of ICCV*, pages 315–320, 2001.

[DDKA05] C. Duriez, F. Dubois, A. Kheddar, and C. Andriot. Realistic haptic rendering of interacting deformable objects in virtual environments. *IEEE Transactions on visualization and computer graphics*, November/December 2005.

[DDLT02] F. Devillers, S. Donikian, F. Lamarche, and J.F. Taille. A programming environment for behavioral animation. *The Journal of Visualization and Computer Animation*, 13 :263–274, 2002.

[DGLC05] G. Drieux, F. Guillaume, J.C. Leon, and N. Chevassus. Samira : A platform for virtual maintenance simulation with haptic feedback incorporating a model preparation process. In *Proceedings of Virtual Concept*, Biarritz, France, November 2005.

[DIG] *Boston Dynamics : DI-Guy*. www.bdi.com.

[DLCG05] G. Drieux, J.C. Léon, N. Chevassus, and F. Guillaume. A formal description of preparation processes for geometric models to improve downstream activities integration in the design process. In *Proceedings of the 9th International Conference on Computer Supported Cooperative Work in Design*, Coventry, UK, month 2005.

[Dur04a] M. Dureigne. Collaborative large engineering : from it dream to reality. *Methods and Tools for Co-operative and Integrated Design*, 2004. Kluwer Academic Publishers.

[Dur04b] C. Duriez. *Contact frottant entre objets déformables dans des simulations temps-réel avec retour d'effort*. PhD thesis, Université d'Evry, December 2004.

[EAL05] C. Esteves, G. Arechavaleta, and J.P Laumond. Motion planning for human-robot interaction in manipulation tasks. In *Special session at the IEEE International Conference on Mechatronics and Automation (ICMA)*, pages 1766–1771, Niagara Falls, Canada, August 2005.

[ECR92] B. Espiau, F. Chaumette, and P. Rives. A new approach to visual servoing in robotics. In *IEEE Trans. on Robotics and Automation*, June 1992.

[EnH] *Projet Européen ENHANCE (Enhanced Aeronautical Concurrent Engineering)*. `http://www.ircyn.ec-nantes.fr/Equipes/CMAO/CMAO_3_3.html`.

[Erg] *Deneb : Ergo*. www.deneb.com/products/ergo.html.

[Fal] P. Faloutsos. *DANCE : Dynamic Animation and Control Environment*.

[FH85] T. Flash and N. Hogan. The coordination of arm movements : An experimentally confirmed mathematical model. *The Journal of Neuroscience*, 5(7) :1688–1703, July 1985.

[Fig] *Adams : Figure*. www.adams.com.

[FMP01] P. Fuchs, G. Moreau, and J.P. Papin. *Le Traité de la Réalité Virtuelle*. Ecole des Mines de Paris - Les Presses, 2001.

[FP03] A.C. Fang and N.S. Pollard. Efficient synthesis of physically valid human motion. In *Proceedings of ACM SIGGRAPH*, San Diego, USA, July 2003.

[fSH05] Human Model for Safety 2 : HUMOS 2. Cordis RTD-Projects, `http://ica.cordis.lu/search/index.cfm?fuseaction=proj.simpledocument&PJ_RCN=5744049&CFID=6456931&CFTOKEN=74433437`, 2005.

[Fuc03] P. Fuchs. *Le traité de la Réalité Virtuelle (2^{eme} édition)*. 2003.

[FVdPT98] P. Faloutsos, M. Van de Panne, and D. Terzopoulos. The virtual stuntman : Dynamic characters with a repertoire of autonomous motor skills. *Computers and Graphics*, pages 356–367, 1998.

[GCL05] M. Guerrand, C. Chabal, and C. Leroy. Moteur de scénarisation interactive, spécification logicielle. Technical report, CEA/LIST, Laboratoire d'Automatique et de Commande, 2005.

[GL98] M. Gleicher and P. Litwinowicz. Constraint-based motion adpatation. *The Journal of Visualization and Computer Animation*, 9 :65–94, 1998.

[Gle98] M. Gleicher. Retargetting motion to new characters. In *Proceedings of SIGGRAPH. In Computer Graphics Annual Conference Series*, 1998.

[Gra98] F.S. Grassia. Practical parameterization of rotations using the exponential map. *Journal of Graphics Tools*, 3(3) :29–48, 1998.

[Gui] F. Guillaume. *Samira*. EADS CCR, `http://www.eads.net/web/lang/en/800/content/OF00000000400004/5/55/33623555.html`.

[Has] Hash, Inc, `http://www.hash.com/`. *Hash Animation Master*.

[HHK+04] K. Harada, H. Hirukawa, F. Kanehiro, K. Fujiwara, K. Kaneko, and M. Kajita, S. Nakamura. Dynamical balance of a humanoid robot grasping an environment. In *Proceedings of 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sendai, Japan, 2004.

[Hil02] L. Hilde. *Algorithmes de résolution des équations du mouvement pour l'animation basée sur la physique*. PhD thesis, Université des Sciences et Technologies de Lille, December 2002.

[HKKH03a] K. Harada, S. Kajita, K. Kaneko, and H. Hirukawa. Pushing manipulation by humanoid considering two-kinds of zmps. In *Proceedings of the International Conference on Robotics and Systems*, Taipei, Taiwan, September 2003.

[HKKH03b] K. Harada, S. Kajita, K. Kaneko, and H. Hirukawa. Zmp analysis for arm/leg coordination. In *Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Las Vegas, USA, October 2003.

[HLK05] D. Hsu, J.C. Latombe, and H. Kurniawati. On the probabilistic foundations of probabilistic roadmap planning. In *12th Int. Symp. on Robotics Research*, San Francisco, USA, October 2005.

[Hod98] T. Hodgson. *Handout 5.7. : Virtual Prototyping*. Comptek Federal Systems, Inc, `http://www.wu-wien.ac.at/marketing/lv/VO_Hasenauer/virtual%20prototyping.doc`, 1998.

[Hog87] N. Hogan. Beyond regulators : Modelling control systems as physical systems. In *Proceedings of the 1987 American Control Conference*, Minneapolis, USA, June 1987.

[HW98] J.K. Hodgins and W.L. Wooten. Animating human athletes. In Springer-Verlag : Berlin, editor, *Robotics Research : The Eight International Symposium*, pages 356–367, 1998.

[IMH05] T. Igarashi, T. Moscovitch, and J.F. Hughes. Spatial keyframing for performance-driven animation. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, Los Angeles, USA, 2005.

[IMKNZ00] A. Iserles, H.Z. Munthe-Kaas, S.P. Norsett, and A. Zanna. Lie-group methods. *Acta Numerica (Cambridge University Press)*, (9) :215–365, 2000.

[Jac] *Eds : Jack*. www.plmsolutions-eds.com.

[JC01] D. Johnson and E. Cohen. Spetialized normal cone hierarchies. In *Proceedings of ACM Symposium on Interactive 3D Graphics*, pages 129–134, March 2001.

[Jol94] L. Joly. Thème et variations sur le concept de mécanisme virtuel. Technical report, CEA/LIST, Laboratoire d'Automatique et de Commande, 1994.

[Jol97] L.D. Joly. *Commande Hybride Position/Force pour la Téléopération : une Approche Basée sur les Analogies Mécaniques*. PhD thesis, Université de Paris 6, 1997.

[JW04] D. Johnson and P. Willemson. Accelerated haptic rendering of polygonal models through local descent. In *Proc. of IEEE Haptics Symposium 2004*, 2004.

[KAAT03] M. Kallman, A. Aubel, T. Abaci, and D. Thalmann. Planning collision-free reaching motions for interactive object manipulation and grasping. In P. Brunet and D. Fellner, editors, *Proceedings of Eurographics*, volume 22, 2003.

[Kel98] D. Kelly. *Character Animation in Depth*. The Coriolis Group, Inc, September 1998.

[KG04] L. Kovar and M. Gleicher. Automated extraction and parameterization of motions in large data sets. In *ACM Transactions on Graphics SIGGRAPH Proceedings*, volume 23, 2004.

[KGP02] L. Koval, M. Gleicher, and F. Pighin. Motion graphs. In *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, volume 21, July 2002.

[Kha95] O. Khatib. Inertial properties in robotic manipulation : an object-level framework. *International Journal of Robotics Research*, 14(1) :19–36, February 1995.

[Kha02] H.K. Khalil. *Nonlinear Systems, third edition*. Prentice Hall, 2002.

[Kin] Kineo, `http://www.kineocam.com/`. *Kineo CAM*.

[KT99] M. Kallmann and D. Thalmann. A behavioral interface to simulate agent-object interactions in real-time. *Proceedings of Computer Animation, IEEE Computer Society Press*, pages 138–146, 1999.

[KWDSS04] H. Khatib, J. Warren, V. De Sapio, and L. Sentis. Human-like motion from physiologically-based potential energies. In Kluwer Academic Publishers, editor, *Advances in Robot Kinematics*, pages 149–163, November 2004.

[Lat91] J.C. Latombe. *Robot Motion Planning*. Boston, USA, 1991.

[LCF00] J.P. Lewis, M. Cordner, and N. Fong. Pose spare deformation : A unified approach to shape interpolation and skeleton skeleton-driven deformation. In *Proceedings of SIGGRAPH*, August 2000.

[LD02] F. Lamarche and S. Donikian. Automatic orchestration of behaviours through the management of resources and priority levels. In *Proceedings of Autonomous Agents and Multiagent Systems (AAMAS'02)*, Bologna, Italy, July 2002.

[Ler00] C. Leroy. *Accessibilité et aptitude au montage - Réalité virtuelle et approche multi-agent*. PhD thesis, Université de Nantes et Ecole Centrale de Nantes, January 2000.

[LIO] LIO : "Laboratoire d'Informatique d'Orléans" and LID : "Laboratoire Informatique et Distribution", `http://netjuggler.sourceforge.net/`. *NetJuggler*.

[LL04] G. Liu and Z. Li. Real-time grasping force optimization for multifingered manipulation :theory and experiments. *IEEE/ASME Transactions on Mechatronics*, 9(1), March 2004.

[Lle52] F.B. Llewellyn. Some fundamental properties of transmission systems. In *Proceedings of IRE*, volume 40, pages 271–283, 1952.

[LP02] C.K. Liu and Z. Popovic. Synthesis of complex dynamic character motion from simple animations. In *Proceedings of ACM SIGGRAPH*, San Antonio, USA, July 2002.

[LS02] T.Y. Li and Y.C. Shie. An incremental learning approach to motion planning with roadmap management. In *Proceedings of International Conference on Robotics and Automation (ICRA)*, 2002.

[LS03] J. Lenarcic and M. Stanisic. A humanoid shoulder complex and humeral pointing kinematics. *IEEE Transactions On Robotics and Automation*, 19(3), June 2003.

[LVdPF96] J. Laszlo, M. Van de Panne, and E. Fiume. Limit cycle control, and its applications to the animation of balancing and walking. In *Proceedings of ACM SIGGRAPH*, New Orleans, USA, August 1996.

[Mai03] B. Maille. *Etude de l'accessibilité par l'approche multi-agent en réalité virtuelle - Emergence de comportement*. PhD thesis, Université de Nantes et Ecole Centrale de Nantes, June 2003.

[MC94] B. Mirtich and J. Canny. Impulse-based dynamic simulation. In *Proceedings of Workshop on Algorithmc Foundations of Robotics*, February 1994.

[MCR02] B. Maillé, P. Chedmail, and E. Ramstein. Multi-agent approach and emergence of a virtual human behaviour in a vr environment. In *IEEE International Conference on Systems, Man and Cybernetics*, Hammamet, Tunisia, 2002.

[MDAK] F. Multon, S. Donikian, B. Arnaldi, and K. Kulpa. *Manageable Kinematics Motion (MKM)*. IRISA Rennes, `http://www.irisa.fr/siames/MKM/French/index.htm`.

[Mer05] X. Merlhiot. Projet lhir, dossier d'analyse des algorithmes embarqués. Technical report, CEA/LIST, SCRI, merlhiotx@zoe.cea.fr, 2005.

[MIH91] F.A. Mussa-Ivaldi and N. Hogan. Integrable solutions of kinematic redundancy via impedance control. *International Journal of Robotics Research*, 10(5) :481–491, 1991.

[Min83] M. Minoux. *Programmation Mathématique*. 1983.

[Mir95] B. Mirtich. Hybrid simulation : Combining constraints and impulses. In *Proceedings of Workshop on Simulation and Interaction in Virtual Environments*, July 1995.

[MK85] A.A. Maciejewski and C.A. Klein. Obstacle avoidance for kinematically redundant manipulators in dynamically varying environmnts. *The International Journal of Robotics Research*, 4(3) :109–117, December 1985.

[MK98] H. Munthe-Kass. Runge-kutta methods on lie groups. *BIT Numerical Mathematics*, 38(1) :92–111, 1998.

[MLSS94] R.M. Murray, Z. Li, and S. Shankar Sastry. *A Mathematical Introduction to Robotic Manipulation*. CRC Press, 1994.

[MMA01] S. Ménardais, F. Multon, and B. Arnaldi. A global framework for motion capture. Technical report, INRIA Rennes, Franck.Multon@uhb.fr, 2001.

[Moa02] M. Moakher. Means and averaging in the group of rotations. *SIAM Journal on Matrix Analysis and Applications*, 24(1), 2002.

[Mon89] D.J. Montana. The kinematics of contact with compliance. In *IEEE International Conference on Robotics and Automation*, pages 770–775, 1989.

[Nak91] Y. Nakamura. *Advanced Robotics : Redundancy and Optimization*. AddisonWesley, Reading, MA, 1991.

[OCY97] Y. Oh, W.K. Chung, and Y. Youm. Extended impedance control of redundant manipulators using joint space decomposition. In Kluwer Academic Publishers, editor, *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1080–1087, April 1997.

[P.03] Iglesias P. Principes variationnels et géométrie symplectique. Technical report, UMPA, Ecole Normale Supérieure de Lyon, France, `http://www.umpa.ens-lyon.fr/~iglesias/articles/Variations%20et%20symplectique/variations.html`, 2003.

[PC04] J. Park and W.K. Chung. Geometric integration on euclidean group with application to articulated multibody systems. *Transactions on Robotics*, november 2004.

[Pha05] Matt Pharr. *GPU Gems 2*. Addison-Wesley, March 2005.

[PLS03] J. Pettre, J.P. Laumond, and T. Simeon. A 2-stages locomotion planner for digital actors. In *ACM SIGGRAPH / Eurographics Symposium on Computer Animation (SCA)*, 2003.

[PTVF02] W.H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery. *Numerical Recipes in C++. The Art of Scientific Computing. Second Edition*. Cambridge University Press, 2002.

[PW99] Z. Popovic and A. Witkin. Physically based motion transformation. In *Proceedings of ACM SIGGRAPH*, 1999.

[Rat03] P. Ratner. *3D Human Modeling and Animation, second edition.* Wiley, April 2003.

[RK97] D.C. Ruspini and O. Khatib. Collision/contact models for the dynamic simulation of complex environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems : IROS'97*, September 1997.

[RMA+04] A. Rennuit, A. Micaelli, C. Andriot, F. Guillaume, N. Chevassus, D. Chablat, and P. Chedmail. Designing a virtual manikin animation framework aimed at virtual prototyping. In *Proceedings of IEEE VRIC*, Laval, France, 2004.

[RMM+05a] A. Rennuit, A. Micaelli, X. Merlhiot, C. Andriot, F. Guillaume, N. Chevassus, D. Chablat, and P. Chedmail. Balanced virtual humans interacting with their environments. In *Proceedings of Summer Computer Simulation Conference*, July 2005.

[RMM+05b] A. Rennuit, A. Micaelli, X. Merlhiot, C. Andriot, F. Guillaume, N. Chevassus, D. Chablat, and P. Chedmail. Integration of a balanced virtual manikin in a virtual reality platform aimed at virtual prototyping. In *Proceedings of Virtual Concept 2005*, November 2005.

[RMM+05c] A. Rennuit, A. Micaelli, X. Merlhiot, C. Andriot, F. Guillaume, N. Chevassus, D. Chablat, and P. Chedmail. Passive control architecture for virtual humans. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Edmonton, Canada, August 2005.

[Roy99] L. Royer. *Etude et Conception de la Commande Téléopérée des Manipulateurs Redondants Application à un Bras Redondant Anthropomorphe.* PhD thesis, Université de Paris 6, September 1999.

[RP04] P.S.A. Reitsma and N.S. Polard. Evaluating motion graphs for character navigation. In *Eurographics/ACM SIGGRAPH Symposium on Computer Animation*, 2004.

[Saf] *Safework.* www.safework.com.

[Sal80] J.K. Salisbury. Active stiffness control of a manipulator in cartesian coordinates. In *Proceedings of the 19th IEEE Int. Conference on Decision and Control*, pages 95–100, Albuquerque, USA, September 1980.

[SB01] S. Stramigioli and H. Bruyninckx. Modeling and control of rigid mechanical systems. In *International Symposium on DIStributed Computing (DISC)*, 2001.

[SFNTH05] A. Shapiro, P. Faloutsos, and V. Ng-Thow-Hing. Dynamic animation and control environment. *Graphics Interface*, pages 61–70, 2005.

[SHP04] A. Safonova, J.K. Hodgins, and N. Pollard. Synthesizing physically realistic human motion in low dimensional behavior specific spaces. In *ACM Transactions on Graphics SIGGRAPH Proceedings*, volume 23, 2004.

[Sig33] S. Signorini. Sopra akune questioni di elastostatica. *Atti della Societa Italiana per il Progresso delle Scienze*, page 38, 1933.

[SK] M. Stilman and J.J. Kuffner. Navigation among movable obstacles : Real-time reasoning in complex environments. *International Journal of Humanoid Robotics (to appear).*

[SK04] L. Sentis and O. Khatib. Task-oriented control of humanoid robots through prioritization. *International Journal of Humanoid Robotics*, December 2004.

[SK05] L. Sentis and O. Khatib. Synthesis of whole-body behaviors through hierarchical control of behavioral primitives. *International Journal of Humanoid Robotics*, in press 2005.

[SKG05] M. Sung, L. Kovar, and M. Gleicher. Fast and accurate goal-directed motion synthesis for crowds. In P. Faloutsos K. Anjyo, editor, *Proceedings of Eurographics/ACM SIGGRAPH Symposium on Computer Animation*, 2005.

[SL04] H. Schmidl and M.C. Lin. Geometry-driven interaction between avatars and virtual environment. *Computer Animation and Virtual Worlds Journal*, July 2004.

[SLCPar] M. Saha, J.C. Latombe, Y.C. Chang, and F. Prinz. Finding narrow passages with probabilistic roadmaps : The small-step retraction method. *Autonomous Robots*, To appear.

[SLGS01] H.J. Shin, J. Lee, M. Gleicher, and S.Y. Shin. Computer puppetry : An importance-based approach. *ACM Transactions of Graphics*, April 2001.

[SMT04] H. Seo and N. Magnenat Thalmann. An example-based approach to human body manipulation. *Graphical Models, Academic Press*, 66(1) :1–23, January 2004.

[Sta] Stanford University, `http://ai.stanford.edu/~lsentis/files/research.html`. *Site web de Luis Sentis.*

[Ste04] J.M. Steyaert. *Théorie des automates, langages formels, calculabilité.* Ecole Polytechnique, www.imprimerie.polytechnique.fr/Enseignements/INF542/Steyaert.pdf, 2004.

[TCBV05] A. Tarault, T. Convard, P. Bourdot, and J.M. Vézien. Cluster-based solution for virtual and augmented reality applications. In *GRAPHITE '05 : Proceedings of the 3rd international conference on Computer graphics and interactive techniques in Australasia and South East Asia*, pages 293–296, Dunedin, New Zealand, 2005.

[Teca] *Tecnomatics.* tecnomatix.fr/showpage.asp ?page=405.

[Tecb] TechViz, `http://www.techviz.net/`. *TechViz Turbo.*

[THO+04] D. Thalmann, C. Hery, H. Ono, D. Sutton, S. Lippman, and S. Regelous. Crowd and group and crowd animation. In *Proceedings of Siggraph*, Los Angeles, USA, 2004.

[TI05] T. Takubo and T. Inoue, K. Arai. Pushing operation for humanoid robot using multipoint contact states. In *Proceedings of the 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Edmonton, Canada, August 2005.

[VB04] M. Vukobratovic and B. Borovac. Zero moment point - thirty five years of its life. *International Journal of Humanoïd Robotics*, 1(1) :157–173, 2004.

[VDP97] M. Van De Panne. From footprints to animation. In *Computer Graphics Forum*, volume 16, 1997.

[Ver03] J.P. Verriest. Man3d : un mannequin numérique pour la simulation ergonomique. In *Humanoid Day, Société Française de Biomécanique*, Valencienne, France, 2003.

[VRL] VRLab, EPFL, `http://vrlab.epfl.ch/~michello/main.html`. *Thierry Michellod.*

[Wat01] Prime Faraday Technology Watch. The availability of low-cost prototyping, November 2001.

[Wie00] P.B. Wieber. *Modélisation et Commande d'un Robot Marcheur Anthropomorphe.* PhD thesis, Ecole des Mines de Paris, Décembre 2000.

[Wie02] P.B. Wieber. On the stability of walking systems. In *Proceedings of the International Workshop on Humanoid and Human Friendly Robotics*, 2002.

[Wik05] L'encyclopédie libre Wikipedia. *Automate fini.* Wikipedia, `http://fr.wikipedia.org/wiki/Automate_fini`, 2005.

[Wil03] T. Will. *Introduction to the Singular Value Decomposition.* University of Wisconsin - LaCrosse, `http://www.uwlax.edu/faculty/will/svd/index.html`, 2003.

[WP95] A. Witkin and Z. Popovic. Motion warping. In *Proceedings of ACM SIGGRAPH*, 1995.

[XLL03] J.J. Xu, G. Liu, and Z.X. Li. A study on geometric algorithms for real-time grasping force optimization. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Las Vegas, USA, October 2003.

[ZH02] V.B. Zordan and J.K. Hodgins. Motion capture-driven simulations that hit and react. In *Proceedings of the 2002 ACM Siggraph/Eurographics Symposium on Computer Animation*, pages 89–96, 2002.

[ZVDH03] V.B. Zordan and N.C. Van Der Horst. Mapping optical motion capture data to skeletal motion using a physical model. In *Eurographics/SIGGRAPH Symposium on Computer Animation*, 2003.