HAL
open science

Spécification et résolution de tâches de manipulation complexes. Application à la téléprogrammation de robots distants

Olivier Piccin

To cite this version:

Olivier Piccin. Spécification et résolution de tâches de manipulation complexes. Application à la téléprogrammation de robots distants. Automatique / Robotique. Université Paul Sabatier - Toulouse III, 1995. Français. NNT : . tel-00144080

HAL Id: tel-00144080

https://theses.hal.science/tel-00144080v1

Submitted on 30 Apr 2007

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Année 1995

THÈSE

présentée au

Laboratoire d'Analyse et d'Architecture des Systèmes du CNRS

en vue de l'obtention du grade de

Docteur de l'Université Paul Sabatier de Toulouse

Spécialité: **Robotique**

par

Olivier PICCIN

Ingénieur ARTS ET MÉTIERS

SPÉCIFICATION ET RÉSOLUTION DE TÂCHES DE MANIPULATION COMPLEXES. APPLICATION À LA TÉLÉPROGRAMMATION DE ROBOTS DISTANTS.

Soutenue le mercredi 15 novembre 1995 devant le jury composé de :

MM.	GEORGES	GIRALT	Président
	ALAIN	GIRAUD	Directeur de thèse
	WISAMA	KHALIL	Rapporteurs
	JEAN-PAUL	LALLEMAND	
	MICHEL	COURDESSES	Examinateurs
	MALIK	GHALLAB	
	VINCENT	RIGAUD	

Rapport LAAS N° 95 459
Cette thèse a été préparée au Laboratoire d'Analyse
et d'Architecture des Systèmes du CNRS
7, avenue du Colonel ROCHE
31077 TOULOUSE Cedex (FRANCE)

À la mémoire de ma grand-mère

À mes parents

À Laurence

Avant-propos

Ce mémoire présente les travaux que j'ai effectués au Laboratoire d'Analyse et d'Architecture des Systèmes du CNRS de Toulouse. Je remercie monsieur Alain COSTES, directeur de ce laboratoire, pour m'y avoir accueilli.

Je remercie également monsieur Georges GIRALT, directeur de recherche au CNRS, pour m'avoir proposé, au mois de mars 1992, de visiter le groupe Robotique et Intelligence Artificielle dont il était responsable à l'époque, de m'y avoir accueilli par la suite et enfin, d'avoir présidé mon jury de thèse.

Je souhaite remercier très sincèrement monsieur Alain GIRAUD, directeur de recherche au CNRS qui par sa compétence, sa disponibilité et son amitié m'a permis de travailler dans d'excellentes conditions. Je voudrais également mentionner ses qualités de rigueur, d'exigence ainsi que sa manière non-conformiste d'aborder les problèmes que j'ai particulièrement appréciée au cours de ces trois années.

Je remercie messieurs Wisama KHALIL, Professeur à l'École Centrale de Nantes et Jean-Paul LALLEMAND, Professeur à l'Université de Poitiers d'avoir accepté de juger mon travail en étant rapporteurs de ma thèse.

Je suis également reconnaissant à monsieur Michel COURDESSES, Professeur à l'Université Paul Sabatier de Toulouse pour l'intérêt qu'il a porté à mon travail et sa participation à mon jury.

Je remercie monsieur Malik GHALLAB, directeur de recherche au CNRS, responsable du groupe RIA et monsieur Vincent RIGAUD, responsable du laboratoire de Robotique de l'IFREMER à Toulon d'avoir participé à mon jury de thèse. Je remercie le CNRS et l'IFREMER pour leur contribution au financement de cette thèse.

Je tiens à remercier mes collègues thésards qui, durant ces trois années, ont su faire régner une ambiance amicale dans le groupe : Adelardo, Alain, Bertrand, Brigitte, Carlos, Christophe, Fawzi, Flavien, Florent, Fred, Gilles, Hanna, Hassan, Hervé, Jean-Marc, Jean-Philippe, Jérôme, Jérôme, Luis, Luiz, Maher, Michel, Monique, Mourad, Patrick, Patricia, Patrick, Philippe, Philippe, Rafael, Ricardo, Samer, Sara, Sepanta, Sho'ji, Simon, Stéphane, Thierry, Vanessa, Vincent.

Merci à Daniel, Marc R., Marc, Matthieu, Jackie, à l'ensemble des membres du groupe RIA pour l'aide qu'ils ont pu m'apporter tout au long de cette thèse. Mes remerciements vont aussi au service Documentation du LAAS qui m'a aidé à la réalisation d'une vidéo et a assuré le tirage de ce mémoire.

Enfin, merci à Laurence qui sait combien je lui suis redevable de l'essentiel.

Toulouse, le 30 novembre 1995.

Introduction

La robotique de manipulation a pour but de satisfaire des objectifs très concrets qui se traduisent immanquablement par la génération de mouvements.

Les évolutions d'un robot manipulateur peuvent être classées selon le niveau d'interaction avec l'environnement. Nous pouvons distinguer les mouvements de grande amplitude dans l'espace libre, les mouvements contraints par la présence d'obstacles et les mouvements fins au contact d'objets de l'espace de travail.

En robotique d'intervention, l'environnement du manipulateur n'est pas assez bien connu pour autoriser la programmation complète anticipée des tâches qui sont généralement non répétitives.

Dans ce contexte opératoire, l'identification paramétrique des position et taille des objets présents dans l'environnement associée aux trois volets de la génération de mouvement représentent les difficultés centrales à résoudre pour envisager la réalisation de tâches de manipulation.

Dans cette perspective, nous proposons une méthodologie pour la spécification et la résolution de tâches robotiques complexes. Cette approche a conduit au développement d'un environnement de programmation de robots manipulateurs. Nous nous sommes principalement intéressés à la génération de trajectoires sur lesquelles pourront s'inscrire des lois temporelles d'évolution. La recherche de telles lois ne fait pas l'objet de cet exposé.

L'organisation du mémoire est la suivante : Le premier chapitre situe notre contexte de travail. Nous souhaiterions nous doter d'un outil permettant d'animer des robots manipulateurs pour l'exécution de tâches dans des environnements peu structurés. Après une brève évocation des principales méthodes utilisées, nous présentons les caractéristiques principales de notre démarche.

Dans le second, nous exposons notre technique de spécification de tâches. À partir d'une description 3D généralement polyédrique du manipulateur et de son environnement, la spécification de tâches consiste à décrire les actions à effectuer en termes de mobilités et de contraintes de contact de type égalité ou inégalité à établir ou maintenir entre certains corps. Les relations de contraintes résultantes sont alors obtenues automatiquement grâce à un raisonnement cinématique sur le système mécanique que constitue le robot et son espace de travail.

Le troisième chapitre décrit la méthode de résolution numérique développée pour traiter les équations de contraintes produites par la spécification de tâches. Cette résolution permet de positionner les corps mobiles pour atteindre les objectifs spécifiés. Basée sur un schéma du type Newton-Raphson, la méthode présentée propose un mécanisme d'élimination de variables utilisant un raisonnement hybride numérique/formel. Cette stratégie permet de détecter dynamiquement le conditionnement des équations pour déterminer automatique-

ment le scénario de résolution adapté. Le traitement de contraintes inégalités a été réalisé par introduction de la méthode du simplexe. Il en résulte une robustesse accrue pour la méthode de résolution ainsi qu'une diminution de l'effort demandé à l'utilisateur pour mettre en forme les problèmes qu'il souhaite résoudre.

Le chapitre quatre expose l'application des concepts précédents à la réalisation de tâches robotisées. Les applications de l'approche incluent en premier lieu la génération de trajectoires complexes pour des systèmes mécaniques quelconques comportant éventuellement des mobilités redondantes vis à vis des tâches envisagées. La mise en place de contraintes peut aussi être exploitée pour accomplir, d'un point de vue local, des évitements d'obstacles. Enfin, la nature de cette approche se prête également à la résolution de tâches de calibration des objets présents dans l'univers du robot. Cette caractéristique est essentielle dans le contexte de manipulation distante où les modèles sont mal connus *a priori*. La possibilité d'intégration de données provenant de capteurs sera discutée et approfondie dans le cas particulier d'une caméra vidéo.

Enfin, le dernier chapitre présente une expérimentation de suivi de joint de soudure mise en œuvre sur un manipulateur réel. Ce scénario expérimental, inspiré d'une application réelle de manipulation sous-marine, permettra de proposer une approche pour l'exécution de tâches à distance.

Chapitre 1

Présentation générale

1.1 Introduction

1.1.1 La robotique d'intervention

Après s'être longtemps cantonnée aux milieux de la production industrielle, l'utilisation des robots manipulateurs évolue de plus en plus en direction de la robotique non manufacturière. Cette activité se caractérise par la mise en œuvre de systèmes robotisés à d'autres fins que la fabrication de biens industriels.

Parmi les domaines impliqués figurent les applications de service (robot d'entretien), domestiques (tondeuse à gazon), paramédicales (aide aux personnes handicapées, assistance chirurgicale) ou de réalité virtuelle (animations de synthèse, simulateurs de conduite). Mais le domaine qui retiendra le plus notre intérêt sera celui de la « robotique d'intervention ».

La robotique d'intervention a pour objet de se substituer à l'homme pour accomplir des tâches que ce dernier ne peut réaliser dans de bonnes conditions, pour des raisons d'accessibilité, de sécurité ou de confort. Nous pouvons mentionner les opérations de sécurité civile (lutte contre les incendies, manipulation d'explosifs), les interventions sur sites dangereux (contamination chimique, nucléaire), les travaux en milieu sous-marin (inspection et maintenance de structures off-shore, exploration sous-marine) ou encore les missions d'exploration spatiale.

Si la robotique manufacturière semble avoir atteint un certain degré de maturité, il n'en est pas de même de la robotique d'intervention pour laquelle le nombre de problèmes scientifiques et techniques ouverts demeure important.

Les deux principales caractéristiques de la robotique d'intervention sont la mauvaise connaissance *a priori* de l'environnement de travail et la non répétitivité des tâches à réaliser. Dans ce contexte, les tâches ne peuvent plus être définies complètement à l'avance et demande au robot des capacités d'adaptation à son environnement de travail.

1.1.2 Exécution de tâches sur sites distants

Nous portons intérêt aux contextes applicatifs dans lesquels le robot est distant de l'opérateur. Dans ces conditions, l'exécution de tâche implique des contraintes supplémentaires relatives à la dégradation des possibilités d'échanges entre le site opérateur et le site robotisé distant.

Au titre des facteurs limitants, nous pouvons mentionner le faible débit du canal de communication, les retards de transmission ou encore la nature intermittente des phases de dialogue. Les applications spatiales et sous-marines (sans lien ombilical) sont deux domaines dans lesquels de tels phénomènes peuvent intervenir pour des raisons résumées dans le tableau suivant :

	Applications spatiales	**Applications sous-marines**
Faible débit	Éventuellement	Oui, ~10 kbits/s
Retard	Distances importantes.	Transmissions acoustiques
Communications Intermittentes	Occultations dûes aux mouvements des astres.	Occultations dûes à des obstacles de l'espace de travail.

Le concept de téléopération qui place l'utilisateur au centre de la boucle de contrôle constitue un mode de commande des systèmes robotiques utilisé de longue date. Mais cette stratégie devient difficilement envisageable lorsque les délais de communication sont d'un ordre de grandeur supérieur à la seconde.

1.1.3 Le concept d'autonomie en robotique

Le niveau d'autonomie pour un système robotique se mesure au degré d'implication de l'homme dans sa boucle de commande. Les systèmes totalement autonomes restant encore hors de portée de l'état de l'art, il n'est sans doute pas inutile de préciser le sens attaché au terme autonomie. Une décomposition classique fait apparaître :

- l'autonomie opérationnelle dont une illustration est la réalisation de tâches en l'absence de capteurs (robots industriels primitifs) ,
- l'autonomie décisionnelle qui regroupe des activités relevant d'un plus haut niveau d'abstraction comme l'interprétation de données capteurs ou la décomposition d'actions complexes en sous-tâches plus élémentaires.

La présence simultanée de ces deux composantes caractérise les robots de troisième génération. Ceux-ci sont définis de manière qualitative par G. GIRALT *et al.* comme étant des machines susceptibles de raisonner sur une tâche et son exécution en associant intelligemment l'action à la perception [GIRALT 89].

La **téléprogrammation** au niveau tâche constitue le mode de commande privilégié des robots de troisième génération [DEGALLAIX 93]. Dans cette approche, les actions ou tâches à faire réaliser par le robot sont décrites sous la forme d'un programme (un plan) que le robot est capable d'interpréter, au moment de l'exécution effective, en fonction d'informations sensorielles dont il dispose. L'homme reste présent dans la boucle de commande mais à un niveau plus élevé.

L'engouement pour la robotique mobile a vu se développer quantité de travaux concernant les questions d'autonomie. Il faut cependant garder à l'esprit que la réalisation d'objectifs utilitaires par un robot mobile impose que l'on s'intéresse également aux problèmes spécifiques du contrôle de bras manipulateurs.

Autonomie et robotique de manipulation

La programmation automatique de tâches de manipulation a fait l'objet de nombreux travaux qui ont parfois débouché sur des implantations effectives [LAUGIER 85, MAZER 87, VIOLERO 91].

Ils ont pour caractère commun de décomposer hiérarchiquement la tâche initiale en sous-problèmes *a priori* plus simples à résoudre. Ces systèmes de planification font appel à différents modules destinés à traiter les sous-problèmes suivants :

- la génération de mouvements de transfert,
- la génération de mouvements fins,
- la synthèse de prises stables.

Les principales difficultés auxquelles les travaux précédents s'attaquent sont la décomposition de la tâche initiale mais surtout, l'intégration correcte des réponses fournies par les différents modules de traitement pour exécuter la tâche voulue.

Indépendamment des questions relatives à la planification de tâches comme la décomposition en sous-tâches élémentaires ou le raisonnement sur les stratégies de prise, il nous paraît crucial de disposer d'une méthode suffisamment générale et flexible pour assurer la génération de trajectoires.

1.2 Stratégies de génération de trajectoire en robotique de manipulation

Dans cette partie, nous réalisons un bref tour d'horizon des approches employées pour animer des manipulateurs dans leur environnement.

1.2.1 Utilisation d'un modèle de réalité virtuelle

La première étape de ce mode de contrôle consiste à construire un modèle graphique représentant le site robotisé. Les fonctionnalités de bases pour la visualisation sont assurées par un modeleur d'objets 3D. La représentation graphique comprendra un modèle du robot que l'on souhaite animer et un modèle décrivant les objets environnant le robot, qui seront le plus souvent immobiles.

Au robot du modèle graphique est associé un périphérique d'entrée de type souris 3D ou bras articulé miniature. Nous reprendrons le terme de « bras maître » en usage en téléopération, pour désigner cet accessoire. La base de données du modeleur ainsi connectée au bras maître maintient le robot graphique dans la position du bras maître.

La génération des mouvements dans le modèle virtuel est assurée par l'action de l'opérateur sur le bras maître. Toutefois, pour engendrer des mouvements sans collision sur le modèle, il est souhaitable de faire ressentir à l'opérateur, en termes d'efforts, les effets des déplacements qu'il provoque : un effort nul lorsque le déplacement se fait dans l'espace libre, un effort résistant lorsqu'un des corps mobiles se trouve au contact d'un objet de l'environnement. Le bras maître devra donc être motorisé et instrumenté pour autoriser ce retour d'effort.

Les efforts ressentis par l'opérateur sont artificiels puisqu'ils découlent des collisions constatées sur le modèle graphique. Leur intensité peut être réglée, en temps réel, par une analyse de proximité entre les corps mobiles et les obstacles. Les retours d'efforts synthétiques ou réels sont une des principales composantes du concept de **téléprésence** qui a pour objectif de donner à l'utilisateur l'impression d'être présent physiquement sur le site de travail.

La première application de cette méthodologie est la **téléopération assistée par ordinateur** (TAO) où il n'existe pas de contraintes de communication entre le robot et l'opérateur.

Les techniques de réalité virtuelle permettent d'améliorer les performances et le confort d'exploitation des systèmes de téléopération. Selon le degré d'instrumentation du robot distant, des retours visuels et d'efforts peuvent être fournis à l'opérateur. Des images de synthèse peuvent, par exemple, se substituer à des retours vidéos de qualité médiocre. Le modèle virtuel permet alors de gérer à loisir les conditions d'éclairement, de masquer les objets sans intérêt pour la tâche ou au contraire, de visualiser des corps occultés partiellement dans l'image réelle [THIBOUT 94].

L'utilisation de modèles de synthèse s'est également imposé dans le domaine de la **télérobotique** qui s'intéresse à l'exécution de tâches robotisées en présence de contraintes de communication entre le site opérateur et le robot.

Différents systèmes, basés sur l'utilisation de modèle de réalité virtuelle ont été proposés et complétés pour prendre en compte les effets de retard significatif et de faible débit.

L'autonomie d'exécution des tâches par le robot réel repose principalement sur la présence plus ou moins massive de capteurs gérés en contrôle coopératif (« shared control ») [HIRZINGER 93]. Ce mode de commande met en œuvre des stratégies d'asservissement sensoriel locales par l'intermédiaire desquelles des commandes grossières sont raffinées de manière autonome, dotant ainsi le robot d'une forme élémentaire d'intelligence sensorielle.

Le décalage temporel entre l'envoi d'une action par l'opérateur et sa réalisation effective sur le robot réel se traduit par le besoin de simulation anticipée de l'exécution des tâches. La superposition d'images réelles avec la scène correspondante du modèle virtuel forme la base des techniques de visualisation prédictive (« predictive display ») [KIM 93].

Dans le système proposé par R. PAUL *et al.* [PAUL 92, SAYERS 94], le concept de téléprogrammation a pour objet de stabiliser le schéma de téléopération vis à vis des contraintes de communication. Le système global est décomposé en deux systèmes bouclés. Dans le premier, l'opérateur, situé dans la boucle de contrôle, génère des déplacements dans un modèle virtuel au moyen d'un bras maître à retour d'effort. Le second comprend le robot esclave commandé en force et en déplacement.

Les commandes de téléprogrammation sont déduites des interactions de l'opérateur avec le bras maître et comportent principalement, sous une forme symbolique, des consignes de position et des commandes gardées en effort. Ces ordres sont alors envoyés au robot esclave qui les interprète avec un décalage temporel.

Nous retiendrons principalement de ces approches que les mouvements sont engendrés de manière analogique par l'opérateur.

1.2.2 Les méthodes de planification de trajectoires

La planification de trajectoire en robotique est le plus souvent un problème de nature géométrique. Son étude par les méthodes de la géométrie algorithmique a connu des développements très importants au cours des dernières années. Étant donnée l'étendue considérable actuelle de cette discipline, nous nous limiterons à la présentation de quelques uns des aspects fondamentaux de cette approche. Des références plus complètes sur la planification de trajectoires en robotique peuvent être trouvées notamment dans [LATOMBE 91, BOISSONNAT 88, FAVERJON 88].

Les principes de base

Considérons un robot constitué de plusieurs corps mobiles reliés par des articulations. On désire faire évoluer ce robot vers une position donnée de son espace de travail en évitant les collisions avec les objets qui s'y trouvent.

Le principe général de ces méthodes est de représenter le robot comme un point se déplaçant dans un espace particulier appelé espace des configurations (EC). Cette méthodologie, largement exploitée par la communauté roboticienne, a été formalisée par Lozano-Pérez pour traiter les problèmes de planification de trajectoires. Une configuration $\mathbf{q}$ du système correspond à un jeu minimal de n paramètres indépendants définissant l'état géométrique du robot. L'espace des configurations, engendré par le paramétrage précédent, représente l'ensemble de toutes les positions possibles pour les corps constituant le robot. L'ensemble des configurations admissibles est l'ensemble des positions pour lesquelles les corps mobiles n'entrent pas en collision entre eux ou avec les obstacles fixes.

On distingue deux types d'approches. Les approches globales consistent à construire l'EC en prenant en compte la totalité des contraintes imposées par le robot et son environnement. Cet espace est ensuite structuré en régions correspondantes à l'espace libre. La recherche d'une solution se ramène à l'exploration d'un graphe décrivant la connexité des zones de configurations admissibles. Une solution au problème existe s'il existe une composante connexe de l'espace libre reliant les configurations initiale et finale.

Les approches locales s'attachent à déterminer, à partir d'une configuration donnée du système, un déplacement incrémental qui rapproche le robot de la configuration finale souhaitée, tout en évitant les collisions. Dans ce cas, seules les contraintes autour de la position courante sont prises en compte ce qui conduit à un traitement moins coûteux que la construction de la totalité de l'EC. De plus, la connaissance complète *a priori* de l'environnement n'est plus obligatoire car elle peut être complétée incrémentalement au cours du déplacement. Cependant, de par leur nature locale, ces méthodes peuvent ne pas garantir l'obtention d'une solution même lorsqu'il en existe.

La possibilité d'utilisation pratique des méthodes globales dépend de la complexité des algorithmes qui permettent de construire l'EC. Il a été montré que cette complexité croît comme une fonction exponentielle de la dimension de l'EC.

La limite pratique est de trois ou quatre degrés de liberté. Pour traiter des systèmes avec plus de mobilités il faut combiner les deux types d'approches globale et locale. À titre d'exemple, Faverjon *et al.* ont proposé une méthode locale de planification de trajectoires pour des robots comportant un nombre élevé de degrés de liberté [FAVERJON 89]. Dans ce planificateur, une tâche est définie par une fonction à minimiser, éventuellement soumise

à des contraintes géométriques. Les conditions d'évitement d'obstacles s'expriment sous la forme de contraintes linéaires et le critère à optimiser est quadratique.

Application à la manipulation

La génération de mouvement pour des manipulateurs est une activité complexe, ne serait-ce qu'en raison du nombre élevé de degrés de liberté à gérer. Par contraste avec les méthodes dites analogiques dans lesquelles l'opérateur engendre, par actions physiques, les déplacements, certains systèmes de programmation de robots manipulateurs font appel aux techniques de la planification de trajectoires pour générer les mouvements.

Le système TELEACT [KACZOR 93] se situe dans le même créneau applicatif que l'environnement de téléprogrammation développé par R. PAUL. En complément des modes d'interaction de type téléopération déjà évoqués plus haut, TELEACT associe l'usage d'un modèle virtuel de l'environnement de travail à des techniques de planification de trajectoires.

À l'aide de la représentation graphique et éventuellement d'un bras maître, l'utilisateur spécifie la configuration finale que doit atteindre le manipulateur. Après avoir résolu ce problème localement en simulation au moyen d'un planificateur de trajectoires, la spécification symbolique « rejoindre la position finale » est transmise au site distant pour y être interprétée puis résolue de nouveau par un planificateur. Cette résolution locale au niveau du site distant engendre les mouvements à envoyer, sans problème de débit ou délai, au robot réel.

Le principal intérêt de cette approche est de limiter le volume des commandes symboliques à échanger entre les deux sites grâce à des ordres de haut niveau.

Cependant, comme c'est d'ailleurs le cas avec une approche de type téléprogrammation analogique, la réussite de l'exécution d'une tâche est conditionnée par le degré de fidélité du modèle par rapport à la réalité. Il paraît donc crucial d'intégrer aux spécifications de tâches des mécanismes permettant d'actualiser la représentation virtuelle.

1.2.3 La spécification par contraintes géométriques dans l'espace de la tâche

L'approche que nous avons favorisé consiste à exprimer les objectifs à atteindre, non pas dans des constructions mathématiques dotés de propriétés particulières comme l'espace des configurations, mais dans l'espace physique dans lequel se déroule effectivement la tâche. La nature des contraintes que l'on se propose de traiter est géométrique. C'est dans ce contexte que nous parlerons de *contraintes dans l'espace de la tâche.*

Nous pouvons opérer une classification des systèmes exploitant cette approche sur la base de la complexité requise pour résoudre les contraintes ainsi exprimées. Les espaces de travail familiers pour l'homme sont essentiellement bidimensionnel et tridimensionnel. Leur espace de configuration associé a pour dimension, respectivement, trois (deux translations, une rotation) et six (trois translations, trois rotations). Les paragraphes suivants donnent des exemples relevant de ces catégories d'espaces de travail.

Contraintes en 2D

Un des premiers domaines d'application de l'expression d'objectifs par contraintes en deux dimensions est certainement celui du dessin technique. Tous les outils traditionnels du des-

sinateur (règle, équerre, compas...) permettent de générer des tracés sous contraintes géométriques.

L'avènement des moyens de calcul et les progrès de l'infographie ont permis le développement d'applications de dessin assisté par ordinateur (DAO) qui proposent de multiples fonctionnalités basées sur la satisfaction de contraintes géométriques.

Pour autoriser la production de tracés précis sans effort excessif de la part de l'utilisateur, de nombreux logiciels de dessin mettent en œuvre des dispositifs d'aide au dessin qui correspondent à l'expression de contraintes géométriques entre des éléments composant le dessin [CLA 89, AUT 89]. Leur utilisation interactive naturelle s'appuie sur des manipulations simples réalisées avec une souris graphique.

Un prototype développé par la société Digital Equipment généralise l'utilisation de contraintes géométriques pour produire des dessins en deux dimensions [HEYDON 94]. Le système proposé fait apparaître deux principaux modes d'interaction : le premier, classique, par l'utilisation d'une souris graphique, le second, moins conventionnel, propose un interpréteur de commandes qui résoud en ligne les contraintes de tracé décrites au moyen d'un langage déclaratif spécialisé.

Contraintes en 3D

La généralisation de cette approche à l'espace tridimensionnel ouvre la voie aux applications robotiques mais entraîne simultanément un accroissement de complexité pour la résolution des contraintes.

L'étude des mouvements au contact est un domaine de prédilection pour la spécification par contraintes. DE SCHUTTER *et al.* ont proposé un formalisme dédié à la génération de mouvements compliants, basé sur l'usage de torseurs cinématiques [DE SCHUTTER 92] où sont exploitées les propriétés de dualité entre cinématique et statique.

Le positionnement d'objets dans l'espace par des contraintes géométriques présente un intérêt évident pour un grand nombre de tâches de robotiques comme l'assemblage ou la manipulation.

Le calcul de ces placements de corps peut être réalisé analytiquement dans certains cas particuliers. Une étude sur le calcul du placement relatif de deux objets a été proposée dans [CELAYA 92]. Les contraintes géométriques autorisées sont des relations binaires entre des points, des arêtes ou des faces appartenant aux deux corps. Ces relations sont de type égalité, parallélisme, orthogonalité ou encore angulaire.

Toutefois, l'extension à la satisfaction simultanée de plusieurs contraintes avec un nombre de corps supérieur à deux paraît difficile par une voie analytique.

La résolution numérique semble plus appropriée pour le traitement de problèmes spatiaux complexes. Par exemple, le module robotique de l'application CATIA développée par la société Dassault-Systèmes propose des primitives de programmation par contraintes de type point sur point, point sur droite ou point sur plan [FOURNIER 80, BORREL 85]. Nous mentionnerons également les travaux de A. GIRAUD et T. TRAN [TRAN 90, GIRAUD 91] qui ont servi de base au travail présenté dans cette thèse.

1.3 Les grandes lignes de l'approche proposée

Notre objectif est de proposer une méthodologie permettant la spécification de tâches de manipulation complexes. La programmation par contraintes dans l'espace de la tâche constitue une approche séduisante car elle est très proche d'une description naturelle des objectifs.

1.3.1 Modélisation des tâches

La description d'un grand nombre de problèmes de robotique peut être réalisée en assimilant les robots et leur environnement à des systèmes mécaniques formés de corps rigides.

Une façon naturelle pour un opérateur de spécifier des tâches de déplacement d'un robot est de décrire celles-ci en termes de contraintes géométriques entre le robot et des entités de son univers de travail. Une telle description est facilitée par l'usage d'un modèle graphique 3D.

L'étude géométrique de mécanismes fait intervenir des termes fortement non linéaires (généralement des fonctions trigonométriques). En revanche, un modèle différentiel de cette description géométrique conduit à des relations linéaires entre les dérivées des paramètres de position. Cette propriété importante explique pourquoi il est plus simple de raisonner sur la cinématique d'un mécanisme que sur sa géométrie [ROTH 94].

Dans notre approche, la traduction des objectifs de placement, fixés par l'utilisateur, en relations de contraintes à résoudre se fait donc par un raisonnement cinématique. La tâche étant représentée par des modèles d'état (définissant la géométrie) et variationnel (relatif à la cinématique), il reste à résoudre les équations de contraintes.

Des approches par calcul formel nous paraissent difficilement envisageables dans notre contexte car les systèmes à modéliser sont chaque fois nouveaux en raison du caractère typiquement non répétitif des tâches à réaliser. L'autre contrainte essentielle concerne la mauvaise connaissance *a priori* de certains constituants du système concerné : c'est généralement le cas de l'environnement du robot. Il faut donc être capable, au cours de l'exécution d'une tâche, d'améliorer la connaissance du modèle de l'environnement en effectuant une identification des paramètres mal connus.

Ces exigences nous ont conduit à retenir une approche numérique qui se prête mieux au traitement de cas généraux et à leur résolution automatique.

1.3.2 La méthode de résolution

Le but de cette résolution est de déterminer les positions successives des objets mobiles conduisant à la satisfaction des contraintes. La solution proposée se développe suivant deux directions.

La première est relative au traitement des systèmes d'équations linéaires issus de l'analyse cinématique. Il repose sur un mécanisme d'élimination de variables utilisant un raisonnement hybride numérique/formel. Cette stratégie permet de détecter dynamiquement le conditionnement des équations pour déterminer automatiquement le scénario de résolution adapté. Le traitement de contraintes inégalités a été réalisé par introduction de la méthode du simplexe.

Le second volet de la résolution exploite les résultats issus de ce traitement dans un schéma numérique de type Newton-Raphson qui permet d'engendrer des mouvements in-

crémentaux vers les objectifs spécifiés. Ce point répond au besoin pratique important de génération de *trajectoires continues.*

1.3.3 Applications

Les applications de l'approche incluent en premier lieu la génération de trajectoires complexes pour des systèmes mécaniques quelconques comportant éventuellement des mobilités redondantes vis à vis des tâches envisagées.

La mise en place de contraintes peut aussi être exploitée pour accomplir, d'un point de vue local, des évitements d'obstacles.

L'expression des contraintes dans l'espace de la tâche rend possible la prise en compte de données issues de capteurs fournissant, par exemple, des informations de distance.

Enfin, la nature de cette approche se prête également à la résolution de tâches de calibration des objets présents dans l'univers du robot. Cette caractéristique est essentielle dans le contexte de manipulation distante où les modèles sont mal connus *a priori.*

1.3.4 Aperçu sur l'environnement de programmation graphique

L'environnement de programmation de robots manipulateurs développé est constitué de différents modules fonctionnels organisés autour d'un simulateur graphique qui comprend le modeleur d'objets 3D associé à un interpréteur de commandes.

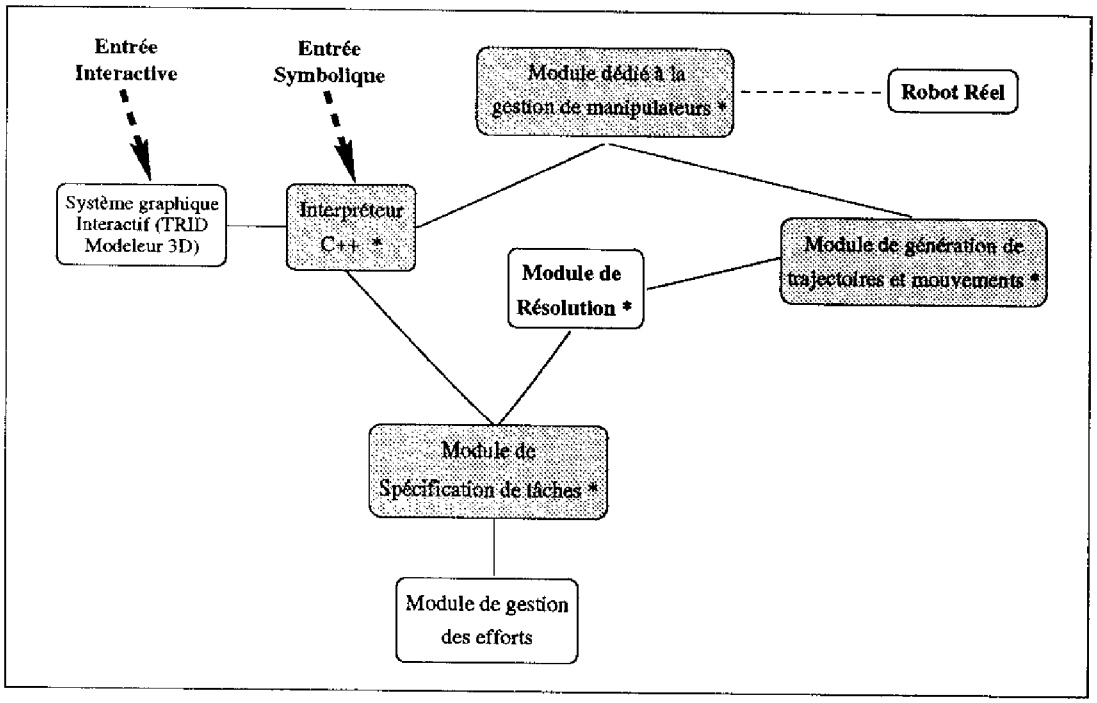

Figure 1.1: Principaux modules de l'environnement de programmation de robots.

Les différents modules ont été agrégés autour de l'interpréteur et du modeleur 3D. Les principaux groupes fonctionnels du systèmes sont (figure 1.1):

- un module de spécification de tâches permettant la formulation de problèmes de robotique de manipulation,
- un module de résolution qui traite les spécifications de tâches fournies par le module précédent,
- un module spécialisé intégrant l'ensemble des fonctions nécessaires à la commande de manipulateurs,
- un module de génération de trajectoires associé à un générateur de mouvements sur trajectoires imposées, utilisant le module de résolution,
- un module pour la gestion des efforts statiques. Ce module, actuellement en phase de développement, constitue une extension du module de spécification de tâches.

Chapitre 2

Méthodologie de spécification de tâches

Lors de la réalisation de tâches par un robot manipulateur, celui-ci est amené à évoluer dans son environnement de travail et interagir avec des objets qui s'y trouvent. Une grande famille de problèmes peuvent être traités en modélisant à la fois le robot et les objets environnant comme un système mécanique formé de corps rigides. Lorsque les exemples traités sont spatiaux et que le nombre de corps augmente, l'utilisation d'une représentation graphique 3D présente un intérêt indiscutable principalement sur le plan de l'ergonomie. La plus grande facilité d'utilisation qui en découle rend possible le traitement de cas qui seraient trop complexes à modéliser sans un support graphique.

Notre objectif est de proposer un système de modélisation basé sur un modeleur d'objets 3D qui permette une description aisée des propriétés cinématiques de mécanismes en vue de spécifier des tâches de déplacement complexe et des problèmes de calibration. Après avoir présenté les principes de bases utilisés pour modéliser un mécanisme, nous décrirons les caractéristiques du modeleur d'objets polyédriques intégré à l'environnement de programmation de robots manipulateurs. Les éléments servant à la spécification de tâches seront ensuite exposés.

2.1 Modélisation des mécanismes

Considérons l'exemple[1] d'un bras à deux degrés de liberté (figure 2.1) que l'on souhaite animer pour que son point terminal se positionne sur une pièce parallélépipédique placée à proximité.

La cinématique d'un tel mécanisme peut être décrite au moyen d'un graphe planaire décrivant les propriétés de mobilité des corps constituant le système. Il s'agit du *graphe des liaisons cinématiques* que nous nommerons simplement par la suite graphe cinématique.

Classiquement, les sommets de ce graphe correspondent aux corps du mécanisme tandis que les arcs renseignent à la fois sur la situation et la mobilité relative des solides. La représentation que nous utiliserons, tout en restant équivalente à la précédente, s'attachera à définir plus précisément les positions des liaisons sur chaque corps. À cet effet, les nœuds du graphe cinématique représenteront des référentiels appartenant aux corps du mécanisme

[1] Ce mécanisme est inspiré d'un exemple présenté au chapitre 4 du livre d'A. Liégeois [Liégeois 84].

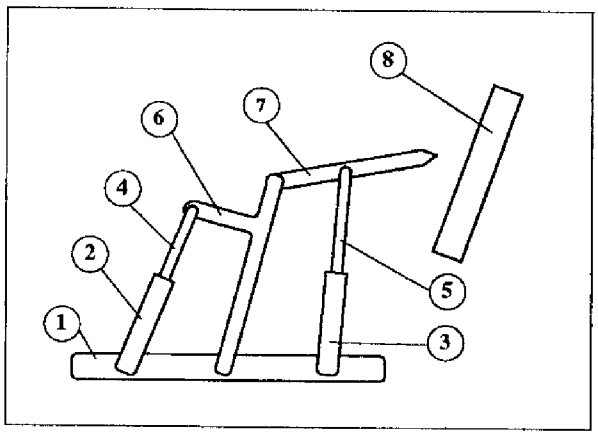

Figure 2.1: Mécanisme à deux degrés de liberté.

et ses arcs décriront la situation et la mobilité relative des référentiels.

Un graphe initial où ne figurent que les corps du mécanisme est présenté sur la figure 2.2. Le graphe de la figure 2.3 est conforme à la représentation utilisée par notre système de spécification. Les arcs en trait continu représentent des liens fixes de positionnement de référentiels tandis que les arcs en trait pointillé définissent des liaisons mobiles.

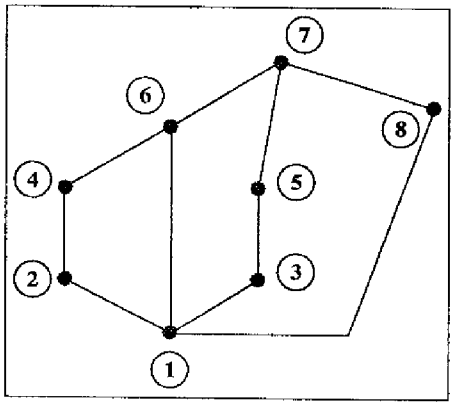

Figure 2.2: Graphe du mécanisme.

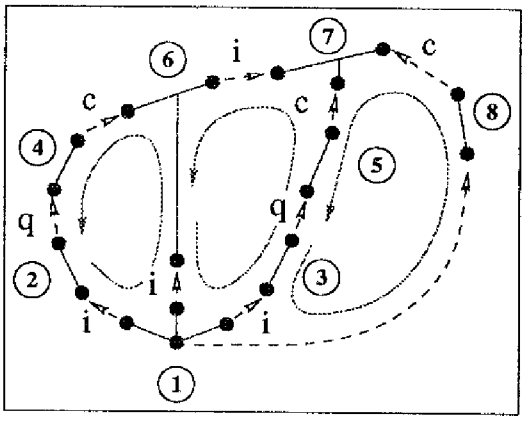

Figure 2.3: Graphe étendu.

Ce bras articulé est animé par deux vérins modélisés par les couples de corps (2,4) et (3,5). Leur état est défini par les variables de commande q. La représentation de l'état géométrique du mécanisme doit se faire, de manière classique, à partir d'un système arborescent équivalent auquel sont rajoutées des contraintes de fermeture de boucles. Dans l'exemple envisagé, on suppose que les liaisons entre (4,6) et (5,7) ont été coupées. Avec ce choix, l'état géométrique du bras est déterminé à l'aide des variables de commande q et des variables articulaires inconnues i qui permettent de satisfaire les contraintes de fermeture c.

Nous voyons apparaître à travers cet exemple les besoins qui se font sentir pour modéliser des mécanismes. Leur étude peut se décomposer en deux phases.
La première qui relève d'un niveau purement descriptif, a pour but l'installation de référentiels sur les corps, la définition des mobilités relatives au moyen d'articulations, et enfin, l'expression des contraintes pertinentes pour la tâche considérée.

Dans une seconde phase, il faut pouvoir engendrer automatiquement, à partir de la description précédente, le graphe cinématique du mécanisme. Enfin, la dernière étape de cette modélisation doit permettre l'extraction des équations de contraintes à résoudre.

2.2 Définitions de base

Situation relative de deux corps

Soient R_F et R_m des repères attachés à deux solides S_1 et S_2. La *situation* de S_2 par rapport à S_1 est définie si l'on connaît la transformation géométrique existante entre R_F et R_m (figure 2.4). Cette localisation relative dans l'espace peut s'exprimer de multiples manières. Étant donnée l'exploitation numérique que nous ferons de cette représentation, un repère de projection R_X sera précisé explicitement dans les notations utilisées. Nous utiliserons les deux représentations équivalentes suivantes.

Matrice de transformation homogène :

$$H_{XFm} = \begin{bmatrix} R_{XFm} & \overrightarrow{P}_{XFm} \\ 0 & 1 \end{bmatrix}$$

R_{XFm} et $\overrightarrow{P}_{XFm}$ désignent la matrice de rotation de $S2$ par rapport à $S1$ et le vecteur translation entre les origines de R_F et R_m, exprimés dans R_X.

Vecteur de configuration de $\mathbb{R}^6$:

$$Q_{XFm} = \begin{bmatrix} tx & ty & tz & rx & ry & rz \end{bmatrix}^T$$

$\overrightarrow{P}_{XFm} = \begin{bmatrix} tx & ty & tz \end{bmatrix}^T$ est le vecteur de translation défini ci-dessus,

$\overrightarrow{rot}_{XFm} = \begin{bmatrix} rx & ry & rz \end{bmatrix}^T$ définit l'axe de rotation,

et $\|\overrightarrow{rot}_{XFm}\|$ correspond à la valeur de l'angle de rotation en radians.

Les principaux avantages de la représentation par vecteur de configuration sont sa non redondance (exactement six paramètres), l'absence de singularités et la possibilité de décrire des rotations supérieures à 2π. Ce dernier point permet notamment de modéliser simplement des liaisons hélicoïdales et des systèmes rotatifs.

Ce choix de représentation n'est en réalité pas fortuit. Ses propriétés agréables de non redondance et de régularité ont rendu possible le développement d'un logiciel pouvant traiter, avec une robustesse importante, n'importe quel mécanisme complexe. D'autres paramétrages qui peuvent s'avérer plus performant (car plus concis) dans un contexte de modélisation spécifique[2] sont simplement inadéquats car ils engendrent des singularités dans le traitement automatique de problèmes plus généraux.

Vitesse relative de deux corps

Le mouvement de R_m par rapport à R_F est décrit, en un point référence O, par le torseur cinématique noté : $V^O_{XFm} = \begin{bmatrix} \overrightarrow{v}^O_{XFm} \\ \overrightarrow{\omega}_{XFm} \end{bmatrix}$ où $\overrightarrow{v}^O_{XFm}$ et $\overrightarrow{\omega}_{XFm}$ désignent respectivement la vi-

[2]par exemple, l'utilisation des paramètres de Denavit-Hartenberg pour la modélisation de robots à structure série.

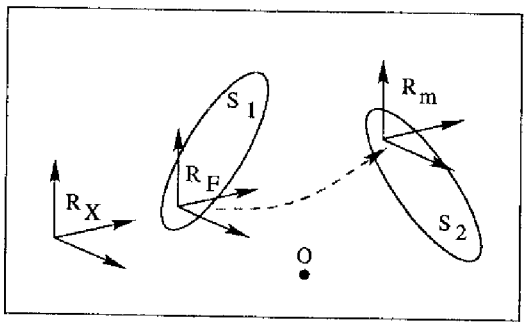

Figure 2.4: Situation relative de deux corps.

tesse de translation et de rotation des deux référentiels [GUINOT 92]. Pour conserver l'ordre dans lequel apparaissent la translation et la rotation dans un vecteur de configuration, nous présenterons le distributeur cinématique du mouvement dans l'ordre (moment,résultante). Comme précédemment, on précise le repère de calcul R_X dans lequel s'expriment ces vecteurs.

Par abus de langage, V^O_{XFm} pourra aussi désigner le vecteur vitesse généralisée de $\Re^6$ formé du bi-vecteur $(\overrightarrow{v}^O_{XFm}, \overrightarrow{\omega}_{XFm})$. Il existe une relation linéaire instantanée entre ce vecteur vitesse généralisée et la dérivée du vecteur de configuration correspondant (voir annexe 5.3).

Comme nous l'avons dit plus haut, la description de systèmes mécaniques généraux demande de raisonner sur les objets tridimensionnels qui le constituent, notamment pour la mise en place de nombreux référentiels. Le paragraphe suivant présente les principales caractéristiques du système de visualisation 3D intégré à l'environnement de spécification de tâches.

2.3 Modélisation graphique d'un système mécanique et de son environnement

Ce paragraphe expose certains principes et caractéristiques de fonctionnement du modeleur 3D disponible dans l'environnement de programmation de robots (Environnement de Programmation de Robots Manipulateurs que nous désignerons désormais par l'abréviation EPRM).

2.3.1 Description topologique - Corps polyédriques

TRID est le nom du modeleur d'objets polyédriques 3D intégré à l'EPRM. Les objets de base disponibles sous une forme générique sont le parallélépipède rectangle, le cylindre à base circulaire, la pyramide droite à base polygonale et la sphère. Un ensemble d'opérateurs sur les corps 3D (union, intersection, différence, homothétie...) permet à l'utilisateur de créer sa propre bibliothèque d'objets.

Les topologies des solides à représenter sont constituées de listes de sommets, d'arêtes et de faces. Un corps polyédrique est défini à partir des trois éléments suivants :

- une topologie,

- un référentiel de l'espace propre à l'objet appelé *repère intrinsèque* (rki),
- une liste de triplets représentant les coordonnées de chaque sommet du corps considéré, exprimées dans le repère précédent.

2.3.2 Positionnement des corps - Graphe de repères

La position relative de deux solides A et B dans l'espace est déterminée si l'on connaît la transformation géométrique existante entre les repères intrinsèques rki_A et rki_B associés à chacun des corps. La stratégie de positionnement des objets dans le modèle 3D repose sur l'utilisation d'un graphe. Les nœuds de ce graphe sont des repères intrinsèques et ses arcs sont des matrices de passage.

Les objets impliqués dans la modélisation s'insèrent dans le graphe des repères sous la forme d'une arborescence dont la racine est une entité appelée *scène graphique*.

Ce choix de représentation permet de décrire de manière naturelle l'état géométrique de n'importe quel mécanisme. Si ce dernier comporte des cycles, nous le représenterons par une topologie arborescente équivalente, les contraintes de fermeture de boucles étant supposées satisfaites. En conséquence, l'arborescence des repères offre à l'utilisateur la possibilité de connaître simplement la situation de n'importe quels objets de la scène: la position relative entre deux corps A et B est évaluée par composition des matrices de passage en parcourant le graphe entre les nœuds rki_A et rki_B. Ce calcul est accompli de manière transparente pour l'utilisateur.

Pour que la position calculée soit définie de façon unique, un seul chemin doit exister entre deux nœuds du graphe. L'unicité de ce chemin est garantie si le graphe des repères est une arborescence, autrement dit, s'il est à la fois connexe et dépourvu de cycles. Les manipulations sur le graphe des repères devront donc impérativement préserver la propriété précédente.

2.3.3 Scène graphique - Visualisation

La visualisation graphique proposée par TRID est organisée hiérarchiquement conformément à la figure 2.5.

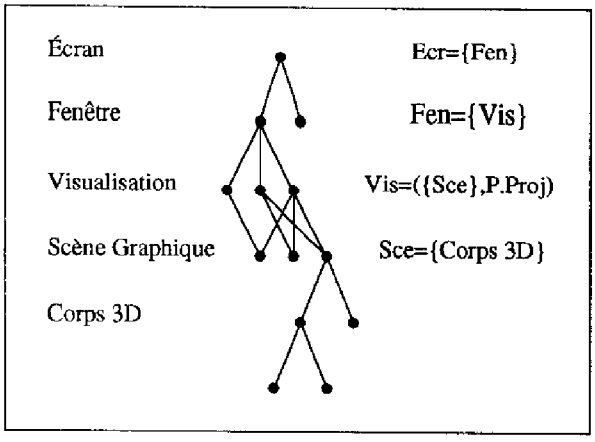

Figure 2.5: Architecture du modeleur 3D.

Tout d'abord, une scène graphique constitue la racine d'une arborescence de corps polyédriques. Un élément de visualisation est formé d'un ensemble de scènes graphiques et d'un jeu de propriétés projectives dans lequel sont définis le point d'observation de la scène et la projection utilisée (cylindrique, conique,...). Une fenêtre graphique est composée d'un ensemble d'éléments de visualisation. Enfin, l'écran de la station de travail peut comporter un nombre quelconque de fenêtres graphiques.

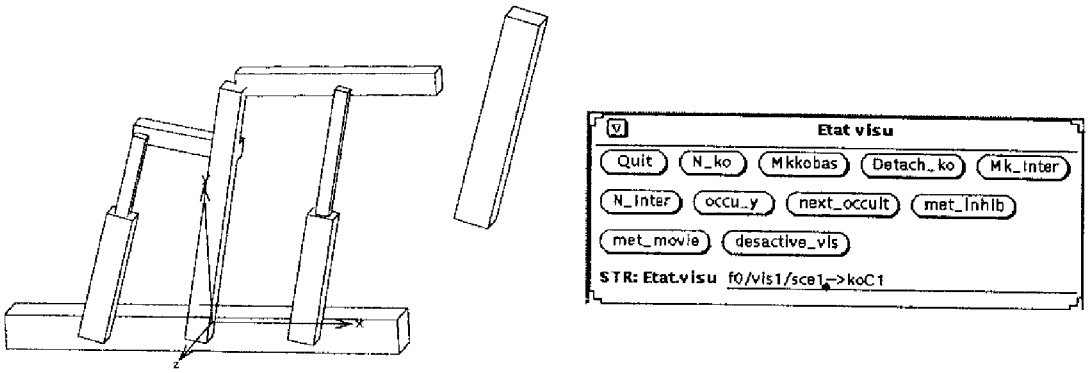

Figure 2.6: Modèle graphique 3D du manipulateur plan.

La figure 2.6 donne un aperçu du modèle 3D, élaboré pour représenter le bras à deux degrés de liberté évoqué plus haut. Sur cette figure apparaît également une fenêtre permettant de passer en revue interactivement toutes les entités attachées à la visualisation graphique.

2.4 Spécification par contraintes de contact

Disposant maintenant du modèle 3D d'un mécanisme, nous souhaiterions décrire ses propriétés de mobilité ainsi que ses interactions avec d'autres objets de l'environnement. La plus grande partie de ces interactions peuvent être interprétées comme des contacts, ce terme étant pris dans son sens le plus général.

Plus précisément, un contact entre deux objets désigne la relation binaire caractérisant le positionnement relatif des objets considérés. Cette définition permet de désigner par le même terme la notion habituelle de contact physique vrai et le contact fictif ou virtuel qui définit simplement une contrainte de placement ou de distance non nulle entre deux objets.

La description du contact d'un objet avec un plan fait intervenir les référentiels de positionnement relatif des deux objets mais aussi des référentiels auxiliaires attachés aux corps qui permettent de caractériser la relation de contact (figure 2.7a). Ces référentiels de travail peuvent être fixes sur les objets comme dans le cas du contact sur un plan, ou bien mobiles, comme par exemple dans l'étude du mouvement d'un cercle roulant sans glisser sur une droite (figure 2.7b).

Pour assurer la visualisation des corps 3D, le modeleur tient à jour une base de données. Cette gestion est assurée au moyen de l'arborescence des repères intrinsèques et répond au principal besoin de la visualisation, à savoir, l'accès aux positions relatives de chaque objet. L'installation des référentiels de travail doit se faire sur la base de cette arborescence de repères. C'est pourquoi il nous faut définir une modalité d'extension du graphe des repères

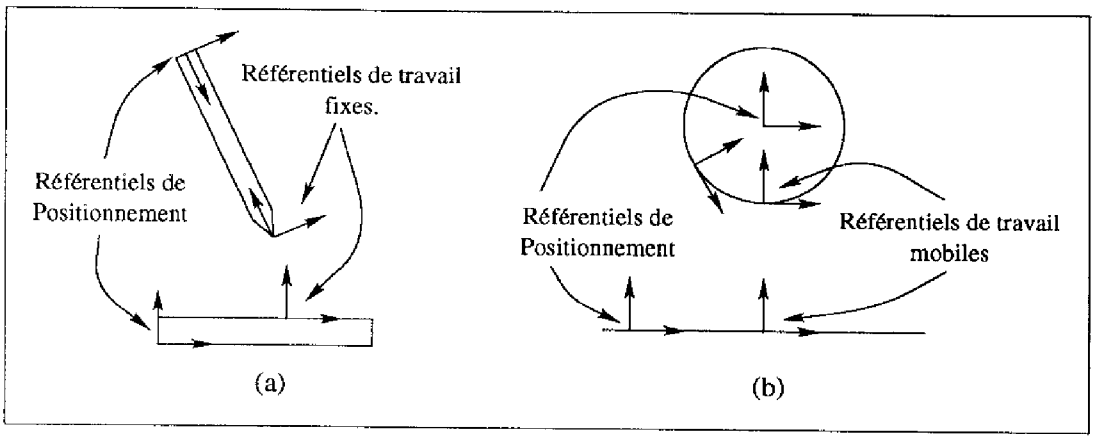

Figure 2.7: Nécessité de repères de travail pour la description de contacts.

qui permette à l'utilisateur de positionner des référentiels sur les corps tout en préservant sa cohérence.

2.4.1 Concept de demi-contact

La modélisation des systèmes que nous proposons met en œuvre un raisonnement cinématique. La description de telles propriétés fait appel à une utilisation massive de référentiels. Ces derniers servent en premier lieu à positionner les corps constituant le système étudié mais interviennent également dans la définition des mobilités des liaisons mécaniques entre ces corps. Le concept de *demi-contacts* permet de définir des référentiels sur les corps en complétant le graphe de repères sous-jacent.

À tout repère intrinsèque de corps correspond un demi-contact racine (*dcRacine*) à partir duquel peut être installée une arborescence de demi-contacts ordinaires (*dc*) conformément à la figure 2.8. Ils comportent essentiellement trois champs (figure 2.9) :

- un repère de demi-contact de base (*rdcBase* α),
- un repère de demi-contact (*rdc* β),
- et un vecteur déplacement $d_{\alpha\beta}$.

Les repères de demi-contact précédents sont des nœuds de l'arborescence de repères mis en place comme indiqué dans le tableau 2.1.

	dcRacine	**dc**
rdcBase	rki du corps	rdc(dc_{pere})
$\mathbf{d}_{\alpha\beta}$	déplacement nul	valeur non nulle (calculée ou imposée par l'utilisateur)
rdc	confondu avec rki	rdcBase(dc_{fils})

Tableau 2.1: champs des demi-contacts

Il apparaît donc le niveau de description « demi-contact » qui gère implicitement le niveau « graphe de repères » et en particulier le maintien de l'arborescence.

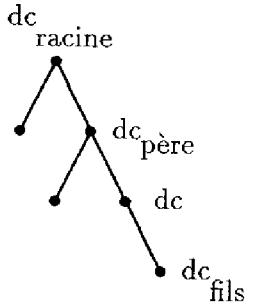

Figure 2.8: Arborescence de demi-contacts.

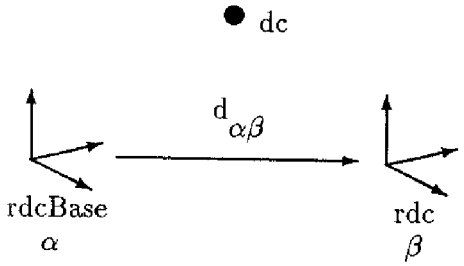

Figure 2.9: Structure d'un demi-contact.

Les demi-contacts correspondent à des nœuds du graphe cinématique. Nous pouvons maintenant installer sur les corps tout référentiel utile à la modélisation par l'intermédiaire des demi-contacts. La spécification de leur mobilité relative constitue notre prochain objectif.

2.4.2 Concept de lien - Expression des mobilités relatives

Un lien désigne l'arc existant entre deux demi-contacts. Il définit à la fois la position relative et la nature de la mobilité entre les demi-contacts. Nous distinguerons deux situations :

Cas 1 :

Les évolutions d'un demi-contact dc1 par rapport à son père dc0 peuvent être décrites comme des combinaisons de mouvements simples de translation et rotation définis dans le référentiel dc0. Pour ce type de lien, il est possible de décrire la mobilité des deux repères en qualifiant les six composantes du vecteur de configuration qui les positionne. Tous les types de liaisons mécaniques bilatérales ou unilatérales entre deux solides relèvent de cette catégorie.
La figure 2.10a illustre la qualification d'un mouvement plan sur plan. dc1 évolue suivant deux translations selon $\vec{x}$ et $\vec{y}$ et une rotation autour de $\vec{z}$, représentées par des fonctions x(t), y(t) et rz(t) indépendantes.

Cas 2 :

Les évolutions des composantes du vecteur de configuration Q_{01} de dc1 par rapport à dc0 ne sont pas découplées. Cette situation survient, par exemple, lorsque la position (et par suite la vitesse) de dc1 découle des positions de corps intermédiaires. Cette catégorie de lien doit définir la mobilité par un modèle géométrique.
La figure 2.10b décrit le cas où dc1 définit la position du corps terminal d'un mécanisme relativement à dc0. Le vecteur Q_{01} représente le modèle géométrique du système considéré.

L'implantation logicielle des liens de seconde espèce ne nous a pas paru prioritaire étant donné que la première catégorie de liens suffit déjà à traiter un grand nombre de problèmes et en particulier le cas fondamental des liaisons entre deux solides. Aussi, dans ce qui va suivre, nous nous intéresserons essentiellement à ce dernier type de lien que nous appellerons parfois « lien cartésien » par opposition au lien à modèle géométrique.

La figure 2.11 illustre le cas élémentaire de deux demi-contacts successifs. Au demi-contact dc_{i+1}, fils de dc_i, est associé le lien L_{i+1} reliant dc_i à dc_{i+1}. Le lien orienté L_{i+1}

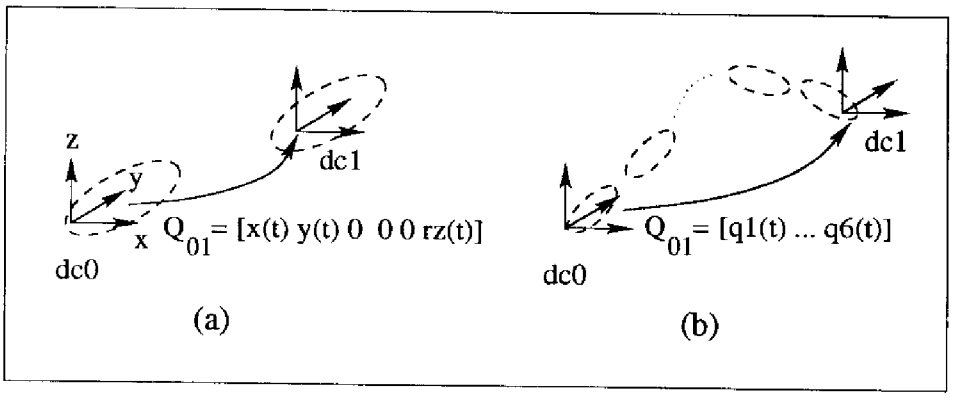

Figure 2.10: Lien ordinaire et à modèle géométrique.

est tout d'abord support de l'information sur la position relative du repère rdc_{i+1} par rapport à rdc_i. Cette position est représentée par le vecteur de configuration $d_{i+1} = \begin{bmatrix} tx & ty & tz & rx & ry & rz \end{bmatrix}^T$ qui définit l'état géométrique des deux repères.

L'expression de la mobilité relative des deux demi-contacts est assurée par un vecteur d'étiquettes qualifiant les composantes de d_{i+1}. Ces étiquettes renseignent sur les *objectifs d'évolution* des composantes géométriques précédentes.

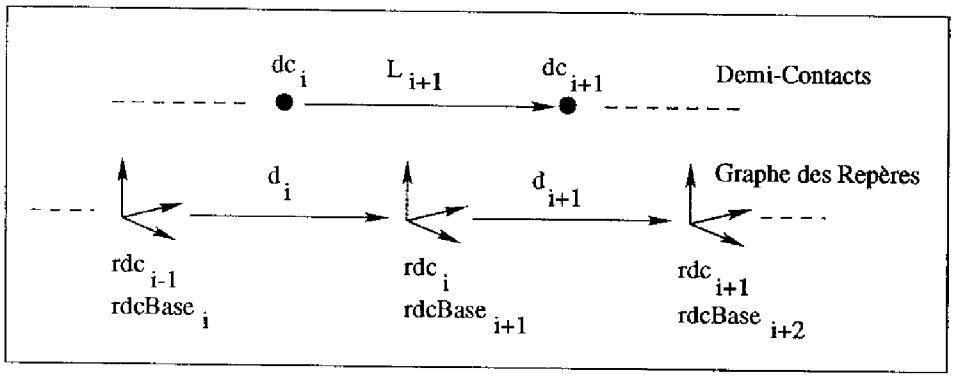

Figure 2.11: Organisation d'un lien.

Quatre grandes catégories de variables sont apparues nécessaires pour caractériser, de la manière la plus générale, la nature des composantes du vecteur de configuration. Une composante de ce vecteur pourra être :

- un paramètre de valeur connue et constante dans le temps,
- une variable inconnue à déterminer,
- une variable de contrainte que l'on souhaite voir satisfaire une certaine relation (égalité, inégalité),
- ou enfin, une variable dont la valeur est indifférente.

Il peut être intéressant, pour l'utilisation pratique, d'affiner cette classification et distinguer par exemple un paramètre nul d'un paramètre non nul. La codification choisie utilise

pour chaque composante un caractère alphabétique. Le tableau 2.2 présente quelques étiquettes de variables.

Étiquette	Interprétation
N,P	Paramètre nul et non nul
Q	Variable inconnue (articulaire)
C	Variable de contrainte (à annuler)
R	Variable intermédiaire (valeur indifférente)

Tableau 2.2: Exemples d'étiquettes de variables.

Quelques exemples de qualification de liaisons mécaniques entre deux demi-contacts sont présentés dans le tableau 2.3

Qualification	Nature du lien
PPP PPP	lien fixe (sur un même corps)
QNN NNN	lien articulaire : glissière d'axe $\vec{x}$
CCC RRR	lien de contact : contact ponctuel, orientation indifférente
NNN RNN	lien permettant une mobilité de rotation autour de $\vec{x}$

Tableau 2.3: Exemples de qualification de liaisons.

On peut remarquer que ce moyen de spécifier des objectifs d'évolution renseigne directement sur la mobilité relative des demi-contacts considérés et convient de ce fait pour la construction du graphe cinématique. Cette technique permet, de plus, de différentier au moment de la résolution les variables inconnues, les variables paramétriques, intermédiaires ou de contrainte.

2.4.3 Concept de forme

Les objets au milieu desquels un robot est amené à évoluer peuvent avoir des topologies variées. Afin de décrire les interactions entre le robot et cet environnement nous devons pouvoir modéliser le positionnement de référentiels sur la surface de ces corps. De plus, pour les besoins de la résolution des contraintes, il sera important de disposer d'un modèle linéaire tangent, en chaque point de ces surfaces.

Appelons S la surface d'un objet K positionné à l'aide d'un demi-contact de référence dc_r. Nous voudrions placer à sa surface un demi-contact terminal dc_t tel que son vecteur $\vec{z}$ constitue la normale extérieure à S. La *forme* associée au demi-contact de référence dc_r définit la procédure à suivre pour positionner et déplacer le demi-contact terminal dc_t sur la surface de l'objet considéré.

Cette définition présente une analogie avec le concept de C-surface utilisé par MASON [MASON 82] dans le cadre de la commande par retour d'efforts et du contrôle hybride. Dans le cas d'une tâche d'insertion de deux objets, la C-surface relative à cette tâche définit l'ensemble des points pour lesquels les deux objets sont en contact. Dans ces conditions, le seul mouvement autorisé est suivant l'hyperplan tangent à la C-surface alors que le contrôle

des efforts a lieu selon la normale à la surface. Contrairement aux C-surfaces qui sont définies dans l'espace des configurations, le concept de forme décrit les propriétés surfaciques des corps dans l'espace physique tridimensionnel.

Nous distinguerons deux situations possibles : la première correspond à des objets modélisés à l'aide de primitives géométriques simples de type polyèdre, cylindre, sphère ou cône. Ce cas de figure correspond à la majorité des objets présents dans des environnements structurés et conçus par l'homme. Il est alors possible de décrire, grâce à une forme, la transformation de dc_r à dc_t comme une suite linéaire de liens cartésiens entre des demi-contacts mobiles intermédiaires. Le modèle du mouvement tangent peut alors être obtenu par composition des vitesses le long de la cascade de repères.

L'autre situation correspond aux objets devant être décrits par des modèles plus complexes. La modélisation d'environnements naturels fait appel à une large variété de représentations. Les modèles par quadriques ou superquadriques constituent deux exemples de primitives de haut niveau utilisées pour définir la surface de tels objets. La forme devra dans ce cas définir la position (resp. la vitesse) du demi-contact dc_t au moyen d'un modèle géométrique (resp. cinématique).

Pour des raisons déjà évoquées lors de la présentation des liens à modèle géométrique, l'implantation de ce type de forme n'a pas été développée.

Exemples

Forme Cylindre

On se propose de modéliser un cylindre de révolution à base circulaire, de rayon R. Conventionnellement, le demi-contact de référence dc_r attaché au cylindre sera choisi tel que son vecteur $\vec{x}$ soit parallèle à l'axe du cylindre. La figure 2.12 représente la cascade de demi-contacts mis en place par la forme cylindre et le tableau 2.4 précise les qualificatifs de mobilité des liens intermédiaires.

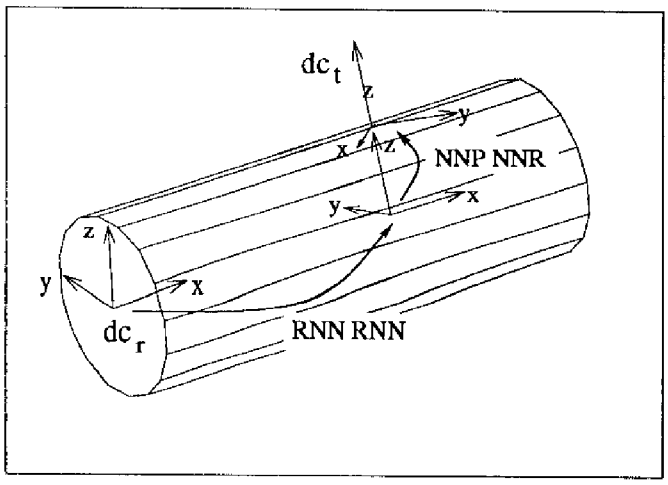

Figure 2.12: Forme cylindre

La vitesse du demi-contact terminal dc_t positionné sur le cylindre par rapport à dc_r

Forme cylindre	Mobilité	Observation
Lien 1	RNN RNN	translation et rotation selon $\vec{x}$
Lien 2	NNP NNR	translation et rotation selon $\vec{z}$ (la variable P représente le rayon R)

Tableau 2.4: Qualificatifs de mobilité de la forme cylindre.

peut être calculée par composition des vitesses relatives des repères de demi-contact intermédiaires.

Forme Cône

Considérons à présent la modélisation d'un cône à base circulaire et de demi-angle au sommet α. Le demi-contact dc_r est situé au sommet de sorte que son vecteur $\vec{x}$ détermine l'axe du cône. Les deux liens suivants permettent de définir une génératrice quelconque sur le cône et le dernier lien positionne un point sur cette génératrice. La figure 2.13 représente la cascade de repères et le tableau 2.5 détaille les qualificatifs de mobilité.

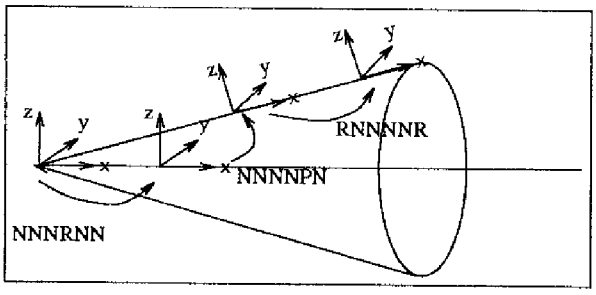

Figure 2.13: Forme cône

Forme cône	Mobilité	Observation
Lien 1	NNN RNN	rotation autour de $\vec{x}$
Lien 2	NNN NPN	rotation axe $\vec{y}$ (angle α)
Lien 3	RNN NNR	translation selon $\vec{x}$ et rotation selon $\vec{z}$

Tableau 2.5: Qualificatifs de mobilité de la forme cône.

Comme précédemment, la vitesse du repère terminal sur le cône par rapport au repère de référence sera calculée par composition des vitesses relatives des repères intermédiaires.

Forme Ellipsoïde

Contrairement à une sphère, une forme ellipsoïdale ne peut pas être décrite par une

une suite linéaire de liens, principalement, en raison de la présence des coefficients d'affinité suivant les directions principales de cette quadrique.

La position et le mouvement tangent du demi-contact terminal dc_t sur cette surface doivent être décrits explicitement à l'aide de modèles géométrique et cinématique particuliers.

2.4.4 Concept d'articulation

Il fait référence à la notion classique de liaison mécanique entre deux solides. Une articulation s'appuie sur deux demi-contacts appartenant à deux corps distincts. Le lien reliant ces demi-contacts, appelé *lien articulaire*, spécifie la position et la mobilité relative des deux corps.

Considérons deux solides $S1$ et $S2$ reliés par une articulation. Sur la figure 2.14 sont représentés les demi-contacts mis en jeu, ainsi que le graphe de repères sous-jacent.

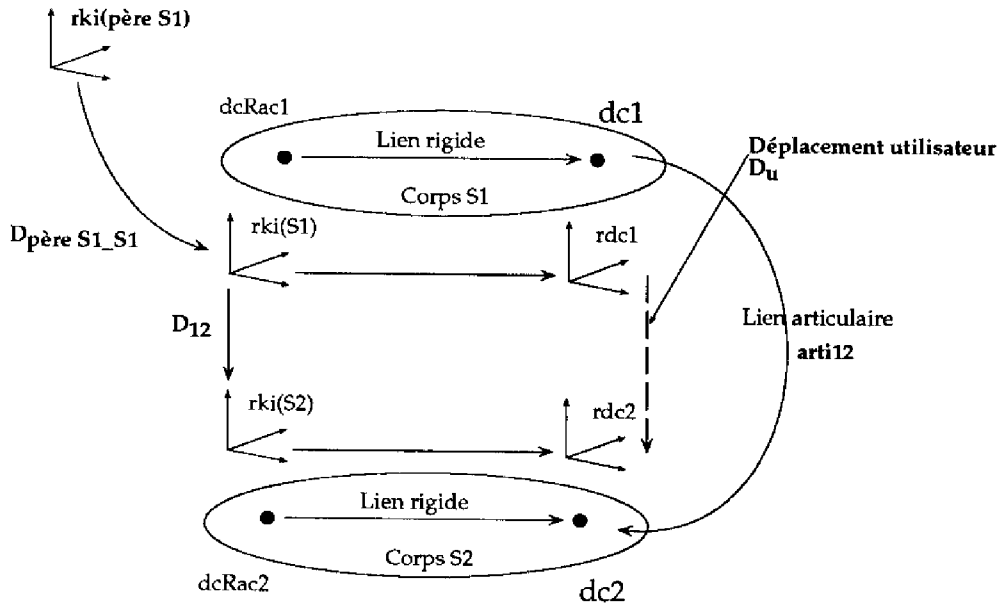

Figure 2.14: Liaison articulaire et graphe de repères.

L'articulation notée $arti12$ s'appuie sur les demi-contacts $dc1$ et $dc2$. Le lien articulaire associé précise la nature de la mobilité à satisfaire entre $S1$ et $S2$ ainsi que la valeur du déplacement utilisateur D_u à produire entre les repères de demi-contact $rdc1$ et $rdc2$. Enfin, l'orientation indique le corps de référence par rapport auquel doit se déplacer l'autre solide.

Le maintien de la cohérence du graphe de repères impose que $S2$ soit fils de $S1$.

Dans cet exemple, le positionnement de $S2$ par rapport à $S1$ conformément à D_u va modifier le graphe de repères en recalculant le déplacement D_{12} entre les repères intrinsèques de $S1$ et $S2$ pour avoir D_u entre $rdc1$ et $rdc2$.

En effet, il ne peut pas exister d'arc ($rdc1 \rightarrow rdc2$) dans le graphe des repères sous peine d'obtenir un cycle.

2.4.5 Concept de contact

Le concept d'articulation présenté plus haut constitue le moyen de modéliser une liaison mécanique simple entre deux corps, à l'aide d'un seul vecteur de configuration. Le « contact » est le concept par lequel l'utilisateur peut spécifier des contraintes complexes de positionnement entre plusieurs solides. Il est défini à partir des éléments suivants :

1) deux demi-contacts de base portés par deux corps distincts,
2) deux formes s'appuyant sur les demi-contacts précédents,
3) un lien de contact définissant le type de contrainte à satisfaire,
4) une procédure de calcul servant à la mise en place initiale des différents repères de demi-contact intermédiaires.

Exemple : contact droite sur droite.

Prenons le cas d'un cube $C2$ que l'on souhaite positionner par rapport à un autre cube $C1$, en exprimant la contrainte géométrique suivante :

« *deux arêtes $Ar1$ et $Ar2$, de $C1$ et $C2$ respectivement, forment un angle α et sont distantes de d* ». (Figure 2.15)

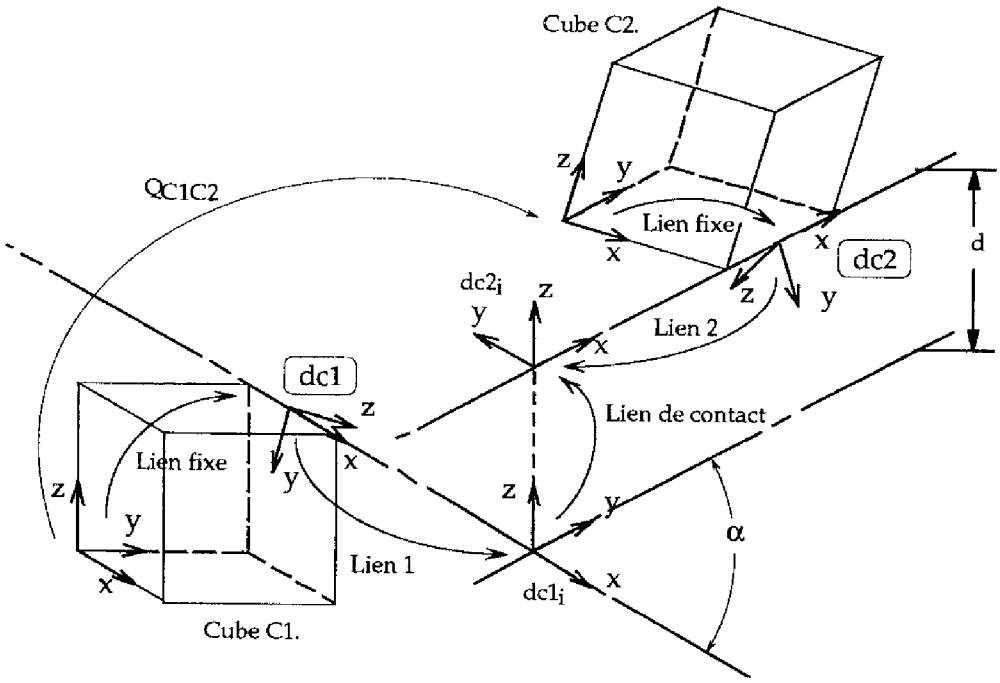

Figure 2.15: Contact arête sur arête.

Nous venons de décrire géométriquement la relation de contact que l'on souhaite établir entre $C1$ et $C2$. Dans un premier temps, les deux demi-contacts $dc1$ et $dc2$ ont été positionnés sur les arêtes $Ar1$ et $Ar2$ respectivement, au moyen de liens rigides. Par convention, les

droites support de ces arêtes sont portées par les axes $\vec{x}$ des repères de demi-contact. Le contact de type «droite sur droite» que l'on souhaite établir, va être défini en précisant :

- le premier demi-contact $dc1$ sur lequel va s'appuyer une forme «droite»,
- le second demi-contact $dc2$ support d'une autre forme «droite».

Étiquette	Interprétation
D	Variable de contrainte nulle, non modifiable
U	Variable de contrainte utilisateur à valeur non nulle

Tableau 2.6: Étiquettes de variables complémentaires.

Contact droite/droite	Mobilité	Observation
Lien 1	RNN RNN	$rdc1_i$ par rapport à $rdc1$
Lien de contact	DDU DDU	$rdc2_i$ par rapport à $rdc1_i$
Lien 2	RNN RNN	$rdc2_i$ par rapport à $rdc2$

Tableau 2.7: Description des liens du contact droite/droite.

Les formes attachées au contact vont installer deux demi-contacts mobiles intermédiaires $dc1_i$ et $dc2_i$. Connaissant, grâce au vecteur de configuration Q_{C1C2}, la position relative des deux cubes, la procédure de calcul initiale du contact installe automatiquement les deux demi-contacts $dc1_i$ et $dc2_i$ (ici, il s'agit du positionnement des pieds et du support d'une normale commune).

Pour les besoins de l'illustration, le tableau 2.6 introduit quelques labels de mobilité supplémentaires et le tableau 2.7 résume les qualificatifs des liens intervenant dans ce contact.

Extensions du concept de contact

Relation linéaire entre composantes de vecteurs de configuration Une relation linéaire de contact f permet de spécifier une dépendance entre des composantes de mobilité. Soient A et B deux liens relatifs aux couples de demi-contacts (A_0, A_1) et (B_0, B_1). Il est possible de définir des relations, notées symboliquement $A = f(B)$, telles que $c_A = kc_B$ où k est un cœfficient réel et c_A, c_B représentent chacun une des six composantes des liens A et B.

Exemple : liaison hélicoïdale On désire positionner un demi-contact $dc2$ sur une hélice de rayon r et d'angle α ($\alpha > 0$), les axes $\vec{x}$ et $\vec{z}$ de $dc2$ étant respectivement tangent et normal au cylindre support de l'hélice (figure 2.16). Ce cylindre peut être modélisé à l'aide d'une forme de type cylindre s'appuyant sur le demi-contact de base $dc1$. La liaison hélicoïdale sera décrite si l'on sait positionner sur le cylindre, de façon adéquate, le repère $dc1_2$. Pour cela, une solution consiste à installer une relation linéaire de contact entre les composantes R du lien RNN RNN. Si l'on note a et b ces composantes, la relation linéaire

est de la forme $a = kb$ dans laquelle $k = \frac{r}{\tan\alpha}$. Dans ces conditions, les paramètres P du lien suivant NNP NNP désignent respectivement le rayon r et l'opposé de l'angle $-\alpha$. Finalement, le lien de contact entre $dc1_2$ et $dc2$ est qualifié CCC CCC. En d'autres termes, on astreint le demi-contact $dc2$ à demeurer confondu avec le demi-contact $dc1_2$ positionné sur l'hélice.

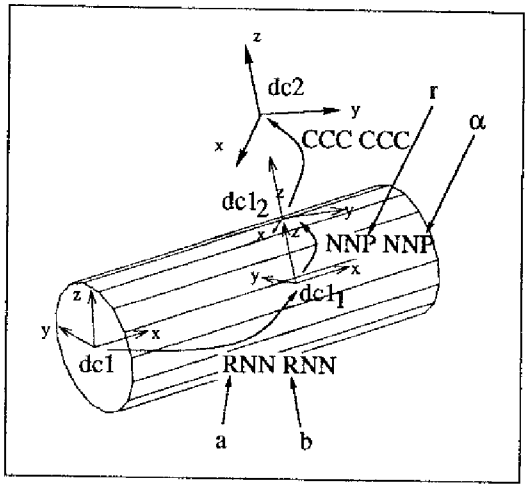

Figure 2.16: Description d'une hélice.

Par un raisonnement analogue, nous pourrions modéliser une « hélice » dont le support serait un cône. Ce concept peut être généralisé et appliqué à un nombre quelconque (supérieur à deux) de liens et de composantes de vecteur de configuration.

Construction de contact complexes Un contact sera qualifié de simple s'il ne fait intervenir qu'un seul lien de contact (contact droite sur droite, repère sur cylindre ...). Dans le cas contraire, nous dirons que le contact est complexe. L'exemple précédent appartient à cette catégorie puisqu'il associe deux contacts de type simple.

Plus généralement, un contact complexe pourra être une structure arborescente dans laquelle les nœuds correspondent à des contact simples. Nous verrons dans le chapitre 4 que la modélisation d'un contact projectif peut être réalisée de cette façon.

2.5 Concept de problème de mouvement

Dans ce paragraphe, nous allons évoquer, d'un point de vue général, les étapes de la construction d'un problème de mouvement. La mise en œuvre pratique et détaillée d'une telle spécification sera décrite en détail dans le chapitre 4.

Les éléments que nous avons décrits jusqu'à présent constituent les bases de notre système de spécification de tâches. Nous en sommes arrivés au stade de leur utilisation concrète pour décrire des tâches de robotique. La modélisation d'une tâche de déplacement, appelée aussi problème de mouvement, se déroule systématiquement suivant trois étapes :

- mise en place, sur les corps du modèle 3D, de tous les demi-contacts utiles à la description des propriétés cinématiques du mécanisme,
- description des mobilités entre les corps par des articulations,
- spécification des objectifs de la tâche et des contraintes structurelles du mécanisme (fermeture de boucles) au moyen de contacts.

Ces phases de la spécification engendrent trois sortes d'entités distinctes : un motif cinématique général (définissant les corps et demi-contacts), un motif cinématique articulaire (détermine souvent un jeu de variables d'état pour le système) et un motif cinématique de contraintes (fixe les objectifs à atteindre en termes de relations de contact).

Un problème de mouvement correctement modélisé sera constitué d'un ensemble de motifs cinématiques faisant apparaître obligatoirement les trois catégories d'entités. Cette hiérarchie permet de structurer le concept de problème de mouvement conformément à la figure 2.17.

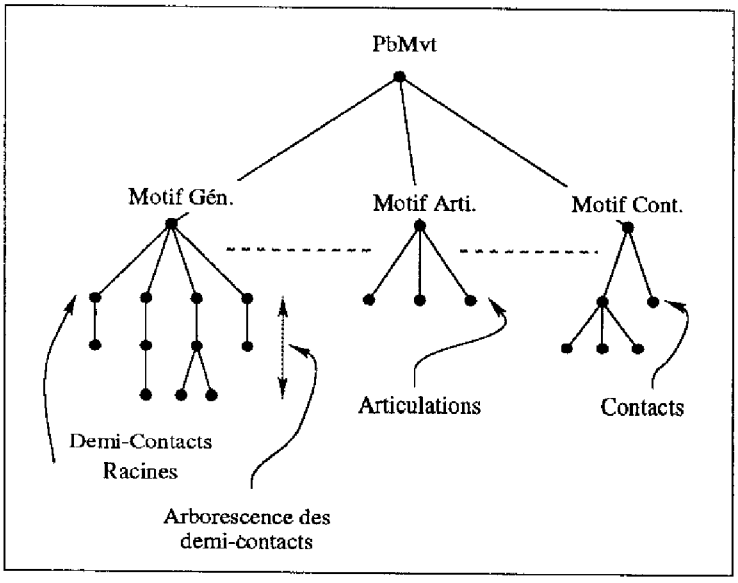

Figure 2.17: Structuration d'un problème de mouvement.

Lorsque le contexte le demande, il peut être pratique de constituer un problème de mouvement par concaténation de plusieurs problèmes de mouvement définissant partiellement l'objectif global. Cette conception modulaire offre une grande flexibilité et souplesse d'utilisation.

Une telle démarche pourra être adoptée pour la spécification de mise en contact du bras à deux degrés de liberté présenté au début de ce chapitre. La tâche : « *atteindre un point situé sur la face de l'objet 8* » se décomposera par exemple en deux sous-problèmes. Le premier aura pour fonction de fermer le mécanisme tandis qu'un motif de contraintes spécifiera l'objectif à réaliser (figure 2.18).

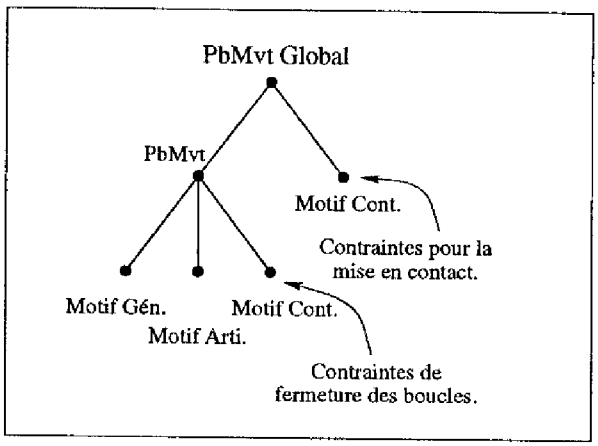

Figure 2.18: Problème de mouvement pour le manipulateur à deux ddl.

Nous exposerons, à la fin de ce chapitre, les modalités d'obtention des équations de contraintes à résoudre pour générer une trajectoire continue vers la configuration finale souhaitée.

2.6 Concept de problème de calibration

Le problème fondamental lié à l'utilisation d'un modèle est de savoir si cette représentation est conforme à la réalité. Pour répondre à cette question, il est nécessaire de confronter les informations, initialement présentes dans le modèle, avec le monde réel.

Au début de son travail, l'utilisateur est supposé disposer d'informations suffisantes pour mener à bien la construction d'un modèle 3D du système à étudier et de son environnement. Il aura par exemple recueilli ces informations sous la forme d'images perçues par des caméras observant le site de travail ou bien plus directement à partir de plans définissant les objets présents dans l'environnement. Le modèle résultant sera réputé fiable quant à la topologie des objets mais pas suffisamment précis quant aux position et taille de ces objets. Il faut donc disposer de moyens permettant d'améliorer ces informations.

Nous désignerons par le terme calibration toutes les activités permettant d'évaluer les différences existantes entre le modèle et la réalité qu'il est sensé représenter. Dans notre contexte de tâches de manipulation, la calibration fait référence aux problèmes d'identification de paramètres de position d'objets dans l'espace. Ce processus tient une place très importante dans la problématique de manipulation distante puisque le robot évolue dans un espace de travail mal connu *a priori*.

Dans l'approche proposée, la calibration s'associe de manière très étroite à la spécification des tâches. L'idée de départ consiste à exploiter les relations de contact établies entre les objets. Le concept de mesure permet de faire le lien entre la spécification de tâches de mouvement et la calibration.

2.6.1 Concept de mesure

Le terme de mesure désigne classiquement l'information fournie par un capteur ou tout autre instrument de métrologie. La commande de mécanismes équipés d'actionneurs nécessite, entre autres choses, la présence de capteurs mesurant l'état de ces systèmes. Dans le cas des robots manipulateurs, ce besoin se traduit par la présence, sur chaque articulation motorisée, de capteurs, généralement de type codeur angulaire, qui donnent accès aux positions de chaque liaison. C'est cette propriété que nous allons exploiter pour le traitement des problèmes de calibration en faisant jouer au manipulateur le rôle d'un instrument de mesure.

Plaçons-nous dans le contexte de la tâche de calibration élémentaire qui consiste à « *évaluer la position d'un plan* ». On se propose de réaliser cette tâche à l'aide d'un outil pouvant établir des contacts avec le plan à calibrer. Les objets en présence sont l'outil, le plan et un corps de référence. Les variables permettant la description de cette tâche de mouvement sont principalement de trois types : des variables inconnues correspondantes aux degrés de mobilité affectés à l'outil (fournis en général par un manipulateur), des variables de contraintes exprimant l'objectif à atteindre (contact avec le plan) et des variables paramétriques définissant la position de l'objet sur lequel portent les contraintes.

Pour les raisons qui apparaîtront plus loin, les inconnues du problème de mouvement et les paramètres à calibrer seront qualifiés respectivement par des étiquettes X et M. Dans ces conditions, la topologie du problème de mouvement est conforme au schéma 2.19a.

Supposons maintenant que l'on soit capable de placer l'outil dans une configuration où les contraintes de contact avec le plan sont satisfaites. La position de l'outil (définie par les inconnues du problème de mouvement) peut alors être interprétée comme une mesure caractérisant l'état du système en contact avec le plan. La répétition de cette procédure permet d'obtenir un jeu de mesures caractérisant plusieurs réalisations de la relation de contact entre l'outil et le plan. Examinons comment ces mises en contact peuvent être exploitées dans le cadre d'un problème de calibration.

2.6.2 Élaboration d'un problème de calibration

Un problème de calibration correspond à la forme duale du problème de mouvement associé. En effet, les inconnues, étiquetées X dans la spécification de mouvement, deviennent des paramètres étiquetés M définissant des mesures. Inversement, les liens paramétriques M du mouvement se transforment en inconnues X dans la calibration.

Considérons le cas d'une calibration à effectuer à partir de n mesures notées par simplicité M_i (figure(2.19b)). Lors de l'acquisition d'une mesure M_i, l'état de la relation de contact entre l'outil et l'objet est défini par la contrainte C_i. Les liens qualifiés X répétés dans chaque mesure sont en réalité relatifs à la même articulation; ils doivent donc être déclarés équivalents. De cette manière, la concaténation des n mesures va engendrer un nouveau graphe comprenant n cycles représenté sur la figure (2.19c)). La calibration a pour but d'évaluer les inconnues X de placement de l'objet de sorte que sa position minimise les erreurs sur chaque contrainte C_i.

Nous pouvons remarquer que la résolution de ce problème est de même nature que celle d'un problème de mouvement. En effet, cette formulation permet, comme dans le cas d'un problème de mouvement, d'engendrer un graphe duquel vont être extraites des équations

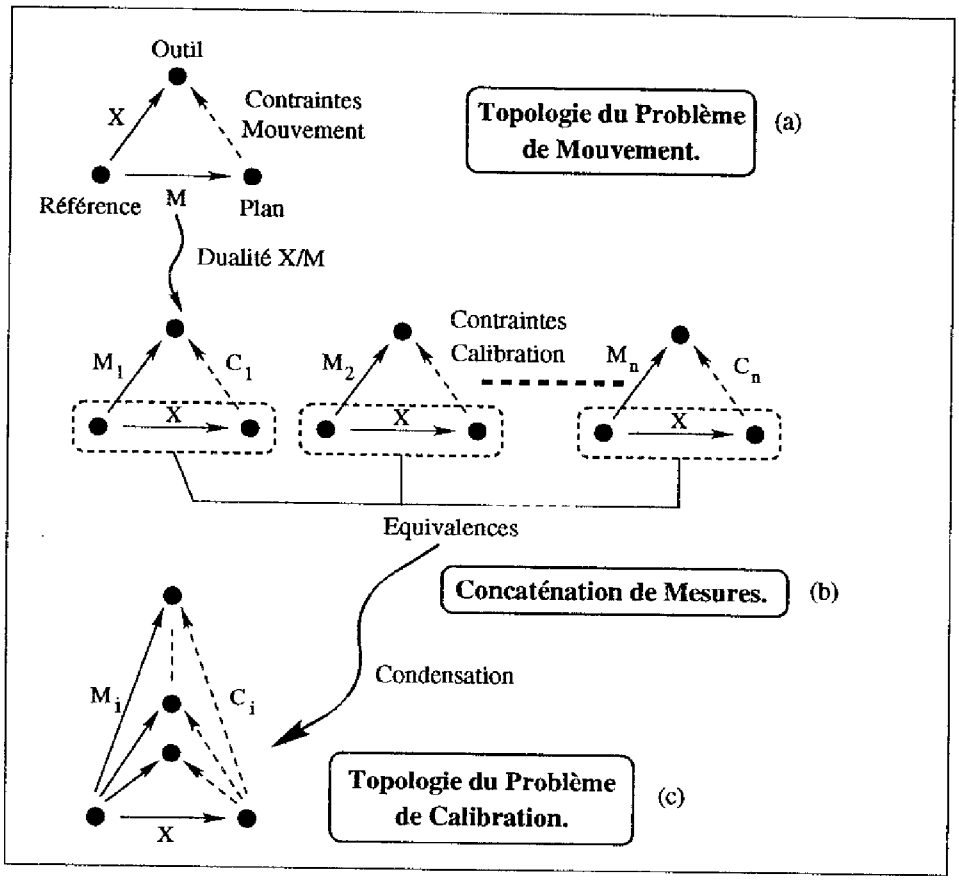

Figure 2.19: Genèse d'un problème de calibration.

de contraintes caractérisant l'objectif de la tâche. Leur traitement est pris en charge par la méthode de résolution.

Remarque :

Le précédent exemple ne faisait intervenir qu'un seul problème de mouvement (instancié n fois pour réaliser les mises en contact de l'outil avec le plan). Le problème de calibration correspondant est donc construit à partir de mesures homogènes.

Mais il est également possible de fusionner dans un seul problème de calibration, plusieurs problèmes de mouvements. Dans ce cas, les mesures à traiter sont hétérogènes. L'intérêt pratique de cette approche est de permettre la calibration simultanée de plusieurs liens géométriques.

Nous pouvons par exemple imaginer un scénario flexible pour la calibration de la position d'un cube, à partir de trois problèmes de mouvement définissant les mises en contact d'un outil avec trois faces orthogonales du cube (figure 2.20). Le problème de calibration global sera donc formé de l'association de trois séries de mesures $(Mes)_1$, $(Mes)_2$ et $(Mes)_3$.

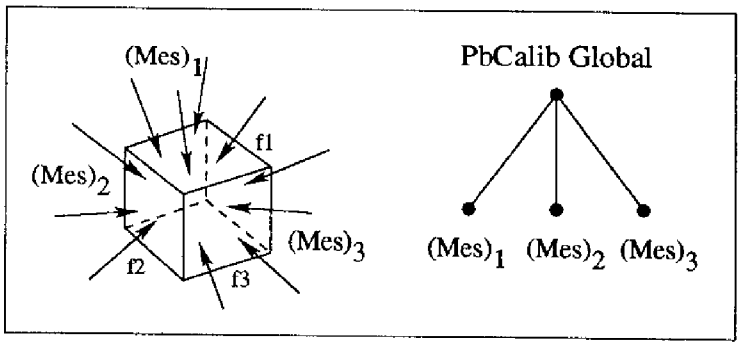

Figure 2.20: Calibration à partir de mesures inhomogènes.

2.7 Forme générale du résultat de la modélisation

Nous arrivons maintenant à l'étape terminale de la spécification qui a pour but de déduire des relations de contraintes à résoudre, en vue de satisfaire les objectifs de la tâche.

2.7.1 Graphe cinématique

La structuration des problèmes de mouvement comme un assemblage de motifs cinématiques nous fournit immédiatement le graphe structurel du système étudié. Les motifs cinématiques généraux décrivent, pour chacun des corps, le positionnement et la mobilité de tous les demi-contacts installés. Si k est le nombre de corps, les motifs généraux engendrent un graphe à k composantes arborescentes non connexes. Les motifs articulaires et de contraintes introduisent des liens supplémentaires permettant de connecter les composantes indépendantes du graphe précédent. Les équations de contraintes que nous recherchons proviennent de l'expression des conditions de fermeture sur les cycles du graphe obtenu. Le nombre nécessaire et suffisant de contraintes pour décrire correctement le problème posé correspond au nombre minimal de cycles permettant de représenter complètement le graphe. La dimension de cette base de cycles, appelée nombre cyclomatique, est égale à $m - n + p$ où m, n et p désignent respectivement le nombre d'arcs, de sommets et de composantes connexes du graphe [GONDRAN 90].

Cette décomposition minimale comporte plusieurs solutions en général. L'algorithme de recherche de cycles implantée dans le logiciel est basée sur une heuristique qui s'efforce de minimiser la longueur des cycles.

2.7.2 Obtention des équations de contraintes

Il existe deux voies possibles pour extraire des cycles les équations de contraintes. Les conditions géométriques de fermeture engendrent des équations complexes à traiter en raison de la présence de termes non linéaires (fonctions trigonométriques). En revanche, les conditions de fermeture provenant d'un raisonnement cinématique ont l'avantage de fournir des équations linéaires reliant les dérivées des variables géométriques. C'est donc par ce moyen que nous allons rechercher les équations de contraintes.

Soit un cycle C constitué de n demi-contacts, numérotés de 1 à n dans le sens direct de parcours du cycle (figure 2.21). Nous pouvons remarquer que :

- C comporte au moins trois demi-contacts,
- le nombre k de corps participant au cycle est au moins égal à deux,
- enfin, puisque chaque demi-contact est attaché à un solide, k est au plus égal à n.

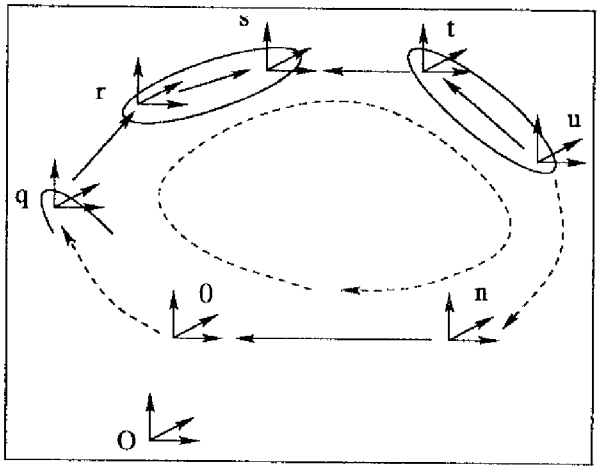

Figure 2.21: Cycle cinématique.

Propriété fondamentale :
L'ensemble des distributeurs (torseurs) de mouvement relatifs des différents solides d'une chaîne cinématique fermée, pris deux à deux dans toute permutation, forme un torseur nul.

Supposons que les n demi-contacts soient attachés à n solides virtuels distincts (dans ce cas, $k = n$). La relation de fermeture de C s'obtient en écrivant la loi de composition des vitesses des demi-contacts. Un point de référence O étant choisi, cette condition peut s'écrire sous la forme générale, indépendamment du repère de projection :

$$\sum_{i=1}^{n-1} V^O_{i,i+1} + V^O_{n,1} = 0 \quad \text{(Torseur nul)} \tag{2.1}$$

Cette relation donne lieu à deux groupes de trois équations scalaires correspondants respectivement à la résultante (vitesse de rotation) et au moment (vitesse de translation) du distributeur cinématique.

Soit (i, j) le lien orienté entre deux demi-contacts. Il existe une relation linéaire instantanée entre le distributeur cinématique et la dérivée du vecteur de configuration $\dot{Q}_{ij}$. Compte tenu du sens direct de parcours du lien (i vers j), la forme la plus simple de cette relation s'obtient en exprimant le torseur cinématique au point i :

$$V^i_{ij} = \begin{bmatrix} I_3 & 0 \\ 0 & J_{\omega_{ij}} \end{bmatrix} \dot{Q}_{ij} \tag{2.2}$$

I_3 désigne la matrice identité et la forme de la matrice $J_{\omega_{ij}}$ est donnée en annexe. Afin d'exploiter cette relation dans la relation générale (2.1), il est nécessaire de calculer, pour chaque lien, la vitesse au point qui convient. Si le lien (i, j) est parcouru dans le sens direct

(resp. indirect), la vitesse relative correspondante devra être calculée en i (resp. j). Par application de la relation classique de changement de point, nous obtenons les relations :

$$V_{ij}^{O} = \begin{bmatrix} I_3 & \widetilde{Oi} \\ 0 & I_3 \end{bmatrix} V_{ij}^{i} \qquad \text{Sens direct.} \tag{2.3}$$

$$V_{ij}^{O} = -\begin{bmatrix} I_3 & \widetilde{Oj} \\ 0 & I_3 \end{bmatrix} V_{ji}^{j} \qquad \text{Sens indirect.} \tag{2.4}$$

où $\widetilde{Ox} = \begin{bmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{bmatrix}$ désigne la matrice de préproduit vectoriel associée au vecteur $\overrightarrow{Ox} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$.

En injectant les relations (2.2), (2.3) et (2.4) dans l'équation (2.1), on obtient la relation générale :

$$\sum_{i=1}^{n-1} \left\{ s \begin{bmatrix} I_3 & \widetilde{Oi}.J_{\omega_{i,i+1}} \\ 0 & J_{\omega_{i,i+1}} \end{bmatrix} - \bar{s} \begin{bmatrix} I_3 & \widetilde{Oi+1}.J_{\omega_{i,i+1}} \\ 0 & J_{\omega_{i,i+1}} \end{bmatrix} \right\} \dot{Q}_{i,i+1} + \left\{ s \begin{bmatrix} I_3 & \widetilde{On}.J_{\omega_{n,1}} \\ 0 & J_{\omega_{n,1}} \end{bmatrix} - \bar{s} \begin{bmatrix} I_3 & \widetilde{O1}.J_{\omega_{n,1}} \\ 0 & J_{\omega_{n,1}} \end{bmatrix} \right\} \dot{Q}_{n,1} = 0 \tag{2.5}$$

où s représente une variable booléenne valant un, si le lien est parcouru dans le sens direct, zéro sinon.

En écrivant l'approximation au premier ordre de la dérivée du vecteur de configuration :

$$\dot{Q}_{i,i+1} \underset{1^{\text{er}}\text{ordre}}{=} \frac{\Delta Q_{i,i+1}}{\Delta t} \tag{2.6}$$

la relation de contrainte exacte (2.5) est traduite en sa formulation variationnelle (2.7) :

$$\sum_{i=1}^{n-1} \left\{ s \begin{bmatrix} I_3 & \widetilde{Oi}.J_{\omega_{i,i+1}} \\ 0 & J_{\omega_{i,i+1}} \end{bmatrix} - \bar{s} \begin{bmatrix} I_3 & \widetilde{Oi+1}.J_{\omega_{i,i+1}} \\ 0 & J_{\omega_{i,i+1}} \end{bmatrix} \right\} \Delta Q_{i,i+1} + \left\{ s \begin{bmatrix} I_3 & \widetilde{On}.J_{\omega_{n,1}} \\ 0 & J_{\omega_{n,1}} \end{bmatrix} - \bar{s} \begin{bmatrix} I_3 & \widetilde{O1}.J_{\omega_{n,1}} \\ 0 & J_{\omega_{n,1}} \end{bmatrix} \right\} \Delta Q_{n,1} = 0 \tag{2.7}$$

Nous observons que (2.7) est linéaire par rapport aux vecteurs incrémentaux ΔQ.

2.7.3 Forme du résultat de la spécification de tâches

Pour chaque cycle, la relation de contrainte est de la forme :

$$\sum_i M_i \Delta Q_i = 0_6 \tag{2.8}$$

Chaque composante de ces vecteurs est dotée d'une étiquette définissant son rôle vis à vis de la tâche spécifiée. Les variables pourront être des paramètres, des inconnues, des contraintes ou bien des variables à valeur indifférente. Compte tenu du nombre de liens présents dans le cycle, le nombre théorique maximal de variables est égal à $6 \times n$. Toutefois, toutes les variables paramétriques (dont la valeur reste constante) n'auront pas de contribution dans l'équation variationnelle et pourront de ce fait être éliminées des calculs dès le départ.

2.8 Conclusion

La technique de description de tâches que nous venons d'exposer dans ce chapitre constitue un outil général dédié à l'animation de systèmes mécaniques articulés.

La méthodologie présentée repose sur la spécification d'objectifs par contraintes, dans l'espace de la tâche. L'utilisation conjointe du modèle graphique 3D et du système d'étiquetage de variables autorise une grande richesse pour l'expression des tâches. De par sa généralité, cette démarche rend possible la spécification de tâches robotiques variées au nombre desquelles figurent la génération de mouvements complexes ou encore la calibration d'objets.

La représentation de ces problèmes sous la forme de graphes cinématiques permet de générer automatiquement les relations de contraintes qu'il faut résoudre pour accomplir les tâches souhaitées. La résolution des systèmes linéaires résultants fait l'objet du chapitre suivant.

Chapitre 3

Méthode de résolution

3.1 Introduction

De nombreuses tâches de robotique comme la génération de trajectoires ou l'identification de paramètres de position conduisent à la résolution de fonctions non linéaires éventuellement soumises à des contraintes inégalités et de problèmes d'optimisation sous contraintes.

Les problèmes auxquels nous nous intéressons correspondent à trois types d'objectifs. Pour le premier, l'utilisateur désire que le manipulateur rejoigne une position finale donnée, sans préciser d'autres directives. Cet objectif correspond au besoin de génération de grands mouvements ou mouvements de transfert.

Ensuite, à l'objectif précédent peuvent s'ajouter des contraintes à satisfaire lors du déplacement vers le but. Ces contraintes seront par exemple dues à la présence d'obstacles que le robot devra éviter pour atteindre la position voulue. Il s'agit donc là de générer des mouvements fins en présence d'obstacles.

Enfin, des mises en contact répétées du robot avec un objet de son environnement peuvent être exploitées pour identifier la position de cet objet. Le problème à résoudre est dans ce cas une minimisation de fonction, éventuellement sous contraintes.

Ces trois problèmes se formulent de la façon suivante :

$$\begin{cases} (a) \quad g(x)=0 & \text{Objectif de position.} \\ (b) \quad g(x^*)=0 & \text{Génération de trajectoire sous contraintes}^1. \\ \quad\quad \text{avec } h(x_n)\geq 0 \ , \ n\in[0,N] & \\ (c) \quad \min\limits_x f^T(x)f(x) & \text{Problème d'identification paramétrique.} \\ \quad\quad \text{avec } g(x)=0 & \end{cases} \tag{3.1}$$

où g, f et h sont des fonctions vectorielles.

Le vecteur d'état x définit généralement la configuration du manipulateur et des objets environnants, alors que la fonction g exprime les relations de contact à établir ou maintenir pour atteindre l'objectif recherché. h spécifie les contraintes d'évitement d'obstacles et f^Tf le critère à minimiser dans le cas d'un problème d'identification.

La première difficulté consiste à exprimer les fonctions g, f et h ainsi que leurs dérivées.

Dans le contexte qui nous intéresse, l'obtention d'une forme analytique de ces fonctions est le plus souvent exclue. En effet, les problèmes à traiter étant tridimensionnels, les équa-

[1] $\{x_n\}$ désigne l'ensemble des positions intermédiaires vers la configuration objectif x^*.

tions de contraintes qui en découlent sont très complexes et non résolubles formellement. De plus, le caractère non répétitif des problèmes à résoudre rendent prohibitifs les efforts de modélisation requis à chaque nouveau problème pour obtenir, quand c'est possible, une formulation compacte analytique.

Les objectifs de généralité et de flexibilité d'emploi sont les principales raisons qui ont motivé l'utilisation d'une méthode de résolution numérique.

Pour des raisons analogues de mise en forme, nous n'avons pas opté pour une utilisation de solveurs mathématiques permettant la résolution de problèmes d'optimisation de la forme (3.1) [MAT 92]. De plus, comme nous le verrons au cours de ce chapitre, la gestion des singularités et la reprise des situations d'échec dans le mécanisme de résolution sont fondamentaux pour assurer un niveau de robustesse élevé.

Ce cahier des charges impose donc une maîtrise des processus internes qui nous a conduit au développement d'un système de résolution adapté à la forme des relation de contraintes générées par la spécification de tâches.

3.2 Résolution de systèmes d'équations non linéaires

3.2.1 Principe général

Les méthodes itératives et en particulier celles basées sur le théorème dit, du point fixe, figurent parmi les techniques numériques les plus performantes permettant la résolution approchée d'équations de toute nature [DEMAILLY 91].

Leur principe général consiste à améliorer, partant d'une estimation initiale donnée, la précision de la solution par l'application répétée d'un algorithme bien choisi. Soit à résoudre, le système de m équations non linéaires suivant :

$$\begin{aligned} K &: \mathbb{R}^p \to \mathbb{R}^m \\ K(x) &= 0 \end{aligned} \tag{3.2}$$

Pour traiter ce système, on cherche à se ramener à la résolution d'un problème équivalent exprimé sous la forme $k(x)=x$, x étant un vecteur de dimension p. Sous certaines hypothèses que doit vérifier la fonction k [BARANGER 91], la suite $x_{n+1}=k(x_n)$ converge vers x^* solution de (3.2). Classiquement, le schéma itératif de recherche des solutions sera de la forme :

$$x_{n+1}=x_n+\Delta x_n \tag{3.3}$$

où l'incrément Δx_n pourra être évalué de multiples manières.

Méthode du gradient

En introduisant la fonction critère $K^TK : \mathbb{R}^p \to \mathbb{R}$, il est possible de calculer l'incrément à l'aide du gradient de K^TK :

$$\Delta x_n=-\alpha\frac{\partial K^TK}{\partial x}(x_n)$$

où α est un cœfficient réel positif.

Cette stratégie présente l'intérêt d'être simple à mettre en œuvre et assez peu coûteuse numériquement. En revanche, la convergence est médiocre près de la solution (le gradient

est quasiment nul). De plus, le bon fonctionnement de l'algorithme est conditionné principalement par le choix du cœfficient α : s'il est trop grand, il y a des problèmes de convergence; s'il est trop petit, la convergence devient très lente, au mieux asymptotique.

Ces limitations nous ont conduit à adopter une technique bénéficiant de meilleures propriétés de convergence.

Méthode de Newton-Raphson

Le principe de la méthode de Newton-Raphson est d'approcher K, au voisinage du point x_n, par son développement de Taylor au premier ordre :

$$K(x) \underset{1^{\text{er}}\text{ordre}}{=} K(x_n) + \left[\frac{\partial K}{\partial x}\right](x_n)\,(x - x_n) \tag{3.4}$$

Si la matrice jacobienne[2] $\left[\frac{\partial K}{\partial x}\right](x_n)$ est inversible, la solution de $K(x) = 0$, tirée de (3.4), permet de calculer le terme suivant :

$$x_{n+1} = k(x_n) = x_n - \left[\frac{\partial K}{\partial x}\right]^{-1}(x_n)\,K(x_n)$$

L'incrément Δx_n vérifie donc le système linéaire : $\left[\frac{\partial K}{\partial x}\right](x_n)\Delta x_n = -K(x_n)$. Une généralisation de l'algorithme de Newton-Raphson consiste à calculer Δx_n par une équation de la forme :

$$A(x_n)\Delta x_n = -\alpha K(x_n) \tag{3.5}$$

dans laquelle α est un gain inférieur ou égal à un et A une matrice $(m \times p)$.

Le principal avantage d'un algorithme de type Newton-Raphson est sa très bonne propriété de convergence à proximité de la solution (convergence quadratique). Ces conditions de convergence locale sont définies par un théorème dont l'énoncé rigoureux pourra être trouvé dans les ouvrages d'analyse numérique déjà cités.

Dans les problèmes de robotique que nous souhaitons résoudre, ces propriétés sont particulièrement appréciables. Elles permettent notamment d'engendrer des mouvements continus qui permettent, qualitativement, de rejoindre les objectifs selon « le plus court chemin » (autrement dit, de manière « radiale » dans l'espace des contraintes K).

Le choix d'utilisation d'une méthode de type Newton-Raphson, nous amène à discerner deux principales sources de difficultés sur le plan numérique. La première concerne la convergence globale de l'algorithme, la seconde est relative à l'inversion de la jacobienne A.

3.2.2 Convergence globale de l'algorithme itératif (3.3)

Les conditions de convergence globale sont généralement difficiles à déterminer. Cette convergence dépend en partie des propriétés de courbure de la fonction K qui pourra posséder plusieurs solutions. Dans ce cas, il est très délicat de connaître, à partir d'une condition initiale donnée, vers quel attracteur l'algorithme est susceptible de converger.

[2] Classiquement, la dérivée du vecteur $K(x)$ par rapport au vecteur x est une matrice B dont un élément $b_{i,j}$ est la dérivée partielle de la $i^{\text{ème}}$ composante de K par rapport à la $j^{\text{ème}}$ composante de x.

Sans apporter une réponse générale au problème de la convergence globale, nous utiliserons le schéma itératif (3.5) avec $A(x_n) = \frac{\partial K}{\partial x}(x_n)$ et $0 < \alpha \leq 1$. La relation (3.5) peut être interprétée comme l'équation de comportement d'un système dynamique linéaire de la forme $\dot{x} = s(x)$ donnant lieu à trois types d'évolutions :

1. convergence vers un point (attracteur) fixe,
2. évolution sur une surface limite,
3. divergence du système (instabilité).

La résolution discrète de ce type d'équation de comportement peut engendrer des évolutions chaotiques qui se manifestent, par exemple lorsque α est voisin de un, par des modifications locale de la stabilité de certains attracteurs (avec apparition/disparition d'attracteurs)[GUMOWSKI 70]. La modulation du gain α offre alors une possibilité de retrouver les propriétés de comportement de l'évolution continue.

Le cœfficient α sera choisi égal à un dans le cas où l'on souhaite une convergence « rapide » (au mieux quadratique). Des valeurs inférieures à un seront utilisées pour respecter des bornes sur les incréments de variables. Cette volonté de convergence « lente » correspond à des besoins pratiques évidents. A titre d'exemple, nous pouvons citer la limitation de l'amplitude des mouvements incrémentaux à générer sur un manipulateur.

3.2.3 Inversion de $A = \frac{\partial K}{\partial x}$

À l'heure actuelle, les techniques de calcul formel peuvent résoudre des systèmes linéaires dont la taille (nombre d'inconnues et d'équations) n'excède pas 25×25 [GOMEZ 95]. Cette limitation associée à la difficulté qui consisterait à générer automatiquement ces systèmes formellement, nous ont conduit à adopter une approche de type numérique.

Les méthodes de l'analyse numérique pour le traitement de systèmes linéaires se décomposent en deux principales catégories [CIARLET 90]. Les méthodes itératives sont utilisées lorsque les matrices des systèmes sont creuses (présence de beaucoup de zéros) et de forme particulière (matrices-bandes, diagonales par blocs, ...). Ces particularités sont exploitées pour définir des conditions assurant la convergence de ces algorithmes.
En revanche, lorsque les matrices sont quelconques, les hypothèses de convergence précédentes peuvent ne pas être vérifiées ce qui conduit à utiliser plutôt des méthodes de résolution dites directes. C'est dans ce cas de figure que nous nous trouvons pour traiter les systèmes d'équations linéaires générés par le module de spécification de tâches. La méthode directe la plus couramment employée pour résoudre des systèmes linéaires est la méthode de Gauss.

De nombreuses librairies de calcul numérique proposent diverses implantations de cet algorithme pour résoudre des systèmes linéaires [NUM 91]. Cependant, leur utilisation nécessite des efforts de mise en forme des systèmes à traiter que nous ne pouvons pas laisser à la charge de l'utilisateur. À l'inverse, nous souhaiterions disposer d'un système de résolution capable d'absorber et de résoudre le système linéaire (3.5) tel qu'il est proposé par le raisonnement cinématique. L'objectif principal recherché est une certaine robustesse de comportement vis à vis du conditionnement du problème à résoudre. Pour cela, la méthode proposée devra pratiquer des analyses de rang automatiques sur les systèmes linéaires à traiter.

Le principe que nous avons retenu pour effectuer cette inversion est inspirée de la méthode de Gauss. Cet algorithme constitue l'élément central de notre système de résolution. Son implantation informatique sous une forme générale a nécessité l'introduction de mécanismes complexes permettant la manipulation intensive et robuste de systèmes numériques.

3.2.4 Résolution de contraintes égalités

Soit à résoudre le problème de type (3.1-a). Le schéma numérique de Newton appliqué à la fonction g :

$$\begin{aligned} g: \quad & \mathbb{R}^p \to \mathbb{R}^m \\ & g(x) = 0 \end{aligned} \tag{3.6}$$

donne lieu au système linéaire $\underbrace{\left[\frac{\partial g}{\partial x}\right](x_n)}_{A} \Delta x_n = -\alpha g(x_n)$.

La méthode proposée pour inverser[3] la matrice A devra être susceptible de détecter automatiquement trois types de situations :

S-1 : $\boxed{rang\,(A) = p}$

Il existe une solution unique au problème. Nous avons nécessairement $m \geq p$ avec $(m-p)$ équations linéaires dépendantes. Sur les les p équations restantes, il est possible de pratiquer une inversion habituelle de matrice.

S-2 : $\boxed{rang\,(A) > p}$

Le système est sur-déterminé. Le nombre d'équations indépendantes excède p, le nombre d'inconnues; on calcule une solution minimisant un certain critère (par exemple, un moindre carré).

S-3 : $\boxed{rang\,(A) < p}$

Le système est sous-déterminé. Le nombre d'inconnues dépasse le nombre d'équations indépendantes; on choisit une solution particulière (par exemple, une solution de norme minimale).

Les cas S-2 et S-3 correspondent à des problèmes d'optimisation résolus par l'emploi de pseudo-inverses. Le calcul de l'incrément Δx_n se fera par des relations de la forme : $\Delta x_n = -\alpha A^+ g(x_n)$ où A^+ représente une pseudo-inverse de A. Selon le cas, A^+ a pour expression[4] : $\begin{cases} \text{S-2} \quad A^+ = (A^T A)^{-1} A^T & \text{pseudo-inverse à gauche} \\ \text{S-3} \quad A^+ = A^T (A A^T)^{-1} & \text{pseudo-inverse à droite} \end{cases}$.

3.2.5 Traitement de problèmes d'optimisation

Minimisation sans contraintes égalités

De nombreuses applications de robotique, nécessitant des minimisations de type $\min_x f^T(x) f(x)$, conduisent à des solutions x^* proches de zéro.

[3] Le terme *inverser* est utilisé ici par abus de langage. Une bonne stratégie ne sera évidemment pas de calculer explicitement la matrice inverse A^{-1}...

[4] En pratique, les calculs effectués sont menés selon la méthode plus efficace, décrite par [ANGELES 89].

Dans ce type de cas, on peut conjecturer que la résolution du problème $\min_x f^T(x)f(x)$ équivaut à résoudre le problème non-linéaire sur-contraint (plus de contraintes que d'inconnues) suivant :

$$\begin{aligned} f: \quad & \mathbb{R}^p \to \mathbb{R}^m \\ & f(x)=0 \end{aligned} \tag{3.7}$$

Dans notre approche, c'est sous cette forme que se présentent les problèmes de minimisation de fonctions. Nous allons développer dans ce qui suit la justification de l'équivalence des deux problèmes.

Le schéma de Newton-Raphson appliqué à (3.7) conduit au système linéaire $A(x_n)\Delta x_n = -\alpha f(x_n)$ dans lequel $A(x_n) = \dfrac{\partial f}{\partial x}(x_n)$. De plus, on suppose que ce système est sur-déterminé (cas S-2). L'emploi de l'inverse généralisée A^+ (3.8-a) permet d'évaluer l'incrément Δx_n conformément à (3.8-b).

$$\begin{cases} (a) \quad A^+ = \left[\left[\frac{\partial f}{\partial x} \right]^T \left[\frac{\partial f}{\partial x} \right] \right]^{-1} \left[\frac{\partial f}{\partial x} \right]^T \\ (b) \quad \Delta x_n = -\alpha \left[\left[\frac{\partial f}{\partial x} \right]^T \left[\frac{\partial f}{\partial x} \right] \right]^{-1} \left[\frac{\partial f}{\partial x} \right]^T f \end{cases} \tag{3.8}$$

Il reste à caractériser les propriétés de convergence locale de ce schéma itératif. Le résultat est que cette méthode converge quadratiquement seulement si f tend vers zéro. Dans le cas contraire, la convergence n'est plus assurée et sera au mieux asymptotique. Les applications robotiques de calibration nous situent à $f \simeq 0$. Des problèmes de calcul de minimum de distance plus généraux, avec par exemple $f \gg 0$, peuvent présenter des difficultés de convergence.

Traitons à présent le problème $\min_x f^T(x)f(x)$ par l'algorithme de Newton, appliqué à la condition nécessaire d'optimalité :

$$\begin{aligned} r: \quad & \mathbb{R}^p \to \mathbb{R}^p \\ & r(x) = \left[\frac{\partial f}{\partial x} \right]^T (x) \times f(x) = 0 \end{aligned} \tag{3.9}$$

La matrice A de la méthode itérative pour l'équation (3.9) est définie par la relation (3.10-a). Par abus de notation, nous utiliserons l'écriture plus condensée (3.10-b). L'incrément Δx_n se calcule selon la relation (3.10-c).

Nous pouvons observer que lorsque f est voisin de zéro[5], le schéma de Newton envisagé pour ce problème (3.9) bénéficie d'une convergence quadratique et devient identique à celui proposé en (3.8) pour résoudre (3.7), d'où le résultat précédent.

[5]Nous supposons également que les « dérivées » de f jusqu'à l'ordre 2 sont bornées.

$$\left\{\begin{array}{ll}
(a) & A=\begin{bmatrix} \sum_{i=1}^{m}\frac{\partial^2 f_i}{\partial x_1^2}f_i & \cdots & \sum_{i=1}^{m}\frac{\partial^2 f_i}{\partial x_1 \partial x_p}f_i \\ \vdots & & \vdots \\ \sum_{i=1}^{m}\frac{\partial^2 f_i}{\partial x_1 \partial x_p}f_i & \cdots & \sum_{i=1}^{m}\frac{\partial^2 f_i}{\partial x_p^2}f_i \end{bmatrix} + \\
 & \qquad \begin{bmatrix} \sum_{i=1}^{m}(\frac{\partial f_i}{\partial x_1})^2 & \cdots & \sum_{i=1}^{m}\frac{\partial f_i}{\partial x_1}\frac{\partial f_i}{\partial x_p} \\ \vdots & & \vdots \\ \sum_{i=1}^{m}\frac{\partial f_i}{\partial x_1}\frac{\partial f_i}{\partial x_p} & \cdots & \sum_{i=1}^{m}(\frac{\partial f_i}{\partial x_p})^2 \end{bmatrix} \\
(b) & A=\left[\frac{\partial^2 f}{\partial x^2}\right]^T f+\left[\frac{\partial f}{\partial x}\right]^T\left[\frac{\partial f}{\partial x}\right] \\
(c) & \Delta x_n=-\alpha A^{-1}\left[\frac{\partial f}{\partial x}\right]^T f
\end{array}\right. \tag{3.10}$$

Conjecture pour améliorer la convergence quand $f(x_{\text{solution}}) \gg 0$

Pour retrouver une convergence quadratique à partir du schéma de calcul de Δx_n (3.8), nous proposons d'introduire la dérivée $\frac{\partial \Delta x_n}{\partial x}$ qui peut être évaluée numériquement. Ceci conduit à la relation récurrente (3.11-a) où Θx_n représente le nouvel incrément de calcul. Son expression est donnée par l'équation (3.11-b).

$$\left\{\begin{array}{ll}
(a) & x_{n+1}=x_n+\Theta x_n \\
(b) & \Theta x_n=\left[\frac{\partial \Delta x_n}{\partial x}\right]^{-1}\Delta x_n \\
(c) & \Theta x_n=\left[\frac{2}{\alpha}\left[\frac{\partial^2 f}{\partial x^2}\right]^T\left[\frac{\partial f}{\partial x}\right]\Delta x_n+\right. \\
 & \qquad \left.\left[\frac{\partial^2 f}{\partial x^2}\right]^T f+\left[\frac{\partial f}{\partial x}\right]^T\left[\frac{\partial f}{\partial x}\right]\right]^{-1}\left[\frac{\partial f}{\partial x}\right]^T f
\end{array}\right. \tag{3.11}$$

Nous faisons la conjecture suivante : cette méthode converge quadratiquement.

La justification de ce résultat peut être perçue dans l'expression (3.11-c), développement de (3.11-b). Cette écriture condensée donne, avec les abus de notation déjà mentionnés en (3.10-b), un aperçu sur la forme des calculs au second ordre. On peut observer que lorsque l'on est proche de la solution ($\Delta x_n \simeq 0$), (3.11-c) devient équivalente à (3.10-c).
Cette conjecture a été vérifiée formellement et numériquement sur un exemple particulier de faible dimension. La généralisation de l'approche à des cas plus complexes peut être envisagée de façon naturelle mais se traduit par une augmentation singulière du coût calcul (dérivation numérique).
Actuellement, l'implantation du module de résolution n'applique pas cette stratégie de calcul (3.11).

Minimisation avec contraintes égalités

Soit à résoudre un problème du type (3.1-c). Par commodité, nous noterons J le critère $f^T f$. Nous donnons ci-après les espaces de définition des fonctions J, f et g :

$$\begin{array}{lll} J: & \mathbb{R}^p & \rightarrow \mathbb{R} \\ g: & \mathbb{R}^p & \rightarrow \mathbb{R}^q \\ f: & \mathbb{R}^p & \rightarrow \mathbb{R}^m \end{array}$$

avec les inconnues $x \in \mathbb{R}^p$, $q<p$ (moins de contraintes g que d'inconnues) et $p<q+m$ (plus de contraintes f et g que d'inconnues).

- a - Conditions théoriques d'optimalité :

Au voisinage d'une solution x^*, les développements de Taylor de J et des fonctions scalaires $g_{i;i\in[1,q]}$ sont de la forme :

$$\begin{aligned} J(x^*+\delta x) &\underset{1^{\text{er}}\text{ordre}}{=} J(x^*) + \left[\ \frac{\partial J}{\partial x}\ \right](x^*)\,\delta x \\ g_i(x^*+\delta x) &\underset{1^{\text{er}}\text{ordre}}{=} \underbrace{g_i(x^*)}_{=0} + \left[\ \frac{\partial g_i}{\partial x}\ \right](x^*)\,\delta x \end{aligned}$$

Lorsque les contraintes g sont satisfaites, l'espace tangent à ces contraintes est engendré par des vecteurs δx orthogonaux aux gradients $\left[\ \frac{\partial g_i}{\partial x}\ \right]^T$, soit :

$$\delta x \in \mathbb{R}^p \quad \text{tel que} \quad \left[\ \frac{\partial g}{\partial x}\ \right]\delta x = 0 \tag{3.12}$$

La condition nécessaire d'optimalité sur la minimisation du critère J sans contraintes est la nullité du gradient de J. En présence de contraintes g (problème (3.1-c)), la condition nécessaire d'optimalité s'écrit, pour tout δx de l'espace tangent de g :

$$\left[\ \frac{\partial J}{\partial x}\ \right](x^*)\,\delta x = 0 \tag{3.13}$$

Le gradient de J, $\left[\ \frac{\partial J}{\partial x}\ \right]^T$ doit être orthogonal à l'espace tangent de $g(x^*)=0$.

$\boxed{\text{Construction de } \delta x \text{ vérifiant (3.12) et (3.13)}}$

Opérons une partition de $\mathbb{R}^p$ en deux sous-espaces permettant d'écrire δx sous la forme : $\delta x = \begin{bmatrix} \delta x_1 \\ \delta x_2 \end{bmatrix}$ avec $(x_1, x_2) \in \mathbb{R}^t \times \mathbb{R}^{p-t}$. Nous supposerons que cette décomposition transforme (3.12) en une relation de la forme :

$$\underbrace{\left[\ \frac{\partial g}{\partial x_1}\ \right]}_{inversible}\delta x_1 + \left[\ \frac{\partial g}{\partial x_2}\ \right]\delta x_2 = 0$$

dans laquelle, par construction, la matrice $\left[\ \frac{\partial g}{\partial x_1}\ \right]$ est inversible et de rang plein ($t=q$). De cette manière, le vecteur δx_1 peut être évalué en fonction de δx_2 par l'équation :

$$\delta x_1 = -\left[\ \frac{\partial g}{\partial x_1}\ \right]^{-1}\left[\ \frac{\partial g}{\partial x_2}\ \right]\delta x_2 \tag{3.14}$$

D'autre part, la condition nécessaire d'optimalité (3.13) peut s'écrire, en tenant compte de la partition précédente de $\mathbb{R}^p$:

$${\delta x_1}^T\left[\ \frac{\partial f}{\partial x_1}\ \right]^T f + {\delta x_2}^T\left[\ \frac{\partial f}{\partial x_2}\ \right]^T f = 0 \tag{3.15}$$

Tous calculs faits, la substitution de (3.14) dans (3.15) conduit à la relation suivante :

$${\delta x_2}^T\underbrace{\left[\ \left[\ \frac{\partial f}{\partial x_2}\ \right]^T - \left[\ \left[\ \frac{\partial f}{\partial x_1}\ \right]\left[\ \frac{\partial g}{\partial x_1}\ \right]^{-1}\left[\ \frac{\partial g}{\partial x_2}\ \right]\ \right]^T\ \right]}_{B} f = 0$$

La condition nécessaire d'optimalité exige que l'expression précédente soit vérifiée pour tout $\delta x_2 \in \mathbb{R}^{p-t}$. La condition nécessaire d'optimalité devient alors :

$$Bf = 0 \tag{3.16}$$

- b - Application de l'algorithme itératif de Newton :

Résolution des contraintes g

L'incrément Δx_n doit être évalué par une relation du type :

$$\left[\ \frac{\partial g}{\partial x}\ \right] \Delta x_n = -g \tag{3.17}$$

On suppose qu'il existe une sous-matrice $\left[\ \frac{\partial g}{\partial x_1}\ \right]$ inversible et de rang plein permettant d'écrire (3.17) sous la forme :

$$\underbrace{\left[\ \frac{\partial g}{\partial x_1}\ \right]}_{inversible} \Delta x_1^n + \left[\ \frac{\partial g}{\partial x_2}\ \right] \Delta x_2^n = 0$$

où $\begin{bmatrix} \Delta x_1^n \\ \Delta x_2^n \end{bmatrix} = \Delta x_n$. Dans ces conditions, l'incrément Δx_n a pour expression :

$$\Delta x_1^n = -\left[\ \frac{\partial g}{\partial x_1}\ \right]^{-1} \left[\ \left[\ \frac{\partial g}{\partial x_2}\ \right] \Delta x_2^n + g\ \right] \tag{3.18}$$

Résolution des contraintes f

En substituant (3.18) dans l'équation $\left[\ \frac{\partial f}{\partial x}\ \right] \Delta x_n = -f$, on obtient la relation suivante qui permet de calculer l'incrément Δx_2^n :

$$\underbrace{\left[\ \left[\ \frac{\partial f}{\partial x_2}\ \right] - \left[\ \frac{\partial f}{\partial x_1}\ \right] \left[\ \frac{\partial g}{\partial x_1}\ \right]^{-1} \left[\ \frac{\partial g}{\partial x_2}\ \right]\ \right]}_{C} \Delta x_2^n = -f + \underbrace{\left[\ \frac{\partial f}{\partial x_1}\ \right] \left[\ \frac{\partial g}{\partial x_1}\ \right]^{-1} g}_{D} \tag{3.19}$$

Il faut noter que le terme D est négligeable au premier ordre, à l'issue de la résolution de (3.17), car g est voisin de 0. D'autre part, en supposant que le rang de C est supérieur à $p-q$ (i.e le système (3.19) est sur-déterminé), l'incrément Δx_2^n peut être calculé au moyen d'une pseudo-inverse. Sous réserve que $C^T C$ soit inversible :

$$\Delta x_2^n = -(C^T C)^{-1} C^T f \tag{3.20}$$

Cette résolution correspond à la situation S-3 (solution de type moindres carrés). Cette valeur de Δx_2^n est alors réinjectée dans (3.18) pour obtenir Δx_1^n et trouver finalement une solution Δx_n de $\mathbb{R}^p$.

Si la méthode itérative converge vers x^* alors $\Delta x_2^n = 0$ ce qui implique que $C^T f = 0$. La solution trouvée vérifie donc la condition nécessaire d'optimalité exprimée en (3.16) du problème (3.1-c) , car $C^T = B$.

Expérimentalement, nous avons observé que les conditions de convergence étaient analogues à celles des problèmes de minimisation sans contraintes décrites plus haut, avec en particulier, une convergence quadratique lorsque $f \simeq 0$ à la solution.

Résolution simultanée de contraintes égalités et inégalités

Nous avons vu plus haut comment l'emploi du cœfficient de pondération α pouvait servir à maîtriser les propriétés locales de convergence de l'algorithme de Newton-Raphson.

Soit à résoudre un problème de type (3.1-b). La généralisation du processus de pondération autorise un calcul de l'incrément x_n selon le procédé suivant :

$$\Delta x_n = -\alpha_n A^{-1} \Lambda_n g(x_n) \tag{3.21}$$

où α_n est un cœfficient positif inférieur ou égal à un et Λ_n une matrice diagonale telle que $0 < \lambda_{i,i} \leq 1$.

Cette modification du schéma de Newton habituel permet d'exploiter le contrôle des conditions de convergence pour satisfaire d'une façon locale les contraintes de type inégalité.

Des valeurs non unitaires pour les cœfficients de la matrice Λ_n entraînent une déformation locale de la trajectoire initiale calculée par l'algorithme de Newton-Raphson (les trajectoires obtenues deviennent « asymptotiques » dans l'espace des contraintes). Leur calcul est réalisé par la résolution d'un simplexe incluant les contraintes inégalités h (cf § 3.7.3).

3.3 Formulation initiale des problèmes

Un problème à résoudre, formalisé selon les relations (3.1), devra être décrit par un modèle d'état et son modèle variationnel associé. Contrairement à une approche formelle qui manipulerait explicitement les fonctions f, g, h et leurs dérivées, notre description des problèmes introduit un grand nombre de variables.

Les valeurs de ces dernières peuvent s'identifier à l'état x du système, aux composantes des fonctions f, g, h, à des termes paramétriques ou encore aux composantes de fonctions intermédiaires facilitant la modélisation du problème à résoudre. Pour exprimer ce lien entre variables et fonctions, un processus de marquage des variables est inclus dans le modèle. Cet ensemble de variables noté v constitue notre support de représentation du problème.

Étant donné un état x du système modélisé, le modèle d'état permet d'évaluer toutes les autres variables de v. Le modèle variationnel au premier ordre est de la forme $M \times \Delta v = 0$ et sert à la résolution des équations de contraintes.

Les principaux avantages de cette approche sont :

- la simplicité et souplesse de modélisation,
- l'adaptation automatique des procédés de résolution permettant de traiter, de manière robuste, une grande variété de problèmes sans effort de mise en forme,
- la représentation éclatée des problèmes (par l'introduction de nombreuses variables) qui permet, en particulier :
  - d'accéder aux évolutions des variables internes du système étudié,
  - de pratiquer des études de sensibilité (par rapport aux paramètres du modèle, aux erreurs sur les contraintes, etc).

Le paragraphe suivant expose, en guise de préalable à la description de l'algorithme de résolution, les conditions d'exploitation des relations de contraintes générées par le système de spécification de tâches.

3.4 Application à la résolution itérative d'un problème de mouvement

Dans le précédent chapitre, nous avons exposé notre technique de spécification de tâches basée sur l'analyse géométrique et cinématique des systèmes étudiés.

Considérons l'exemple simple d'une tâche de mouvement, correspondante à un problème de type (3.1-a), modélisée par une seule boucle cinématique. Chaque lien de cette boucle fait intervenir un vecteur de configuration K^i dont les six composantes sont des variables traitées par la méthode de résolution.

A l'étape n du mouvement vers l'objectif, l'état du système est évalué en explorant l'arborescence des repères. Le raisonnement cinématique permet d'engendrer l'équation de contrainte provenant du cycle qui exprime la dépendance linéaire entre les incréments de toutes les composantes des vecteurs K_i. Cette relation s'écrit :

$$\sum_i M_n^i \Delta K_n^i = 0_6 \tag{3.22}$$

Supposons, dans un premier temps, que ce système fasse intervenir essentiellement deux types de variables, à savoir, des variables inconnues étiquetées Q (composantes du vecteur d'état x) et des variables de contraintes étiquetées C (composantes de la fonction g). (3.22) est donc de la forme :

$$M_Q \Delta Q_n = -M_C \Delta C_n \tag{3.23}$$

Le calcul de la position suivante Q_{n+1} sur la trajectoire se fait en évaluant l'incrément ΔQ_n à partir de (3.23), par l'emploi d'un mécanisme d'élimination de variables aboutissant à l'équation $\Delta Q_n = -M \Delta C_n$. Numériquement, l'incrément ΔQ_n sera évalué par une relation similaire à (3.21) :

$$\Delta Q_n = -\alpha_n \, M \, \Lambda_n \, \Delta C_n \tag{3.24}$$

Le contrôle de la vitesse de convergence par les valeurs du gain α_n se traduit en pratique par la possibilité de maîtriser l'amplitude de variation des incréments de déplacements. Cette caractéristique est importante pour la génération de mouvements continus pour un manipulateur, mais constitue aussi un des moyens permettant de maîtriser certains problèmes numériques lorsque les mécanismes étudiés atteignent des configurations singulières.

Enfin, la prise en compte de contraintes inégalités par l'intermédiaire des cœfficients de Λ_n autorise des capacités d'évitement local d'obstacles.

3.5 Manipulation hybride numérique/formelle de systèmes d'équations linéaires

Le paragraphe suivant expose les principales notations qui seront employées dans la suite de l'exposé.

3.5.1 Conventions d'écriture

Étiquettes de variables

Les étiquettes employées dans la méthode de spécification doivent être interprétées en termes se rapportant aux formulations des problèmes (3.1). Afin de ne pas introduire de confusion avec les étiquettes définies précédemment, les qualificatifs de variables dans la méthode de résolution seront désignées par des lettres majuscules calligraphiées. Le tableau (3.1) présente les étiquettes de variables utilisées pour la description de la méthode de résolution.

Tout en conservant la répartition en quatre variétés (paramètres, variables intermédiaires, inconnues, contraintes), la famille des variables de type contrainte est ventilée en trois sous-classes, ce qui porte finalement à six le nombre de catégories.

Étiquette	Caractéristique
$\mathcal{P}$	Paramètre
$\mathcal{R}$	Variable intermédiaire
$\mathcal{X}$	Composante du vecteur d'état (Inconnue)
$\mathcal{G}$	Composante de la fonction g (Contrainte égalité)
$\mathcal{F}$	Composante de la fonction f (Contrainte moindres carrés)
$\mathcal{H}$	Composante de la fonction h (Contrainte inégalité)

Tableau 3.1: Étiquettes de variables.

Les ensembles de variables seront des lettres majuscules. L'ensemble P contient toutes les variables étiquetées $\mathcal{P}$. Les ensembles R, X, G, F et H sont définis avec des règles analogues.

Enfin, l'ensemble $V^* = P \cup R \cup X \cup G \cup F \cup H$ regroupe la totalité des variables. $V = V^* - P$ définit l'ensemble des variables qui n'ont pas le statut de paramètre.

Écritures matricielles

Matrices et vecteurs constituent les éléments de base manipulés dans le système de résolution.

Les équations de contraintes à résoudre fournies par la spécification de tâches se présentent sous la forme matricielle générale suivante :

$$M \times \Delta V = 0 \tag{3.25}$$

Les systèmes linéaires variationnels impliqués dans la résolution font intervenir des incréments de variables. C'est pourquoi, nous omettrons la mention explicite Δ chaque fois que c'est possible.

Plus généralement, un symbole K pourra désigner, en fonction du contexte, soit un vecteur d'incréments de variables, soit l'ensemble auquel appartiennent ces variables. Cette convention permet d'alléger les notations sans générer d'ambiguïtés d'interprétation.
La relation (3.25) peut être séparée en deux parties :

$$MG \times V_G = MD \times V_D$$
$$\text{où} \quad V_G \subseteq V^*, \quad V_D \subseteq V^* \text{ et } V_G \cap V_D = \emptyset$$

les matrices MG et MD désignant respectivement les membres gauche et droit de l'équation.

Afin de décrire la méthode de résolution, nous décomposerons le système précédent en nl systèmes formés de matrices à une ligne :

$$LG_l \times V_G = LD_l \times V_D \quad \forall l \in [1, nl] \tag{3.26}$$

Comme nous le verrons plus loin, la gestion des zéros présents dans les matrices lignes LG_l est un élément essentiel pour le traitement robuste des systèmes linéaires. Une analyse des zéros appliquée à des systèmes-lignes sera signalée par une notation du type :

$$ZG_l \times V_G = ZD_l \times V_D \quad \forall l \in [1, nl], \tag{3.27}$$

dans laquelle tous les cœfficients des matrices lignes ZG_l et ZD_l sont non nuls.

3.5.2 Origine du besoin de manipulation formelle des systèmes d'équations

Un problème qui se pose immédiatement est la maîtrise du volume des systèmes fournis par l'analyse cinématique. Pour l'exemple du manipulateur plan à deux degrés de liberté présenté au chapitre précédent, le graphe cinématique comporte 22 liens pour trois cycles (voir figure 2.3). Par conséquent, le nombre total théorique de variables introduites s'élève à $6 \times 22 = 132$. Fort heureusement, un grand nombre de ces variables sont de type paramétriques et n'interviennent donc pas dans les relations variationnelles fournies par la spécification de tâches. Cette remarque ramène à 24 le nombre de variables pertinentes du problème[5], chiffre qui donne une idée du nombre de variables qui pourront intervenir dans des exemples spatiaux.

Nous venons d'illustrer un premier niveau de simplification formelle des équations variationnelles au moyen duquel les variables de type paramétrique sont écartées de la résolution. Une autre source de simplification se situe, cette fois sur le plan numérique, au niveau de l'analyse des zéros présents dans les matrices. Le bénéfice de cette mesure est multiple. Elle permet de :

- de réduire le vecteur initial du système linéaire aux seules variables pertinentes,
- d'accroître la robustesse numérique de la résolution en évitant les erreurs d'arrondis dus aux termes très petits,
- de décomposer, le cas échéant, le jeu initial d'équations en systèmes linéaires que l'on peut résoudre indépendamment.

Cette analyse des « zéros » constitue un des éléments clés du système de résolution. La robustesse numérique de la méthode proposée et la possibilité de traiter des systèmes complexes résultent, pour une part importante, de la prise en compte systématique de ces zéros numériques. A la lumière de cette remarque, la manipulation des systèmes linéaires est bien réalisée à l'aide d'un mécanisme hybride numérique-formel. Décrivons ci-après le principe de cette analyse.

[5] six inconnues, six contraintes et 12 variables intermédiaires (voir la spécification de la figure 4.9).

3.5.3 Zéros structuraux et conjoncturels

Les problèmes (3.1) que nous souhaitons résoudre sont traduits par la spécification de tâches sous la forme de systèmes linéaires du type $MG \times V = 0$.

Cette forme brute ne se prête pas à une résolution immédiate. Le traitement numérique de systèmes dans lesquels cohabitent des termes dont l'ordre de grandeur est très fortement différent pose de nombreuses difficultés et engendre en particulier des défauts de robustesse.

Les étapes de la détection de zéros sont les suivantes. Tout d'abord, il convient de normaliser les cœfficients de la matrice MG. Après s'être fixé une valeur seuil ϵ proche de zéro, un cœfficient sera déclaré nul s'il est inférieur à ce seuil. La contribution relative de la variable correspondante à ce terme quasi-nul étant très faible (on suppose que les incréments dans le vecteur V sont bornés), cette dernière peut être éliminée de l'équation initiale.

La nullité de ce cœfficient de la matrice MG découle, soit de l'absence d'influence de la variable concernée sur le problème quel que soit l'état x : nous parlerons alors de zéro structural; soit d'une configuration particulière de l'état du système : nous qualifierons alors ce zéro de conjoncturel.

Par ce moyen, le système est capable de changer *dynamiquement* le modèle du problème à traiter en une forme que l'on peut résoudre.

L'application de ce procédé sur chaque cœfficient fait apparaître des zéros dans la matrice MG. Après une réorganisation des lignes et des colonnes, une première situation pourra conduire à l'élimination d'un certain nombre de variables de V (équation (3.28)). Une autre éventualité sera la décomposition du problème initial en plusieurs problèmes indépendants (équation (3.29)).

$$\begin{bmatrix} M_1 & 0 \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \end{bmatrix} = 0 \iff M_1 V_1 = 0 \tag{3.28}$$

$$\begin{bmatrix} M_1 & 0 \\ 0 & M_2 \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \end{bmatrix} = 0 \iff \begin{cases} M_1 V_1 = 0 \\ M_2 V_2 = 0 \end{cases} \tag{3.29}$$

La validité de la condensation du système initial réalisée à l'étape n doit être révisée à l'itération suivante pour le nouveau système linéaire fourni par le raisonnement cinématique.

Si la condensation ou décomposition précédentes sont toujours valides à l'itération $n+1$ alors les zéros détectés étaient effectivement des zéros structuraux. Une telle situation apparaît par exemple lorsque le système étudié évolue dans un plan. Dans ce cas, seulement trois variables, sur les six que comporte chaque vecteur de configuration, auront un rôle dans les relations de contraintes (dans l'exemple du manipulateur à deux ddl : les composantes en translation selon $\vec{z}$ et en rotation autour de $\vec{x}$ et $\vec{y}$ sont nulles).

Mais il peut aussi arriver que la condition de nullité de certains termes des matrices, établie à l'étape n, ne soit plus vérifiée à l'étape suivante. Dans ce cas, les zéros supposés structuraux à l'instant n étaient simplement conjoncturels. La condensation du système doit alors être remise en cause et les variables écartées précédemment doivent reprendre leur place dans les relations. De cette façon, la forme du système à résoudre sera différente à l'étape $n+1$.

Ce cas de figure intervient par exemple lorsqu'un robot, au cours de son mouvement, passe transitoirement par un état géométrique particulier, entraînant une réduction de mo-

bilité. Qualitativement, cette situation correspond à un état de configuration singulière. A titre d'illustration, si les axes de certaines des liaisons pivots d'un manipulateur série deviennent parallèles, la mobilité assurée par ces liaisons ne peut se faire que dans un plan.

Dans de nombreuses situations, nous pouvons remarquer que des états réputés singuliers pour les calculs correspondent plus à des singularités inhérentes à la modélisation adoptée qu'à la présence de singularités physiques vraies dans les mécanismes. D'une manière similaire, une situation dite singulière pourra parfois trouver son origine dans l'obstination à vouloir atteindre des objectifs inatteignables. En robotique de manipulation, un exemple immédiat de cette situation consiste à spécifier au robot la prise d'un objet situé en dehors de son domaine de travail. Une réponse cohérente du système sera, par exemple, de positionner le robot à la distance minimale de l'objet.

En d'autres termes, nous souhaiterions que de telles « singularités » puissent être absorbées par notre méthode de résolution libérant ainsi l'utilisateur, au moins partiellement, des contraintes de reformulation et de mise en forme des problèmes qu'il souhaite résoudre.

3.6 Principe fondamental de résolution: élimination de Gauss

La méthode de résolution proposée est construite autour d'un opérateur fondamental de triangulation de matrices. Cet opérateur, que nous appellerons aussi principe fondamental, réalise des manipulations de matrices-lignes selon un schéma d'élimination de Gauss. Sa fonction principale a pour objet l'analyse de rang de matrices. Dans ce paragraphe nous décrivons le fonctionnement de cet opérateur.

Pourquoi analyser le rang des matrices ?

Ce point est fondamental pour la résolution d'un système linéaire

$$Ax = b$$

Dans notre contexte, nous souhaitons résoudre, c'est à dire calculer le vecteur inconnu x, quelles que soient les propriétés de la matrice A et pas seulement dans le cas particulier où A est inversible. La connaissance du rang de A renseigne sur le type de résolution à appliquer au système (solution exacte, aux moindres carrés ou de norme minimale).

Par quel moyen y parvenir ?

De façon intrinsèque, la méthode de Gauss permet de déterminer cette information de rang. Dans le cas où la matrice est inversible, elle présente l'intérêt de fournir la solution directement (à l'étape de remontée près). Dans les autres cas, le déroulement de cet algorithme assure la mise en forme des systèmes, à résoudre par une procédure adaptée (moindres carrés ou norme minimale).

Ces caractéristiques correspondent parfaitement aux objectifs que nous poursuivons pour la résolution automatique de systèmes linéaires. La principale difficulté consiste à implanter cet algorithme sous une forme générale.

Supposons que le système linéaire à traiter (par exemple fourni par la spécification de tâche) soit de la forme (3.25). Pour des raisons de simplicité, nous considérerons que ce

système ne comporte que deux sortes de variables : des variables *inconnues* et des variables de type *contrainte* que nous désignerons respectivement par les symboles $\mathcal{I}$ et $\mathcal{K}$. Le principe fondamental de résolution est un opérateur monoadique qui va recevoir en entrée le système brut et fournir en sortie l'expression des variables $\mathcal{I}$ en fonction des variables $\mathcal{K}$. Les ensembles formés de ces types de variables seront notés I et K.

Le système à résoudre se décompose en un nouvel ensemble de nl équations linéaires à une ligne :

$$LG_l \times W_l = 0 \quad \forall l \in [1, nl] \tag{3.30}$$

3.6.1 Élimination des zéros structuraux

Pour nous prémunir des problèmes liés à la présence de termes voisins de zéros dans les matrices-lignes LG_l, nous leur appliquons la procédure d'élimination des zéros structuraux[6] décrite en 3.5.3. Le système (3.30) devient :

$$ZG_l \times V_l = 0 \quad \forall l \in L \quad \text{tel que} \quad V_l \subseteq W_l \quad \text{et} \quad card(L) \leq nl \tag{3.31}$$

dans lequel les matrices lignes ZG_l sont normalisées et dépourvues de zéros structuraux. Le système (3.31) constitue l'ensemble des relations linéaires de travail.

Deux systèmes lignes l_1 et l_2 de (3.31) sont déclarés indépendants si et seulement si $V_{l_1} \cap V_{l_2} = \emptyset$. En conséquence, ils pourront être résolus séparément. En étendant cette notion, il est possible de partitionner un ensemble d'équations en sous-ensembles minimaux d'équations indépendantes que l'on peut traiter séparément.

3.6.2 Choix des variables et recherche du meilleur pivot

Le vocable de *pivot relatif à une inconnue* constitue le terme consacré pour désigner un des cœfficients de l'inconnue considérée. L'influence désastreuse de l'utilisation de pivots « trop petits » est bien connue dans l'application pratique de l'algorithme de Gauss. Si l'étape précédente a contribué à l'élimination des termes quasi-nuls, nous pouvons encore améliorer le comportement numérique de la méthode en adoptant une stratégie de choix du « meilleur » pivot.

Désignons par $\{p^t\}$, l'ensemble des cœfficients issus des matrices lignes ZG_l relatifs à une variable inconnue à traiter i^t, étiquetée $\mathcal{I}$. Le meilleur pivot p^t_* pour le calcul de la variable i^t dans (3.31) est le terme de $\{p^t\}$ de norme maximale.

La meilleure inconnue i^1 à traiter correspond, quant à elle, à la variable dont le pivot est maximum dans l'ensemble $\{p^t_*\}$ de tous les meilleurs pivots.

En supposant que la meilleure inconnue se trouve dans la matrice ZG_j (figure 3.1), cette $j^{\text{ième}}$ équation permet de calculer la première variable à traiter :

$$i^1 = \frac{ZD_j}{p^1_*} \times \{K_j \cup I_j - \{i^1\}\} \tag{3.32}$$

[6]Dans ce qui suit, nous emploierons exclusivement l'adjectif structural, délaissant de ce fait la nuance introduite plus haut entre entre zéros structuraux et conjoncturels.

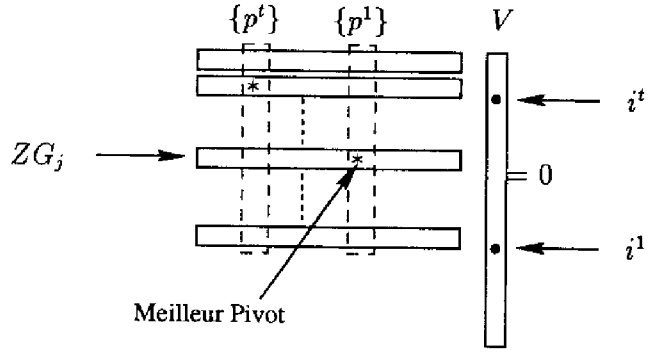

Figure 3.1: Recherche de la meilleure inconnue et du meilleur pivot.

3.6.3 Substitution directe

La substitution de l'expression (3.32) de i^1 dans (3.31) conduit à des équations de la forme :

$$LG_l \times \{K_l \cup I_l - \{i^1\}\} = 0 \quad \forall l \in (L_2 = L_1 - \{j\}) \tag{3.33}$$

ne contenant plus la variable i^1.

En appliquant la procédure d'élimination des zéros structuraux, un nouveau jeu d'équations analogues à celles présentées en (3.31) peut être obtenu.

$$ZG_l \times \{K_l^r \cup I_l^r\} = 0 \quad \forall l \in (L_3 \subseteq L_2) \tag{3.34}$$

Les ensembles K_l^r and I_l^r constituent des sous-ensembles réduits de K_l et I_l. Les étapes de sélection des variables de meilleur pivot, d'élimination de zéros structuraux et de substitution directe sont alors répétées récursivement sur le système (3.34) jusqu'à ce que l'ensemble des variables $\mathcal{I}$ de I_l^r ou l'ensemble d'indices L_3 s'épuisent.

A l'issue de ce traitement, nous disposons d'un ensemble de variables résolues I_S définies par des relations de la forme :

$$\begin{cases} i^1 = \dfrac{ZD_j}{p_*^1} \times \{K_j \cup I_j - \{i^1\}\} \\ \vdots \\ i^s = \dfrac{ZD_k}{p_*^s} \times \{K_k \cup I_k - \{i^1, \ldots, i^s\}\} \end{cases} \tag{3.35}$$

Ce système d'équations est triangulaire supérieur par rapport aux variables résolues I_S. Dans le cas général, il peut rester un ensemble de variables non résolues I_U égal à $I - \{i^1, \ldots, i^s\}$ et le cas échéant, des équations relatives à un ensemble de variables résiduelles $\mathcal{K}$:

$$ZG_j \times K_j = 0 \quad \forall j \in J$$

Il faut noter que s peut être inférieur ou égal à nl de (3.30) dans le cas de dépendance linéaire de certaines équations de (3.30).

3.6.4 Propagation inverse

L'évaluation des inconnues I_S est réalisée par substitution inverse des variables dans le système triangulaire (3.35). Après élimination des zéros structuraux et réorganisation des équa-

tions, apparaît un système linéaire transformé, équivalent au jeu d'équations initial (3.30) :

$$\begin{array}{lll} (a) & I_S = MD \times \{I_U \cup K_1\} \quad \text{avec} & I_S \cap I_U = \emptyset \\ (b) & MG \times K_2 = 0 & \end{array} \tag{3.36}$$

3.6.5 Analyse de la situation avant résolution

Dans le système d'équations résultant (3.36), les ensembles de variables inconnues non résolues I_U et des contraintes résiduelles K_2 vont déterminer la nature du traitement à appliquer pour obtenir la solution du problème. Quatre principaux cas sont à considérer en fonction de l'état des ensembles I_U et K_2.

C-1 : $\boxed{I_U = \emptyset \text{ et } K_2 = \emptyset}$

Cette situation apparaît lorsque le système initial peut s'écrire de façon globale $MG \times I = MD \times K$, la matrice MG étant à la fois carrée et bien conditionnée. La méthode d'élimination de Gauss que nous venons de décrire fournit directement la solution avec le système (3.36-a).

C-2 : $\boxed{I_U \neq \emptyset \text{ et } K_2 = \emptyset}$

Dans ce cas, le nombre de composantes du vecteur d'état dépasse le nombre de contraintes à satisfaire (le système est sous-défini). Une solution de norme minimale peut donc être calculée. Des cœfficients de pondération seront introduits pour permettre éventuellement à l'utilisateur de favoriser l'influence de certaines variables par rapport aux autres dans le choix de la solution.

C-3 : $\boxed{I_U = \emptyset \text{ et } K_2 \neq \emptyset}$

À présent, c'est le nombre des contraintes à vérifier qui est plus important que le nombre d'inconnues (système sur-défini). On élabore, à partir de (3.36), un nouveau système prenant en compte l'ensemble des équations de contrainte excédentaires correspondantes aux variables K_2. Une solution de type moindres carrés qui minimise les erreurs sur les variables de contrainte K_2 est alors proposée.

Comme dans le cas précédent, il peut être intéressant d'inclure des termes de pondération grâce auxquels l'utilisateur hiérarchise les contraintes à vérifier. Avec l'exemple d'un problème de calibration (*ie* identification des paramètres de position d'objets), les contraintes K_2 à satisfaire correspondent à des mesures. Les cœfficients de pondération caractérisent donc le degré de confiance vis à vis des mesures.

C-4 : $\boxed{I_U \neq \emptyset \text{ et } K_2 \neq \emptyset}$

Lorsque l'on cherche une solution terminale de la forme (3.37), celle-ci ne peut pas être construite. Ce cas de figure survient probablement lorsque le problème initial n'a pas été bien spécifié. Cette situation peut également apparaître, cette fois de façon normale, dans les étapes intermédiaires du conditionnement des équations exposé ci-après.

3.6.6 Résolution finale

En résolvant si nécessaire (3.36) par le procédé adapté (norme minimale ou moindres carrés), la solution globale de (3.30) est de la forme :

$$I = MD \times K \tag{3.37}$$

Nous venons de présenter le principe fondamental de résolution appliqué à un système linéaire ne comportant que deux catégories de variables (des inconnues et des contraintes).

La démarche proposée pour résoudre le problème général est d'utiliser l'opérateur que nous venons de décrire comme outil d'analyse permettant l'étude de rang des matrices et la mise en forme des systèmes linéaires en vue de leur résolution. La diversité des types de variables mises en jeu et la combinatoire des situations possibles entraînent un accroissement de la complexité de résolution du problème général. Le paragraphe suivant aborde le traitement d'un problème dans le cas général.

3.7 Traitement du problème général

Nous revenons maintenant au problème posé initialement sous la forme (3.25). Le principe fondamental décrit dans le paragraphe précédent va servir à sa résolution.

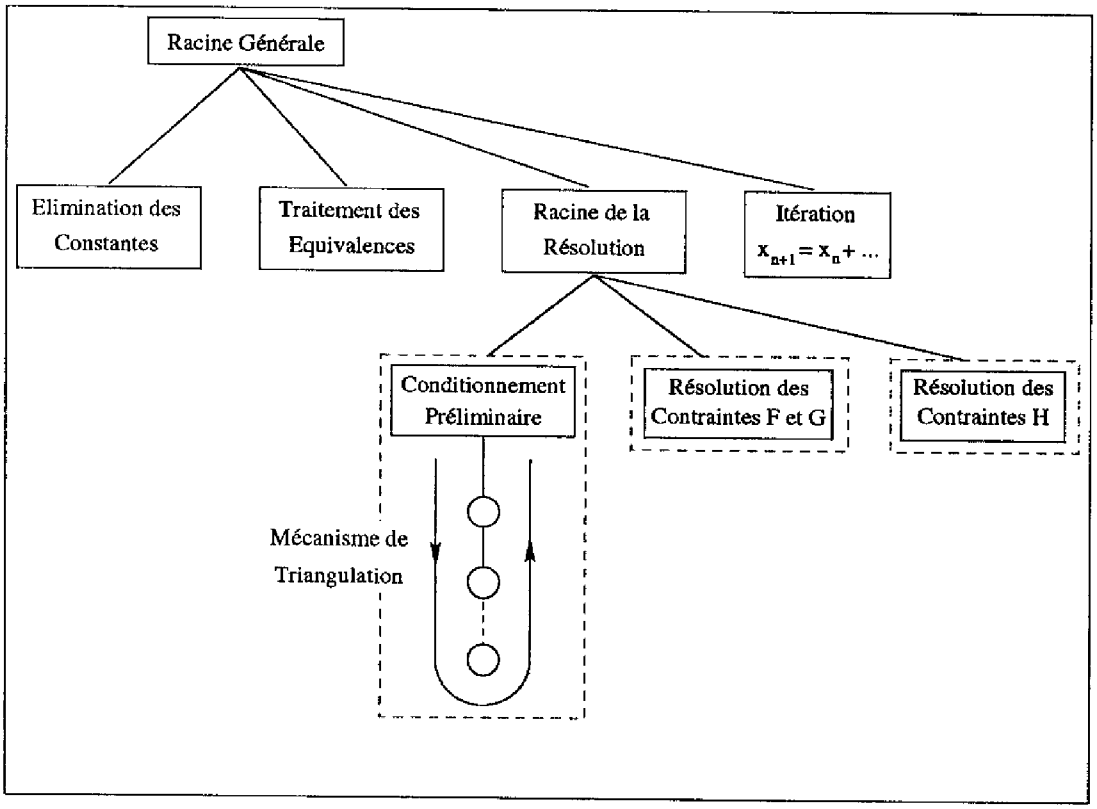

Figure 3.2: Arbre de résolution

À ce stade de l'exposé, il est important de donner un aperçu sur l'implantation de la méthode de résolution. En effet, dans la présentation de cette technique, on ne peut pas

totalement dissocier le fond de la forme. La figure 3.2 décrit la structuration du système de résolution. Elle s'appuie sur un schéma arborescent dont la ramification évolue dynamiquement à chaque étape du traitement.

Deux opérations précèdent la résolution du problème proprement dit. Il s'agit tout d'abord de l'élimination des constantes présentes dans les équations variationnelles provenant du système de spécification. Cette phase a déjà été évoquée plus haut dans le paragraphe 3.5. Ensuite, les variables déclarées équivalentes au moment de la génération des équations de contraintes sont condensées dans les relations matricielles. Ce cas se produit en particulier lors du traitement de problèmes de calibration.

Les paragraphes qui vont suivre détaillent les principales sous-phases de la résolution.

3.7.1 Conditionnement préliminaire des équations

L'objectif de cette étape consiste à transformer le problème initial (3.25) en un système d'équations équivalent dans lequel les variables inconnues X apparaissent de préférence dans les membres droits des équations.

Pour cela, la technique de triangulation présentée plus haut doit être appliquée aux différentes catégories de variables. Nous traiterons successivement les contraintes (G, F, H) puis les variables intermédiaires R.

Les contraintes G à résoudre à l'égalité constituent la première catégorie de variables à traiter : dans le principe fondamental, elles jouent le rôle de $\mathcal{I}$. Les variables restantes (F, H, R, X) représentent les «contraintes» $\mathcal{K}$. Les relations issues de cette triangulation s'écrivent de manière symbolique sous la forme (3.38 G_a,G_b).

$$\left\{\begin{array}{ll}
(G_a) & G_1 = MD \times \{F, H, R, X\} \cup \{G_2\} \\
(G_b) & \qquad\qquad MG \times \{F, H, R, X, G_2\} = 0 \\
(F_a) & F_1 = MD \times \{H, R, X\} \cup \{G_2, F_2\} \\
(F_b) & \qquad\qquad MG \times \{H, R, X, G_2, F_2\} = 0 \\
(H_a) & H_1 = MD \times \{R, X\} \cup \{G_2, F_2, H_2\} \\
(H_b) & \qquad\qquad MG \times \{R, X, G_2, F_2, H_2\} = 0 \\
(R_a) & R_1 = MD \times \{X\} \cup \{G_2, F_2, H_2, R_2\} \\
(R_b) & \qquad\qquad MG \times \{X, G_2, F_2, H_2, R_2\} = 0 \\
(X_a) & X_1 = MD \times \{G_2, F_2, H_2, R_2, X_2\} \\
(X_b) & \qquad\qquad MG \times \{G_2, F_2, H_2, R_2, X_2\} = 0
\end{array}\right. \tag{3.38}$$

Le principe fondamental est ensuite appliqué au système(3.38 G_b) pour exprimer les variables F en fonction des variables restantes (H, R, X et G_2). Ce traitement est répété récursivement pour l'expression des catégories de variables restantes (H, R, X).

Le processus de remontée (ou substitution arrière) est déclenché et donne lieu aux relations (3.39).

$$\left\{\begin{array}{ll}
(a) & G_1 = MD \times \{X_1\} \cup \{G_2, F_2, H_2, R_2, X_2\} \\
(b) & F_1 = MD \times \{X_1\} \cup \{G_2, F_2, H_2, R_2, X_2\} \\
(c) & H_1 = MD \times \{X_1\} \cup \{G_2, F_2, H_2, R_2, X_2\} \\
(d) & R_1 = MD \times \{X_1\} \cup \{G_2, F_2, H_2, R_2, X_2\} \\
(e) & X_1 = MD \times \{G_2, F_2, H_2, R_2, X_2\}
\end{array}\right. \tag{3.39}$$

Dans les relations écrites ci-dessus figure la totalité des termes « résiduels » pouvant apparaître, conformément aux règles d'application du principe fondamental de résolution.

Lors du traitement des contraintes F, G et H, l'apparition de termes G_2, F_2 et H_2 dans les membres droits des équations (3.38 G_a, F_a, H_a) peut être révélateur d'une « mauvaise formulation » du problème. En raison de la faible fréquence d'apparition de telles situations dans l'utilisation pratique du solveur, l'analyse circonstanciée de tous les cas possibles ne nous a pas parue primordiale. Cependant, nous donnons ci-après quelques éléments d'interprétation qualitative d'une mauvaise formulation de problème.

Cas où $G_2 \neq \emptyset$

La présence de variables G_2 dans (3.38 G_a) rend compte de l'impossibilité d'exprimer indépendamment la totalité des contraintes G en fonction des inconnues X et des autres variables : l'objectif correspondant aux contraintes G_2 n'est donc pas réalisable. Toutefois, décider l'arrêt du traitement représenterait un comportement trop sévère du système.

Pour assurer un fonctionnement plus souple face à une telle situation, le solveur informe l'utilisateur de l'impossibilité rencontrée et poursuit le traitement en s'efforçant de résoudre les contraintes G_2, non plus à l'égalité stricte mais avec un critère de type moindres carrés : la qualification de ces contraintes passe donc de G à F.

Les relations (3.39-a,b,c) expriment les contraintes en fonction des autres variables. Cette forme se prête bien à la pondération des variables dont l'utilité est double. Celle-ci permet tout d'abord de tenir compte du fait que chaque variable manipulée dans notre approche est homogène soit à une longueur, soit à un angle. Ensuite, la pondération constitue le moyen de classer par ordre de priorité (ou de confiance) les contraintes à résoudre, en particulier dans les procédures de calcul de solution de type norme minimale ou moindres carrés.

Finalement, dans les développements suivants, nous supposerons que les systèmes d'équations issues de cette phase de conditionnement préliminaire sont pondérés et disponibles sous la forme suivante :

$$\begin{cases} (a) & G = MD_g \times X \\ (b) & F = MD_f \times X \\ (c) & H = MD_h \times X \\ (d) & R = MD_r \times X \end{cases} \tag{3.40}$$

La mise en forme des équations en vue de leur résolution est terminée. La prochaine étape consiste à résoudre les inconnues en fonction des contraintes.

3.7.2 Résolution des contraintes égalités

Nous désirons à présent résoudre les inconnues pour satisfaire les contraintes égalités G et les contraintes F à minimiser aux moindres carrés. Nous exploiterons donc les équations issues des systèmes (3.40-a) et (3.40-b).

La mise en forme réalisée par le processus de triangulation permet de dégager des groupes de systèmes linéaires indépendants que l'on peut résoudre séparément. Cette décomposition introduit donc n problèmes Pb_i représentés sur la figure 3.3.

Le principe fondamental de résolution est appliqué successivement aux relations du type (3.40-a) et (3.40-b) pour préparer la résolution des contraintes G et F. Passons en revue les quatres situations possibles à l'issue du traitement des contraintes G.

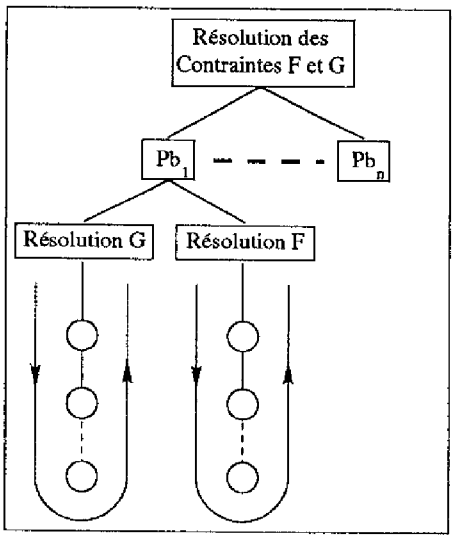

Figure 3.3: Résolution des contraintes F et G

• **Cas C-1**(voir 3.6.5) :
les contraintes sont satisfaites et toutes les inconnues absorbées. Résoudre les contraintes F n'est donc plus possible puisque toutes les inconnues sont déterminées. Dans ce cas, le nœud de résolution des contraintes F n'est pas développé dans le schéma 3.3.

• **Cas C-2 :**
les variables inconnues sont séparées en deux groupes (inconnues principales et secondaires) pour obtenir un résultat intermédiaire de la forme $X_G = MD \times \{G \cup X_F\}$. La résolution de type norme minimale[7] est pratiquée si les contraintes F à optimiser dans (3.40-b) sont absentes.

Dans le cas contraire, le résultat intermédiaire précédent est conservé et les relations provenant du système (3.40-b) sont traitées par le principe fondamental.

La substitution du résultat intermédiaire sur les contraintes G dans les relations (3.40-b) donne des relations de la forme $F = MD \times \{X_F \cup G\}$. La méthode de triangulation pour déterminer X_F se développe et donne lieu à une combinatoire de cas possibles similaire à celle que nous décrivons. Enfin, la résolution finale est accomplie pour donner une relation du type (3.37).

• **Cas C-3 :**
Dans cette situation, les contraintes G sont présentes en nombre excédentaire. Ce défaut de modélisation ayant été signalé à l'utilisateur, la qualification initiale G est modifiée en F. Le jeu correspondant de variables vient s'ajouter aux contraintes à résoudre aux moindres carrés[7].

• **Cas C-4 :**
La méthode de résolution est arrêtée et il est demandé à l'utilisateur d'intervenir pour re-examiner la formulation du problème.

En fonction de la situation rencontrée, le résultat final de cette étape du calcul sera de

[7]Les schémas numériques employés sont décrits dans [ANGELES 89] à partir de méthodes développées plus en détails dans [GOLUB 89].

la forme suivante :

$$\begin{cases} \text{(C-1)} & X = MD \times G \\ \text{(C-2)} & X = MD \times \{G, F\} \\ \text{(C-3)} & X = MD \times F \end{cases} \tag{3.41}$$

3.7.3 Résolution des contraintes inégalités

Après avoir examiné le cas des contraintes F et G, nous nous intéressons au traitement des équations (3.40-c) correspondantes aux contraintes inégalités.

Cette étape coïncide avec le développement d'un nouveau nœud de l'arbre de résolution décrit par la figure 3.4. Comme pour le traitement de F et G, les relations (3.37-c) peuvent conduire à une décomposition en problèmes indépendants qui n'est pas nécessairement identique à celle pratiquée pour F et G dans la figure 3.3. On obtient un découpage en problèmes Pb_i comme indiqué sur la figure 3.4.

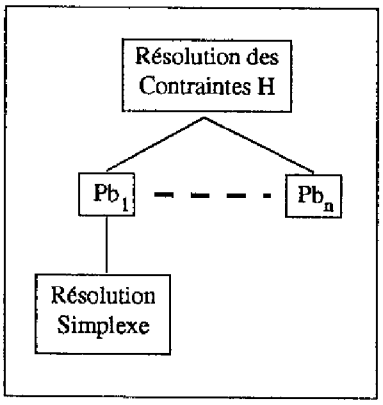

Figure 3.4: Résolution des contraintes inégalités H.

Le principe proposé de prise en compte des contraintes inégalités consiste à évaluer, à partir du résultat du traitement des contraintes F et G, des valeurs pour les termes de la matrice de pondération Λ_n dans la relation (3.21).

Description de la méthode

La résolution de F et G (relations (3.41-a,b,c)) conduit au calcul de l'incrément Δx_n par la relation :

$$\Delta x_n = -\alpha_n MD \times \Lambda_n \times \begin{bmatrix} \Delta G \\ \Delta F \end{bmatrix} \tag{3.42}$$

dans laquelle Λ_n est égale à la matrice identité notée Id.

La substitution de (3.42) dans (3.40-c) permet de calculer ΔH_n et contrôler si les contraintes H sont violées. Si elles ne le sont pas, le traitement s'arrête.

Dans le cas contraire, (3.42) est réécrite sous la forme suivante :

$$\Delta x_n = -\alpha_n MD \times \Lambda_n \times \begin{bmatrix} \Delta F \end{bmatrix} \underbrace{-\alpha'_n \times MD' \times \Lambda'_n \times \begin{bmatrix} \Delta G \end{bmatrix}}_{K} \tag{3.43}$$

avec $\Lambda'_n = Id$ et $\alpha'_n = 1$.

On ne peut pas agir sur le terme K qui représente l'objectif de satisfaction des contraintes égalités (on souhaite converger rapidement sur ces contraintes d'où les valeurs $\alpha'_n = 1$ et $\Lambda'_n = Id$).

En revanche, les contraintes F à résoudre aux moindres carrés nous donnent une latitude d'action pour satisfaire les inégalités[8]. L'injection de (3.43) dans (3.40-c) nous fournit une relation du type :

$$\Delta H_n = -\alpha_n M D \times \Lambda_n \times \left[\ \Delta F \ \right] + K' \tag{3.44}$$

Nous pouvons remarquer à la lumière de cette expression que la modulation de la valeur des cœfficients $\lambda_{i,i}$ aura une influence sur les contraintes H qui doivent rester positives. Comme nous allons le montrer, le calcul de ces termes pondérateurs revient à résoudre un problème d'optimisation linéaire. L'optimisation linéaire, parfois appelée programmation linéaire, s'intéresse au problème suivant :

Étant données n variables indépendantes :
$X = \left[\ x_1, \ldots, x_n \ \right]^T$
maximiser la fonction scalaire :
$z = d^T \times X$ avec les contraintes :
$X \geq 0$
$A \times X = b$
A, b et d correspondent, respectivement, à une matrice
et deux vecteurs de cœfficients réels.

Pour calculer les cœfficients $\lambda_{i,i}$, nous sommes amenés à résoudre le problème d'optimisation suivant, présenté sous une forme non standard :

$$\begin{cases} \text{Maximiser} \\ (a) \quad z = d^T \times X \quad \text{avec} \\ \qquad d = \left[\ 1 \ldots 1 \ \right]^T \\ \qquad X = \left[\ \lambda_{1,1} \ldots \lambda_{n,n} \ \right]^T \\ \text{soumis à :} \\ (b) \quad 0 \leq \lambda_{i,i} \leq 1 \\ (c) \quad h_{n+1} = h_n + \Delta h_n \geq 0 \end{cases} \tag{3.45}$$

La positivité des cœfficients $\lambda_{i,i}$ assure la décroissance simultanée de toutes les erreurs tandis que la fonction critère z tente de rendre égale à n la somme des $\lambda_{i,i}$ (tous les λ égaux à un). Ces deux éléments réunis représentent une condition nécessaire à la convergence *quadratique* de l'algorithme de Newton-Raphson.

Par introduction de variables vectorielles complémentaires B et y_n, les conditions (3.45-b,c) peuvent être transformées sous leur forme standard :

$$(3.45-b) \Longleftrightarrow \begin{cases} \lambda_{i,i} \geq 0 \\ \lambda_{i,i} - 1 + b_i = 0 \\ B \geq 0 \end{cases} \qquad (3.45-c) \Longleftrightarrow \begin{cases} h_n + \Delta h_n - y_n = 0 \\ \text{avec} \\ y_n \geq 0 \end{cases}$$

[8]Cette latitude d'action existe d'une manière analogue lorsque les contraintes F sont moins nombreuses que les inconnues (cas C-2 : résolution de type norme minimale). Le développement de ce cas, très similaire au cas exposé, n'est pas donné.

Le problème d'optimisation linéaire à traiter est de la forme :

$$\begin{cases} d = \left[\ 1 \ldots 1 \quad 0 \ldots 0\ \right]^T \\ X = [\ \underbrace{\lambda_{1,1} \ldots \lambda_{n,n}}_{\text{Variables hors base}} \quad \underbrace{y_n^T \quad B^T}_{\text{Variables de base}}\ ]^T \\ \text{Sous les contraintes :} \\ \begin{cases} \lambda_{i,i} \geq 0 \\ \lambda_{i,i} - 1 + b_i = 0 \\ B \geq 0 \\ C \times X = k \quad {}^{9} \\ y_n \geq 0 \end{cases} \end{cases}$$

Ce problème est alors résolu par l'algorithme du simplexe [PRESS 88]. Si une solution existe, les variables de base optimales $\lambda_{i,i}$ sont évaluées et réinjectées dans (3.44) pour le calcul de l'incrément Δx_n.

Exemple en deux dimensions :

Considérons un problème plan de trajectographie de type(3.1-b). On désire générer une trajectoire continue de A vers B, points de l'espace d'état (x_1, x_2).

L'état courant et la distance au but sont représentés par deux couples de variables (x_1, x_2) et les contraintes à annuler (f_1, f_2). Dans la configuration de cet exemple, les mesures de l'état et de l'erreur sont identiques.

Les parties grisées de la figure 3.5-a expriment que l'erreur ne peut que décroître à l'issue de l'itération suivante. Dans le cas d'une convergence quadratique, la résolution par l'algorithme de Newton-Raphson déplace le point mobile sur la trajectoire radiale indiquée en pointillés.

La figure 3.5-b illustre les effets de la modulation des cœfficients $\lambda_{i,i}$, sur la trajectoire nominale de convergence quadratique.

En présence d'un obstacle vertical entre A et B, la trajectoire se trouve déformée comme indiqué sur la figure 3.5-c. Au terme de la résolution, le point mobile sera en position C où seule la contrainte f_2 est satisfaite.
Ce cas correspond à la solution fournie par le simplexe : $(\lambda_{1,1}=0, \lambda_{2,2}=1)$.

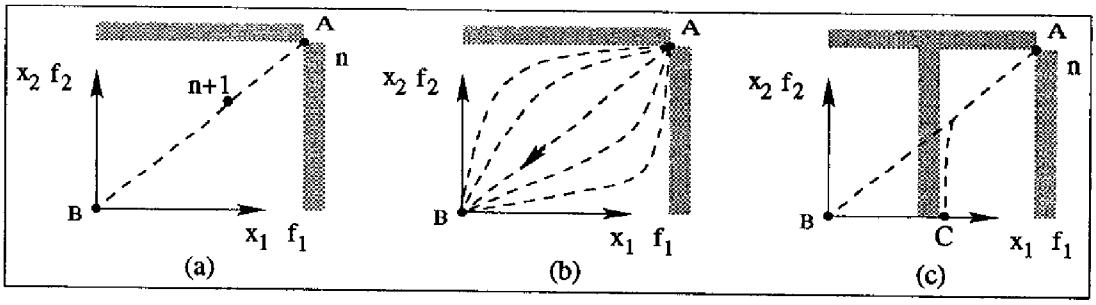

Figure 3.5: Exemple plan illustrant le traitement de contraintes inégalités.

[9] La matrice C et le vecteur k sont construits à partir de l'égalité $h_n + \Delta h_n - y_n = 0$.

Limitations de l'approche - Extensions

Tout d'abord, comme toute méthode locale, l'approche proposée peut ne pas trouver de solution au problème même s'il en existe une. Un exemple illustratif de cette situation est le blocage de l'objet mobile dans la concavité d'un obstacle.

D'autre part, la stratégie implicitement appliquée par cette méthode consistant à « améliorer » à chaque itération la satisfaction des contraintes peut se révéler trop limitative et conduire à des domaines de recherche de solutions restreints de façon arbitraire (dans l'exemple exposé précédemment, le domaine présente une forme rectangulaire).

Une politique plus souple serait d'autoriser des valeurs négatives pour les cœfficients λ permettant ainsi, de manière limitée, des mouvements de « contournement » des obstacles. Mais cette relaxation de contraintes peut affecter la convergence de l'algorithme qui risque de devenir problématique.

Pour y remédier, une solution possible serait de contrôler la convergence non plus à partir d'un critère strict de décroissance des erreurs mais plutôt par des conditions données par des fonctions de *Lyapunov*, introduites dans l'espace des variables F (dans l'exemple des figures 3.5, de telles fonctions remplaceraient la zone rectangulaire en une ellipse, centrée en B et de petit-axe BA). En fait, ces fonctions seraient approximées par des hyperplans construits à partir de raisonnements sur les distances entre le point mobile, l'objectif et les obstacles.

Des développements partiels dans cette voie ont été menés et ont montré que cette approche n'était pas simple à mettre en œuvre. Outre la difficulté d'implantation, une difficulté particulière concerne la gestion des erreurs non-linéaires induites par l'utilisation dans le simplexe d'un modèle au premier ordre.

3.7.4 Calcul itératif et vitesse de convergence

Les étapes décrites jusqu'à présent relevaient de calculs essentiellement non itératifs pour l'évaluation de l'incrément de vecteur d'état Δx_n. La résolution globale du problème fait appel à l'équation récurrente : $x_{n+1} = x_n + \Delta x_n$, permettant d'évaluer l'état suivant (voir figure 3.2). L'arrêt de ce processus est donné par une condition de la forme $||\Delta x_n|| \leq \epsilon$. Dans l'application pratique de la méthode, les bornes maximum sur les erreurs sont de l'ordre de 10^{-3} mm pour les longueurs et 10^{-4} rad pour les angles.

Une valeur faible pour $||\Delta x_n||$ peut être causée par deux facteurs : le premier est la satisfaction des contraintes (dans ce cas, les termes ΔG et $\left[\ \frac{\partial f}{\partial x}\ \right]^T f$ sont nuls), le second correspond à l'annulation des cœfficients de pondération α_n et/ou λ. La signification des termes de la matrice Λ_n a été explicitée au paragraphe précédent, traitant de la prise en compte de contraintes inégalités. Examinons plus en détails le rôle du cœfficient α_n.

Nous avons fait plus haut une interprétation du cœfficient α_n en termes de propriétés de convergence de la résolution numérique. L'interprétation suivante peut également être faite dans un contexte orienté vers la robotique.

La modélisation des problèmes que nous proposons définit des buts à atteindre sous la forme de contraintes que doivent vérifier certaines variables. Ces objectifs peuvent être hiérarchisés en distinguant les contraintes qui devront être satisfaites impérativement à tout instant (par exemple, la contrainte de fermeture d'un mécanisme bouclé ou la condition de saisie d'une pince sur un objet) de celles qui correspondent par exemple à une position

finale à atteindre. Les premiers expriment un besoin de convergence « rapide » (contraintes G) tandis que les seconds correspondent plutôt à une convergence « lente » (contraintes F).

En considérant la forme la plus générale où interviennent simultanément des variables F et G (relation (3.41 C-2), l'incrément Δx_n se calcule par la relation (3.42) où l'on suppose la matrice Λ_n égale à l'identité. Cette relation peut aussi s'écrire de la façon suivante :

$$\Delta x_n = -MD \times \begin{bmatrix} \begin{bmatrix} 1 & & 0 \\ & \ddots & \\ 0 & & 1 \end{bmatrix} & 0 \\ 0 & \begin{bmatrix} \alpha_n & & 0 \\ & \ddots & \\ 0 & & \alpha_n \end{bmatrix} \end{bmatrix} \times \Lambda_n \times \begin{bmatrix} \Delta G \\ \Delta F \end{bmatrix}$$

Les cœfficients α_n affectés aux variables G sont pris égaux à un pour autoriser une convergence rapide (au mieux quadratique). Sur le plan applicatif, des contraintes G doivent être satisfaites au premier ordre, soit par calcul géométrique (voir contact de type normale commune § 2.4.5), soit après résolution itérative en tant que contraintes F.

La résolution progressive des autres objectifs F est assurée par des valeurs de α_n strictement inférieures à un. Dans ce contexte d'utilisation, les variables F ne correspondent donc pas exactement à des contraintes à résoudre au moindres carrés (cas des problèmes de calibration), mais plutôt aux objectifs terminaux de la tâche à réaliser (cas des problèmes de mouvement).

3.8 Extraction de matrices de sensibilité

L'étude de sensibilité d'un système a pour objet de déterminer l'influence de la variation de certains termes entrant en jeu dans la modélisation du système sur d'autres variables.

L'exploitation des résultats intermédiaires et final des manipulations de systèmes variationnels fournit directement ce type d'information.

3.8.1 Sensibilité par rapport aux contraintes

Les résultats de résolution (3.41) définissent la relation linéaire entre des variations sur les contraintes et les variations sur les inconnues recherchées.

À partir d'un état x^* correspondant à la satisfaction des contraintes, ces relations permettent d'évaluer la variation autour de x^* si l'on introduit des erreurs sur la satisfaction expérimentale des contraintes.

Cette possibilité est très intéressante en pratique, en particulier, pour la prise en compte des erreurs inhérentes aux protocoles de calibration.

3.8.2 Sensibilité par rapport à des paramètres

De manière similaire, on peut désirer connaître l'influence sur le résultat, d'une erreur sur des paramètres de description du modèle employé. Pour cela, il est nécessaire de conserver la variation de ces termes de type paramétrique dans les étapes de calcul.

Cet objectif est atteint en dotant les variables concernées d'un statut différent de celui de paramètre puisque ceux-ci sont éliminés d'emblée des calculs (leur variation étant nulle par essence). Notons A, cette nouvelle catégorie de variable.

Le comportement de ce type de variables est intermédiaire entre celui des variables R et X. La forme du résultat obtenu, après introduction de variables A, est le suivant :

$$\left\{\begin{array}{lll} \text{(C-1)} & X & = & MD\times\{G,A\} \\ \text{(C-2)} & X & = & MD\times\{G,F,A\} \\ \text{(C-3)} & X & = & MD\times\{F,A\} \end{array}\right. \tag{3.46}$$

Nous verrons dans le chapitre suivant consacré aux applications un exemple d'exploitation de matrices de sensibilité.

3.9 Détection et gestion des singularités

Nous avons déjà décrit le traitement des singularités numériques induites par la présence de termes proches de zéro. Le mécanisme de détection de zéros structuraux ou conjoncturels permet d'obtenir des systèmes bien conditionnés pour la résolution.

Une autre catégorie de singularités trouve son origine dans la discrétisation introduite par le processus de résolution itératif. Ces singularités se manifestent, entre deux itérations successives, par le changement de signe d'un ou plusieurs pivots.

En effet, en réduisant l'amplitude des incréments dans le processus itératif (i.e. en se rapprochant d'une évolution continue des variables), la valeur de ces pivots serait devenue nulle. La nullité d'un pivot provoque une singularité puisque ce dernier joue le rôle de dénominateur dans le mécanisme de Gauss.

La détection de ces changements de signe nous renseigne donc sur la proximité d'une situation singulière. La singularité peut provenir d'un changement de zéros conjoncturels : cet événement signale que la forme des relations linéaires à résoudre a changé par rapport aux itérations précédentes.
L'animation d'un mécanisme rotatif illustre bien ce cas de figure (exemple : un système bielle-manivelle). En effet, la présence de termes trigonométriques donne lieu à l'annulation périodique des variables régissant les lois du mouvement d'un tel système. La singularité est alors traitée en reconstruisant les systèmes d'équations en utilisant de nouveaux pivots.

L'annulation d'un pivot peut également révéler une singularité intrinsèque au problème que l'on se propose de résoudre. Nous avons déjà évoqué un tel cas de singularité avec l'exemple d'un manipulateur que l'on commande pour rallier un point inatteignable.

Dans l'état actuel du système, les singularités de ce type entraînent l'arrêt de la résolution, accompagnée d'un compte-rendu de l'anomalie à l'utilisateur. En cas de besoin, il serait parfaitement envisageable de poursuivre la résolution après avoir introduit des contraintes supplémentaires (comme se placer à distance minimale, dans l'exemple précité).

3.10 Un exemple simple d'utilisation de l'approche

En guise d'illustration, nous nous proposons de traiter le problème de recherche de l'intersection de deux paraboles. Il s'agit d'un problème non linéaire de complexité « minimale ».

Deux paraboles P_0 et P_1 sont décrites au moyen de deux points mobiles M_0 et M_1 de coordonnées $\begin{bmatrix} x_0 \\ y_0 \end{bmatrix}$ et $\begin{bmatrix} x_1 \\ y_1 \end{bmatrix}$. Partant de positions de M_0 et M_1 données, l'objectif est de déterminer l'intersection des paraboles. Pour cela, nous choisissons x_0 et x_1 comme variables d'état de ce système que l'on doit faire évoluer pour annuler la distance M_0M_1.

Pour illustrer le comportement de la méthode de résolution vis à vis des singularités, nous allons considérer le cas où les paraboles ne sont pas sécantes (figure 3.6). En partie gauche du tableau 3.2 figure un modèle d'état de la tâche décrivant les variables mises en jeu, et en partie droite, leur modèle variationnel associé.

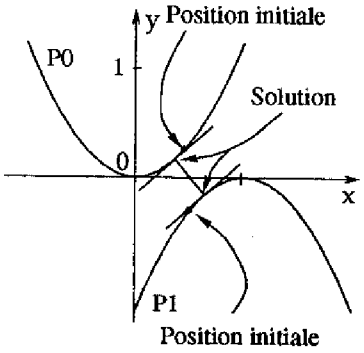

Figure 3.6: Intersection de deux paraboles.

Le problème comporte :

- deux inconnues (x_0, x_1),
- deux contraintes à résoudre en convergence lente (e_x, e_y),
- et deux variables intermédiaires (y_0, y_1).

L'élimination des variables intermédiaires Δy_0 et Δy_1 des relations du modèle variationnel fournissent les relations (a et b).

À partir de la condition initiale $x_0 = 0.51$ et $x_1 = 0.52$, le mécanisme de choix du meilleur pivot impose le calcul de Δx_0 avec l'équation (b) et le pivot $-2x_0$. Sa valeur, donnée par la relation (c), est injectée dans (a) et conduit à l'équation (d).

Le pivot de la variable suivante Δx_1 noté p est écrit dans l'équation (e). Le calcul de Δx_1 peut être réalisé grâce à la relation (d) si p n'est pas nul. Bien évidemment, ces choix de pivots peuvent changer automatiquement au gré des itérations, conduisant au calcul de Δx_1 puis Δx_0.

Au cours du processus itératif de résolution, des changements répétés du signe de p indiquent la présence d'une singularité. La solution au problème ne pouvant pas être trouvée, la résolution s'arrête avec un compte rendu de la singularité rencontrée.

La nullité de p s'interprète aisément sur cet exemple comme correspondant aux points en lesquels les tangentes à P_0 et P_1 sont parallèles (le numérateur de p est proportionnel à la différence des pentes des tangentes).

Pour éviter l'arrêt de la résolution et tenter de répondre à la question, il faut reformuler légèrement l'objectif initial qui se trouve être inatteignable.

Modèle d'état	Étiq.	Modèle variationnel
x_0, x_1	$\mathcal{X}$	
$y_0 = x_0^2$	$\mathcal{R}$	$[1, -2x_0] \times [\Delta y_0, \Delta x_0]^T = 0$
$y_1 = -(x_1 - 1)^2$	$\mathcal{R}$	$[1, 2(x_1 - 1)] \times [\Delta y_1, \Delta x_1]^T = 0$
$e_x = x_0 - x_1$	$\mathcal{F}$	$[1, -1, 1] \times [\Delta e_x, \Delta x_0, \Delta x_1]^T = 0$
$e_y = y_0 - y_1$	$\mathcal{F}$	$[1, -1, 1] \times [\Delta e_y, \Delta y_0, \Delta y_1]^T = 0$

Tableau 3.2: Modélisation de l'intersection de deux paraboles.

(a)	$[1, -1, 1] \times [\Delta e_x, \Delta x_0, \Delta x_1]^T = 0$
(b)	$[1, -2x_0, -2(x_1 - 1)] \times [\Delta e_y, \Delta x_0, \Delta x_1]^T = 0$
(c)	$\Delta x_0 = [\frac{1}{2x_0}, \frac{1-x_1}{x_0}] \times [\Delta e_y, \Delta x_1]^T$
(d)	$[1, \frac{-1}{2x_0}, \frac{x_0+x_1-1}{x_0}] \times [\Delta e_x, \Delta e_y, \Delta x_1]^T = 0$
(e)	$p = \frac{x_0+x_1-1}{x_0}$: Étiq. $\mathcal{G}$
(f)	$[-1, \gamma_0, \gamma_1] \times [\Delta p, \Delta x_0, \Delta x_1]^T = 0$
(g)	p : Étiq. $\mathcal{G}$ e_x, e_y : Étiq. $\mathcal{F}$

Tableau 3.3: Équations de résolution.

Puisque les courbes P_0 et P_1 ne se coupent pas, un objectif raisonnable et voisin du problème initial pourra être de déterminer les points réalisant la distance minimum entre les deux paraboles.

Ceci peut être spécifié très facilement. Le nouveau problème à résoudre fait intervenir une nouvelle contrainte à résoudre en priorité : il s'agit de l'annulation du pivot p pour garantir que les tangentes en M_0 et M_1 sont parallèles. Pour cela, la variable p est dotée d'un qualificatif $\mathcal{G}$. Il faut, de plus, ajouter la relation variationnelle (f) correspondante à l'annulation de p, les termes γ_0 et γ_1 pouvant être évalués formellement ou par calcul différentiel numérique.

La méthode de résolution peut alors être relancée et résoud cette fois le problème de minimisation sous contrainte égalité :

$$\min_{x_0, x_1} (e_x^2 + e_y^2) \quad \text{avec la contrainte} \quad p = 0$$

Avec un cœfficient α_n bornant les variations des variables $\mathcal{X}$ à 0.01, le système trouve la solution : $x_0 = 0.39$ et $x_1 = 0.61$.

Le mécanisme de détection de singularités laisse entrevoir la possibilité de rendre systématique la modification automatique d'un problème lorsqu'il devient singulier. La généralisation à des dimensions de problèmes plus élevées et la gestion du coût des calculs au

second ordre sont cependant des éléments qui incite à une certaine prudence.

3.11 Implantation de la méthode de résolution

Comme nous l'avons vu, le système de résolution possède une structure arborescente. Sa création consiste principalement à mettre en place les nœuds mentionnés sur la figure 3.2. Son évolution est intrinsèquement dynamique, guidée par les contingences numériques du problème à résoudre.

La mise en œuvre des manipulations hybrides numériques/formelles que nous avons décrites plus haut a demandé le développement de pas moins d'une trentaine d'opérateurs matriciels spécifiques (concaténation de matrices, substitution de variables...).

La nature arborescente du système associée à la volonté d'améliorer les performances de traitement nous ont conduit à définir la notion de « compilation ». La compilation de l'arbre a pour objet d'allouer les matrices et définir l'ensemble des opérateurs matriciels nécessaires à la résolution d'un problème donné.

Un état compilé correspond à la situation où les structures mises en place dans l'arbre peuvent être réutilisée sur plusieurs itérations successives. Cette possibilité améliore fortement les performances. La rapidité de calcul, pour une itération, peut s'accroître d'un facteur quatre par rapport à une itération comprenant la phase de compilation de l'arbre.

Évoquons brièvement les modalités d'utilisation des structures compilées dans l'arbre de résolution. La validité de la compilation est directement liée à la détection de singularités telles que le changement de zéros structuraux, la modification de l'ordre des variables sélectionnées par des choix de meilleurs pivots ou encore, un changement de signe de ces pivots. Dans l'une quelconque de ces circonstances, l'arbre doit être recompilé à partir du nœud où est survenu la singularité, toute la partie amont demeurant valide.

Bien d'autres « détails » de mise en œuvre (ne présentant pas d'intérêt théorique majeur) sont passés sous silence. Nous conclurons ce volet implantation en signalant que de nombreuses informations complémentaires sont contenues dans le code C++ développé.

3.12 Conclusion

La méthode de résolution présentée dans ce chapitre constitue un outil général pour la résolution de fonctions non linéaires dont on connaît, numériquement, un modèle d'état et le modèle variationnel au premier ordre. Elle s'applique également au traitement de problèmes d'optimisation de fonctions non linéaires soumises ou non à des contraintes de type égalité ou inégalité. Le système met en œuvre un mécanisme de triangulation de Gauss qui permet d'alimenter un schéma numérique itératif de type Newton-Raphson associé à une méthode du simplexe.

La démarche proposée opère des manipulations hybrides numériques/formelles sur les systèmes linéaires variationnels associés aux fonctions à résoudre. Au titre des bénéfices de cette approche, nous pouvons mentionner l'accroissement de robustesse des traitements numériques, la gestion dynamique des processus de résolution à appliquer et la décomposition automatique du problème initial en sous-problèmes indépendants.
Le procédé de résolution itératif autorise la génération de trajectoires continues dans l'espace d'état. Le contrôle des vitesses de convergence est exploité, d'une part pour borner

l'amplitude des incréments d'évolution, d'autre part pour assurer localement l'évitement d'obstacles.
La maîtrise du grand nombre de variables introduites dans la modélisation entraîne une flexibilité d'utilisation de la méthode importante et une grande variété d'applications potentielles.

L'ensemble de ces caractéristiques fait de ce module de résolution, un outil adapté à l'animation de mécanismes complexes mais convient également au traitement de problèmes d'identification paramétrique de modèles.
Ces volets applicatifs correspondent à deux besoins importants rencontrés en robotique pour accomplir des tâches de manipulation complexes dans des environnements mal connus *a priori*.

Chapitre 4

Application à l'exécution de tâches robotisées

L'objet de ce chapitre est de présenter l'utilisation concrète de notre méthodologie de spécification de tâches. Après avoir exposé les principes de fonctionnement de l'environnement de simulation graphique, nous illustrerons les possibilités d'utilisation du système à travers une série d'exemples.

4.1 Exploitation de l'environnement de simulation graphique

4.1.1 Présentation générale

L'environnement de programmation de robots manipulateurs est constitué de différents modules fonctionnels organisés autour d'un simulateur graphique qui comprend le modeleur d'objets 3D décrit dans le chapitre 2 (TRID) associé à un interpréteur de commandes.

Le schéma 1.1 du chapitre 1 donne les principales connexions entre les modules. Les nouvelles fonctionnalités apportées par chacun d'entre eux peuvent être intégrées si nécessaire à l'interpréteur. Le développement de ce système, débuté en 1990, est le fruit d'un travail d'équipe avec A.GIRAUD. Les astérisques signalent les modules à la conception desquels nous avons participé et les parties grisées correspondent à notre contribution sur le plan de l'implantation informatique.

D'un point de vue logiciel, le système a été développé dans l'esprit de la conception orientée objets en C++. Le souhait de conserver l'ouverture maximum de l'application pour les extensions futures est la principale raison pour laquelle certaines parties du logiciel ont été développées localement au LAAS malgré l'existence de bibliothèques commerciales ou du domaine public [CAMERON 89, HAYWARD 88] : c'est le cas notamment du modeleur 3D ou certaines fonctions relatives à la commande de manipulateurs.

En effet, l'une des plus grandes difficultés rencontrée au cours du développement d'un tel système est de réussir à intégrer au mieux des fonctionnalités diverses. Pour y parvenir, il est évident qu'une homogénéité dans la conception des différents éléments est très appréciable. Pour cette raison, l'utilisation de sous-systèmes vus comme des boîtes noires (la représentation interne n'étant le plus souvent pas accessible) empêche ou limite ces possi-

bilités d'intégration. Ce choix, qui peut ne pas être applicable dans un contexte autre que celui de la recherche, nous garantit la maîtrise complète de l'ensemble du logiciel.

4.1.2 Modalités d'utilisation

Nous pouvons différencier deux niveaux d'utilisation du système. Le **niveau 1** d'utilisation correspond à l'exploitation ordinaire du logiciel et le **niveau 2**, au développement de fonctionnalités complémentaires.

Par exemple, l'utilisateur peut avoir besoin, pour la description d'une tâche particulière, de formes ou de relations de contact qui n'existent pas d'emblée dans le module de spécification de tâches (voir l'exemple du contact projectif pour la modélisation d'une caméra vidéo au § 4.9.1). De la même manière, l'utilisation d'un nouveau type de robot nécessitera l'introduction des fonctionnalités spécifiques dans le module dédié à la gestion de manipulateurs, telles que son modèle géométrique ou les fonctions de dialogue avec son armoire de commande. La conception modulaire par objets en C++ facilite grandement la mise en œuvre de ces extensions de modules.

Pour ce qui concerne son exploitation courante, le système comprend deux points d'entrées possibles et complémentaires : l'un par l'intermédiaire de l'interpréteur de commandes, l'autre par interaction graphique avec des fenêtres.

À l'ouverture de la session de travail, l'interpréteur fait la lecture d'un fichier contenant les instructions nécessaires à la construction initiale de l'environnement de travail. La syntaxe de ces commandes s'apparente fortement à celle du C++. Le contenu du script se décompose en deux grandes parties. La première est relative à la création du modèle graphique 3D : paramètres de visualisation, création et positionnement des corps polyédriques. La seconde concerne la définition des tâches proprement dite et comprend la modélisation cinématique et la spécification des problèmes.

Partant de cette base de travail, l'utilisateur peut alors modifier et compléter le système initial soit par l'intermédiaire de l'interpréteur, soit en appelant interactivement grâce à la souris des fonctions disponibles par des systèmes arborescents de fenêtres. Cette double possibilité d'interaction augmente la souplesse d'exploitation du logiciel.

4.1.3 Déroulement d'une spécification

La production de scénarios de spécification de tâches se déroule en deux temps. La première étape est relative à la création des structures de données initiales pour la description des tâches. À partir de cette base, la deuxième partie du travail consiste à mettre au point le scénario par des étapes successives de vérification/modification.

Création

La première phase du travail consiste à élaborer le script définissant la tâche à modéliser et son contexte. Sa construction va se dérouler selon les étapes suivantes :

- Construction du modèle 3D du système considéré.
- Positionnement de demi-contacts sur les corps définis précédemment.

- Définition de la cinématique du mécanisme par l'installation d'articulations entre les corps.
- Définition des relations de contacts à établir entre les objets.

Les deux principales difficultés de la définition initiale du script sont la déclaration des qualificatifs de mobilité entre les demi-contacts et le positionnement de ceux-ci.

Le premier objectif repose sur l'expertise de l'opérateur qui connaît les objectifs de la tâche et peut donc les traduire en termes de mobilités relatives entre les demi-contacts.

Cependant, pour aider à cette tâche, des procédures de plus haut niveau pourraient être introduites et mises à disposition sous la forme de boîte à outils. Par exemple, la mise en place d'un mouvement de deux corps par un contact droite sur droite se limiterait à la sélection sur le graphique de deux arêtes de polyèdres.

La satisfaction du second objectif est grandement facilitée par l'utilisation du modèle polyédrique 3D. Ce dernier donne accès, par des actions interactives sur l'écran, à toute information de distance utile comme la position relative des repères intrinsèques des corps ou les coordonnées de sommets sur les polyèdres. En complément, l'interpréteur offre des fonctionnalités similaires.

Il faut souligner l'importance de cette utilisation, en synergie, des ressources de la représentation graphique et de celles de l'interpréteur qui permettent à l'opérateur de jongler entre des modes de représentation graphique et symbolique.

Vérification et modification du canevas initial

Les modalités de spécification sont le reflet des structures de données internes du logiciel. L'importance, en volume, du nombre des ingrédients entrant en jeu dans la spécification de problèmes souligne la nécessité de moyens de vérification et de modification des structures de données créées initialement par le script.

Les principaux éléments auxquels l'utilisateur doit avoir accès sont les corps, les demi-contacts, les articulations et les contacts. Un problème ayant été construit, il doit pouvoir vérifier et éventuellement modifier les attributs de chaque lien du graphe cinématique. La vérification ou contrôle portera sur l'état et l'erreur des liens. Les modifications interviennent sur la valeur des consignes, des bornes (pour la garde des variables) et des qualificatifs des liens.

4.1.4 Mécanismes d'accès aux structures

Tous les éléments créés explicitement (corps, demi-contacts, articulations et contacts) disposent d'un nom symbolique déterminé par l'utilisateur au moment de l'élaboration du script.

À l'exécution de ce fichier de commande, les ordres symboliques sont interprétés et génèrent à leur tour de nouveaux éléments que l'utilisateur n'a pas explicitement baptisés. En l'absence de nom, ces éléments ne peuvent pas être désignés bien qu'ils soient présents dans la structure de données. L'accès aux informations relatives à ces entités créées dynamiquement a nécessité la mise en œuvre d'une convention de représentation symbolique des constituants des motifs cinématiques. Nous appellerons noms volatiles ces identificateurs attachés aux

entités qu'ils représentent, par opposition aux noms symboliques donnés explicitement par l'utilisateur.

L'existence des noms volatiles a rendu possible le développement de systèmes arborescents de fenêtres permettant à l'utilisateur de passer en revue (pour contrôle et modification éventuelle) la totalité des liens contenus dans les motifs cinématiques. Cette fonctionnalité est essentielle dans le processus de spécification de tâches. Ces symboles étant propagés jusqu'au cœur de la méthode de résolution, ils facilitent la mise au point et le débogage des scripts de spécification.

La figure 4.2 montre l'arborescence des fenêtres au cours de la phase d'examen d'un contact repère sur cylindre représenté sur la figure 4.1. Les caractéristiques du lien entre le repère tangent au cylindre et le repère de l'outil apparaissent dans la fenêtre inférieure.

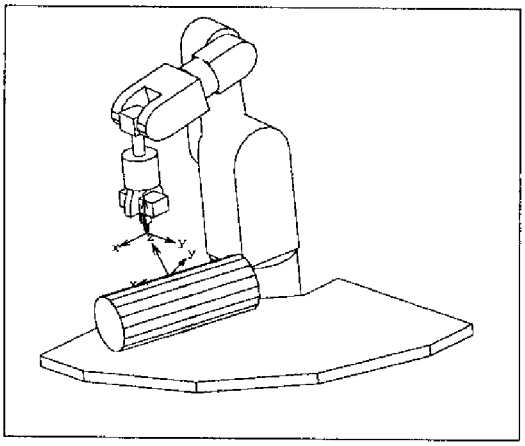

Figure 4.1: Contact repère sur cylindre.

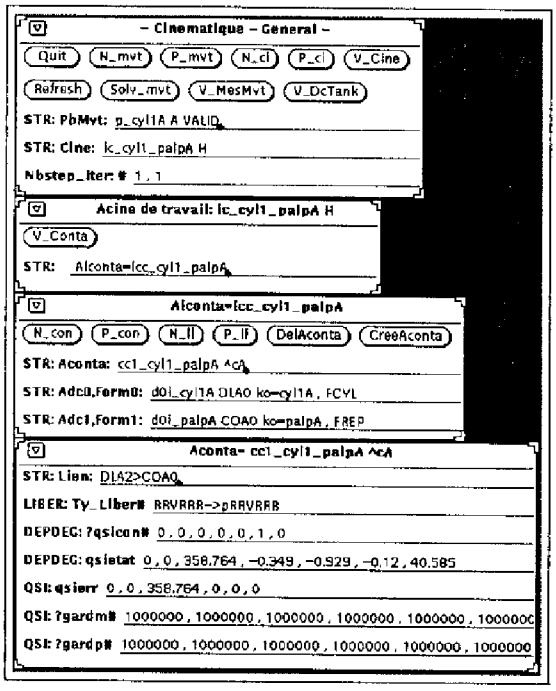

Figure 4.2: Arborescence de fenêtres pour la revue des liens de contact.

Appellation symbolique des structures de spécification

Les problèmes de mouvement étant structurés sous la forme de motifs cinématiques de type distinct, nous avons élaboré un mécanisme automatique de création de noms qui traduit la nature arborescente des motifs cinématiques. De cette façon, tout élément de la spécification possède un identificateur auquel l'utilisateur peut faire référence en cas de nécessité.

Exemple : noms volatiles pour les demi-contacts.

Ces noms sont constitués de quatre caractères $C_1C_2C_3C_4$ dont la signification est décrite ci-dessous :

C_1	Caractère [A-Z,a-z]	Nom du motif cinématique.
C_2	Caractère [A-Z,a-z]	Nom du corps de référence.
C_3	Caractère [A-Z,a-z]	Nom du demi-contact défini par rapport corps de référence.
C_4	Numéro [0-9]	Profondeur dans l'arborescence de demi-contacts.

La figure 4.3 représente des exemples de noms volatiles de demi-contacts construits sur deux corps A et B, eux-mêmes rattachés à un motif cinématique A. Une démarche analogue est employée pour baptiser les articulation et les contacts.

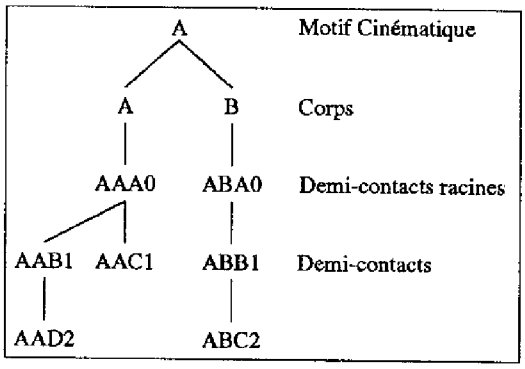

Figure 4.3: Noms volatiles de demi-contacts.

4.1.5 Expression des contraintes par étiquetage

La méthode de spécification proposée est basée sur la qualification des variables intervenant dans la formulation des problèmes. Nous avons vu qu'il existait quatre catégories de variables : les paramètres, les inconnues, les contraintes et les variables intermédiaires. Afin d'accroître la richesse et la souplesse de spécification, il convient d'affiner la classification précédente en proposant des types de variables plus spécialisés.

Les quatre tableaux suivants récapitulent, par catégories, les étiquettes de variables que nous utiliserons dans les exemples qui vont suivre. Le terme **modification** (abrégé en Modif.), présent dans ces tableaux, est relatif à la possibilité de changement de la valeur de consigne d'une variable selon son type d'étiquette.

	Interprétation	**Modif.**
N,P	Paramètres nuls	oui
Z,F	et non nuls	non
M	Paramètre correspondant à une mesure (dual de X)	non
A	Paramètre dont on recherche la sensibilité	oui

Tableau 4.1: Variables paramétriques.

	Interprétation	Garde sur valeur	Garde sur incrément
R	Variable intermédiaire	non	non
S	Variables intermédiaires «gardées» (arrêt du mouvement)	oui	
i		$i \geq 0$	
j		$j \leq 0$	
k		$.. \leq i \leq ..$	

Tableau 4.2: Variables secondaires.

	Interprétation	Valeur	Modif.	Convergence
D[1]	contraintes égalité	nulle	non	rapide
C,U		nulle/non nulle	oui	
V,W	contraintes moindres carrés	nulle/non nulle	oui	lente
I	contraintes inégalité	$I \geq 0$	oui	contrôlée par le Simplexe
J		$J \leq 0$		
K		$... \leq K \leq ...$		

Tableau 4.3: Variables de contraintes.

	Évaluation	Interprétation	Modif.	Garde sur valeur	Garde sur incrément
G	Non itérative en général	Inconnue associée à une propriété géométrique	non	non	non
Q	Itérative	Inconnue (Typiquement articulaire)	oui	oui	
O				oui (Simplexe)	oui
X		Inconnue correspondante à une mesure (dual de M)		non	oui

Tableau 4.4: Variables inconnues.

4.2 Mouvement de deux cubes

Examinons de nouveau l'exemple du positionnement relatif de deux arêtes appartenant à deux cubes. La figure 4.4 rappelle la qualification de chaque lien pour ce problème.

[1]Exprime une propriété géométrique à satisfaire impérativement (usage conjoint avec G)

4.2.1 Traitement des contraintes géométriques impératives

Le placement relatif de deux droites (supports des arêtes) est défini par une distance et un angle. Ces grandeurs géométriques interviennent par l'intermédiaire de la normale commune aux deux droites. Cette notion est introduite dans notre approche au moyen de formes de type droite qui s'appuient sur les demi-contacts $dc1$ et $dc2$. Leur fonction est de mettre en place deux demi-contacts mobiles $dc1_i$ et $dc2_i$ définissant les pieds de la normale commune. Ils possèdent une mobilité exclusivement en translation et rotation selon $\vec{x}$ qualifiée GZZ GZZ. Remarquons que les variables G peuvent être interprétées physiquement comme des composantes de « glissement » ou « roulement » du contact droite sur droite.

La contrainte de contact de $dc2_i$ par rapport à $dc1_i$ est caractérisée par le vecteur d'étiquettes DDW DDW qui exprime que l'on souhaite annuler les translations et rotation selon $\vec{x}$ et $\vec{y}$ et obtenir des valeurs données pour la distance et l'angle suivant $\vec{z}$. Les quatres contraintes D non modifiables doivent être résolues de manière impérative sous peine de ne plus garantir la propriété géométrique de perpendiculaire commune. Les inconnues disponibles pour les résoudre sont les quatres variables G. Les contraintes étiquetées W correspondent à l'objectif de la tâche et doivent donc être résolues avec une convergence lente. En d'autres termes, on souhaite atteindre l'objectif progressivement. Les inconnues affectées à la satisfaction de ces contraintes sont les variables Q du lien articulaire qui permet de déplacer le second cube par rapport au premier.

Il existe deux solutions pour traiter le cas des contraintes impératives D : utiliser le calcul itératif ou bien évaluer d'emblée les valeurs correctes des inconnues G. Dans l'exemple proposé, nous utilisons la seconde solution puisque le positionnement des repères $dc1_i$ et $dc2_i$ est accessible directement par un calcul géométrique. Ce calcul est assuré à chaque itération par une procédure attachée au contact droite sur droite. Les contraintes D sont donc résolues immédiatement sans faire appel au processus itératif. La mise en place de ce type de routines de calcul entre dans le cadre d'une utilisation de niveau 2.

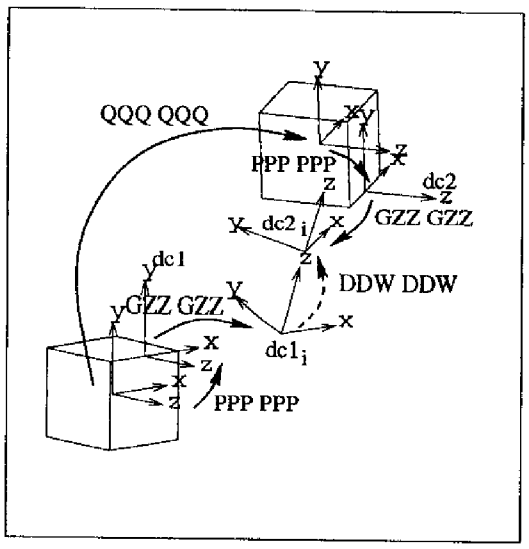

Figure 4.4: Positionnement relatif de deux arêtes.

Remarques

- Comme les variables G sont évaluées par calcul direct, les contraintes D qui leur correspondent dans le lien DDW DDW sont d'emblée satisfaites. Pour cette raison, il n'est pas strictement nécessaire de traiter ces variables G comme des inconnues dans la méthode de calcul. Cette remarque justifie la spécification qui avait été proposée dans l'exposé initial de cet exemple au paragraphe 2.4.5 dans lequel ces variables étaient qualifiées en variables intermédiaires R.

- La spécification du mouvement de deux droites dans l'espace présente une difficulté lorsque ces droites deviennent parallèles. En effet, cette situation introduit une singularité dans la procédure de construction de la normale commune des deux droites. Cette configuration géométrique correspond à une singularité inévitable dès lors que l'on cherche à placer les pieds de la normale. Cependant, si les variables G sont calculées itérativement, il est probable que le « passage » de cette singularité se déroule correctement (grâce au caractère typiquement continu des trajectoires calculée par la méthode de résolution).

4.2.2 Modification interactive de la spécification

Le système de fenêtres permet de visualiser les attributs définissant la relation de contact considérée. Une fenêtre relative à un lien de contact affiche :

- les noms symboliques et volatiles des demi-contacts mis en jeu,
- le qualificatif du lien défini sous la forme d'étiquettes de variables,
- trois vecteurs de configuration désignant respectivement la consigne à atteindre, l'état courant et l'erreur par rapport à cette consigne.

Des raisons d'ergonomie et de confort de lecture nous ont conduit à utiliser, pour l'affichage des informations, des vecteurs de configuration à sept composantes. Les trois premières définissent le vecteur translation en millimètres, les trois suivantes correspondent au vecteur unitaire de rotation et la dernière valeur est égale à l'angle de rotation en degrés. Les deux arêtes étant initialement distantes de 10mm avec un angle de 30 degrés, on souhaite atteindre la configuration 20mm et 65 degrés. Les indications fournies par la fenêtre du lien de contact sont de la forme :

Qualificatif:	DDW DDW						
Consigne:	0	0	20	0	0	1	65
État:	0	0	10	0	0	1	30
Erreur:	0	0	-10	0	0	1	-35

La figure 4.5 donne l'allure du mouvement continu calculé que doit suivre le second cube pour atteindre la configuration but. Cette trajectoire est obtenue avec la mobilité maximale pour le cube (QQQ QQQ) et deux contraintes W. Toutes les variantes de ce problème, réduction de la mobilité, relaxation d'une des deux contraintes, peuvent être traitées par modification des qualificatifs dans les fenêtres ou par l'interpréteur.

Par exemple, si l'on souhaite spécifier incomplètement ce problème en fixant seulement la distance (ou l'angle) entre les deux droites, il suffit de transformer le lien de contact

en DDW DDR (ou DDR DDW). Le mouvement obtenu pour la contrainte DDR DDW est représenté sur la figure 4.6.

Nous pouvons remarquer sur cet exemple et les exemples qui suivront que l'allure des trajectoires obtenues dépendent fortement des conditions initiales de placement des objets.

De par le schéma de résolution numérique adopté, les trajectoires obtenues minimisent, qualitativement, le trajet à parcourir entre les configurations de départ et d'arrivée. Intuitivement, le système tentera de converger vers le point d'équilibre le plus proche de sa condition initiale, ce qui correspond à un objectif raisonnable en pratique.

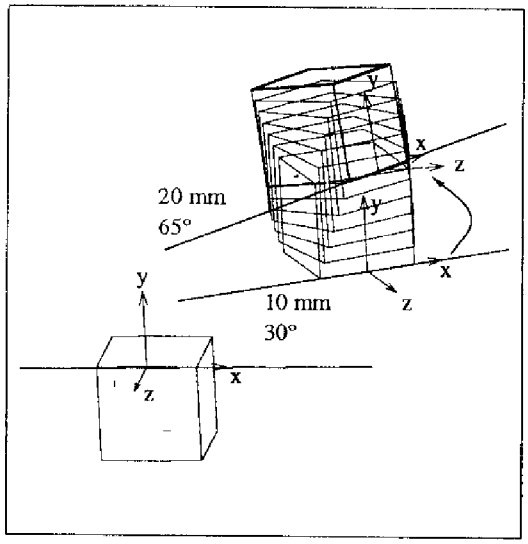

Figure 4.5: Mouvement avec contraintes angulaire et de distance.

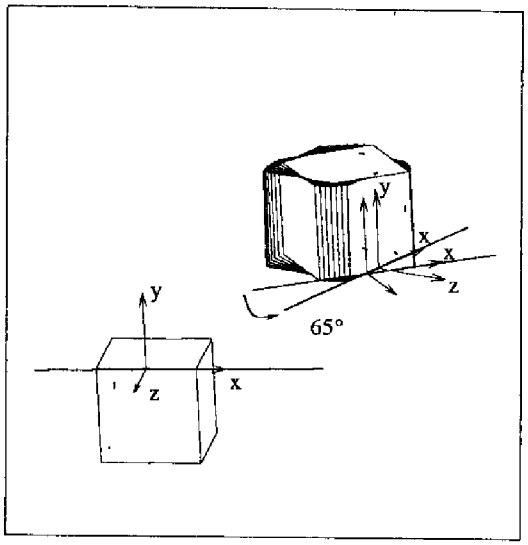

Figure 4.6: Mouvement avec contrainte angulaire seule.

4.3 Traitement de mécanismes comportant des boucles

Revenons à l'exemple du manipulateur à deux degrés de liberté introduit au chapitre 2. Pris séparément, ce robot comporte deux boucles cinématiques, le troisième cycle correspondant à l'expression de la contrainte de tâche. De par la nature même de notre approche, le traitement de mécanismes comportant des boucles est immédiatement accessible. En effet, la notion de cycle sert de support à l'expression des contraintes.

Le manipulateur considéré travaille dans le plan de la figure et les liaisons pivots définissent des mobilités de rotation selon des axes orthogonaux à ce plan. Les liaisons glissières définissent des translations appartenant au plan de la feuille. Arbitrairement, on suppose que l'axe de chaque liaison est confondu avec l'axe $\vec{z}$ des demi-contacts définissant l'articulation considérée. Dans ces conditions, les liaisons pivots (resp. glissières) sont qualifiées par des mobilités de la forme ZZZ ZZQ (resp. ZZQ ZZZ).

Le graphe du mécanisme est rendu arborescent en coupant de manière fictive deux articulations du robot. La figure 4.7 présente l'une des solutions possibles. Afin de rétablir la propriété réelle du mécanisme, on installe sur chaque liaison coupée une relation de contact destinée à « ressouder » l'articulation correspondante.

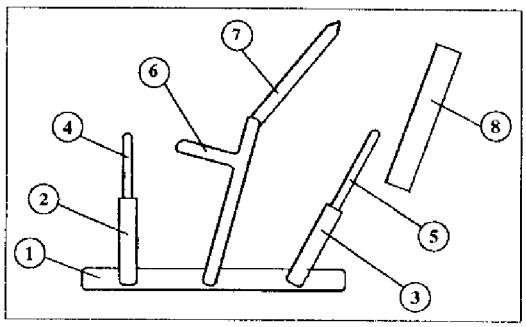

Figure 4.7: Mécanisme arborescent.

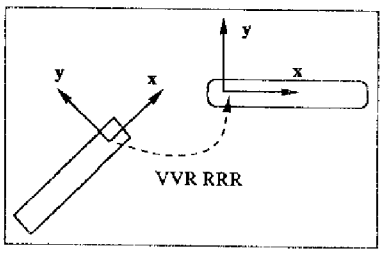

Figure 4.8: Contraintes de fermeture de boucle.

Pour *rétablir* la liaison, autrement dit, annuler les composantes de translation selon $\vec{x}$ et $\vec{y}$, le qualificatif sur ce contact sera du type VVR RRR (figure 4.8).

La liaison étant rétablie, le qualificatif sera cette fois du type DDR RRR pour maintenir nulles les composantes de translation. Ces contraintes de fermeture de boucles sont à résoudre à l'égalité avec une convergence rapide pour éviter l'accumulation des erreurs au second ordre. Les quatre autres composantes étiquetées R ont des valeurs indifférentes. Le problème complet sera donc spécifié comme indiqué sur la figure 4.9.

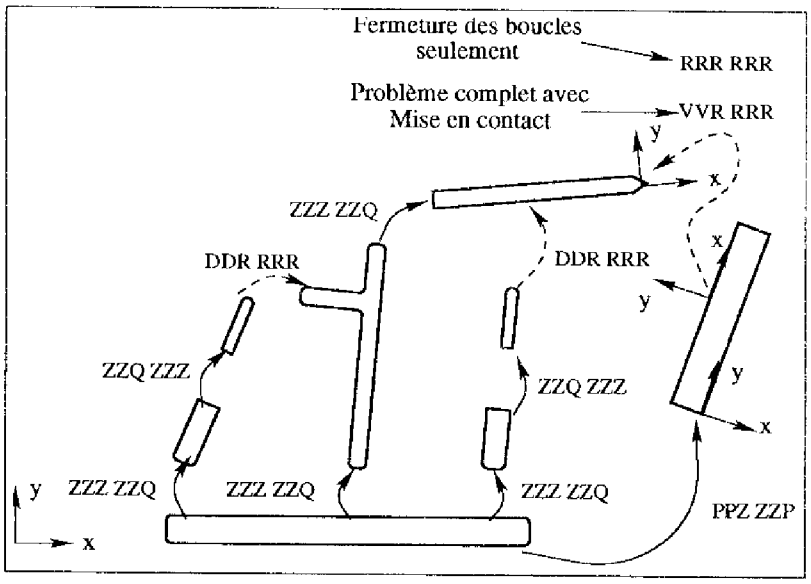

Figure 4.9: Spécification complète.

Les articulations introduisent six variables inconnues (Q) et les contacts de fermeture 2×2 contraintes égalités (D). La résolution du seul problème de fermeture du mécanisme conduit donc à une solution de type norme minimale pour le vecteur formé des six variables Q. En effet, dans ce cas, le contact entre 7 et 8 est qualifié RRR RRR et n'introduit donc pas de contrainte. Le problème complet, à savoir positionner le point terminal sur le corps 8 introduit deux contraintes supplémentaires par la relation de contact VVR RRR. Ces contraintes étiquetées V sont à résoudre avec une vitesse de convergence lente.

4.4 Modélisation de systèmes robotiques

Les systèmes robotiques que nous envisageons de traiter sont essentiellement des bras manipulateurs équipés de préhenseurs. Comme ce fut le cas jusqu'à présent, ces systèmes seront modélisés par des mécanismes formés de corps rigides.

Nous distinguons principalement deux modalités de représentation pour un manipulateur : une approche globale qui décrit les propriétés du système de façon externe et une approche par décomposition dans laquelle il est possible de contrôler les mouvements des corps intermédiaires du robot.

4.4.1 Approche globale

La manière classique de procéder lorsqu'on souhaite modéliser un robot manipulateur consiste à calculer certains modèles qui définissent des relations entre des grandeurs de l'espace articulaire et de l'espace opérationnel dans lequel évolue le repère terminal du robot. Le tableau (4.5) rappelle la nature possible de ces grandeurs en fonction du type de modèle considéré.

Modèle	Espace Articulaire	Espace Opérationnel
Géométrique	Coordonnées articulaires	Position du repère terminal
Cinématique	Vitesses articulaires	Vitesse du repère terminal
Statique	Couples moteurs	Effort transmis au repère terminal

Tableau 4.5: Modèles de description du comportement de robots.

Lorsque l'on cherche à faire fonctionner un robot à des vitesses importantes, il peut être utile de prendre en compte sa dynamique. C'est le cas, par exemple, de l'utilisation de manipulateurs en milieu industriel où la cadence de production constitue un objectif prioritaire. Le modèle dynamique d'un système mécanique articulé traduit les relations entre ses variables d'état (modèle géométrique), ses dérivées et les efforts extérieurs agissant sur le robot.

À l'inverse, le contexte d'exécution des tâches en manipulation distante (environnement mal connu, présence éventuelle d'obstacles) fait que les déplacements du manipulateur se font à faible vitesse. C'est pourquoi nous ne nous intéressons ici qu'au modèle géométrique qui fournit la position de l'organe terminal en fonction des coordonnées articulaires (Modèle Géométrique Direct) ou bien les coordonnées articulaires relatives à une position de l'organe terminal (Modèle Géométrique Inverse).

Il convient de noter que la fonction de ces modèles n'est que de calculer des positions discrètes dans l'espace opérationnel ou articulaire. La commande du robot réel demande, quant à elle, une trajectoire continue entre un point de départ et d'arrivée.

La stratégie employée habituellement se décompose comme suit. Tout d'abord on utilise un générateur de trajectoire qui calcule un chemin continu entre des points de contrôle disposés par l'utilisateur dans l'espace de travail. Cette trajectoire doit renseigner sur la position mais aussi sur l'orientation de l'organe terminal du robot. Elle est ensuite discrétisée en un nombre suffisant de points qui sont transformés dans l'espace articulaire par le modèle

géométrique inverse. C'est cet ensemble de configurations articulaires qui peut être traité par l'armoire de commande du robot. Un des problèmes délicats à résoudre est notamment celui de la détermination d'une loi temporelle de déplacement le long de la trajectoire.

Nous avons implanté un tel générateur de trajectoires qui utilise le module de résolution décrit dans le chapitre précédent ainsi qu'un algorithme pour générer un mouvement quasi-optimal en temps le long de la trajectoire (Annexe B).

L'intégration de ces fonctionnalités à l'environnement de programmation de manipulateurs n'a pas été achevée dans le système actuel pour des questions de priorités. Ce module de génération de trajectoires et mouvements sera certainement exploité utilement dans la nouvelle commande du manipulateur dont le développement est en cours.

4.4.2 Approche par décomposition

Le robot est cette fois décomposé comme un ensemble de corps mobiles. C'est la technique que nous utilisons pour modéliser n'importe quel type de mécanisme. Cette stratégie permet de spécifier des mouvements complexes au cours desquels certaines parties du manipulateur doivent satisfaire, le cas échéant, des contraintes de positionnement particulières. Cette possibilité devient intéressante lorsque le manipulateur évolue dans un environnement encombré. Il est alors nécessaire d'utiliser des fonctionnalités d'évitement d'obstacles. Une autre application est la gestion des postures du manipulateur. À la spécification du mouvement proprement dite, on ajoute des contraintes supplémentaires sur les variables articulaires du robot afin qu'elles restent dans des intervalles de valeurs autorisées, évitant en particulier, les commutations brutales de configuration. Sur des manipulateurs redondants cette gestion présente un intérêt encore plus évident car la redondance peut être exploitée pour générer des mouvements internes sans changer la configuration de l'organe terminal.

Contrairement à l'approche globale précédente, la spécification par contraintes permet d'obtenir directement une trajectoire continue vers le but. En effet, la résolution itérative des relations de contraintes fournit des incréments de déplacements à appliquer au robot pour atteindre l'objectif. Le tableau 4.6 résume les principales caractéristiques des deux approches.

	App. par décomposition	**Approche globale**
Gestion des mouvements internes	Possible par ajout de contraintes.	Impossible.
Optimalité en temps de calcul	Non.	Oui.
Prise en compte des butées	Oui, de façon continue au cours du mouvement.	Oui, de façon ponctuelle.
Gestion des configurations et de la redondance	Implicitement guidée par le calcul numérique sans effort supplémentaire.	Explicite. Nécessite l'inventaire de tous les cas possibles.

Tableau 4.6: Approches globale et par décomposition

Toutefois, dans les phases de mise au point, il est commode de pouvoir faire bouger le robot graphique simulé au moyen de son modèle géométrique sans recourir à la méthode de

spécification. La modification du vecteur de configuration articulaire doit alors être répercutée sur la représentation par décomposition. Inversement, le calcul d'un déplacement par la méthode de spécification va modifier la représentation globale du robot. Cette cohérence est assurée de façon transparente pour l'utilisateur.

4.4.3 Mise en œuvre pratique de l'approche globale

La cohabitation des représentations globale et par décomposition améliore l'ergonomie et le confort d'utilisation du simulateur graphique.

Ces besoins de simulation nous ont conduit à hiérarchiser le concept de robot selon le schéma 4.10. En particulier, cette organisation permet de structurer les informations nécessaires à la représentation globale d'un robot (modèles géométriques, fonctions d'interface avec un contrôleur, modalités de déplacement).

À partir d'un noyau générique commun à tout type de robot (entité notée « Robot Général ») peut être développé un robot de type particulier (Robot-T1 sur la figure). Cette famille de robot se subdivise alors en trois branches correspondantes à des utilisations distinctes dans le simulateur : des robots de type graphique (suffixe D), des robots de type physique (suffixe P) à qui sont attachés les routines d'interface avec l'armoire de commande et des robots mixtes dessin/physique (suffixe DP) qui permettent de faire transiter dans les deux sens des informations entre un robot réel et sa représentation virtuelle.

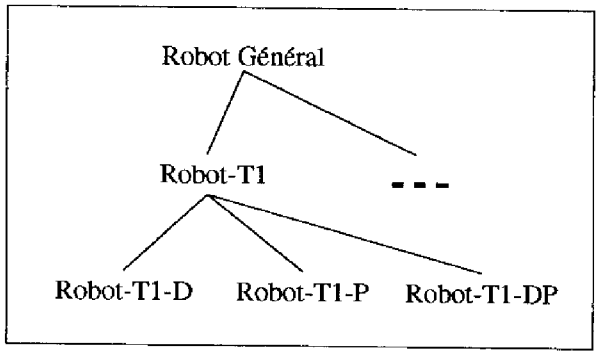

Figure 4.10: Hiérarchie du concept de robot.

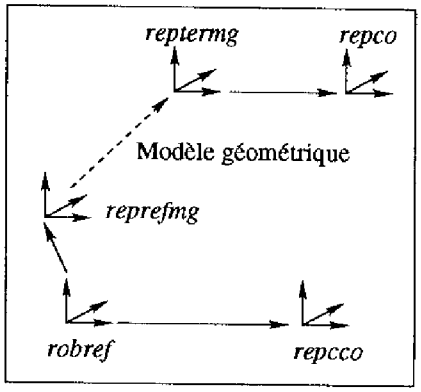

Figure 4.11: Référentiels attachés à un robot.

Repères associés à un robot et modalités de déplacement

Afin de gérer les modalités de déplacement des robots indépendamment de leur type, nous avons adopté une convention de représentation basée sur l'utilisation de référentiels gérés grâce au graphe des repères. Nous associerons à un robot un certain nombre de repères structurés conformément à la figure 4.11 :

- un repère *robref* lié à l'embase du robot,
- deux repères *reprefmg* et *repterm g* définissant respectivement, les repères de référence et terminal du modèle géométrique du robot considéré,

- un repère de contact *repco* constituant le repère de travail du robot (ou repère outil),
- un repère de consigne de contact *repcco*, positionné par rapport à la base du robot qui définit la position objectif que *repco* doit atteindre.

Un robot graphique pourra être déplacé selon deux modalités : un mode de commande en coordonnées articulaires et un mode de déplacement cartésien qui utilise les référentiels définis plus haut. Dans ce dernier cas, les déplacements pourront être définis de manière incrémentale ou bien absolue par rapport aux repères.

Système multi-fenêtres interactif

Il permet de passer en revue tous les robots présents dans l'environnement de travail (manipulateurs et préhenseurs attachés à ces derniers). Parmi les principales fonctionnalités apportées par ces fenêtres arborescentes figurent :

- la visualisation des repères robots,
- l'accès à différentes modalités de positionnement des repères de contact et de consigne de contact (*repco*,*repcco*),
- le déplacement du robot en mode articulaire ou cartésien,
- la mémorisation de configurations-robot particulières,
- et le cas échéant, les fonctions de transfert entre robot graphique simulé et robot réel.

Nous avons vu que les représentations globale et par décomposition pouvaient agir mutuellement l'une sur l'autre. Il est donc indispensable de maintenir automatiquement la cohérence des informations données par les fenêtres interactives de la spécification cinématique d'une part, et les fenêtres de gestion de robots d'autre part.

4.4.4 Modèle d'un robot manipulateur SCEMI et de sa pince

Le manipulateur dont nous disposons au LAAS est un robot SCEMI à six liaisons rotoïdes. Nous l'avons utilisé pour mettre en œuvre une expérimentation que nous décrirons dans le dernier chapitre. Pour une utilisation aisée dans le simulateur, nous avons introduit le modèle global du manipulateur. Nous avons également réalisé la modélisation de la pince installée sur le robot. La modélisation de cette pince intervient dans l'exemple de mouvement coordonné de deux manipulateurs (§ 4.7). Elle ne sera pas utilisée dans l'expérimentation. Les modèles par décomposition du robot et de la pince seront ensuite décrits.

Modèle global

On peut trouver dans la littérature de nombreuses études relatives à l'obtention automatique des modèles analytiques pour des manipulateurs de type série [Gorla 84, Dombre 88]. Les modèles géométriques du manipulateur SCEMI implantés dans le logiciel sont ceux décrits dans [Irigoyen 86]. Le modèle inverse gère en particulier les huits postures possibles pour cette morphologie de manipulateur.

En ce qui concerne la pince, nous nous limitons à son modèle géométrique élémentaire qui, à une valeur donnée de l'écartement des mors, réactualise la représentation graphique de la pince.

Modèle par décomposition

Le robot est représenté en utilisant l'approche par décomposition décrite plus haut (figure 4.12). Chacune des six articulations du manipulateur est modélisée par un couple de demi-contacts positionnés sur les corps de telle sorte que $\vec{z}$ soit confondu avec l'axe de rotation. Les mobilités correspondantes sont donc de la forme FFF FFQ. Les débattements maximaux sur les liaisons sont spécifiés explicitement à l'aide d'une fonction d'initialisation des bornes de variation des variables Q. Dans ces conditions, le mouvement du manipulateur sera interrompu si l'une des butées articulaires est atteinte (fonctionnement du type commande gardée).

À l'utilisation de ce modèle, il sera possible de changer les qualificatifs Q en O pour que les déplacements soient gardés par calcul à l'aide du simplexe. Dans ce cas, les butées articulaires sont considérées comme des obstacles et la procédure de calcul en tient compte pour évaluer le mouvement à produire.

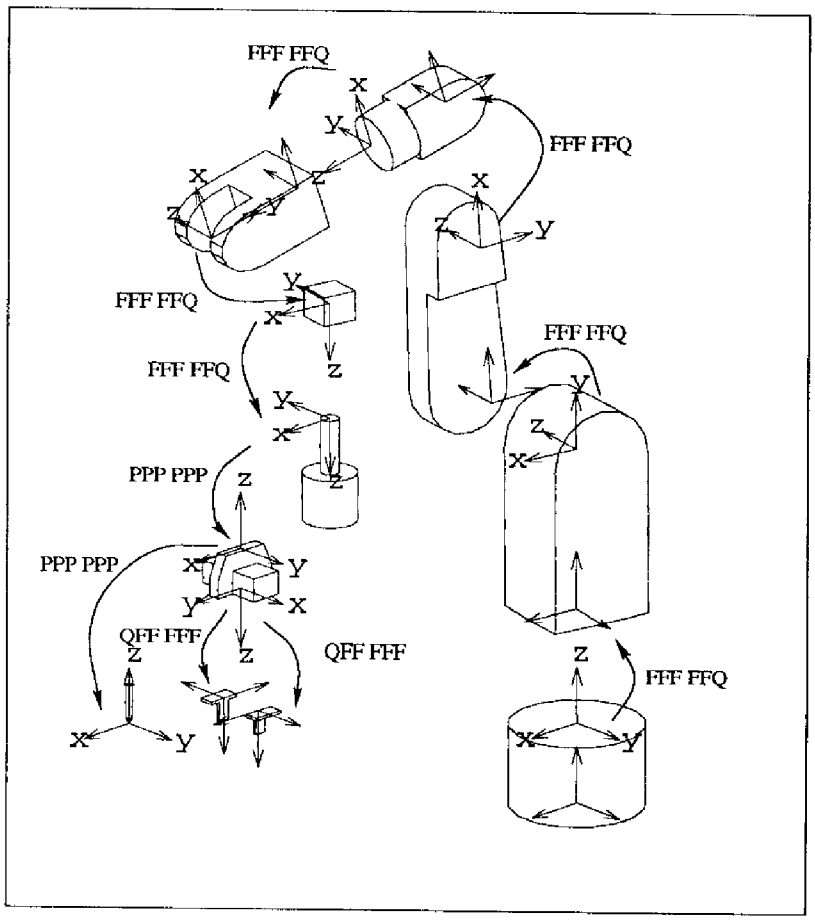

Figure 4.12: Modélisation du robot et de la pince.

La pince à deux mors parallèles est considérée dans notre approche comme un robot à part entière. Les deux articulations de type glissière entre l'embase de la pince et les mors forment une structure arborescente. La mobilité des mors est de la forme QFF FFF (glissière d'axe $\vec{x}$). Le mouvement simultané des deux mors est décrit par l'intermédiaire d'une relation linéaire entre les composantes Q précédentes. Symboliquement, cette relation est de la forme $q_1 = kq_2$ avec k pris égal à un. Cette contrainte se traduit par l'introduction d'une variable d'écart $e = q_1 - kq_2$ étiquetée C. La modélisation d'une pince à trois mors, de type mandrin, se ferait d'une façon analogue. Comme dans le cas du manipulateur, des butées ont été mises en place pour limiter l'ouverture de la pince.

L'embase de la pince est positionnée par rapport au corps terminal du manipulateur grâce à un lien qualifié PPP PPP. Il en est de même de la position de l'outil par rapport à la pince. Comme nous l'avons déjà mentionné, la pince n'étant pas utilisée dans l'expérimentation, les articulations relatives aux mors seront «désactivées» dans les exemples suivants par l'utilisation de mobilités du type PFF FFF.

4.5 Mouvement avec évitement local d'obstacles

L'outil que nous avons développé permet, sous certaines conditions que nous allons discuter, d'inclure l'évitement d'obstacles dans la spécification de tâches.

Pour un système robotique donné, la notion d'obstacle peut avoir deux origines. Tout d'abord, une source externe d'obstacles est formée par les objets situés dans l'espace de travail du robot que celui-ci doit éviter au cours de ses déplacements. Plus marginalement, la deuxième catégorie d'obstacles provient du robot lui-même. Ce type d'obstacles, que nous pourrions qualifier d'endogènes, est présent dans la plupart des mécanismes polyarticulés. Dans un manipulateur série, ils correspondent souvent aux butées mécaniques des liaisons. Avec des systèmes de type main articulée, ils représentent les possibilités de collision entre les différents doigts.

La manière la plus naturelle d'exprimer une condition de non collision entre deux objets consiste à raisonner sur la distance qui les sépare. L'expression des distances entre objets dans l'espace de la tâche associé au traitement de contraintes inégalités, introduit dans la méthode de résolution, vont nous permettre de spécifier des objectifs d'évitement d'obstacles.

L'évitement d'obstacles proposé est donc local et suppose une action volontaire de l'utilisateur pour la mise en place des contraintes. L'installation automatique des contraintes d'évitement d'obstacles sur la base d'une analyse des proximités entre les corps constitue une voie de développement à envisager.

Nous allons illustrer au moyen de deux exemples des cas concrets de spécification de contraintes inégalités qui conduisent à la génération de trajectoires sans collision.

4.5.1 Exemple 1 : insertion de deux pièces

La tâche que nous désirons accomplir est l'insertion d'une pièce en forme d'équerre dans un logement de forme complémentaire. La figure 4.13a représente les positions initiale et finale de l'équerre mobile. L'objectif de la tâche, c'est à dire la position terminale souhaitée, est spécifié au moyen de deux relations de contact Obj1 et Obj2 décrites dans les tableaux suivants et la figure 4.13b.

Contact:		**Obj1**		$\mathbf{A_1 \rightarrow B_1}$				Contact:		**Obj2**		$\mathbf{A_2 \rightarrow B_2}$			
Qualificatif:		WRR RRW						Qualificatif:		RWR RRR					
Consigne:	-1	0	0	0	0	1	0	Consigne:	0	0.3	0	0	0	1	0
État:	50	-68.5	5	0	0	1	-90	État:	10	80	5	0	0	1	90
Erreur:	51	0	0	0	0	1	-90	Erreur:	0	79.7	0	0	0	1	0

La trajectoire obtenue avec simplement ces deux contraintes engendre des collisions entre les pièces (figure 4.13c). Pour les éviter, nous rajoutons plusieurs contraintes de type inégalité qui expriment des conditions de non interférence.

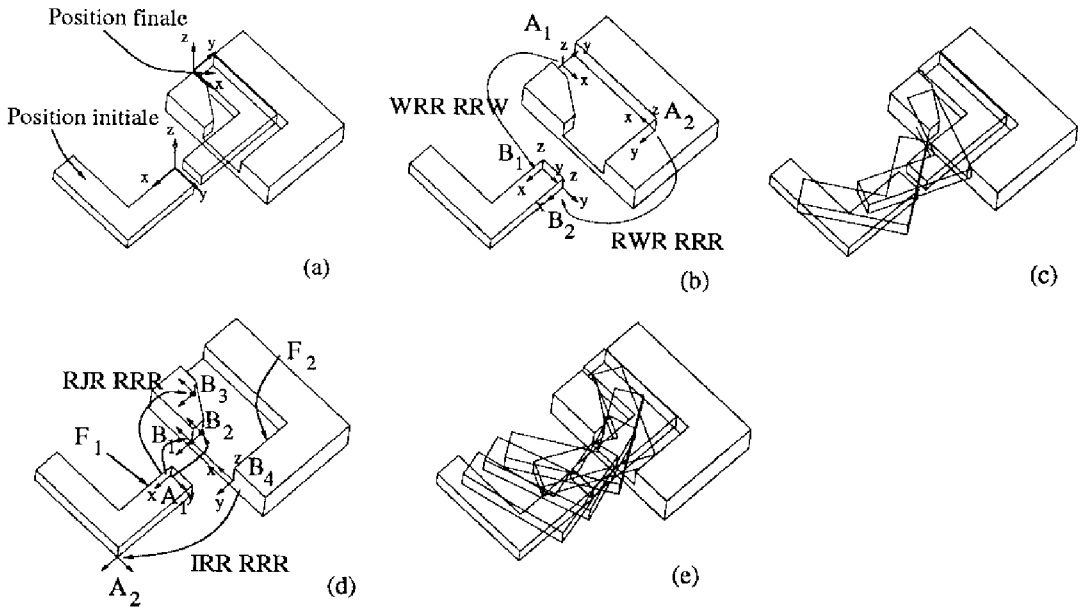

Figure 4.13: Tâche d'insertion d'une équerre.

Trois premières relations de contact sont établies entre A_1 et les trois référentiels B_1, B_2 et B_3 avec des qualificatifs de la forme RJR RRR (figure 4.13d). Ces contacts expriment que les origines de B_1, B_2 et B_3 doivent rester à l'extérieur de la face F_1 de normale $A_1(\vec{y})$. Ces contraintes se traduisent par la négativité (étiquette J) de la composante des liens de contact selon $\vec{y}$. Les valeurs des autres composantes sont indifférentes (étiquettes R).

Une autre relation de contact entre B_4 et A_2, dotée du qualificatif IRR RRR, exprime que l'origine de A_2 demeure à une abscisse positive selon $B_4(\vec{x})$, normale de la face F_2.

L'introduction de ces contraintes inégalités entraîne l'activation du simplexe dans la méthode de résolution.

La trajectoire résultante est représentée sur la figure 4.13e. Cet exemple montre qu'une spécification relativement simple conduit à une trajectoire satisfaisante sur le plan pratique, sans problèmes de convergence.

4.5.2 Exemple 2 : gestion des butées sur un manipulateur redondant

Le robot envisagé dans cet exemple est le manipulateur redondant Mitsubishi PA-10. Il comporte sept liaisons rotoïdes disposées en série (figure 4.14).

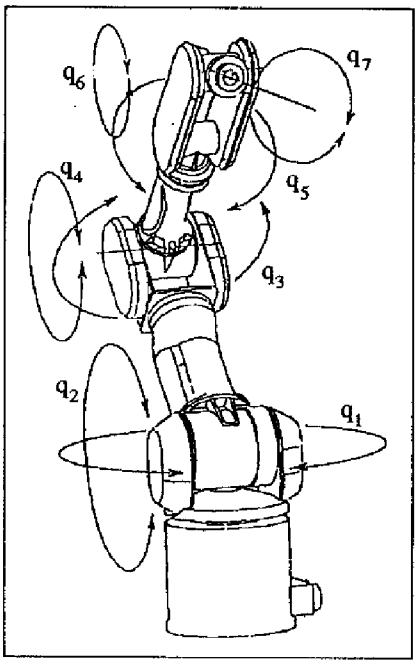

Figure 4.14: Robot Mitsubishi PA–10.

Un modèle graphique 3D du manipulateur a été construit. Bien que les topologies exactes des corps intermédiaires ne soient pas respectées dans ce modèle graphique, le positionnement des axes des liaisons a été réalisé conformément aux valeurs des paramètres géométriques du robot réel. Les butées articulaires de chaque liaison sont spécifiées par le procédé évoqué plus haut, lors de la modélisation du robot SCEMI.

La tâche à faire réaliser par le robot consiste à positionner l'outil tenu par le robot par rapport à un référentiel situé à proximité de l'embase du manipulateur (figure 4.15a). L'objectif du déplacement se traduit par une relation de contact entre ces deux repères, qualifiée VVV RRR : position confondue et orientation indifférente.

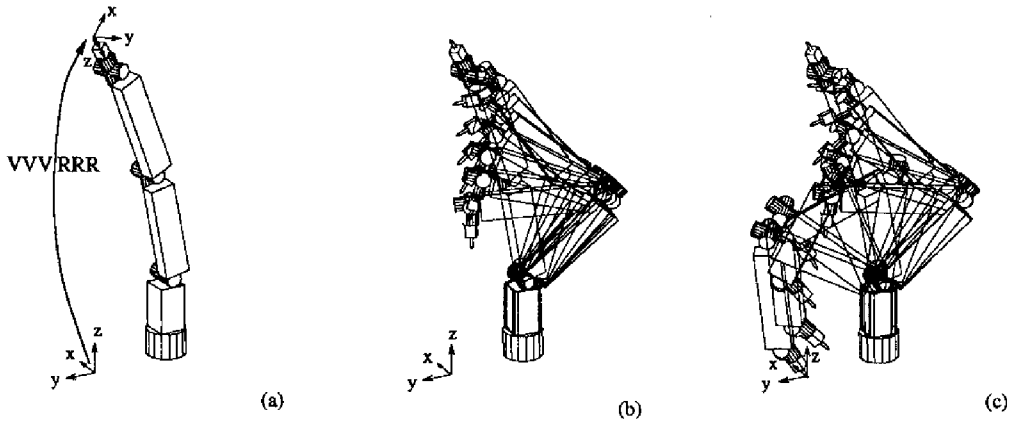

Figure 4.15: Gestion des butées sur un manipulateur à sept degrés de liberté.

Nous présentons deux situations de fonctionnement : pour la première, les mouvement

ont été simplement gardés (qualificatifs des articulation de la forme FFF FFQ). Comme indiqué sur la figure 4.15b, le manipulateur doit s'arrêter avant d'avoir rallié le but car une de ses articulations est en butée (il s'agit de la coordonnée articulaire q_4).

Si maintenant nous qualifions les articulations FFF FFO, les butées mécaniques sont gérées par le simplexe qui parvient, par une déformation locale de la trajectoire nominale, à trouver une solution vers le but (figure 4.15c).

4.6 Mouvement au contact d'un manipulateur vers un cylindre

Dans cette partie, nous allons traiter un exemple mettant en œuvre un manipulateur qui a pour tâche de se déplacer au contact d'un cylindre. Le manipulateur considéré est un robot SCEMI à six degrés de liberté équipé d'une pince à mors parallèles. Cette dernière porte un outil qui permet au manipulateur d'établir des contacts avec les objets de l'environnement.

Modélisation du robot et de l'environnement de travail

Le robot comprenant le manipulateur et la pince est décrit au moyen de ses modèles global et par décomposition évoqués au paragraphe 4.4.4.

L'environnement de travail est constitué d'une table et d'un cylindre posé sur cette table. L'embase du robot se trouve également positionnée par rapport à la table. Les qualificatifs de mobilité sont détaillés sur la figure 4.16. On suppose dans cette partie que les positions du cylindre et de l'embase du robot par rapport à la table sont connues.

Expression des contraintes

On souhaite déplacer le manipulateur pour que son outil vienne en contact avec le cylindre. Cette contrainte s'exprime au moyen d'un contact de type repère sur cylindre entre les demi-contacts dc00 et dc10 désignant respectivement la référence du cylindre et le repère extrémité de l'outil (voir figure 4.17). La forme cylindre attachée à dc00 définit deux demi-contacts mobiles dc01 et dc02. L'axe $\vec{z}$ de dc02 définit la normale d'un plan tangent au cylindre. Ce référentiel tangent est positionné par rapport à dc01 par une translation selon $\vec{z}$ égale au rayon du cylindre et une rotation d'axe $\vec{z}$ (ZZP ZZG). dc01 est mobile en translation et rotation selon l'axe $\vec{x}$ de dc00 (GZZ GZZ).

La mise en contact de l'outil avec le cylindre peut alors être spécifiée par les qualificatifs DDW RRR : les contraintes D expriment une condition géométrique à satisfaire impérativement au moyen des variables inconnues étiquetées G. Comme dans le cas de la forme droite, l'évaluation de ces inconnues peut être réalisé par un simple calcul géométrique : un choix par défaut raisonnable consiste à calculer les inconnues G de sorte que le centre de dc02 soit la projection du centre de dc10 sur le cylindre. En raison de la symétrie de révolution, deux points peuvent convenir. Nous choisirons le point en regard de la pointe de l'outil. La contrainte W représente la distance d'approche du cylindre choisie par l'utilisateur ou bien imposée par les conditions opératoires. Cette situation se produit si l'on souhaite interposer une cale d'épaisseur entre l'outil et le cylindre. Dans cet exemple, l'orientation de dc10 est rendue indifférente par emploi des étiquettes R.

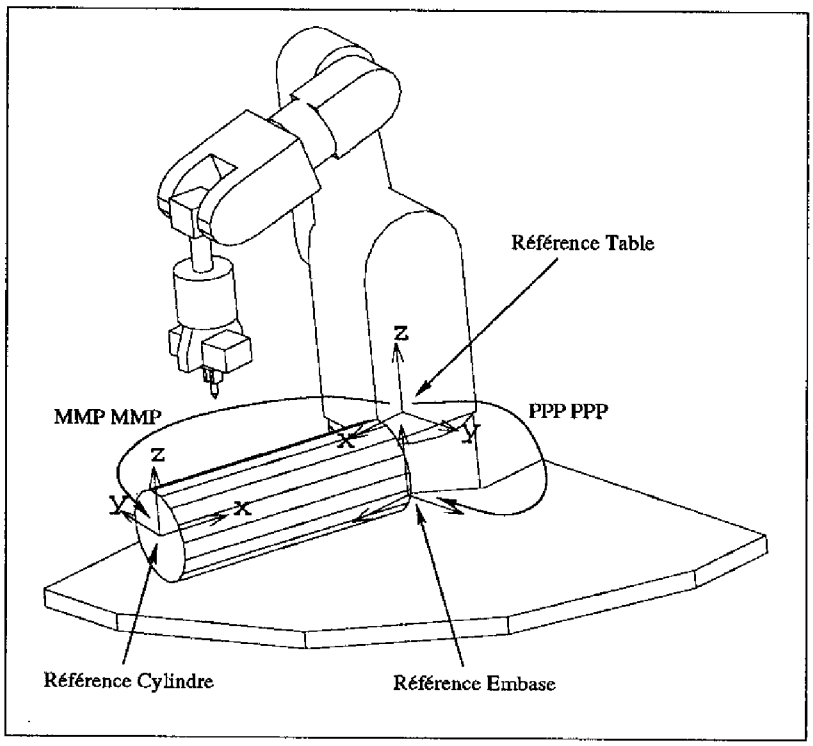

Figure 4.16: Modélisation de l'environnement de travail.

Partant d'une configuration initiale donnée, le solveur résoud les inconnues articulaires X pour satisfaire W, seule contrainte activée dans cet exemple. La solution mise en œuvre par le système sera, de toute évidence, du type norme minimale. Concrètement, le robot va converger vers une configuration solution qui aura minimisé l'amplitude des rotations articulaires (voir figure 4.19).

Mais l'utilisateur peut souhaiter faire évoluer le robot vers un point particulier du cylindre. Pour cela le schéma de spécification précédent peut être immédiatement adapté en introduisant un demi-contact supplémentaire dc2 qui va servir à désigner le point particulier à atteindre grâce au lien paramétrique PNN PNN (voir figure 4.18). Les contraintes de positionnement du repère intermédiaire dc01 sont de la forme VRR VRR. Le mouvement continu généré est représenté sur la figure 4.20.

Performance du système

Le tableau 4.7 résume les principales informations relatives aux temps de calcul mesurés sur une station de travail Sun SparcStation-10. Ce comparatif de vitesse a été réalisé pour la résolution de quelques uns des problèmes de mouvement décrits jusqu'à présent.

Les problèmes de mouvement utilisés pour cette étude de performance sont :

- Pb_1: Positionnement de deux objets cubiques (paragraphe 4.2).
  $Err_1 =$ 0 0 -10 0 0 1 -35
- Pb_2: Insertion d'une équerre (paragraphe 4.5.1).
  $Err_2 =$ 51 0 0 0 0 1 -90

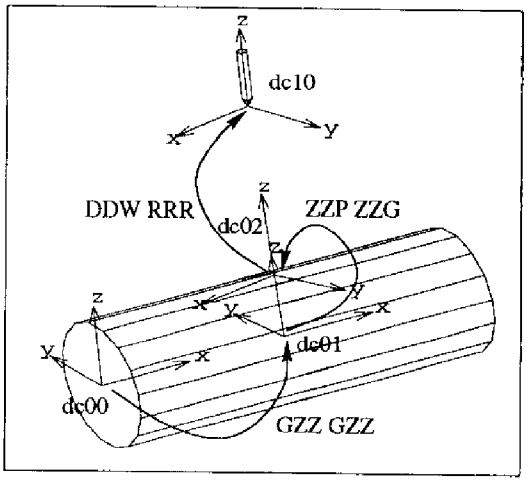

Figure 4.17: Contrainte de contact avec le cylindre.

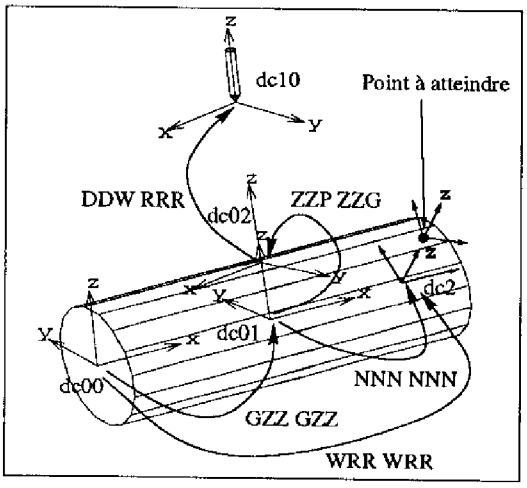

Figure 4.18: Rejoindre un point donné du cylindre.

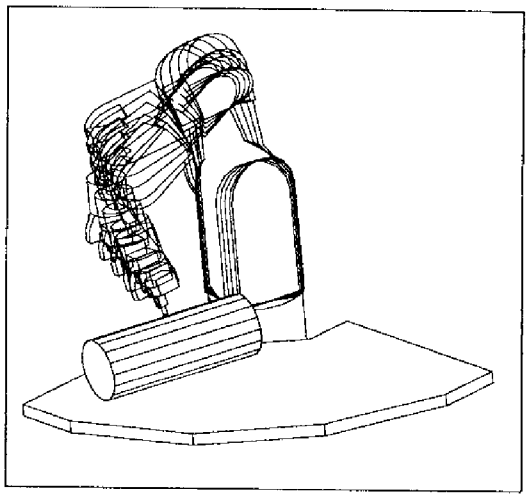

Figure 4.19: Mouvement vers le point le plus proche.

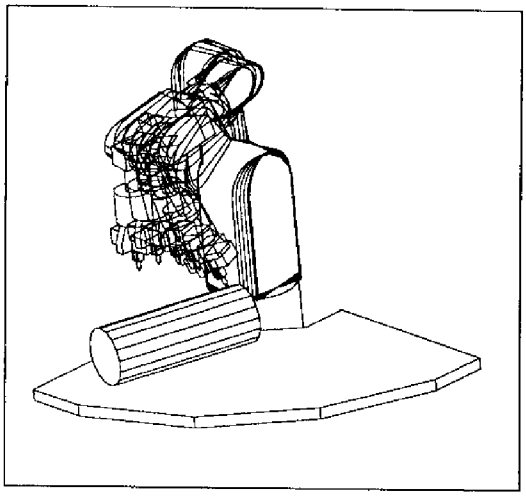

Figure 4.20: Mouvement vers un point spécifié.

- Pb_3: Tâche de déplacement du manipulateur PA–10 (paragraphe 4.5.2).
  $Err_3 =$ -296.75 151.52 1524.61 0 0 1 0

- Pb_4: Mouvement au contact du robot SCEMI sur un cylindre (paragraphe 4.6).
  $Err_4 =$ 43.91 163.88 -111.44 0 0 1 0

Pour pouvoir confronter, de manière qualitative, les résultats obtenus, nous nous sommes basés sur quelques uns des paramètres reflétant la « complexité » des différents problèmes. Les facteurs retenus et présentés dans le tableau sont :

- le nombre total de variables, avec le décompte du nombre d'inconnues et de contraintes spécifiées dans le problème,

	Pb_1	Pb_2	Pb_3	Pb_4
Variables	60	102	132	162
Inconnues	10	3	7	9
Contraintes	6	7	3	5
Cycles	3	6	1	2
Erreur initiale	Err_1	Err_2	Err_3	Err_4
Itérations	17	31	152	23
Recompilations	4	15	13	4
Durée totale (s)	1.439	8.541	8.456	1.763
Durée moyenne d'une itération (s)	0.085	0.276	0.056	0.077
Taux de recompilation	0.24	0.48	0.09	0.17

Tableau 4.7: Temps de calcul.

- le nombre de cycles cinématiques mis en jeu.

Les incréments de déplacement étant bornés (10 mm pour les translations et 0.1 rad pour les rotations), le nombre d'itération requis pour résoudre un problème dépend de l'amplitude de l'erreur initiale. C'est pourquoi nous l'avons mentionnée sous la forme de vecteur de configuration.

Les durées indiquées correspondent au temps de calcul des incréments de déplacement. Plus précisément, ce que nous entendons par calcul comprend la génération des équations de contraintes à partir de la spécification de tâches et leur résolution.

De façon prévisible, nous pouvons observer que le taux de recompilation et le nombre de cycles influent sur la durée moyenne d'une itération. Grâce aux mécanismes d'élimination de variables mis en œuvre dans la méthode de résolution, les performances globales ne sont pas directement proportionnelles au nombre total de variables.

4.7 Mouvement coordonné de deux manipulateurs

Dans cette partie, nous illustrons l'utilisation de notre méthode de spécification pour la résolution d'une tâche d'insertion exécutée par deux robots dont les mouvements doivent être coordonnés.

Les robots utilisés dans cette simulation sont des manipulateurs SCEMI équipés de leur pince. Il s'agit de saisir une boîte munie de poignées, de la déplacer au dessus d'un réceptacle, puis de réaliser l'insertion dans ce logement suivant une stratégie adaptée à une commande par retour d'efforts. (figure 4.21-a). La spécification de cette tâche se décompose naturellement en trois étapes : la saisie des poignées, le soulèvement suivi du transfert de la charge et l'insertion proprement dite.

Prise de la charge par les poignées

Le mouvement à effectuer consiste à mettre les mors de chaque pince en contact avec les poignées, en leur milieu tout en gardant les faces de prise dans un plan vertical. Le nombre de contraintes étant égal au nombre de degrés de liberté des robots, les manipulateurs ne présentent pas de redondance pour cette tâche. Les contraintes de saisie des poignées sont de type V ou W (objectifs à résoudre en convergence lente, puisque les robots sont initialement éloignés des poignées).

La figure 4.21-b met en évidence le caractère tangent des mouvements accomplis par les deux manipulateurs pour venir au contact des poignées (figure 4.21-c). Cette propriété intrinsèque de la résolution permet donc la synthèse de mouvements fins et découle des propriétés de convergence quadratique de la méthode de Newton-Raphson.

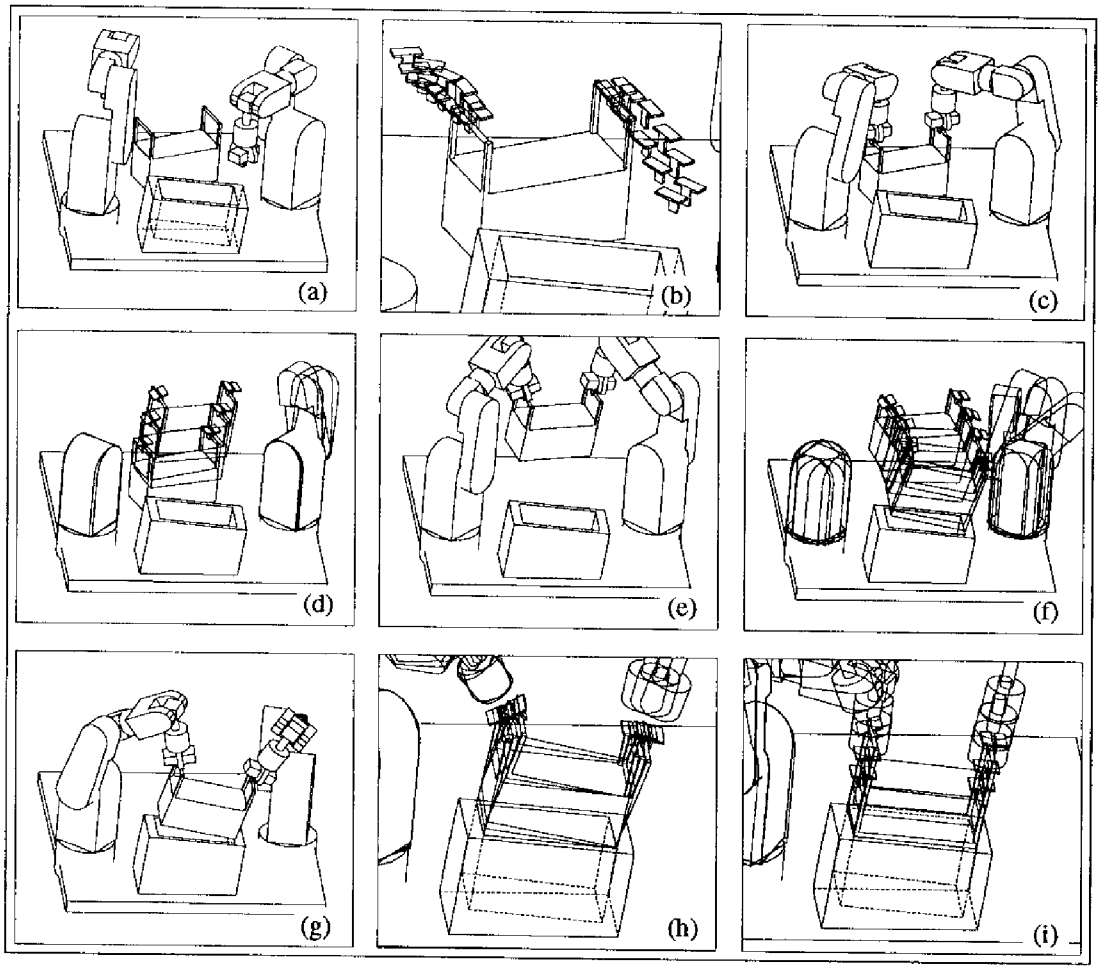

Figure 4.21: Scénario complet d'une tâche d'insertion réalisée par deux manipulateurs.

Transfert de la charge

Le déplacement des robots doit être spécifié de sorte que les mors demeurent en contact avec les poignées : ce seront des contraintes de type C à résoudre en priorité (ce choix de résolution

est consistant puisque les mors ont préalablement été mis au contact des poignées). Pour ne pas contraindre inutilement le mouvement, on autorise une rotation des prises autour des poignées, dans un certain intervalle. Cette latitude est définie par des contraintes de type inégalité.

Après avoir soulevé la charge, les manipulateurs déplacent la boîte parallèlement à un plan incliné fictif qui s'appuie sur une arête du réceptacle (figures 4.21-d, e, f).

Contrairement à la phase précédente, le problème est cette fois redondant puisque les deux robots (2×6 degrés de liberté) assurent le déplacement de la boîte. Cette redondance du problème est résolue en minimisant les incréments de déplacement de la charge et des manipulateurs. Elle permet également de satisfaire les contraintes inégalités introduites au niveau des préhenseurs et sur les articulations de chaque robot.

Insertion

À partir de la position de présentation au dessus du réceptacle, l'engagement successif des sommets de la boîte est spécifié. L'insertion se termine par la translation de la charge dans le logement (figures 4.21-g, h,i).

Comme nous venons de le voir, trois problèmes de mouvements de base ont été programmés pour décrire ce scénario de manipulation coopératif. De simples modifications d'objectifs dans ces problèmes nous ont permis d'obtenir la totalité des trajectoires requises pour la simulation réaliste de cette tâche.

4.8 Problèmes de calibration

Les spécifications de tâches que nous avons présentées jusqu'à présent reposaient sur l'hypothèse que les modèles de travail étaient conformes à la réalité. Le passage de la simulation de tâches à leur exécution dans le monde réel met en évidence la question fondamentale de la validité du modèle. La résolution de ce problème nécessite la mise en place de stratégies de calibration.

Le plus souvent, les questions touchant à la calibration sont traitées spécifiquement par des procédures et des dispositifs dédiés. Un exemple commun dans le domaine de la vision consiste à utiliser des appareils de mesure tels que télémètres et théodolites qui permettent d'évaluer des positions d'objets situés dans une scène.

Un système commercialisé par la société Northern Digital [NOR 91] est basé sur la mesure et le traitement en temps réel des signaux émis par un certain nombre de diodes infrarouges installées sur les objets dont il faut identifier la position.

Le caractère commun à ces techniques de calibration est d'appréhender le problème par une approche de type métrologique qui demande la mise en place préalable de dispositifs très particuliers tels que capteurs ou mires de référence. Ces impératifs opératoires peuvent s'avérer impraticables, en particulier lorsque le site de travail n'est pas accessible. Dans ces situations, les moyens de calibration doivent être mieux intégrés à l'exécution des tâches.

L'approche proposée permet de traiter par la méthode de spécification de tâches, des problèmes de calibration. Nous avons vu au paragraphe 2.6 que la définition d'un problème de calibration va s'appuyer sur la ou les tâches de mouvement qui lui sont associées.

4.8.1 Obtention de mesures

La génération de mesures peut se faire de deux manières. La première correspond au cas où l'on utilise un manipulateur réel, la seconde relève de la simulation sur le modèle virtuel.

Cas de l'expérimentation

Le ou les objets à calibrer existent physiquement dans l'environnement de travail d'un manipulateur réel. Le principe de base pour identifier la position et/ou la taille de ces corps consiste à établir, entre eux et le robot, différents contacts. Dans ce contexte, le robot est utilisé comme une machine à mesurer.

Si les informations relatives au robot et son espace de travail sont accessibles à l'utilisateur sans limitations, par exemple grâce à un retour visuel, une stratégie possible est de commander manuellement le manipulateur pour sa mise en contact avec les objets. Cette situation correspond à un contexte de téléopération. Si ces conditions opératoires ne sont pas réunies (par exemple, lorsque le site robotisé est distant), le fonctionnement du robot en commande gardée pourra se substituer à l'action de l'opérateur.

Une fois le contact réalisé par l'une ou l'autre des stratégies, les valeurs des codeurs angulaires du robot sont lues et donnent lieu à une mesure.

Nous reviendrons plus en détail sur les modalités pratiques d'acquisition de mesures dans le prochain chapitre relatif à l'expérimentation.

Utilisation de la simulation

La génération de mesures simulées ne pose pas de problème. Si l'objectif est toujours de réaliser des situations de contact entre le robot et les objets à calibrer, on raisonne cette fois sur le modèle de l'environnement.

Le problème de mouvement relatif au contact désiré est exploité pour déplacer le robot graphique. Pour simuler l'écart existant entre la réalité et le modèle, une consigne de contact non nulle est spécifiée. L'état du système lorsqu'il a atteint ce contact permet d'engendrer une mesure.

L'itération de ce procédé conduit à un ensemble de mesures à traiter dans le problème de calibration. Nous retrouvons alors exactement la méthodologie exposée en 2.6.2 et résumée par la figure 2.18.

En guise d'illustration, nous présentons la calibration de la position d'un cylindre. Cet exemple, exposé ici dans sa version de simulation, servira de base à la mise en œuvre pratique de l'expérimentation sur manipulateur réel présentée dans le prochain chapitre.

4.8.2 Exemple : calibration d'un cylindre par contact

Revenons à la spécification du mouvement d'un manipulateur SCEMI au contact d'un cylindre, présentée plus haut. La spécification proposée permet d'envisager la calibration du lien articulaire entre la table et le cylindre grâce aux qualificatifs MMP MMP ($arti_1$ sur la figure 4.22). Les inconnues correspondantes sont les six articulations du robot FFF FFX. Les mors de la pince ont été bloqués par des qualificatifs du type PFF FFF. Décrivons les étapes de la construction de ce problème de calibration.

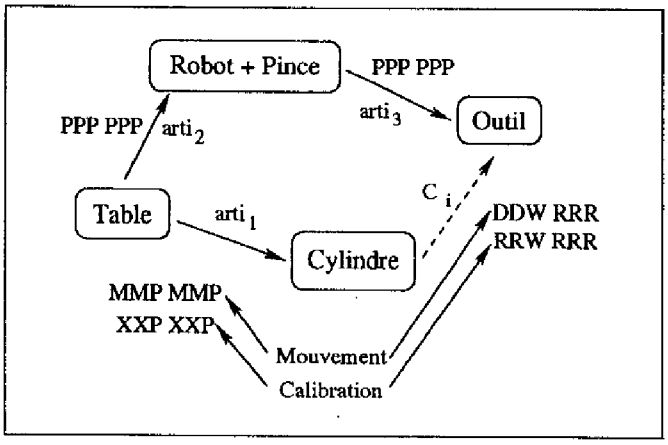

Figure 4.22: Les liaisons sujettes à calibration.

La synthèse de cinq mesures est réalisée selon les modalités indiquées dans le tableau 4.8. Ces mises en contact de l'outil et du cylindre sont représentées sur la figure 4.23.

	Position sur le cylindre[2]	Consigne du contact simulé
Contact 1	50 0 0 1 0 0 -45	0 0 5 0 0 0 1 0
Contact 2	300 0 0 1 0 0 -30	0 0 -6 0 0 0 1 0
Contact 3	350 0 0 1 0 0 40	0 0 -7 0 0 0 1 0
Contact 4	100 0 0 1 0 0 30	0 0 2 0 0 0 1 0
Contact 5	220 0 0 1 0 0 -90	0 0 5 0 0 0 1 0
Position initiale du cylindre	650 130 75 0 0 1 165	

Tableau 4.8: Génération des mesures.

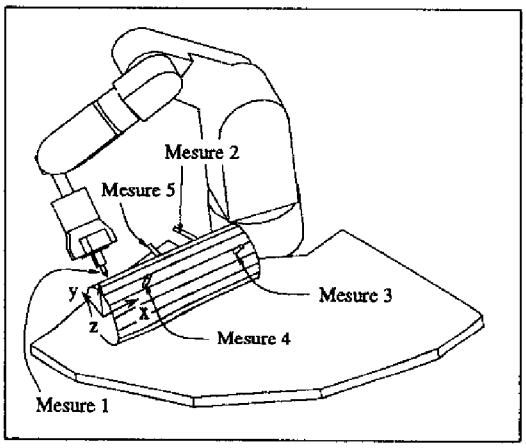

Figure 4.23: Ensemble de mesures.

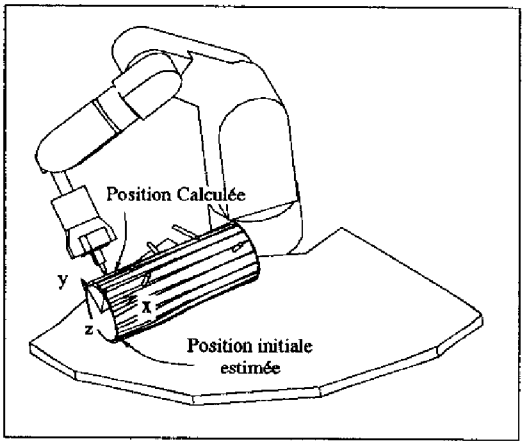

Figure 4.24: Résultat de la calibration.

Le problème d'optimisation à résoudre consiste à calculer le lien dual de $arti_1$ XXP XXP

pour satisfaire au mieux, avec chaque mesure, les contraintes de contact C_i entre l'outil et le cylindre. Pour autoriser une latitude de mouvement du cylindre par rapport à la table, les relations de contact C_i à vérifier ne sont plus DDW RRR mais RRW RRR. En effet, la seule contrainte pertinente à optimiser dans la calibration est la distance de l'extrémité de l'outil au cylindre représentée par la variable W.

	État du contact après calibration
Contact 1	0 0 -0.683 0 0 0 1 0
Contact 2	0 0 -1.571 0 0 0 1 0
Contact 3	0 0 1.247 0 0 0 1 0
Contact 4	0 0 0.960 0 0 0 1 0
Contact 5	0 0 2.456 0 0 0 1 0
Position du cylindre	650 126.974 82.504 0 0.175 0.985 164.7

Tableau 4.9: Résultat de la calibration du cylindre.

Variables	792
Inconnues	4
Contraintes	5
Cycles	5
Itérations	9
Recompilations	4
Durée totale (s)	7.448
Durée moyenne d'une itération (s)	0.828
Taux de recompilation	0.44

Tableau 4.10: Temps de calcul.

Le résultat fourni par le module de résolution définit une nouvelle position pour le cylindre (figure 4.24). Cette nouvelle position donne lieu aux écarts de contact indiqués dans le tableau 4.9.

Un problème de calibration est toujours plus volumineux en nombre de variables à traiter que son problème de mouvement associé. L'augmentation du nombre de variables est quasiment proportionnel au nombre de mesures utilisées puisque chacune d'elle est à l'origine d'un nouveau cycle cinématique.

Cette remarque souligne encore une fois la nécessité de disposer d'un outil numérique susceptible de maîtriser cette croissance du nombre de variables.

Les temps de calcul mesuré sont présentés dans le tableau 4.10. On peut constater que les performances ne se dégrade pas fortement en dépit du nombre de variables (792).

Remarques

- En réalité, cet exemple fait apparaître plusieurs problèmes de calibration. En particulier, les liens relatifs aux articulations $arti_2$ et $arti_3$ (figure 4.22) sont en général connus avec une grande imprécision. Comme nous le verrons dans le prochain chapitre, l'identification de ces liaisons peut être réalisée de façon analogue.
- La stratégie de calibration présentée permet également d'identifier le rayon du cylindre. Pour cela, il suffit de changer, dans la spécification du mouvement, le lien ZZP ZZG en ZZM ZZG. De cette manière, la composante selon $\vec{z}$ définissant le rayon du cylindre devient inconnue dans le problème de calibration.

[2] Il s'agit de la consigne relative au lien de contact WRR WRR défini sur la figure 4.18

4.8.3 Étude de sensibilité par rapport à la réalisation des contacts

Au cours du processus de résolution, l'algorithme de calcul établit les relations matricielles qui relient les incréments de variables inconnues aux incréments des variables de contraintes. À l'issue de la résolution, nous disposons d'une relation matricielle de la forme :

$$\Delta X^* = M \Delta W^* \tag{4.1}$$

telle que $\Delta X^* = 0$; cet état définit un critère d'arrêt de la méthode numérique itérative.

En explicitant la nature des composantes des vecteurs incrémentaux, le système linéaire précédent s'écrit :

$$\begin{array}{c} \text{TY} \\ \text{TZ} \\ \text{RY} \\ \text{RZ} \end{array} \begin{bmatrix} x_1^* \\ x_2^* \\ x_3^* \\ x_4^* \end{bmatrix} = \begin{bmatrix} M_{trans} \\ M_{rot} \end{bmatrix} \begin{bmatrix} w_1^* \\ w_2^* \\ w_3^* \\ w_4^* \\ w_5^* \end{bmatrix} \begin{array}{l} \text{Contact 1} \\ \text{Contact 2} \\ \text{Contact 3} \\ \text{Contact 4} \\ \text{Contact 5} \end{array} \tag{4.2}$$

La valeur nominale, pour chaque mesure, de l'écart (en mm) entre le palpeur et le cylindre est notée w_i^* avec $i \in [1, 5]$. Les écarts w_i^* correspondent aux valeurs des composantes selon $\vec{z}$ des vecteurs de configuration donnés dans le tableau 4.9. Les variables x_1^*, x_2^* d'une part et x_3^*, x_4^* représentent respectivement les composantes de translation et de rotation du lien qui positionne le cylindre.

La matrice de sensibilité obtenue avec l'exemple proposé vaut :

$$M = \begin{bmatrix} 0.99111 & -0.30222 & 0.43831 & -0.91676 & 0.21805 \\ -0.70477 & 0.08777 & 0.28935 & -0.98508 & 0.09846 \\ -0.01834 & 0.01905 & 0.03235 & -0.03622 & 0.00482 \\ 0.00738 & -0.00657 & -0.00301 & 0.00391 & -0.00216 \end{bmatrix} \tag{4.3}$$

Si l'on suppose que les mesures proposée sont connues avec une marge d'erreur sur la réalisation des contacts égale à plus ou moins un millimètre, la relation (4.2) permet d'évaluer l'erreur produite sur le placement du cylindre par rapport à sa position nominale x_i^*.

Une erreur de un millimètre sur les contacts, mesurée dans le sens positif, génère des erreurs Err_x sur les variables de positionnement du cylindre (tableau 4.11).

		Err_x
x_1^*	126.97 mm	0.43 mm
x_2^*	82.50 mm	-1.21 mm
x_3^*	28.82°	0.10°
x_4^*	162.23°	-0.02°

Tableau 4.11: Analyse de sensibilité sur le positionnement du cylindre.

Bien que nous en ayons donné le principe au paragraphe 3.8.2, l'étude de sensibilité par rapport à d'autres sources potentielles d'erreurs n'a pas été développée. Sur le plan pratique, il peut être pertinent d'examiner certaines imprécisions du modèle (l'entr'axe de deux liaisons du robot), ou encore l'effet des incertitudes sur les valeurs données par les codeurs angulaires du robot.

4.9 Modélisation de capteurs

Dans ce paragraphe nous présentons une application de notre approche de spécification par contraintes visant à intégrer des informations provenant de capteurs (ce type de modélisation illustre le niveau 2 d'exploitation du logiciel).

La nature des capteurs plus particulièrement envisagés sont des capteurs dont la sortie peut être interprétée, de façon plus ou moins directe, comme une mesure de distance. Cette caractéristique couvre une très grande variété de capteurs utilisés en robotique.

Certains capteurs « spécialisés », comme les télémètres à laser ou proximètres à infrarouge, fournissent directement une information de distance. Inversement, d'autres capteurs pour lesquels la densité d'information est plus grande, demandent des traitements supplémentaires pour accéder à l'information de distance. Les caméras vidéos appartiennent à cette catégorie. Nous présentons ci-après l'intégration des informations d'une caméra vidéo dans notre système de spécification de tâches.

4.9.1 Modélisation d'un capteur de type caméra

Les images provenant d'une caméra vidéo représentent une source importante d'information qu'il serait souhaitable d'exploiter pour permettre à des systèmes robotiques d'évoluer dans des environnements mal connus. Pour tenter de résoudre cette question, il est nécessaire de structurer ces informations sous une forme symbolique qui servira de support au raisonnement en vue de la reconnaissance ou bien la localisation des objets perçus. Ces préoccupations constituent quelques uns des thèmes abordés en vision artificielle.

Notre objectif dans cette partie est d'envisager l'application de notre méthode de spécification par contraintes pour résoudre certains problèmes spécifiques rencontrés en traitement d'images comme l'exécution de tâches référencées vision ou bien la calibration d'objets.

Hypothèses de travail - Modèle

Le modèle le plus couramment utilisé en vision est le modèle du sténopé [FAUGERAS 88]. Il définit la caméra par son axe optique orthogonal au plan de l'image et son centre optique C situé à une distance f (distance focale) du plan image (figure 4.25).

Ce modèle applique à chaque point perçu une projection sur le plan image, de centre C, appelée projection perspective. Dans le cas d'une caméra vidéo, le plan image se matérialise par une matrice CCD. Les coordonnées d'un point m image de M, issues du capteur CCD, doivent être corrigées par des facteurs d'échelle.

Une représentation plus conforme à la réalité devrait en outre tenir compte de certains défauts du système optique tels que les aberrations ou les distorsions. Le plus souvent, ces imperfections sont inclues dans un modèle non linéaire qui complète la description par projection perspective. Nous ne traiterons pas ces problèmes dans notre approche.

Nous nous plaçons dans le cadre des hypothèses de travail suivantes :

- les cœfficients d'échelle de la matrice CCD sont connus,
- les données provenant du traitement d'image sont disponibles sous la forme de segments 2D,

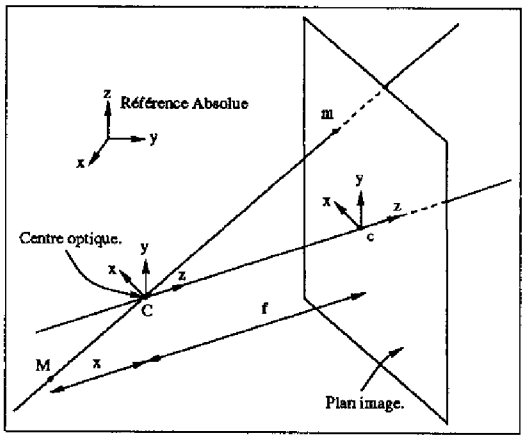

Figure 4.25: Modèle du sténopé.

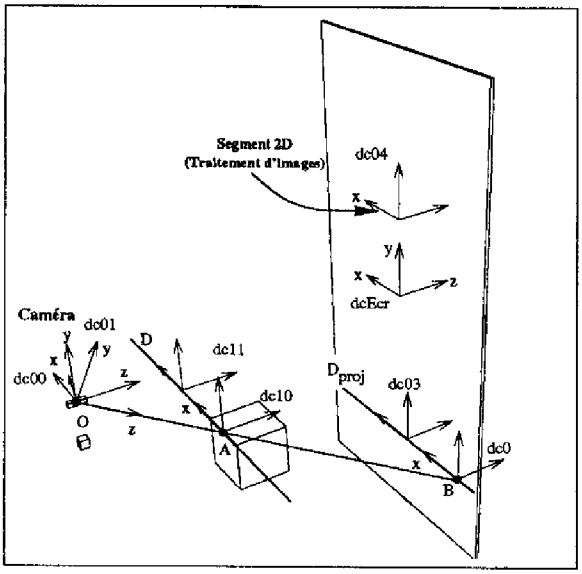

Figure 4.26: Modélisation d'une caméra par contraintes.

- l'appariement de ces segments avec les arêtes des corps du modèle 3D est réalisé.

Description de la modélisation par contraintes

Pour inclure les informations 2D issues d'une caméra, il est nécessaire d'obtenir une image géométrique de l'environnement, de même nature que celle donnée par la caméra vidéo. Pour cela nous allons mettre en œuvre une projection qui transforme les arêtes 3D des objets en segments situés sur un écran fictif.

Examinons le cas de la projection de l'arête d'un cube (figure 4.26). La caméra est matérialisée par un corps dont l'axe $\vec{z}$ définit la direction de son axe optique. À une distance f le long de cette direction est positionné un corps écran représentant le plan de projection. Les demi-contacts dc00 et dcEcr sont les repères de référence de la caméra et de l'écran.

La droite D support de l'arête à projeter est définie par l'axe $\vec{x}$ du demi-contact dc10. Un second demi-contact dc11 est positionné sur D par le lien PNN NNN. L'axe $\vec{x}$ du demi-contact dc02 définit la droite Dproj, image de D par la projection. D'une manière symétrique, on place dc03 relativement à dc02 avec le lien PNN NNN. Enfin, l'axe $\vec{z}$ du demi-contact dc01 détermine la direction de visée pour la transformation du point A.

Les conditions géométriques de projection consistent à exprimer d'une part l'alignement des points O, A et B, d'autre part la coplanéité des droites D et Dproj.

Un segment issu du traitement d'images sera défini dans notre représentation par l'axe $\vec{x}$ du demi-contact dc04 positionné par rapport à dcEcr avec une articulation qualifiée MMZ ZZM. On peut à présent exprimer une contrainte de contact entre dc02, le demi-contact projeté et dc04, le demi-contact relatif à l'information capteur.

Le détail de la spécification de ce contact projectif est présenté sur la figure 4.27. Structurellement, le contact projectif est un contact complexe construit autour de cinq contacts C_i

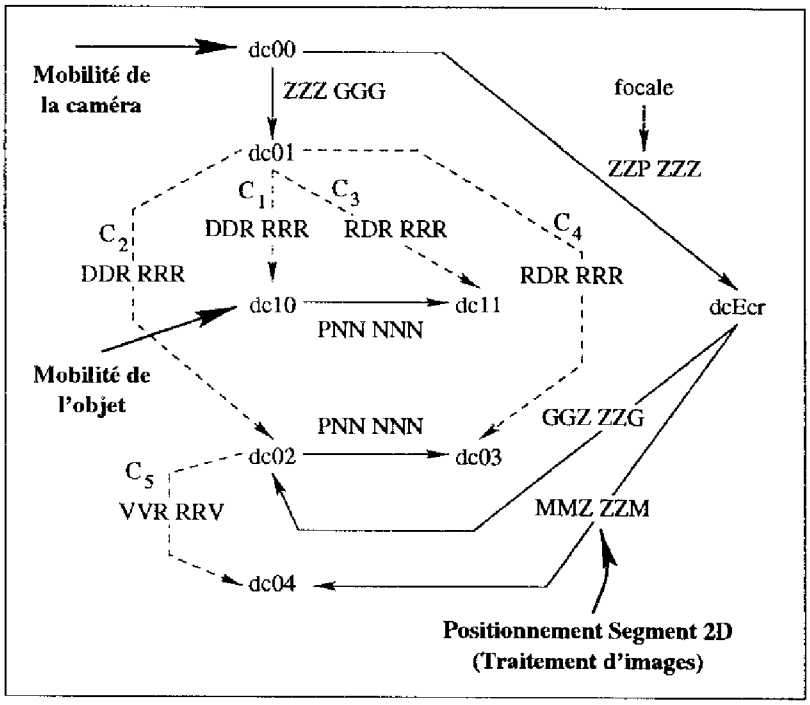

Figure 4.27: Spécification d'un contact projectif.

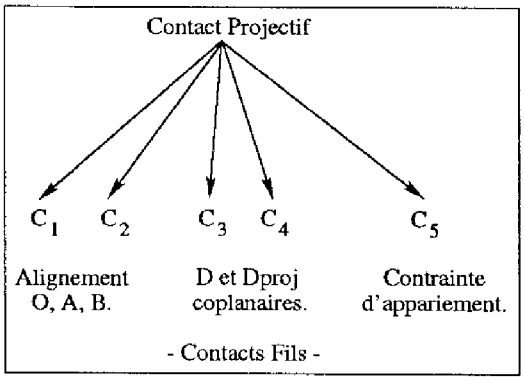

Figure 4.28: Structuration du contact projectif.

élémentaires (cf figure 4.28). Ces contacts sont générés, de façon transparente , à la création du contact projectif. Pour cela l'utilisateur doit fournir :

- les demi-contacts se rapportant à la caméra et l'écran (dc00 et dcEcr),
- le demi-contact à projeter (dc10),
- le demi-contact définissant le segment 2D à apparier (dc04).

Simulation de mouvement référencé vision

Considérons un robot de type SCEMI, modélisé au paragraphe 4.4.4, équipé d'une caméra vidéo. On désire garder dans le champ de vision de la caméra, l'image d'une pièce de forme

hexagonale qui se déplace[3] en ligne droite sur le plan de la table.

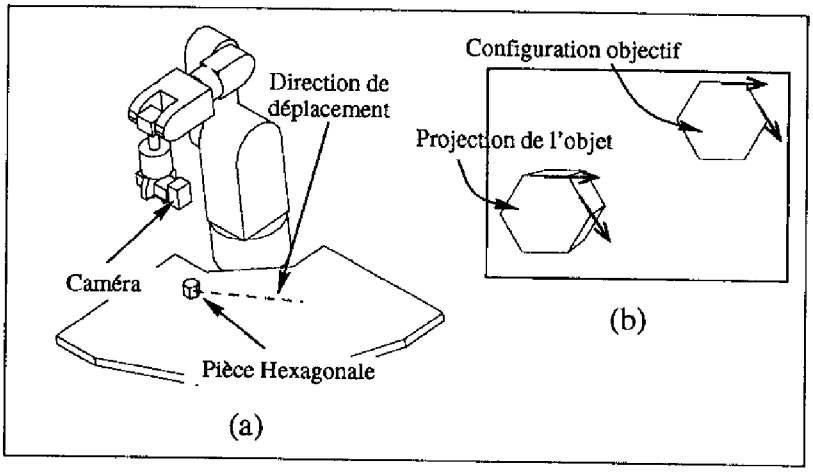

Figure 4.29: Position initiale du manipulateur.

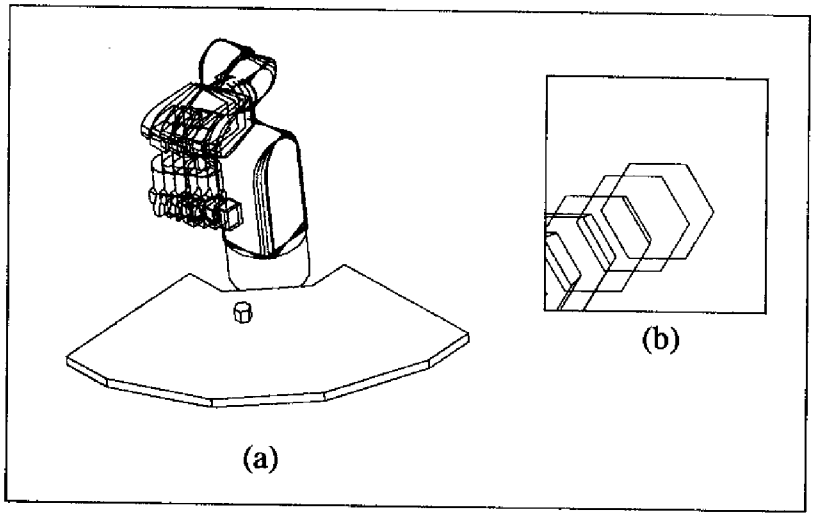

Figure 4.30: Déplacement référencé vision d'un manipulateur.

Les contraintes du problème sont définies à l'aide de deux demi-contacts dont les axes $\vec{x}$ sont positionnés sur deux arêtes de l'hexagone. Deux contacts projectifs s'appuyant sur les demi-contacts précédents sont déclarés pour exprimer que la projection des deux arêtes de l'hexagone doit se faire en une position donnée du plan image (figure 4.29b).

Partant d'une configuration dans laquelle les contraintes de projections ne sont pas vérifiées, les inconnues du problème de mouvement proposé sont les coordonnées articulaires du manipulateur (figure 4.29a).

La satisfaction de ces contraintes conduit au déplacement incrémental représenté sur la figure 4.30a. La série de positions perçues par la caméra est donnée sur la figure 4.30b.

[3]La vitesse de déplacement est bien sûr supposée être en accord avec les temps de calcul nécessités par la résolution de problèmes.

4.9.2 Intégration de données capteurs

Nous avons vu qu'il était possible d'exprimer la tâche que l'utilisateur souhaite faire accomplir au robot sous la forme d'un problème de minimisation d'un vecteur d'erreurs mesurant la satisfaction de contraintes de contact. La description des mobilités relatives donne accès, par le biais du raisonnement cinématique, à un modèle linéaire tangent de la fonction à minimiser.

Cette méthodologie peut s'apparenter, au moins dans son principe, à l'approche présentée par B. ESPIAU et C. SAMSON [ESPIAU 88]. Les auteurs considèrent en effet les interactions entre un ensemble de capteurs et son environnement local comme une liaison virtuelle dont les propriétés sont décrites et analysées par le formalisme des torseurs.

D'une façon analogue, l'application de notre approche pourrait s'envisager pour générer des commandes référencées capteurs sous réserve d'obtenir des temps de calcul compatibles avec les contraintes de l'automatique.

Les impératifs de calcul en temps réel semblent accessibles à condition d'utiliser une forme améliorée, sur le plan informatique, des compilations réalisées par la méthode de résolution.

En effet, la structure d'arbre de résolution contient et organise, dans un contexte d'exécution local, l'ensemble des opérations matricielles à réaliser pour satisfaire les objectifs de la tâche considérée. De cette manière, la méthodologie présentée peut donc être exploitée pour générer des fonctions de tâches nécessaires à la mise en œuvre de boucles référencées capteurs [SAMSON 90].

Toutefois, la mise en pratique effective pose la question du domaine de validité de la fonction de tâche ainsi élaborée. Les événements qui conduisent aux détections de singularités ou aux changements des choix de variables, modifient l'organisation des calculs dans l'arbre de résolution. Il faut alors reconsidérer la forme de la fonction de tâche initiale et commuter vers la nouvelle représentation. Une partie du travail à accomplir porte donc sur l'étude de ces transitions et des conditions de changement de fonction de tâche.

4.10 Conclusion

La méthodologie de spécification de tâches proposée, associée à la méthode de résolution a pour ambition de constituer un outil adapté à la programmation de tâches robotiques complexes.

La décomposition en problèmes et motifs cinématiques permet de maîtriser le volume important des structures de données mises en œuvre tandis que l'exploitation simultanée des ressources du modèle graphique 3D et de l'interpréteur autorise une spécification naturelle des problèmes.

La dépendance des solutions vis à vis des conditions initiales est une conséquence directe de l'approche numérique employée sans pour autant constituer un handicap. À l'inverse, cette caractéristique est mise à profit par l'utilisateur pour obtenir des trajectoires physiquement acceptables qui assurent des évolutions continues vers l'objectif. Les stratégies développées pour la gestion des singularités et le contrôle de la convergence améliorent fortement la robustesse globale du système.

Au titre des perspectives de développement de ce système, nous pouvons mention-

ner l'introduction possible d'un planificateur qui réaliserait l'installation automatique de contraintes comme, par exemple, celles relatives à l'évitement d'obstacles.

Chapitre 5

Expérimentation

Dans ce chapitre, nous allons illustrer, à travers un exemple inspiré d'un contexte d'application sous-marine, la mise en œuvre pratique de notre méthode de spécification de tâches sur un manipulateur réel. Nous ferons ensuite des propositions quant à l'application de ce système au contexte de la manipulation distante.

5.1 Une expérimentation canonique

La tâche canonique retenue pour une mise en œuvre expérimentale consiste à réaliser au moyen d'un robot manipulateur l'inspection d'un cordon de soudure existant entre deux tubes cylindriques. Cette opération représente l'une des tâches de maintenance fréquemment pratiquée sur des structures off-shore [Boeke 91, Maddalena 92].

Dans l'état actuel des technologies sous-marines, ce type de travaux est réalisé soit directement par des plongeurs soit à partir d'un submersible équipé d'un bras manipulateur. Dans ce cas, une première solution utilise un sous-marin à l'intérieur duquel un pilote téléopère le manipulateur. Une autre possibilité consiste à commander l'engin inhabité depuis un navire de surface auquel il est relié par un cordon ombilical[1].

Mais la présence d'un câble entraîne des difficultés pratiques importantes. Par exemple, la torsion excessive du câble ou le coincement de ce dernier dans la structure à inspecter sont des situations critiques pouvant conduire à la perte de l'engin.

L'utilisation d'engins mobiles autonomes de type AUV (Autonomous Untethered Vehicle) serait donc particulièrement adaptée. Cependant, pour y parvenir, de nombreuses difficultés restent à surmonter telles que la localisation du robot mobile, le contrôle de son attitude dans les courants marins ou encore la programmation au niveau tâche de l'engin qui sera doté le plus souvent d'un bras manipulateur.

Nous présentons dans ce qui suit, un scénario expérimental de spécification de tâche pour un manipulateur. Tout en reconnaissant le caractère modeste de l'expérimentation, nous pensons que ce travail pourrait servir de base au développement d'outils nécessaires à l'exécution de missions en milieu sous-marin.

[1] L'acronyme anglo-saxon ROV (Remotely Operated Vehicle) est souvent employé pour désigner ce type d'engins.

5.1.1 Site expérimental

Le robot utilisé pour l'expérimentation est un manipulateur SCEMI comportant six liaisons rotoïdes. Son armoire de commande, développée au LAAS, est construite à partir d'une architecture multi-processeurs assurant la gestion et la coordination des axes du robot. Un langage spécialisé de bas niveau permet d'accéder à différents modes d'asservissement du manipulateur (les fonctionnalités actuellement accessibles depuis une station de travail sont principalement les commandes en vitesse et position dans l'espace articulaire ou cartésien).

Notre logiciel de spécification de tâches fonctionne sur une station de travail de type SparcStation 10. La communication entre le logiciel et l'armoire de commande se fait par une ligne asynchrone. Depuis le logiciel, les fonctions bas niveau du contrôleur sont visibles grâce à un module dédié à la gestion du robot SCEMI. L'architecture ouverte et évolutive du logiciel permettrait d'intégrer sans difficulté d'autres modules pour commander des robots physiques différents. Les commandes qui transitent par la ligne asynchrone se limitent actuellement aux ordres permettant :

- d'initialiser le robot après sa mise sous tension,
- la lecture de la configuration articulaire courante,
- le déplacement du manipulateur de sa position courante vers une configuration articulaire donnée, avec une certaine vitesse.

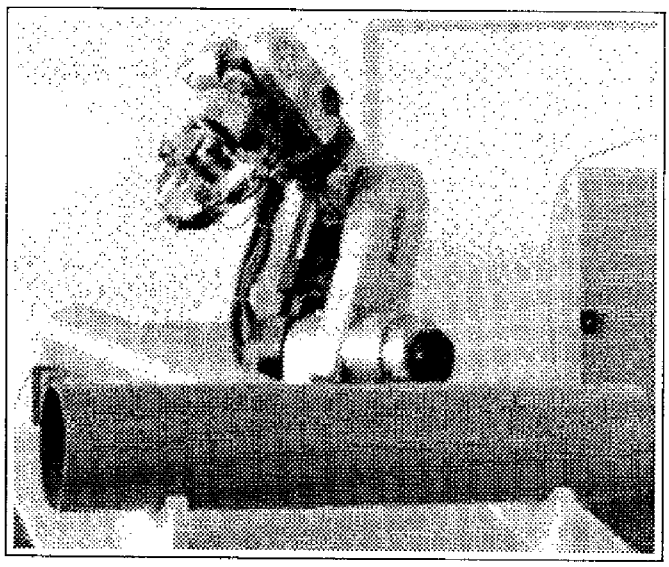

Figure 5.1: Robot SCEMI et tubes à inspecter.

Le montage expérimental est formé de deux tubes de PVC collés, assemblés sur un support. Ces tubes ont des diamètres extérieurs de 100 et 160 mm et leurs axes forment approximativement un angle de 145 degrés. Sur l'image de la figure 5.1 apparaissent le robot et les tubes à inspecter. Cette prise de vue a été obtenue en inclinant le support des tubes.

La réalisation des contacts entre le robot et l'environnement ainsi que l'inspection de l'intersection se fait au moyen d'un outil positionné sur l'organe terminal. L'outil choisi est un stylo installé dans un support que nous avons fait réaliser. Ce porte-outil est vissé à la place d'un des mors de la pince qui équipe le robot.

5.1.2 Spécification de la tâche

Analyse préliminaire

Les objets en présence dans cette tâche sont le robot, le palpeur, la table et les deux cylindres. La connaissance de l'environnement, d'un point de vue topologique, est supposée acquise. En revanche, les positions exactes des objets ne sont pas connues. Le déroulement de la manipulation va donc se décomposer en deux phases distinctes. La première portera sur la calibration du modèle de l'environnement tandis que la seconde consistera en l'exécution de la tâche de suivi d'intersection.

La tâche d'inspection de joint nécessite que l'on positionne un référentiel, attaché à un capteur, au dessus de l'intersection de deux cylindres en respectant des contraintes géométriques de distance et d'orientation.

L'approche de programmation par contraintes proposée permet de spécifier très naturellement cet objectif, et ce, indépendamment de la nature des surfaces qui s'intersectent. Nous présentons ci-après la spécification générale de ce problème.

Positionnement sur l'intersection de deux surfaces

Soit R_I un demi-contact à placer sur l'intersection de deux surfaces S_1 et S_2 sécantes. Nous écartons les situations dégénérées dans lesquelles l'intersection Γ n'est pas une courbe gauche de l'espace, par exemple, lorsque S_1 et S_2 réalisent un contact surfacique.

Par commodité, nous supposerons que le référentiel R_I est attaché à un corps que nous qualifierons de virtuel, susceptible de se déplacer indépendamment de S_1 et S_2 par le lien articulaire QQQ QQQ.

Les conditions d'orientation du repère R_I sont choisies de sorte que l'axe $\vec{x}(R_I)$ soit confondu avec la tangente à Γ. De cette manière, $\vec{x}(R_I)$ définit un sens positif de parcours le long de l'intersection. Finalement, la rotation de R_I autour de $\vec{x}(R_I)$ est mise à profit pour positionner l'axe $\vec{z}(R_I)$ par rapport aux normales aux surfaces S_1 et S_2.

L'utilisation des concepts de relation de contraintes entre corps conduit à la spécification de tâche représentée sur la figure 5.2. Les surfaces S_1 et S_2 repérées par les demi-contacts Ref_1 et Ref_2 sont modélisées grâce au concept de forme. Cette opération permet d'installer les demi-contacts R_1 et R_2 dont les axes $\vec{z}$ sont normaux à S_1 et S_2. La condition géométrique de positionnement sur l'intersection définie plus haut s'exprime simplement par les liens de contacts qualifiés VVV RVV entre les couples de demi-contacts (R_1,R_I) et (R_2,R_I).

Les contraintes d'orientation RVV imposent que l'axe $\vec{x}(R_I)$ soit dans les plans tangents aux surfaces S_1 et S_2. La satisfaction simultanée des contraintes de placement VVV et des contraintes d'orientation implique donc que l'axe $\vec{x}(R_I)$ soit tangent à l'intersection Γ.

Le degré de liberté en rotation autour de $\vec{x}$ est absorbé par une relation linéaire entre les composantes α_1 et α_2 étiquetées R, de la forme $\alpha_1 = k\alpha_2$. Si l'on choisit $k = -1$, nous pouvons introduire la variable d'écart e, qualifiée V, qui vaut $e = \alpha_1 + \alpha_2$. L'axe $\vec{z}(R_I)$ définit alors la bissectrice des directions des normales aux deux surfaces.

Le problème à résoudre consiste donc à calculer les composantes Q de positionnement du corps virtuel et les inconnues relatives aux formes S_1 et S_2 qui déterminent le placement de R_1 et R_2, pour satisfaire les contraintes V et la relation linéaire.

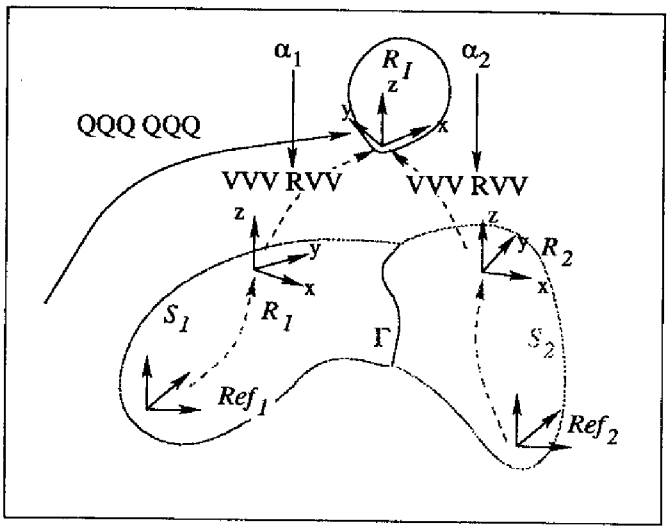

Figure 5.2: Positionnement sur l'intersection de deux surfaces.

Déplacement le long de l'intersection

Il faut maintenant trouver un moyen d'animation du corps virtuel pour le mouvoir sur l'intersection.

Le problème de positionnement sur l'intersection ayant été résolu, le corps virtuel se trouve en un point A de Γ (figure 5.3). On place alors, à l'aide d'un lien paramétrique PPP PPP, un demi-contact R_L confondu avec R_I. Une contrainte de progression est spécifiée en établissant un contact WRR RRR entre R_L et R_I. Ce lien de contact exprime que la prochaine position du repère R_I sera en A' situé à x mm du point A, suivant la tangente à Γ. Mais comme R_I est toujours astreint à demeurer sur l'intersection, la position atteinte à l'issue des itérations suivantes sera un point B de Γ.

L'écart de position entre les points A' et B sont dus aux effets non linéaires du second ordre. En particulier, si l'amplitude d'avancement x est « trop importante » par rapport à la courbure locale de Γ, le système pourra rencontrer des difficultés pour converger vers un point de l'intersection. La règle naturelle d'utilisation consiste donc à spécifier des incréments de progression suffisamment petits (i.e. adaptés au rayon de courbure local de l'intersection). Enfin, le sens de déplacement le long de l'intersection est fixé par le signe de la consigne x.

Dans la procédure décrite, le demi-contact R_L joue le rôle de « lièvre » que le repère R_I attaché au palpeur doit rattraper pour assurer la mise en mouvement du système.

D'une manière générale, ce mécanisme peut être employé dès lors que l'on souhaite animer un mécanisme soumis par ailleurs à un ensemble de contraintes.

La spécification de la tâche de suivi de joint est désormais immédiatement accessible. En effet, puisque les surfaces considérées dans l'intersection sont des cylindres, il suffit de reprendre la primitive de contact repère sur cylindre décrite au paragraphe 4.5 pour spécifier le mouvement du manipulateur vers l'intersection. La simulation de l'inspection est réalisée quant à elle au moyen d'un contact lièvre.

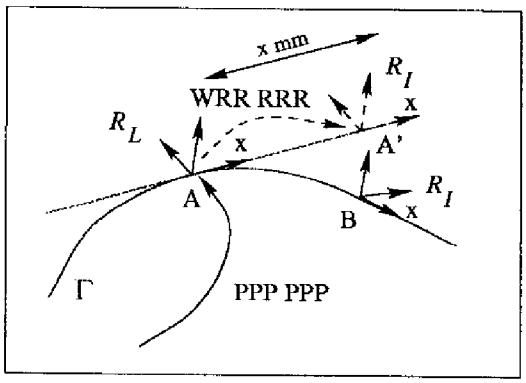

Figure 5.3: Progression le long d'une courbe de contrainte.

5.1.3 Calibration de l'environnement

Revenons maintenant au problème essentiel de la calibration des objets de l'environnement.

Calibration de la position de la table et du palpeur

Nous désirons localiser la table servant de support aux tubes et identifier la position du palpeur installé sur le manipulateur. Pour cela, nous proposons de toucher la table en plusieurs endroits avec l'outil. Les contacts ainsi réalisés seront supposés ponctuels.

Pour définir la normale au plan de la table ainsi que son « altitude », nous qualifierons l'articulation entre la table et le robot PPM MMP (voir figures 5.4 et 4.14 pour l'orientation des référentiels). Les position et longueur de l'outil sont déterminées en qualifiant l'articulation (pince,palpeur) MMM PPP. Enfin, la contrainte de contact à satisfaire entre le demi-contact de la table et l'outil sera de la forme RRW RRR.

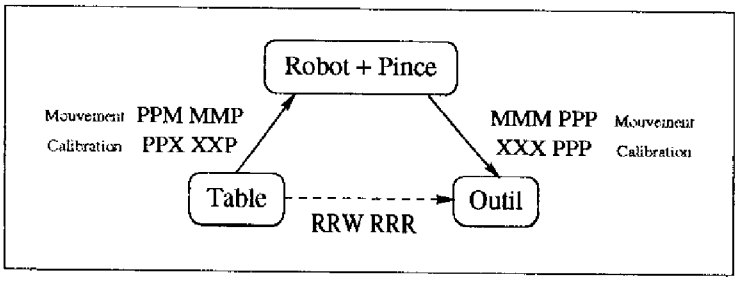

Figure 5.4: Contraintes pour la calibration de la table et du palpeur.

Calibration des cylindres

À ce stade, on suppose connues les positions de la table et du palpeur et nous adoptons le schéma de travail défini au paragraphe 4.8.2 et sur la figure 4.20. Cette spécification permet de calibrer à tour de rôle les deux cylindres posés sur la table.

5.1.4 Mise en œuvre des stratégies de calibration

La technique de calibration envisagée est basée sur la réalisation de mouvements au contact entre l'outil et les objets dont il faut identifier les positions.

Naturellement, lorsque débute la session de travail, l'espace de travail du robot n'est connu que partiellement et la précision du modèle 3D ne permet pas de spécifier directement les mouvements au contact.

La calibration par contact de l'environnement comporte deux niveaux de difficulté. Le premier consiste à définir une stratégie de prise de mesures, c'est à dire déterminer les points de contact ainsi que l'orientation à imposer à l'outil. Le second volet du problème consiste à générer des déplacements vers les points de mesure en évitant les collisions avec les objets de l'espace de travail.

Stratégie de collecte de points de mesure

Sans parler des incertitudes inhérentes à tout processus de mesure, la qualité du résultat d'une calibration dépend fortement du choix des points de mesure et de leur localisation sur l'objet dont on cherche à identifier la position.
Par exemple, pour la calibration d'un plan, il faudra s'efforcer de générer des points de mesure à la fois non alignés et suffisamment « éloignés » les uns des autres. Pour l'évaluation de la position (pince, palpeur), des orientations différentes du palpeur par rapport à la table sont de nature à améliorer la précision de la calibration.

La génération automatique d'une stratégie globale pour le choix de points de mesure judicieux nous paraît difficilement envisageable dans le cas général. En revanche, s'il dispose d'informations visuelles sur le site de travail, un opérateur humain est capable de juger, au moins d'un point de vue qualitatif, de la pertinence ou non des points de mesure. C'est de cette façon que nous avons généré les jeux de mesures nécessaires aux différentes calibrations.

Gestion des mesures

Une mesure est constituée par l'ensemble des informations relatives à chaque lien du problème de mouvement qui lui est associé. Concrètement, les éléments consignés dans une mesure sont le qualificatif et le vecteur de consigne de chaque lien du graphe cinématique du problème de mouvement. Le transfert bidirectionnel entre un problème de mouvement et une mesure se fait par copie partielle ou totale de ces informations.

Afin de pouvoir récupérer des mesures acquises au cours de sessions antérieures, nous avons développé les fonctionnalités permettant la lecture et l'archivage de mesures sur disques.

L'augmentation de la souplesse d'utilisation lors des échanges entre problème de mouvement et mesure, nous a conduit à définir la notion de *vecteur d'état* d'un problème. Ce vecteur est constitué de l'ensemble des valeurs des composantes qualifiées X, Q ou O. À un problème de mouvement donné, nous avons attaché plusieurs vecteurs d'état distincts qui permettent de stocker ou rappeler, par une procédure moins lourde que l'utilisation de mesures, différentes positions du modèle de simulation. Parallèlement, une mesure comportera également des vecteurs d'états.

Les possibilités de transfert entre mesure et problème sont résumées sur la figure 5.5.

Une utilisation typique est l'exécution de tâches de déplacements point à point du robot.

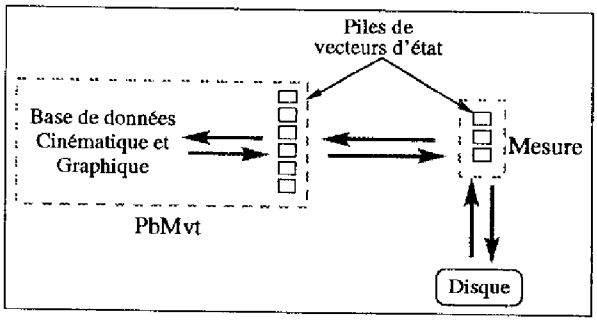

Figure 5.5: Transfert des informations entre problème de mouvement et mesure.

Évitement des obstacles lors des mouvements vers les points de mesure

Nous avons utilisé les fonctionnalités offertes par le contrôleur pour faire exécuter au robot des déplacements point à point, dans l'espace articulaire.

La stratégie adoptée est très simple et consiste à réaliser pour chaque mesure, la séquence suivante : Approche ⇒ Mise en contact ⇒ Dégagement.

Pour éviter les collisions, le mouvement d'approche passe successivement par trois points distincts : une position de dégagement, un point de passage intermédiaire et enfin, une position proche du contact. La phase de dégagement, qui suit la mise en contact, reproduit, en sens inverse, le mouvement d'approche.

Ces trois positions sont déterminées par apprentissage de sorte que le transfert d'un point au suivant ne génère pas de collision. Cette procédure fait appel au modèle graphique de simulation.

Après avoir généré les trois points (dégagement, intermédiaire et approche), une mesure et trois vecteurs d'état sont créés. Ces vecteurs d'état correspondent aux points de passages précédents.

5.1.5 Résultats

Calibration du palpeur et de la table

La figure 5.6 représente la séquence d'acquisition d'un point de mesure. La phase (c) correspond au déplacement incrémental du palpeur vers la table, commandé en mode interactif depuis le logiciel (en l'absence de capteur sur le robot). Un mode de commande gardée du mouvement (par mesure des efforts de contacts et usage d'un détecteur de contact adapté) est évidemment indispensable pour une utilisation en téléprogrammation.

Les résultats de calibration de la table et de l'outil sont présentés dans le tableau suivant. Ces résultats ont été obtenus avec six contacts.

Calibration des cylindres

La base de données graphique prenant en compte les positions calculées précédemment pour la table et l'outil, il est possible de calibrer les deux cylindres.

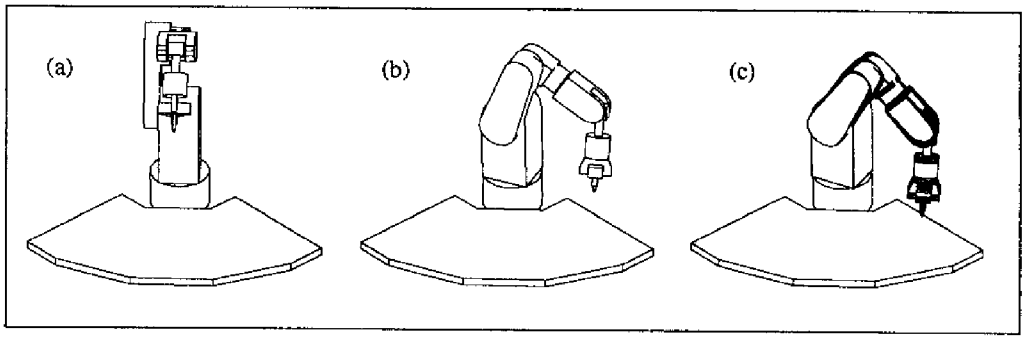

Figure 5.6: Un cycle de mouvement pour la calibration de la table et du palpeur.

	Estimation initiale	Position calculée
Articulation Robot ⇒ Table	80 80 -90 0 0 1 45	80 80 -119.736 -0.003 -0.005 1 45.0009
Articulation Pince ⇒ Palpeur	0 0 -140 0 0 0 1 0	-5.325 -11.813 -176.302 0 0 0 1 0

Tableau 5.1: Résultat de la calibration de la table et du palpeur.

La collecte d'un point de mesure sur le cylindre 1 est décrite sur la figure 5.7. Les positions des cylindres représentés correspondent aux estimations initiales. Le nombre de contacts nécessaires pour la calibration s'élève à sept pour le premier et six pour le second cylindre.

Une difficulté pratique importante est liée aux problèmes d'accessibilité du robot pour venir en contact avec les cylindres. Dans la configuration du montage expérimental, seuls des points situés sur un secteur angulaire d'environ 170 degrés, étaient accessibles au robot.

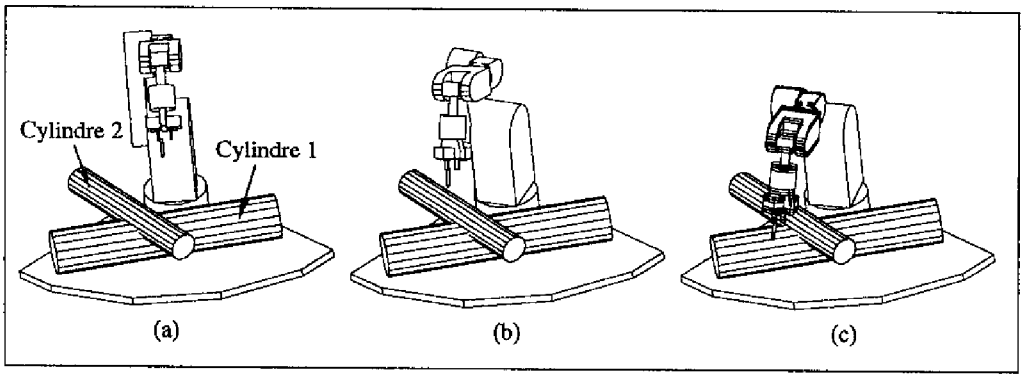

Figure 5.7: Mouvement au contact pour la calibration du premier cylindre .

Tâche de suivi de joint

Les positions des objets de l'environnement ayant été déterminées, la tâche de suivi d'intersection peut être réalisée.

	Estimation initiale	Position calculée
Articulation Table ⇒ Cylindre 1	850 200 80 0 0 1 145	849.594 225.973 152.145 0.005 0.029 2.532 145.095
Articulation Table ⇒ Cylindre 2	440 490 90 0 0 1 -85	461.697 514.497 150.345 -0.019 0.016 -1.705 97.697

Tableau 5.2: Résultat de la calibration des cylindres.

L'approche de description des tâches par contraintes offre une grande souplesse d'utilisation. Trois problèmes de mouvement ont été élaborés pour réaliser l'objectif souhaité :

- $\mathbf{Pb_1}$: positionnement du corps virtuel sur l'intersection des cylindres.
- $\mathbf{Pb_2}$: mouvement du manipulateur vers le corps virtuel.
- $\mathbf{Pb_3}$: déplacement le long de l'intersection.

Cette décomposition de la tâche permet en particulier de définir très simplement les contraintes de placement du palpeur par rapport à l'intersection grâce à un lien WWW WWW entre le corps virtuel (situé exactement sur l'intersection) et le palpeur.

Dans une première phase, un point particulier de l'intersection est choisi et définit le lieu d'accostage au palpeur. Le corps virtuel est alors déplacé à cet endroit par la résolution des problèmes $\mathbf{Pb_1}$ et $\mathbf{Pb_3}$ (figure 5.8a).

À partir de sa configuration de dégagement, le manipulateur rallie le corps virtuel dans un déplacement continu en résolvant $\mathbf{Pb_2}$ (figure 5.8b).

Finalement, la résolution simultanée des trois problèmes engendre un mouvement continu du robot le long de l'intersection (figure 5.8c).

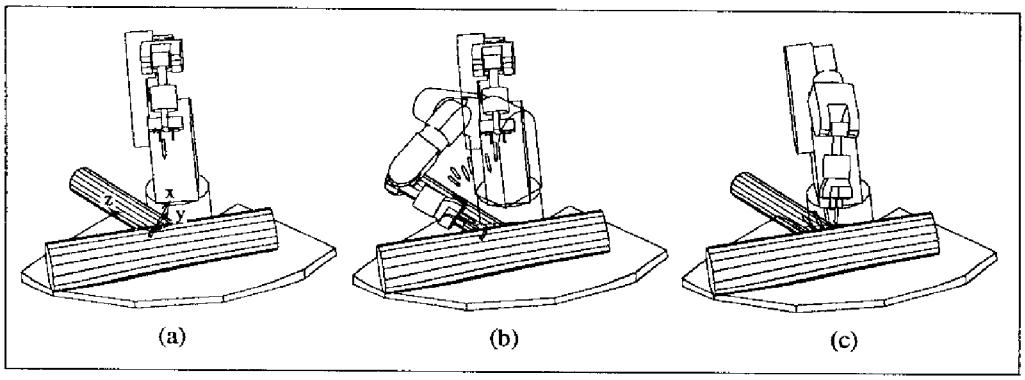

Figure 5.8: Déroulement de la tâche de suivi de joint.

5.1.6 Discussion

Les précisions de positionnement le long de l'intersection obtenues après calibration sont de l'ordre de 5 mm. Ce résultat moyennement satisfaisant est obtenu sachant que le robot

SCEMI utilisé dispose de codeurs incrémentaux nécessitant une procédure d'initialisation préalable dont la répétabilité et la précision ne sont pas excellentes.

La qualité de calibration est influencée par deux principaux facteurs. Nous pouvons mentionner la précision du manipulateur en tant qu'instrument de mesure mais aussi, la qualité du protocole de mise en contact pour la collecte des mesures.

Les erreurs de modélisation

Les imprécisions sur le modèle d'un manipulateur sont nombreuses et ont des conséquences géométriques comme les défauts d'entr'axe, d'orientation des axes de liaisons, de positionnement des zéros codeurs, mais également non géométriques telles que l'élasticité des transmissions ou les jeux mécaniques.

Ces erreurs proviennent, pour une grande part, des processus de fabrication dont l'amélioration directe affecte fortement les coûts. Il paraît judicieux de corriger les effets de ces erreurs sur les modèles et les outils logiciels d'exploitation du robot.

L'approche proposée permet d'évaluer la conséquence des erreurs géométriques par l'extraction de matrices de sensibilité relatives à des paramètres du modèle. En cas de besoin, on peut imaginer un scénario pour la calibration géométrique du robot conduisant à l'identification de paramètres de son modèle [HOLLERBACH 89].

Pour ce qui concerne le protocole de calibration par contacts, une analyse, analogue à celle décrite au paragraphe 4.8.3, permettrait d'améliorer le choix des points de mesure par l'étude de l'influence de ces positions sur le résultat de calibration.

Alternative au contact physique réel

La réalisation de contacts physiques avec l'environnement peut présenter des inconvénients dans certains contextes applicatifs. C'est notamment le cas des applications spatiales en l'absence de gravité. Une solution serait d'utiliser une caméra vidéo avec des contacts projectifs sur cylindres. La spécification de contraintes (de nature voisine à celle présentée au paragraphe 4.9.1) porterait alors sur les couples d'arêtes extraites de l'image vidéo, correspondantes aux contours apparents des cylindres.

Remarquons au passage que pour les applications sous-marines, l'obtention d'images vidéos de bonne qualité est parfois problématique. Une cause fréquente de cette limitation est la présence de particules en suspension dans l'espace de travail provenant du nettoyage préliminaire des structures à inspecter. Dans ces circonstances, la réalisation de contacts physiques peut alors s'avérer indispensable.

Intégration d'outillages spécifiques

Dans l'hypothèse où la précision de modélisation pour accomplir la tâche serait toujours insuffisante en dépit des tentatives de prise en compte des erreurs précédentes, l'usage d'un outillage spécialisé pourrait s'envisager et s'intégrer à la modélisation de la tâche.

L'accroissement de la précision d'exécution de la tâche passe par une collecte d'informations relatives à l'environnement local du joint de soudure à inspecter. Pour atteindre cet objectif, nous pourrions imaginer un dispositif articulé de taille réduite, associé à la sonde

d'inspection de cordon de soudure, permettant de « mesurer » le placement local par rapport aux deux cylindres.

Un tel système pourrait être formé de deux ou trois doigts télescopiques à maintenir en contact simultané avec les deux cylindres. Cette configuration, donnée initialement par le modèle calibré des cylindres, n'est pas suffisamment précise pour accomplir l'inspection.

Connaissant par construction la géométrie du mécanisme et la position relative du capteur, la mesure des élongations de chaque doigt pourrait alors être interprétée dans le but de repositionner la sonde plus précisément au dessus de l'intersection. Nous voyons donc apparaître une décomposition naturelle pour la gestion des incertitudes :

- un premier niveau où la calibration initiale des cylindres permet de placer un dispositif de mesure locale,
- le second niveau qui exploite une mesure locale de la tâche pour affiner le placement du capteur.

Quelle que soit la stratégie employée, il subsistera toujours un degré d'incertitude lié à l'occurrence de faits imprévus tel que la présence d'obstacles non introduits dans le modèle. Face à de telles situations, une réponse adaptée est donnée par l'activation automatique d'une stratégie de garde du mouvement.

5.2 Perspectives d'utilisation dans un contexte de manipulation distante

Lorsque les échanges d'informations entre un système robotisé distant et sa station de contrôle sont limités en volume et présentent des retards, la commande du robot devient problématique.

Les contraintes de communication et les incertitudes inhérentes à l'exécution de tâches à distance nécessitent un modèle de l'espace de travail servant de base au raisonnement et à la définition des tâches.

Scénario de travail proposé

L'outil de spécification de tâches de manipulation complexes que nous avons présenté peut jouer un rôle central dans un système de téléprogrammation de manipulateurs distants.

Une architecture de téléprogrammation

Le contexte de manipulation distante dans lequel nous nous plaçons fait intervenir une station opérateur et un site distant dans lequel un manipulateur est amené à intervenir.

Cette partition naturelle se retrouve dans l'architecture de téléprogrammation décrite sur la figure 5.9. Elle comprend deux sites local et distant, tous deux pourvus d'un environnement de spécification et de résolution de tâches.

L'opérateur intervient au niveau du site local pour définir et générer des spécifications de tâches : il s'agit de la première boucle locale de simulation.

Au niveau du site distant, le schéma de simulation se développe selon le triptyque : interprétation – résolution – exécution, indépendamment des contraintes de communication entre les sites local et distant.

Sur la figure 5.9, nous mentionnons à titre prospectif la présence d'un planificateur qui pourrait intervenir, en fonction d'informations locales provenant de capteurs du robot, sur les spécifications de problèmes. La boucle de simulation ainsi réalisée caractériserait le degré d'autonomie pour l'exécution de tâches.

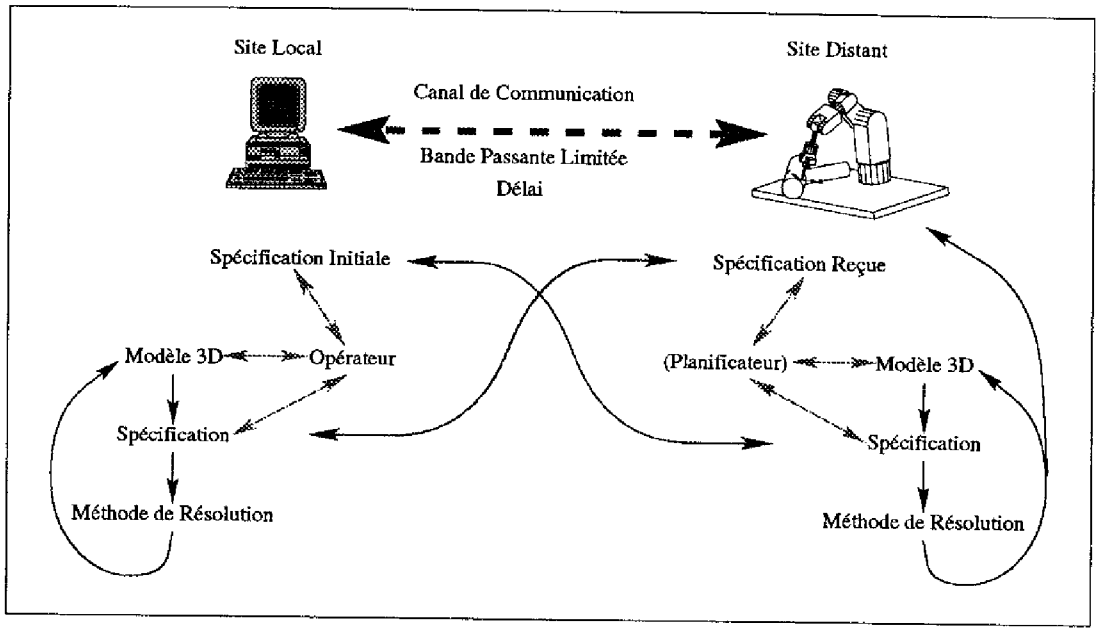

Figure 5.9: Architecture pour la téléprogrammation de manipulateurs distants.

Commandes transitant par le canal de communication

En début de session, il est nécessaire d'établir une base de travail sur laquelle l'opérateur pourra s'appuyer pour générer la spécification de tâche. Généralement, les intentions de l'opérateur seront suffisamment précises pour autoriser la construction d'un script initial regroupant la plupart des objets présents dans la zone de travail et une ébauche des spécifications de tâches.

Au besoin, des informations supplémentaires provenant de caméras vidéos installées sur le site distant devront être récupérées pour compléter le modèle initial.

Le canevas de travail étant élaboré, le script correspondant est téléchargé sur le site distant. Pour donner un ordre d'idée, le script de spécification de la tâche d'inspection de joint représente un volume d'environ 40 ko.

Les besoins d'échanges entre les deux sites concernent principalement :

1. Les actions à accomplir, fournies sous la forme symbolique de problèmes de mouvement ou de calibration à résoudre.

2. La mise à jour réciproque des modèles; ce besoin se manifeste notamment après une phase de calibration.

3. Les comptes-rendus d'exécution qui renseignent l'opérateur sur l'avancement de la tâche.

4. Les procédures de reprise d'erreur qui nécessitent l'intervention de l'opérateur pour décider de la suite à donner aux opérations.

Nous avons développé les deux premiers points qui correspondent aux fonctionnalités de base d'un système de téléprogrammation.
L'objectif 1 est assuré par l'utilisation de l'interpréteur de commande qui traduit puis résoud les spécifications de tâches. Le résultat de cette résolution est fournie au manipulateur réel sous la forme de suites de configurations articulaires directement exploitables par son armoire de commande.
La mise à jour des modèles s'appuie sur l'échange de mesures. En terme de volume, la transmission de ce type de données est peu coûteuse. Pour la tâche de suivi de joint, une mesure typique correspond à environ 2.5 ko de données.

5.3 Conclusion

Pour prétendre à un système opérationnel, il reste à traiter les points 3 et 4. De tels développements impliquent une confrontation avec des conditions réelles d'utilisation.

Sans parler de la présence d'un planificateur dans le schéma de simulation distante, l'autonomie du manipulateur réel repose avant tout sur l'existence de primitives de commandes référencées capteurs. Comme nous l'avons déjà mentionné à plusieurs occasions (§ 4.9 et § 5.1.6), l'intégration de données capteurs à notre système de spécification de problèmes devrait permettre d'accroître l'autonomie d'exécution des tâches par le manipulateur distant.

Cette autonomie sensorielle doit être recherchée en priorité pour donner un sens concret aux efforts déployés pour le développement d'une autonomie décisionnelle.

Conclusion

Ce mémoire fait la présentation de nos travaux sur la spécification et la résolution de tâches de manipulation complexes.

Ce travail s'efforce de répondre d'une part, aux besoins de génération de mouvements pour des robots manipulateurs, d'autre part, aux questions concernant la calibration d'objets dans l'environnement de travail. Le système de programmation de tâches proposé réalise l'intégration de nombreuses fonctionnalités, regroupées par modules pour former un ensemble homogène.

La principale caractéristique de l'approche est de proposer une définition des objectifs par contraintes. La spécification par contraintes dans l'espace de la tâche constitue une approche naturelle pour la description de tâches de manipulation. Elle représente un choix alternatif aux techniques basées sur la téléopération ou celles utilisant des méthodes de planification de trajectoires.

Pour mener à bien la modélisation des problèmes, les robots et leur environnement de travail sont assimilés à des systèmes mécaniques de corps rigides. La spécification des tâches repose sur la description des propriétés géométriques et cinématiques de ces mécanismes, facilitée par l'utilisation d'un modèle graphique 3D. L'expression des objectifs est alors réalisée par un système d'étiquetage de variables introduites lors de la phase de modélisation des problèmes. Par ce moyen, les objectifs sont traduits en termes de contraintes de contacts à établir ou maintenir entre les corps. La structuration des problèmes sous la forme de graphes cinématiques permet d'engendrer automatiquement les relations de contraintes qu'il faut résoudre pour satisfaire les objectifs de la tâche.

Cette résolution est assurée par une méthode fondée sur l'utilisation d'un mécanisme de triangulation de Gauss. Ce dernier permet d'alimenter un schéma numérique itératif de type Newton-Raphson associé à une méthode du simplexe qui autorise la prise en compte de contraintes inégalités.
La mise à l'épreuve de cette méthode dans la résolution d'un grand nombre de cas a fait ressortir un niveau très élevé de robustesse. Principalement, ce résultat est le fruit des stratégies mises en œuvre pour le traitement des singularités d'ordre numérique et des problèmes de convergence.
Ces caractéristiques font de ce module de résolution, un outil adapté à l'animation de mécanismes complexes mais convient également au traitement de problèmes d'identification paramétrique de modèles.

L'approche proposée a fait l'objet d'une expérimentation sur un manipulateur réel pour accomplir une tâche de suivi de joint de soudure entre deux cylindres. Cet exemple illustre la mise en œuvre d'un protocole de calibration de l'environnement et la génération de mouvements complexes. Ces deux aspects sont fondamentaux pour la téléprogrammation de tâches

de manipulation sur site distant.

Pour cette classe d'applications, nous avons présenté une architecture basée sur l'emploi du système de spécification et de résolution de tâches. Elle comprend les sites local et distant, chacun équipé d'un environnement de simulation. Les ordres transitant par le canal de communication sont principalement des spécifications de tâches. Leur nature symbolique permet de réduire le volume des échanges à opérer entre les deux sites et diminue ainsi les conséquences dûes au retard et au faible débit.

En perspective, les potentialités d'exploitation opérationnelle de cette approche nous paraissent prometteuses.

L'accroissement de l'autonomie d'exécution des tâches demande des efforts suivant plusieurs directions. Tout d'abord, mentionnons la nécessité d'instrumentation du robot associée au développement, dans le système de commande, de primitives référencées capteurs qui doterait le robot d'un certain niveau d'« intelligence sensorielle ».

D'autre part, il nous paraît important de faire preuve de créativité pour concevoir des outillages ou dispositifs péri-robotiques nouveaux qui permettraient d'améliorer localement le contrôle d'exécution des tâches. En effet, dans le cadre de missions d'intervention, un robot est généralement amené à changer son outillage en cours d'exécution des tâches.

Il importe donc de disposer d'un outil très flexible qui permette, sans effort excessif, l'intégration, dans les modèles de la tâche, de nouveaux éléments. Inversement, cette flexibilité d'emploi peut servir d'aide à la définition d'outils nouveaux grâce aux larges possibilités de simulation.

Une autre source d'amélioration concernerait les modalités de spécification qui devraient certainement être aménagées en terme d'ergonomie en proposant l'accès aux fonctionnalités par l'intermédiaire de boîtes à outils. Les cahiers des charges de telles boîtes à outils devront être guidés par les besoins pratiques.

Dans le système actuel, la spécification des tâches est prise en charge par l'utilisateur. Les contraintes liées à l'évitement d'obstacles relèvent essentiellement de raisonnements sur les distances séparant les objets. Pour contribuer à l'autonomie, nous pourrions envisager de rendre automatique, au moins partiellement, l'introduction ou la désactivation de telles contraintes par usage, sur le modèle graphique, de méthodes de détection d'interférences. La mesure correspondante, dans l'espace réel, serait mise en œuvre au moyen de capteurs proximétriques.

Enfin, pour ce qui concerne le système de simulation, nous pouvons mentionner les travaux entrepris pour prendre en compte les efforts statiques dans les mécanismes. En prenant pour base la modélisation cinématique et le module de résolution, le calcul des efforts statiques devrait permettre l'intégration de données provenant de capteurs d'efforts et favoriser le développement de commandes pour générer des mouvements compliants.

Annexe A

Relation entre vitesse angulaire et dérivée du vecteur rotation

Les possibilités pour représenter les rotations dans l'espace sont multiples. Parmi les formes les plus employées figurent les matrices de rotations, les quaternions ou encore les angles d'Euler. La forme différentielle de chacune de ces représentations peut être obtenue et permet de faire le lien entre le vecteur vitesse de rotation et la dérivée des paramètres de la représentation.

Avec le paramétrage que nous avons retenu, une rotation d'angle θ (en rad) est définie par un vecteur de dimension trois définissant l'axe de rotation :

$$Q_{rot} = \begin{bmatrix} rx & ry & rz \end{bmatrix}^T \quad \text{avec} \quad \theta^2 = rx^2 + ry^2 + rz^2$$

Nous recherchons l'expression reliant le vecteur rotation V_{rot} à $\dot{Q}_{rot}$ sous la forme :

$$V_{rot} = J\dot{Q}_{rot} \qquad \text{avec} \quad V_{rot} = [\ \omega_x \quad \omega_y \quad \omega_z\ ]^T \tag{A.1}$$

La méthode implantée initialement dans le système, valable pour tout θ de $\mathbb{R}$, était basée sur la relation entre la dérivée de la matrice rotation et le vecteur V_{rot} :

$$\frac{dR}{dt} = \widetilde{V}_{rot}\, R \qquad \text{avec} \quad \widetilde{V}_{rot} = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix} \tag{A.2}$$

où $\widetilde{V}_{rot}$ représente la matrice de préproduit vectoriel de V_{rot}.

Le principe du calcul est le suivant : après avoir transformé la représentation de la rotation Q_{rot} sous la forme d'une matrice de rotation R, on calcule explicitement la jacobienne $\frac{dR}{dQ_{rot}}$. L'équation précédente s'écrit : $\widetilde{V}_{rot} = \frac{dR}{dQ_{rot}}\ \dot{Q}_{rot}\ R^T$, où la matrice J peut être identifiée. En effet, cette expression définit une relation linéaire entre V_{rot} et $\dot{Q}_{rot}$ qu'il suffit de mettre sous la forme désirée (A.1).

Cette procédure, coûteuse en temps de calcul, peut être améliorée en recherchant par un moyen plus direct la matrice J de (A.1).

Pour des valeurs de θ comprises entre zéro et π, nous avons le calcul explicite suivant [Renaud 95] :

$$\begin{array}{rclcrcl} C & = & \cos|\theta| & & \Sigma & = & \dfrac{|S|}{|\theta|} \\ |S| & = & \sin|\theta| & & & & \\ \Gamma & = & \dfrac{1-C}{\theta^2} & & N & = & \dfrac{1-\Sigma}{\theta^2} \end{array}$$

$$J = \begin{bmatrix} Nrx^2+\Sigma & Nrxry-\Gamma rz & Nrxrz+\Gamma ry \\ Nrxry+\Gamma rz & Nry^2+\Sigma & Nryrz-\Gamma rx \\ Nrxrz-\Gamma ry & Nryrz+\Gamma rx & Nrz^2+\Sigma \end{bmatrix} \tag{A.3}$$

Nous nous sommes intéressés à l'extension de ce résultat à $\mathbb{R}$ tout entier. Pour cela, nous avons comparé numériquement les valeurs calculées par (A.3) avec celles obtenues par la méthode (A.2) qui est définie sur $\mathbb{R}$.

Dans un premier temps, la relation (A.3) s'est également avérée exacte pour $\theta \in [2k\pi; (2k+1)\pi]$ et les intervalles symétriques $[-(2k+1)\pi; -2k\pi]$, $k \in \mathbb{N}$. En revanche, elle ne l'est pas sur les intervalles restant $[(2k+1)\pi; (2k+2)\pi]$ (et leurs symétriques).

Le problème provient du terme Σ qui comporte des valeurs absolues. Nous avons constaté qu'avec $\Sigma = \frac{S}{\theta}$, l'expression (A.3) était valable quelle que soit la valeur de θ pris dans $\mathbb{R}$. Nous avons donc adopté cette méthode de calcul plus efficace que la procédure (A.2).

Annexe B

Génération de trajectoires et de lois de mouvements

Dans les situations où l'espace de travail est très bien connu, il peut être avantageux de générer des trajectoires par des techniques classiques d'interpolation, sans faire appel à la méthode de spécification par contraintes décrite dans ce mémoire.

Les tâches robotiques envisagées dans ce contexte sont principalement les déplacements de transfert dans l'espace libre mais également les mouvements à proximité d'obstacles. Dans cette annexe, nous présentons des développements relatifs à la génération de trajectoires et la génération de lois de mouvement sur des trajectoires imposées. Ces éléments ont été développés au début de notre travail de thèse dans l'intention de les incorporer à une nouvelle armoire de commande du manipulateur.

B.1 Génération de trajectoires dans l'espace cartésien

La génération de trajectoires cartésiennes en robotique de manipulation implique des raisonnements sur les positions mais aussi sur les orientations des objets mobiles.

B.1.1 Modélisation

Considérons une trajectoire Γ construite à partir d'un ensemble de points définissant des configurations cartésiennes. Ces points de contrôle sont positionnés au moyen de référentiels dans lesquels peuvent être exprimées les directions des tangentes à la trajectoire. Nous utiliserons notre représentation habituelle par des vecteurs de configurations de $\mathbb{R}^6$.

Pour appréhender de la même manière la gestion des positions et des orientations, Γ est décrite sous la forme d'une fonction vectorielle de dimension six dépendante d'un paramètre d'abscisse curviligne s.

$$\Gamma(s) = \begin{bmatrix} \left.\begin{matrix} x(s) \\ y(s) \\ z(s) \end{matrix}\right\} = Pos(s) \\ \left.\begin{matrix} rx(s) \\ ry(s) \\ rz(s) \end{matrix}\right\} = Rot(s) \end{bmatrix}$$

Les composantes des fonctions Pos et Rot réalisent l'approximation et/ou l'interpolation des points de contrôle de la trajectoire par des cubiques. Ce choix correspond à la complexité

minimale compatible avec la richesse de description souhaitée. Cependant, d'autres fonctions d'interpolation (courbes de Bézier, splines...) auraient pu être employées d'une manière similaire.

Considérons deux points M_1 et M_2 à relier par une trajectoire continue, paramétrée par une abscisse curviligne dans le sens positif de M_1 vers M_2. La différence des abscisses curvilignes $s(M_2) - s(M_1)$ est prise égale à la norme du vecteur de configuration $Q_{M_1M_2}$.

Sur les six composantes de la trajectoire Γ , le problème consiste à déterminer les cœfficients des cubiques en exprimant les conditions de passage et de tangence aux points de contrôle M_1 et M_2. Pour chaque composante entre deux points apparaissent donc quatre équations qui permettent de calculer les cœfficients d'une cubique.

Si l'on considère la composante $x(s)$, modélisée par une cubique de cœfficients inconnus (a, b, c, d), nous obtenons le système linéaire suivant :

$$\begin{bmatrix} s_1^3 & s_1^2 & s_1 & 1 \\ s_2^3 & s_2^2 & s_2 & 1 \\ 3s_1^2 & 2s_1 & 1 & 0 \\ 3s_2^2 & 2s_2 & 1 & 0 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} x(s_1) \\ x(s_2) \\ \frac{dx}{ds}(s_1) \\ \frac{dx}{ds}(s_2) \end{bmatrix} \tag{B.1}$$

où s_1 et s_2 sont les abscisses curvilignes de M_1 et M_2. Ce système linéaire est traité par le module de résolution décrit dans le chapitre 3.

Pour avoir une plus grande souplesse de définition des trajectoires, l'utilisateur a la possibilité de spécifier des points de passage imposés (interpolation) ou bien approchés (approximation). Pour cela, un système d'étiquettes permet de qualifier les composantes des vecteurs $Pos(s)$ et $Rot(s)$ relatifs aux positions et orientations des points de contrôle. Cette technique s'applique également aux vecteurs dérivés $\frac{dPos}{ds}(s)$ et $\frac{dRot}{ds}(s)$ définissant les tangentes à la trajectoire. Les composantes à interpoler (resp. approximer) seront résolues en contraintes de type égalité (resp. moindres carrés).

B.1.2 Utilisation

Trajectoire sans gestion de l'orientation

La figure B.1-a présente un exemple de trajectoire cartésienne passant par trois points de contrôle A, B et C. Le tableau suivant précise les conditions de tangence aux points de contrôle. Les orientations ne sont pas pilotées (vecteurs $\frac{dRot}{ds}$ nuls). La figure B.1-b montre la trajectoire résultante d'une pièce parallélépipédique.

	$\frac{d\Gamma^-}{ds}$ [1]	$\frac{d\Gamma^+}{ds}$ [1]	Commentaires
A		1 0 0 0 0 0	Tangence selon $\vec{x}_A$
B	0 0 -1 0 0 0		Continuité des tangentes selon $-\vec{z}_B$.
C	1 0 0 0 0 0		Tangence selon $\vec{x}_C$

En changeant l'amplitude des valeurs des tangentes on peut déformer localement les trajectoires. Cette possibilité est illustrée sur la trajectoire de la figure B.1-c pour laquelle la tangente à droite $\frac{d\Gamma^+}{ds}(A)$ est prise égale à 3 0 0 0 0 0. La trajectoire obtenue est « plus tendue » que la précédente. On peut exploiter cette propriété pour générer des mouvements tangents à des obstacles.

[1] L'exposant + ou - désigne la dérivée à droite ou gauche.

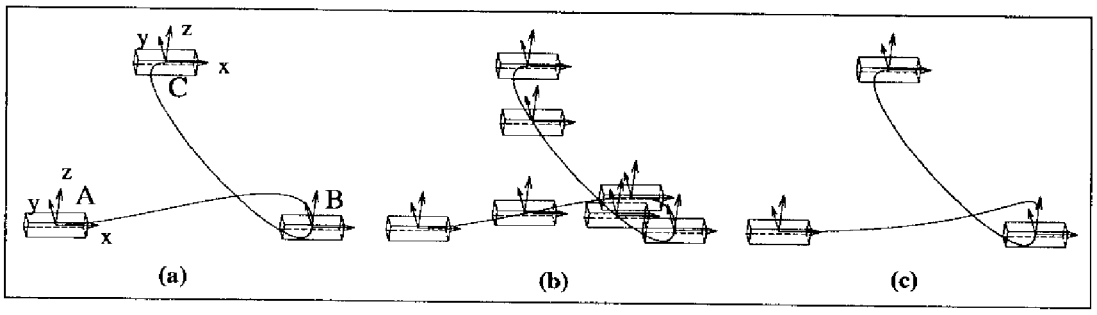

Figure B.1: Trajectoire cartésienne passant par trois points, à orientation constante.

Trajectoire avec gestion des orientations

Pour générer des mouvements fins ou à proximité d'obstacles, il est nécessaire de gérer les orientations des pièces en mouvement.

Dans l'exemple suivant, on souhaite déplacer un objet d'une position A vers une position B en imposant des contraintes sur les orientations (figure B.2-a).

Pour quitter le logement P, la pièce mobile doit tout d'abord être déplacée, à orientation constante, en translation selon $\vec{x}_A$. D'autre part, l'utilisateur souhaite que la pièce atteigne la configuration finale B avec une rotation tangentielle autour de $\vec{x}_B$.

Le tableau suivant décrits les conditions de tangence appliquées pour obtenir la trajectoire B.2-b. On observe sur cet exemple la «répartition» des contributions de translation et d'orientation. La translation est prédominante dans la première partie de la trajectoire tandis que la rotation est distribuée en fin de parcours.

Cette propriété s'exprime simplement en spécifiant des tangentes d'amplitude convenable aux points de contrôle.

	$\frac{d\Gamma}{ds}^-$	$\frac{d\Gamma}{ds}^+$	Commentaires
A		3 0 0 0 0 0	Tangence selon $\vec{x}_A$
B	1 0 0 0.1745 0 0		Tangence en translation et rotation selon $\vec{x}_B$

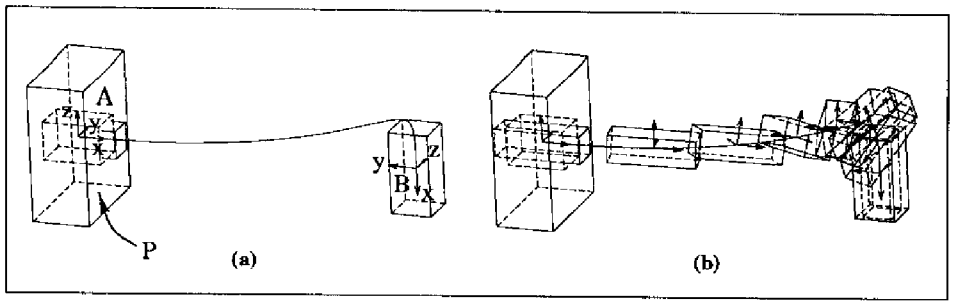

Figure B.2: Trajectoire cartésienne pilotée en orientation entre deux points de contrôle.

B.2 Génération de mouvements sur trajectoire imposée

L'exécution d'une trajectoire de transfert par un manipulateur nécessite l'élaboration d'une loi définissant le mouvement le long de cette trajectoire. La difficulté majeure consiste à générer et coordonner des lois de vitesse pour chaque actionneur compatibles avec les limites physiques de ces derniers.

Pour la génération de mouvements, la plupart des systèmes emploient des lois de vitesse de type trapèze. Avec de tels schémas de contrôle, les mouvements de grande amplitude sur des trajectoires doivent soit être réalisés à vitesse peu élevée, soit comporter une succession de segments point à point enchaînés à vitesse nulle.

L'objectif poursuivi est l'obtention de mouvements mieux enchaînés et plus rapides. Pour cela, il faut engendrer des lois de vitesse coordonnées et respectant les limitations des actionneurs en fonction des caractéristiques géométriques locales de la trajectoire suivie.

Notre intention n'est pas d'appréhender ce problème de commande en temps minimal sous sa forme la plus générale, notamment avec une prise en compte explicite de la dynamique du manipulateur [FOURQUET 90]. Nous avons limité notre ambition à la recherche d'une solution proche du temps optimal au moyen d'un algorithme de complexité modérée, en vue d'une exploitation efficace par le système de commande d'un robot manipulateur.

Nous présentons ci-après le point de départ des développements que nous avons réalisés en direction de cet objectif.

Transformation vers l'espace articulaire

Soit Γ_C une trajectoire de l'espace cartésien que doit suivre (en position et orientation) l'organe terminal d'un manipulateur. On suppose que l'on dispose de la trajectoire de l'espace articulaire Γ_Q transformation de Γ_C par le modèle géométrique inverse du robot. Γ_Q, paramétrée par un paramètre d'abscisse curviligne s, pourra être obtenue en interpolant des points de Γ_C tranformés au préalable vers l'espace articulaire. Cette trajectoire sera de la forme :

$$\Gamma_Q(s) = [\ q_1(s) \quad \dots \quad q_{nq}(s)\ ]^T$$

où nq représente le nombre d'actionneurs du robot et les fonctions q_i sont des cubiques.

Les conditions en vitesse et accélération à satisfaire

Générer un mouvement sur cette trajectoire donnée revient à rechercher une fonction s dépendante du temps :

$$\begin{array}{l} s: \ \mathbb{R} \to \mathbb{R} \\ \quad t \mapsto s(t) \end{array} \quad \text{telle que} \quad \begin{array}{l} \forall i \in [1, nq] \\ \forall i \in [1, nq] \end{array} \left| \begin{array}{l} \dfrac{dq_i}{dt}(s(t)) \\ \dfrac{d^2 q_i}{dt^2}(s(t)) \end{array} \right| \begin{array}{l} \leq \dot{q}_{imax} \quad \text{cont. en vitesse} \\ \leq \ddot{q}_{imax} \quad \text{cont. en accélération} \end{array} \tag{B.2}$$

Les $\dot{q}_{imax}$ et $\ddot{q}_{imax}$ désignent respectivement, la vitesse et l'accélération maximum admises par l'articulation i du manipulateur.

Les contraintes (B.2) peuvent être réécrites en faisant apparaître les dérivées $\frac{dq_i}{ds}$ et $\frac{d^2q_i}{ds^2}$ qui sont calculables en tout point s et les dérivées $\dot{s}$ et $\ddot{s}$.

La fonction $s(t)$ est alors recherchée sous la forme d'un polynôme de degré deux qui doit satisfaire, pour les nq composantes de la trajectoire, les conditions précédentes sur $\dot{s}$ et $\ddot{s}$.

Références bibliographiques

[ANGELES 89] J. ANGELES et O. MA. *QUADMIN: an integrated package for constrained nonlinear least-square problems.* ASME International Computers in Engineering Conference, pages 247–252, Anaheim (California), July, 30 1989.

[AUT 89] Autodesk Inc. *AutoCAD LT: Manuel d'utilisation*, 1989.

[BARANGER 91] J. BARANGER. Analyse numérique, chapitre 1. Hermann, 1991.

[BOEKE 91] M. BOEKE, E. AUST et G.F. SCHULTHEISS. *Underwater operation of a 6-axes robot based on offline programming and graphical simulation.* 91 ICAR International Conference On Advanced Robotics, pages 1342–1347, 1991.

[BOISSONNAT 88] J.D. BOISSONNAT. *Complexité géométrique et robotique.* Techniques de la Robotique - Perception et planification. Hermès, 1988.

[BORREL 85] P. BORREL, E. DOMBRE, A LIÉGOIS, F. BERNARD et D. BOURCIER. *La fonction robotique dans le système CATIA.* État de la Robotique en France. Hermès, 1985.

[CAMERON 89] S. CAMERON et J. AYLETT. *Robmod User Guide.* Rapport technique, University of Edinburgh, Forest Hill - Edinburgh, November 1989.

[CELAYA 92] E. CELAYA. *LMF : a program for positioning objects using geometric relationships.* 92'International Conference on Applications of Artificial Intelligence in Engineering, 1992.

[CIARLET 90] P.G. CIARLET. Introduction à l'analyse numérique matricielle et à l'optimisation, chapitre 4, 7. Masson, 1990.

[CLA 89] Claris Corporation. *Claris CAD : Guide de l'utilisateur*, 1989.

[DE SCHUTTER 92] J. DE SCHUTTER et H. BRUYNINCKX. *Model-Based Specification and Execution of Compliant Motion.* 1992 IEEE International Conference on Robotics and Automation, Nice, France, mai 1992. Tutorial.

[DEGALLAIX 93] B. DEGALLAIX. *Une Architecture pour la Téléprogrammation au Niveau Tâche d'un Robot Mobile.* Thèse de l'Université Paul Sabatier, LAAS/CNRS - Toulouse, France, janvier 1993.

[DEMAILLY 91] J.P. DEMAILLY. Analyse numérique et équations différentielles. P.U.G., 1991.

[DOMBRE 88] E. DOMBRE et W. KHALIL. Modélisation et commandes des robots. Hermès, 1988.

[ESPIAU 88] B. ESPIAU et C. SAMSON. *Le point sur la commande proximétrique.* Techniques de la Robotique - Perception et planification. Hermès, 1988.

[FAUGERAS 88] O. D. FAUGERAS. *Quelques pas vers la vision artificielle en trois dimensions.* Techniques de la Robotique - Perception et planification. Hermès, 1988.

[FAVERJON 88] B. FAVERJON et P. TOURNASSOUD. Techniques de la Robotique - Perception et planification, chapitre 7. Hermès, 1988.

[FAVERJON 89] B. FAVERJON et P. TOURNASSOUD. *Motion planning for manipulators in complex environments.* J.D. BOISSONNAT et J.P. LAUMOND, editeurs, Lecture Notes in Computer Science - Geometry and Robotics, pages 87–115. Springer Verlag, 1989.

[FOURNIER 80] A. FOURNIER. *Génération de mouvements en robotique. Applications des inverses généralisées et des pseudo-inverses.* Thèse d'État, USTL, Montpellier, Avril 1980.

[FOURQUET 90] J.Y. FOURQUET. *Mouvements en temps minimal pour les robots manipulateurs en tenant compte de leur dynamique non linéaire.* Thèse de l'Université Paul Sabatier, LAAS/CNRS - Toulouse, France, décembre 1990.

[GIRALT 89] G. GIRALT, R. ALAMI et R. CHATILA. *Autonomy versus Teleoperation for Intervention Robots? A Case for Task Level Teleprogramming.* Intelligent Autonomous Systems-2, Amsterdam, Netherlands, December 1989.

[GIRAUD 91] A. GIRAUD et T. TRAN. Kinematic modeling of tasks under constraints. application to fine motion and calibration, chapitre 2, pages 89–98. Springer-Verlag, 1991.

[GOLUB 89] G.H. GOLUB et C.F. VAN LOAN. Matrix computations. The John Hopkins University Press, 1989.

[GOMEZ 95] CL. GOMEZ, B. SALVY et P.ZIMMERMANN. Calcul formel : Mode d'emploi - Exemples en maple, chapitre 5-2. Masson, 1995.

[GONDRAN 90] M. GONDRAN et M. MINOUX. Graphes et algorithmes. Eyrolles, 1990.

[GORLA 84] B. GORLA et M. RENAUD. Modèle des robots-manipulateurs: application à leur commande. Cepadues, 1984.

[GUINOT 92] J.C. GUINOT. *Cours de Mécanique.* DEA de Robotique - ENSAM - Univ. Paris 6, 1992.

[GUMOWSKI 70] I. GUMOWSKI et C. MIRA. L'Optimisation - la Théorie et ses Problèmes. Dunod, 1970.

[HAYWARD 88] V. HAYWARD, L. DANESHMARD et S. HAYATI. *An overview of KALI: asystem to program and control cooperative manipulators.* 88 ICAR Fourth International Conference On Advanced Robotics, 1988.

[HEYDON 94] A. HEYDON et G. NELSON. *The Juno-2 Constraint-Based Drawing Editor.* Rapport technique, Digital Equipment - Systems Research Center, décembre 1994.

[HIRZINGER 93] G. HIRZINGER. *ROTEX - The First Robot in Space.* 93 ICAR Sixth International Conference On Advanced Robotics, pages 9–33, 1993.

[HOLLERBACH 89] J. HOLLERBACH. *A survey of kinematic calibration.* O. KATHIB, J. CRAIG et T. LOZANO-PÉREZ, editeurs, Robotics Review 1, pages 207–242. MIT Press, Cambridge, 1989.

[IRIGOYEN 86] X. IRIGOYEN. *Commande en position et en force d'un robot d'assemblage.* Thèse de l'Université Paul Sabatier, LAAS/CNRS - Toulouse, France, octobre 1986.

[KACZOR 93] M.Y. KACZOR, C. LAUGIER et E. MAZER. *TELE-ACT: a task level teleprogramming system.* IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 310–313, Yokohama (Japan), July, 26-30 1993.

[KIM 93] W.S. KIM. *Graphical operator interface for space telerobotics.* IEEE International Conference on Robotics and Automation, pages Vol3, 761–768, Atlanta (USA), may 2-6 1993.

[LATOMBE 91] J.C. LATOMBE. Robot Motion Planning. Kluwer Academic Publishers, 1991.

[LAUGIER 85] C. LAUGIER et J. TROCCAZ. *SHARP : a system for automatic programming of manipulation robots.* O. FAUGERAS et G. GIRALT, editeurs, Robotics Research : the Third International Symposium. MIT Press, Cambridge, October 1985.

[LIÉGEOIS 84] A. LIÉGEOIS. Les robots - Analyse des performances et CAO, volume 7. Hermès, 1984.

[MADDALENA 92] D. MADDALENA et M. ZAMPATO. *Model based pose evaluation of offshore structure nodes.* International Advanced Robotics Programme - 4th workshop on underwater robotics, Genova, Italy, novembre 1992.

[MASON 82] M. T. MASON. *Compliant motion.* M. BRADY ET AL., editeur, Robot Motion-Planning and Control. The MIT Press, 1982.

[MAT 92] The Math Works. *MATLAB - Reference Guide*, 1992.

[MAZER 87] E. MAZER. *HANDEY : un modèle de planificateur pour la programmation automatique des robots.* Thèse d'État, Institut National Polytechnique de Grenoble, décembre 1987.

[NOR 91] Northern Digital Inc., Waterloo, Ontario - Canada. *OptoTrak : 3D Motion Measurement System*, October 1991.

[NUM 91] Numerical Algorithms Group. *NAG Fortran concise Reference*, 1991.

[PAUL 92] R.P. PAUL, T. LINDSAY et C. SAYERS. *Time delay insensitive teleoperation.* IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 247–254, Rayleigh (USA), July 7-10 1992.

[PRESS 88] W.H. PRESS, B.P. FLANNERY, S.A. TEUKOLSKY et W.T. VETTERLING. Numerical Recipes in C. Cambridge University Press, 1988.

[RENAUD 95] M. RENAUD. *Communication personnelle*, 1995.

[ROTH 94] B. ROTH. *Computational advances in robot kinematics.* J. LENARCIC et B. RAVANI, editeurs, Advances in Robot Kinematics and Computational Geometry. Kluwer Academic Publishers, 1994.

[SAMSON 90] C. SAMSON, B. ESPIAU et M. LE BORGNE. Robot control : the task function approach. Oxford University Press, 1990.

[SAYERS 94] C. SAYERS, M. STEIN, A. LAI et R.P. PAUL. *Teleprogramming to perform sophisticated underwater manipulative tasks using acoustic communications.* IEEE OCEANS, Brest, France, septembre 1994.

[THIBOUT 94] C. THIBOUT, P. EVEN et R. FOURNIER. *Virtual reality for teleoperated robot control.* ORIA'94 - From telepresence towards virtual reality, pages 131–139, Marseille - France, décembre 1994.

[TRAN 90] T. TRAN. *Programmation graphique interactive de tâches non répétitives de manipulation au contact.* Thèse de l'Université Paul Sabatier, LAAS/CNRS - Toulouse, France, novembre 1990.

[VIOLERO 91] P. VIOLERO. *SPARA : un système de programmation automatique de tâche de manipulation robotisée.* Thèse de l'Université Paul Sabatier, LAAS/CNRS - Toulouse, France, novembre 1991.

Table des matières

Introduction 1

1 Présentation générale 3

1.1 Introduction 3

1.1.1 La robotique d'intervention 3

1.1.2 Exécution de tâches sur sites distants 3

1.1.3 Le concept d'autonomie en robotique 4

1.2 Stratégies de génération de trajectoire en robotique de manipulation 5

1.2.1 Utilisation d'un modèle de réalité virtuelle 5

1.2.2 Les méthodes de planification de trajectoires 7

1.2.3 La spécification par contraintes géométriques dans l'espace de la tâche 8

1.3 Les grandes lignes de l'approche proposée 10

1.3.1 Modélisation des tâches 10

1.3.2 La méthode de résolution 10

1.3.3 Applications 11

1.3.4 Aperçu sur l'environnement de programmation graphique 11

2 Méthodologie de spécification de tâches 13

2.1 Modélisation des mécanismes 13

2.2 Définitions de base 15

2.3 Modélisation graphique d'un système mécanique et de son environnement . . 16

2.3.1 Description topologique - Corps polyédriques 16

2.3.2 Positionnement des corps - Graphe de repères 17

2.3.3 Scène graphique - Visualisation 17

2.4 Spécification par contraintes de contact 18

2.4.1 Concept de demi-contact 19

2.4.2 Concept de lien - Expression des mobilités relatives 20

2.4.3 Concept de forme 22
2.4.4 Concept d'articulation 25
2.4.5 Concept de contact 26
2.5 Concept de problème de mouvement 29
2.6 Concept de problème de calibration 30
2.6.1 Concept de mesure 31
2.6.2 Élaboration d'un problème de calibration 31
2.7 Forme générale du résultat de la modélisation 33
2.7.1 Graphe cinématique 33
2.7.2 Obtention des équations de contraintes 33
2.7.3 Forme du résultat de la spécification de tâches 35
2.8 Conclusion 36

3 Méthode de résolution **37**
3.1 Introduction 37
3.2 Résolution de systèmes d'équations non linéaires 38
3.2.1 Principe général 38
3.2.2 Convergence globale de l'algorithme itératif (3.3) 39
3.2.3 Inversion de $A = \frac{\partial K}{\partial x}$ 40
3.2.4 Résolution de contraintes égalités 41
3.2.5 Traitement de problèmes d'optimisation 41
3.3 Formulation initiale des problèmes 46
3.4 Application à la résolution itérative d'un problème de mouvement 47
3.5 Manipulation hybride numérique/formelle de systèmes d'équations linéaires . 47
3.5.1 Conventions d'écriture 48
3.5.2 Origine du besoin de manipulation formelle des systèmes d'équations . 49
3.5.3 Zéros structuraux et conjoncturels 50
3.6 Principe fondamental de résolution: élimination de Gauss 51
3.6.1 Élimination des zéros structuraux 52
3.6.2 Choix des variables et recherche du meilleur pivot 52
3.6.3 Substitution directe 53
3.6.4 Propagation inverse 53
3.6.5 Analyse de la situation avant résolution 54
3.6.6 Résolution finale 55
3.7 Traitement du problème général 55

3.7.1 Conditionnement préliminaire des équations 56
3.7.2 Résolution des contraintes égalités 57
3.7.3 Résolution des contraintes inégalités 59
3.7.4 Calcul itératif et vitesse de convergence 62
3.8 Extraction de matrices de sensibilité 63
3.8.1 Sensibilité par rapport aux contraintes 63
3.8.2 Sensibilité par rapport à des paramètres 63
3.9 Détection et gestion des singularités 64
3.10 Un exemple simple d'utilisation de l'approche 64
3.11 Implantation de la méthode de résolution 67
3.12 Conclusion 67

4 Application à l'exécution de tâches robotisées 69
4.1 Exploitation de l'environnement de simulation graphique 69
4.1.1 Présentation générale 69
4.1.2 Modalités d'utilisation 70
4.1.3 Déroulement d'une spécification 70
4.1.4 Mécanismes d'accès aux structures 71
4.1.5 Expression des contraintes par étiquetage 73
4.2 Mouvement de deux cubes 74
4.2.1 Traitement des contraintes géométriques impératives 75
4.2.2 Modification interactive de la spécification 76
4.3 Traitement de mécanismes comportant des boucles 77
4.4 Modélisation de systèmes robotiques 79
4.4.1 Approche globale 79
4.4.2 Approche par décomposition 80
4.4.3 Mise en œuvre pratique de l'approche globale 81
4.4.4 Modèle d'un robot manipulateur SCEMI et de sa pince 82
4.5 Mouvement avec évitement local d'obstacles 84
4.5.1 Exemple 1 : insertion de deux pièces 84
4.5.2 Exemple 2 : gestion des butées sur un manipulateur redondant 85
4.6 Mouvement au contact d'un manipulateur vers un cylindre 87
4.7 Mouvement coordonné de deux manipulateurs 90
4.8 Problèmes de calibration 92
4.8.1 Obtention de mesures 93

4.8.2 Exemple : calibration d'un cylindre par contact 93
4.8.3 Étude de sensibilité par rapport à la réalisation des contacts 96
4.9 Modélisation de capteurs 97
4.9.1 Modélisation d'un capteur de type caméra 97
4.9.2 Intégration de données capteurs 101
4.10 Conclusion 101

5 Expérimentation **103**
5.1 Une expérimentation canonique 103
5.1.1 Site expérimental 104
5.1.2 Spécification de la tâche 105
5.1.3 Calibration de l'environnement 107
5.1.4 Mise en œuvre des stratégies de calibration 108
5.1.5 Résultats 109
5.1.6 Discussion 111
5.2 Perspectives d'utilisation dans un contexte de manipulation distante 113
5.3 Conclusion 115

Conclusion **117**

A Relation entre vitesse angulaire et dérivée du vecteur rotation **119**

B Génération de trajectoires et de lois de mouvements **121**
B.1 Génération de trajectoires dans l'espace cartésien 121
B.1.1 Modélisation 121
B.1.2 Utilisation 122
B.2 Génération de mouvements sur trajectoire imposée 124

Références bibliographiques **125**

Table des matières **129**

Liste des figures **133**

Liste des figures

1.1 Principaux modules de l'environnement de programmation de robots. 11

2.1 Mécanisme à deux degrés de liberté. 14
2.2 Graphe du mécanisme. 14
2.3 Graphe étendu. 14
2.4 Situation relative de deux corps. 16
2.5 Architecture du modeleur 3D. 17
2.6 Modèle graphique 3D du manipulateur plan. 18
2.7 Nécessité de repères de travail pour la description de contacts. 19
2.8 Arborescence de demi-contacts. 20
2.9 Structure d'un demi-contact. 20
2.10 Lien ordinaire et à modèle géométrique. 21
2.11 Organisation d'un lien. 21
2.12 Forme cylindre 23
2.13 Forme cône 24
2.14 Liaison articulaire et graphe de repères. 25
2.15 Contact arête sur arête. 26
2.16 Description d'une hélice. 28
2.17 Structuration d'un problème de mouvement. 29
2.18 Problème de mouvement pour le manipulateur à deux ddl. 30
2.19 Genèse d'un problème de calibration. 32
2.20 Calibration à partir de mesures inhomogènes. 33
2.21 Cycle cinématique. 34

3.1 Recherche de la meilleure inconnue et du meilleur pivot. 53
3.2 Arbre de résolution 55
3.3 Résolution des contraintes F et G 58
3.4 Résolution des contraintes inégalités H. 59
3.5 Exemple plan illustrant le traitement de contraintes inégalités. 61
3.6 Intersection de deux paraboles. 65

4.1 Contact repère sur cylindre. 72
4.2 Arborescence de fenêtres pour la revue des liens de contact. 72
4.3 Noms volatiles de demi-contacts. 73
4.4 Positionnement relatif de deux arêtes. 75

4.5 Mouvement avec contraintes angulaire et de distance. 77
4.6 Mouvement avec contrainte angulaire seule. 77
4.7 Mécanisme arborescent. 78
4.8 Contraintes de fermeture de boucle. 78
4.9 Spécification complète. 78
4.10 Hiérarchie du concept de robot. 81
4.11 Référentiels attachés à un robot. 81
4.12 Modélisation du robot et de la pince. 83
4.13 Tâche d'insertion d'une équerre. 85
4.14 Robot Mitsubishi PA–10. 86
4.15 Gestion des butées sur un manipulateur à sept degrés de liberté. 86
4.16 Modélisation de l'environnement de travail. 88
4.17 Contrainte de contact avec le cylindre. 89
4.18 Rejoindre un point donné du cylindre. 89
4.19 Mouvement vers le point le plus proche. 89
4.20 Mouvement vers un point spécifié. 89
4.21 Scénario complet d'une tâche d'insertion réalisée par deux manipulateurs. . . 91
4.22 Les liaisons sujettes à calibration. 94
4.23 Ensemble de mesures. 94
4.24 Résultat de la calibration. 94
4.25 Modèle du sténopé. 98
4.26 Modélisation d'une caméra par contraintes. 98
4.27 Spécification d'un contact projectif. 99
4.28 Structuration du contact projectif. 99
4.29 Position initiale du manipulateur. 100
4.30 Déplacement référencé vision d'un manipulateur. 100

5.1 Robot SCEMI et tubes à inspecter. 104
5.2 Positionnement sur l'intersection de deux surfaces. 106
5.3 Progression le long d'une courbe de contrainte. 107
5.4 Contraintes pour la calibration de la table et du palpeur. 107
5.5 Transfert des informations entre problème de mouvement et mesure. 109
5.6 Un cycle de mouvement pour la calibration de la table et du palpeur. 110
5.7 Mouvement au contact pour la calibration du premier cylindre 110
5.8 Déroulement de la tâche de suivi de joint. 111
5.9 Architecture pour la téléprogrammation de manipulateurs distants. 114

B.1 Trajectoire cartésienne passant par trois points, à orientation constante. . . . 123
B.2 Trajectoire cartésienne pilotée en orientation entre deux points de contrôle. . 123

Spécification et résolution de tâches de manipulation complexes. Application à la téléprogrammation de robots distants.

Les travaux présentés dans cette thèse s'inscrivent dans le contexte de la robotique de manipulation distante où les possibilités de communication entre l'opérateur et le site robotisé sont réduites. Pour accroître l'autonomie de fonctionnement du manipulateur distant, il faut pouvoir raisonner sur un modèle du robot et de son environnement de travail. En raison de la connaissance imprécise de ce dernier, il est indispensable de disposer de moyens d'identification des positions et de la taille des objets en interaction avec le robot. De plus, le caractère non répétitif et la complexité des tâches envisagées nécessitent l'emploi d'un système rapidement reconfigurable et susceptible de traiter des problèmes suffisamment généraux.
Dans cette perspective, l'approche proposée de spécification de tâches par contraintes autorise une description des actions à effectuer en termes de mobilités et de contraintes de contact à établir ou maintenir entre certains corps. Les relations de contraintes résultantes sont alors traitées par un module de résolution basé sur un schéma numérique itératif de type Newton-Raphson. Un niveau important de robustesse a été atteint grâce à une gestion dynamique du conditionnement des équations qui permet un choix automatique du scénario de résolution.
Les applications de l'approche incluent en premier lieu la génération de trajectoires complexes et continues pour des systèmes robotisés quelconques. La mise en place de contraintes peut aussi être exploitée pour accomplir, localement, des évitements d'obstacles. Enfin, cette démarche se prête également à la résolution de tâches de calibration des objets présents dans l'univers du robot. Cette caractéristique est essentielle dans le contexte de manipulation distante où les modèles sont mal connus *a priori*. Finalement, la présentation d'un scénario expérimental de suivi de joint de soudure, mis en œuvre sur un manipulateur réel, nous permet de formuler une proposition pour l'exécution de tâches robotisées sur sites distants.

Mots clés : *Robotique de manipulation - Programmation graphique - Génération de trajectoires - Calibration - Résolution numérique d'équations non linéaires.*

Specification and resolution of complex manipulation tasks. Application to remote robots teleprogramming.

The work presented in this thesis comes within the scope of remote manipulation with restricted communication properties between the operator and the remote site. This context renders traditional teleoperation infeasible. To enhance the autonomy of the remote manipulator, it is necessary to reason on a model of the robot and its workspace. However, discrepancies between the real world and its representation require calibration capabilities to identify both position and size of objects interacting with the robot. Moreover, the non-repetitiveness and complexity of the tasks demand that the specification system remains easy to re-program and capable of treating a wide range of problems.
The proposed constraint-based approach permits the specification of complex manipulation tasks in which tasks' objectives are expressed in terms of mobilities and contact relationships to achieve or maintain between parts. The resulting constraint relationships are then treated by a numerical solver based on a Newton-Raphson scheme. An enhanced robustness has been achieved through a dynamic management of equations' conditioning. This enables the system to choose automatically for the most appropriate resolution scenario.
The first main class of applications is complex motion generation for any kind of robotic mechanisms possibly including redundancy. Constraints setting can also be exploited to realize local obstacle avoidance. The proposed approach makes it possible to deal with calibration tasks within the same framework. This constitutes an essential feature in the context of remote manipulation where models are unprecisely known. Lastly, a weld line inspection experiment performed on a real manipulator allows us to put forward a strategy for robotic task performance at a remote location.

Keywords : *Manipulation - Graphical Programming - Motion Generation - Calibration - Numerical Resolution of Nonlinear Equations.*