

Traitement des incertitudes en programmation automatique des robots

C. Laugier

▶ **To cite this version:**

C. Laugier. Traitement des incertitudes en programmation automatique des robots. RR-0933, INRIA. 1988. inria-00077182

HAL Id: inria-00077182
https://inria.hal.science/inria-00077182

Submitted on 29 May 2006

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.

INRIA

UNITÉ DE RECHERCHE
INRIA-ROCQUENCOURT

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
BP 105
78153 Le Chesnay Cedex
France
Tél. (1) 39 63 55 11

Rapports de Recherche

N° 933

Programme 6

TRAITEMENT DES INCERTITUDES EN PROGRAMMATION AUTOMATIQUE DES ROBOTS

Christian LAUGIER

GROUPE DE RECHERCHE
GRENOBLE

Décembre 1988

RR_0933

2943

Traitement des incertitudes en programmation automatique des robots
Dealing with uncertainty in automatic robot programming

Christian LAUGIER*

Septembre 1988

Résumé

Les incertitudes sont à l'origine de très nombreux échecs survenant lors de l'exécution des programmes de manipulation. Il est donc essentiel de maîtriser au maximum ces incertitudes, si l'on veut synthétiser automatiquement des programmes de commande de robots aussi robustes que possible. Dans cet article, nous montrons au travers de l'analyse d'un exemple simple, qu'un programme de manipulation doit faire face à deux catégories d'incertitudes: celles qui peuvent être traitées localement par des stratégies appropriées, et celles qui nécessitent un traitement global. Nous présentons ensuite les approches utilisées dans le système SHARP (système de programmation automatique de robots développé au Laboratoire LIFIA) pour résoudre les deux problèmes correspondants: le problème de la planification des mouvements fins de montage, et celui de la vérification/correction de plans de manipulation.

Abstract

Geometric uncertainty may cause various failures during the execution of a robot control program. Avoiding such failures makes it necessary to reason about the effects of uncertainty in order to implement robusts strategies. In this paper, we first show using a simple exemple that a manipulation program has to be faced with to types of uncertainty: those that might be locally processed using appropriates sensor based strategies, and those that require a more global processing. Then, we briefly describe the approaches we have partly implemented in the SHARP system (SHARP is an automatic robot programming system currently under development at the LIFIA laboratory) for solving the two related problems: (1) how to automatically synthesize fine motions programs from a high level description of the task to perform, and (2) how to apply the verification/ correction scheme on a manipulation program that includes an explicit representation of position uncertainty.

*LIFIA/IMAG, 46 Avenue Felix Viallet, 38031 Grenoble cedex, France

PAPIER RECUPÉRÉ ET RECYCLÉ

1 Introduction:

1.1 Les incertitudes en robotique:

Un robot et son environnement de travail représentent un système mécanique souvent très complexe. Cette complexité rend impossible la construction d'un modèle théorique complet, qui permette de prévoir exactement le comportement de l'ensemble. C'est pourquoi les informaticiens utilisent des modèles approchés pour programmer les robots. Les écarts qui existent entre ces modèles et la réalité physique, proviennent essentiellement des erreurs engendrées par les calculs numériques, par les mouvements du robot, par les données transmises par les capteurs, par les systèmes d'alimentation et de positionnement des pièces mécaniques, et par les tolérances de fabrication.

Toutes ces erreurs ne sont jamais connues de manière exacte, mais il est en général possible d'obtenir une information sur leurs évolutions. Cette information consiste très souvent en un domaine de variation associé à chaque terme d'erreur; elle peut aussi être représentée sous la forme d'une loi mathématique symbolisant un comportement statistique des erreurs. Une telle information représente *l'incertitude* associée à la grandeur considérée.

Dans le contexte des systèmes de programmation de robots, les incertitudes considérées portent sur les données de position et sur les paramètres morphologiques associés aux objets manipulés et au robot. Dans la suite, nous ne traiterons explicitement que les *incertitudes de position.* Celles qui portent sur la forme des objets seront ignorées au niveau des calculs, du fait qu'elles sont sensées intervenir uniquement dans la spécification des *jeux de montage.*

Conceptuellement, l'exécution d'un programme de commande de robot peut être vue comme une succession d'actions de manipulation et/ou de perception, conduisant chacune à la réalisation d'un état particulier de l'ensemble robot-environnement. Si il n'y avait pas d'incertitude, chaque état de l'univers du robot pourrait être représenté sans ambiguïté par un ensemble de variables. Ces variables sont celles qui sont utilisées dans le programme pour définir les configurations du bras, les positions et les orientations des objets, ainsi que les données transmises par les capteurs (forces appliquées par l'effecteur en particulier). Dans la réalité, seules les valeurs nominales de ces variables et les incertitudes qui leur sont associées sont connues du système. On est donc amené à raisonner sur des *classes d'états*, bien que la description des actions du robot soit toujours réalisée à partir des *états nominaux.*

Lorsque les incertitudes sont inférieures à la précision requise par la tâche, tous les états qu'il est possible d'obtenir après exécution d'une action du programme réalisent la relation désirée. Il est dans ce cas inutile d'expliciter les classes d'états rencontrées, du fait qu'elles ne jouent aucun rôle dans le séquencement des opérations. Le programme peut alors être exécuté "en aveugle", c'est à dire sans utiliser les données transmises par les capteurs extéroceptifs du robot.

Dans la réalité, les erreurs de commande et de perception sont souvent supérieures à celles tolérées par la tâche. Une même action du robot peut alors conduire à des situations

physiques fonctionnellement différentes. Chacune de ces situations correspond à une classe d'états particulière, qui doit être identifiable à l'aide des moyens perceptifs du robot. Par exemple, une erreur de centrage du goujon lors de l'exécution d'une opération d'insertion cylindrique, peut conduire à heurter le chanfrein. La détection de cette situation à l'aide des capteurs de position et de force, aura pour effet de provoquer le lancement d'une action corrective dirigée par la composante horizontale de la force. Le programme de manipulation peut alors être vu comme un graphe, dans lequel chaque sommet représente une classe d'états identifiable, et chaque arc définit une occurrence de mouvement. Une exécution particulière du programme est alors représentée par un chemin dans ce graphe, et le séquencement des opérations du robot est guidé par l'identification des classes d'états rencontrées.

Tout le travail du programmeur ou du système de planification consiste alors à construire ce type de graphe, sur la base d'une analyse minutieuse des incertitudes liées à la tâche à exécuter. Il doit pour cela:

- Déterminer les différentes sources d'incertitudes, puis prévoir de quelle manière celles-ci peuvent remettre en cause le succès des actions menées par le robot.
- S'affranchir au mieux de ces incertitudes en choisisant des stratégies appropriées, en utilisant des mouvements compliants, en exploitant des données sensorielles pour réduire certaines composantes d'incertitudes, en vérifiant les positions atteintes, en intégrant des actions correctives...

Le but de cet article est de montrer de quelle manière il est possible de résoudre ce problème dans le cadre d'un système de programmation automatique de robots.

1.2 Principe de modélisation des incertitudes de position:

L'approche généralement utilisée en robotique d'assemblage consiste à définir l'incertitude sur un paramètre, comme *l'ensemble des erreurs* qui peuvent entâcher la valeur de ce paramètre. Par exemple, l'incertitude de position associée à un robot cartésien possédant trois axes, sera définie par le pavé $[-\Delta x\ +\Delta x]\times[-\Delta y\ +\Delta y]\times[-\Delta z\ +\Delta z]$, centré sur la position nominale P considérée. Ce pavé est alors appelé "volume d'incertitude" associé à P. Ce mode de représentation a été à la base de travaux ayant pour objectif d'évaluer les incertitudes de position en divers points d'un programme de manipulation [Taylor 76] [Brooks 82] [Lozano-Perez, Brooks 85]. Il a également été utilisé par Brooks [Brooks 84] pour analyser les incertitudes associées aux déplacements d'un robot mobile.

Cette représentation ensembliste est attrayante dans son principe, car elle met en oeuvre des procédures de calculs conceptuellement simples: toutes les opérations sur les incertitudes (composition, comparaison ...) sont traduites en termes de manipulations d'ensembles simples (union, intersection, projection ...). L'inconvénient majeur de cette approche réside dans la nature des représentations utilisées, qui conduisent à raisonner en permanence sur les erreurs maximales. Il est de ce fait impossible de prendre en compte certains phénomènes de compensations statistiques des erreurs. Cette méthode est cependant la seule possible, lorsque l'on veut s'assurer de la faisabilité d'une séquence

d'opérations portant sur des objets physiques.

Une autre approche souvent utilisée pour estimer la position d'un robot mobile [Chatila, Laumond 85] ou pour résoudre des problèmes de localisation visuelle d'objets [Faugeras, Hebert 86] [Bolle, Cooper 86] [Durrant-Whyte 87], consiste à associer une loi de probabilité aux grandeurs physiques sujettes à incertitude. On représentera par exemple la variation d'une erreur par une loi gaussienne normale, ayant pour valeur moyenne la valeur théorique du paramètre. Smith [Smith et al. 86] a ainsi développé un formalisme mathématique de type probabiliste, pour représenter et manipuler les incertitudes liées à un univers structuré d'objets. Ce formalisme utilise des matrices de covariance pour représenter les relations de dépendance des paramètres, afin de pouvoir combiner et propager les incertitudes de position. Il répond assez bien à la problématique de la robotique mobile. Son extension à l'univers de la manipulation pose cependant quelques problèmes théoriques dûs à la complexité des situations rencontrées, en particulier: traitement des contacts multiples et estimation des répartitions statistiques d'erreurs qu'il est possible de combiner pour satisfaire une contrainte imposée par la tâche.

Compte tenu des limitations théoriques de l'approche statistique et de la relative simplicité des opérations mises en jeu par l'autre méthode, nous avons choisi de représenter les incertitudes de position par des ensembles simples. Ce choix a été en grande partie dicté par la nature du problème traité, qui consiste avant tout à s'assurer de la validité théorique des séquences d'actions engendrées par le système. Il se justifie par le petit nombre d'opérations de propagation qu'il est nécessaire d'exécuter, pour calculer les modifications d'incertitudes induites par les actions du robot (chaque opération de montage du plan de manipulation étant réalisée en quelques mouvements). Cette propriété permet d'éviter une accumulation trop importante des erreurs, lors de la "fusion" des domaines d'incertitudes. L'importance de ces erreurs est de plus largement diminuée par la présence de nombreux contacts, qui contribuent à réduire progressivement les domaines d'incertitudes obtenus.

Il convient de noter que le choix d'une représentation ensembliste conditionne uniquement les procédés de calculs mis en jeu. Il n'affecte en aucune manière les principes généraux sur lesquels reposent l'approche que nous avons développé.

1.3 La présentation:

L'article présenté est structuré en trois parties. Nous analyserons tout d'abord dans le paragraphe 2 le rôle joué par les incertitudes dans le processus de programmation d'un robot. Nous nous appuierons pour cela sur un exemple simple, apte à mettre en valeur les deux formes de traitement qu'il convient d'appliquer pour prendre en compte les diverses contraintes d'incertitudes. Une étude bibliographique des différentes techniques développées complétera cette partie générale.

Nous présenterons ensuite dans le paragraphe 3 de quelle manière nous avons abordé le problème de la planification des mouvements fins de montage. Le but recherché est alors de s'affranchir au moyen de stratégies appropriées, des incertitudes qui s'opposent

"localement" à la réalisation d'une relation d'assemblage. Nous insisterons dans cette partie sur le rôle prépondérant attribué aux contacts.

Nous aborderons finalement le problème de la vérification/correction de plans dans le paragraphe 4. Il s'agit alors de traiter les interdépendances "globales", induites par les erreurs accumulées au cours de diverses séquences d'actions du robot. Nous développerons dans cette partie les différents aspects de la propagation des contraintes d'incertitudes.

2 Intégration des incertitudes dans le processus de programmation d'un robot:

2.1 Approche traditionnelle et programmation automatique:

Les systèmes traditionnels de programmation de robots reposent sur des outils informatiques plus ou moins sophistiqués, conçus dans le but de servir d'interface entre le programmeur et les différentes fonctionnalités du robot [Latombe, Laugier 85]. Le rôle du programmeur consiste alors à décrire explicitement l'ensemble des actions et des opérations du robot, qui sont nécessaires à l'exécution de la tâche. Il doit pour cela faire face à de nombreux problèmes géométriques et physiques, afin d'éviter les collisions et les incidents mécaniques. Il doit en particulier prévoir dans quelle mesure les opérations programmées mèneront au résultat attendu, compte tenu des erreurs accumulées par les systèmes de commande et de perception. Dans la pratique, le programmeur n'a qu'une connaissance empirique et très qualitative de ces phénomènes. Il procède alors par corrections successives de son programme, afin de s'affranchir au mieux des contraintes d'incertitudes. Chaque correction est généralement provoquée par un échec détecté lors d'une tentative d'exécution du programme. Elle repose alors entièrement sur l'expérience du programmeur.

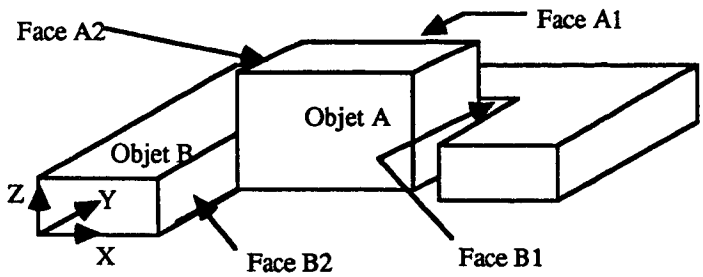
En programmation automatique, ce travail est à la charge du système. Il dispose pour cela d'une description symbolique de la tâche, réalisée par des assertions du type: *placer le cube A dans l'angle C de l'objet B.* Cette description est complétée par un modèle du robot et de son environnement, incluant une représentation des jeux de montage et des incertitudes de commande et de perception. Le rôle du système est alors de convertir la description symbolique de la tâche en un programme de commande de robot permettant d'exécuter cette tâche. Il doit pour cela résoudre trois classes de problèmes: la planification globale de la tâche, la planification des actions nominales du robot, et la modification du plan nominal afin de l'adapter à la réalité physique.

Les différents aspects de ces problèmes sont présentés dans [Laugier 87]. Ils sont introduits de manière intuitive dans le paragraphe qui suit, en s'appuyant sur un exemple simple. Cet exemple sera utilisé pour mettre en évidence le rôle joué par les incertitudes au niveau des différentes étapes de planification.

2.2 Analyse d'un exemple de planification:

2.2.1 Description de la tâche:

L'exemple traité consiste à positionner un bloc parallélépipèdique dans l'angle d'une pièce coudée (cf. figure 1). Il est développé dans un langage dérivé de celui utilisé dans le système SHARP (système de programmation automatique de robots du LIFIA [Laugier, Troccaz 85] [Laugier 87]). Le formalisme géométrique utilisé pour spécifier la tâche est celui du langage LM-GEO [Mazer 83]. Le montage considéré peut alors être décrit comme suit:



Description symbolique: *Dessous-A CONTRE(0) Dessus-table*
Face A1 CONTRE(0) Face B1
Face A2 CONTRE(0) Face B2

Figure 1: Exemple d'un "assemblage" simple.

MONTER A SUR B TEL-QUE $(R_1 \wedge R_2 \wedge R_3)$
avec:
$R_1 = dessous\text{-}A \;\; CONTRE(0) \;\; dessus\text{-}table$
$R_2 = face\, A_1 \;\; CONTRE(0) \;\; face\, B_1$
$R_3 = face\, A_2 \;\; CONTRE(0) \;\; face\, B_2$

2.2.2 Planification globale de la tâche:

Cette phase constitue le point de départ du processus de planification. Elle a pour objet de définir les paramètres non explicitement spécifiés. Dans l'exemple traité, il s'agit de déterminer les emplacements initiaux des objets A et B, ainsi que les ressources nécessaires à l'exécution de la tâche (type de préhenseur, outillage de maintien pour B). Si l'assemblage traité avait nécessité plusieurs opérations de montage, le système aurait également dû déterminer l'ordre d'exécution de ces opérations.

Une caractéristique de ce type de planification est d'opérer à un niveau conceptuel assez éloigné des détails de la manipulation réelle. La méconnaissance de certaines contraintes de manipulation, amène le système à faire des choix arbitraires qui peuvent conduire à des échecs. Ces choix sont alors remis en cause dès la détection de l'échec. Par exemple, l'opération de localisation de A imposée par un risque de collision entre A et B (voir plus loin), nécessite de placer initialement A dans la région située sous la caméra.

2.2.3 Planification des actions nominales du robot:

Cette phase de planification a pour objet de construire un plan de manipulation nominal, c'est à dire un plan bâti uniquement sur une base géométrique. *Le système ignore alors la plupart des contraintes d'incertitudes* qui dérivent des erreurs de modélisation, de commande et de perception. Le plan produit est alors linéaire. Il peut être représenté par la

séquence suivante:

1. *INITIALISER ROBOT*
2. *REALISER pince SUR prise-A VIA <chemin>*
3. *SAISIR objet A <écart(prise-A) - ε >*
4. *REALISER* $R_1 \wedge R_2 \wedge R_3$ *VIA <chemin>* (*)
5. *POSER objet-A <écart(prise-A) + ε >*

Dans ce plan, la situation symbolique *"pince SUR prise-A"* correspond à une prise à la fois stable, accessible et compatible avec le contexte de l'opération de saisie. Les paramètres de mouvement représentés par le symbole < *chemin* > définissent des trajectoires qui évitent les collisions. L'instruction marquée d'une étoile est une instruction nominale qui est irréalisable compte tenu des contraintes d'incertitudes liées à la tâche. Afin d'être exécutable par le robot, cette instruction devra être "éclatée" en une action d'approche, suivie d'une séquence de commandes gardées et de mouvements compliants. Cette tâche incombe à l'étape de planification suivante.

Le plan construit à ce niveau n'est donc pas directement exécutable par le robot. Seule sa validité globale vis à vis de l'objectif visé et des contraintes d'accessibilité est analysée par le système.

2.2.4 Amendement du plan nominal:

Cette étape de planification permet de compléter et de corriger le plan construit antérieurement. Le but recherché est de transformer ce plan en un plan exécutable par le robot. *Les modifications apportées sont alors orientées vers le traitement des incertitudes.* Elles se traduisent en général par l'intégration d'opérations sensorielles et/ou de stratégies de mouvements fins. Une construction possible pour l'exemple traité peut avoir la forme suivante:

1. *INITIALISER ROBOT*
2. (a) *LOCALISER objet-A PAR VISION <position-prévue>*
   (b) *REALISER pince SUR prise-A VIA <chemin>*
3. *SAISIR objet-A <écart(prise-A) - ε >*
4. (a) *REALISER approche*($R_1 \wedge R_2 \wedge R_3$) *VIA <chemin>*
   (b) *DEPLACER objet-A SUIVANT* $< -dz >$ *JUSQUA <contact-table>*
   (c) *DEPLACER objet-A SUIVANT* $< -dy >$
   EN-MAINTENANT <contact-table> JUSQUA <contact-B_1 *>*
   (d) *DEPLACER objet-A SUIVANT* $< -dx >$
   EN-MAINTENANT <contact-table $\wedge$ *contact-*B_1 *>*
   JUSQUA <contact-B_2 *>*

5. *POSER objet-A* $<$*écart(prise-A)* $+ \varepsilon >$

avec:
$approche(R_1 \wedge R_2 \wedge R_3) = position(R_1 \wedge R_2 \wedge R_3) + dz + dx - dy$
$contact\text{-}table = (F_z > seuil_z)$
$contact\text{-}B_1 = (-F_y > seuil_y)$
$contact\text{-}B_2 = (F_x > seuil_x)$

Dans le processus de construction du plan, certaines contraintes ont un rôle actif dès la phase de planification. C'est en particulier le cas des incertitudes de montage qui sont à l'origine de la génération des stratégies de mouvements fins. Les actions $4b$, $4c$ et $4d$ sont de ce type. D'autres contraintes proviennent des phénomènes de propagation d'erreurs. Elles n'interviennent alors qu'à postériori sur le plan construit. Ainsi, l'opération de localisation de l'objet A a été introduite dans le plan afin de réduire l'incertitude liée à la future position de A dans les mors de la pince. En l'absence de cette opération sensorielle, les erreurs cumulées pourraient conduire l'objet A à heurter la pièce B au cours de l'exécution de l'instruction $4a$.

2.2.5 Modes de traitement des incertitudes:

Il apparaît au travers de l'exemple traité, que deux catégories d'incertitudes interviennent au niveau du processus de planification (cf. figure 2): celles qui sont prises en compte localement par les fonctions de planification, et celles qui sont traitées à postériori du fait de leurs natures fondamentalement globales.

- Le premier type de traitement a pour objet de contraindre le choix des actions, afin de réduire au maximun les risques d'échecs. Il repose sur l'utilisation de stratégies de manipulation, dont les conditions d'application sont dictées par une estimation grossière des incertitudes de position. Le système engendrera par exemple une prise stable conduisant à limiter les erreurs suivant certaines directions; il construira aussi des séquences de mouvements compliants permettant de réaliser une relation d'assemblage, en prenant appui sur des surfaces soigneusement choisies. Toutes les décisions prises à ce niveau ont un caractère strictement local.

- L'autre mode de traitement des incertitudes a pour objet de s'assurer de la validité globale du plan engendré vis à vis des contraintes de précision imposées par la tâche. Il s'appuie pour cela sur une étude des valeurs d'incertitudes présentes dans le plan, en propageant celles-ci au travers des actions planifiées. En cas d'incompatibilité entre les valeurs obtenues et celles imposées par la tâche, le système recherche les causes possibles de l'échec; il apporte ensuite les modifications nécessaires à l'obtention des valeurs requises. L'instruction $2a$ a ainsi été insérée dans le plan, afin d'obtenir la précision requise par l'instruction $4a$.

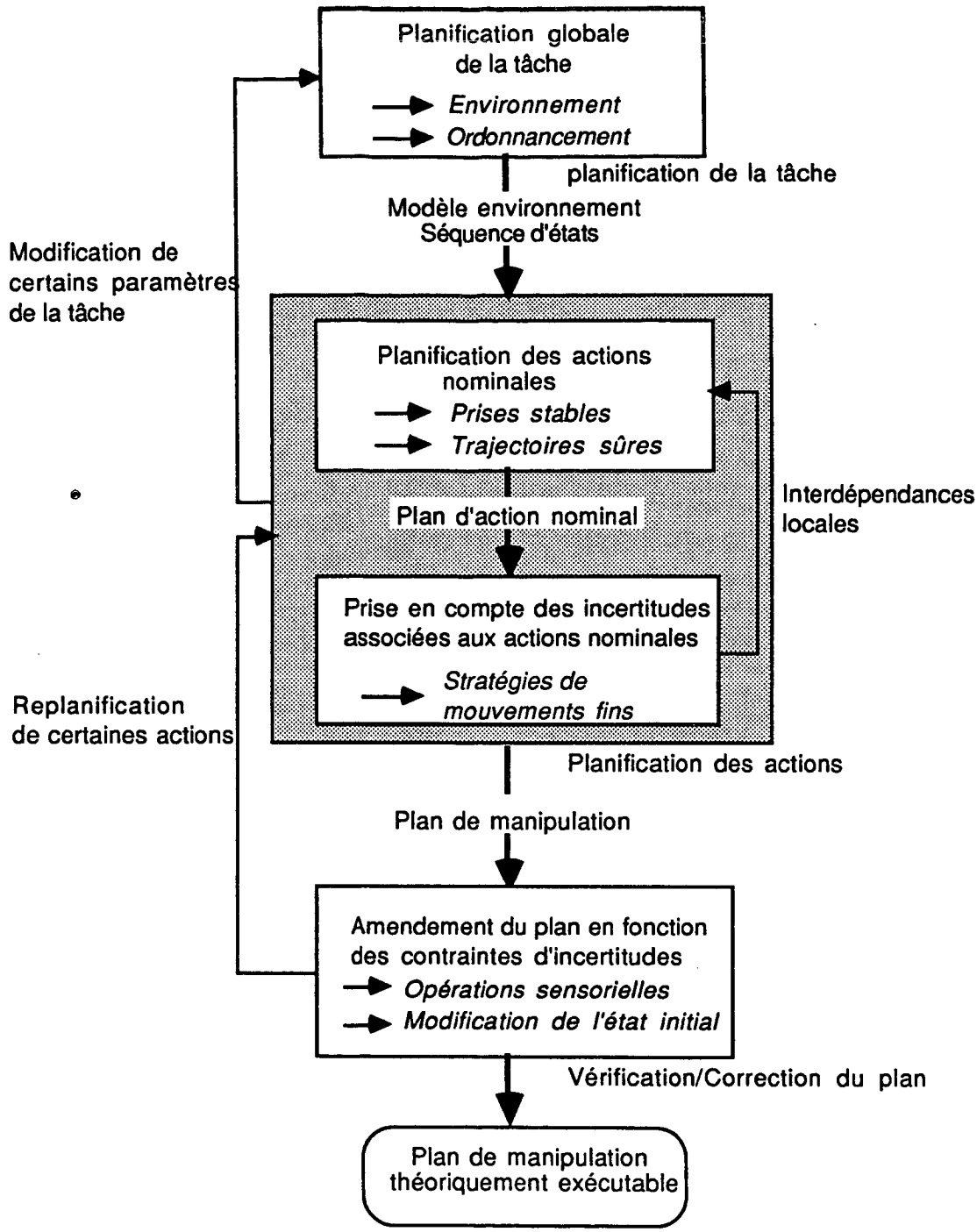


Figure 2: Intégration des incertitudes dans le processus de planification.

2.3 Travaux antérieurs:

Les problèmes posés par les incertitudes en robotique ont suscité des travaux de recherche, orientés suivant deux directions: la commande en présence d'incertitudes et la synthèse

de mouvements fins. Les travaux liés à la commande ont pour objet de faire exécuter des mouvements qui exploitent au maximun la géométrie des pièces. De tels mouvements sont guidés par les informations sensorielles issues des contacts. Les autres travaux adressent le problème du développement des algorithmes qui permettront d'organiser ces mouvements en vue de la réalisation des tâches projetées. Ils reposent sur une analyse locale des incertitudes et des relations de contact.

2.3.1 Travaux sur la commande:

Les travaux portant sur la commande ont donné lieu au développement des notions de "compliance" [Whitney 77] et de "mouvement gardé" [Will, Grossman 75]. Les techniques qui en découlent, conduisent à utiliser implicitement les contraintes géométriques de la tâche pour guider l'exécution des opérations de montage. Les *mouvements gardés* permettent d'atteindre des positions précises caractérisées par des conditions sensorielles (situations de contact en particulier). Les *mouvements compliants* reposent sur le principe du maintien des relations de contact qui sont aptes à guider le mouvement. Ce dernier point a motivé un grand nombre de recherches orientées vers l'implantation de lois de commande intégrant un contrôle explicite des forces [Whitney 77] [Raibert, Craig 81] [Mason 82] [Merlet 86] [Reboulet, Robert 85]. Quelques uns de ces travaux ont conduit au développement d'instructions appropriées, implantées dans des langages de programmation de robots: langage LUCIFER [Giraud 83], langage LM [Gandon 86].

D'autres chercheurs ont étudié les effets des frottements, afin de mieux maîtriser les phénomènes de glissement, de blocage et de coincement. La plupart de ces études ont été réalisées dans le contexte de l'insertion cylindrique [Inoue 74] [Simunovic 75]. Mason [Mason 82] a débordé de ce cadre en considérant le problème plus général du comportement d'un objet soumis à certaines forces de manipulation. Sa méthode permet de dériver les conditions de succés de la tâche, à partir d'une analyse du but à atteindre et de l'état initial.

2.3.2 Synthèse de mouvements fins à partir de stratégies prédéfinies:

Les premières tentatives de synthèse automatique de mouvements fins étaient basées sur le concept de "schéma de programme" [Taylor 76] [Lozano-Perez 76]. Le principe consistait à analyser les contraintes de précision imposées par la tâche, afin de choisir puis de compléter des schémas types connus à priori. Les choix réalisés reposaient alors essentiellement sur une estimation des erreurs rencontrées. Ils conduisaient à spécifier quantitativement tous les paramètres des mouvements contenus dans le corps des procédures choisies. Plus tard, Brooks [Brooks 82] a amélioré le procédé en propageant symboliquement les incertitudes dans le programme. Le mode de calcul induit permet d'estimer en tout point les erreurs engendrées par les actions antérieures; il permet aussi d'évaluer les valeurs initiales qu'il est possible d'affecter à certains paramètres, sans dégrader les chances de succés du programme. Les spécifications du système TWAIN [Lozano-Perez, Brooks 85] étaient basées sur ce type d'approche, mais aucune implantation significative n'a été réalisée. Une limitation importante du procédé réside dans la difficulté qu'il y a à apprécier certaines erreurs,

et dans les techniques de propagation utilisées qui conduisent à surestimer très largement les incertitudes.

Les difficultés liées à l'évaluation précise des incertitudes ont motivé le développement d'une autre approche basée sur des techniques d'apprentissage [Dufay 83] [Dufay, Latombe 84]. Cette approche consiste à combiner à l'issue de plusieurs tentatives d'exécution, des stratégies partielles décrites par des règles expertes. Dans un premier temps, le système réalise plusieurs exécutions de la tâche en interaction avec les actionneurs et les capteurs du robot. Il utilise pour cela des connaissances expertes permettant d'analyser les situations rencontrées, et d'engendrer des actions correctives. La deuxième étape consiste à combiner les résultats des différents essais, afin d'obtenir un programme de mouvements fins aussi complet que possible. Le système procède pour cela à des transformations itératives du graphe obtenu par combinaison des traces d'exécution. Ce graphe représente alors la structure de contrôle du programme à synthétiser. Il peut comporter des cycles et des points de branchements multiples. Cette méthode permet de construire des programmes de mouvements fins fiables et efficaces. Elle n'est cependant pas capable d'élaborer des solutions nouvelles, ni d'analyser des situations symboliques non décrites explicitement dans la base de connaissances.

Les deux approches précédentes reposent sur l'utilisation d'un répertoire de stratégies de manipulation. Chaque stratégie représente une classe d'opérations susceptible de résoudre un problème de montage particulier. Cette hypothèse de granularité de l'univers du montage facilite la tâche du processus de planification. Elle ne correspond cependant pas à la réalité de l'assemblage mécanique: une simple variation locale de la géométrie d'une pièce peut avoir des répercussions importantes sur la stratégie de montage utilisée. La complexité géométrique et mécanique qui en découle rend impossible la constitution d'un ensemble raisonnable de classes identifiables.

2.3.3 Synthèse de mouvements fins par raisonnement géométrique:

Une autre manière de procéder consiste à construire pas à pas le plan de montage, en raisonnant sur la géométrie de la tâche [Lozano-Perez et al. 83] [Valade 85] [Laugier, Theveneau 86]. Les bases formelles de cette approche ont été définies par Lozano-Perez, Mason et Taylor [Lozano-Perez et al. 83]. Elles ont été ensuite développées par Mason [Mason 84] et par Erdmann [Erdmann 84]. La méthode proposée opère dans l'espace des configurations de l'objet manipulé. Elle conduit à construire récursivement une séquence de mouvements compliants permettant de réaliser le montage. Cette séquence est obtenue en raisonnant à partir du but à atteindre. Le principe consiste à représenter l'objet manipulé par un point P, et l'objectif à atteindre par une surface G située sur le sous assemblage. La recherche des mouvements à exécuter repose alors sur le calcul des "pré-images" de G, c'est à dire sur le calcul des ensembles des positions de P qui permettent d'atteindre G en un seul mouvement de direction fixe. Lorsque P n'appartient à aucune des pré-images construites antérieurement, le procédé est appliqué récursivement jusqu'à l'obtention d'une séquence satisfaisante. Les effets des incertitudes et des frottements sont pris en compte dans l'algorithme en introduisant le concept de "pré-image forte", c'est à dire de pré-image permettant d'atteindre avec certitude la surface objectif.

La méthode des pré-images est potentiellement très puissante, mais elle fait surgir des problèmes théoriques qui sont loins d'être résolus. Les travaux les plus avancés dans cette voie sont ceux de Erdmann [Erdmann 84]. Ils apportent des solutions applicables à des problèmes de l'espace bidimensionnel, mais la généralisation de ces solutions à l'espace tridimensionnel semble difficilement envisageable: la complexité déjà exhibée par les algorithmes proposés laisse prévoir une explosion combinatoire dans des cas réels. C'est pourquoi nous avons développé une méthode permettant de réduire la taille du graphe de recherche en guidant heuristiquement le raisonnement géométrique, et en explorant uniquement les branches qui apparaissent comme les plus prometteuses [Laugier, Theveneau 86]. Ce procédé de planification des mouvements fins est décrit dans le paragraphe qui suit.

3 Planification des mouvements fins:

3.1 Présentation du problème et de l'approche:

Le problème posé consiste à synthétiser automatiquement un programme de mouvements fins. L'objectif visé est alors d'engendrer des mouvements aptes à réaliser une relation d'assemblage donnée, malgré les contraintes d'incertitudes imposées par la tâche. Les stratégies implantées dans ce programme combinent des opérations sensorielles avec des petits mouvements du robot. Elles permettent ainsi de s'affranchir des erreurs de commande et de perception.

Planifier une opération de montage peut ainsi être vue comme la recherche d'une séquence ordonnée de relations de contact, aptes à guider le robot vers l'objectif à atteindre. Chaque relation réalisée permet de se rapprocher du but, en réduisant certaines composantes d'incertitude. Elle met en jeu des mouvements destinés à rechercher un contact, et/ou à en maintenir d'autres créés par des opérations antérieures. Le processus de planification tente d'abord de définir symboliquement les contacts qui semblent présenter un intérêt pour le montage. Il choisit ensuite parmi les solutions possibles, celles qui lui paraissent être les mieux adaptées au contexte de la tâche. Les paramètres des mouvements engendrés (direction, amplitude, conditions d'arrêt, forces) sont alors déterminés en fonction de critères d'accessibilité et de fiabilité.

La méthode que nous avons développée, procède par chaînage arrière afin de *déduire un plan de montage à partir d'une étude des démontages possibles de l'objet.* Cette approche se justifie par le fait que les contacts existants contraignent les mouvements relatifs des deux objets, ce qui réduit le nombre des hypothèses à considérer. L'algorithme implanté opère en deux phases d'analyse et de recherche [Laugier, Theveneau 86]:

- *La phase d'analyse* a pour objet de construire un graphe qui représente les différentes manières théoriques de démonter l'assemblage. Le système fait alors abstraction des contraintes réelles de la manipulation (risques d'échecs dus à l'imprécision du robot en particulier). Il analyse uniquement les contraintes de mouvement imposées par les contacts. Les transitions du graphe représentent alors des mouvements potentiels, dont le rôle est de relâcher un à un les contacts mis en jeu par l'assemblage.

- *La phase de recherche* conduit à choisir un "chemin" inverse dans le graphe de démontage, afin de construire une stratégie de mouvements fins particulière. Cette recherche peut conduire en cas d'échec à raffiner certaines parties du graphe. Par exemple, une relation de contact n'est pas réalisable à cause des erreurs de commande. La recherche est alors guidée par des heuristiques et des connaissances expertes qui tentent d'optimiser la solution en termes d'efficacité et surtout de fiabilité des opérations.

La principale originalité de cette approche réside dans le fait que le graphe d'états construit peut être localement affiné lors de la recherche d'une solution. Les motivations qu'il y a derrière cette approche sont doubles:

- Réduire l'arbre de recherche.
- Exécuter les calculs géométriques coûteux uniquement lorsque cela s'avère nécessaire.

Les modes de raisonnement utilisés pour la planification des mouvements fins reposent sur une analyse géométrique des contacts mis en jeu. Deux aspects de ces contacts sont alors pris en compte par le système:

- Un *aspect statique* décrivant la géométrie et les propriétés mécaniques des contacts.
- Un *aspect dynamique* décrivant les contraintes de mouvements induites par les surfaces en contact.

Les modèles utilisés pour ce raisonnement sont décrits dans les deux paragraphes qui suivent.

3.2 Spécification symbolique des contacts:

Un contact entre deux objets O1 et O2 met en relation deux parties non nécessairement connexes de la surface de ces objets. Il peut être spécifié de la manière suivante:

$$CONTACT(O_1, O_2) = \{((e_1, e'_1)(e_2, e'_2) \cdots (e_m, e'_m)), T, P\}$$

où les couples (e_i, e'_i) représentent les entités géométriques en contact, T définit le type du contact (ponctuel, linéique ou surfacique), et P représente les paramètres géométriques qui caractérisent analytiquement le contact.

3.2.1 Entités géométriques contribuant au contact:

Les entités géométriques (EG) présentes dans les couples (e_i, e'_i), proviennent des modèles surfaciques associés aux objets: sommets, arêtes rectilignes ou circulaires, faces planes, cylindriques, coniques ou sphériques. Les combinaisons possibles pour ces entités sont celles qui engendrent des situations physiques potentiellement réalisables malgré les incertitudes de position, c'est à dire tous les couples de type: face-face, face-arête, face-sommet et arête-arête (lorsque ces dernières ne sont pas colinéaires). Les couples de type arête-arête n'étant pas pertinents pour les opérations de montage, nous ne prendrons en considération que les contacts qui mettent en jeu des relations de type $face - X$, avec X = face, arête ou sommet. Les couples (e_i, e'_i) d'entités (avec $e_i \in modele(O_1)$ et $e'_i \in modele(O_2)$) qui vérifient de telles relations sont appelés *éléments de contact.* Les autres combinaisons d'entités sont alors considérées comme des situations instables, qui marquent la limite de rupture d'autres contacts (cf. figure 4). Nous parlerons alors de *contact par adjacence.* Cette forme "dégénérée" de contact est irréalisable physiquement, à cause des incertitudes liées à la commande et à la perception. Sa détermination symbolique est cependant importante, car elle permet de mettre en évidence les discontinuités produites par la géométrie des objets.

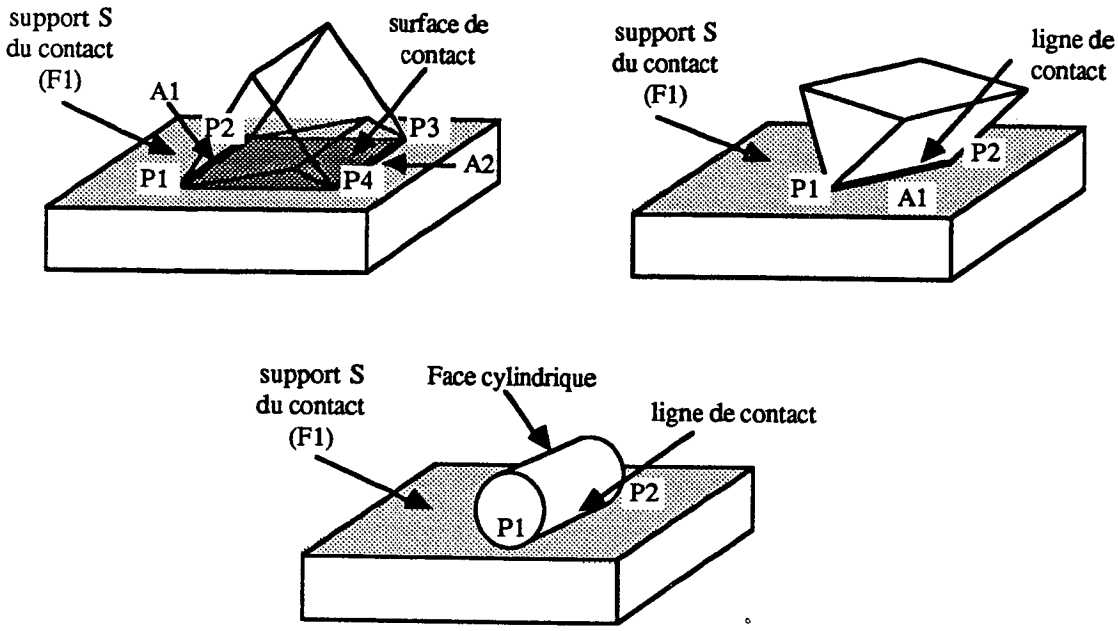


Figure 3: Exemples de contacts
a- Contact surfacique engendré par une face plane et deux arêtes.
b- Contact linéique engendré par une face plane et une arête.
c- Contact linéique engendré par une face plane et une face cylindrique.

3.2.2 Type du contact:

Le type d'un contact est défini par la topologie de l'ensemble des points en contact (point, courbes ou surfaces). Les caractéristiques de cet ensemble varient en fonction du type et de l'organisation spatiale des EG concernées. Par exemple, un couple de faces cylindriques "pleines" peut suivant l'orientation respective des deux faces, donner lieu à un contact ponctuel ou linéique. La combinaison de plusieurs éléments de contact contribue aussi à définir le type du contact. Un contact surfacique peut par exemple être construit à partir de deux éléments de contact de type linéique (cf. figure 3). Intuitivement, nous dirons que ce contact multiple peut être assimilé à un contact plan, du fait qu'il possède des propriétés statiques analogues. La surface définie par l'enveloppe convexe des points de contact représente alors un contact plan fictif, dont la caractérisation dérive directement

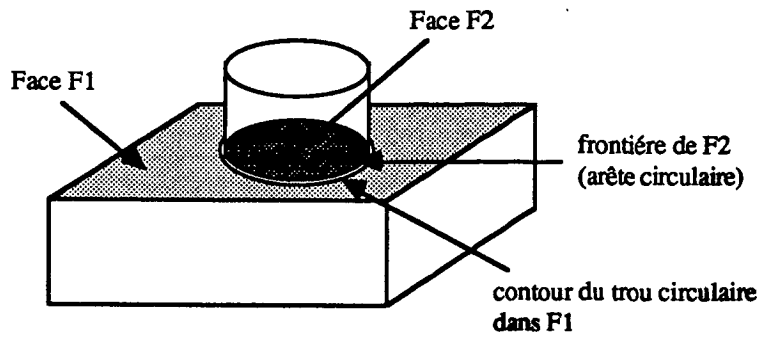


Figure 4: Exemple de contact par adjacence.

du concept de "polygone de sustentation" utilisé en statique [Joyal, Provost 66].

Le nombre réduit de cas possibles permet de déterminer le type des contacts sans passer par un calcul d'enveloppe convexe en $O(n \log n)$, dans lequel n est le nombre d'EG concernées. La technique employée repose alors sur une simple analyse des ensembles $e_i \cap e'_i$ produits par les éléments de contact (e_i, e'_i) [Laugier, Dufay 82]. Dans le cas des contacts multiples, ces éléments sont ensuite combinés par "fusion" des ensembles $e_i \cap e'_i$ associés [Ijel 86]. Par exemple, deux contacts ponctuels mettant en jeu une même face plane, produiront par fusion un contact linéique caractérisé par le segment $P1P2$ ($P1$ et $P2$ sont les points de contacts). L'algorithme proposé calcule également les paramètres géométriques des contacts analysés.

3.2.3 Paramètres géométriques du contact:

Les paramètres géométriques caractérisent analytiquement le contact par un couple $(S, \delta S)$, dans lequel S est une surface définie par les points en contact, et δS délimite la région de S qui contient les points de $conv(E1 \cap E2)$. La surface S est appelée *support du contact*; elle peut être plane, cylindrique, conique ou sphérique. La région délimitée par δS est appelée *surface de contact*; elle peut représenter des domaines plans polygonaux ou circulaires, des segments de droites, des points, ou des secteurs cylindriques, coniques ou sphériques.

Dans le cas d'un contact surfacique, S est définie sans ambiguïté par l'ensemble $E1 \cap E2$. Dans les autres cas, nous avons choisi de prendre une face de $E1$ ou de $E2$ comme génératrice de S. En cas d'ambiguïté (par exemple: contact entre une face cylindrique et une face plane), nous prendrons par convention une des faces de l'objet qui subit le contact, c'est à dire de l'objet non tenu par le robot. Les contacts illustrés par la figure 3 sont alors représentés de la manière suivante:

(a) (CONTACT ((FACE F1 ARETE A1) (FACE F1 ARETE A2))
(TYPE plan)

(SUPPORT plan (0 0 1))
(FRONTIERE (P1 P2 P3 P4)))

(b) (CONTACT ((FACE F1 ARETE A1))
(TYPE droite)
(SUPPORT plan (0 0 1))
(FRONTIERE (P1 P2)))

(c) (CONTACT ((FACE F1 FACE FC2))
(TYPE droite)
(SUPPORT plan (0 0 1))
(FRONTIERE (P1 P2)))

Dans le cas du contact (c), F1 est choisi comme support car cette face appartient à l'objet fixe. Le cylindre peut alors se déplacer sur cette face (dans certaines limites) sans détruire le contact.

3.3 Contraintes de mouvements associées aux contacts:

3.3.1 Notion de mouvement potentiel:

Chaque contact réduit le nombre de d.d.l de l'objet mobile, c'est à dire de l'outil terminal ou de l'objet tenu par le robot. Afin de pouvoir décider des mouvements à exécuter, le système de planification doit raisonner sur les directions de déplacement contraintes par les contacts.

Le problème à résoudre est alors celui de la détermination des mouvements qui sont *potentiellement exécutables* par l'objet mobile, à partir d'une situation mettant en jeu une ou plusieurs relations de contact. Ces mouvements définissent soit des translations pures, soit des rotations pures. Ils sont considérés sur un plan *strictement local*, indépendamment des amplitudes de déplacement qu'il est possible de leur associer. Si A est un objet mobile en contact avec un autre objet B, nous définirons un mouvement potentiel pour A comme un "mouvement d'amplitude ds supérieure à l'erreur maximum de commande, tel que le déplacement exécuté n'engendre aucune collision entre les parties en contact" [Laugier 87]. Sur le plan pratique, cette définition nous a conduit à développer un support analytique pour représenter explicitement les mouvements interdits, ceux qui conservent les contacts et ceux qui les détruisent. Le formalisme mathématique utilisé est décrit ci-dessous.

3.3.2 Représentation analytique des mouvements en translations:

Une translation de l'espace tridimensionnel est communément représentée par un vecteur de $\Re^3$. Dans notre modèle, nous avons choisi de la représenter par un couple (v, a), dans lequel v est un vecteur unitaire de $\Re^3$ et a est un réel. v et a définissent respectivement la direction de la translation et son amplitude. Un groupe de translations peut alors être vu comme un sous-ensemble de $S(1)X\Re$, dans lequel $S(1)$ est la sphère unité. Ce mode de représentation permet de raisonner séparément sur les directions de déplacements

(ensembles de vecteurs v), et sur les mouvements complets (ensembles de couples (v,a)). A ce niveau, nous nous intéressons uniquement aux ensembles de vecteurs de $S(1)$.

Un point de la sphère unité définit alors sans ambiguïté une direction de translation, et une calotte sphérique représente un ensemble connexe de déplacements possibles. Par exemple, une face plane intervenant dans un contact engendre un domaine en forme de demi-sphère, et un couple de contacts détermine un ensemble de déplacements valides délimité par l'intersection des domaines associés à chacun des contacts. Le calcul des degrés de liberté (d.d.l) résultants peut alors être exécuté dans un espace bidimensionnel, en projetant les domaines obtenus sur un plan (φ, ϑ).

Contacts à supports plans:
Dans le cas des contacts ayant pour support une surface plane (contacts de type "face plane - X", avec X quelconque), ce calcul met en jeu des fonctions du type [Theveneau 85]:

$$\vartheta = \arctan((N_x \cdot \cos\varphi + N_y \cdot \sin\varphi)/ - N_z) \qquad (1)$$

où $(N_x\, N_y\, N_z)$ représente la normale extérieure au plan de contact, c'est à dire la normale orientée vers l'extérieur de l'objet fixe. La calotte sphérique représentée est alors une demi-sphère délimitée par le plan de contact d'équation $N_x.x + N_y.y + N_z.z = 0$. Il peut être aisément montré que les fonctions de cette famille possèdent de "bonnes propriétés", qui permettent de calculer très facilement les domaines résultants. Ce calcul est illustré graphiquement par la figure 5.

Contacts à supports non planaires:
Un calcul similaire peut être appliqué lorsque les parties en contact ne sont pas planes. Les fonctions utilisées pour ce calcul varient alors avec le type des surfaces traitées. Un contact de type cylindrique déterminera par exemple deux directions de mouvements portées par l'axe du cylindre. Un contact de type conique définira par contre un ensemble de mouvements potentiels plus complexe, dont les limites sont spécifiées par la fonction:

$$\cos\vartheta(a\cos\varphi + b\sin\varphi) + c\cos\vartheta - \cos\alpha = 0 \qquad (2)$$

où $(a\,b\,c)$ et α représentent respectivement l'axe du cône et son angle d'ouverture.

D'autres types de contacts peuvent aussi être engendrés par un couple d'EG non planaires. Une technique possible pour se ramener aux cas connus consiste à introduire des plans de contacts fictifs, définis tangentiellement aux parties en contact. Il est alors possible de traiter ces contacts en combinant simplement les fonctions précédentes. En particulier, un contact surfacique ne mettant en jeu qu'une portion de la surface d'une EG, détermine un domaine délimité à la fois par le contact surfacique et par les plans tangents situés aux bornes de cette surface. Le calcul est alors réalisé à l'aide des fonctions (1) et (2). Par exemple, un contact cylindrique partiel définit un domaine caractérisé par l'expression "$D(P1) \cap D(P2) \cap D(P3) \cup D(cylindre)$", dans laquelle $D(P1)$, $D(P2)$ et $D(P3)$ représentent les domaines associés aux trois plans tangents associés aux "bords" de la surface de contact, et $D(cylindre)$ correspond aux deux directions axiales.

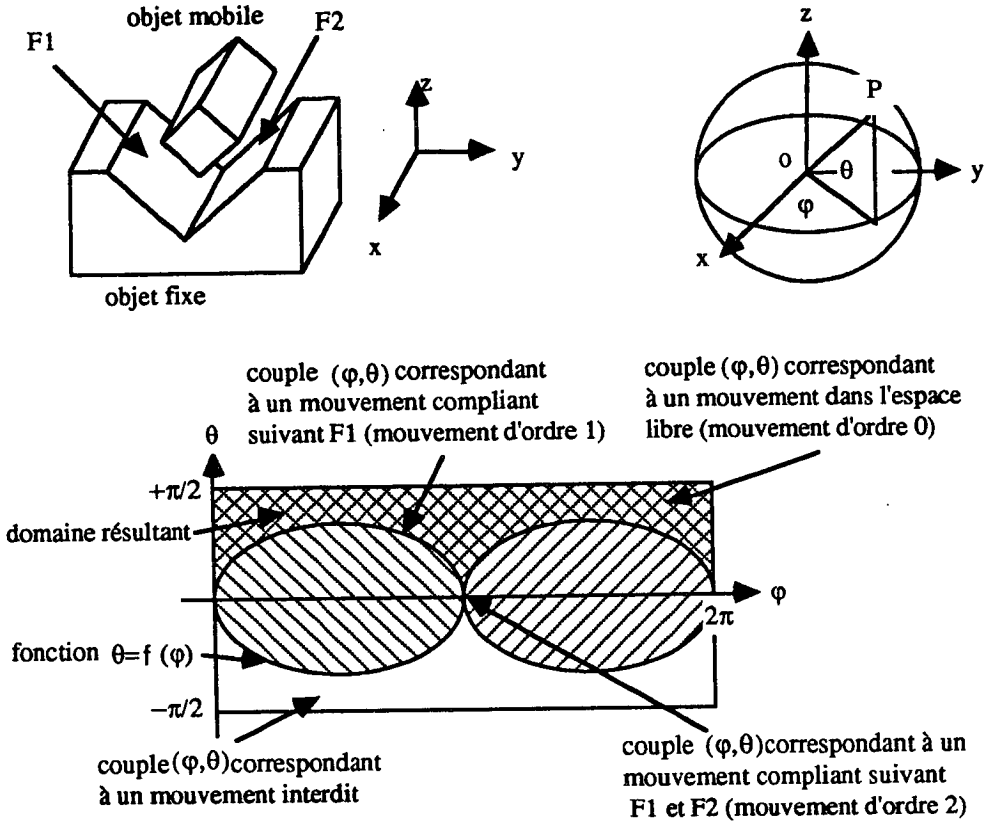


Figure 5: Représentation analytique des mouvements potentiels associés à un couple de contacts plans.

3.3.3 Propriétés de la représentation:

Soit $S = \{C1, C2, \ldots Cn\}$ une situation mettant en jeu n contacts $C1, C2, \ldots Cn$. Les contraintes de mouvements associées à S sont représentées par un domaine D, défini comme l'intersection des domaines $D1, D2, \ldots Dn$ associés aux contacts $C1, C2, \ldots Cn$. Chaque domaine D_i $(i = 1 \ldots n)$ est construit à l'aide des fonctions (1) ou (2). On note D_i^* son intérieur, et δD_i sa frontière $(D_i = D_i^* \cup \delta D_i^*)$. On note de la même manière l'intérieur et la frontière de D.

Une caractéristique importante de la représentation utilisée est de permettre de *distinguer les différentes catégories de mouvements* qu'il est possible d'associer à S (cf. figure 5):

- tout couple (φ, ϑ) n'appartenant pas à D représente une direction de déplacement interdite;

- tout couple (φ, ϑ) inclus dans D^* définit un mouvement qui conduit à relâcher tous les contacts;

- tout couple (φ, ϑ) de δD représente un mouvement compliant qui conduit à glisser sur certaines surfaces de contact:
si $(\varphi, \vartheta) \in \delta D_{i_1} \cap \delta D_{i_2} \cap \cdots \delta D_{i_j}$ et $(\varphi, \vartheta) \notin \delta D_k \, (k \neq i_1, i_2 \cdots i_j)$, alors (φ, ϑ) définit un mouvement qui conserve les contacts $C_{i_1}, C_{i_2} \ldots C_{i_j}$, et qui relâche les autres contacts.

Cette caractéristique est essentielle pour le processus de planification, car elle permet de raisonner à chaque étape sur les modifications de contacts engendrées par les mouvements sélectionnés.

Une autre propriété de ce mode de représentation est d'être "complet", c'est à dire d'être apte à représenter tous les mouvements qui sont possibles. Ainsi, n'importe quelle solution potentielle au problème de la planification d'un mouvement à partir de S, peut théoriquement être trouvée dans le domaine D associé. La difficulté réside alors dans l'impossibilité qu'il y a à analyser toutes les solutions représentées. Ce point est discuté plus loin.

La propriété de complétude n'est cependant pas conservée dans le cas des contacts courbes qui mettent en jeu des plans tangents fictifs. En effet, ces plans conduisent à approximer les domaines représentés, en ne considérant aux limites que les mouvements engendrés tangentiellement aux surfaces de contact. Cette approximation est cependant raisonnable dans le sens où elle tente de caractériser *localement* les mouvements possibles de l'objet mobile. Ceci est parfaitement cohérent avec le fait que les mouvements destinés à conserver le contact ont dans ce cas des directions tangentielles qui varient constamment avec la courbure de la surface. Il est cependant probable que l'extension de ce mode de représentation au cas de surfaces courbes plus générales, introduise des difficultés de calcul qui en limitent l'intérêt.

3.3.4 Traitement des rotations:

Les rotations sont intrinsèquement plus difficiles à manipuler que les translations. Elles pourraient théoriquement être représentées à l'aide d'une sphère unitaire, dans laquelle chaque vecteur définit la direction d'un axe de rotation. Mais les domaines obtenus dépendent presque toujours de la position spatiale des axes considérés. Par exemple, un axe de rotation contenu dans le plan de contact et situé hors de la surface de contact, engendrera un mouvement qui détruit ce dernier; cet axe conduira par contre à modifier la nature du contact obtenu par rotation, lorsqu'il est tangent à la surface de contact.

Cette particularité des rotations rend inutilisable le modèle précédent, du fait de la multiplicité des représentations qu'il est possible d'associer à un contact. Elle suggère par contre une autre approche consistant à regrouper dans une même représentation, tous les axes qui ont un comportement "homogène" vis-à-vis des contacts. On regroupera par exemple tous les axes de rotation qui conservent potentiellement le contact considéré; on procédera de même pour ceux qui font passer d'un contact surfacique à un contact linéique.

Dans la pratique, les objets manipulés définissent des ensembles aisément représentables pour les cas courants. Par exemple, les axes qui conservent potentiellement un contact plan sont tous perpendiculaires au plan de contact; ceux qui conservent un contact cylindrique ou conique sont réduits à l'axe de révolution de la surface. Cette approche permet de traiter simplement un grand nombre de cas courants. Elle mériterait cependant d'être développée de manière plus complète dans le futur.

3.4 Graphe d'états associé aux mouvements fins:

Les ensembles de contacts et de mouvements potentiels exhibés par le raisonnement précédent sont organisés sous la forme d'un graphe. Ce graphe est appelé *graphe d'états associé aux mouvements fins.* Il représente les séquences de mouvements et d'états du robot, qui ont été retenues comme des *solutions potentielles* pour le problème posé. La nature incomplète de ce graphe provient de l'impossibilité qu'il y a à représenter explicitement toutes les solutions possibles (cet ensemble étant infini). Nous verrons plus loin que le graphe construit n'est pas figé, et qu'il peut être localement affiné en cas d'échec. Un tel échec est alors détecté lors de la recherche d'une solution particulière. Il indique qu'aucune stratégie exécutable n'est contenue dans le graphe d'états courant.

3.4.1 Représentation du graphe d'états:

Soient A et B les deux objets à assembler. Le graphe d'états $G(A/B)$ associé aux mouvements fins de montage de A sur B, est modélisé par un graphe orienté $(\mathcal{S}, \mathcal{A})$. Les nœuds de $\mathcal{S}$ représentent des états E_P du robot, qu'il est possible de réaliser avec les commandes du système, et de différencier à l'aide des moyens perceptifs disponibles. Les arcs de $\mathcal{A}$ symbolisent les mouvements qui permettent de passer d'un état à l'autre.

Un nœud s_i de $\mathcal{S}$ est caractérisé par les informations suivantes:

$$s_i = (P, C, D, I, Q)$$

où P est un ensemble de positions du robot, C est un ensemble de contacts, D est un ensemble de mouvements potentiels, I est une information qualitative portant sur la situation physique représentée (insertion cylindrique par exemple),et Q est une évaluation heuristique de la "qualité" de cette situation (robustesse vis-à-vis des incertitudes et des aléas mécaniques). Les paramètres P et C définissent symboliquement l'état E_P. Le paramètre D indique les mouvements qui peuvent être théoriquement appliqués à partir de l'état E_P; cette information est utilisée pour affiner localement le graphe en cas de nécessité. Les paramètres I et Q ont pour objet de guider la recherche d'une solution.

Un arc a_{ij} de $\mathcal{A}$ est caractérisé par les informations suivantes:

$$a_{ij} = (T, C, A, Q)$$

où T est une transformation géométrique, C et A sont des ensembles de faces sur lesquelles l'objet mobile doit respectivement glisser et s'arrêter, et Q est une pondération heuristique

traduisant la "qualité" du mouvement (risques de collisions ou de blocages dus à de petites variations de l'environnement, sensibilité aux effets de la pesanteur ...). Les paramètres T, C et A définissent le mouvement qui permet de passer de l'état E_{P_i} à l'état E_{P_j}. Le paramètre Q est utilisé pour guider la recherche d'une solution.

3.4.2 Construction du graphes d'états:

La représentation analytique que nous avons développée, permet de caractériser l'ensemble des mouvements qui sont potentiellement exécutables par le robot, à partir d'une situation de contact donnée (soit d'un état E_i du robot). Elle n'est cependant pas directement exploitable par le système de planification, du fait que les domaines représentés définissent chacun une infinité de solutions possibles (et donc une infinité d'arcs possibles pour chaque nœud du graphe).

Une technique très souvent utilisée en robotique pour résoudre ce problème, consiste à *discrétiser les ensembles de solutions étudiés.* Cette technique conduit à découper le domaine analysé en calottes sphériques du type $\Delta\varphi X\Delta\vartheta$. Chaque calotte ΔS représente alors un ensemble de mouvements qui seront étudiés "globalement" par le système. Cette approche impose que toutes les directions m de ΔS conduisent théoriquement (compte non tenu des erreurs de commande) à une même situation de contact ou de non contact, après un déplacement issu du point courant P, et exécuté suivant m. Dans le cas où les objets sont polyédriques et où les mouvements sont des translations, cette contrainte peut être vérifiée au moyen de techniques d'analyse de "visibilité" [Buckley 87]. Le calcul est alors réalisé dans l'espace des configuration de l'objet mobile.

La complexité algorithmique liée à cette approche, ainsi que son inaptitude à traiter des surfaces courbes et des rotations (du moins dans l'état actuel de nos connaissances), nous ont conduit à appliquer une méthode de recherche heuristique. Le principe consiste à n'étudier qu'un sous-ensemble des solutions possibles, en sélectionnant dans D les directions de mouvement qui semblent être les prometteuses. Cette approche se justifie par le fait que la plupart des mouvements nécessaires à la réalisation des montages mécaniques sont exécutés suivant des *directions privilégiées* définies par les surfaces de contacts. En particulier, de nombreux mouvements fins peuvent être appliqués perpendiculairement ou parallèlement aux contacts voisins. C'est pourquoi les directions d'étude sélectionnées par la fonction choix, correspondent toutes à des points caractéristiques du modèle analytique: points d'intersections entre des courbes de type (1) ou (2), points répartis périodiquement sur les limites des domaines.

L'algorithme utilisé pour construire le graphe d'états est alors le suivant:

1. Créer le nœud EF, et le mettre dans la liste $OUVERT$.
2. Si $OUVERT = \emptyset$ retourner $G(A/B)$, sinon traiter le nœud x situé en tête de liste $OUVERT$.
   Choisir n directions d'étude $d_1, d_2 \cdots d_n$ dans D_x.
   Créer un arc a_{xi} pour chaque direction d_i précédente.

3. Créer un nœud y_i pour chaque arc a_{xi}.
   Si l'état E_i associé à y_i est déjà représenté par un nœud z de $G(A/B)$, fusionner y_i et z.

4. Mettre tous les nouveaux nœuds qui possèdent un ensemble de contacts vide dans EI; mettre les autres dans la liste $OUVERT$.
   Retourner en (2).

$OUVERT$ désigne la liste des nœuds non encore traités, et D_x est l'ensemble des mouvements potentiels associés au nœud x. EF symbolise l'état où A et B sont assemblés, et EI représente l'ensemble des nœuds de $G(A/B)$ qui possèdent un ensemble vide de contacts (ces nœuds sont considérés comme des points de départ possibles pour la construction d'une stratégie de mouvements fins, du fait qu'ils représentent des états directement réalisables par le robot).

Choix des directions d'étude:
Les règles de sélection sont activées par les données contenues dans le champ I des nœuds du graphe. Par exemple, quatre directions de déplacement seront initialement engendrées pour un couple de contacts supportés par deux faces non parallèles F_1 et F_2: $d_1 = N_1 \wedge N_2$, $d_2 = -d_1$, $d_3 = d_1 \wedge N_1$ et $d_4 = d_2 \wedge N_2$, où N_1 et N_2 représentent respectivement les normales extérieures de F_1 et de F_2. Les directions d_1 et d_2 définissent des mouvements compliants qui conservent les deux contacts; d_3 et d_4 conduisent à rompre respectivement les contacts suivant F_2 et F_1. Les hypothèses de mouvement ne sont retenues par le système que dans la mesure où elles vérifient les contraintes représentées par D_x.

Création des nœuds et des arcs de $G(A/B)$:
Les nœuds de $G(A/B)$ sont créés à l'issue d'un calcul géométrique permettant de:

- Déterminer les contacts mis en jeu (paramètre C).
- Calculer l'ensemble des mouvements potentiels (paramètre D).
- Vérifier que l'état E_i considéré n'est pas déjà représenté dans $G(A/B)$, et qu'il peut être discerné des états voisins.

Les calculs utilisés pour résoudre les deux premiers points sont ceux décrits précédemment; ceux appliqués pour traiter le dernier point utilisent des notions développés dans [Laugier 87]. Les paramètres P et I sont des sous-produits des calculs précédents.

Les arcs de $G(A/B)$ sont créés à l'issue d'un calcul géométrique permettant de:

- Déterminer l'amplitude maximum du déplacement (paramètre T).
- Déterminer les contacts impliqués dans le mouvement (paramètres C et A).

Le premier point est traité à l'aide d'une fonction de calcul de "débattements valides" [Laugier 87]; l'autre point met en jeu des calculs tels que ceux mentionnés dans le paragraphe 3.2.2.

3.5 Recherche d'une stratégie de mouvements fins:

3.5.1 Principe de la recherche:

Intuitivement, nous dirons que tout chemin de $G(A/B)$ permettant de passer d'un nœud de EI au nœud EF, représente une stratégie de mouvements fins apte à réaliser le montage de A sur B. Le problème posé consiste donc à rechercher un "bon chemin" dans le graphe, puis à convertir la séquence obtenue en un programme exécutable par le robot. L'algorithme utilisé pour cela est le suivant:

1. Rechercher un chemin SG partant d'un nœud de EI et se terminant sur le nœud EF.

2. Vérifier que chaque arc de SG représente un mouvement exécutable par le robot (pas de collision, faisabilité de l'objectif).
   En cas d'échec affiner $G(A/B)$, puis retourner en (1).

3. Synthétiser le programme de mouvements fins représenté par SG.

Remarque: Dans le cas général, SG est un sous-graphe qui inclut à chaque niveau l'ensemble des états (nœuds de $G(A/B)$) pouvant être atteints après exécution du mouvement retenu. Ceci provient du fait que les erreurs de commande peuvent dans certains cas conduire le robot à terminer son mouvement dans différentes situations de contact. La prise en compte de cet aspect est décrite dans [Laugier 87].

3.5.2 Fonction de parcours:

La nature dynamique du graphe $G(A/B)$ (certaines parties pouvant être affinées pendant le parcours), ne se prête pas à l'utilisation d'un algorithme de type A^*. C'est pourquoi nous avons choisi d'appliquer une méthode de parcours guidée simultanément par une fonction coût et par des "conseils dynamiques" codés sous la forme de règles expertes:

- *La fonction coût* combine les pondérations heuristiques attribuées aux nœuds et aux arcs du graphe. Elle conduit d'une part à minimiser le nombre de mouvements, et d'autre part à maximiser la qualité des solutions choisies. Un autre effet de cette fonction est de contraindre le système à toujours tenter d'accroître le nombre de degrés de liberté bloqués par les contacts.

- *Les conseils dynamiques* sont activés chaque fois qu'une situation singulière est détectée dans le graphe. Ils permettent essentiellement de traiter des situations intermédiaires irréalisables par le robot, à cause des erreurs de commande et de perception (contacts par adjacence, obstacles proches $\cdots$). Par exemple, si un contact ne peut être directement relâché à cause de la présence d'un obstacle dans son environnement immédiat, il est conseillé de glisser auparavant sur la surface de contact; cette opération a pour effet de créer de nouveaux nœuds et de nouveaux arcs dans le graphe. De même, la rencontre d'une relation d'adjacence conduira à rechercher des contacts voisins permettant de guider le robot.

Cette approche permet de réduire la complexité algorithmique, en ne développant que les parties significatives du graphe de recherche. Elle permet aussi d'introduire des stratégies locales connues, afin de résoudre certains problèmes de montage caractéristiques (utilisation de stratégies d'insertions par exemple).

3.5.3 Analyse de validité de la solution:

La procédure d'analyse de validité des mouvements retenus, réalise deux types de vérifications:

- Aucune collision n'est susceptible de se produire, compte tenu de l'incertitude initiale de position ε_i et de l'erreur maximum de commande ε_p. Un test de collision est alors réalisé, après avoir "grossi" les obstacles potentiels d'une épaisseur égale à $\varepsilon = 2(\varepsilon_p + \varepsilon_i)$:

  $$Balayage(A, d) \cap Gros_\varepsilon(B) = \emptyset$$

  où $Balayage(A, d)$ est le volume balayé par A au cours du déplacement d, et $Gros_\varepsilon(B)$ représente les obstacles grossis par une épaisseur ε.

- Le contact objectif est toujours réalisé, compte tenu des paramètres ε_p et ε_i précédents. Un test d'interférence est alors réalisé, après avoir réduit les obstacles potentiels d'une épaisseur égale à $\varepsilon = 2(\varepsilon_p + \varepsilon_i)$:

  $$A(p) \cap Gros_\varepsilon^{-1}(B) \neq \emptyset$$

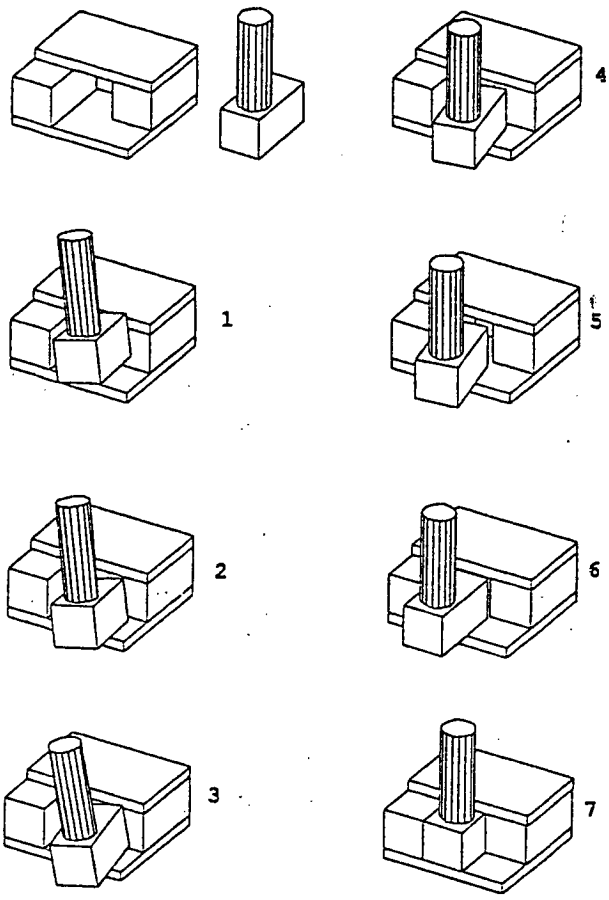
  où p est la position commandée, $A(p)$ est l'objet A en position p, et $Gros_\varepsilon^{-1}(B)$ représente les obstacles réduits par ε.

3.5.4 Synthèse du programme de mouvements fins:

La solution SG retenue à l'issue du traitement précédent est transformée au moyen de règles de ré-écriture, afin d'obtenir un programme exécutable par le robot. Ces règles conduisent à engendrer une instruction de manipulation pour chaque transition qui permet au robot de passer de l'état E_i courant à l'état E_j voisin. Les instructions produites ont la forme suivante:

$$DEPLACER <objet\text{-}A> \; SUIVANT <T> \; EN\text{-}MAINTENANT <C> \; JUSQUA <A>$$

où T et C sont les paramètres de déplacement associés aux arcs issus du nœud s_i traité (transformation géométrique et liste de contacts à maintenir), et A indique tous les contacts qui sont susceptibles de stopper le mouvement. Ces instructions sont codées dans un formalisme de nature géométrique. Elles sont complétées à l'exécution par des valeurs numériques, calculées à partir du modèle géométrique et des termes T, C et A: la position commandée est directement dérivée de la transformation T; les conditions sur les forces sont calculées à partir du vecteur vitesse et des caractéristiques des faces de contact. Une face d'arrêt engendrera par exemple une condition de type "$F_v > seuil$", pour stopper




```
(REALISER-S
    (R-robot SUR R0)
    (VIA (R1 SUR R2)(R3 SUR R4) ..... (Ri SUR Rj)))

(REALISER-C
    (point-P1 SUR face-B1)
    (SUIVANT (TRANSLATION :VECT vecteur-V1)))

(REALISER-C
    (arete-A1 SUR face-B1)
    (SUIVANT (ROTATION  :AXE (make-axe    :PT point-P1
                                          :VECT vecteur-V2)))
    (EN-MAINTENANT (point-P1 SUR face-B1)))

(REALISER-C
    (face-F1 SUR face-B1)
    (SUIVANT (ROTATION  :AXE (make-axe    :PT point-P1
                                          :VECT arete-A1)))
    (EN-MAINTENANT (arete-A1 SUR face-B1)))

(REALISER-C
    (arete-A2 SUR face-B2)
    (SUIVANT (TRANSLATION :VECT vecteur-V3))
    (EN-MAINTENANT (face-F1 SUR face-B1)))

(REALISER-C
    (face-F2 SUR face-B2)
    (SUIVANT (ROTATION  :AXE (make-axe    :PT point-P2
                                          :VECT arete-A2)))
    (EN-MAINTENANT (face-F1 SUR face-B1)(arete-A2 SUR face-B2)))

(REALISER-C
    (point-P3 SUR face-B3)
    (SUIVANT (TRANSLATION :VECT arete-A1))
    (EN-MAINTENANT (face-F1 SUR face-B1)(face-F2 SUR face-B2)))
```

Figure 6: Exemple de stratégie de mouvements fins engendrée par le système.

un déplacement de direction $-v$ (F_v est la composante suivant v de la force de réaction, et v est supposé être contenu dans le cône de frottement). De même, la direction de la force nominale à conserver au cours d'un mouvement compliant simple (c'est à dire d'un mouvement ne mettant pas en jeu des forces qui s'opposent), pourra être obtenue en ad-

ditionnant les normales extérieures aux faces de glissement. Dans la pratique, cette force devra rester dans un petit domaine défini par un cône d'erreurs et par un intervalle de valeurs. Cet aspect du système mériterait d'être amélioré dans le futur, afin d'arriver à une meilleure maîtrise des relations entre planification et exécution. Les travaux de Donald [Donald 86] et de Buckley [Buckley 87] vont dans ce sens.

Remarque: Dans le cas où SG est un sous-graphe non linéaire, les règles de ré-écriture conduisent aussi à engendrer des instructions conditionnelles et des itérations. Ces constructions sont alors associées aux expressions booléennes qui définissent les "conditions d'arrêt" des mouvements appliqués.

3.5.5 Implantation et expérimentations:

Le système de planification de mouvements fins a été implanté en LUCID-LISP sur machine SUN 260. Des expérimentations ont été faites en simulation sur des objets polyédriques comportant moins de 20 faces. Les temps d'exécutions sont relativement longs à cause du volume des calculs et des choix d'implantation. En particulier, nous avons basé la détermination des contacts et de leurs propriétés sur un mécanisme de "démons" [Winston 77]. Ce procédé a l'avantage d'automatiser la gestion des contacts dans le modèle (toute modification de celui-ci entraîne une mise à jour automatique des relations de contact). Il facilite ainsi l'introduction de nouvelles stratégies ou de nouvelles fonctions dans le système. Il possède par contre l'inconvénient d'augmenter de manière importante les temps de traitement.

Considérons par exemple le montage présenté dans la figure 6. Les objets concernés possèdent respectivement 8 et 12 faces. Le traitement de ce montage a nécessité 9 minutes de temps CPU. Il a donné lieu à un graphe d'états comportant 50 nœuds et 80 arcs. Le programme synthétisé est alors constitué de six instructions de manipulation.

4 Vérification/correction de plans:

4.1 Présentation du problème et de l'approche:

Le problème posé consiste à s'assurer que le plan d'actions engendré par le module de planification, vérifie toutes les contraintes d'incertitudes imposées par la tâche. Ce problème vient de ce que les variables de position spécifiées dans le plan, ne sont jamais connues de manière exacte lors de la planification. Le système est donc amené à raisonner sur des valeurs nominales, auxquelles sont associées des termes d'erreurs. Une variable X est alors représentée par un couple (X^*, ε), dans lequel X^* est la position nominale de l'objet considéré, et ε est le terme d'erreur. Seule la valeur X^* est utilisée par le module de planification. La fonction de base du système de vérification/correction est donc de déterminer toutes les valeurs possibles des variables X (c'est à dire des couples (X^*, ε)), afin de vérifier que ces valeurs sont "compatibles" en tout point du plan avec les contraintes imposées par la tâche.

Dans un formalisme ensembliste, nous dirons que X prend ses valeurs dans un ensemble $Poss_i(X^*)$, après que la i-ème action du plan ait été exécutée. Une contrainte $C_i(X^*)$ portant sur la variable X, spécifie alors l'erreur maximum qui peut être commise relativement à X^*, sans risquer de mettre en échec l'opération projetée. Par exemple, l'erreur sur la position de la pièce à assembler doit être inférieure aux jeux de montage. Nous dirons alors que X *doit* appartenir à un ensemble $Adm_i(X^*)$. Une formulation intuitive du problème de vérification peut ainsi être énoncée comme suit: *l'action A_i est "compatible" avec les contraintes imposées par la tâche, ssi l'ensemble $Poss_i(X^*)$ est inclus dans l'ensemble $Adm_i(X^*)$, pour chaque variable X modifiée par A_i.*

Tous le problème consiste alors à calculer les ensembles $Poss_i(X^*)$. La principale difficulté provient de ce que ces ensembles dépendent de la nature des actions exécutées antérieurement. Par exemple, l'erreur portant sur la position X d'un objet saisi et transporté par le robot, comportera un terme lié à l'incertitude initiale de position de la pince par rapport à l'objet, et un terme lié à l'imprécision de la commande de mouvement. Le mécanisme de calcul associé devra donc procéder par *propagation* de ces données à travers les actions du plan.

Les bases formelles de ce problème sont développées dans [Troccaz, Puget 87] et dans [Puget 88]. Les paragraphes qui suivent décrivent le principe des calculs mis en oeuvre. Ces calculs conduisent à rechercher les contraintes d'incertitudes non satisfaites ainsi que les origines possibles des échecs, afin d'apporter les amendements nécessaires au plan. Le procédé appliqué repose sur un modèle formel de propagation de contraintes, permettant de calculer les valeurs d'incertitudes en n'importe quel point du plan. Le processus itératif de vérification/correction appliqué, opère alors suivant deux modes illustrés par la figure 7 [Puget, Troccaz 86]:

- Un mode de propagation "descendant", qui permet de vérifier que les incertitudes obtenues en n'importe quel point du plan sont "compatibles" avec les contraintes imposées par la tâche. Le mécanisme correspondant calcule à chaque pas les incertitudes résultantes, à partir des incertitudes présentes avant l'exécution de l'action.

- Un mode de propagation "ascendant", qui est utilisé en cas d'échec pour rechercher les causes possibles de cet échec. Le procédé appliqué opère par régression, afin de calculer les contraintes d'incertitude qui auraient dû être vérifiées dans les étapes antérieures du plan.

4.2 Modélisation des incertitudes et des opérateurs associés:

4.2.1 Représentation des erreurs de position:

La position d'un objet A est classiquement définie par la transformation géométrique T_a qui permet de passer du repère de référence de l'univers au repère R_a de l'objet. Cette position représente une donnée théorique (appelée "position nominale") qui n'est jamais atteinte dans la réalité. Une *erreur* sur une position se traduit par un "écart" entre la position nominale et la position réelle. Compte tenu du formalisme utilisé pour localiser les objets dans l'univers du robot, cette erreur définit une transformation géométrique ϵ_a telle que:

$$R_a = Base \star T_a \star \epsilon_a \qquad (1)$$

De la même manière, les positions relatives de deux objets A et B sont définies comme suit:

$$\begin{array}{ll} \text{position nominale:} & R_b = R_a \star T_{ab} \\ \text{position réelle:} & R_b = R_a \star T_{ab} \star \epsilon_{ab} \end{array} \qquad (2)$$

dans lesquelles T_{ab} représente la transformation géométrique qui permet de passer de R_a à R_b, et ϵ_{ab} est l'erreur de position du repère R_b par rapport au repère R_a.

Raisonner sur les positions des objets conduit donc à raisonner sur des repères liés par des transformations. Chacune de ces transformations comporte une composante nominale bien définie, et une composante variable qui représente le terme d'erreur. Suivant le type de raisonnement considéré, le procédé de calcul appliqué opère avec l'une ou l'autre de ces données. Dans le cas de la propagation symbolique des incertitudes, c'est le terme d'erreur qui est exploité.

4.2.2 Représentation des incertitudes de position:

Dans la représentation ensembliste que nous avons retenue, l'incertitude de position d'un repère R_b par rapport à un autre repère R_a est représentée par l'ensemble des transformations géométriques ϵ_{ab} qui permettent d'obtenir une position réelle de R_b en appliquant l'équation (2). Cette incertitude est notée $I(R_b/R_a)$. Elle définit un sous ensemble de $\Re^3 \times SO^3$, dans lequel $\Re^3$ est l'espace des translations et SO^3 représente le groupe des rotations orthogonales.

Le principe de modélisation que nous avons choisi, conduit à considérer une erreur comme l'association de trois données: un vecteur de translation, un vecteur unitaire représentant l'axe de rotation, et un angle de rotation. Une incertitude de position peut

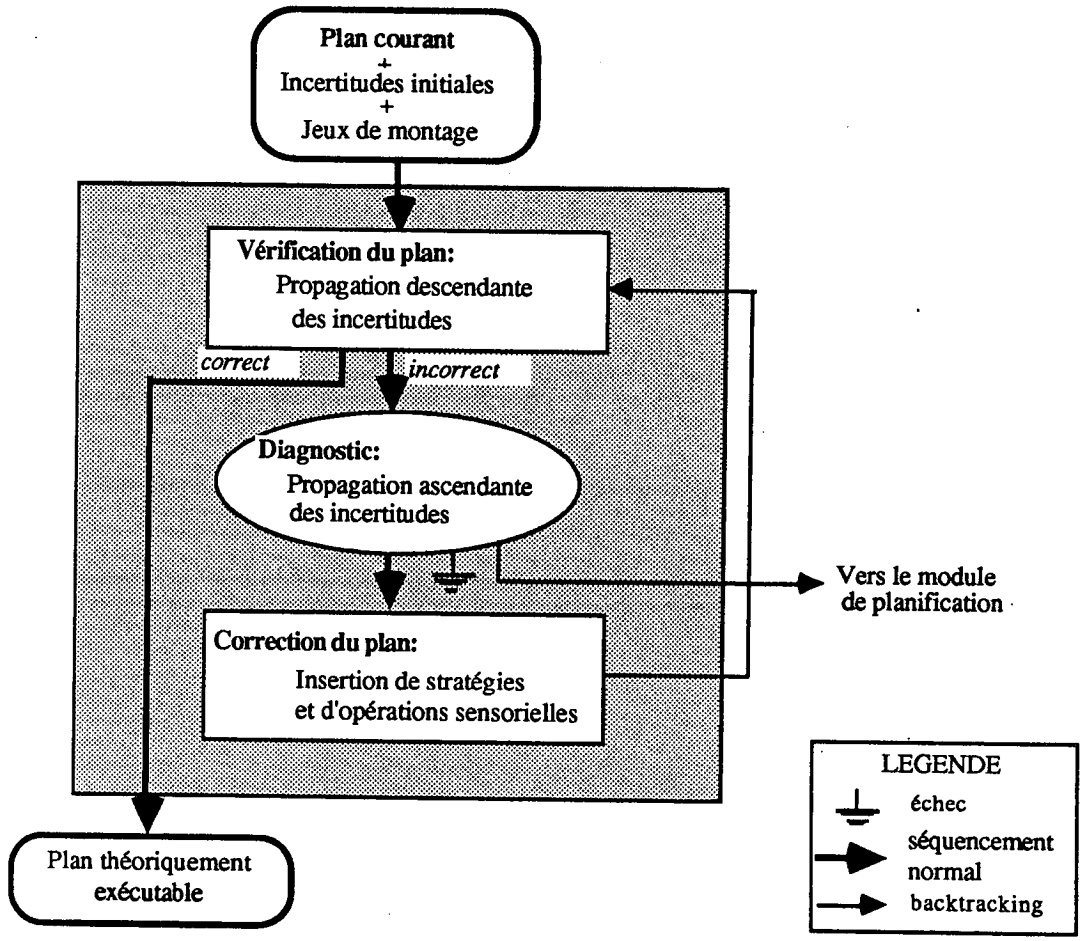


Figure 7: Principe de vérification/correction de plans.

alors être vue comme un sous-ensemble du produit cartésien $\Re^3 \times S(1) \times [-\pi \ + \pi]$, où $S(1)$ représente la sphère unité [Puget 85]:

$$I(R_b/R_a) = Tr \times U \times A$$

Dans la pratique, les sous-ensembles Tr, U et A ne sont pas quelconques du fait qu'ils proviennent de situations physiques mettant en jeu des relations de contact qui possèdent une certaine "isotropie" vis à vis des incertitudes. Ainsi, le terme Tr peut être représenté par des volumes de type sphère, disque ou segment de droite. Les rotations sont plus complexes à manipuler. Mais la réalité des contacts d'assemblage (contacts de nature essentiellement surfacique) conduit à fixer très souvent deux degrés de liberté en rotation; l'incertitude d'orientation résultante est alors localisée autour d'un axe. C'est pourquoi nous avons choisi d'approximer une incertitude par le plus petit ensemble englobant du type:

$$Tr \times U \times A$$

avec:
Tr = sphère, disque ou segment de droite
U = sphère ou vecteur unique
A = intervalle $[-\alpha\ +\alpha]$, α petit

Par exemple, un objet reposant sur un plan z=Cste présente une incertitude de position qui peut être représentée comme suit (cf. figure 8):

Tr = disque de rayon δ et d'axe z
U = vecteur unitaire porté par l'axe z
$A = [-\alpha\ +\alpha]$

Si le contact est réalisé suivant une surface cylindrique d'axe z, l'incertitude de position peut être définie par les ensembles suivants (cf. figure 8):

Tr = segment porté par z, de longueur 2δ, et centré sur la position nominale de l'objet
U = vecteur unitaire porté par l'axe z
$A = [-\alpha\ +\alpha]$

Dans le cas d'un objet évoluant dans l'espace libre (outil terminal du robot en particulier), les volumes d'incertitudes sont maximums car aucun contact ne permet d'en réduire certaines composantes:

Tr = sphère de rayon δ centrée sur la position nominale de l'objet
U = sphère unitaire $S(1)$
$A = [-\alpha\ +\alpha]$

Remarque: Ce type d'approximation se justifie par l'importante simplification qu'il apporte au niveau des calculs de propagation des incertitudes. Il se traduit par contre par une perte d'information et par une surcontrainte des incertitudes dans certaines situations physiques (contacts linéiques en particulier). Dans l'assemblage mécanique, de telles situations représentent très souvent une étape intermédiaire vers la réalisation de situations plus robustes, par exemple: passage d'un contact linéique à un contact surfacique. L'approximation appliquée est alors sans réelle conséquence sur la suite, du fait de la réduction d'incertitude apportée par le contact surfacique.

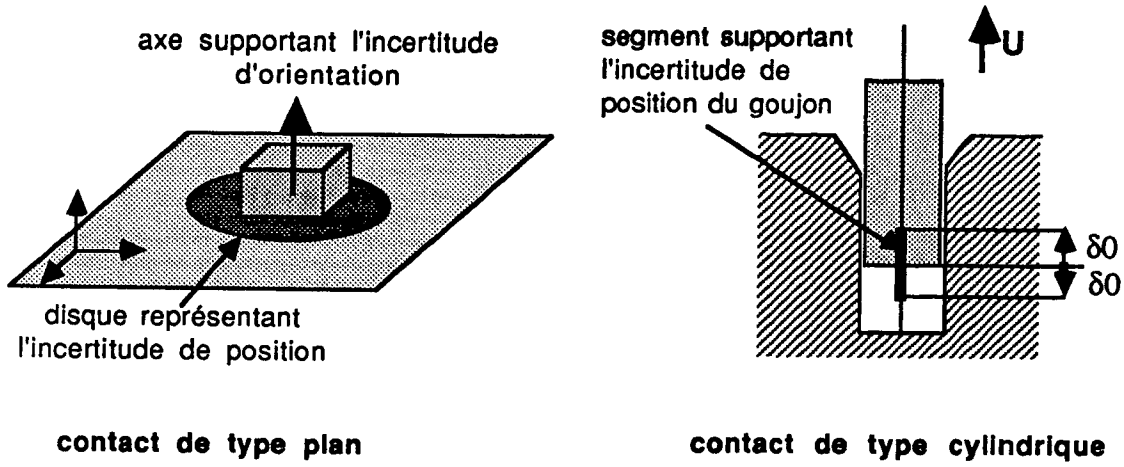


Figure 8: Représentation des incertitudes de position associées à un contact de type plan et à un contact de type cylindrique.

4.2.3 Structure de l'univers du robot:

Les structures spatiales utilisées en robotique combinent toujours une représentation des positions des objets avec une représentation des relations qui lient ces objets entre-eux. Un *état de l'univers* du robot est alors représenté par un graphe orienté, dans lequel chaque noeud symbolise un repère de localisation d'objet, et chaque arc représente une transformation géométrique complétée par l'incertitude qui lui est associée. La structure de base de ce graphe est une arborescence ayant pour racine le repère de référence Ro de l'univers. Elle permet de définir la position de chaque objet dans un référenciel fixe connu du robot. Les autres arcs représentent des relations entre objets, créées par des opérations du robot (perception, mise en contact ou mise en correspondance d'entités géométriques). Ils permettent de raisonner sur les positions relatives des objets concernés.

Cette structure dynamique est fondamentale pour le traitement des incertitudes, car elle permet de les calculer aux endroits où elles sont significatives. Par exemple, la réalisation d'un contact entre deux objets A et B se traduit par une *réduction de l'incertitude relative* de A et de B. Il est donc naturel de représenter explicitement cette information dans la structure, et d'en propager ensuite les effets sur les autres arcs concernés, par exemple (cf. figure 9):

Structure avant contact:

$Arc(R_o, R_a) = \{T_{oa}, I(R_a/R_o)\}$
$Arc(R_o, R_b) = \{T_{ob}, I(R_b/R_o)\}$

Structure après contact:

$Arc(R_o, R_a)$ = supprimé

$$Arc(R_b, R_a) = \{T_{ba}, I(R_a/R_b)\}$$
$$Arc(R_o, R_b) = \{T_{ob}, I(R_b/R_o)\}$$

Dans cette modification du modèle, le calcul de la transformation T_{ab} est exécuté en composant la transformation associée au déplacement exécuté, avec celles rencontrées sur le chemin reliant les objets A et B dans le graphe antérieur. Le calcul de l'incertitude $I(R_a/R_b)$ est exécuté de manière similaire, en utilisant des opérateurs appropriés (cf. paragraphe suivant). La suppression du lien $Arc(R_o, R_a)$ est due au fait que la valeur de l'incertitude $I(R_a/R_o)$ dépend alors directement des termes $I(R_a/R_b)$ et $I(R_b/R_o)$.

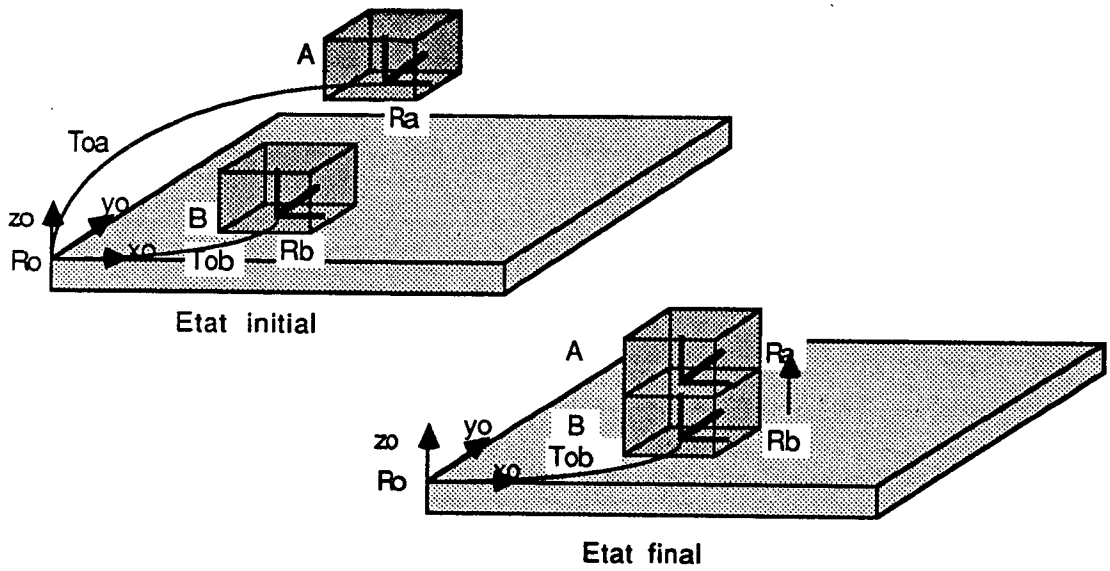


Figure 9: Exemple de modification des arcs du modèle après réalisation d'un contact.

4.2.4 Opérateurs associés au modèle:

Quatre types d'opérateurs géométriques sont nécessaires pour combiner et propager les incertitudes au travers de la structure précédente: les opérateurs d'inversion, de composition, de projection et de conjonction. Les deux premiers opérateurs sont utilisés par le système pour calculer les incertitudes qui résultent de la combinaison de données déjà présentes dans le modèle. L'opérateur de projection est employé pour réduire les composantes d'incertitudes affectées par des relations de contact. Celui de conjonction permet de calculer l'incertitude résultante, après application d'une opérations sensorielle.

- *Inversion d'incertitudes*: On note I^{-1} l'opérateur d'inversion permettant de calculer $I(R_b/R_a)$ en fonction de $I(R_a/R_b)$:

$$I(R_b/R_a) = I^{-1}(R_a/R_b)$$

- *Composition d'incertitudes*: On note $*$ l'opérateur de composition permettant de calculer $I(R_a/R_c)$ à partir de $I(R_a/R_b)$ et de $I(R_b/R_c)$:

$$I(R_a/R_c) = I(R_a/R_b) * I(R_b/R_c)$$

- *Projection d'incertitudes*: On note $Proj(I(R_a/R_b),(E,V,B))$ l'opérateur permettant de "projeter" l'incertitude $I(R_a/R_b)$ sur le sous ensemble $E \times V \times B$ de l'espace $\Re^3 \times S(1) \times [-\pi \ +\pi]$:

$$I^*(R_a/R_b) = Proj(I(R_a/R_b),(E,V,B))$$

Tr^*, U^*, A^* sont respectivement les projections de Tr sur E, de U sur V et de A sur B.
$I(R_a/R_b)$ est l'incertitude présente dans le modèle avant réalisation de la relation de contact.

- *Conjonction d'incertitude*: On note $\wedge$ l'opérateur permettant de calculer la conjonction de deux incertitudes portant sur un même arc $Arc(R_b,R_a)$ du modèle:

$$I^*(R_a/R_b) = I(R_a/R_b) \wedge I_{capteur}(R_a/R_b)$$

Tr^*, U^*, A^* sont les intersections des ensembles correspondants de $I(R_a/R_b)$ et de $I_{capteur}(R_a/R_b)$.
$I(R_a/R_b)$ est l'incertitude présente dans le modèle avant utilisation du capteur; $I_{capteur}(R_a/R_b)$ est l'incertitude qui résulte de la mesure faite par le capteur.

Les calculs sur lesquels reposent ces quatre opérateurs, mettent en jeu des opérations qui portent sur des ensembles de vecteurs: somme de Minkowski, intersection, application d'une rotation sur un ensemble, application d'un ensemble de rotations sur un ensemble de vecteurs... Une description détaillée de ces calculs n'étant pas nécessaire pour la compréhension de la suite, nous renvoyons le lecteur intéressé par cet aspect à [Puget 88]. Dans la pratique, tous ces calculs sont réalisés à partir des ensembles élémentaires (sphères, disques et segments de droites) utilisés pour approximer les volumes d'incertitudes.

Prenons par exemple le cas d'un cube A que le robot vient poser sur un cube B placé sur une table (cf. figure 6.5). Soient R_a et R_b les repères associés respectivement aux objets A et B; R_o est le repère de référence, choisi de manière à ce que le plan X_oOY_o coïncide avec la surface de la table. Si l'on fait abstraction du robot, les informations codées dans le modèle avant exécution de l'opération, sont les suivantes:

$$Arc(R_o,R_b) = \{T_{ob},\ I(R_b/R_o) = D(\delta,z_o) \times \{z_o\} \times [-\alpha \ +\alpha]\}$$
$$Arc(R_o,R_a) = \{T_{oa},\ I(R_a/R_o) = S(\delta') \times S(1) \times [-\alpha' \ +\alpha']\}$$

où $D(\delta,z_o)$ est un disque de rayon δ et d'axe z_o, et $S(\delta')$ est une sphère de rayon δ' centrée sur la position nominale de R_a.

Supposons pour simplifier, que le déplacement du cube A n'introduit aucune erreur supplémentaire. après exécution de l'opération, le modèle est alors transformé comme suit:

$Arc(R_o, R_b)$ = inchangé
$Arc(R_o, R_a)$ = supprimé
$Arc(R_b, R_a) = \{T_{ba},\ I^*(R_a/R_b)\}$

avec:
$I(R_a/R_b) = I(R_a/R_o) * I^{-1}(R_b/R_o)$
$I(R_a/R_b) = \{S(\delta') \times S(1) \times [-\alpha' + \alpha']\} * \{D(\delta, z_o) \times \{z_o\} \times [-\alpha + \alpha]\}$
$I(R_a/R_b) = S(\delta + \delta') \times S(1) \times [-(\alpha' + \alpha) + (\alpha' + \alpha)]$

$I^*(R_a/R_b) = Proj(I(R_a/R_b), (plan\ x_o O y_o, \{z_o\}, [-\pi + \pi]))\}$
$I^*(R_a/R_b) = D(\delta' + \delta, z_o) \times \{z_o\} \times [-(\alpha' + \alpha) \quad (\alpha' + \alpha)]$

4.3 Modélisation des actions du robot:

4.3.1 Caractéristiques des actions traitées par le système:

Les actions de manipulation utilisées dans la spécification du plan, sont exprimées dans un formalisme géométrique qui présente un niveau sémantique plus riche et plus homogène que celui offert par les langages traditionnels de programmation de robots. Deux objectifs ont guidé le développement de ce formalisme [Laugier 87]: éviter les ambiguïtés d'interprétation des situations rencontrées par le robot, et décrire des opérations dont les effets sont prédictibles en termes de positions et d'incertitudes.

Compte tenu des modèles incomplets que nous possédons, la propriété de "prédictibilité" ne peut être vérifiée que par des actions simples ayant les caractéristiques suivantes:

- L'action possède un *objectif unique*, exprimé sous la forme d'une relation géométrique ou d'une relation de contact. Dans le premier cas, il s'agit de déplacer le robot vers une position de l'espace libre; dans l'autre cas, le système doit engendrer un mouvement ayant pour objet de créer ou de détruire le (les) contact(s) spécifié(s).
- L'action ne met en oeuvre *aucune opération compliante*, conduisant à glisser sur des surfaces d'appuis.

Les actions actuellement prises en compte par le mécanisme de vérification/ correction, se limitent donc aux opérations suivantes: déplacement dans l'espace libre, saisie et lâcher d'objets, localisation d'objets. La structure des plans traités est alors globalement linéaire. Les constructions plus complexes associées aux stratégies de mouvements fins, sont dans ce cas considérées comme "globalement correctes". Cette approche est cohérente avec le fait que ces stratégies ont toutes été engendrées sur la base d'une analyse locale des incertitudes liées aux opérations concernées. De telles constructions sont considérées par le système comme des *macro-actions*, permettant de réaliser des relations d'assemblage avec une incertitude finale inférieure aux jeux de montage.

4.3.2 Principe de modélisation:

Le formalisme géométrique utilisé pour décrire les actions de manipulation s'appuie sur des éléments simples du modèle de l'environnement (points, droites, plans, faces, repères). Nous définirons la sémantique d'une action par les modifications que cette action apporte sur deux composantes du modèle:

- Les positions (positions nominales et incertitudes associées) des éléments géométriques impliqués dans la description.
- Les relations spatiales qui lient ces éléments dans le modèle.

Plusieurs facteurs interviennent dans la spécification de ces modifications: la précision avec laquelle le robot exécute ses déplacements, la précision de mesure des capteurs utilisés, et la transformation des relations de contact (création ou suppression). Prenons par exemple une action de déplacement de type "*REALISER A SUR B*". On note $T(R_i/R_j)$ la position du repère R_i par rapport au repère R_j, et $I(R_i/R_j)$ l'incertitude de position associée. Cette action à pour effet de modifier le modèle comme suit (cf. figure 10):

$$T(Robot/Base) = T(R_b/Base) * T(R_a/R_b) * T^{-1}(R_a/Robot)$$
$$I(Robot/Base) = I_{dep}$$

où *base*, *Robot*, R_a et R_b sont respectivement les repères de référence de l'univers, de la pince du robot, de l'objet contenant l'entité A, et de l'objet contenant l'entité B; I_{dep} est l'incertitude de position liée à la commande de mouvement. Ces expressions sont calculées à l'aide des fonctions décrites précédemment.

Dans le cas où l'action met en jeu des contacts, les modifications induites se traduisent par une projection de volume d'incertitude, par une modification des structures spatiales associées. Par exemple, l'action "*SAISIR <prise>*" conduira à appliquer les opérations suivantes sur le modèle (cf. figure 11):

- Suppression de l'arc $ARC(Base, R_a)$ et création de l'arc $ARC(Robot, R_a)$.
- Calcul des paramètres de $ARC(Robot, R_a)$:
  $T(R_a/Robot) = T^{-1}(Robot/Base) * T(R_a/Base)$
  $I(R_a/Robot) = Proj(I_{dep}^{-1} * I(Ra/Base), (E, V, B))$
  avec:
  E= plan des mors, V= vecteur normal aux mors, $B = [-\pi \ \ +\pi]$

Une action perceptive destinée à localiser une pièce A dans le champ de la caméra, conduira à modifier les paramètres de position et d'incertitude associés à A. Cette modification est réalisée à l'aide de l'opérateur de "conjonction d'incertitude". Elle pose cependant quelques problèmes théoriques qui sont développés dans [Puget 88].

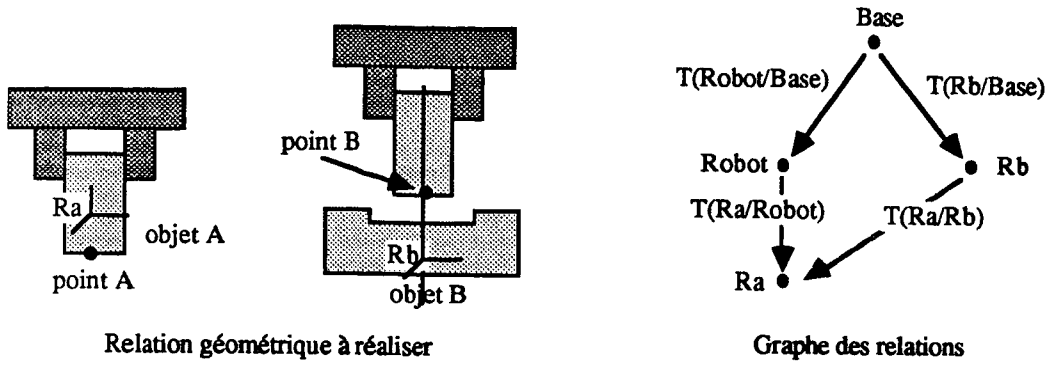


Figure 10: Modifications apportées au modèle par une action de déplacement du type: *REALISER A SUR B.*

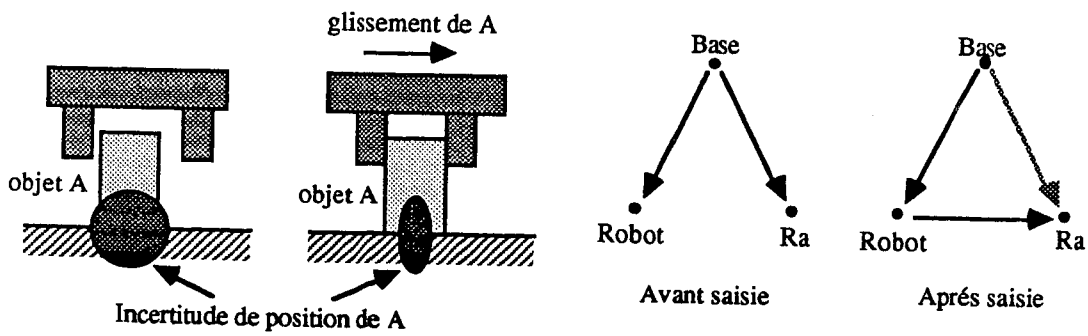


Figure 11: Modification des incertitudes de position à la suite d'une opération de saisie.

4.4 Principe de vérification/correction:

4.4.1 Concept de plan contraint:

Conformément aux hypothèses énoncées précédemment, un plan de manipulation peut être vu comme une séquence d'actions A_1, $A_2 \cdots A_n$, conduisant à faire passer progressivement l'univers du robot d'un état initial E_0 à un état final E_n:

$$E_0 \xrightarrow{A_1} E_1 \xrightarrow{A_2} E_2 \cdots E_{n-1} \xrightarrow{A_n} E_n$$

Chaque état se différencie du précédent par les modifications apportées aux éléments du modèle, c'est-à-dire aux couples (position nominale, incertitude). Les actions du plan sont celles définies dans le paragraphe précédent (actions simples et macro-actions).

Une première nécessité induite par le problème de vérification, consiste à faire apparaître explicitement toutes les contraintes de la tâche au niveau de la spécification du plan. Ces contraintes représentent des conditions qui doivent impérativement être vérifiées par les variables de position (terme nominal et/ou terme d'incertitude), pour que les actions qui suivent soient exécutables, ou pour que certains objectifs de la tâche soient considérés comme atteints.

Chaque contrainte formulée s'exprime relativement à un état E_k particulier. Elle porte de ce fait sur la valeur dans E_k, des variables de position concernées. Si $I(R_a/R_b)$ est une incertitude définie par une expression conforme au formalisme développé, l'évaluation de cette expression dans l'état E_k donne un volume d'incertitude que nous noterons $I_k(R_a/R_b)$. La formulation d'une contrainte portant sur les positions relatives dans E_k des objets A et B, pourra alors être représentée par les expressions suivantes:

$$T(R_a/R_b) = t_0$$
$$I_k(R_a/R_b) < I_0 \qquad \text{avec:} \quad I_0 = Tr_0 * U_0 * A_0$$

où $T(R_a/R_b)$ et t_0 sont les transformations géométriques qui caractérisent les positions nominales de R_a par rapport à R_b, et la relation d'ordre utilisée pour comparer les incertitudes est définie comme suit:

$$I_1 > I_2 \iff (Tr_1 \subset Tr_2) \wedge (U_1 \subset U_2) \wedge (A_1 \subset A_2)$$

Définition 4.1 *Soit CO_k ($k = 0 \cdots n$) les contraintes imposées dans l'état E_k. Nous appelons "plan contraint" toute séquence alternée d'actions et de contraintes portant sur les variables de position modifiées par ces actions. Nous représenterons un tel plan par l'expression suivante:*

$$[CO_0]\, A_1\, [CO_1]\, A_2 \cdots [CO_{n-1}]\, A_n\, [CO_n]$$

Définition 4.2 *Un plan contraint est déclaré correct relativement à un état initial E_0, ssi toutes les contraintes de CO_k, pour k variant de 0 à n, sont satisfaites dans chaque état E_k résultant de l'exécution des actions $A_1 \ldots A_{k-1}$.*

4.4.2 Propagation descendante des contraintes et vérification du plan:

On note C_i les conditions vérifiées dans l'état E_i, par les variables de position utilisées dans le plan. Ces conditions résultent de l'exécution des actions $A_1 \cdots A_{i-1}$; elles peuvent aussi dériver directement de conditions présentes dans l'état initial E_0. Par exemple, l'incertitude de position $I(R_a/Base)$ d'un objet A peut provenir des mécanismes d'alimentation employés pour placer A, d'une localisation par un capteur, de manipulations réalisées par des actions antérieures, ou encore d'un mélange des trois.

Afin de calculer les C_i, le système est donc amené à propager symboliquement les contraintes d'incertitudes de l'état E_0 vers l'état E_n. Ce mode de propagation est appelé *propagation descendante*. Il conduit à calculer à chaque étape les plus fortes conditions qui portent sur les variables de position. Ces conditions sont alors directement dérivées de la sémantique de l'action exécutée, et des conditions qui étaient antérieurement vérifiées. Nous parlerons de *plus fortes postconditions* $POST_i(C)$, associées aux conditions C et à l'action A_i. Par exemple, la condition "$I(R_a/Base) < i_0$" sera transformée en "$I(R_a/Base) < i_0 + I_{dep}$", par l'action *"DEPLACER A DE T"*. Dans le cas des actions de manipulation que nous avons définies, ce calcul est réalisé sur la base des expressions décrites précédemment.

Le procédé de propagation permettant de calculer les C_i est alors le suivant:

C_0 = Conditions initiales de E_0
Pour $i = 1 \cdots n$: $\quad C_i = POST_i(C_{i-1})$

Le plan contraint est déclaré correct, ssi toutes les conditions de C_i impliquent celles de CO_i, pour i variant de 0 à n.

Remarques: Le procédé de calcul sur lequel repose le processus de vérification de plans, permet de garantir qu' un plan déclaré correct, l'est réellement (le calcul des plus fortes postconditions étant toujours basé sur les hypothèses les plus défavorables en ce qui concerne les erreurs). Il peut, par contre conduire à déclarer incorrect un plan qui ne l'est pas, à cause des défauts de modélisation et des phénomènes de compensation d'erreurs. Il convient cependant de noter qu'un plan déclaré "correct" n'est pas obligatoirement exécutable par le robot, à cause des aléas physiques non modélisés (glissements par exemple). Seule une phase dite "d'apprentissage" permet alors d'adapter complètement le plan produit à son environnement réel d'exécution.

4.4.3 Propagation ascendante des contraintes et amendement du plan:

Le principe de la propagation ascendante des contraintes consiste à "régresser" dans le plan, afin d'établir les contraintes qui devraient être vérifiées dans les états antérieurs pour que l'échec rencontré soit levé. Ce mode de propagation repose sur le calcul des *plus faibles préconditions* $PRE_i(C)$ associées aux conditions C et aux actions A_i. En d'autres termes, il s'agit de déterminer à chaque étape les plus faibles conditions $PRE_i(C)$ qui permettent de garantir que C sera toujours vérifié après exécution de A_i. Par exemple, la condition "$I(R_a/Base) < i_0$" sera toujours vérifiée après exécution de l'action *"DEPLACER A*

DE T", si $I(R_a/Base)$ vérifie la condition "$I(R_a/Base) + I_{dep} < i_0$" dans l'état qui précède l'exécution de l'action. Comme précédemment, le mode de calcul des plus faibles préconditions représente une partie de la sémantique des actions de manipulation.

On note C_j^i, $j < i$, les contraintes de E_j dérivées de CO_i par le processus de propagation ascendant des contraintes. Ces contraintes garantissent si elles sont vérifiées, que la séquence $A_{j-1} \cdots A_i$ est "correcte" (et en particulier que les contraintes de CO_i sont satisfaites). Elles sont calculées en appliquant l'algorithme suivant:

$C_i^i = CO_i$
Pour $j = i - 1 \cdots 0$: $\quad C_j^i = PRE_{j-1}(C_{j-1}^i) \wedge CO_j$

Le procédé décrit ci-dessus permet de calculer les contraintes qu'il faudrait satisfaire dans les différents états antérieurs à celui où l'echec a été détecté. Il ne permet cependant pas de déterminer *où* corriger ni *comment* corriger.

Il est donc nécessaire de choisir dans un premier temps, l'état E_k qui paraît être le plus propice à la correction. Une possibilité consiste à rechercher pour chaque contrainte à satisfaire, l'endroit où la différence entre l'état souhaité et l'état obtenu est la plus facile à maîtriser. Il n'existe cependant (dans l'état actuel de nos connaissances) aucune règle précise qui permette de réaliser ce choix.

Le deuxième problème consiste à rechercher parmi les corrections possibles, celle qui semble être la mieux adaptée à la situation. Chaque correction apportée possède alors un caractère local. Elle se traduit, soit par une modification de l'état initial de certaines variables, soit par l'insertion d'une procédure de correction appelée "rustine". Le premier type de correction est appliqué en priorité lorsqu'il est possible; il conduit à contraindre un peu plus l'environnement initial de la tâche. L'autre type de modification se traduit par l'insertion d'une action ou d'une macro-action permettant de réduire certaines incertitudes; le choix réalisé est alors conditionné par plusieurs facteurs: conditions d'applicabilité de la rustine, nature de l'incertitude à réduire et contraintes à satisfaire. Cet aspect est encore mal maîtrisé.

5 Conclusion:

Nous avons présenté dans cet article de quelles manières interviennent les incertitudes dans le processus de programmation d'un robot de manipulation. Nous avons ainsi montré qu'il était possible de distinguer deux catégories d'incertitudes de position: celles qui peuvent être traitées localement par des stratégies appropriées, et celles qui nécessitent un traitement global. Après une étude générale des techniques employées pour résoudre ces problèmes, nous avons proposé des solutions permettant d'une part de synthétiser automatiquement des programmes de mouvements fins, et d'autre part de procéder à une phase vérification/ correction sur les plans engendrés. Ces solutions sont celles que nous avons développées dans le cadre du projet SHARP, dont l'objectif est de construire un système intégré de programmation automatique de robots. Une première version expérimentale de ce système est actuellement en cours de test sur le site robotique du LIFIA. Elle a été implantée en LUCID-LISP sur machine SUN 260.

L'aspect le plus avancé en ce qui concerne le traitement des incertitudes, est celui de la synthèse automatique de mouvements fins de montage. Le logiciel développé a été testé en simulation sur plusieurs objets de formes polyédriques. Les résultats obtenus sont alors visualisés à l'aide des fonctions graphiques du système LISP-3D [Laugier 82]. D'autres tests en grandeur réelle sont en cours sur un robot SCEMI six axes controllé par une armoire de commande ITMI.

Le module de vérification/correction de plans n'a par contre donné lieu qu'à des implantations partielles. En particulier, l'aspect concernant l'amendement de plan n'a pas encore été réellement abordé. Il est donc impossible de donner des résultats expérimentaux complets sur ce point.

Remerciements:

Le contenu de cet article repose pour une large part sur les travaux de P.Theveneau (synthèse des mouvements fins) et de P.Puget (modélisation et propagation des incertitudes). Ces travaux s'inscrivent dans le cadre du projet SHARP du Laboratoire LIFIA. Ils ont été en partie supportés par le projet ARA (Automatique et Robotique Avancées) du CNRS et par l'Agence de l'Informatique.

References

[Bolle, Cooper 86] R.M.Bolle, D.B. Cooper: *"On optimally combining pieces of information, with application to estimating 3D complex object position from range data"*, IEEE Trans. Pattern Anal. Machine Intell., PAMI-8, pp. 619-638, Sept. 1986.

[Boyse 79] J.W.Boyse: *"Interference detection among solids and surfaces"*, CACM, vol.22, nb.1, Janvier 1979.

[Brooks 82] R.A.Brooks: *"Symbolic error analysis and robot planning"*, International Journal of Robotics Research, vol 1, no 4, Decembre 1982.

[Brooks 84] R.A.Brooks: *"Aspects of mobile robot visual map making"*, 2nd International Symposium of Robotics Research, Kyoto, Août 1984.

[Buckley 87] S.J.Buckley: *"Planning and teaching compliant motion strategies"*, Ph.D Thesis, Artificial Intelligence Laboratory, MIT, Janvier 1987.

[Chatila, Laumond 85] R.Chatila, J.P.Laumond: *"Position referencing and consistent world modeling formobile robots"*, Proc. IEEE Int. Conf. Robotics and Automation, St.Louis, 1985.

[Cohen, Feigenbaum] P.R.Cohen, E.A.Feigenbaum: *"The handbook of Artificial Intelligence"*, Tome 3, pp 513/563, William Kaufmann Inc.

[Donald 86] B.Donald: *"Robot motion planning with uncertainty in the geometric models of the robot and environnement: a formal framework for error detection and recovery"*, PROC. IEEE Int. Conf. on Robotics and Automation, San Francisco, Avril 1986.

[Dufay 83] B.Dufay: *"Apprentissage par induction enRobotique: Application à la synthèse de programmes de montage"*,Thèse de 3ème cycle, INPG, Grenoble, Juin 1983.

[Dufay, Latombe 84] B.Dufay, J.C.Latombe: *"An approach to automatic robot programming based on inductive learning"*, Rapport de recherche IMAG no 433, Février 1984/ Publié aussi dans:1st International Symposium on Robotics Research, Bretton Woods, Août 1983.

[Durrant-Whyte 87] H.F.Durrant-Whyte: *"Consistant integration and propagation of disparate sensor observations"*, Int. Journal of Robotics Research, vol.6, nb.3, 1987.

[Erdmann 84] M.Erdmann: *"On motion planning withuncertainty"*, AI-TR-810, Artificial Intelligence Laboratory, MIT,1984.

[Faugeras, Hebert 86] O.D.Faugeras,M.Hebert: *"The representation, Recognition, and locating of 3D Objects"*, Int. Journal of Robotics Research, vol.5, nb.3, 1986.

[Gandon 86] C.Gandon: *"Introduction de la compliance dans la programmation des robots"*, Thèse de 3ème Cycle, INPG, Grenoble, Octobre 1986.

[Giraud 83] M.Giraud: *"Generalized active compliance for part-mating with assembly robots"*, 1st International Symposium of Robotics Research, Bretton Woods, 1983.

[Ijel 86] A.Ijel: *"Modélisation des aspects physiques de l'univers d'un robot"*, Rapport de DEA, INPG, Septembre 1986.

[Inoue 74] H.Inoue: *"Force feedback in precise assembly tasks"*, Artificial Intelligence Laboratory, MIT, AIM-308, Août 1974.

[Joyal, Provost 66] M.Joyal, P.Provost: *"Statique"*, Masson & Cie, 1966.

[Latombe, Laugier 85] JC.Latombe, C.Laugier: *"Les systèmes de programmation de robots"*, MICAD-85, Paris, Février 1985.

[Laugier, Dufay 82] C.Laugier, B.Dufay: *"Geometrical reasoning in automatic grasping and contact analysis"*, PROLAMAT 82, Leningrad, Mai 1982.

[Laugier 82] C.Laugier: *"LISP-3D: logiciel graphique pour la manipulation et la visualisation de scènes tridimensionnelles"*, Rapport de Recherche IMAG, no 328, Septembre 1982.

[Laugier, Troccaz 85] C.Laugier, J.Troccaz: *"SHARP: A system for automatic programming of manipulation robots"*, 3rd International Symposium of Robotics Research, Gouvieux, Octobre 1985.

[Laugier, Theveneau 86] C.Laugier, P.Theveneau: *"Planning sensor-based motions for part-mating using geometric reasonning"*, ECAI'86, Brighton, Juillet 1986.

[Laugier 87] C.Laugier: *"Raisonnement géométrique et méthodes de décision en robotique. Application à la programmation automatique des robots"*, Thèse d'Etat, INPG, Grenoble, Décembre 1987.

[Lozano-Perez 76] T.Lozano-Perez: *"The design of a mechanical assembly system"*, AI-TR-397, M.I.T. Artificial Intelligence Laboratory, Cambridge, Decembre 1976.

[Lozano-Perez et al. 83] T.Lozano-Perez, M.T.Mason, R.H.Taylor: *"Automatic synthesis of fine-motions strategies for robots"*, 1st International Symposium of Robotics Research, Bretton Woods, Août 1983.

[Lozano-Perez, Brooks 85] T.Lozano-Perez, R.A.Brooks: *"An approach to automatic robot programming"*, AI Memo 842, Artificial Intelligence Laboratory, M.I.T., Avril 1985.

[Mason 82] M.T.Mason: *"Manipulator grasping and pushing operations"*, AI-TR-690, Artificial Intelligence Lab., M.I.T., Cambridge, Juin1982.

[Mason 84] M.T.Mason: *"Automatic planning of fine motions: correctness and completeness"*, IEEE International Conference on Robotics and Automation, Atlanta, 1984.

[Mazer 83] E.Mazer: *"Geometric programming of assembly robots (LM-GEO)"*, International Symposium on Advanced software in Robotics, Liège, Mai 1983.

[Merlet 86] J.P.Merlet: *"Contribution à la commande par retour d'effort en Robotique. Application à la commande de robots parallèles"*, Thèse de Doctorat, Paris 6, Juin 1986.

[Nilsson 80] N.J.Nilsson: *"Principles of Artificial Intelligence"*, Tioga, Palo Alto, 1980.

[Puget 85] P.Puget: *"Problèmes de prise en compte d'incertitudes en Robotique d'assemblage"*, Rapport de DEA, LIFIA/IMAG Laboratory, Juin 1985.

[Puget, Troccaz 86] P.Puget, J.Troccaz: *"Contrôle dans le système de programmation automatique SHARP: gestion des interdépendances liées aux contraintes d'accessibilité et d'incertitude"*, Rapport de Recherche IMAG/LIFIA, no. 615, Juin 1986.

[Puget 88] P.Puget: *"Utilisation de techniques de preuve de programme pour la programmation automatique des robots"*, Thèse de l'Institut National Polytechnique de Grenoble, 1988 (à paraître).

[Raibert, Craig 81] M.Raibert, J.Craig: *"Hybrid position/force control of manipulators"*, Journal of Dynamic Systems, Measurement and control, no. 102, Juin 1981.

[Reboulet, Robert 85] C.Reboulet, A.Robert: *"Hybrid control of a manipulator with an active compliant wrist"*, 3rd ISRR, Gouvieux, pp76-80, od 1985.

[Simunovic 75] S.N.Simunovic: *"Force information in assembly processes"*, Proc. 5th International Symposium on Industrial Robots, Chicago, Sept. 1975.

[Smith et al. 86] R.Smith, M.Self, P.Cheeseman: *"A stochastic map for uncertain spatial relationships"*, IEEE Conf. on Robotics and Automation, Raleigh, Mars 1987.

[Taylor 76] R.H.Taylor: *"Synthesis of manipulator control programs from task-level specifications"*, AIM 228, Stanford Artificial Intelligence Laboratory, Juillet 1976.

[Theveneau 85] P.Theveneau: *"Une méthode de génération de mouvements fins basée sur une analyse des contacts"*, Rapport de DEA, LIFIA/IMAG Laboratory, Juin 1985.

[Theveneau 88] P.Theveneau: *Planification de mouvements fins de montage dans un système de programmation automatique de robots"*, Thèse de l'Institut National Polytechnique de Grenoble, 1988 (à paraître).

[Troccaz 86] J.Troccaz: *"Modélisation du raisonnement géométrique pour la programmation des robots"*, Thèse de l'Institut National Polytechnique de Grenoble, Avril 1986.

[Troccaz, Puget 87] J.Troccaz, P.Puget: *"Dealing with uncertainties in robot planning using program proving techniques"*, 4 th International Symposium of Robotics Research, Santa Cruz, Août 1987.

[Valade 85] J.M.Valade: *"Raisonnement géométrique et synthèse de trajectoire d'assemblage"*, Thèse de Docteur-Ingénieur, Université Paul Sabatier, Laboratoire d'Automatique et d'Analyse des Systèmes, Toulouse, Janvier 1985.

[Whitney 77] D.E.Withney: *"Force feedback control of manipulator fine motions"*, Journal of Dynamic Systems Measurement and Control, Juin 1977.

[Will, Grossman 75] P.M.Will, D.D.Grossman: *"An experimental system for computer controlled mechanical assembly"*, IEEE Transactions on Computers, Vol. 24, n0. 9, Septembre 1975.

[Winston 77] P.Winston: *"Artificial Intelligence"*, Addison-Wesley Publishing Company, 1977.

Imprimé en France
par
l'Institut National de Recherche en Informatique et en Automatique