Implantation optimale d'un robot en fonction d'une tâche à réaliser en environnement contraint. Analyse, synthèse et développement d'un module d'aide à l'implantation des robots

Joseph Doulcier

Ecole Nationale des Ponts et Chaussées

THESE

Présentée pour obtenir le diplôme de DOCTORAT
Spécialité : Mathématiques et Informatique

Implantation optimale d'un robot en fonction d'une tâche à réaliser en environnement contraint

Analyse, synthèse et développement d'un module d'aide à l'implantation des robots

par

Joseph DOULCIER

Soutenue le 14 Janvier 1993 devant le jury composé de :

Messieurs	Marc	RENAUD	President
	Wissama	KHALIL	Rapporteur
	Emmanuel	MAZER	Rapporteur
	Dominique	CHEVALLIER	
	Pierre	TOURNASSOUD	
	Rene	LALEMENT	

Avant propos

Les travaux présentés dans ce mémoire ont été réalisés au Centre d'Enseignement et de Recherche en Mathématiques Appliquées de l'Ecole Nationale des Ponts et Chaussées.

Ils ont fait l'objet d'un Contrat Industriel de Formation par la REcherche avec la Direction de la Recherche et des Affaires Scientifiques de PEUGEOT S.A. Etudes et Recherches.

Ils ont été réalisés avec l'aide logistique et les conseils de la Direction de l'Informatique, des Télécommunications, et des Automatismes de PSA, où s'est déroulée la mise en oeuvre informatique de ces travaux.

Ces travaux doivent beaucoup à Pierre Tournassoud, qui les a fait bénéficier de son expérience et a soutenu par ses conseils l'ensemble du projet.

Je tiens à exprimer mes remerciements avec ma plus vive gratitude au Professeur Nicolas Bouleau qui m'a accueilli dans son laboratoire , ainsi qu'au professeur Dominique Chevallier qui a accepté de diriger cette thèse.

Je remercie tout particulièrement le professeur Marc Renaud du LAAS-CNRS de Toulouse, qui me fait l'honneur de présider le jury de cette thèse.

Que le professeur Wissama Khalil et le professeur Emmanuel Mazer soient assurés de ma profonde gratitude pour leurs conseils pertinents et pour avoir accepté de juger ce travail.

Je tiens aussi à remercier Le professeur René Lalement qui a suivi ces travaux depuis leur origine en leur apportant un point de vue particulier et pour sa participation au jury de cette thèse.

Enfin je remercie monsieur Jean-Marie Heinrich et son équipe, responsable des développements informatiques en robotique pour Peugeot-SA, qui m'a accueilli dans le monde industriel.

Je souhaite d'autre part rendre hommage à Monsieur Lebérichel, Responsable de la robotique pour la DRAS, et à Monsieur Caillot, responsable du Centre d'Expérimentation en Robotique et en Automatisme, qui ont apporté pour ces travaux le cadre industriel indispensable.

Résumé

Ce mémoire présente une étude de l'implantation optimale d'un robot en fonction de la tâche à réaliser, en prenant en compte les contraintes de non-collisions.

La démarche proposée s'est appuyée sur un système de C.A.O. classique. Elle se fonde sur une définition des tâches en termes de trajectoire de référence, définie par des points d'arrêt induits par le processus de fabrication et des points de passages disposés pour assurer la non collision avec les obstacles.

Elle aboutit à une méthode permettant d'optimiser la trajectoire du robot de manière conjointe avec son implantation, en étendant le concept de variables articulaires à l'ensemble des variables décrivant les possibilités d'implantation du robot.

Ces points de passage sont mis en place au cours du processus d'optimisation générale en fonction de l'implantation du robot et de la disposition des obstacles.

Cette approche est introduite par une étude de la planification de trajectoires. à travers une méthode locale dissociant les objectifs de réalisation du *process* des objectifs d'anti-collision, suivie d'une étude du domaine d'implantation du robot.

Ces travaux ont été appliqués aux problèmes de l'industrie automobile à travers l'exemple des postes de soudure par points et des postes de montage.

Mots-Clefs :

[Planification de trajectoire] [Evitement d'obstacles] [Accessibilité]
[Simulation] [Optimisation] [Distance entre solides] [CAO Robotique]
[Tâche robotique] [Implantation]

Table des Matières

Présentation

La robotique des manipulateurs est depuis longtemps un pôle d'intérêt majeur des chercheurs en Mécanique des solides articulés notamment en ce qui concerne l'inversion de coordonnées ou la modélisation géométrique puis dynamique ainsi que la planification de trajectoires.

La recherche en robotique s'est en premier lieu intéressée à la modélisation des robots, avec l'intention de les commander d'une manière adéquate en s'attachant à définir :

- le bon fonctionnement des armoires de commande, avec leur bonne simulation en conception assistée par ordinateur (CAO), pour une estimation convenable des temps de cycle,
- une commande prenant en compte une analyse des efforts exercés,
- une commande adéquate pour des manipulateurs de cinématique complexe (boucles fermées...),
- des tâches plus souples, commandes en coordonnées articulaires, cartésiennes, à l'aide d'un langage normalisé dit de "niveau tâche".

Si une maîtrise correcte effective du fonctionnement du robot est en soi un objectif essentiel, il n'est pas l'objet principal de ce mémoire, dont les ambitions sont orientées davantage dans une optique industrielle vers l'optimisation des procédés de fabrication utilisant des robots.

Le robot en est un mécanisme important qu'il faut utiliser rationnellement en fonction de l'objectif fondamental :

Comment passer du processus de fabrication défini par la direction des méthodes, à une exécution optimale de ce qui sera une tâche du point de vue de la robotique ?

Ce qui nous a conduit aux problèmes suivants :

- Quelles sont les contraintes du processus de fabrication robotisé et les critères qui définissent une tâche optimale ?
- Comment passer de la tâche telle que définie par le processus industriel, à la tâche optimale pour le robot envisagé ?
- En particulier, où implanter le robot pour que la tâche qui lui est assignée soit exécutée au mieux ?

Ces problèmes ne peuvent être abordés au dernier moment sans inévitablement provoquer des bouleversements coûteux. Ils sont même presque toujours cruciaux dans la détermination du process lui-même.

Pour y répondre, les industriels qui utilisent les robots de manière extensive ont déjà des équipes entraînées qui, au prix d'un effort important, s'appuient sur la CAO pour visualiser et analyser la cellule robotisée envisagée. Même en disposant souvent d'un modèle CAO des pièces constituant l'environnement du robot ainsi que d'un module, généralement sommaire, de simulation des robots, cette activité entraîne des dépenses d'énergie et de temps considérables.

La recherche réalisée commence ainsi par analyser les processus industriels mettant en oeuvre des robots de manière extensive. Cette étude est une contribution à la reconnaissance et au repérage des problèmes et de leurs points-clefs. Elle s'attache à la mise en forme et à l'explicitation de chaque composante essentielle pour mettre en place les bases des solutions envisagées.

Nos travaux se présentent ainsi sous 3 aspects, permettant de conduire à la méthode des articulations virtuelles qui constitue l'apport majeur de cette thèse :

- La première partie expose la problématique de l'implantation des robots avec les objectifs de l'industrie, ainsi que la nature spéciale des problèmes de modélisation pour la robotique qu'il faut harmoniser avec les modélisations industrielles plus générales. Nous proposons dans ce but des éléments et des outils informatiques. Enfin, puisque la prise en compte des obstacles dans le processus d'implantation optimale est une difficulté majeure, le contournement des obstacles est tout d'abord envisagé pour la réalisation d'un programme de générations de trajectoires sans collisions. Ce programme sera utilisé par la suite comme un outil par notre algorithme mettant en oeuvre la méthode des articulations virtuelles.

  Cette partie est ainsi composée de trois éléments :

  - D'abord une partie préparatoire, d'analyse des problèmes d'implantation comme ils se posent dans l'industrie, des éléments de solutions apportées par la robotique pour la modélisation des tâches et le calcul de leur durée, ainsi que la planification de trajectoires. Ceci nous permet de préciser les variables, les critères et les contraintes nécessaires à l'implantation optimale du robot, le principal critère étant la durée de la tâche.
  - Ensuite une partie informatique, traitant les problèmes particuliers d'intégration de notre module d'implantation optimale dans un système de CAO. Ceci nous permet d'envisager l'analyse de la scène en vue de contrôler les distances pour éviter les collisions, et d'adapter la partie surfacique du modèle à ses exigences. Cette partie débouche sur la programmation du calcul de distances sur l'ensemble du modèle et d'un algorithme d'approximation polyédrique des surfaces limitées.
  - Enfin, préalablement au problème d'implantation, se pose celui de l'existence de trajectoires sans collisions. C'est pourquoi nous traitons ce problème dans la partie suivante, où nous présentons, adaptons et programmons la méthode des contraintes proposée par [FT87a]. Cette méthode est ensuite utilisée dans la méthode des articulations virtuelles, clé de cette thèse, pour conserver un contrôle

adéquat de la trajectoire au cours du processus d'implantation optimale.

- La seconde partie propose la méthode des articulations virtuelles elle-même. L'implantation optimale des robots sans préoccupations de collisions est d'abord présentée : une méthode efficace est proposé et réalisée dans ce but en donnant de bons résultats industriels lorsque les collisions peuvent être traitées indépendament de l'implantation du robot. La méthode des articulations virtuelles y est présentée en détail et sous chacun de ses aspects.

  Cette partie est ainsi composée de deux éléments.

  - L'implantation optimale d'un robot peut, dans un premier temps, ne pas prendre en compte les collisions. Apparament paradoxale, cette partie est justifiée par les problèmes industriels concrets, dont une classe importante, celle de la soudure par points dans les zones de reprise, permet souvent un traitement indépendant pour les collisions. Cette partie permet d'envisager le problème de l'accessibilité d'une trajectoire, de proposer une méthode d'implantation simple de type cellulaire, ainsi qu'une méthode mixte, dont nous avons programmé l'algorithme rapide en vue d'une utilisation dans le cadre d'un projet d'intelligence artificielle. Cette partie débouche sur la présentation et la programmation d'un algorithme d'implantation optimale de type cellulaire, qui est actuellement utilisé industriellement avec des résultats intéressants.
  - Puis, nous présentons notre proposition pour l'implantation optimale des robots, tenant compte du contrôle des collisions. La méthode des articulations virtuelles que nous proposons consiste à joindre les variables d'implantation à celles des articulations du robot et de traiter alors ces variables globalement. Après avoir énoncé et explicité les considérations fondamentales, puis décrit les fondements mathématiques nécessaires, enfin discuté des caractéristiques de chaque étape du processus, nous formalisons l'ensemble de l'algorithme. Celui ci n'a pas encore été testé dans les conditions opérationnelles de l'industrie, cependant les conclusions essentielles ont déjà pu être énoncées.

- Une dernière partie présente sous forme d'annexes deux éléments particuliers liés au problème de l'implantation optimale :

- notre proposition d'une méthode déterminant de manière très efficace le volume d'implantation du robot sans considérations de collisions, avec des avantages particuliers pour les projets relevant de l'intelligence artificielle. Cette proposition fait l'objet d'un article [YD92a].
- une étude générale des problèmes d'optimisation aboutissant au choix de la méthode de la tolérance flexible, particulièrement efficace dans les problèmes d'optimisation sous contraintes sévères.

Première Partie

Problématique de l'implantation des robots, de son assistance par ordinateur et de la gestion des collisions

Les analyses présentées dans cette partie permettent de déterminer puis de justifier le choix des variables d'implantation du robot, de la durée de la tâche comme critère d'optimisation et de la sûreté de trajectoire comme contrainte de cette optimisation.
Elle expose les motivations du choix de sa réalisation informatique en étroite liaison avec le système de CAO CATIA. La cohabitation de nombreux types d'entités dans sa base de données nous a conduit à considérer en premier lieu les solides. Nous proposons ainsi la réalisation d'un algorithme de calcul général de distances entre deux polyèdres quelconques. Il est couplé à un deuxième programme proposé et réalisé qui hiérarchise les éléments du modèle de la cellule et permet un contrôle des distances plus rapide sur l'ensemble du modèle. Ensuite, les éléments manipulés ou rencontrés par le robot étant souvent modélisés par des surfaces resteintes, nous proposons un algorithme les transformant en polyèdres à facettes triangulaires. Sa réalisation efficace a prouvé des qualités très utiles même pour des domaines en dehors de la robotique, comme la stéréophotolithographie.
Elle est conclue, puisque le but final est de tenir compte des obstacles, par une analyse des objectifs et des moyens de génération de trajectoires sans collisions, et présente dans ce but la méthode des contraintes proposée par Bernard Faverjon. Cette méthode locale prend en compte les problèmes de collisions sous la forme de contraintes. Nous proposons une réalisation de cet algorithme adaptée aux problèmes industriels abordés et à la base de données utilisée. Cet algorithme sera utilisé par la suite pour générer des trajectoires sûres permettant d'assurer un contrôle des collisions sans exiger de prendre en compte un grand nombre de points de la trajectoire.

Chapitre 1

Problématique et contexte de l'étude

Il s'agit de la présentation des fondements industriels mais aussi des définitions, des variables, des contraintes ou des critères concernant les éléments concrets ou les concepts de base de la thèse.
Les cas d'utilisation des diverses méthodes ainsi que la notion de tâche considérée y sont explicités. L'utilisation de méthodes locales pour l'optimisation est mis en évidence, la durée de la tâche étant considérée comme le critère principal.

1.1 Les postes robotisés du monde industriel

L'industrie manufacturière utilise des robots depuis déjà longtemps : les constructeurs automobiles sont parmi les pionniers de la robotique. Les tâches périlleuses, comme le service des presses d'emboutissage, sont robotisées dans leur quasi-totalité. Toutes les tâches de soudure, chaque fois que c'était possible, ont été robotisées. Les postes manuels ont été presque totalement éliminés des ateliers de construction de la structure métallique de la voiture.

Pour ce qui est du montage, les gestes sont géométriquement et physiquement plus complexes, plus précis et plus divers avec des tolérances souvent faibles de telle sorte que les efforts à appliquer aux pièces deviennent difficiles à reproduire

et à contrôler par un robot. Ainsi seuls quelques aspects ponctuels du montage sont-ils robotisés : dépose et pose des portes, montage du hayon, installation des sièges, pose de la planche de bord, montage de quelques sous-ensembles, pose du pare-brise et de la lunette arrière, encollage et mise en place de joints, pose de la garniture du plafond.

Il convient ainsi d'étudier les éléments majeurs des processus de fabrication robotisés notamment les divers postes et les outils.

Les pinces de soudure

Leur géométrie est déterminée en visualisant des coupes adéquates sur chacun des points de soudure à exécuter ; pour chaque point de soudure, les mors de la pince doivent passer autour des tôles "emprisonnées" entre les portes électrodes sans toucher les obstacles extérieurs. En conséquence, la pince doit être dimensionnée en fonction de l'ensemble des obstacles qu'elle "embrasse" d'une part et en fonction de l'ensemble des obstacles extérieurs d'autre part. Pour traiter ce problème d'une manière convenable il faut établir l'étude en trois dimensions, considérer le "voisinage" du point de soudure, détecter les collisions éventuelles de la pince et ainsi calculer pour ce point les vecteurs, direction et distance, d'allongement et d'écartement des mors de la pince. Ceci doit être fait pour tous les points de soudure. Une marge de sécurité est généralement appliquée ensuite aux résultats ainsi obtenus.

Les postes de conformation

Définition 1 *Ce sont ceux qui réalisent les quelques points de soudure qui sont faits les premiers, alors que l'ensemble des tôles sont tenues ensemble à leur bonne place par un appareillage de serrage. Ces points servent à garantir les positions relatives des pièces pendant les étapes suivantes de la fabrication.*

Le critère principal est ici l'absence de collisions :

En de tels cas, comme sur la figure 1.1, la plus grande difficulté est de trouver une implantation du robot et une trajectoire de celui-ci qui permette de faire tous ces points indispensables en évitant les obstacles constitués par l'appareillage de serrage généralement très important.

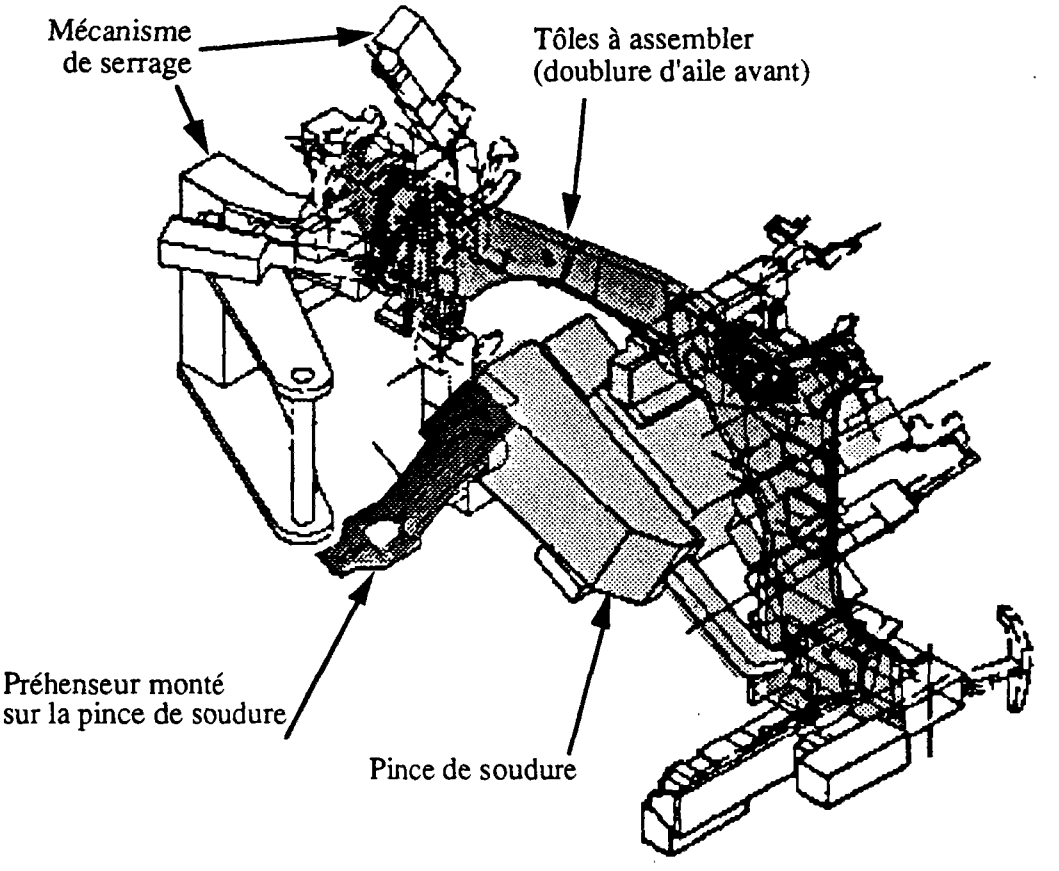


Figure 1.1 : Exemple de poste de conformation. Ce modèle CAO (Catia) représente une doublure d'aile d'automobile, les dispositifs de serrages qui maintiennent les éléments en place pendant que la pince de soudure réalise quelques points. La pince est équipée d'un préhenseur pour extraire la pièce après réalisation des points de soudure.

Ces postes sont très encombrés, notamment par les outils de serrage. Ils se trouvent très souvent sur un chemin critique : en effet, si peu de points de soudure sont à réaliser, entre deux points successifs pouvant être proches, le détour à faire peut se révéler très grand : ainsi les problèmes de collision ont-ils une importance très aigüe avec une très grande difficulté de résolution.

Véritablement l'étude du contournement de tels obstacles exige une analyse globale du problème car toute approche locale conduit rapidement à des

blocages. Il convient tout au moins que l'opérateur ait fourni au système quelques indications globales, par exemple quelques points de passages jugés stratégiques.

Sans assistance de l'ordinateur, l'étude de ces postes est un travail rendu très délicat par l'encombrement de l'espace de travail du robot. La vision en trois dimensions, l'observation de la scène sous des angles différents, ne suffisent pas toujours pour être sûr d'éviter les collisions, il faut alors visualiser des coupes aux endroits délicats. Par tâtonnements successifs prenant souvent plus d'une semaine, en déplaçant un peu le robot, en modifiant la trajectoire, parfois en modifiant les appareils de serrage des pièces, les préparateurs organisent le poste.

Les postes de reprise

Définition 2 *Ce sont ceux qui réalisent les points de soudure destinés à garantir la tenue mécanique de la voiture.*

Le critère principal est ici la vitesse d'exécution :

Dans ce cas, les tôles sont déjà reliées entre elles par les points de soudure de conformation ; il y a donc peu d'appareils pour tenir les tôles, ce qui diminue l'importance des problèmes de collisions. En revanche beaucoup de points doivent être exécutés le plus rapidement possible : c'est pour diminuer cette durée qu'il faut implanter le robot là où il sera le plus efficace.

Les points étant réalisés à la suite les uns des autres en suivant une ligne de soudure, peu de détours sont à faire et les risques de blocage sont faibles ; ainsi une méthode locale peut-elle généralement résoudre de tels problèmes.

Pour les tâches de soudure aux postes de reprise, dans les conditions actuelles une journée est en moyenne nécessaire pour définir une position convenable du robot.

Les postes de montage

Dans les applications de montage, peu robotisées actuellement et qui apparaissent comme les grandes voies de développement de la robotique, tous les problèmes s'agrègent :

- Précision des mouvements des pièces transportées déterminante pour le succès de l'opération,
- Difficulté à se garantir contre les risques de collision entre les organes du robot et son environnement,
- Pendant les grands mouvements du robot, difficulté de se garantir contre les risques de collision entre la pièce manipulée par le robot et la structure sur laquelle on doit la monter,
- Pendant les phases terminales des mouvements, difficulté de se garantir contre les risques de collision "très fine" entre la pièce manipulée par le robot et la structure sur laquelle on doit la monter,
- Au milieu de toutes ces contraintes, difficulté de choisir un emplacement du robot aussi bon que possible.

Les problèmes de collisions y sont permanents, les réalisations très délicates comme dans le cas de la figure 1.2. Contrairement aux postes de reprise de soudure par points, les contraintes sont ici telles qu'il s'agit bien davantage de passer loin des obstacles plutôt que d'exécuter très vite le travail considéré, comme le montre la figure 1.2.

Pour les montages un tant soit peu complexes, la plupart du temps, les objets sont reçus des bureaux d'études définis sous forme de surfaces (définition 14) ; une transformation en solides (définition 17) est alors nécessaire pour permettre l'élimination des parties cachées, de manière à tenter de mieux comprendre la scène et si possible d'essayer de voir les intersections éventuelles avec l'environnement. c'est par exemple le cas des tableaux de bord. Un tel travail peut prendre plusieurs jours.

1.2 Les notions d'espace en robotique

Espace ambiant et espace des configurations

Définition 3 *Espace des configurations : C'est l'espace mathématique qui représente les positions relatives des éléments d'un système mécanique. C'est*

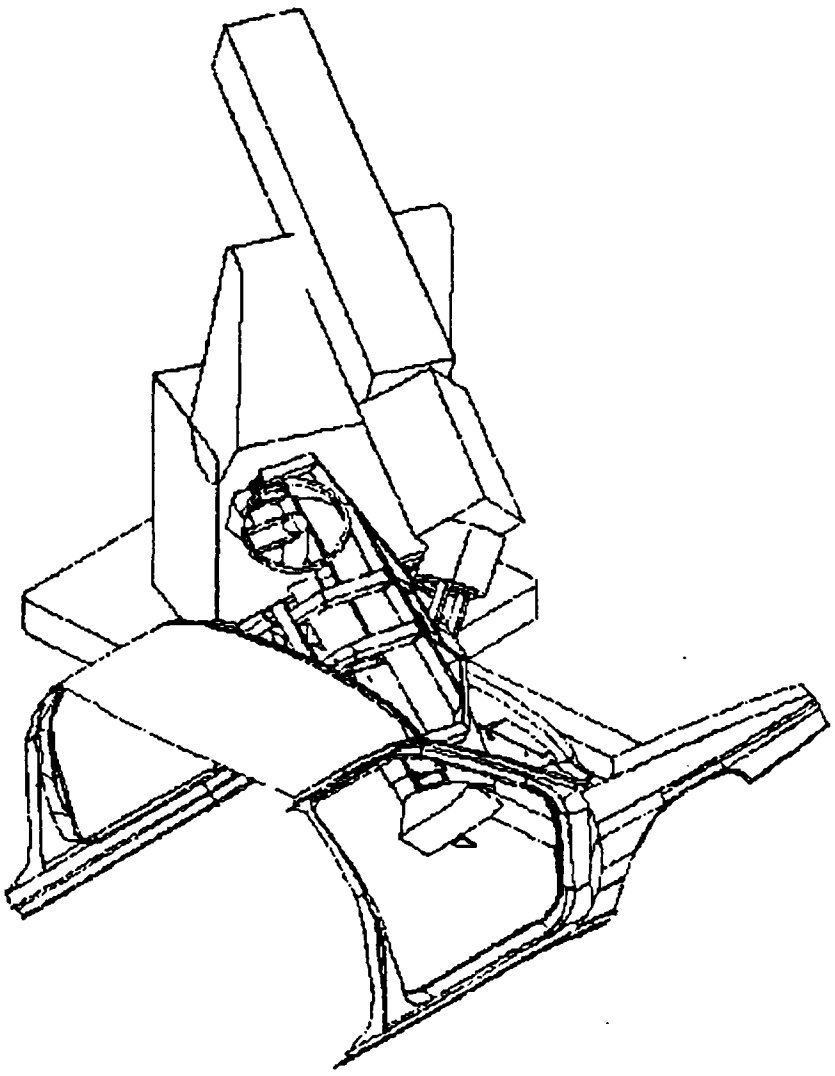


Figure 1.2 : Montage du poste de conduite Citroën XM (modèle CAO Catia). Le robot introduit le poste de conduite par la baie du pare-brise.

souvent le cadre de description des commandes à donner au robot, "l'espace tangent" (au sens mathématique) à l'espace des configurations définissant naturellement l'espace des commandes.

Définition 4 *Variables articulaires : Elles sont un système de coordonnées de l'espace des configurations. Ces variables sont dites variables articulaires, même s'il ne s'agit pas d'un angle de rotation à une articulation.*

notations : les n variables articulaires d'un robot seront notées Q^i, i variant de 1 à n. la vitesse maximale de chaque articulation est notée V^i_{max}.

Les formes et dimensions des organes du robot sont définies, ainsi que les mouvements possibles entre ces organes. La position et l'orientation de l'outil se déduisent de la configuration du robot, connue de manière extrinsèque par les variables articulaires du robot [Ler87].

La formulation la plus utilisée pour les robots est celle de Denavit-Hartenberg [Del87] [Meg84], qui date de 1955. Depuis ont été proposés les paramètres de Denavit-Hartenberg modifiés qui sont très souvent utilisés [Kha85] [Cra86], éventuellement avec des modifications d'indices de paramètres [FR89] [Ren87].

Une formulation classant les articulations complexes du robot suivant leur type est utilisée pour les travaux réalisés pour cette thèse [Gom90]. Même si l'ordinateur peut aujourd'hui calculer automatiquement les équations de la mécanique et de la dynamique d'un robot [KL86] [Ren87], l'étude des systèmes mécaniques de solides par les groupes de Lie [Ler88] [Che84], propose aujourd'hui des formulations plus homogènes et plus générales.

Définition 5 *Espace ambiant : C'est l'espace à 3 dimensions dans lequel évoluent le robot et l'objet transporté. Il est le cadre naturel pour définir la géométrie des différentes pièces de l'atelier.*

Lorsqu'il s'agit de robots manipulateurs, le seul espace vraiment adéquat pour tout contrôler est celui des *variables articulaires*. Un robot du type R6, c'est à dire offrant la possibilité de 6 rotations, peut accéder à un point donné avec une orientation fixée de plusieurs façons (jusqu'à huit pour les robots que l'on connaît habituellement dans le monde de l'industrie manufacturière, jusqu'à 16 en théorie [YL84]). A une position correspondent des configurations différentes, avec retournement de poignet, coude en haut ou en bas, porteur en avant ou en arrière... et même une infinité de façons, dans les cas singuliers où le robot est dans une configuration particulière [SF88] (alignement de certains axes). Les

possibilités sont encore augmentées lorsqu'une des articulations du robot peut réaliser plus d'un tour (en effet, les rotations ne peuvent pas être considérées à 2π près, sinon des reconfigurations inutiles pourraient alors avoir lieu).

Pour rendre compte de ces possibilités, l'espace ambiant est bien plus difficile à utiliser que l'espace des configurations, celui des variables articulaires. Car dans l'espace réel il faut prendre en considération tous les organes qui constituent le robot et les pièces manipulées, tandis que dans l'espace des configurations, au prix certes d'avoir à considérer un plus grand nombre de dimensions (souvent six), une configuration du robot est représentée par un point.

Cependant la transformation de la représentation des obstacles dans l'espace des configurations est très difficile sans approximations, ceci d'autant plus que l'image d'un obstacle unique de l'espace ambiant n'est pas obligatoirement connexe dans l'espace des configurations. Il est également bien plus difficile que dans l'espace ambiant d'apprécier concrètement ce qui est fait. En effet un "obstacle" dans l'espace des configuration est formé de configurations pour lesquelles il existe au moins un couple d'objets en interaction, c'est à dire au moins un couple d'objets trop proches ou en collision.

Définition 6 *Une Pose est un repère de l'espace ambiant; elle définit la position et l'orientation que devra atteindre le repère actif de l'outil.*

Il est vrai que l'on sentirait plus directement ce qui est fait si le raisonnement était réalisé dans l'espace ambiant. Cependant l'accessibilité aux poses [ISO88] de l'espace réserve bien des pièges, dus au fait que le robot n'est pas un point mais un solide articulé. Dans l'espace des configurations, un point représente le robot tandis que toutes les impossibilités engendrent des zones qui représentent complètement les configurations inaccessibles, et ce quelles qu'en soient les raisons. Dans l'espace ambiant, l'accessibilité n'est pas prévisible aussi simplement, surtout en orientation.

De plus, la proximité d'un obstacle limite les orientations possibles pour accéder en un point et réduit l'espace accessible, comme le montrent les travaux de P. Wenger analysant l'espace de travail dans un environnement encombré [WEN89].

Dans ces conditions,

- l'espace ambiant est le cadre naturel de description des obstacles,
- l'espace des configurations est l'espace naturel de description des commandes ou instructions données au robot.

Pour la famille des robots "série" à laquelle appartiennent tous les robots habituels (ce n'est pas vrai pour les robot "parallèles, mais ils sont très rares en situation industrielle),

- le calcul des positions spatiales à partir des coordonnées articulaires dans l'espace des configurations est généralement assez simple,
- par contre, l'expression de la transformation réciproque est généralement très compliquée.

En conséquence, la représentation des obstacles et l'expression des conditions de collisions dans l'espace des configurations se révèlent difficile à mettre en oeuvre dans les cas réalistes.

Cependant quelques auteurs ont tenté de gérer les problèmes de configurations du robot directement dans l'espace ambiant, ainsi est développée la méthode des "aspects", qui, de manière assez complexe, permet de trouver si une trajectoire est possible, c'est à dire s'il n'y a pas de changement d'aspects [Bor85] (voir chapitre 10.3).

Exemples de planification de trajectoire dans l'espace ambiant : simplification de la géométrie

Pour raisonner malgré tout dans l'espace ambiant, la plupart des auteurs ont recours à des simplifications de géométrie : remplacer les organes du robot par des collections de sphères, remplacer les différents bras du robot par des segments de droites, ce qui revient à inclure chaque corps du robot et l'outil ainsi que la charge transportée dans une enveloppe constituée d'un cylindre terminé à chaque extrémité par une demi-sphère. Les auteurs proposent alors une stratégie fondée sur le déplacement du segment correspondant dans un

environnement décrit par rapport à ce segment. Les modélisations de cet environnement et les stratégies sont en général très subtiles et originales, mais peu industrielles car fondées sur des approximations très limitantes. Les algorithmes d'Hasegawa [HT88], de Reif [RS85] ou de Donald [Don83], sont probablement les plus représentatifs d'entre eux par l'ingéniosité déployée pour sortir le bras d'un robot manipulateur d'un environnement encombré, ou par la structuration générale de tout l'espace.

1.3 La tâche à accomplir par le robot en suivant une optique "process"

La tâche à accomplir par le robot n'a malheureusement pas une définition unique : en effet le sens de ce terme est différent suivant les intervenants. c'est pourquoi cette partie a pour objectif de définir précisément le sens utilisé pour nos travaux, et surtout ceux des éléments de la tâche qui sont nécessaires pour appliquer les méthodes présentées dans les chapitres suivants.

Que prendre en considération pour définir les tâches ?

Il convient d'abord de remarquer qu'un processus robotisé met en jeu un ou plusieurs robots mais aussi un ou plusieurs préhenseurs ou pinces qui possèdent de multiples petits mécanismes annexes qu'il faut contrôler, mais faut-il intégrer leur gestion au sein de la tâche ? De plus en plus, ces process peuvent mettre en jeu des systèmes d'auto-contrôle utilisant des capteurs de toutes sortes, par exemple des systèmes de vision. Ces accessoires permettent ainsi, dans des conditions convenables, de prendre des pièces dans un vrac, de reprendre une pièce de manière propre à réaliser un assemblage, de contrôler les efforts appliqués aux pièces, par exemple pour monter des pièces souples grâce à un contrôle des efforts appliqués, ou encore manipuler les objets au moyen de prises stables [TLM87], ce qui engendre souvent des prises de décisions, dont les choix doivent être "optimaux" pour une exécution convenable de la tâche [Gha87]. La tâche doit pouvoir les gérer s'il est souhaitable de simuler ceux-ci.

Dans tous les cas envisagés, le contrôle de la précision exige des efforts importants d'organisation et d'imagination. Pourtant, il n'est pas souvent géré dans la définition de la tâche elle-même.

Doit-on considérer l'environnement de la tâche ?

En effet, le poste robotisé n'est pas seul dans l'usine. Il doit communiquer avec les autres entités pour rendre compte de ce qu'il fait et surtout pour coordonner son action avec celle des autres machines : approvisionnement des pièces qu'il manipule, évacuation des pièces réalisées, coopération entre les robots d'une même cellule ...

Ces problèmes sont en grande partie masqués lors de la simulation des processus par des modèles informatiques. Bien entendu, ces aspects sont peu à peu introduits dans le fonctionnement des modèles simulés.

Ainsi la simulation des capteurs proximétriques, ou des capteurs constitués d'un couple laser et caméra est déjà réalisé chez les industriels [Ver89] ou [Bou86], [Cro87], [Gom88]. Cependant le but de cette simulation n'est pas de produire une tâche réalisable par le robot mais d'étudier le comportement des capteurs pour apprécier la faisabilité d'une opération les mettant en oeuvre, ou d'envisager plusieurs hypothèses techniques en cherchant une bonne implantation des capteurs. Cette simulation peut également permettre le choix de capteurs appropriés, en évitant leur prolifération lorsque c'est possible car la fusion multitensorielle est très délicate à mettre en oeuvre. Comment la tâche doit-elle prendre en compte la gestion de ces capteurs ?

La prise en compte des erreurs de positionnement des pièces, appelée recalage, ou la prise en compte des imperfections du robot appelée calibrage, sont de même réalisées de manière industrielle. Cependant, il s'agit surtout d'une modification de la tâche à accomplir par le robot : cette modification intervient après les simulations pour vérifier que la tâche est encore réalisable. Dans la plupart des cas, ces modifications, qui sont faibles, sont réalisées non pas en CAO, mais directement sur le fichier décrivant la tâche, sur un micro-ordinateur qui sera relié à l'armoire de commande du robot. La définition de la tâche doit en tenir compte, si l'on souhaite faire réaliser celle-ci par des robots différents, ou encore directement à la suite des simulations.

Une définition progressive de la tâche, par étapes

Dans une première approche, il est ainsi apparemment raisonnable de décomposer la génération de la tâche en étapes :

- première définition en vue de la simulation des mouvements,
- évolution de celle-ci en vue de la mise en place et de la simulation des capteurs,
- dernière évolution de celle-ci en vue de la simulation des erreurs géométriques et pour l'amélioration de la précision.

Bien entendu, cette décomposition est discutable dans beaucoup de cas, notamment dans ceux où le choix des capteurs modifie sensiblement la géométrie des préhenseurs. La simulation initiale des mouvements servira alors de base pour mettre en lumière les problèmes de faisabilité géométrique, en permettant de construire une trajectoire avec prise en compte des capteurs, ceci en connaissant déjà une partie des problèmes géométriques : zones où il est souhaitable de vérifier précisément la distance, pièces gênantes, modifications des éléments... Cette étude sera alors préliminaire à la construction d'une tâche faisant référence aux capteurs, comme le proposent B. Espiau et C. Samson [Esp88].

La définition des tâches envisagée dans cette thèse est explicitée dans les paragraphes qui suivent en privilégiant la simulation des mouvements.

Même lorsqu'il s'agit uniquement de mouvements, bien des paramètres entrant dans le calcul sont cachés, ignorés ou simplifiés [Sam87]. Ainsi, à ce jour, il n'y a pas de simulateur de robots qui puisse prévoir le temps d'exécution d'une tâche avec une précision convenable, inférieure à quelques pour-cents, car une description réaliste de l'armoire de commande du robot mettrait en jeu un grand nombre de paramètres que la simulation a nécéssairement négligés [Dom90].

Pour une utilisation industrielle, il convient d'utiliser une définition de trajectoire compatible avec les outils actuels, aussi simple d'utilisation que possible.

Bien que la simulation de la cellule robotique aboutisse à la réalisation d'un programme propre à commander chaque robot, il est proposé d'effectuer les travaux en gardant une vision globale du processus à réaliser.

Dans l'optique "Langage de Commande"

Pour la commande du robot, il s'agit à la base d'un problème d'interventions à des stades divers, constitué par le calcul des tensions électriques ou des pressions

hydrauliques à appliquer aux actionneurs afin de minimiser un vecteur d'erreur qui varie en fonction du temps et des consignes demandées.

Cette intervention peut parfois être un calcul des forces ou des couples à appliquer pour chacun des actionnneurs.

Cette approche est générale et permet de traiter presque tous les cas [BFM88].

Autour de ce noyau généraliste, des langages de programmation de robots sont développés en général autour des mêmes principes visant à introduire des macro-commandes, c'est-à-dire à décomposer les tâches quelconques en sous-tâches plus simples et d'utilisation générale. Les commandes robotiques de haut niveau peuvent ainsi être proposées à partir de contraintes géométriques [Gas87] ou directement issues de la réponse de capteurs ou même de vision 3D [Yan87].

Cependant ces langages sont pénalisés par la difficulté de maîtriser le contournement d'obstacles au moyen de commandes élémentaires simples.

Pour l'instant, quelques langages d'armoires de commande comme Karel ou LM sont suceptibles d'exploiter quelques unes de ces possibilités [HP87].

Dans l'optique "Simulation"

Le système de commande du robot possède, en tant que composant, un ordinateur à part entière chargé d'une partie du travail, avec une grande quantité de capteurs dont le rôle est d'informer le système sur son état et sur la dérive du modèle du régulateur par rapport à la réalité des phénomènes physiques. En effet il est bien difficile de modéliser en simulation le comportement du robot en prenant en compte par exemple tous les frottements, lesquels dépendent de l'usure du robot et de la température...

Même si le simulateur pouvait comporter un module de commande similaire, il ne simulerait guère que le cas idéal quasi-utopique où les aléas de fonctionnement seraient nuls.

Il faut également compter avec la réalité industrielle : à ce titre l'architecture et la logique de l'armoire de commande font partie du patrimoine de l'entreprise fabricante, toute information dans ce domaine est impossible à obtenir de manière complète, fidèle et exacte : il n'est pas toujours aisé, même pour le constructeur, d'expliquer précisément ce qui est fait, qui procède parfois davantage d'un savoir-faire expérimental que d'une technique très formalisée. Parfois

d'ailleurs les caractéristiques du système de commande sont extrêmement originales, parfois au contraire elles sont obsolètes.

Certains réalisateurs de simulateurs envisagent même d'analyser *a posteriori* le comportement du robot réel, pour en déduire la loi de commande utilisée et ses caractéristiques (par exemple Robcad de Technomatix).

Dans l'optique "Process"

Pour l'opérateur qui définit le processus à robotiser par apprentissage ou par simulation sur son ordinateur, la définition de la tâche est simplifiée par le fait qu'il construit sa trajectoire à partir de primitives simples très peu nombreuses. Ainsi les problèmes de poursuite de trajectoires sont-ils souvent occultés ou simplifiés par des trajectoires d'interpolation-type qui seront ensuite approchées par le système de commande du robot.

Les trajectoires types sont définies par des lois d'interpolation :

- Loi trapézoïdale articulaire sans synchronisation, appelée aussi "Bang-coast-bang" : toutes les articulations du robot accélèrent, freinent ou vont à vitesse constante, ce avec leur vitesse ou accélération maximales possibles. Dans de telles conditions les axes du robot s'arrêtent indépendamment les uns des autres selon leurs possibilités, ce qui engendre des trajectoires très peu intuitives et ainsi dangereuses.
- Loi trapézoïdale synchronisée, appelée aussi "articulaire" : tous les axes du robot accélèrent, vont à vitesse constante, ou freinent en même temps : la trajectoire ainsi définie est une droite dans l'espace des configurations du robot, seul l'un des axes, l'axe le plus chargé, se déplace à vitesse, accélération ou décélération maximales.
- Loi linéaire dans l'espace ambiant, appelée aussi "linéaire cartésienne" : la trajectoire suivie est une droite dans l'espace ambiant... Il est difficile alors de préciser ce que deviennent les accélérations et les vitesses...

L'utilisateur est amené à concevoir une trajectoire définie pas à pas : les points à atteindre en position et orientation, les poses, sont certes définis dans l'espace ambiant mais comme les possibilités pour les atteindre sont souvent multiples, puisque la solution du modèle inverse n'est généralement pas unique, il faut

contrôler les configurations du robot. Dans de telles conditions, l'utilisateur est ainsi conduit à un post-traitement dont le but est de traduire la tâche à réaliser dans l'espace des coordonnées articulaires du robot, en choisissant les meilleures configurations en chaque pose [Gom86].

1.3.1 Les points de contrôle

Définition 7 *Configurations de contrôle (improprement mais couramment appelées points de contrôle) : la tâche du robot est définie ordinairement par une séquence de configurations du robot, dites configurations de contrôle, à atteindre successivement.*

Cette tâche ne se trouve ainsi définie que par une suite de points, dits points de contrôle, ou configurations de contrôle, dans l'espace des coordonnées articulaires du robot (espace des configurations). La manière d'aller d'un point de contrôle à un autre n'est pas essentielle, pourvu que ce soit physiquement possible. Dans la tâche définie de cette manière, subsistent encore beaucoup de libertés d'intervention que nous mettrons à profit par la suite lorsqu'il s'agira de contournement des obstacles et d'optimisation : en particulier, la trajectoire passant par les points n'est pas encore définie (elle sera choisie au moins sans collisions) ni le mouvement sur cette trajectoire (L'armoire de commande le définit en fonction des caractéristiques dynamiques du robot et des éléments en mouvement).

Définition 8 *Une tâche, et donc la trajectoire associée, est valide si elle est ainsi effectivement réalisable et correspond aux objectifs de l'utilisateur.*

Ces points de contrôle sont de deux sortes :

Définition 9 *Les points d'arrêt : ils sont fixés à l'avance, il correspondent généralement à une opération à effectuer, les passages en ces points sont une des conditions de la validité de la trajectoire, ils sont ainsi des points de passage obligé absolu. Le robot doit y attendre généralement qu'une opération soit effectuée : c'est le cas par exemple des points d'arrêt correspondant aux points de soudure.*

Définition 10 *Les points de passage : Ce sont des points analogues aux points d'arrêt, mais ils ne sont généralement implantés que pour se garantir contre les risques de collisions. Ils peuvent être modifiés, ajoutés ou supprimés, pourvu qu'aucune collision n'ait lieu.*

A condition de maintenir la sûreté de trajectoire, des procédés de lissage peuvent éventuellement permettre un passage sans discontinuité de courbure au voisinage de ces points et non pas en ces points eux-mêmes. Ceci est possible en imposant a priori ce voisinage, par exemple au moyen d'une distance maximale autorisée entre la trajectoire théorique et la trajectoire lissée.

Ils sont appelés souvent "points de passage" grossiers ou fins selon les exigences de proximité de passage.

Pour l'utilisateur, de tels points de contrôle permettent de définir la trajectoire de manière discrète par un nombre fini de points. La trajectoire entre ces points importe assez peu à l'utilisateur qui a placé les points de contrôle de manière telle que le risque de collision ailleurs qu'en ces points soit nul pour peu que le robot suive une trajectoire correctement tendue qui corresponde à l'intuition qu'il en a. Dans de telles conditions il devient interdit d'exagérément "bien viser" d'une position à une autre, avec passage en frôlant un obstacle entre les deux, car auprès d'un tel obstacle, le risque de collision pour un "petit" déplacement pourrait se manifester.

De fait, la trajectoire entre deux poses du robot n'est pas toujours aussi rectiligne qu'on s'y attendrait, car le suivi de trajectoire réalisé par l'armoire de commande du robot doit tenir compte de paramètres physiques qui ne sont pas pris en compte dans la définition de la tâche : l'inertie par exemple. Dans une telle zone à haut risque, situer un point de contrôle aura un effet plus sûr que de passer là même avec une bonne précision et une maîtrise suffisante de la trajectoire...

Qui plus est, la modélisation de la trajectoire à l'aide de points de passage pose des problèmes particuliers car les points de passage, comme leur nom ne le laisserait pas supposer, ne sont pas des points atteints effectivement par le robot. Ils ne sont qu'approchés par celui-ci, suivant un algorithme propre à chaque armoire de commande, algorithme qui "raccorde" les lignes brisées par des courbes.

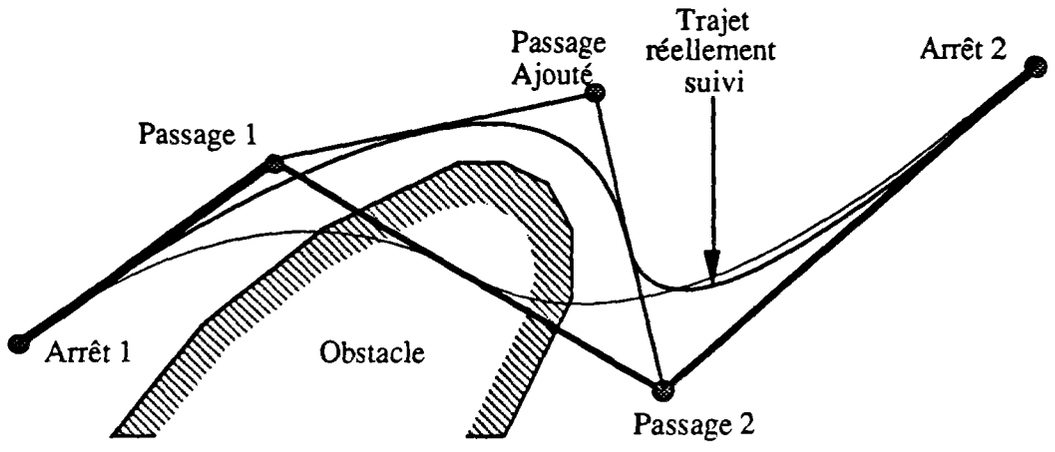


Figure 1.3 : Influence de l'ajout d'une configuration de passage supplémentaire. La trajectoire réelle dépend du type de commande du robot.

Par exemple (figure 1.3), pour certaines armoires de commande, la trajectoire théorique entre un point d'arrêt et un point de passage est suivie jusqu'à la moitié de la distance à parcourir entre ces points, au delà la consigne est de viser le point suivant.

Dans ces conditions, si un point de passage est ajouté pour contourner un obstacle, il modifiera la trajectoire en amont et en aval mais également les segments de trajectoire situés un segment plus loin.

1.3.2 Les primitives simples utilisées

Pour les tâches de soudure par points, les mouvements utilisés sont de deux types :

- Les poses de soudure sont souvent définies par un point, une normale à la tôle et une ligne de pince, représentant le repère à atteindre à mettre en coïncidence avec un point et deux droites représentant le repère actif de l'outil . Ces mouvements sont de ce fait absolus et définis dans l'espace ambiant. (ils sont appelés à tort cartésiens.)

- mouvements articulaires "absolus", définis par une configuration à atteindre, indépendante de la configuration courante.

Pour les tâches de montage, les mouvements utilisés sont de deux sortes :

- mouvements cartésiens absolus définis à partir d'un repère situé sur le préhenseur ou sur la pièce transportée et d'un repère à atteindre dans l'environnement,
- mouvements articulaires "absolus".

1.3.3 Le suivi de trajectoire

Si les êtres statiques que sont les trajectoires sont définies de manière discrète, c'est-à-dire par quelques points de passages, il convient de connaître la manière dont est réalisée l'interpolation entre ces points [Ton84]. Les constructeurs d'armoires de commandes proposent quelques options simples mais néanmoins incomplètement spécifiées notamment pour ce qui est de la précision et de la fidélité de trajectoire. Elles seront étudiées dans le chapitre portant sur le calcul de temps, qu'elles conditionnent puisque le mouvement (être dynamique) utilisé pour suivre une trajectoire est optimisé (sommairement, car cela est très dificile) conjointement avec la trajectoire. Pour les points de passages par exemple, un choix est souvent possible entre un mouvement lent suivant précisément une trajectoire, et des mouvements plus rapides suivant des trajectoires approchant la trajectoire théorique.

Lors de la simulation, la manière d'interpoler la trajectoire entre deux points de passage fait partie des options proposées. Dans le cas du contournement d'obstacles, elle influence les situations de blocage et l'allure de la trajectoire.

Remarque 1 *Trajectoire linéaire dans l'espace des configurations.*
Tous les robots possèdent un mode (non unique éventuellement) de construction de trajectoire par interpolation "articulaire", signifiant que les coordonnées articulaires du robot évoluent linéairement. Ce mode permet l'évolution la plus progressive des paramètres. Ainsi, pour la construction de la trajectoire, est-il souhaitable qu'en l'absence d'obstacles, le robot suive une droite dans l'espace des configurations, depuis la position courante du robot jusqu'au but.

Remarque 2 *Trajectoire linéaire dans l'espace ambiant.*
Ce mode de construction de trajectoires est très courant. Du point de vue de la commande, il est plus compliqué à cause de la nécessaire utilisation du modèle géométrique inverse qui permet parfois plusieurs configurations pour une même pose à atteindre, ce qui rendait le mouvement plus lent et peut conduire à des blocages.

Ce mode etait rarement utilisé industriellement à cause de sa lenteur passée. Maintenant, pour les robots industriels récents, le modèle inverse peut être calculé très rapidement. Ce faisant, comme la trajectoire est définie dans l'espace ambiant, les problèmes liés aux obstacles sont plus faciles à résoudre par l'opérateur. Les problèmes de collisions sont en général un peu moins blocants car les tâches sont en général définies pour mettre en correspondance des corps dans l'espace ambiant.

Remarque 3 *Autres trajectoires :*
D'autres modes de trajectoire sont possibles : interpolation circulaire, courbes polynomiales ou rationnelles construites à partir des points indiqués... Ces modes, définis dans l'espace ambiant, ne sont à peu près jamais utilisés industriellement, sauf à titre expérimental, car à ces modes sont préférés une interpolation linéaire entre des points de passages plus nombreux.

Ces options de mouvements suivant une trajectoire ont une telle importance que D. Johnson et E. Gilbert [JG85] ont proposé d'optimiser la durée des tâches en fonction d'une déviation tolérée donnée par rapport à la trajectoire nominale et en fonction d'un écart maximal donné entre la trajectoire et les points de passage théoriques.

1.3.4 La durée d'une tâche et le temps de cycle

Pour espérer calculer la durée d'une tâche, il faudrait avoir défini une trajectoire à suivre et un mouvement suivant cette trajectoire. Même si cela est accompli, la durée de la tâche ne pourait généralement pas être connue précisément par le calcul. Celui-ci est approché en utilisant un modèle plus ou moins précis du comportement dynamique du robot.

Une première approximation peut être envisagée pour les interpolations articulaires : la durée du déplacement entre deux points est donnée par l'axe le plus chargé qui évolue au maximum de ses possibilités.

En supposant une vitesse constante pendant le déplacement, la durée du déplacement pour un robot à n axes serait

$$\Delta T = Max_{i=1}^{n} \frac{|\Delta Q^i|}{V_{max}^i}$$

où n est le nombre de variables articulaires, Q^i représente la i^{eme} coordonnée du vecteur des variables articulaires et V_{max}^i la vitesse maximale suivant cette coordonnée.

Cette approximation unifiant les notions de trajectoires et de mouvement, a l'avantage de rendre équivalent un calcul de longueur de trajectoire avec un calcul de temps, mais elle est peu précise, puisque dans les petits déplacements, les valeurs d'accélération et de décélération sont déterminantes.

Une autre approximation est souvent envisagée, modélisant la loi de vitesse par une loi trapézoïdale. Cette modélisation correspond quelquefois à la loi réellement utilisée par le robot. Ce calcul est plus précis, mais pose des problèmes pour le calcul des points de passage sans arrêt : pour y passer avec une certaine précision le robot ralentit, ce qui n'était pas trictement indispensable car il n'est pas vraiment nécessaire que le robot y passe de manière précise, ces points servant de pôles tirant la courbe, comme pour les splines.

Calcul de la durée de la tâche basé sur les caractéristiques de moteurs

Ce paragraphe expose le calcul de temps tel qu'il a été réalisé dans les recherches faisant l'objet de ce mémoire.

Les lois de variations généralement utilisées pour calculer les durées des mouvements, sont appliquées traditionnellement aux articulations du robot. Or les robots les plus utilisés dans l'industrie possèdent des articulations fortement couplées entre elles : ainsi, généralement, les trois moteurs commandant le poignet sont situés en arrière du robot, le mouvement étant généralement transmis par des tubes coaxiaux jusqu'au poignet où des couples coniques renvoient les mouvements. La dernière articulation du robot est de la sorte mobilisée

par une combinaison des mouvements des moteurs 4, 5 et 6 du robot. Si l'on souhaite modifier l'articulation 4 sans modifier l'articulation 6, le moteur 6 devra fonctionner de manière à compenser les mouvements induits par la rotation du moteur 4.
De plus, il arrive sur des robots courants comme l'Acma X58, que certaines articulations soient commandées par des vis à billes, ce qui signifie que le mouvement de l'articulation n'est pas lié au mouvement de son moteur de commande (ou de ses moteurs s'il y a couplage) par une fonction linéaire.

Définition 11 *Tops codeurs : les rotations des moteurs de commande des articulations du robot sont exprimées en radians. Cependant, la mesure de ces rotation est effectuée dans la réalité en "tops" codeurs, correspondant aux positions du capteur associé au moteur. Ces positions sont utilisées par le logiciel du contrôleur du robot pour réaliser les asservissement nécessaires à la réalisation convenable des mouvements.*

Dans ce cas, il faut connaître les mécanismes de transmissions internes du robot et le nombre de tops par tours pour chaque moteur.

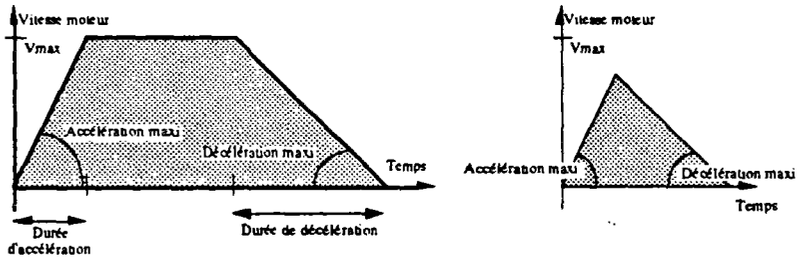


Figure 1.4 : Loi trapézoïdale pour la vitesse des moteurs. Suivant la durée du mouvement, le pallier à vitesse constante peut ne pas avoir lieu.

Dans de telles modélisations le comportement de chaque moteur du robot utilise une loi de vitesses trapézoïdale (figure 1.4), qui peut dégénérer en loi triangulaire si le mouvement est trop court pour que la vitesse maximale soit atteinte [Gom87].

Le constructeur du robot prévoit une vitesse maximale, une accélération et une décélération maximales pour chaque articulation du robot, qui correspondent à

des vitesses, accélérations et décélérations maximales pour chacun des moteurs. Des variations de position et d'orientation de l'extrémité du robot se déduisent les variations pour chaque moteur comme sur la figure 1.5, ce qui permet de déterminer le moteur le plus chargé. La durée du mouvement est obtenue d'après la loi de vitesse de ce moteur .

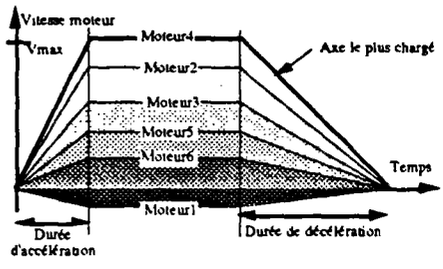


Figure 1.5 : Loi de vitesse de chaque moteur : l'axe le plus chargé est celui qui est sollicité au maximum.

Cette loi est théorique, ce n'est qu'une approximation de la loi de commande réelle utilisée dans l'armoire de commande du robot pour fournir des consignes d'asservissement. Même si cette loi est effectivement utilisée par le contrôleur, la courbe de vitesse réelle peut s'éloigner de ce modèle, ne serait-ce que du fait des mouvements des mécanismes qui ne peuvent dépendre de ces seuls paramètres.

Les points de passage

La modélisation adoptée envisage les points de passage.

Le modèle adopté considère d'abord le point de passage comme un point d'arrêt. Lorsqu'il ne reste plus qu'une durée Δt avant l'arrêt sur ce point, la position et la vitesse de chaque moteur sont calculées. Alors le mouvement suivant est envisagé avec les mêmes lois triangulaires ou trapézoïdales (figure 1.6).

La durée Δt dépend de la précision souhaitée pour le point de passage considéré. Il n'est pas possible d'obtenir des constructeurs les moyens de la calculer, aussi cette durée est-elle seulement estimée par une fraction de la durée du mouvement si le point était un point d'arrêt.

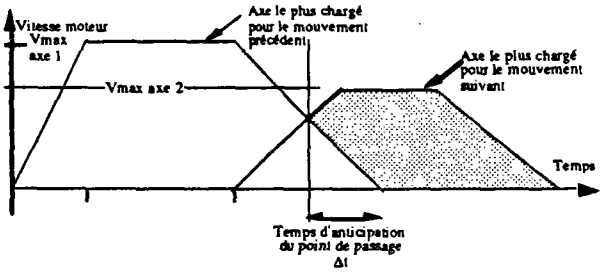


Figure 1.6 : traitement d'un point de passage : Une anticipation sur le mouvement suivant est proposée.

Continuité entre les tâches élémentaires : optimisation de la tâche

Pour que l'optimisation de la tâche soit réalisée dans son ensemble, il convient de déterminer quelles doivent être les configurations en chaque point de passage ou d'arrêt pour que la tâche soit réalisée le plus rapidement possible.

En particulier, il faut éliminer quand c'est possible, au moins détecter, tous les retournements de poignets, d'une manière générale tous les changements de configuration, qui sont source de grands mouvements de longue durée, peu intuitifs et ainsi dangereux, pouvant engendrer des collisions imprévues. La méthode utilisée, décrite par O. Gomart [Gom86], optimise la gamme des points de soudures grâce à l'algorithme "A^*". Cet algorithme recherche le plus court chemin à l'aide d'une fonction heuristique, dans un graphe reliant les configurations possibles pour chaque étape de la tâche, avec les configurations possibles aux étapes précédentes et suivantes.

Calcul de temps de gamme à partir d'un modèle dynamique

En réalité [LWP 83], le calcul de temps et la simulation des trajectoires et des mouvements sont liés. Dès lors, il faut simuler pour la C.A.O. un robot virtuel et une armoire virtuelle de commande, qui correspondent au fonctionnement réel de ces deux éléments fondamentaux. Puisque M. Fayet et M. Renaud [FR89] ou M. Giordano [Gio87] ont proposé des calculs rapides du modèle dynamique inverse des robots manipulateurs, des méthodes existent permettant de calculer un temps de cycle prenant en compte les aspects dynamiques [Kle86], en simulant le fonctionnement de l'armoire de commande dans son calcul de mouvement et de trajectoire en fonction de la tâche donnée. Des études sont en cours dans ce sens, et sont déjà validées [Dom90]. L'armoire virtuelle de

commande est chargée de modéliser les consignes fournies à chaque instant au robot. Le robot virtuel simule le comportement dynamique du robot réel, afin de prendre en compte les éléments suceptibles d'introduire des traînées dans le suivi de ces consignes, en particulier les moments d'inertie et les frottements.

La détermination d'un mouvement optimal le long d'une trajectoire spécifiée, complètement ou même de manière minimale, est un travail extrêmement difficile [FOU90], ce qui exige la connaissance des algorithmes effectivement utilisés dans les armoires de commandes réelles des robots, pour calculer la durée réelle du mouvement. Certains constructeurs de robots, en particulier Acma et Kuka, proposent d'ailleurs des programmes permettant une évaluation plus précise des durées d'exécution des tâches en fonction des algorithmes qu'ils utilisent.

Tant qu'un tel système ne sera pas validé, les temps ne peuvent être calculés précisément, mais ils font toujours l'objet d'approximations grossières, corrigées en partie par des études statistiques. La fiabilité de tels résultats reste encore décevante.

1.3.5 Les problèmes de la recherche de trajectoires sans collisions

Les planificateurs de trajectoire proposés par les chercheurs [Lau87] [Per86] montrent que les trajectoires sont successivement de deux natures différentes exigeant des stratégies adaptées.

En effet, la plupart des auteurs [Kha86] [Gou84] trouvent commode de décomposer les trajectoires en "grands mouvements" d'une part, trajectoires d'accostage et de dégagement d'autre part. De ce fait, la trajectoire est décomposée en trois parties :

- Une première phase consiste à quitter la configuration de départ pour une configuration plus sûre, en s'éloignant suffisamment des obstacles.
- Une deuxième phase part de cette configuration jusqu'à une autre située près de l'objectif, mais telle que les problèmes de collision sont encore peu cruciaux.

- Enfin une dernière phase cherche à rejoindre l'objectif final.

Accostage et dégagement : trajectoires jusqu'au contact

Le problème de l'accostage [LMT84] est très délicat, car il exige une gestion très élaborée des collisions éventuelles. En général, la trajectoire est courte, sa planification ne met en jeu ni des problèmes de minima locaux ni des problèmes de reconfiguration du robot. Les planificateurs pour grands mouvements sont inopérants à cause de leurs principes-mêmes qui mettent en jeu des approximations trop importantes de l'espace libre, ou refusent systématiquement le contact entre les pièces.

Les jeux et imprécisions dans les modélisations peuvent nuire au bon déroulement des algorithmes. M. A. Erdmann [Erd84] et R. Smith [SC86], ont montré qu'il est possible de planifier des trajectoires d'accostage pour les robots, en tenant compte des imprécisions pour garantir l'absence de collisions. J. Hopcroft et G. Wilfong ont même montré la faisabilité pour certaines classes de tâches, de trajectoires avec contacts et glissements [HW86a].

La "méthode des potentiels" est souvent proposée comme une référence dans ce domaine. O. Khatib [Kha86] est un précurseur dans ce domaine par sa méthode des potentiels avec obstacles répulsifs : les mouvements du robot sont obtenus en calculant un potentiel combinant une partie attractive vers le but à atteindre et une part répulsive de la part des obstacles. Comme les charges électriques dans un champ de potentiel, le robot subit des forces le déplaçant vers les zones de faible potentiel. mais les problèmes topologiques que cette méthode soulève sont nombreux [Kha86], [Kod87]. Elle convient bien aux grands mouvements, mais la recherche du contact est plus difficile, donnant alors lieu à des solutions ponctuelles pour chaque problème rencontré.

La répulsivité des obstacles, l'attractivité du but à atteindre, ou la tolérance de discrétisation ainsi que les marges de sécurité, doivent être très précisément contrôlées pour espérer qu'un algorithme ne subisse pas un échec.

Dans l'exemple de la figure 1.7, la pince de soudure doit venir coiffer les tôles qui se présentent. Bien que l'écartement de ses mors soit supérieur à l'épaisseur de la tôle, le mouvement est impossible car la définition de la zone de répulsivité peut conduire à la prise en compte d'une épaisseur d'obstacle supérieure à l'écartement des mors de la pince.

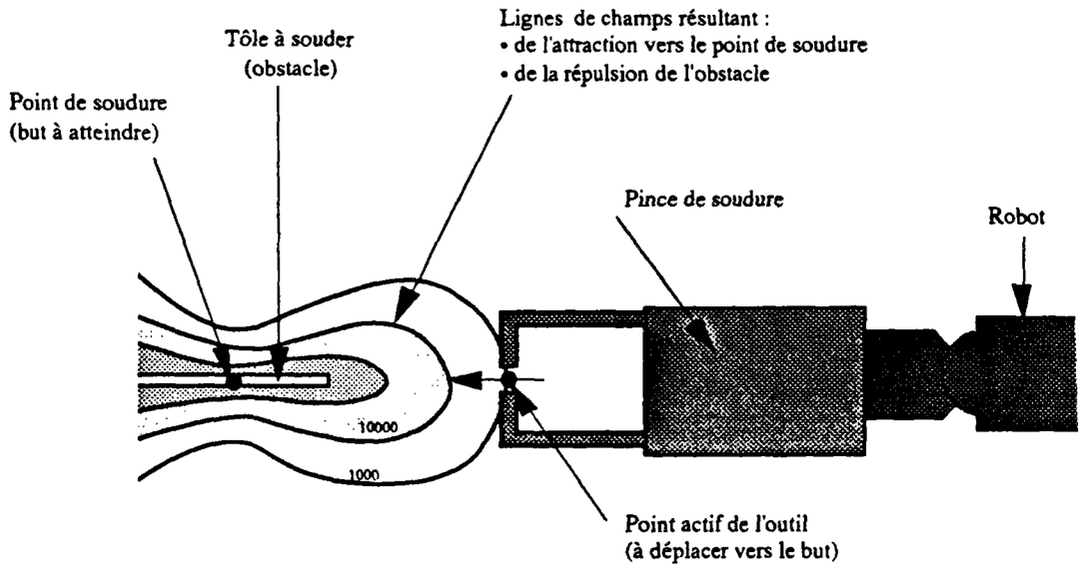


Figure 1.7 : Exemple d'échec pour la méthode des potentiels : le champ de potentiel combinant l'attraction vers le but à atteindre, et la répulsion des obstacles, ne permet pas de s'approcher du but

Cependant des règles générales de stratégies peuvent s'envisager :

- Afin de pouvoir obtenir *in fine* un vrai contact, la prise en compte des obstacles doit être de moins en moins contraignante au fur et à mesure de l'approche de l'objectif :
  - répulsion des obstacles en baisse,
  - distances de sécurité moindres,
- De la même manière, quand la vitesse de progression vers le but ralentit, les effets des obstacles doivent diminuer.
- Si la convergence semble s'effectuer sur une configuration qui n'est pas encore le but, on peut envisager de forcer la progression en repoussant le robot hors du minimum relatif ou absolu mais incorrect vers lequel il se dirige.

- Près du but, il faut contrôler les répulsivités des obstacles pour que le contact devienne possible, c'est à dire obtenir un processus permettant d'arriver au contact en un temps fini ou avec une convergence suffisante.

Grands mouvements : loin des obstacles

Nombre d'auteurs [Gou84] [Ver87] [LMJ*87] [FT87a] [Loz86] [Pap85] [LG87] [IO85] proposent des trajectoires pour les grands mouvements.

Il existe deux classes de méthodes envisageables :

- **Les méthodes locales**, qui proposent une progression vers le but en fonction des informations sur l'environnement et le robot données par la configuration courante du robot. Ces informations ne sont connues convenablement que dans un voisinage autour de cette configuration, ce qui induit des risques de convergence vers un optimum local seulement.

  En revanche, ces méthodes sont assez rapides, car assez peu coûteuses en temps de calcul.

  Dubowski et Shiller [SD86] [SD87] [Shi87], Bobrow [Bob89] ou Ashton [AH88], proposent sur ce principe une méthode de détermination d'une trajectoire sans collision de durée optimale à partir de considérations dynamiques. S. Suh [SS88], propose une approche semblable avec des préoccupations de sécurité dynamique.

- **Les méthodes globales :** En théorie, ces méthodes permettent de traiter tous les cas, mme pour des tâches mettant en jeu plusieur robots [HW86b] [NHT88]. Elles supposent une connaissance générale de toute la scène. En particulier, une connaissance préalable de l'espace libre dans l'espace des configurations serait indispensable. Cette connaissance ne peut être obtenue que pour un coût souvent prohibitif pour un cas d'application réel, ce qui jusqu'ici en empêche une utilisation vraiment industrielle. Généralement, ces méthodes consistent en une discrétisation de l'espace des configurations en cellules qui sont soit libres, soit caractéristiques d'une collision. Puis les cellules sont liées par un graphe de voisinage. Il reste alors à parcourir ce graphe de manière optimale. Parfois le système de cellules est hiérarchisé, ce qui permet d'améliorer les performances de recherche dans le graphe ainsi réalisé [WF86].

La dimension de l'espace (6 en général) associée à des temps de réponse raisonnables ne sont vraiment concevables qu'avec une discrétisation sommaire : les cellules, qui sont alors trop grosses, interdisent bon nombre de possibilités et restreignent abusivement l'espace des solutions.

- Une des solutions proposées par B. Faverjon et P. Tournassoud dans [FT87b], consiste à coupler un planificateur global à un planificateur local : la taille des cellules du planificateur global n'a plus besoin d'être aussi petite, car chaque cellule contient non pas un indicateur binaire de liberté ou de collision mais un indicateur de la probabilité d'atteindre effectivement le but par un chemin passant par cette cellule. Les indicateurs des cellules peuvent être mis à jour à l'avance, ou mieux encore par apprentissage, uniquement en fonction des besoins, suivant les échecs du planificateur local chargé de trouver une trajectoire sûre entre deux configurations de passage fournies par le planificateur global.

Définitions optimales des poses de soudures grâce au contrôle des collisions

Comme les poses de soudure sont définies par un point, une normale à la tôle et une ligne de pince, la direction de la ligne de pince est souvent choisie par l'opérateur suivant des critères de non-collision. Pourtant cette direction, lorsque la pince est choisie, peut jouir d'une certaine liberté, exprimée par l'intermédiaire d'un intervalle de rotation autorisé autour de la normale. Les programmes d'optimisation actuellement utilisés par l'industriel, que nous avons développés [Gom87], en tiennent compte seulement si l'opérateur a étudié cet intervalle manuellement pour chaque pose envisagée. Si le programme d'optimisation de la trajectoire dispose d'une protection contre les éventuelles collisions, c'est lui qui se charge de choisir l'angle de rotation autour de la normale à la tôle, en fonction de critères d'optimisation et de critères de collisions.

1.4 L'implantation du robot et les critères d'optimisation

De nombreuses directions de recherches sont possibles pour optimiser les tâches ; elles sont très variées et très originales, de surcroît elles reflètent bien

les désirs réels de l'industriel, même si, pour certains domaines, elles restent des préoccupations à priorité mineure.

Définition 12 *Dans le cadre de cette optimisation, il nous semble utile de définir la* **tension d'une trajectoire**. *Tendre une trajectoire, c'est optimiser la caractéristique fondamentale de celle-ci. La tension d'une trajectoire est ainsi la valeur de cette caractéristique. Ainsi, une trajectoire tendue peut être une trajectoire au cours de laquelle la distance minimale entre les objets en mouvements est aussi proche que possible de la limite minimale fixée, si la caractéristique principale est la sûreté de la trajectoire. Ce peut être aussi une trajectoire dont le mouvement associé est d'une durée proche de la durée minimale possible, si le critère est le temps. Ce peut être aussi une trajectoire de longueur proche du minimum possible, si tel est le critère.*

1.4.1 Les variables de l'optimisation

L'implantation du robot

Définition 13 *Réaliser l'implantation du robot consiste à situer sa base par rapport à l'atelier. De manière extrinsèque, il s'agit de définir le repère de sa base par rapport au repère absolu de l'atelier.*

Les robots ne peuvent être implantés que selon des orientations imposées par les contraintes technologiques de construction ou d'entretien, par exemple l'horizontalité de la fixation au sol. Les robots manipulateurs industriels sont d'ailleurs souvent disponibles en plusieurs variantes suivant qu'ils devront être fixés au plafond, au sol ou au mur. Le choix de base, une fois fait le choix du type de fixation, devient ainsi celui de la position (translation), avec éventuellement une rotation autour d'un axe vertical.

Les problèmes d'implantation des robots se posent notamment dans les domaines industriels suivants,

- lors de l'introduction d'une nouvelle fabrication de véhicule dans un atelier "flexible", puisque l'une des premières modifications à faire concerne la position des robots : c'est pourquoi choisir rationnellement une zone d'implantation possible des robots est véritablement très important.

- lors de la conception d'un poste robotisé, puisqu'il est important que la bonne position du robot par rapport à son environnement permette de réaliser un temps de cycle minimum.

P. Chedmail [CRW91] [CW89] propose d'utiliser un processus d'optimisation pour minimiser la partie du volume visé non accessible, ou bien pour minimiser la distance à ce volume visé impossible à atteindre. L'implantation du robot est ainsi optimale, au sens où le plus grand nombre de points visés possible sont accessibles. Les travaux de cette thèse proposent l'optimisation de l'implantation du robot en trouvant le domaine d'implantation s'il existe, et en implantant le robot au mieux à l'intérieur de ce domaine.

Le nombre d'axes du robot

Une autre direction de recherche serait de tenter de diminuer le nombre d'axes du robot nécessaires pour la réalisation d'une tâche donnée. Il faudrait ainsi optimiser la morphologie du robot en fonction de la tâche.

Le choix du nombre d'axes du robot n'est pas toujours un problème exclusivement technique : de nombreuses considérations de stratégie industrielle sont prises en compte, par exemple l'uniformisation des robots pour faciliter la maintenance, ou encore des considérations à long terme de flexibilité, pour un usage évolutif de tel robot en fonction des impératifs de production.

La forme de l'outil

Une direction de recherche tout à fait intéressante, puisqu'elle reflète directement les méthodes réelles et manuelles utilisées dans les ateliers, est l'optimisation de l'implantation du robot en fonction du temps de cycle, mais avec comme paramètre la structure et la forme de l'effecteur. C'est le problème de l'optimisation de la cale entre la platine, à l'extrémité du robot, et l'outil proprement dit. Ceci est très couramment utilisé en atelier. Dans ces conditions, c'est ainsi l'influence de la forme de l'effecteur sur le temps de cycle qui importe.

La tâche d'un groupe de robots

Comme fréquemment plusieurs robots coopèrent -un tient une pièce pendant qu'un autre visse, par exemple- l'optimisation peut avoir à porter sur un groupe de robots et même sur l'ensemble de la cellule robotisée. Dans ce cas, ce n'est pas seulement le temps de cycle, mais l'utilisation et la répartition qui deviennent des thèmes sensibles de l'optimisation.

1.4.2 Les critères de l'optimisation

L'accessibilité à chaque élément de la tâche, doit être bien sûr nécéssairement acquise pour envisager une quelconque optimisation. Si ce n'est pas le cas, le processus général envisagé, dont le but est d'optimiser la tâche, doit en priorité rendre accessibles les éléments de la tâche qui ne le sont pas.

Durée minimale de la tâche

C'est l'objectif principal : la tâche doit être réalisée dans le temps le plus court possible, avec comme contraintes de base les limites mécaniques du robot et les obstacles géométriques de l'environnement, en optimisant l'implantation du robot.

Pour le court terme, les préoccupations majeures actuelles s'orientent autour de la soudure par points pour la tôlerie. Dans ce domaine où 200 robots se répartissent 2500 points, le critère général est le temps.

D'autres directions peuvent être évoquées. Toujours en vue de diminuer le temps de cycle, mais dans une acception plus complexe, il s'agit d'une utilisation globale maximale du robot car, surtout au montage, il y a beaucoup de choses à faire en temps masqué, en parallèle avec une tâche principale.

Efforts mécaniques minimaux

Dans le but d'améliorer la maintenance, et de "fatiguer" le moins possible le matériel pour que les réglages durent plus longtemps, ce afin d'éviter surtout les à coups mais également les efforts d'intensité très grande, il peut être intéressant de configurer le robot en fonction des efforts auxquels il est soumis pour qu'il agisse avec la gravité plutôt que contre elle en évitant notamment les configurations singulières, en statique d'abord, mais également en dynamique. L'estimation des efforts auquel est soumis un robot en statique est accessible, celle des efforts auquel il est soumis en dynamique fait l'objet de recherches en vue de la commande des robots. Cependant, pour une optimisation suffisante des efforts, des approximations simples paraissent convenables. Ainsi D. Georges et Y. Hammam proposent en générant des trajectoires riches en informations dynamiques, de minimiser les pertes par effet Joule dans les n moteurs et les vitesses angulaires en moyenne quadratique [GH87], des contraintes supplémentaires pouvant éventuellement être ajoutées pour tenir compte des obstacles ou du suivi d'une trajectoire imposée.

Optimisation des configurations

On peut penser à une implantation optimale du robot pour que les configurations de blocage ou les configurations singulières soient évitées au maximum : par exemple, si la tâche consiste à souder un disque constituant le fond d'une cuve cylindrique, à cause des limites de débattement du robot, il sera certainement obligatoire d'arrêter la soudure, de retourner complètement le bras du robot, et de reprendre la soudure au point où on l'avait suspendue, avec une configuration du bras complètement différente. Ces configurations critiques sont fonction de l'implantation du robot, il est ainsi particulièrement intéressant de trouver une implantation dans laquelle les manoeuvres pour franchir les zones de singularité seront le moins nombreuses possibles.

Les configurations singulières du robot sont encore très insuffisamment traitées par les armoires de commandes. Une étude complémentaire serait utile pour comprendre comment sont évitées les configurations singulières, d'après le savoir faire des experts.

1.4.3 Les contraintes de l'optimisation

Sûreté optimale de la trajectoire

L'amélioration de l'emplacement du robot peut également être abordée sous l'angle de la sécurité. Sont souvent atteintes, quand un processus est optimisé, des configurations limites, par exemple une configuration du robot telle que celui-ci frôle, à un moment ou à un autre, une partie de son environnement, ou passe par une configuration dans laquelle un des axes vient en butée, ce qui risque de diminuer soit la fiabilité soit la reproductibilité de l'ensemble. L'optimisation de l'implantation du robot pourrait être organisée de telle sorte que ce qui bouge passe régulièrement le plus loin possible des obstacles fixes (c'est très important pour un atelier flexible) tandis que le robot s'écarte le moins possible de sa configuration médiane ou neutre. En tous cas, lors d'une optimisation du temps de réalisation de la tâche, il serait bon, si c'est possible, de prévoir une marge de sécurité suffisante sur les paramètres du robot par rapport à son environnement. D'autre part tous les éléments d'une même tâche n'exigent pas une fidélité égale suivant toutes les directions, il pourrait ainsi être intéressant d'optimiser l'implantation du robot pour privilégier les directions

les plus sensibles, ce d'autant plus que des objectifs de commodité ou de qualité technique sont également à atteindre.

Le passage au plus loin des obstacles a déjà fait l'objet d'une étude intéressante car une structuration particulièrement bien appropriée de l'espace libre induit naturellement la trajectoire optimale selon ce critère [Fav84]. Cette structure est un graphe classique de connexité entre les cellules libres, au dessus duquel est bâtie une structure hiérarchique arborescente d'octree subdivisant les cellules, chaque noeud ayant huit descendants.

L'espace des configurations structuré sous la forme d'un octree, permet de construire l'espace libre à partir de cellules de taille fine au voisinage des obstacles ou lorsque l'espace libre est de forme complexe, et à partir de cellules de tailles d'autant plus importantes que la zone d'espace libre est large. En affectant une pénalisation plus forte aux cellules de petite taille, le parcours du graphe des cellules libres (par un algorithme de la famille de "A*" permettant d'y trouver le plus court chemin), favorisera les trajectoires passant par les grosses cellules, et ainsi choisira une trajectoire plus sûre. Contrairement à une méthode par augmentation de la taille apparente des obstacles, celle-ci permet de trouver plus souvent une solution, même si, pour celà, il faut passer vraiment très près des obstacles. Le taux de pénalisation des cellules en fonction de leur taille permet de contrôler la "tension" de la trajectoire par rapport aux contours des obstacles : il est possible par exemple de considérer qu'à partir d'une certaine taille de cellule, la sécurité est suffisante pour ne pas inciter la trajectoire à passer dans des cellules de taille supérieure, sauf si le chemin y est plus court.

Travail minimum des flexibles

Dans la plupart des applications, de très nombreux flexibles sont chargés d'amener à l'outil fixé sur l'extrémité du robot les fluides et l'énergie dont il a besoin. Il s'agit souvent d'une alimentation de fort diamètre amenant le courant continu à forte intensité pour la soudure par points. Pour éviter la surchauffe, une circulation d'un liquide de refroidissement est nécessaire dans le flexible, dans l'outil et jusque dans les électrodes. Il faut ainsi amener deux canalisations d'eau à l'extrémité du robot. De plus, le fonctionnement mécanique de la pince est réalisé de manière pneumatique, ce qui impose des tuyaux d'air, qui peuvent être nombreux dans le cas d'un préhenseur à ventouses ou à verrous mécaniques.

Enfin, des câblages pour l'électronique sont nécessaires pour les dispositifs de commande et les capteurs.

Ces câbles et canalisations sont mis à rude épreuve, tout particulièrement par les rotations du poignet du robot. L'usure des électrodes de soudure reste certes le problème majeur mais ces éléments sont de loin les parties les plus sensibles du dispositif robotisé.

Une optimisation visant à limiter les efforts subis par ces flexibles et en particulier les torsions, peut être envisagée dans un premier temps en choisissant l'implantation du robot qui réalise un mouvement pour lequel l'éloignement des butées des axes du poignet est maximum.

Ceci sera réalisé de manière plus efficace lorsque l'estimation du comportement de ces flexibles aura une précision raisonnable, ce qui est l'objet d'études en cours [Ala90] [Jos89]. L'optimisation de ces efforts pourra alors être prise en compte au même titre que celle des efforts dans les corps du robot eux-mêmes.

Fonctionnement optimal des capteurs

Puisque l'utilisation des capteurs se développe, l'optimisation de l'implantation du robot ne peut se concevoir à plus long terme sans celle des capteurs qui le concernent. On peut envisager d'optimiser :

- la position des capteurs sur le robot et dans l'environnement en fonction de la tâche à effectuer.
- l'implantation du robot en faisant intervenir dans le processus d'optimisation les différentes informations fournies par les capteurs, par exemple en faisant intervenir pour le contournement des éventuels obstacles les réponses (simulées) enregistrées par les capteurs de proximité.
- l'implantation du robot en tenant compte de la position des capteurs.

Connaître la logique et sentir les intuitions qui conduisent les experts à implanter les capteurs liés au robot et à sa tâche sont des informations quasi indispensables pour vraiment établir les bases de ces processus.

Chapitre 2

Quelles bases de données CAO pour la robotique

Après l'analyse des fondements industriels permettant d'envisager l'implantation optimale, après une présentation des éléments de solutions apportées par la robotique et la recherche opérationnelle, il convient d'envisager le cadre de la modélisation envisagée. C'est l'objet de ce chapitre que de proposer pour nos outils d'évaluation des cellules robotisées, une réalisation en liaison directe avec le système général de conception CATIA, en se fondant sur sa base de données déjà employée préalablement à nos travaux pour définir les principaux objets qui interviennent dans la cellule robotisée. Ce chapitre propose un outil qui permet le calcul des distances entre polyèdres quelconques, puis un autre permettant le contrôle des distances sur l'ensemble du modèle de manière économique, sans remettre en cause de la base de données. Nous proposons enfin un outil efficace permettant de simplifier le modèle en transformant les surfaces restreintes en polyèdres.

Le développement de ces outils est un préalable nécessaire d'abord à la construction de trajectoires sûres, ensuite à la garantie de l'anti-collision pendant les phases d'optimisation de l'implantation.

2.1 Choix du système de CAO

Dans l'industrie automobile, pendant les cinq années de gestation du produit, de nombreux intervenants, ensemble ou successivement, concourent à l'élaboration de ce produit et à la conception du processus de fabrication. Il est important que les divers intervenants conçoivent les éléments sur un même système, car ils peuvent alors disposer de données communes et éventuellement les modifier d'une manière cohérente. La conception des cellules robotisées est notablement facilitée par l'utilisation d'un module de robotique intégré à ce système commun, même si, ce module n'étant qu'une petite partie du système général, il n'est pas doté des tous derniers perfectionnements dans ce domaine.

Certes, les logiciels spécialisés en robotique sont en général plus riches, mais les allers et retours vers le système commun de CAO ne sont encore réalisés que laborieusement ou de manière incomplète.

Comme l'ensemble des autres développements en robotique toutes nouvelles propositions, celles de cette thèse notamment, doivent venir s'ancrer sur le module robotique de base du système commun.

Choix d'un système de CAO : CATIA

L'un des deux systèmes communs actuellement utilisés chez l'industriel est CATIA (Conception Assistée Tridimensionnelle InterActive) de Dassault systèmes. C'est un système de dessin en trois dimensions (3D) qui bénéficie maintenant d'une longue expérience. Ainsi presque toutes les entités graphiques 3D sont-elles disponibles, ayant été introduites dans le système au fur et à mesure qu'elles passaient du stade de la recherche au stade industriel.

Ce système n'est opérationnel que sur gros ordinateurs IBM avec le système d'exploitation VM (machines virtuelles). Quelques bibliothèques de sous-programmes sont disponibles pour les programmeurs, pour leur permettre de lire, créer ou modifier certaines entités de la base de données, ou pour effectuer quelques opérations sur ces données. Cependant ces bibliothèques, apparues récemment, sont encore incomplètes.

La réalisation des programmes y est obligatoirement faite en Fortran.

Le développement du simulateur et de la modélisation, autour d'un logiciel de C.A.O. existant permet d'intégrer ces travaux dans la ligne de produits proposée par l'industriel à l'ensemble de ses services. [Gom88] et [Ver89]

Les caractéristiques essentielles de ce système

CATIA, autour de sa base de données et de son modeleur, dispose d'un ensemble de modules spécialisés chargés d'intégrer le savoir-faire et le métier de chaque classe d'utilisateurs : tournage, fraisage,...[Das89]

Le module cinématique permet de faire effectuer des mouvements à des ensembles poly-articulés quelconques, de définir les lois de déplacement, de vitesse et d'accélération des éléments. Il prévoit certes un modèle cinématique direct qui définit la position et l'orientation de l'extrémité active du robot en fonction des coordonnées articulaires, mais il ne prévoit pas un modèle cinématique inverse qui définirait des n-uplets de coordonnées articulaires en fonction de la position de cette extrémité active du robot. De ce fait il est, pour l'instant, mal adapté à la simulation des robots car il ne peut ainsi définir les mouvements dans l'espace ambiant.

C'est pourquoi il existe un module séparé pour les roboticiens. Ce module ne dispose cependant pas des mêmes richesses de mouvement car les lois de vitesses et d'accélérations ne sont définies qu'à partir d'un cas simple imposé (loi de vitesse trapézoïdale).

Mais ce module dispose de 3 fonctions complémentaires :

- **Robot**, qui permet de définir la cinématique du robot. Les robots y sont des chaînes de solides articulées sans boucles, admettant cependant des parallélogrammes vrais, bien que ce cas impose la création de nombreux axes auxiliaires pour le calcul,
- **Task**, qui permet de définir interactivement une séquence de réalisations à effectuer, telle que des mouvements,
- **WorkCell**, qui permet d'exécuter simultanément quelques tâches pour vérifier le comportement de la cellule robotisée : test de collisions dans certains cas particuliers, vérification des trajectoires, calcul sommaire du temps de réalisation des tâches.

L'intérêt d'utiliser un tel logiciel est de pouvoir disposer des définitions géométriques des pièces constituant la cellule robotisée, telles qu'elles sont

définies par les autres services de conception, sans transfert de données, donc rapidement mais surtout sans altération des données.

Le module GII pour l'intégration des développements externes

Un module marque nettement l'ouverture du logiciel aux développements d'applications spécifiques : GII (Graphic Interactive Interface), permet de développer des applications dont aussi bien l'ergonomie que le résultat présentent une apparence très proche de celles des fonctions standard du logiciel.

Les principales caractéristiques de ce module sont les suivantes :

- il met à disposition de l'utilisateur un macro-langage facilitant la description du dialogue, avec une interactivité comparable à celle des modules internes de CATIA, soit par un choix dans un menu, soit par une représentation sous forme arborescente de la séquence d'interactions à réaliser,
- il permet des échanges de données entre l'opérateur et le programme en appelant des sous-programmes,
- il autorise des temps d'exécution convenables grâce à une compilation de la fonction ainsi définie par l'utilisateur,
- il peut lancer des programmes écrits en Fortran pouvant faire appel à des fonctions élémentaires des modules internes.

Pour le développement des modules externes tels que ceux proposés pour l'application de nos travaux, les fonctions les plus essentielles de presque tous les modules internes sont disponibles en Fortran à travers deux bibliothèques :

- Catgeo, qui donne la possibilité de créer ou de modifier soit la géométrie soit les caractéristiques des entités présentes dans le modèle,
- Catmsp, qui permet d'effectuer quelques opérations mathématiques sur un groupe d'objets.

2.2 Modélisation de l'atelier en CAO

2.2.1 Les éléments de l'atelier robotisé

Description de l'atelier

Les objets utilisés en simulation robotique sont les suivants :

- les éléments du robot, systématiquement représentés par des solides, ce qui permet par exemple d'éliminer les parties cachées, de chercher leurs intersections éventuelles...
- le préhenseur, fixé à l'extrémité du robot, toujours défini sous forme de solide lorsque l'outil est une pince de soudure par points. S'il s'agit d'un appareil de préhension adapté à une pièce particulière, comme c'est le cas en montage, le préhenseur est modélisé soit sous forme de surfaces ou de faces, soit sous forme de solides, suivant la complexité des pièces et également suivant les habitudes des utilisateurs.
- l'objet fixé au préhenseur, comme le pare-brise, le tableau de bord, les fauteuils, les pièces de carrosseries,... est en général de forme complexe, sa modélisation est ainsi plutôt réalisée sous forme de faces ou de surfaces.

La modélisation de l'environnement est réalisée de deux manières, soit par un modèle surfacique, soit par un modèle solide :

- les objets qui sont une partie de l'atelier, comme les poteaux, les passerelles, les armoires de commandes, sont modélisés sous forme de solides, souvent de manière simplifiée,
- les pièces de carrosserie qui composent la voiture sont toutes définies sous forme de faces, depuis leur conception, ceci très longtemps à l'avance par les différents services qui ont participé à leur élaboration,
- les supports de ces pièces, tels que les chariots de transport, les fixations des pièces de voitures sur ces chariots, les serrages, même s'ils sont articulés, sont définis comme des solides,

- lorsqu'il s'agit d'un poste de conformation, dans lequel les éléments de la voiture qu'il faut assembler sont maintenus en place par des appareils articulés, souvent de grande dimension, nombreux et de forme très "technique", les conformateurs sont définis sous forme de solides.

2.2.2 Les outils de représentation : surfaces restreintes et solides

Les surfaces restreintes

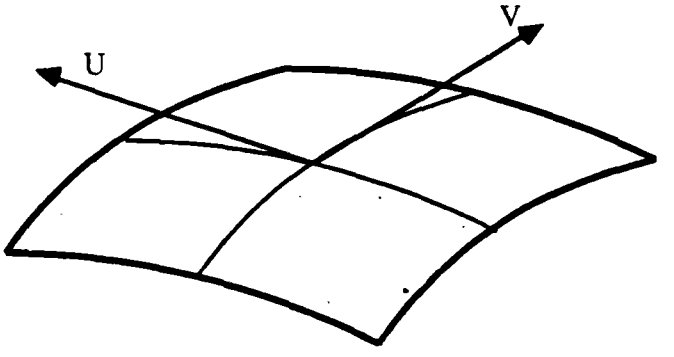


Figure 2.1 : Représentation d'une surface et de ses deux paramètres u et v

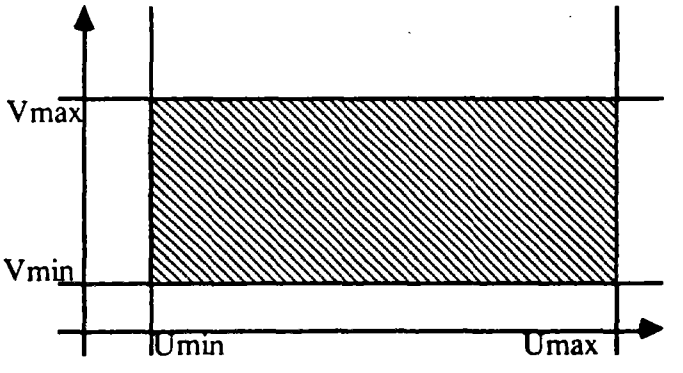


Figure 2.2 : Surface : domaine de validité de ses paramètres u et v

Définition 14 *Les SURFACES sont définies en suivant les figures 2.2et 2.2+1 par :*

$$x = \mathcal{F}_x(u, v)$$
$$y = \mathcal{F}_y(u, v)$$
$$z = \mathcal{F}_z(u, v)$$

où $\mathcal{F}_x$, $\mathcal{F}_y$, $\mathcal{F}_z$, *sont des polynômes en* u *et* v

$u \in [u_{min}; u_{max}]$ *avec en général* $u \in [0; 1]$

$v \in [v_{min}; v_{max}]$ *avec en général* $v \in [0; 1]$

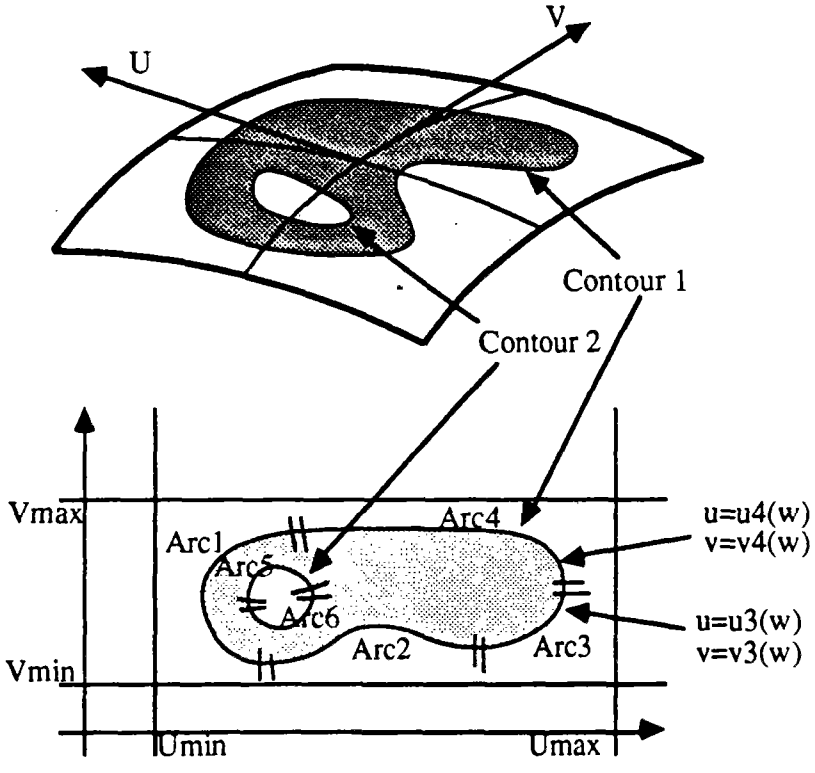


Figure 2.3 : Représentation des paramètresd'une face (surface restreinte) : les contours sont définis dans l'espace des paramètres u et v à l'aide d'un paramètre de contour w

Définition 15 *Les FACES sont définies (figure 2.3) par une surface support et des contours définis par une suite d'arcs définis chacun par :*

$$u = \mathcal{G}_u(w)$$
$$v = \mathcal{G}_v(w)$$
où $\mathcal{G}_u$ *et* $\mathcal{G}_v$, *sont des polynômes en* w
$w \in [w_{min}; w_{max}]$ *avec en général* $w \in [0; 1]$

Les bibliothèques d'ouverture de CATIA aux développements externes autorisent les actions suivantes :

- Girbas : lecture de la surface supportant la face,
- Girnli : lecture du nombre de contours de la face,
- Gislim : lecture des contours limitant la face,
- Gsoiof : position dedans/dehors en fonction des paramètres de la face.

Les solides

Les solides peuvent être définis suivant trois représentations complémentaires les unes des autres : [Bel87]

- B-Rep (Boundary Representation), pour laquelle le solide est défini par ses frontières de manière exacte,
- CSG (Constructive solid Geometry), pour laquelle le solide est défini par un arbre de construction à partir de primitives simples (prismes, cônes, cylindres,...) et d'opérations mettant en oeuvre ces primitives,
- Polyèdres, variante du B-Rep qui se caractérise par une approximation de la frontière du solide au moyen de morceaux plans et polygonaux formant des sommets et des arêtes.

Dans toutes les modélisations possibles des volumes avec CATIA les solides sont systématiquement approchés par des *polyèdres* qui sont représentés sous la forme d'une double liste de sommets et de faces planes au contour polygonal fondé sur les sommets.

Cependant CATIA donne accès à l'arbre de construction de ces solides, c'est à dire accès aux *primitives* solides. Celles ci sont :

- les sphères,
- les cylindres,
- les cônes,
- les parallélipipèdes parrallèles aux axes de coordonnées,

... mais aussi des solides construits sur d'autres entités telles que :

- les courbes : solides de révolution, prismes, tuyaux, structures,
- les surfaces : solides obtenus par discrétisation, épaississement...

Ces primitives peuvent être non-convexes telles que :

- les tuyaux,
- les tores,
- les prismes.

Certaines primitives ne sont même pas des solides, telles que :

- les plans : en vue d'une coupe,
- les surface : en vue d'une coupe.

De plus ces primitives sont associées par des opérations autres que l'union, telles que :

- coupes par une surface,
- intersection,
- différence.

De surcroît, l'utilisateur a la possibilité de "détruire" l'arbre de construction pour ne garder que le solide final, ce qui peut faire gagner de la place dans des modèles CAO déjà bien surchargés.

De ce fait il est difficile d'utiliser à plein de tels arbres, puisque on ne peut passer facilement des propriétés d'un des éléments de l'arbre à celles d'un autre : par exemple s'il y a collision avec les solides élémentaires $\mathcal{S}_A$ et $\mathcal{S}_B$, y a t-il collision avec le solide $(\mathcal{S}_A \cap \mathcal{S}_B)$?

Définition 16 *Une FACETTE est formée d'une liste de contours polygonaux plans formés sur des sommets ordonnés successifs, le premier contour représentant le contour extérieur, les suivants (si nécessaire) les éventuels contours intérieurs correspondant à des trous. Chaque Facette est orientée par la définition d'une normale.*

Définition 17 *Le POLYEDRE est une liste de sommets et de Facettes planes formées à partir de ces sommets (figure 2.4).*

Définition 18 *Un SOLIDE est un polyèdre particulier parce qu'il est nécessairement fermé. Les Facettes qui le composent sont toutes orientées vers l'extérieur du solide (figure 2.4).*

Ce concept de solide permet beaucoup d'opérations particulières impossibles avec les polyèdres, comme des opérations ensemblistes, la suppression des parties cachées, et facilite beaucoup les algorithmes [Mar88].

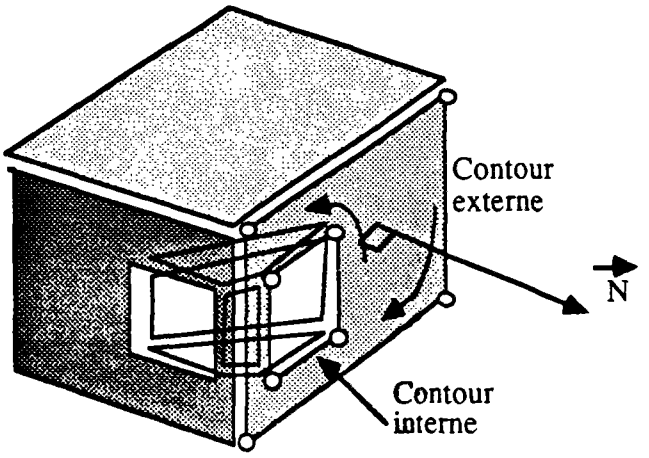


Figure 2.4 : Solide et polyèdre : Ce sont des collections de facettes planes avec éventuellement des trous. Les contours internes ou externes son polygonaux.

2.2.3 Les problèmes des surfaces restreintes et ceux des solides pour l'étude des collisions

Le problème des faces : leurs bords

Les problèmes concernant les faces sont beaucoup plus difficiles à traiter que ceux concernant les surfaces à cause du fait qu'en manipulant les paramètres de la surface de support, il est possible de sortir à l'improviste de la face. Tenir convenablement compte des paramètres décrivant les bords, en ne se bornant pas aux seuls deux paramètres décrivant la surface, pour définir une direction de recherche convenable est difficile.

Même le test consistant à savoir si on est à l'intérieur de la face n'est pas aussi facile ni aussi rapide qu'on le souhaiterait. Ce n'est pas plus facile pour l'opération consistant à projeter un point de la surface sur le bord.

Les problèmes entre solides et surfaces restreintes

Il est inévitable d'avoir à considérer à la fois des solides et des surfaces : en effet les tôles sont naturellement définies par des surfaces, de même lors de la conception d'objets de formes complexes comme les planches de bord, il est plus facile de les modéliser par des surfaces.

Mais pour la détection de collisions, on a besoin d'informations à base de topologie qui font défaut dans la définition en tant que surfaces : c'est la modélisation en tant que solide qui correspond tout à fait aux exigences des calculs relatifs aux collisions. Si ces deux entités cohabitent dans la plupart des logiciels de CAO, en revanche la manipulation des deux conjointement est toujours très délicate car l'opérateur est toujours obligé de choisir son mode de modélisation, les ponts entre les deux modes n'existant de fait pas, obligeant ainsi l'opérateur à transformer péniblement les entités d'un modèle dans l'autre.

Les solides et les surfaces sont en effet des entités complètement étrangères l'une à l'autre.

Que se passe-t-il s'il est envisagé de transformer les surfaces restreintes en solides ? La taille du modèle devrait croître alors très vite car la définition des solides prend beaucoup de place : l'ordre de grandeur du nombre de surfaces atteint couramment 2000, il semblerait ainsi tout à fait irréaliste d'envisager de transformer toutes les surfaces en solides, cela ne semble possible que pour quelques unes, ou bien le cas échéant "au vol" pour traiter le cas d'une collision probable.

En fait, si la définition des solides est effectivement très coûteuse, la définition des faces l'est très sensiblement moins. Mais la création des faces impose la présence de courbes définissant le contour et de surfaces support. La création des faces forme des liens entre ces entités, entités qu'il sera impossible de détruire. De plus, la présence simultanée et inévitable de courbes de contour définies sur la surface en paramétrique par rapport à cette surface (*EDGE de Catia), et de courbes correspondantes définies en coordonnées cartésiennes (*CRV de Catia), font double emploi : elles ont vraisemblablement permis un calcul plus rapide mais elles encombrent le système.

Ainsi la transformation des faces en polyèdres ne rajoute pas une quantité inacceptable de données. De plus, les faces d'origine restent indépendantes des polyèdres qu'elles ont engendrés, ce qui rend possible leur effacement : le modèle prend alors nettement moins de place en Base de Données que le modèle d'origine.

Exemple : Pour un modèle contenant toutes les faces, surfaces et courbes définissant l'aile avant et sa doublure pour un véhicule (Peugeot 605), on obtient, avec une définition polyédrique très fine de 20 mm pour la longueur

maximale du coté des facettes : le tableau suivant permet de comparer les tailles en kb dans la base de donnée :

	initial (faces seules)	Faces et polyèdres	Polyèdres seuls
Index	237 Kb	266 Kb	29 Kb
Data	1135 Kb	1785 Kb	650 Kb

La table d'index correspond aux pointeurs stockés pour les liens entre les données qui se correspondent, la table des Data correspond aux données numériques de la définition géométrique des objets.

2.3 Contrôle rapide des distances

Le contrôle des distances sur l'ensemble du modèle numérique décrivant la cellule robotisée est de première importance pour l'évitement d'obstacles et le maintien d'une trajectoire sûre pendant le processus d'optimisation de l'implantation du robot. C'est pourquoi cette section traite d'abord du calcul de distance entre deux solides, puis d'une structure hiérarchique simple permettant d'éviter le calcul systématique de distances précises entre tous les couples de solides en présence. Les algorithmes présentés ont été programmés et testés sur des modèles industriellement réalistes.

Une première partie traite de la structure qu'il est possible de mettre en place sur la description numérique de la cellule pour un coût de calcul préparatoire non prohibitif. En effet, la *réussite* des algorithmes, la *simplicité* et l'*efficacité* des méthodes employées, dépendent considérablement de la façon dont les données sont organisées, c'est à dire de la représentation de la scène et des mécanismes. M. Basseville [Bas87] par exemple propose une méthode de calcul de distances efficace surtout pour la reconnaissance des formes. La plus grande part des méthodes utilisées en CAO pour effectuer précisément une tâche donnée sont fondées sur l'utilisation efficace d'une structure de données très adéquate. Cependant, les données géométriques sont définies avec l'aide du système de CAO standard et peuvent être utilement modifiées pour la mise en oeuvre du processus robotisé. C'est pourquoi nous tenterons d'établir une structure éfficace en fonction des possibilités de la base de données et des structures sous-jacentes.

2.3.1 Proposition d'un algorithme de calcul de la distance entre des polyèdres quelconques

Certes cet algorithme est extrêmementement coûteux, il n'est vraisemblement pas un optimum définitif, mais il est capable de traiter sans risque les polyèdres quelconques...

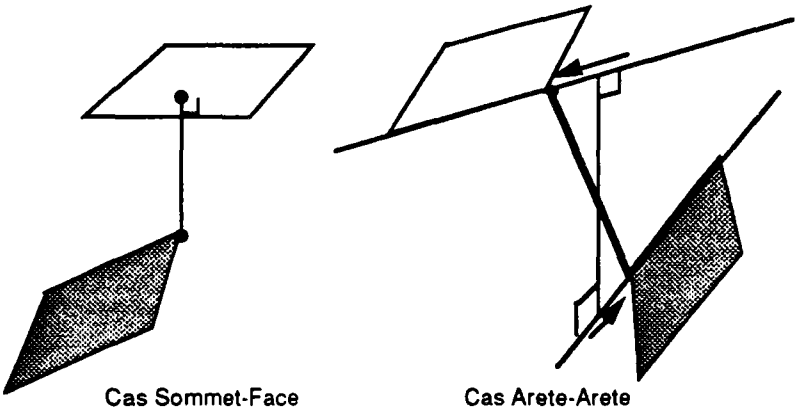


Figure 2.5 : Les différents cas possibles pour le calcul de distance entre deux faces : soit par projection d'un sommet sur une face, soit par calcul de la perpendiculaire commune à deux arêtes

Les points concernés sur chacun des solides qui concrétisent la distance minimale pour des solides non convexes polyédriques, sont (figure 2.5) :

- soit l'un un sommet et l'autre dans une face
- soit l'un un sommet et l'autre sur une arête
- soit l'un un sommet et l'autre un sommet
- soit l'un sur une arête et l'autre sur une arête
- soit l'un sur une arête et l'autre dans une face
- soit l'un dans une face et l'autre dans une face

Tous les cas ne concernant que des sommets et des arêtes peuvent être traités ensemble comme cas particuliers du cas général arête-arête.

Algorithme :

```
distance_solide_solide(solide_1,solide_2)
/* calcul de distance entre deux solides */
{
    répéter pour toutes les arêtes de solide_1 {
        répéter pour toutes les arêtes de solide_2 {
            Calculer_perpendiculaire_comune(aretes)
            Calculer_points_concernés(aretes,perpendiculaire)
            Mise_a_jour(meilleure_distance)
        }
    }
    répéter pour tous les sommets de solide_1 {
        répéter pour toutes les facettes de solide_2 {
            distance_mini = distance_sommet_facette(sommet,facette);
        }
    }
    répéter pour tous les sommets de solide_2 {
        répéter pour toutes les facettes de solide_1 {
            distance_mini = distance_sommet_facette(sommet,facette);
        }
    }
}


distance_sommet_facette(sommet,facette)
/* calcul de distance entre un point et une facette polygonale plane */
{
    point_1 = Projection(sommet,plan(facette) );
    Demie_droite=creer_demie_droite(point_1,premier_sommet(facette) );
    répéter pour toutes les segments du contour de la facette {
        si(intersection(demie_droite,segment)=vrai) {
            incrementer(nombre_intersections)
        }
    }
    si(nombre_intersections=pair) {
    /* le point_1 est à l'interieur de la facette si c'est pair*/
        mettre_a_jour(resultat_interraction)
    }
    /* le cas impair sera trouvé d'une autre manière*/
}
```

Cas particulier : méthode de calcul de distance utilisant la représentation des solides par leur arbre de construction (CSG)

Ceci est proposé par B. Faverjon [FT86].

Si les solides sont représentés sous forme de solides élémentaires appartenant à une classe bien définie telle que sphères, cylindres, cônes, parallélépipèdes...

le calcul de projection d'un point sur un solide est immédiat, obtenu par un calcul simple, mais différemment organisé pour chaque type de solide. Il a été montré [GJK88] [Bob89] [Tou88] que si le solide est un polyèdre quelconque mais convexe, la quantité de calcul pour la projection d'un point sur ce solide est, dans les meilleures conditions, sensiblement proportionnelle au nombre de sommets, tandis que si le polyèdre n'est pas convexe, elle est au mieux proportionnelle au carré de ce nombre.

C'est une méthode basée sur le fait que les obstacles sont décrits par des *unions* de solides élémentaires aussi simples que possible. Dans ces conditions, les procédures de calcul du premier point d'une primitive solide (parallélépipède, cylindre, cône, sphère...) rencontré dans une direction donnée et la projection d'un point sur une primitive solide sont réduits à quelques calculs assez simples pour être considérés de coût unitaire.

Les primitives étant toutes convexes, les tuyaux, les tores en sont par exemple exclus, le calcul des distances s'effectue par projection alternée sur l'un puis l'autre solide.

Au départ, on choisit un point dans le solide $\mathcal{S}_1$ -au hasard- que l'on projette, orthogonalement si possible, sur le solide $\mathcal{S}_2$. Le point trouvé est projeté de la même manière sur le solide $\mathcal{S}_1$. Par itération, si les solides sont convexes, la convergence a été démontrée comme assurée [Tou88].

Cette option est séduisante, cependant la restriction qui consiste à ne considérer que des opérations d'union ensembliste est contraignante, tout à fait supportable cependant pour un système de simulation robotique indépendant et rustique.

Dans nos cas industriels, les définitions numériques des objets de l'atelier robotisé sont réalisées en dehors du cadre restreint de la robotique. L'intégration du modèle utilisé pour la robotique au sein du système commun de conception du produit, conduit à profiter des richesses communes, notamment de définition des solides. CATIA autorise une liste importante d'opérations ensemblistes et d'autres opérations (voir section 2.1). Nous avons choisi de ne pas limiter l'opérateur pour les définitions géométriques nécessaires au modèle ; ainsi une association convenable au logiciel de CAO est-elle assurée, au prix d'un temps de réponse du logiciel plus important.

Cas particulier : méthode de calcul de distance utilisant un modèle hiérarchique de l'espace

La structuration de l'espace de la cellule robotisée est une véritable nécessité : il est hors de question de calculer des distances entre les corps à grand frais s'il est possible de s'en dispenser car il serait vraiment d'un coût exorbitant de calculer des distances ultra-précises entre tous les couples d'objets élémentaires de l'espace. A titre d'exemple une doublure d'aile avant est formée d'environ 300 éléments...

Beaucoup de propositions ont été faites dans ce domaine. La plupart sont d'un usage général. Certaines offent un intérêt particulier pour le problème de contournement des obstacles ou offrent des possibilités permettant de traiter le problème de la planification de trajectoire sûres de manière originale :

Octree dans l'espace des configurations

Une méthode originale est proposée par B. Faverjon [Fav84]. Il est utile de la citer ici car elle permet un contournement des obstacles sans obligatoirement serrer au plus près ceux-ci, en permettant même de passer le plus au large possible.

Par une représentation transformée des obstacles dans l'espace des configurations, elle peut proposer une solution efficace au problème de la détection de collisions. Il est toujours délicat de calculer le volume balayé par le robot pendant les mouvements, or avec cette méthode ce n'est plus nécessaire. Cependant le problème qui consiste à trouver un chemin sans collisions à partir des informations de collisions pour un chemin donné reste toujours difficile à résoudre car les modifications du chemin restent limitées à chaque étape, aussi le chemin trouvé est-il rarement optimal.

Les caractéristiques de cette méthode sont l'utilisation d'espaces abstraits (espace des configurations), une décomposition du problème (bras/poignet) et surtout une description hiérarchique de ces espaces abstraits. Elle utilise des algorithmes globaux.

La description hiérarchique entre cellules de l'espace des configurations est justifiée parce qu'il n'est pas nécessaire de décrire homogènement tout l'espace notamment si on préfère passer le plus loin possible des obstacles plutôt que parcourir le plus court chemin au sens strict. Ainsi le chemin ne passera-t-il près des obstacles que si c'est absolument indispensable.

Le bras n'a généralement pas à passer près des obstacles, il peut ainsi être décrit avec quelque approximation. Mais le poignet doit, lui, passer plus près

des obstacles.
Les mouvements initiaux et finaux sont généralement contraints par un vecteur d'approche imposé. Une heuristique est utilisée pour changer la position et l'orientation du poignet jusqu'à ce qu'il se soit suffisamment éloigné des obstacles.
Les grands mouvements du bras sont étudiés en suivant un algorithme tendant à former le chemin minimum parmi ceux qui passent convenablement le plus loin possible des obstacles.
Les mouvements du poignet pendant les grands mouvements sont traités séparément, de manière plus simple si au stade de l'étape des grands mouvements tout a bien fonctionné.

Il est proposé de minimiser la durée de la trajectoire dans le graphe lié à l'arbre en prenant en compte d'une part un terme lié à la longueur dans l'espace des configurations des segments de trajectoires à vitesse constante, d'autre part un terme prenant en compte les ralentissements de vitesse dans les virages.

Pour une sûreté maximale de la trajectoire présentée dans la section sur les contraintes de l'optimisation au chapitre 1, pour forcer le système à passer loin des obstacles, il est possible d'affecter un coût plus élevé aux cellules les plus petites, ou encore limiter l'utilisation de l'octree aux niveaux les plus bas, en utilisant des cellules qui soient plus grandes que les feuilles de l'arbre, puisque la structure proposée les prend en compte.

L'utilisation de l'octree permet d'obtenir un facteur de réduction de l'ordre de 10 sur le temps de recherche de trajectoire, à cause de la diminution considérable du nombre de cellules mises en cause et des liens établis entre celles ci.

Cas particulier : Calcul rapide des distances utilisant des graphes de voisinages

S'il a des ressemblances partielles avec le graphe du modèle hiérarchique de l'espace décrit dans le paragraphe suivant, ce graphe en est différent, car il s'agit de décrire les proximités entre objets, de manière à prévoir les interactions possibles à l'état (n+1) de la cellule en fonction des interactions à l'état (n) et du déplacement envisagé.

Une anticipation des mouvements est possible. Cette anticipation est justifiée en remarquant que les éléments du robot et de l'organe terminal se déplacent à vitesse limitée ; pendant un intervalle de temps donné, ils ont parcouru une

distance limitée, ainsi le robot ne peut-il entrer en collision avec les objets qui étaient à une distance supérieure à celle-ci avant le mouvement. On peut calculer a priori la vitesse maximale d'un point du robot ou de l'organe terminal, tester les distances entre tous les objets, faire la liste ; à la fin du mouvement élémentaire, il n'est pas utile de tester à nouveau les collisions de celui-ci avec les objets qui étaient suffisament loin pour que le robot n'ait pu les atteindre même avec sa vitesse maximale pendant la durée du mouvement.

2.3.2 Définitions à précisions multiples dans l'espace ambiant

Certains auteurs comme [BSNA89] suggèrent une modélisation "conceptuelle" de l'environnement à l'aide de systèmes experts d'analyse de l'environnement du robot. Plutôt que d'être fondée sur ces systèmes d'analyse indisponibles le plus souvent aujourd'hui, l'approche envisagée utilise les informations sur la structure de l'environnement facilement extraite à partir du modèle tel qu'il est décrit à travers le système standard de conception.

La structuration proposée est fondée sur la modélisation standard de l'atelier, exactement telle que définie par les services impliqués en amont dans la définition des composants, et telle que la souhaitent les intervenants en aval du processus.

La pièce est composée d'une multitude de morceaux juxtaposés : ainsi les pièces compliquées contiennent-elles beaucoup de morceaux de taille réduite, les pièces simples en contenant peu. Par exemple une carrosserie complète de véhicule peut compter 4000 entités, une doublure d'aile (c'est il est vrai un élément très compliqué) environ 300 éléments, un pare brise 3 éléments pour chaque face du verre.

D'après les observations du paragraphe 4.1, seules les pièces de carrosserie sont généralement définies comme des surfaces. Or il est peu probable que la mise au point de la cellule robotisée amène à transformer ces pièces. Cette observation sera mise à profit pour les transformer en polyèdres, puisque aucun retour dans la chaîne de conception n'est envisagé.

Les éléments constituant la carrosserie sont rangés individuellement dans des groupes différents appelés "SETS" (notion standard du système commun). On

obtiendrait ainsi un modèle hiérarchique, à deux niveaux seulement, si la notion de set contenait des informations géométriques telles que le volume-enveloppe, même avec une définition grossière, mais ce n'est pas le cas.

Les informations fournies en standard par la base de données ne permettent pas de calculer les distances de manière convenable, ce qui a conduit à utiliser des sous-programmes de lecture mieux adaptés. On peut alors disposer d'entités aussi simples que possible pour représenter l'encombrement des objets : les enveloppes simples extrinsèques ou intrinsèques. L'utilisation de l'enveloppe convexe d'un solide [Pre77], ou la décomposition des solides en solides convexes, permettent d'utiliser des algorithmes plus rapides car leur complexité est généralement proportionnelle au nombre de sommets du solide initial.

Enveloppes simples extrinsèques

Définition 19 *Les boîtes d'encombrement (figure 2.6) : la boîte d'encombrement d'un objet est le plus petit parallélépipède à faces parallèles aux axes de coordonnées qui contienne entièrement l'objet.*

L'utilisation des "boîtes d'encombrement" est à la fois efficace et rapide car ces boîtes peuvent directement être extraites des données de la CAO, leurs éventuelles collisions sont très faciles à traiter par un test simple : les cas de collisions "facilement observables" peuvent ainsi être très rapidement explorés.

Mais il est manifeste qu'elles ne sont qu'une image quelque peu simpliste de l'objet ; bien évidemment, comme pour toutes les enveloppes extrinsèques, par définition leur forme dépend énormément de l'orientation des axes de coordonnées.

Enveloppes simples intrinsèques

Définition 20 *Les sphères d'encombrement (figure 2.7) : la sphère d'encombrement d'un objet est la plus petite sphère qui contienne entièrement l'objet.*

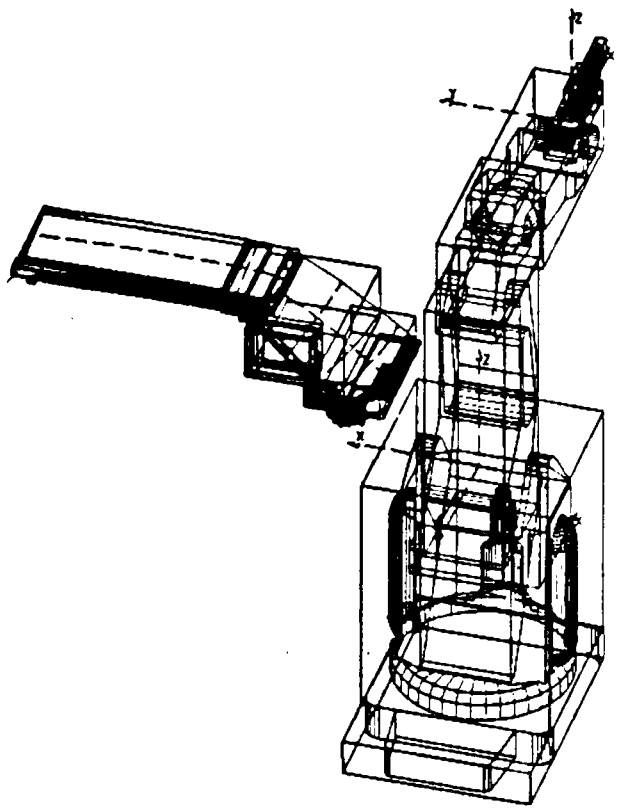


Figure 2.6 : Boîtes d'encombrement d'un robot et d'une partie d'une voiture (modèle Catia)

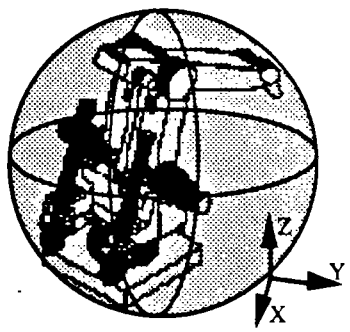


Figure 2.7 : Sphère d'encombrement d'un robot

Remarque 4 *La définition de la boîte d'encombrement est une définition extrinsèque. Ainsi dépend-elle des axes de coordonnées choisis. Ceci impose au cours du mouvement du robot de recalculer constamment la boîte correspondant à la position atteinte. Cet inconvénient ne peut être contourné : il est possible de définir cette boîte dans un repère lié au corps de l'objet, mais alors les calculs d'intersection sont beaucoup plus compliqués, ce n'est plus un simple test. L'intérêt de cette définition est d'être sommaire, ce qui abrège les calculs.*

La définition de la sphère d'encombrement est intrinsèque et aucun "recalcul" n'est à prévoir, si ce n'est évidemment celui consistant à appliquer au point définissant le centre de la sphère le même déplacement que celui du corps auquel elle est rattachée.

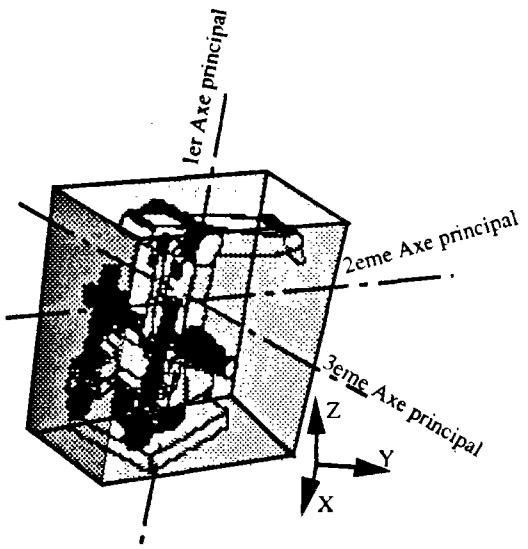


Figure 2.8 : Boîte principale d'un robot

Définition 21 *La Boîte principale (figure 2.8) : c'est le plus petit parallélépipède contenant l'élément dont les faces sont perpendiculaires aux axes principaux de l'objet.*

Une fois déterminés las axes principaux, il suffit de construire la boite d'encombrement dans le repère associé.

Ces boîtes principales, par rapport aux boîtes d'encombrement, ont l'avantage de correspondre de manière plus fidèle à la forme générale de la pièce. Elle ne dépendent plus du repère choisi pour l'atelier, mais le calcul de distance entre ces boîtes sera moins simple que pour les boîtes d'encombrement ordinaires.

Description des éléments et des sous-ensembles

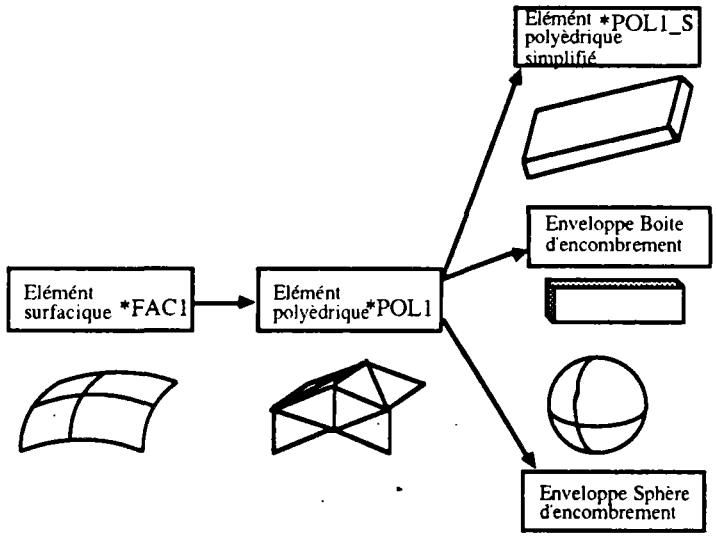


Figure 2.9 : Précision multiple pour un élément : 3 enveloppes simplifiées permettent des choix rapides dans les cas simples, sinon la représentation polyèdrique précise est utilisée.

Lors de la polyédrisation des surfaces restreintes élémentaires l'approximation est réalisée avec deux précisions distinctes (figure 2.9).

La précision demandée par l'utilisateur construira un polyèdre approchant la surface de manière suffisament précise pour que le calcul de distance soit valide pour l'application envisagée par l'opérateur, par exemple, de l'ordre du demi-centimètre pour les applications robotiques ordinaires.

Une précision nettement moindre aura pour but de construire un gros polyèdre modélisant sommairement le sous-ensemble ou l'objet, auquel appartient l'élément surfacique considéré.

- Cette précision peut être choisie en fonction de celle donnée par l'opérateur, par exemple dix fois plus grossière que la précédente.

- Elle peut être choisie minimale, c'est à dire telle que le polyèdre correspondant contienne le minimum de facettes (une ou deux), ce qui est obtenu en demandant une précision volontairement trop grossière pour la discrétisation : le polyèdre s'appuiera alors uniquement sur les points caractéristiques du contour de l'élément.

- Une variante de ce principe serait que le polyèdre grossier de base soit constitué d'une seule facette par élément du set : on "discrétise" le contour extérieur avec une précision beaucoup plus grossière que celle choisie par l'utilisateur, on calcule un plan moyen passant par ces points, on projette les points sur ce plan et on obtient un contour polygonal d'une facette : toutes ces facettes seront ensuite assemblées pour former le polyèdre grossier modélisant le sous-ensemble.

Tous les polyèdres grossiers ainsi définis pour chaque élément du sous ensemble, sont assemblés pour former un polyèdre représentatif de la géomérie du sous-ensemble.
Le même principe de précision multiple pour un ensemble d'éléments que pour les éléments peut être appliqué comme sur la figure 2.9.

Description de l'atelier

Pour passer de la représentation polyédrique grossière du "set" (groupe d'éléments) aux polyèdres précis des ses éléments l'atelier robotisé est structuré en différentes parties :

- les objets fixes, c'est à dire les éléments de l'environnement, les différentes parties de l'atelier, les chariots, les pièces de serrage, les pièces de conformation, et bien sûr les pièces constituant la partie fixe de la carrosserie sur laquelle le robot viendra réaliser des soudures ou placer des sous-ensembles.

  Si certaines pièces de cette partie s'interpénètrent ou sont au contact, il s'agit vraisemblablement d'une définition trop sommaire de l'environnement ou d'un état toléré par le concepteur. Ces collisions ne sont pas imputables au processus robotisé et ne seront par conséquent pas testées, ce d'autant moins que tout effort par le robot pour tenter de corriger cela serait inutile puisqu'il n'en est pas responsable.

- les corps du robot : la définition du robot doit prendre en compte des limites de débattement sur chacun des axes qui peuvent être commandés. Ces limitations tiennent compte des collisions possibles entre deux corps. Dans ces conditions, s'il y a collision entre deux corps consécutifs du robot, cela est dû soit à une modélisation imprécise du robot, soit à une erreur dans l'appréciation des limites de débattement. Le processus à réaliser au moyen du robot ne peut corriger les premières et la prise en compte des dernières ferait double emploi.

- les corps fixés à l'extrémité du robot : le dernier axe du robot, le préhenseur et les objets maintenus dans le préhenseur sont solidaires. S'il y a collision entre eux, cela est dû à une définition imprécise du préhenseur ou des différentes pièces. Une modification du process à robotiser ne peut la corriger.

Ainsi le test des collisions peut-il être réalisé à plusieurs niveaux :

- Premier niveau :

  - collisions entre tous les corps transportés par le robot (par exemple le préhenseur et la pièce fixée dessus) et tous les corps du robot (chaque élément mécanique, du premier à l'avant dernier axe), sauf sa base, avec l'environnement,
  - collisions entre chacun des corps du robot et tous les autres corps du robot qui ne sont pas ses voisins dans la chaîne articulée,
  - collisions entre les objets transportés par le robot et chaque axe du robot, sauf le dernier.

  Dans ces conditions par rapport à une étude systématique de tous les couples de sous-ensembles, l'économie de calculs est appréciable : environ les deux-tiers des couples de sous-ensembles n'ont pas à être étudiés, bien davantage encore au niveau des éléments.

- Deuxième niveau :

  Le calcul est réalisé entre les sphères d'encombrement de chaque sous-ensemble. Dans le cas où la distance entre les sphères est supérieure à la distance d'influence, il n'est pas nécessaire de préciser davantage.

Puis le calcul est réalisé entre les boîtes d'encombrement de chaque sous-ensemble. Dans le cas où la distance entre les boîtes d'encombrement est supérieure à la distance d'influence il n'est pas nécessaire de préciser davantage.

Ensuite le calcul est réalisé entre les boîtes principales de chaque sous-ensemble. Dans le cas où la distance entre les boîtes principales d'encombrement est supérieure à la distance d'influence il n'est pas nécessaire de préciser davantage.

Enfin le calcul est réalisé entre les polyèdres grossiers décrivant chaque sous-ensemble. Dans le cas où la distance entre les polyèdres grossiers est supérieure à la distance d'influence il n'est pas nécessaire de préciser davantage. Si tel n'est pas le cas, les couples de facettes qui sont en interaction sont repérés. Par cette manière simple de connaître l'élément qui correspond à chaque facette du polyèdre grossier, on obtient une liste de couples d'éléments en interaction possible, le premier élément du couple appartenant au premier sous-ensemble, le second élément appartenant à l'autre sous-ensemble.

Si ces étapes n'ont pas donné satisfaction, il faut faire le calcul de distances au niveau des élements de ces couples.

Pour chaque couple possible le calcul est réalisé entre les sphères d'encombrement des deux éléments. Dans le cas où les sphères sont distantes de plus que la distance d'influence, il n'est pas nécessaire de préciser davantage.

Puis le calcul est réalisé entre les boîtes d'encombrement des deux éléments. Dans le cas où la distance entre les boîtes d'encombrement est supérieure à la distance d'influence il n'est pas nécessaire de préciser davantage.

Ensuite le calcul est réalisé entre les boîtes principales des deux éléments. Dans le cas où la distance entre les boîtes principales d'encombrement est supérieure à la distance d'influence il n'est pas nécessaire de préciser davantage.

Enfin le calcul est réalisé entre les polyèdres décrivant de manière précise la géométrie des deux éléments.

Ce mécanisme se déduit de la figure 2.9.

2.3.3 Algorithme programmé correspondant

Calcul des distances sur tout le modèle

```
distance_modele(outil,robot,environnement)
/* calcul general de distance sur toute la cellule robotisée */
{
    repeter pour tous les sous_ensembles SM du modele { ;
        si(SM n'appartient pas au robot) {
            si(SM appartient a outil) {
                repeter pour tous les sous-ensembles du robot SR sauf le dernier axe { ;
                    d = Distance_sous_ensembles(SR,SM) ;
                }
            sinon
                repeter pour tous les sous ensembles SRO du robot ou de l'outil { ;
                    d = Distance_sous_ensembles( SRO,SM) ;
                }
            }
        }
    }


distance_sous_ensembles(Sa,Sb)
/* calcul general de distance entre deux sous ensembles */
{
    d = Calcul_distance_sphères(Sa,Sb) ;
    si(d < distance d'influence) {
        d = Calcul_distance_boite_principale(Sa,Sb) ;
        si(d < distance d'influence) {
            d = Calcul_distance_boite_encomb(Sa,Sb) ;
            si(d < distance d'influence) {
                d = Calcul_distance_précise(Sa,Sb) ;
            }
        }
    }


distance_précise(Sa.Sb)
/* calcul general de distance sur deux sous ensembles Sa et Sb */
{
    répéter pour tous les elements Ea de Sa { ;
        répéter pour tous les elements Eb de Sb { ;
            d = Distance_solides(Ea,Eb) ;
        }
    }
```

2.4 Approximation polyédrique des surfaces

2.4.1 Pourquoi transformer les surfaces limitées ?

Méthodes existantes pour les surfaces

La manipulation des surfaces limitées s'avère effectivement difficile à cause du manque de lien entre la définition de la surface de base et la définition des contours.

Le calcul de distances entre surfaces non limitées est réalisé quelquefois en deux étapes.

D'abord un calcul grossier : les surfaces sont chacunes discrétisées en 11x11=121 points. La distance est calculée pour chaque point de l'une des surfaces avec chaque point de l'autre. Parmi les 121^2 couples testés, le meilleur, c'est à dire celui correspondant à la plus petite valeur, ou l'un d'entre eux si plusieurs présentent simultanément cette petite valeur, sert d'initialisation à un calcul plus précis : Ceci permet d'éviter la majorité des cas de minimum relatifs dus à la non convexité des surfaces.

Le calcul plus précis utilise les dérivées premières de chacune des surfaces aux deux points correspondants. On en déduit pour chacun les déplacements à effectuer pour trouver un point sur le plan tangent, qui soit au plus près du point considéré sur l'autre surface. Les points finalement adoptés pour l'itération suivante sont à mi-chemin de ceux trouvés sur le plan tangent, ceci pour éviter les oscillations. Le calcul est réitéré jusqu'à ce que la précision demandée soit atteinte.

Distance entre faces par la méthode du plan tangent

Cette méthode traitant des surfaces non limitées, peut être étendue pour traiter des surfaces restreintes. En ce cas, lorsqu'un bord est franchi (mais il faut être capable de détecter qu'on sort de la face), on se déplace suivant le paramètre du polynôme décrivant le bord, tant que la direction donnée par le plan tangent ne pousse à revenir vers l'intérieur de la face.

Distance entre faces par la discrétisation récursive

Une autre possibilité consiste à discrétiser les deux faces suivant une grille de 11 points sur 11 points, à condition de disposer d'un algorithme de détection de l'appartenance d'un point à une face. Le calcul de distance entre ces points donne un point sur chaque face, autour duquel est fabriqué un nouvel intervalle de discrétisation jusqu'à ses plus proches voisins. Cet intervalle est à son tour discrétisé suivant une grille de 11 points sur 11 points, et ainsi de suite jusqu'à obtention d'une précision suffisante.

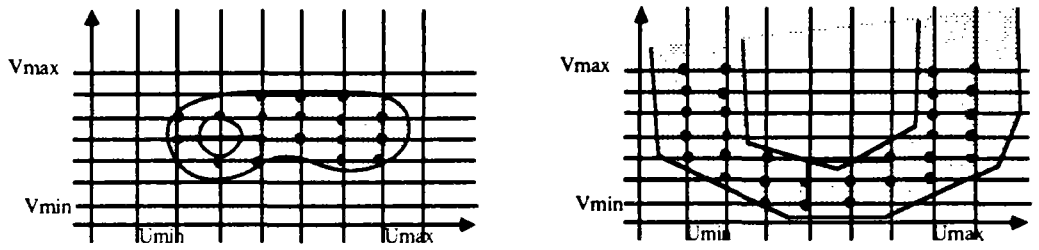


Figure 2.10 : Discrétisation récursive : la discrétisation permet de déterminer un intervalle moindre de recherche de la solution et d'utiliser alors une maille plus fine.

Homogénéïté du modèle

Ce calcul ne peut être facilement adapté au calcul de distances entre solides et surfaces restreintes .

Il est bien entendu que le calcul de distance entre un point et un solide est facilement conçu à partir du calcul de distance entre deux solides. Ce calcul peut alors être appliqué en modifiant simplement un des algorithmes de calcul de distances entre faces, pour obtenir un algorithme de calcul de distance face-solide. Mais le coût de tout ceci est vraiment prohibitif.

C'est pourquoi il est plus raisonnable d'homogénéïser le modèle géométrique de la cellule robotisée en n'utilisant qu'une seule représentation, celle des solides, étendue aux polyèdres (solides non fermés).

Cette opération de polyédrisation, de surcroît, est non seulement un facteur d'homogénéïté, de rapidité de calcul de distance et de simplification des algorithmes, mais également un module susceptible d'être utilisé pour d'autres

applications hors de la robotique, ce qui en fait un module d'usage général. Ainsi est-il utilisé pour mailler des pièces en vue du calcul par éléments finis. Il est utilisé pour visualiser des pièces complexes avec élimination des parties cachées.

Approximation à flèche constante

En un point P de la surface de base, les dérivées premières et secondes par rapport aux paramètres u et v de cette surface sont connues.

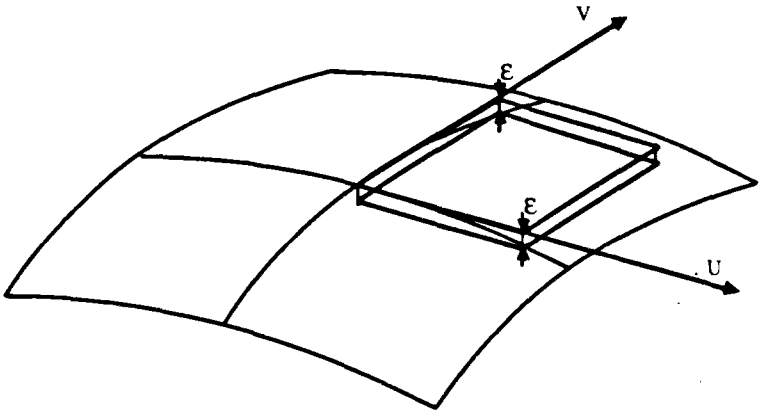


Figure 2.11 : polyèdre défini par une flèche constante par rapport à une surface.

Il est possible de connaître les points disposés sur le plan tangent en P à la surface de base, suivant les axes u et v locaux, tel que ces points soient distants de la surface de base d'une distance ϵ donnée comme l'indique les figures 2.11et 2.12 :

Le rectangle de paramètres u et v est ainsi découpé en rectangles dont les dimensions en u et v sont variables avec la courbure de la surface : une surface quelconque sera décomposée en solides élémentaires en forme de tronc de pyramide ; un cylindre sera découpé selon des pavés parallèles aux génératrices ; une face plane sera découpée en un seul solide élémentaire.

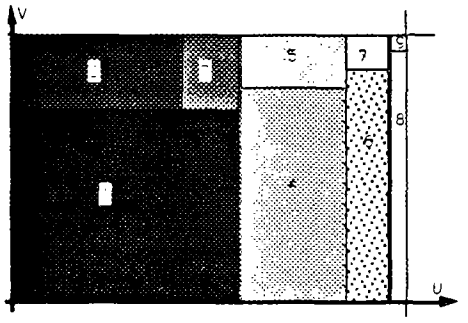


Figure 2.12 : découpage de la surface selon ses deux paramètres. Chaque zone correspond à un polyèdre simple dont la flèche par rapport à la surface est inférieure au maximum imposé.

2.4.2 Approximation par maillage triangulaire spatial des surfaces

Approximation des surfaces simples :

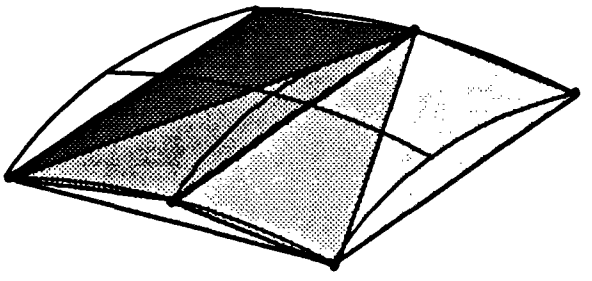


Figure 2.13 : Découpage récursif de la surface en "rectangles" puis en triangles adjacents

Ce principe semble utilisé notamment par certains logiciels de robotique. La surface possède toujours 4 points correspondant aux extrema des paramètres de la surface. Par ces points on fait passer un quadrilatère non plan qu'on découpe en deux triangles, pour garantir la condition de planéïté des facettes. On calcule sommairement la distance de chaque triangle à la surface. Si elle est plus grande que la distance maximale tolérée, on redécoupe le quadrilatère en deux selon la direction la plus favorable, et l'opération est réïtérée jusquà

l'obtention d'un écart suffisament petit par rapport à la surface de base. Cette méthode très simple ne convient pas bien pour traiter les surfaces restreintes. La figure 2.13explique ce fonctionnement.

Approximation des surfaces restreintes :

Cette méthode s'appuie sur les bords de la surface. A ce titre, elle convient particulièrement bien à la polyédrisation des surfaces restreintes.

Elle se classe plutôt parmi les algorithmes de maillage triangulaire en vue de calculs par éléments finis. Une méthode voisine est proposée en deux dimensions par [Lo 87] . Cependant le modèle étudié ici est en dimension trois, car un calcul de triangulation suivant les paramètres de la surface de base ne permet pas de contrôler correctement la taille des triangles et leur forme.

En général, la flèche par rapport à la surface de base n'est pas primordiale pour le maillage "éléments finis" . Cependant un raffinement du maillage est proposé dans un second temps, afin d'intégrer un contrôle réel de la flèche, indispensable pour le calcul de distance et ainsi pour l'étude du contournement des obstacles.

Le principe de base consiste à discrétiser de manière régulière le ou les contours (en cas de trous) de la face, chaque segment du contour étant orienté de manière à ce que l'intérieur de la face se trouve toujours du même côté.

L'algorithme maintient à jour en permanence :

- une liste des segments de contour, orientés,
- une liste des sommets possibles, soit situés sur le contour, soit situés à l'intérieur.

A chaque itération, un triangle est construit à partir d'un des segment du contour et d'un des sommets qui ne sont pas extrémité du segment. La construction du triangle obéit aux règles suivantes :

- Création du triangle du côté de la matière.
- Non-croisement avec le contour (tolérance)
- Troisième sommet le plus proche des deux autres : ceci est réalisé en écrivant que la somme des carrés des distances du troisième sommet aux deux autres doit être la plus faible possible

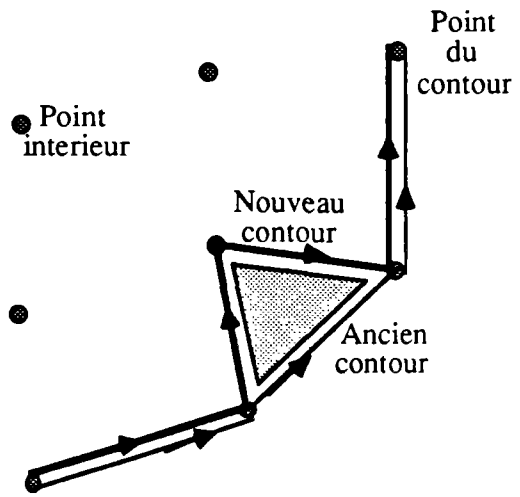


Figure 2.14 : Création d'un nouveau triangle à partir d'un point intérieur puis mise à jour du contour courant au voisinage de celui-ci.

- Triangle peu éloigné de l'isocèle
- Angles aigus ou proches de 60 degrés

le triangle créé, le contour est remis à jour puis un autre segment du contour est traité, comme l'explique la figure 2.14.

L'algorithme est accompli lorsque la liste des segments du contour est vide.

Suivant le choix du contour suivant, on peut observer que la facettisation suit le contour de la face en créant des anneaux de facettes triangulaires en couches successives, ce que montre la figure 2.15.

La triangulation dans l'espace

En principe les algorithmes de traitement des surfaces restreintes supposent que le contour externe tourne dans le sens trigonométrique, les contours interne tournant dans l'autre sens. Dans l'espace, ceci n'a généralement pas de sens.

L'orientation du contour sert à décider si un triangle créé est construit à l'intérieur ou bien à l'extérieur de la face, auquel cas il convient de l'éliminer.

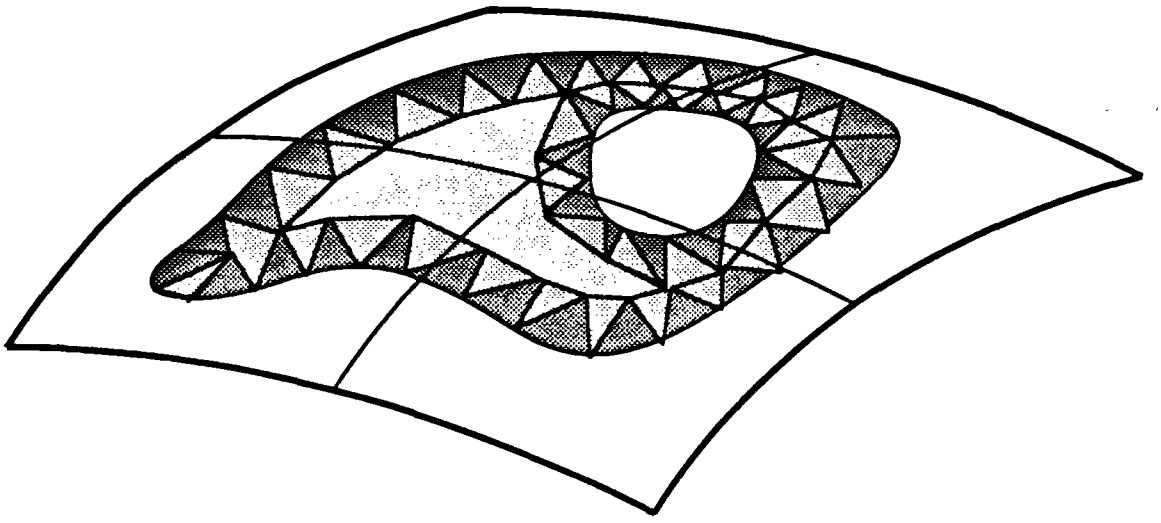


Figure 2.15 : Maillage triangulaire obtenu par cette méthode d'approximation des surfaces restreintes.

Il faut choisir une orientation. Cependant, pour les surfaces très courbes, comme un cylindre par exemple, on voit bien qu'il ne suffit pas de choisir un axe fixe qui déterminerait le côté vers lequel les normales devraient pointer : en effet dans le cas du cylindre, ceci donnerait une moitié de cylindre tournée vers l'extérieur, une autre moitié tournée vers l'intérieur. L'orientation doit ainsi être propagée de proche en proche, être voisine pour deux points voisins sur la surface, sans rien préjuger pour des points éloignés.

Un faisceau de normales

L'algorithme construit ainsi un faisceau de normales à la face, l'ordre de création de celles ci garantissant que deux normales créées successivement sont voisines, ce qui permet de choisir l'orientation de la suivante tournée vers le même demi-espace que celle de la précédente. La construction de ce réseau est réalisée en même temps que la création des points intérieurs, ainsi la discrétisation pour les normales est la même que pour les points intérieurs. Enfin, chaque sommet intérieur se voit affecter une telle normale.

Normales pour les sommets du contour

Le passage de la surface de base aux contours est très pénible : en particulier il est presque impossible, avec les moyens fournis par CATIA, de trouver les

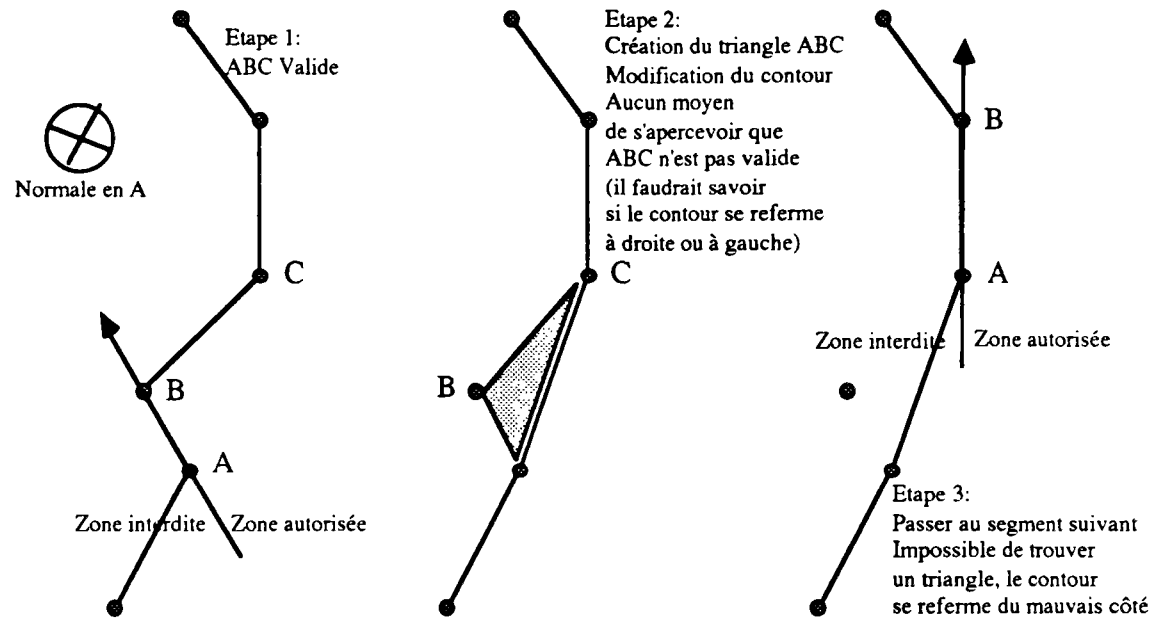


Figure 2.16 : Choix du faisceaux de normales, déterminant le sens de parcours du contour et ainsi les sommets candidats pour créer un nouveau triangle.

paramètres u et v sur la surface de base, pour un point de contour connu uniquement par ses coordonnées cartésiennes.

C'est pourquoi les sommets de contour se voient affecter la normale du faisceau qui correspond au point le plus proche. Certes, ce n'est pas à proprement parler la véritable normale en ce point de contour à la surface de base (qui aurait été difficile à calculer), mais en est une approximation suffisante pour l'usage que l'algorithme en fera, c'est à dire déterminer si un triangle en cours d'étude est du bon côté ou non du segment de contour.

Orientation de la première normale

Il reste à choisir l'orientation de la première normale. Si ce choix est réalisé au hasard, l'algoritme risquera de commencer à créer des triangles du mauvais côté sans le savoir, ce qui est possible s'il existe des sommets potentiels du côté déterminé par la normale choisie et le segment considéré. Au bout d'un certain temps, il deviendra impossible de créer de nouveaux triangles et l'algorithme s'arrêtera en échec. Il est plus efficace de traiter ce problème a priori en vérifiant

qu'il existe des sommets potentiels du côté choisi a priori, et d'inverser tout le faisceau de normales si ce n'est pas le cas pour un des sommets du contour.

Liste des troisièmes sommets potentiels

Au cours de l'évolution de la liste des segments de contour, les sommets initialement intérieurs sont utilisés et deviennent des sommets de contour.

Le contour évolue au cours de l'algorithme. Ainsi, certains points sont quittés par le contour : ils sont éliminés de la liste des sommets potentiels (mais les triangles formés avec eux sont conservés dans la liste des facettes triangulaires) parce qu'ils sont à l'extérieur du contour actif venant d'évoluer, ainsi il ne sera pas possible de construire un autre triangle fondé sur ces sommets.

2.4.3 Algorithme correspondant : fonctionnement, résultats et performances

- Première étape :
  Le programme commence par discrétiser les bords intérieurs et extérieurs de la face. Cette discrétisation est effectuée avec un pas constant suivant l'abscisse curviligne, chaque arc découpé en intervalles égaux, de manière que la longueur curviligne entre deux sommets soit proche de celle demandée par l'utilisateur. Pour les arcs de longeur inférieure à la longueur demandée, les deux sommets d'extrémités sont créés.

- Une phase très importante consiste à réordonner les segments, renuméroter les sommets, éliminer les sommets en double (arcs de longueur nulle par exemple) et les segments inutiles, les segments parcourus deux fois (cas des petits trous par exemple) et à construire ainsi la suite des segments de contour.

  L anécessité de ce retraitement est dûe au fait que l'orientation des arcs est quelconque par rapport à l'orientation du contour, ce qui n'a pas d'importance pour CATIA, mais nuit au bon parcours du contour.

- Deuxième étape :
  Elle consiste à créer des sommets intérieurs, après que le pas de discrétisation ait été déterminé pour la surface de base, en fonction de ses deux paramètres u et v. Si la surface de base est un plan, aucun sommet

intérieur n'est créé, ce qui poussera l'algorithme à fabriquer des triangles grands et étroits.

Ce fait est dû à la définition particulière des plans laquelle est différente de celle des surfaces. Une option possible est de transformer au préalable le plan en surface, ou bien de créer une seule face au contour polygonal comportant éventuellement de nombreux sommets. Ces deux options seront disponibles en fonction des demandes de l'utilisateur.

A l'occasion de la détermination des sommets intérieurs, l'algorithme détermine les normales associées à chacun des sommets.

A ce stade, la liste des segments de contour et la liste des sommets, chacun avec sa "normale", sont disponibles.

- Troisième étape : Des triangles sont créés successivement selon le procédé défini plus haut. Pour chaque triangle créé, le ou les segments de contour qu'il utilise sont remplacés dans la liste par le ou les cotés libres, s'ils existent, de ce triangle. La liste des segments de contour et la liste des sommets potentiels sont ainsi mises à jour. Ceci est réalisé jusqu'à ce que la liste des segments de contour soit vide.

  A ce stade, la liste des sommets et des facettes triangulaires sont disponibles.

- Suit un traitement de relaxation itérative, (assez courant dans le cas du maillage pour les éléments finis). Il est destiné à améliorer l'homogénéité du maillage, en replaçant tous les sommets qui ne sont pas sur le contour initial au barycentre des sommets reliés à lui par une arête, puis à projeter le point obtenu sur la surface de base.

- Il reste à mettre la liste des sommets et des facettes triangulaires sous la forme standard des polyèdres CATIA.

Objectifs du programme réalisé

Le programme, dénommé "Facettes", a pour but de créer des entités Polyèdres (*POL de CATIA) à partir des entités surfaces restreintes (*FAC de CATIA).

Il traite soit une seule face au choix de l'utilisateur, soit toutes les faces visibles du modèle.

Le programme est entièrement homogène avec CATIA, c'est à dire qu'il ne crée aucune donnée cachée ou incompatible, aucune entité spécifique ni aucun sous-ensemble spécial (application set).

Utilisation

précision : Le réglage de la précision du maillage triangulaire est réalisé en demandant à l'utilisateur l'ordre de grandeur souhaité pour la longueur des arêtes du polyèdre résultant. Une précision de 50 mm signifie que chaque arête du polyèdre aura 50 mm environ de longueur.

Remarque 5 *Ceci n'est qu'un ordre de grandeur, car le programme présuppose que le créateur des faces a découpé les contours en arcs de manière logique. Le programme propose un sommet à chaque extrémité des arcs : si un arc est court, il suppose qu'il y avait une bonne raison, comme par exemple une courbure très serrée, un point de rebroussement, un angle...*

Limite d'utilisation

Certaines faces ne peuvent actuellement être traitées par le programme, qui répond alors "facettisation impossible". Les raisons en sont principalement les suivantes :

- La face est trop petite par rapport à la taille souhaitée pour les facettes du polyèdre, et il a été impossible de trouver au moins trois sommets sur le contour.
- La face possède un contour intérieur mal orienté, c'est à dire qui tourne dans le même sens que le contour externe courant, au lieu de tourner dans le sens contraire. Ceci est autorisé par CATIA, qui n'utilise pas cette particularité. L'ordre de parcours du contour interne peut seulement être modifié en redéfinissant la face, et en sélectionnant les courbes de bord dans le bon ordre.
- La face possède une forme particulièrement tourmentée, par rapport à la taille souhaitée pour les facettes, le programme ayant alors du mal à détecter l'intérieur et l'extérieur des contours. La seule solution consiste à demander une précision plus serrée.

Informations à fournir à l'ordinateur et réponse de celui-ci

Dans le cas où est choisie l'option "une seule face", l'utilisateur sélectionne à l'écran la face à traiter.

Puis il doit modifier éventuellement au clavier l'ordre de grandeur de la longueur des côtés des facettes, la valeur 20 mm lui étant proposée par défaut.

L'ordinateur crée en base de données CATIA correspondant au modèle à l'écran, un polyèdre (entité de type 16, *POL), et rend invisible la face correspondante (NO SHOW) afin de ne pas surcharger la visualisation du modèle à l'écran.

En cas d'impossibilité, il crée une liste ordonnée de points (*CST, contrainte CATIA) et de lignes visualisant les triangles que le programme avait essayé de créer jusqu'au moment où s'est produit une impossibilité. Ceci sert à l'opérateur pour déterminer la cause d'erreur et éventuellement décider d'une action adéquate. Dans ce cas, la face clignote pour un repérage plus facile.

Organisation des programmes

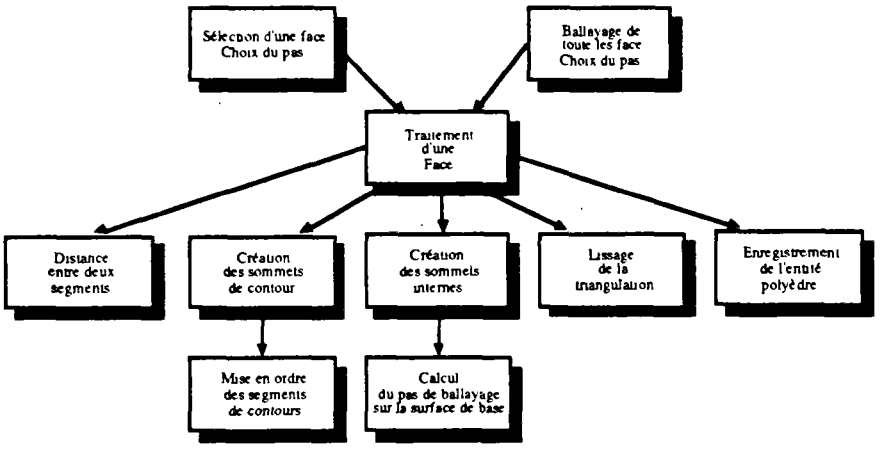


Figure 2.17 : Organisation des sous-programmes permettant la facettisation des surfaces restreintes, en trois dimensions.

Utilisation de la fonction et résultats

C'est une fonction GII, c'est à dire que son utilisation et sa présentation sont identiques à celles des fonctions CATIA intégrées.

Menu : La fonction est disponible avec l'article "Facettes" du menu général. Le sous-menu spécifique donne le choix entre :

- "Facettes"+ "Une" : polyédrisation d'une seule face
- "Facettes"+ "Toutes" : Facettisation de tout le modèle

Un fichier de résultats est créé, qui contient toutes les informations sur le polyèdre : coordonnées de chaque sommet, liste des triangles créés, segments situés sur les contours. Ce fichier peut ainsi être utilisé pour transmettre les informations à un programme de calcul par éléments finis.

Résultat

La fonction ainsi décrite est très robuste, elle répond convenablement aux cas singuliers ; le traitement d'une face est suffisamment rapide pour ne pas gêner l'interactivité pour l'opérateur puisque le résultat est obtenu en quelques secondes. En revanche, le traitement d'une caisse de voiture complète est long (plus d'une heure), mais le nombre de surfaces restreintes traitées est très grand, puisqu'il est couramment supérieur à 1000 !

la figure 2.18 illustre la transformation en facettes triangulaires pour un côté de caisse, en vue de la réalisation d'une trajectoire sans collisions pour un robot de soudure chargé de réaliser les points de soudure autour des ouvertures de portes. Le modèle est créé directement à partir du modèle surfacique (surfaces restreintes) utilisé pour toutes les études de conception et de fabrication du véhicule.

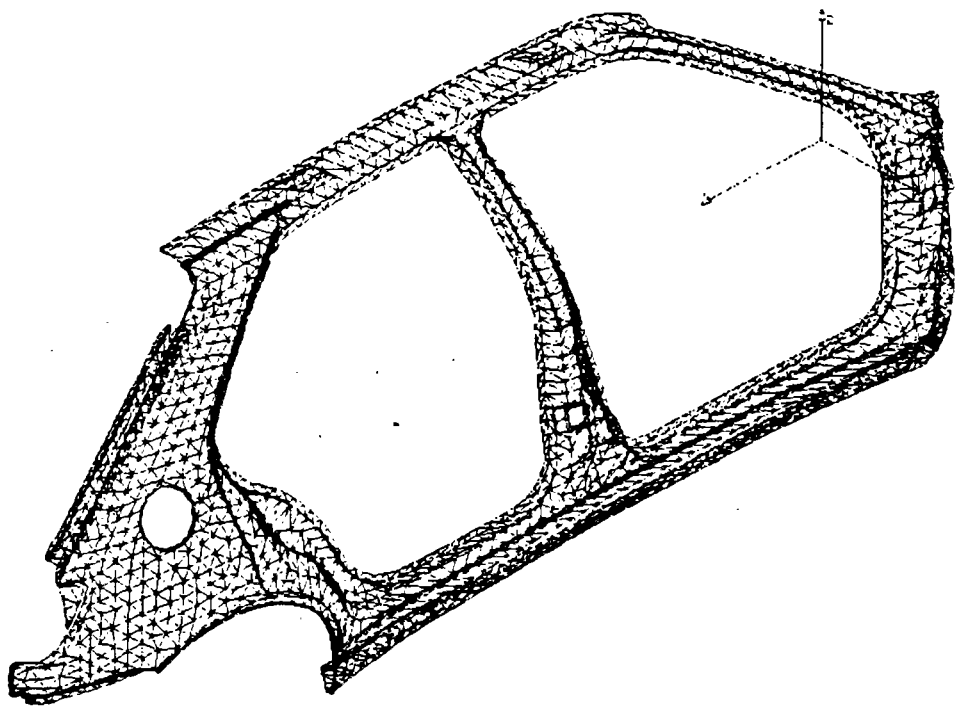


Figure 2.18 : Résultat : facettisation entièrement automatique d'un côté de caisse 205 Peugeot (modèle CATIA). Ce modèle sera utilisé ensuite pour le contournement d'obstacles, lors de la construction d'une trajectoire sûre réalisant des points de soudure sur cette pièce.

Chapitre 3

La génération de trajectoires sans collisions pour les robots manipulateurs industriels

La démarche vers l'étude d'implantation optimale sans collisions pour les robots, après la prise en compte des effets induits par des recherches menées au sein d'un système de CAO, exige d'envisager convenablement le domaine des trajectoires sans collisions.

Ce chapitre analyse les problèmes de la génération de trajectoires sans collisions, évoque les solutions actuellement proposées par les chercheurs, et présente la méthode des contraintes proposée par B. Faverjon. Nous proposons une adaptation de cette méthode au système de CAO Catia et à sa base de données. nous proposons en outre des solutions ponctuelles pour sortir des situations de blocages peu sévères. La programmation de cette méthode que nous avons effectuée donne satisfaction en usage industriel : nous présentons un exemple de ce type pour en expliciter les performances concrètes.

3.1 Le contournement des obstacles

Le concept de "collision" exige la résolution de problèmes très différents [PS85] :

- La simple *détection* de collision, test de contrôle : oui ou non y a t-il intersection entre les entités ?
- Le *calcul d'intersection* entre solides ou surfaces pouvant conduire à la construction d'une nouvelle entité [PS85]
- Le calcul de *distance* entre deux entités dans le cas où il n'y a pas collision, avec détermination sur chaque entité des points les plus proches (Tournassoud, Gilbert) [GJK88] [Tou88]
- Le calcul des *paramètres-clefs* de la collision dans le cas où elle intervient : distance de pénétration, direction de pénétration,... (CATIA) [Das89]

Dans l'étude de ces problèmes la quantité d'information est très onéreuse, notamment en temps de calculs. Mais pour contourner efficacement les obstacles il est extrêmement utile, et même nécessaire, de disposer d'un champ d'informations aussi pertinent que possible.

Il y a ainsi presque toujours à apprécier subjectivement un optimum entre la complexité d'élaboration d'un résultat assez sûr, assez complet et la rapidité d'obtention d'un résultat plus approximatif.

Ce genre de calcul pourrait dans certains cas particuliers, déjà exister dans CATIA en interactif, mais pour l'instant on ne peut pas utiliser ces sous-programmes pour réaliser un programme nouveau.

Le choix de la méthode est un facteur important puisqu'en dépend la vitesse des calculs de collisions, calculs qui seront utilisés ensuite de manière intensive et systématique.

La quasi-totalité des algorithmes existants, s'il sont souvent performants, supposent des solides convexes. Ceux qui ne présupposent pas ce genre de limites sont peu publiés car leur généralité se paie d'une complexité épouvantable ralentissant les calculs. Mais peut-on raisonnablement travailler avec de telles limitations ?

3.1.1 Méthodes globales ou locales

Aucune n'est sans inconvénient à mettre en balance avec ses avantages, car dans les applications industrielles, les simplifications des problèmes qui "se payent" d'un risque de ne pas trouver de solution bien qu'il en existe, sont difficiles à admettre.

Les méthodes globales de type cellulaire sont fiables et faciles à utiliser, mais peu précises avec des durées d'exécution du programme très longues, ce qui implique une utilisation lourde et peu pratique.

Les méthodes locales comme la méthode des potentiels sont intéressantes, malgré le risque d'obliger l'opérateur à intervenir si les dispositions conduisent à des blocages.

Les méthodes locales sont vraiment souples et rapides. Leur utilisation est presque interactive. Il leur manque essentiellement un opérateur ou un planificateur global pour les guider en cas de problème.

3.1.2 Approche mixte avec apprentissage

Pour une reconnaissance préventive ou apprentissage, une méthode est proposée par Bernard Faverjon et Pierre Tournassoud [FT87b] [Tou88].

Toute méthode permettant de fournir par apprentissage les informations générales nécessaires au planificateur global est d'un très grand intérêt car ceci présente l'avantage de ne décrire que l'espace réellement utile.

Cependant un grand nombre de chemins doivent être essayés. Pour ce faire, on peut imaginer de donner à l'avance une valeur a priori à toutes les cellules du graphe. Cette valeur doit décrire aussi bien que possible les probabilités d'échec ou de réussite à passer dans les cellules voisines, par exemple fonction du volume occupé par les obstacles par rapport au volume total de la cellule. Ceci devrait améliorer la rapidité d'obtention du chemin optimal dans les cas où l'environnement est très compliqué et où l'apprentissage aurait à chercher par trop d'essais répétés ce même chemin.

Dans les cas difficiles la seule utilisation d'une méthode locale a un très grand risque d'échec, mais, comme l'environnement est complexe, souvent exigu,

toute méthode globale du type cellulaire doit avoir une maille de cellules très fine pour espérer aboutir à un cheminement possible. Dans ces conditions il devient intéressant de faire coopérer une méthode locale gérant les obstacles "à vue" avec une méthode globale laquelle gère l'ensemble du cheminement par une succession de buts intermédiaires vers le but final. En ce cas le pas de la maille des cellules peut ne pas être très petit, il peut rester à l'échelle des "détails significatifs" des organes et des obstacles concernés, la méthode locale gérant les collisions à l'intérieur d'une telle cellule de la maille générale. Cette approche n'est pas sensiblement différente de ce que serait une approche humaine naturelle laquelle se préoccuperait d'abord des grandes lignes possibles et des décisions de principe puis testerait les points successifs en les regardant à la loupe.

L'utilisation industrielle de telles méthodes doit conduire à un résultat s'il existe, car on voit mal une simulation, qui en principe devrait proposer des solutions difficiles à obtenir intuitivement, répondre trop souvent qu'elle ne sait pas résoudre le problème alors que l'opérateur a le sentiment d'entrevoir une solution intuitivement ou visuellement.

A cet égard, il faut une méthode globale. Mais, pour obtenir un temps de calcul raisonnable, il est souhaitable de faire coopérer une méthode globale avec telle méthode locale qui par son attraction vers l'objectif permette des trajectoires plus tendues.

3.2 Une méthode locale : la méthode des contraintes

3.2.1 Principes de base

Objectifs, fonctions à minimiser

Il faut une méthode qui permette d'approcher des obstacles jusqu'au contact. La trajectoire obtenue aura à se rapprocher le plus possible des trajectoires simples et standard des robots. De cette manière, la trajectoire pourra être simplifiée en utilisant les possibilités du robot, sans crainte d'écarts trop importants entre la trajectoire espérée et la trajectoire réellement réalisée par le robot.

La méthode présentée ici s'appuie sur la méthode des contraintes, dont les principes ont été présentés par Faverjon et Tournassoud [FT87a], [Tou88], [FT88] et [ABF88].

Elle est voisine de la méthode des potentiels déjà bien connue [Kha86].

Le principe de base de la méthode des contraintes consiste à définir des configurations successives du robot depuis la configuration d'origine. On calcule chaque nouvelle configuration à partir de la précédente, en minimisant une fonction $\mathcal{U}$ (comme un potentiel) qui doit rapprocher le robot de sa configuration finale. La fonction $\mathcal{U}$ est construite à partir du but à atteindre et du type de trajectoire désirée en l'absence d'obstacles. Chaque couple de corps (fixes ou en mouvements) en interaction, c'est à dire ceux pour lesquels il est indispensable de faire attention aux collisions éventuelles, est étudié dans l'espace ambiant, puis traduit en une contrainte dans l'espace des configurations, pour la minimisation de cette fonction ressemblant à un potentiel.

Dans ce qui suit, pour faciliter la représentation, les exemples sont étudiés pour un robot à deux degrés de liberté.

Adaptation de la définition de la tâche aux caractéristiques de la méthode.

Dans l'espace ambiant, la tâche consiste, à partir d'une position et d'une orientation courante du robot, à amener le repère actif de l'outil du robot en coïncidence avec un repère de l'espace.

Dans l'espace des configurations du robot, la tâche est définie par une configuration d'origine Q_0 et une configuration finale Q_f.

Dans ce cas, le problème est contrôlé par $\overrightarrow{QQ_f}$, tel que $\overrightarrow{QQ_f} = 0$ lorsque le but recherché est atteint.

La méthode des potentiels utilise ordinairement un potentiel attractif classique proportionnel au carré de la distance au but :

$$\mathcal{U}(Q) = \tfrac{1}{2}\|\overrightarrow{QQ_f}\|^2$$

Ceci n'est pas satisfaisant car une minimisation de ce potentiel va faire évoluer davantage les Q^i pour lesquels l'écart entre la configuration courante et la configuration du but est le plus grand : en effet, voulant faire décroître $\mathcal{U}(Q)$ le plus efficacement possible, l'action est portée sur le terme de plus fort poids, c'est à dire sur la composante la plus éloignée du résultat souhaité. Une conséquence sera une trajectoire s'écartant excessivement de la linéarité souhaitée, tant dans l'espace ambiant que dans l'espace des configurations.

3.2.2 En l'absence d'obstacles : type de trajectoire, buts intermédiaires et contraintes

Choix du type de trajectoire

Trajectoire linéaire dans l'espace des configurations

Comme tous les robots possèdent un mode de construction de trajectoire par interpolation "articulaire", le robot suit aisément une droite dans l'espace des configurations. Ainsi, pour la construction de la trajectoire en l'absence d'obstacles, le robot suivrait-il une droite dans l'espace des configurations, depuis la position courante du robot jusqu'au but.

Ceci se traduit par :

$$\frac{\overrightarrow{dQ}}{dt} = k_b \overrightarrow{QQ_f}$$

le coefficient k_b déterminant un but intermédiaire à atteindre, d'où la fonction à minimiser à chaque étape :

$$\mathcal{U}(Q) = ||\frac{\overrightarrow{dQ}}{dt} - k_b \overrightarrow{QQ_f}||^2$$

Puisque la trajectoire sera discrétisée pour des intervalles de temps Δt du mouvement selon cette trrajetoire, il s'agira de minimiser :

$$\mathcal{U}(Q) = ||\frac{\overrightarrow{\Delta Q}}{\Delta t} - k_b \overrightarrow{QQ_f}||^2$$

D'autres types de trajectoires peuvent être modélisés de façon analogue, en particulier des trajectoires linéaires dans l'espace ambiant ou bien d'autres trajectoires encore : splines,cercles,...

Choix du sous-but

Le choix du coefficient k_b (de dimension $\frac{1}{T}$) est très important, car il détermine la position du minimum de la fonction $\mathcal{U}$.

Dans l'espace des configurations et en l'absence d'obstacles, le minimum de cette fonction est obtenu pour $\mathcal{U}(Q) = 0$. Il se situe sur la droite (Q, Q_f).

$0 < k_b \leq 1$ si on veut que le minimum de potentiel se trouve sur le segment $]Q.Q_f]$.

Le minimum de potentiel est atteint pour une configuration Q_b telle que :

pour : $\overrightarrow{\Delta Q} = \overrightarrow{QQ_b}$

$$\frac{1}{\Delta t}\overrightarrow{QQ_b} = k_b \overrightarrow{QQ_f}$$

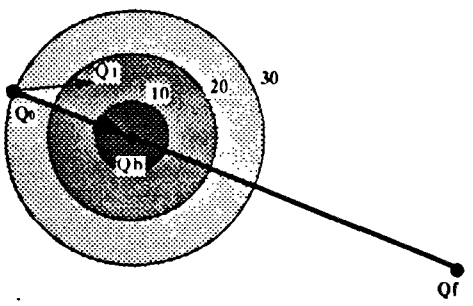


Figure 3.1 : Valeurs de $\mathcal{U}(Q)$ autour de Q_b, défini comme le minimum à atteindre et situé par construction sur la droite (Q_0, Q_f).

Choix de Q_b :

La nouvelle solution Q_1 est calculée avec un pas en temps de Δt, le même que celui définissant le champ $\mathcal{U}(Q)$.

Pendant le temps Δt, la représentation du robot dans l'espace des configurations, ne peut s'être déplacée, pour chaque variable articulaire, de plus de :

$$\Delta Q^i = V^i_{max}\, \Delta t$$

La solution doit ainsi se trouver à l'intérieur d'un "rectangle" centré en Q ayant pour côtés les composantes de $\overrightarrow{V_{max}}$. Si la configuration Q_b est en dehors de ce rectangle, la minimisation va buter sur ses limites, le minimum de $\mathcal{U}(Q)$ pourra alors se trouver ailleurs que sur la droite $(Q; Q_f)$, même en l'absence d'obstacles.

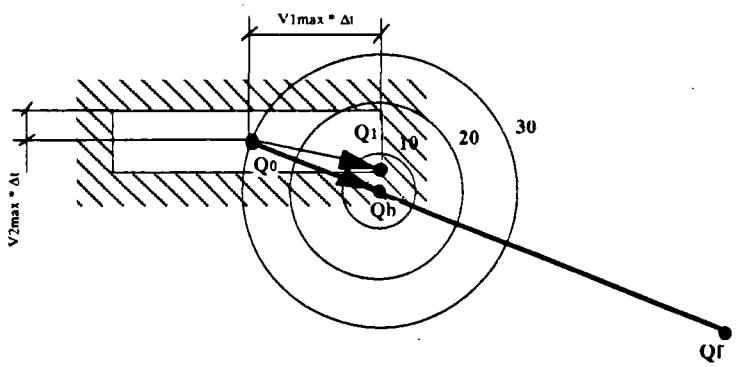


Figure 3.2 : Limites de déplacement du robot au bout d'un temps Δt si Q_b est en dehos de ces limites, le minimum atteint peut ne pas être sur la droite (Q_0, Q_f), même en l'absence d'obstacles.

Un bon choix pourrait être l'intersection de la droite $(Q_0; Q_f)$ avec le "rectangle" $\overrightarrow{V_{max}}\Delta t$ autour de la configuration Q_0, ceci quand Q_f n'est pas dans ce rectangle. Sinon il faut impérativement $Q_b = Q_f$ pour éviter d'avancer au-delà de la configuration finale Q_f.

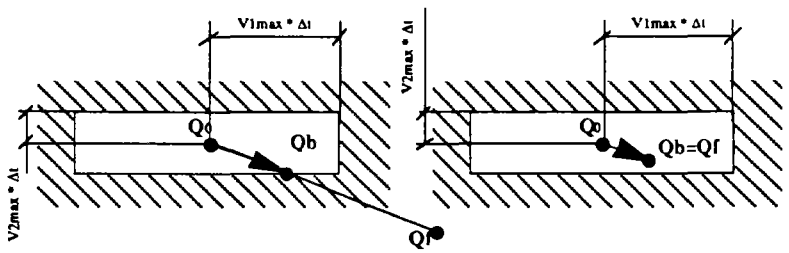


Figure 3.3 : Détermination de Q_b suivant que Q_f est accessible pendant la durée Δt ou non.

Contraintes de vitesses

Notation : lorsque chaque composante d'un vecteur $\overrightarrow{U}$ est inférieure ou égale à chaque composante d'un vecteur $\overrightarrow{V}$, cette relation sera notée :

$$\overrightarrow{U} \leq \overrightarrow{V}$$

Les limites de vitesse sont déjà intervenues pour définir la fonction à optimiser et ainsi garantir qu'une configuration visée est correctement placée.

Ainsi en l'absence d'obstacles, le robot ira-t-il le plus loin possible vers ce but, aussi loin que le permettent les limites de vitesse pour l'intervalle de temps Δt fixé. Mais en présence d'obstacles, les déviations engendrées peuvent pousser le robot à prendre une configuration différente de Q_b, le minimum de la fonction $\mathcal{U}(Q)$ (supérieur à la valeur en Q_b) pouvant se situer alors à l'intérieur du domaine limité par les vitesses maximales autorisées.

Chaque axe du robot ne peut dépasser une vitesse maximale :

$$-\overrightarrow{V_{max}} \leq \frac{d\overrightarrow{Q}}{dt} \leq \overrightarrow{V_{max}}$$

c'est à dire

$$-\overrightarrow{V_{max}}\Delta t \leq \overrightarrow{QQ_0} \leq \overrightarrow{V_{max}}\Delta t$$
$$Q_0 - \overrightarrow{V_{max}}\Delta t \leq Q \leq Q_0 + \overrightarrow{V_{max}}\Delta t$$

Pour la considération complète de ces limites, il faut prendre en compte les configurations correspondant aux butées absolues de chaque articulation :

$$Q \in [Q_{min}, Q_{max}]$$

d'où :

$$Max(Q_{min}; Q_0 - \overrightarrow{V_{max}}\Delta t) \leq Q \leq Min(Q_{max}; Q_0 + \overrightarrow{V_{max}}\Delta t)$$

Articulations complexes du robot

Lorsque le robot possède des articulations complexes (ni rotation pure, ni translation pure), les variables articulaires du robot sont couplées. Ceci se traduit par des inéquations dans l'espace des configurations, qui limitent le domaine de validité des variables articulaires.

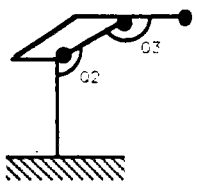


Figure 3.4 : Articulation de type parallélogramme

Dans le cas d'une articulation de type parrallélogramme, qui est un des cas les plus courants, le couplage est exprimé sous la forme :

$$Q^2 + Q^3 \in [a_1, a_2]$$

3.2.3 En présence d'obstacles

3.2.3.1 Précisions et définitions

Il est opportun d'étudier d'abord le comportement du robot en présence d'obstacles.

D = distance entre les deux corps $\mathcal{C}_1$ et $\mathcal{C}_2$.

Notation :
Distance : La distance employée ici est la distance euclidienne. Pour un couple d'objets $S_1; S_2$), le point x_1 est le point appartenant à S_1 qui est le plus proche de S_2, le point x_2 est le point appartenant à S_2 qui est le plus proche de S_1. La distance est ainsi $D = ||\overrightarrow{x_1x_2}||$ et $\overrightarrow{N} = \frac{\overrightarrow{x_1x_2}}{||\overrightarrow{x_1x_2}||}$, c'est à dire un vecteur unitaire colinéaire à $\overrightarrow{x_1x_2}$.

Définition 22 *Distance d'influence : L'interaction entre deux corps est à prendre en compte si la distance entre eux D est inférieure à une distance d'influence D_I.*

Définition 23 *Distance de sécurité : On interdit à deux corps de se rapprocher si près que leur distance deviendrait inférieure au double de la distance de sécurité D_S.*

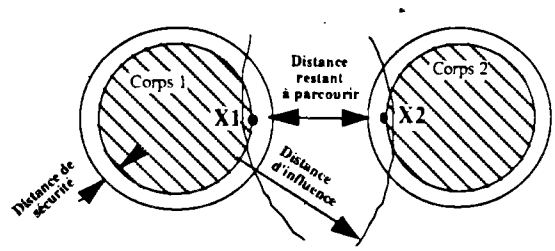


Figure 3.5 : Interaction entre deux corps

Pendant le temps Δt, il faut que D ne diminue pas exagérément : pas plus que de $D - D_S$ distance admissible restant à parcourir :

$$\Delta D \geq -K_d\,(D - D_S)$$
($K_d \in [0; 1]$ est un coefficient sans dimension)

Choix de K_d :

K_d diminue avec la répulsivité de l'obstacle : plus K_d est grand, plus il est facile de se rapprocher de l'obstacle car l'inégalité est plus facile à réaliser.

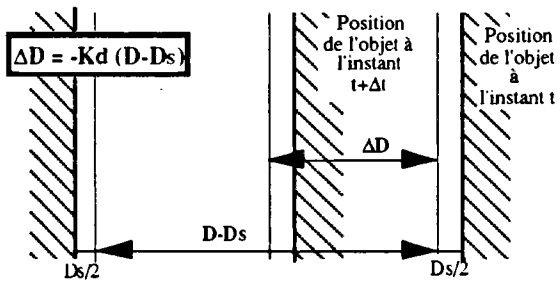


Figure 3.6 : Rapprochement maximal ΔD au bout d'un temps Δt

A chaque pas Δt, les objets se rapprochent au plus de $\Delta D = K_d(D - D_S)$. Ainsi, pour $K_d = 1$, les deux objets peuvent venir en contact en un seul pas. Il faut au moins $\frac{1}{K_d}$ petits déplacements successifs pour que les objets arrivent au contact l'un de l'autre.

Si $K_d = 0$, dès que les objets sont en influence, ils ne peuvent plus se rapprocher.

Remarque 6 *Choix de la distance de sécurité D_S :*

La distance de sécurité est initialement prévue pour se prémunir contre :

- *Les erreurs de modèle dues à l'approximation dans la modélisation des objets (facettisation, simplifications,...).*
- *Les erreurs de contrôle dues aux imprécisions dans l'exécution de la trajectoire.*

De fait, comme les objets engendrent dans l'espace des configurations des contraintes linéarisées, les mouvements des objets étant linéarisés par les matrices jacobiennes, on choisit une marge de sécurité en fonction de Δt et des vitesses maximales, de manière à se prémunir contre les erreurs dues à la linéarisation. Plus Δt est grand, plus les vitesses maximales des articulations du robot sont rapides, plus il faut prendre une marge de sécurité grande. Cette marge de sécurité est à considérer soit par l'intermédiaire de K_d, soit par celui de la distance de sécurité D_S.

Le choix de K_d dépend essentiellement de la façon dont on veut longer les obstacles :

A K_d grand (voisin de 1), le robot se rapprochera vite très près de l'obstacle (à peine plus loin que la distance de sécurité D_S), puis ensuite le longera à la distance D_S pour l'éviter.

A K_d petit (voisin de 0), le robot commencera à contourner l'obstacle d'assez loin, et se rapprochera lentement de l'obstacle (jusqu'à la distance D_S).

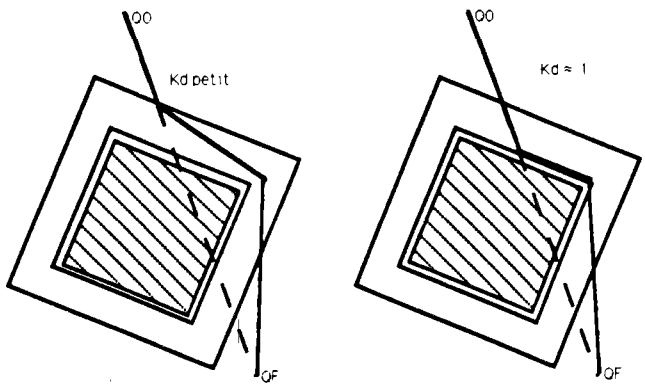


Figure 3.7 : Influence de K_d : sa valeur permet de régler la façon dont le robot se rapprochera des obstacles.

3.2.3.2 Gestion des obstacles dans l'espace des configurations

Comme (figure 8.4) :

$$D = \overrightarrow{x_1x_2}.\overrightarrow{N}$$
$$dD = d(\overrightarrow{x_1x_2}.\overrightarrow{N})$$
$$dD = d(\overrightarrow{x_1x_2}).\overrightarrow{N} + \overrightarrow{x_1x_2}.d(\overrightarrow{N})$$

Comme $\overrightarrow{N}$ est normé, $d(\overrightarrow{N})$ est perpendiculaire à $\overrightarrow{N}$ donc perpendiculaire à $\overrightarrow{x_1x_2}$:

$$\overrightarrow{x_1x_2}.d(\overrightarrow{N}) = 0$$

$$dD = d(\overrightarrow{x_1x_2}).\overrightarrow{N}$$
$$dD = d(\overrightarrow{ox_2} - \overrightarrow{ox_1}).\overrightarrow{N}$$
$$dD = (d(\overrightarrow{ox_2}) - d(\overrightarrow{ox_1})).\overrightarrow{N}$$

Si J_i est la matrice Jacobienne correspondant aux mouvements du corps C_i, nous avons pour un point solidaire de ce corps :

$$\overrightarrow{dx_i} = J_i\overrightarrow{dQ_i}$$

Remarque 7 *Pour le point réalisant la distance minimale sur chaque solide, les petits mouvements sont composés*

- *du mouvement du solide sur lequel il se trouve,*
- *du déplacement de ce point relativement à son solide, à la surface de celui-ci :* $\overrightarrow{dx_S}$

$$\overrightarrow{dx_i} = J_i\overrightarrow{dQ_i} + \overrightarrow{dx_S}$$

Le point de distance minimale sur chaque corps évolue sur le corps. Pour ses petits déplacements,**sur les solides strictement convexes** , il évolue dans le plan tangent en ce point sur le solide, ce plan étant prependiculaire à la direction de distance minimale. Par conséquent, la vitesse relative du point de distance minimale par rapport à son solide support n'intervient plus dans la dérivée première de la distance car elle est perpendiculaire à $\overrightarrow{N}$.

Dans notre cas de **solides facettisés**, le point de distance minimale sur l'un des solides peut se trouver soit à l'intérieur d'une des facettes, soit sur une arête (extrémités exclues), soit en un sommet.

S'il est sur un **sommet**, toutes les arêtes et facettes passant par ce sommet sont dans le demi-plan perpendiculaire au vecteur unitaire portant la distance $\overrightarrow{N}$ et

passant par le sommet considéré, du côté opposé à ce vecteur. Par conséquent $\overrightarrow{dx_S} = 0$ et le point de distance minimale reste sur le même sommet.

S'il est sur une **arête**, celle-ci est perpendiculaire à $\overrightarrow{N}$. Les petits mouvements vont faire évoluer le point sur l'arête ce qui assure que $\overrightarrow{dx_S}$ est bien perpendiculaire à $\overrightarrow{N}$ et qu'il n'interviendra pas dans le calcul de dD .

S'il est sur une **facette**, celle-ci est perpendiculaire à $\overrightarrow{N}$. Les petits mouvements vont faire évoluer le point sur le plan de la facette, le point et $\overrightarrow{N}$ évoluant, $\overrightarrow{N}$ restant normal à la facette. Ainsi $\overrightarrow{dx_{iS}}$ évoluant perpendiculairement à $\overrightarrow{N}$, il n'interviendra pas non plus dans le calcul de dD .

et ainsi :

$$dD = (J_2\overrightarrow{dQ_2} - J_1\overrightarrow{dQ_1}).\overrightarrow{N}$$

Les $\overrightarrow{dQ_i}$ représentent les petits mouvements articulaires des mécanismes auquels sont liés les corps $\mathcal{C}_i$. On peut rassembler tous les degrés de liberté du modèle sous une seule configuration Q.

Si tous les paramètres de mouvements (axes de déplacements pouvant être commandés) sont regroupés, ou plus simplement s'il n'y a qu'un seul robot, les $\overrightarrow{dQ_1}$ et $\overrightarrow{dQ_2}$ correspondent à des corps distincts du **même** robot, du même point dans l'espace des configurations. Puisque la matrice Jacobienne dépend du point considéré sur le corps du robot (dans l'espace ambiant), ceci devient :

$$\begin{aligned} dD &= (J_2 - J_1)\overrightarrow{dQ} \,.\, \overrightarrow{N} \\ dD &= J\overrightarrow{dQ} \,.\, \overrightarrow{N} \\ dD &= J^t\overrightarrow{N} \,.\, \overrightarrow{dQ} \end{aligned}$$

d'où

$$\overrightarrow{gradD} = J^t\overrightarrow{N}$$

Ceci conduit à exprimer la contrainte dans l'espace des configurations :

$$\begin{aligned}
\overrightarrow{gradD} \,.\, \frac{\overrightarrow{dQ}}{dt} &\geq -K_d \frac{D - D_S}{\Delta t} \\
\overrightarrow{gradD} \,.\, \overrightarrow{\Delta Q} &\geq -K_d(D - D_S) \\
\overrightarrow{gradD} \,.\, \overrightarrow{\Delta Q} + K_d(D - D_S) &\geq 0
\end{aligned}$$

Ceci est une inéquation linéaire en $\overrightarrow{\Delta Q}$ avec terme constant.

Il y a autant d'inéquations que de couples de solides en interaction, qu'il s'agisse d'un solide lié au robot interagissant avec un solide de l'environnement ou de deux solides du même robot.

Cela est graphiquement représenté dans l'espace des paramètres, par des droites, qui si $D \neq D_S$, ne passent pas par la configuration Q_0 :

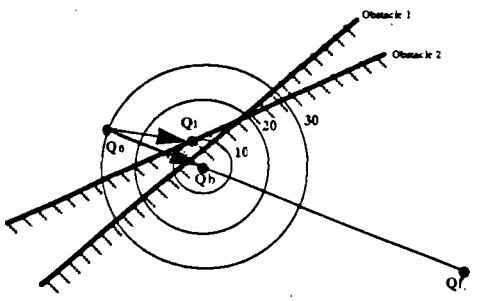


Figure 3.8 : Influence des obstacles suivant la position de Q_b : si Q_b est près de Q_0.

Ainsi ne suit-on pas la même trajectoire suivant que Q_b est éloigné ou non de Q_0 : plus il est éloigné, plus on s'écarte des obstacles, mieux on utilise les possibilités de vitesses maximales du robot, mais les contraintes d'obstacles sont de plus en plus écartées de la réalité puisque linéarisées autour de Q_0.

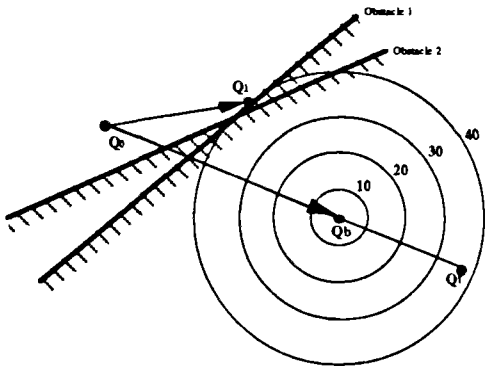


Figure 3.9 : Influence des obstacles suivant la position de Q_b : si Q_b est loin de Q_0.

.

3.3 Module réalisé pour l'application de la méthode des contraintes

3.3.1 Structure de l'algorithme

Nous avons réalisé cet algorithme :

```
construire_tache_sure(Tache,robot,environnement)
/* realisation d'une tache sans collisions */
{
    répéter pour toutes les configurations C_i de la tache ;
        depart = initialisation(C_i) ;
        arrivee = initialisation(C_i+1) ;
        construire_trajectoire_sure (depart,arrivee,robot,environnement) ;
    }
}


construire_trajectoire_sure(depart,arrivee,robot,environnement)
/* realisation d'une trajectoire sans collisions entre deux configurations*/
{
    configuration_courante= depart ;
    tant que (arrivee non atteinte) faire {
        deplacement_corps = modele_direct(robot,configuration_courante) ;
/* "interaction" : enregistrement des donnees de chaque couple d'objets a une distance suffisamment petite l'un
de l'autre */
/* "jacob" : enregistrement dela jacobienne relative aux points de distance minimale pour chaque interaction
*/
        Interactions = distances_modele(environnement,robot,deplacement_corps) ;
            jacob = calcul_jacobienne(interaction,robot,Configuration_courante) ;
        contraintes_intervalles = contraintes_intervalles(V_max,butées) ;
        contraintes_lineaires = contraintes_lineaires(V_max,configuration_courante, interactions, jacob) ;
        but_intermediaire = construire_but(V_max,configuration_courante,arrivee,Δt) ;
        configuration = minimise(potentiel,contraintes_intervalles,contraintes_lineaires) ;
        deplacer (robot,configuration) ;
        configuration_courante= configuration ;
    }
}


construire_but(V_max,configuration_courante, arrivee,Δt)
/* construction du minimum absolu du potentiel */
{
    proche = Calculer_acessibilite(Δt, (V_max_robot,arrivee) ;
    /* proche est vrai si on peut atteindre arrivee */
    si(proche = vrai) {
        but = arrivee ;
    sinon
        pour chaque articulation du robot faire {
            coefficient = minimum(Dt*V'_max/ (arrivee - configuration_courante) ),
        }
        construire_but = configuration_courante + coefficient * (arrivee-configuration_courante ) ;
    } ;
```

3.3.2 Résultats du programme

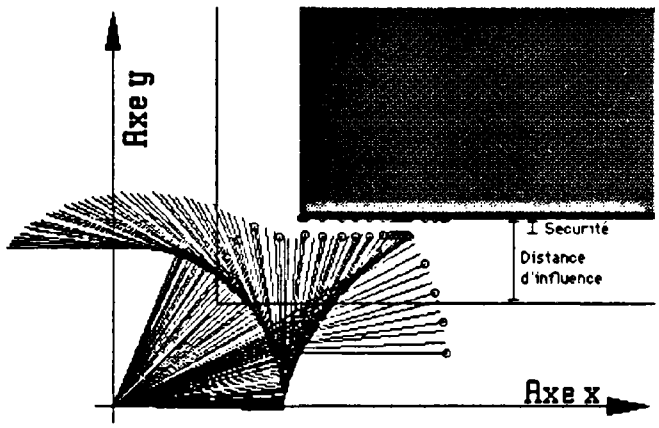


Figure 3.10 : Espace ambiant : Positions successives d'un robot à deux axes.

L'exemple suivant montre le comportement de l'algorithme pour un robot à deux bras et un obstacle simple, dans l'espace ambiant sur la figure 3.10, :

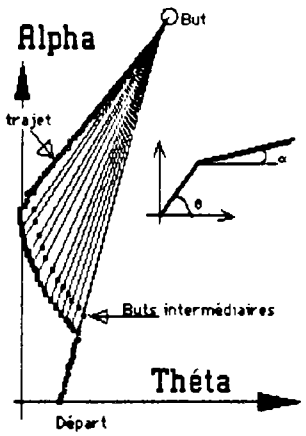


Figure 3.11 : Espace des configurations : Configurations successives corresondant à la même trajectoire que dans la figure précédente.

et dans l'espace des configurations sur la figure 3.11.

Ce programme a été réalisé et intégré comme une commande supplémentaire du module robotique du logiciel de CAO Catia.

Les tests ont été réalisés pour des applications dans l'industrie automobile, dans le domaine de la soudure par points et du montage.

Si les cas d'échecs de ce programme ont été déjà vus du point de vue théorique, il est intéressant d'observer son comportement dans les cas industriels.
Le programme trouve bien la solution lorsqu'elle comporte une déviation de faible amplitude. Les cas industriels comportent souvent des points de soudure à réaliser en ligne les uns à la suite des autres, ce qui rend efficace le contrôle des collisions. Dans les postes de conformation, le contournement d'un "serrage" implique une très grande déviation que le programme ne trouve pas seul : heureusement, dans les cas ordinaires, l'ajout manuel d'un seul point de passage imposé est facile et permet au programme d'obtenir une solution convenable.

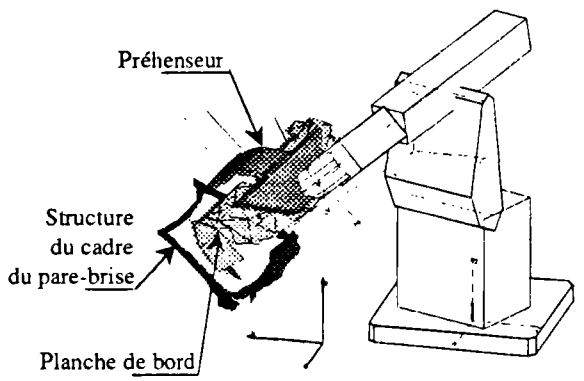


Figure 3.12 : Modèle CAO simplifié du poste de montage de la planche de bord (image Catia). Ce modèle a été utilisé pour les simulations de trajectoires de mise en place du poste de conduite.

Un exemple industriel de montage

Nous avons testé le programme sur le montage d'une planche de bord complète de véhicule XM en passant par le pare-brise, présenté figure 3.12. une tâche équivalente avait été réalisée aussi à la main en CAO avec un contrôle visuel des collisions et une mise en place manuelle des points de passage. Cette tâche avait été réalisée au prix d'une très grande quantité de points de passages. Les mouvements proposés par cette tâche étaient le reflet des hésitations et des difficultés de contrôle manuel.

Pour cet exemple industriel test très particulier, le programme a toujours échoué pour obtenir une trajectoire, même lorsque des points de passages intermédiaires lui étaient donnés manuellement . En général, il essayait de faire passer la planche de bord, mais le haut du pavillon venait se bloquer entre le volant et la planche de bord.

En fait, il est apparu que la trajectoire manuelle comportait des collisions très difficiles à visualiser à cause de la complexité de la scène. Une conviction s'est faite qu'il était probablement impossible de trouver une tâche sur cette base. Il aurait été très utile qu'une méthode puisse prouver la non-existance de solution.

Le même montage, sans le volant, a été réalisé avec succès. Comme le montre la figure 3.12, l'algorithme a trouvé une solution convenable grâce à l'ajout manuel de deux points de passage obligés : l'un pour indiquer d'orienter convenablement la planche pour la faire entrer par une de ses extrémités, l'autre pour situer une position de dégagement avant l'accostage final.

3.3.3 Modification de trajectoire pour éviter les échecs

Les méthodes locales, parce qu'elles ne disposent pas d'information loin de la position courante, ne sont pas assurées de trouver toujours une solution, même s'il en existe une.

Il faut alors faire des propositions, afin de détecter les échecs par blocage à cause d'une configuration particulière des obstacles.

Détection des blocages

Le robot avance le plus vite possible vers les sous-buts Q_b successifs, tels que la fonction $\mathcal{U}(Q)$ les définit. Les obstacles, sous forme de contraintes linéaires, dévient le robot de sa trajectoire nominale qui aurait été d'aller droit vers le sous-but.

En cas de concavité sur un obstacle, la contrainte dévie le robot vers le fond de la concavité. Cette éventualité peut même se produire sur une face faiblement convexe, éventuellement plane, d'un obstacle si le point qui minimise $\mathcal{U}(Q)$ est tel que la face présente une courbure moindre que celle de la sphère centrée sur ce point. La contrainte y est perpendiculaire au gradient du potentiel et le robot ne peut plus progresser.

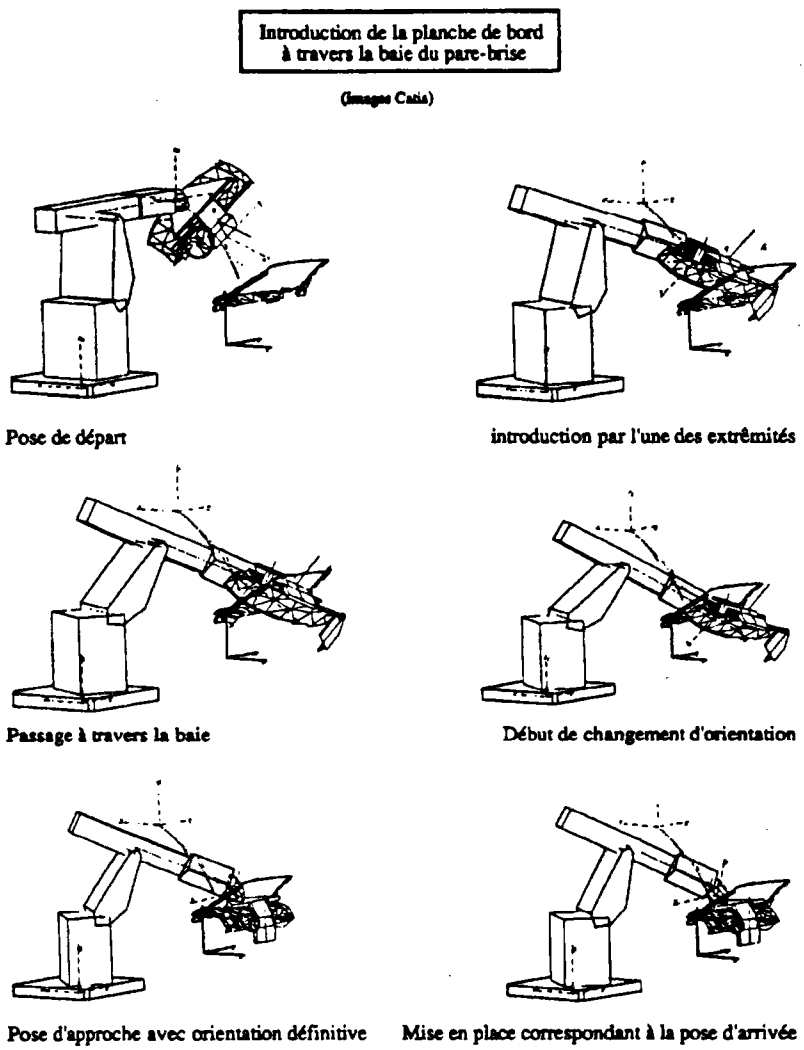
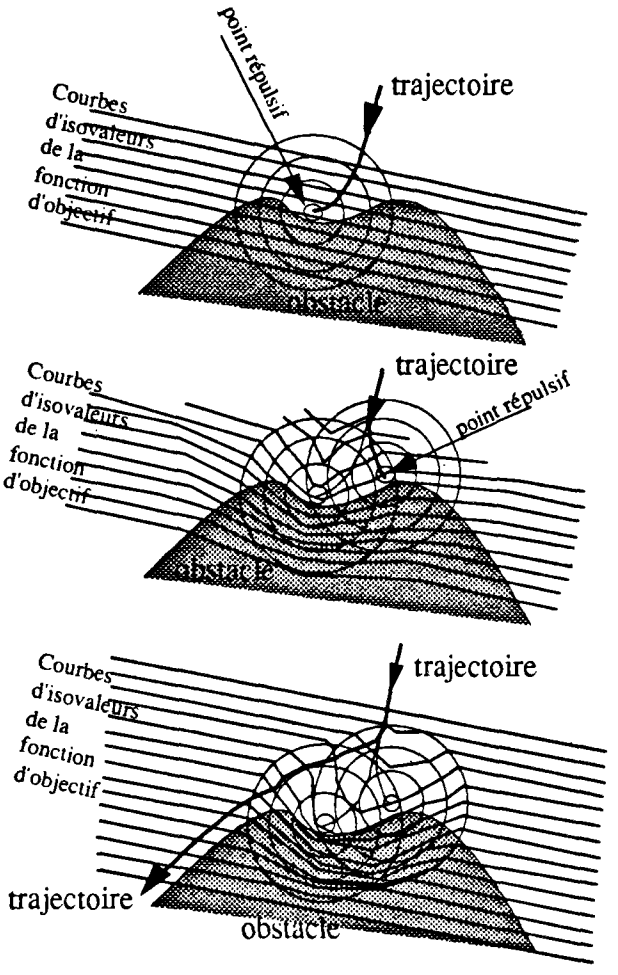


Figure 3.13 : Simulation CAO avec contournement d'obstacle : la trajectoire est obtenue par la méthode des contraintes. La présence du volant est négligée car il est probablement impossible de monter la planche de bord avec le volant (collision visible à l'étape 3). Les position intermédiaires 3 et 5 ont été données manuellement, le programme construisant alors une trajectoire sans collision pour l'ensemble de la trajectoire

En cas de pluralité d'obstacles, la dimension du champ d'évolution des paramètres diminue, car il s'agit d'intersection des hyper-surfaces les

représentant. Ainsi peuvent apparaître des restrictions de mouvements, lesquelles risquent d'engendrer des phénomènes ressemblant à ceux des minimums locaux.

Tentatives de déblocage



Le robot est dévié puis bloqué par l'obstacle.
Au point de blocage est créé un potentiel répulsif. Le mouvement est recommencé depuis un point nettement moins contraint.

La force de répulsion pousse le robot vers un autre point de blocage où un autre potentiel répulsif est ajouté.

La répulsion combinée des deux points et l'attaction vers l'objectif poussent le robot hors de la zone de blocage. Le robot évolue ensuite le long de l'obstacle et tente de rejoindre le but.

Figure 3.14 : Principe de débloquage efficace si la zone de difficulté est petite

Sous l'effet de la fonction d'objectif attractive vers le but en ligne droite, un blocage ou une tendance au blocage au voisinage d'un minimum local peut se manifester.

Il peut être intéressant dans ce cas d'introduire une composante répulsive centrée sur le point où ce blocage se manifeste. La combinaison des deux

composantes a ainsi tendance à pousser le robot hors du voisinage du point ou se manifestait le blocage.

Le blocage peut ainsi avoir disparu, mais il peut aussi s'être déplacé vers un autre lieu de minimum local. au bout de quelques essais, la zone de risque est entièrement dans un domaine ou la fonction d'objectif est très élevée, ce qui favorise les trajectoires ne passant pas dans cette zone.

L'intérête de cette méthode est de dissocier les problèmes de collision des difficultés de convergence, permettant ainsi une gestion simple des collisions dans l'espace ambiant, tandis que la fonction d'objectif porte sur les configurations. Cependant, en cas de blocage sévère, le processus envisagé est inefficace car Il est difficile de prévoir la réussite ou l'echec de l'algorithme. La pénalisation de la fonction d'objectif sur une grande envergure est très lente, puisqu'il faut redéfinir la trajectoire depuis un point où la fonction d'objectif progressait normalement. Ce critère peut être surveillé au cours de la trajectoire en vérifiant que la trajectoire progresse vers le but avec des variation de la fontion d'objectif du même ordre de grandeur que la moyenne des progressions précédentes.

Le plus simple est d'introduire un "potentiel" répulsif centré sur le point de blocage. La combinaison des deux "potentiels" (celui du sous-but en ligne droite et celui du point de blocage) pousse le robot hors du point de blocage. Par la suite, le blocage peut avoir lieu à un endroit différent à cause de la combinaison des deux champs. On introduit un nouveau champ répulsif centré sur ce nouveau point de blocage, et ainsi de suite en espérant qu'ainsi le robot atteindra effectivement le but. Par cette méthode, on remplit par "apprentissage" les zones de blocage par des nuages de points répulsifs. Quand une zone est suffisament couverte par de tels essais infructueux, le robot est expulsé dans une autre zone, vers le but final peut-on l'espérer.

Cette méthode est valable pour sortir le robot des petits à-plats ou des petites concavités. Si la concavité est importante, le remplissage en sera fastidieux.

La ou les contraintes blocantes étant connues, une fois le blocage réalisé, on peut modifier les obstacles eux mêmes, par exemple en remplaçant les objets fixes responsables du blocage par l'enveloppe convexe de leur union. En pratique, ceci ne conduit pas vraiment au succès, car l'image d'un objet convexe dans l'espace ambiant, n'est pas nécessairement convexe dans l'espace des configurations. Le blocage risque fort de persister.

Deuxième Partie

Synthèse et propositions : Implantation des robots en tenant compte des obstacles : méthode des articulations virtuelles

Les analyses des propositions de la partie précédente permettent d'envisager maintenant l'implantation optimale et sûre des robots.
Cette partie étudie ce problème et l'analyse par étapes : l'accessibilité d'abord, ensuite le domaine d'implantation possible du robot sans tenir compte des problèmes de collisions puis l'implantation optimale d'un robot sans tenir compte des obstacles. Enfin, elle aboutit à la présentation en détails la "méthode des articulations virtuelles" que nous proposons pour résoudre l'implantation optimale des robots en tenant compte des obstacles.

Nous proposons ainsi pour trouver le domaine d'implantation du robot un algorithme rapide utilisant une méthode mixte géométrique et analytique, dont le fonctionnement est présenté en détails en annexe B. Son intérêt réside dans sa rapidité, son aptitude à détecter la présence de vides éventuels dans le domaine d'implantation, son aptitude permettant d'envisager des heuristiques de déplacement du robot et de changement des caractéristiques de l'outil, performances qui ont été appliquées avec succès pour un projet "SARTRE" de reconfiguration de tôleries polyvalentes.

Nous proposons ensuite un algorithme d'implantation optimale de robots très efficace lorsque les problèmes de collisions peuvent être traités indépendamment par un autre moyen, ce qui est le cas de nombreux problèmes de soudure par points en environnement peu encombré des chaines de montage de voitures. Ce programme est utilisé massivement dans ce but par l'industriel.

Enfin, nous proposons cette méthode des articulations virtuelles pour implanter le robot en tenant compte des collisions. Il s'agit d'une méthode locale procédant par petits déplacements, considérant les libertés d'implantation du robot comme des articulations supplémentaires virtuelles du robot. Ces variables virtuelles doivent respecter des contraintes particulières permettant d'aboutir à une tâche réalisable par le robot réel.
Cette méthode utilise une tâche définie d'une part par des configurations d'arrêts, dont les buts imposés sont de faire coïncider le repère actif de l'outil avec un repère de l'espace ambiant, d'autre part par des configurations de passage librement disposées pour éviter les collisions. L'optimisation déplace la base du robot et les points de passage pour diminuer la durée du mouvement : cette durée est calculée à partir de la tâche définie par les configurations d'arrêt et de passage, cela sans essayer d'optimiser le mouvement sur la trajectoire.
Les configurations de passage doivent être disposées aux endroits de risque maximum de collision, ce qui est fait à chaque cycle du processus itératif en

utilisant la méthode des contraintes de B. Faverjon. Un processus de lissage simple intervient pour limiter la prolifération des configurations de passage.

Ces travaux permettent de conclure sur des implications industrielles encourageantes, bien que cette méthode des articulations virtuelles proposée n'ait pas encore été testée sur des applications industrielles : en particulier, elle peut aisément être étendue pour résoudre les problèmes d'optimisation conjointe des variables de la tâche et d'autres variables du problème. Par exemple, il est envisagé d'optimiser la forme de l'outil de cette manière.

Chapitre 4

Deux propositions préalables pour définir le domaine d'implantation puis pour l'implantation optimale des robots sans prise en compte des collisions

Ce chapitre permet de présenter le problème de l'accessibilité non pas par rapport au robot lui-même, mais au sein de la cellule robotisée, en fonction de l'implantation du robot, en proposant d'abord le calcul du domaine d'implantation des robots, sans prise en compte des collisions. Ce calcul s'adresse aux cas ou le problème des collisions est envisagé a priori, soit par le choix de dimensions de l'outil adéquates, soit par l'implantation de points de passages systématique pour contourner l'obstacle, soit, la tâche comportant un risque vraiment faible, par une résolution considérée comme sans influence sur l'implantation du robot. Quoi qu'il en soit, il existe une large classe de tels postes robotisés dans l'industrie automobile, en particulier les postes de reprise.

Nous proposons d'abord une méthode mixte géométrique et analytique, pour déterminer le domaine d'implantation du robot. Nous avons programmé cette méthode. Son fonctionnement précis est exprimé en annexe A.

Puis nous proposons une méthode cellulaire d'optimisation de l'implantation d'un robot pour une tâche donnée sans prise en compte des obstacles. Notre programme permet des gains très importants dans les cas industriels, où il est de ce fait utilisé massivement, même si des obstacles sont à considérer, quitte à choisir une implantation parmi les meilleures trouvées.

4.1 Implantation du robot : l'accessibilité

Le problème de l'accessibilité est presque toujours traité à partir du robot [Gup86], sans préoccupations d'ensemble sur la tâche qu'il aura à effectuer. Cette approche est probablement encouragée par le fait que le domaine accessible ainsi défini est une caractéristique propre au robot. Il est ainsi assez facile de positionner le robot de manière à rendre accessible un point de l'espace. Cependant, comment faire pour rendre accessible une gamme de points, avec une orientation de l'outil convenable ?

Cette facon de procéder inverse généralement les rôles : l'industriel ne possède pas un robot donné dont il ne sait que faire, et auquel il se propose d'affecter si possible une tâche. Le processus industriel propose plus souvent des tâches à réaliser a priori. Le robot doit être ainsi choisi de manière optimale parmi ceux proposés dans le commerce, ou plus rarement conçus spécialement pour la tâche prévue.

En réalité, si les tâches sont souvent conçues a priori, c'est que leur concepteur fait en sorte qu'elles semblent convenir aux types de robots qu'il prévoit. Quelquefois même, la tâche envisagée doit venir en complément d'une tâche déjà effectuée par un robot existant pour une partie de son temps de travail.

Au problème de l'accessibilité correspond toujours un couple (robot,tâche) indissociable. Les projets industriels commencent à prendre en compte cette globalité en travaillant à un niveau encore plus général, sur une tâche d'ensemble du process. Chaque "atome" de tâche doit être affecté à l'un des robots disponibles ou envisagé, la construction des tâches pour chaque robot étant ainsi réalisée au mieux en fonction de critères bien choisis. C'est ce dont il s'agit dans le projet industriel Sartre [Yva90].

4.1.1 Domaine d'implantation

Espace de travail

L'approche du problème de l'accessibilité, menée en suivant une optique "commande", conduit à la recherche de l'espace de travail du manipulateur, c'est à dire de la zone dans laquelle un repère visé pourra être atteint par l'organe terminal du robot, c'est à dire le repère actif de l'outil. l'optimisation de ce volume à partir de variables mécaniques est envisagée par [VW86].

Mais certaines approximations de cet espace, suivant les besoins, sont proposées :

Définition 24 *L'espace constructeur :*
Les constructeurs de robots proposent tous dans leur documentation une description de la zone accessible par le manipulateur. C'est plus exactement l'espace accessible par le centre du poignet lorsque le poignet est à axes concourants, ou celle d'un point caractéristique de celui-ci dans les autres cas. Il faut ensuite, connaissant l'outil, déduire de cette zone l'espace de travail du robot muni de cet outil.

Définition 25 *L'espace de travail du robot en position est le volume engendré par le centre du poignet et le repère actif de l'outil dans leurs possibles mouvements. Tous les points de cet espace sont accessibles s'il n'y a pas de contraintes d'orientation, car les orientations ne sont pas toujours toutes possibles.*

Définition 26 *L'espace approché de travail du robot en position est l'espace constructeur augmenté de tous les points dont la distance à celui-ci est inférieure à la distance entre le centre du poignet et celui du repère actif de l'outil.*

Cet espace correspond en grande partie à l'espace de travail du robot en position, mais certains points de cet espace ne sont pas accessibles à cause des angles morts du poignet, ou à cause de la forme de l'outil, qui, s'il est de grande dimension, peut laisser des zones inaccessibles au centre de cet espace.

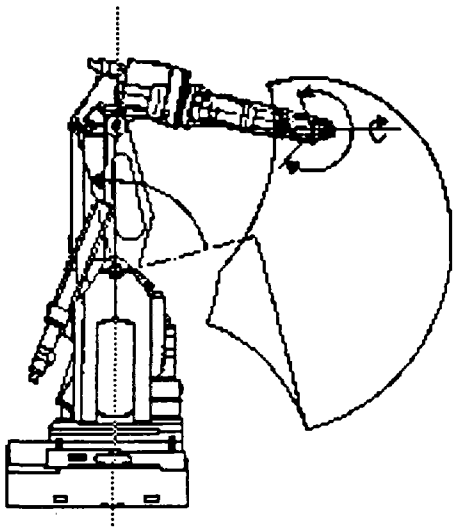


Figure 4.1 : Accessibilité du poignet pour le robot Acma X58 (Renault Automation).

Définition 27 *L'espace de travail complètement accessible décrit l'espace de travail dont tous les points sont accessibles quelle que soit l'orientation de l'outil.*

Cet espace, s'il existe, est généralement très réduit : fort peu de points de l'espace sont accessibles quelle que soit l'orientation demandée. Il ne permet pas aisément l'étude du process de manière acceptable.

Orientation de l'outil

Ces espaces ne suffisent évidemment pas à déterminer si le robot est capable d'accéder à une pose de l'espace, puisque les poses à atteindre exigent en outre une orientation précise de l'outil. L'espace de travail du robot, pour tenir compte de l'orientation, est de dimension 6, ce qui est difficile à représenter.

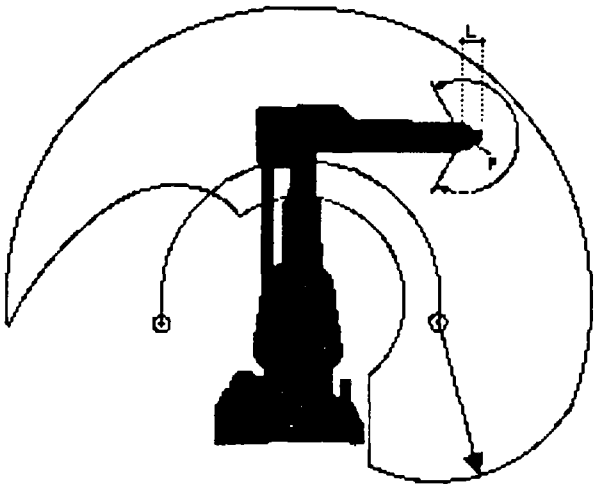


Figure 4.2 : Accessibilité du poignet pour le robot Asea IRB 2000 (ABB).

Configurations du robot

Le robot pouvant accéder à une pose donnée selon plusieurs configurations, l'espace de travail est à considérer pour chacune d'elles de manière indépendante, afin de contrôler les changements de configurations qui donnent lieu à des impossibilités ou à des mouvements de grande amplitude (envoi du bras du robot dans une zone libre, pour effectuer un retournement de poignet par exemple) : ils sont de la plus grande importance pour la conception du process.

A orientation donnée de l'organe terminal, la description de l'espace de travail est réalisée en considérant les limites de débattement des articulations du poignet du robot. Les travaux réalisés jusqu'à présent montrent qu'il est difficile de tenir compte de l'orientation de l'organe terminal du robot [WC90] [RD87].

L'une des approches [Lu 88] suppose que les articulations du poignet n'ont pas de limites à leurs débattements. Ceci n'est acceptable que lorsque les angles

morts des différentes articulations mises en cause sont très petits. Un raisonnement uniquement géométrique construit le volume accessible (à orientation fixée) à partir des zones balayées successivement par chacun des axes du robot.

Une autre approche, [Bor85] et [CW87], fondée sur le concept d'aspects (les différentes configurations du robot), étudie complètement l'orientation de l'organe terminal. Il permet de bien comprendre la complexité du problème de l'accessibilité. En particulier, l'étude des suivis de courbes, montre que, bien que chaque pose de la trajectoire soit accessible, la tâche est souvent impossible à réaliser à cause d'une reconfiguration nécessaire (retournement de poignet par exemple). Pour atteindre ces résultats, et en particulier pour déterminer les frontières correspondant à chaque aspect, de nombreuses méthodes numériques sont mises à contribution, ce qui rend le processus très complexe et coûteux en temps de calcul.

Espace d'implantation

Une approche différente consiste à traiter le problème de l'accessibilité avec l'optique process, laquelle conduit à tenter de définir l'espace d'implantation du robot.

Définition 28 *L'espace d'implantation du robot est obtenu en décrivant la zone dans laquelle la base du robot doit être placée pour pouvoir accéder à une pose donnée (orientation et position).*

Cette approche peut apporter de sérieux avantages déterminants pour la conception, la mise au point et la simulation des sites robotisés.

4.1.2 La méthode cellulaire

Une approche possible pour définir l'espace d'implantation est l'approche "discrète", c'est à dire celle faite par un découpage de l'espace en cellules élémentaires. Bien qu'aucun système de C.A.O. robotique ne propose vraiment d'outil de définition de cet espace, certains disposent d'une fonction dite "Auto-place" qui propose des implantations possibles du robot, selon une discrétisation simple, grâce à un procédé de ce genre.

Une zone de recherche est définie à partir de considérations géométriques élémentaires sur le robot et sur l'environnement (centre de la sphère minimale contenant les poses à atteindre, rayon d'action maximal du robot muni de son outil,...), elle est ensuite "discrétisée" par un balayage en position ou en orientation sur la base du robot.

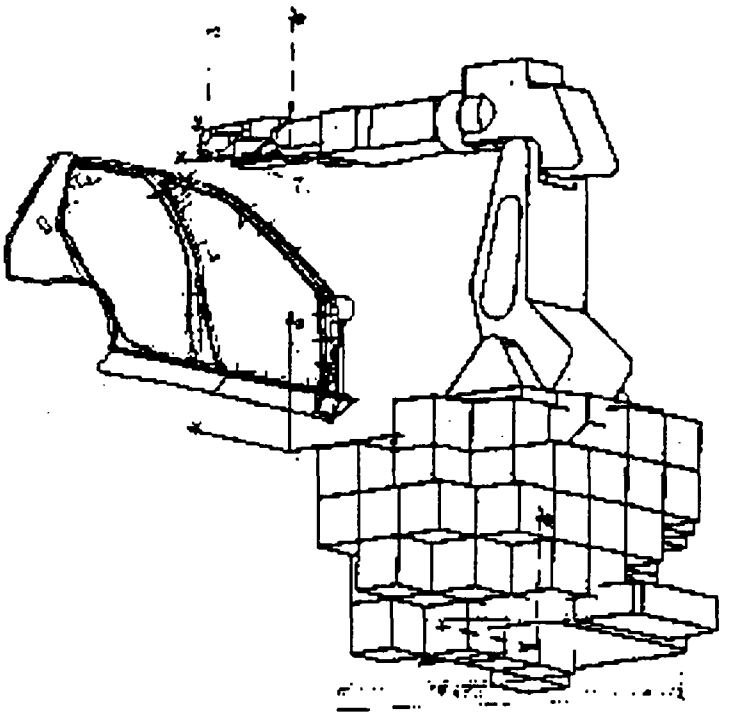


Figure 4.3 : Implantation possible pour une tâche (Acma X58) : approximation du volume d'implantation par discrétisation des variables d'implantation.

Ceci
permet (figure 4.3) d'obtenir une approximation de l'espace d'implantation. Cette méthode peut se généraliser aux robots industriels quelconques, si on possède des inverseurs de coordonnées efficaces pour chaque manipulateur.

Mais le coût en temps de calcul de l'algorithme est élevé. Le volume obtenu est difficile à manipuler, en particulier la présence de trous dans le volume n'est pas facile à détecter. Si l'espace d'implantation est très découpé ou très réduit, l'algorithme risque de ne pas trouver de solution alors même qu'il en existe une.

Par ailleurs, en vue d'une meilleure utilisation de l'espace d'implantation, il n'est pas facile d'en dégager une heuristique commode de déplacement du robot pour ajouter une tâche élémentaire à son travail, ce qui aurait permis plus facilement d'envisager des applications autour de systmes experts.

4.2 Notre proposition d'une méthode nouvelle mixte géométrique et analytique

Cette méthode propose de définir le volume global d'implantation. Elle se fonde sur une étude du volume d'implantation pour chaque pose à atteindre de la tâche. Puis le volume d'implantation du robot pour l'ensemble de la tâche à effectuer en est déduit par intersection des volumes d'implantation pour chaque tâche élémentaire.

En chaque pose à ateindre, l'étude menée dépend du robot utilisé. L'annexe A montre celle-ci pour un robot vertical à poignet concourant, qui est le cas le plus courant dans l'industrie manufacturière. La méthode utilise la géométrie du robot pour déterminer le volume d'implantation du robot pour une pose, en tenant compte des limites de débattement des articulations du poignet et de celles des boucles cinématiques (figure 4.4).

L'interêt de cette méthode réside dans sa rapidit, car elle évite de ballayer tout l'espace. Elle permet aussi de prévoir et de détecter facilement les trous dans le domaine d'implantation. Elle permet enfin de proposer des stratégies de déplacement du robot qui ont été utilisées dans le système expert d'aide à la reconfiguration de tôleries polyvalentes SARTRE [Yva90].

4.3 Implantation optimale du robot par la méthode cellulaire

4.3.1 Nouveau processus proposé

Il s'agit, pour chaque pose de la tâche, de passer du stade de la seule accessibilité à celui de l'optimisation de l'implantation en permettant de répondre aux questions suivantes :

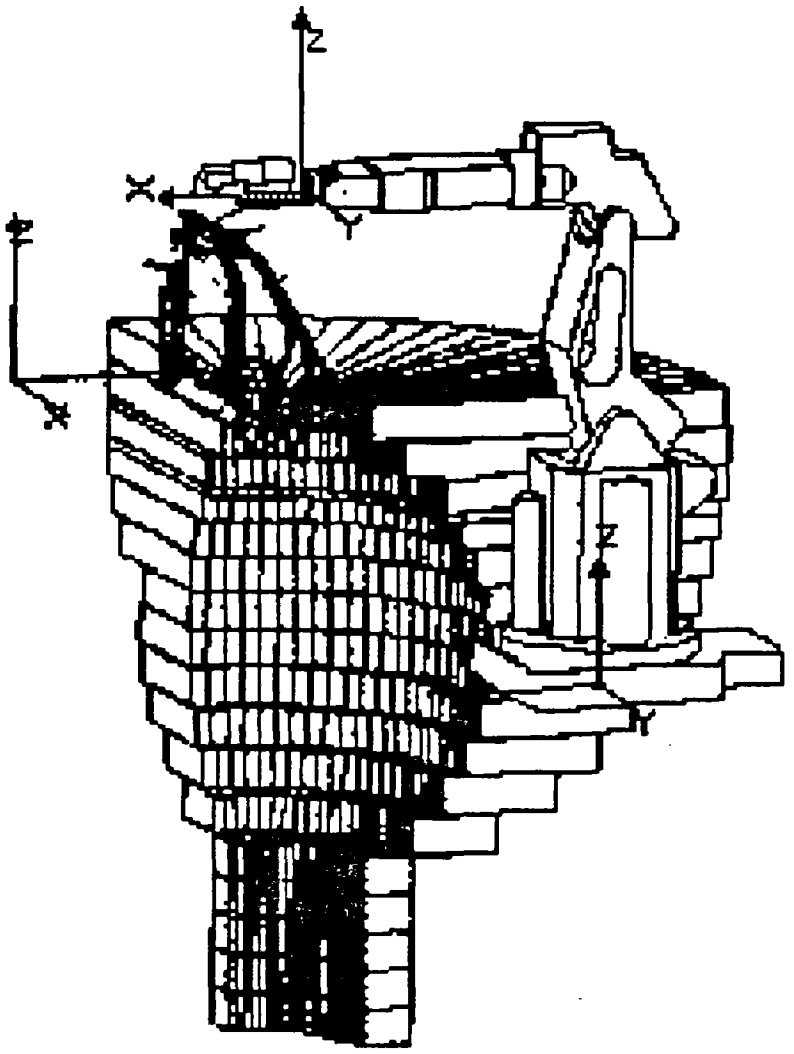


Figure 4.4 : Demi-zone d'implantation pour un point de soudure (Acma X58).

- Où peut-on implanter le robot pour que la tâche qui lui est assignée soit faisable ?

- Où faut-il implanter le robot pour que cette tâche soit effectuée de manière optimale, c'est à dire généralement dans le temps le plus court possible ?

Obstacles : la méthode d'optimisation considérée ici suppose que les problèmes de collisions auront été traités par l'opérateur en dehors des problèmes d'implantation optimale. Elle s'applique ainsi aux problèmes d'implantation optimale pour lesquels les risques de collisions sont faibles, ou à ceux pour lesquels on considère que les collisions sont évitées par avance, grâce à des points de passages judicieusement disposés par l'opérateur. Dans l'industrie, ces cas correspondent en particulier aux postes de reprise de soudure par point.

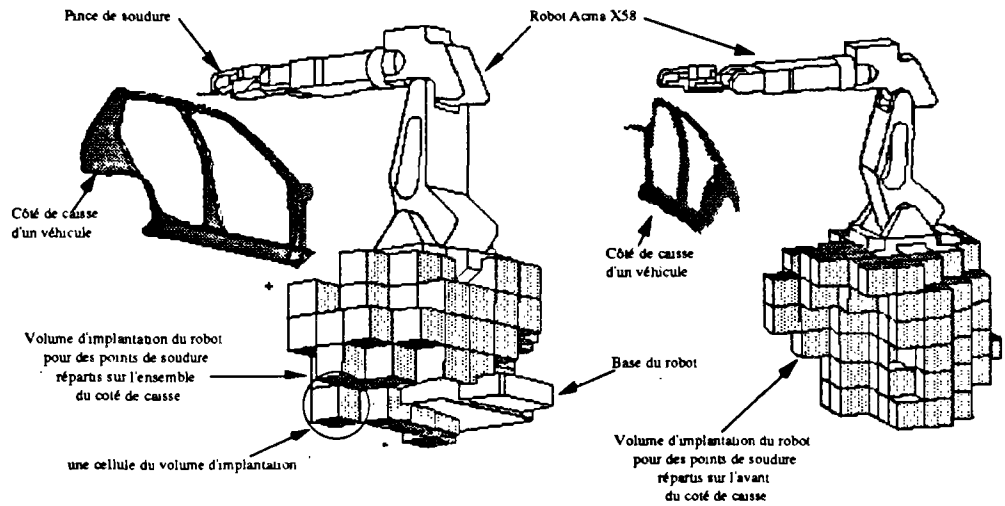


Figure 4.5 : Méthode cellulaire : l'espace est balayé systématiquement ; les cellules d'implantation possible sont visualisées au cours de l'étude ; le calcul de la durée de la tâche est effectué. Les meilleures implantations possibles sont mémorisées.

Recherche globale : Il ne s'agit pas d'une méthode d'optimisation au sens habituel de la recherche opérationnelle, dont les méthodes sont presque toujours locales. C'est ici une méthode par ballayage où, grâce à une étude répétée pour chaque cellule de l'espace, l'optimum est trouvé non seulement en étudiant par un tel processus l'accessibilité, condition minimale d'existence de la tâche, mais aussi en optimisant le critère désiré, généralement le temps d'exécution de la tâche. Du début à la fin du processus de recherche par balayage avec un pas donné sur chacune des libertés d'implantation autorisées, une liste des implantations valides est constamment tenue à jour pour construire une approximation du volume d'implantation. Une liste ordonnée des meilleures implantations trouvées jusque là permettra à l'opérateur de choisir la meilleure

implantation trouvée, parfois l'une des meilleures s'il doit tenir compte d'autres facteurs non pris en compte dans cette étude.

Durée de la tâche : puisqu'il s'agit de simulations, l'état du modèle avant le lancement de la tâche dépend des tentatives et des solutions qui ont été essayées précédemment, lesquelles ne reflètent en rien l'enchaînement des tâches à accomplir par le robot réel. Le calcul de durée est mené en neutralisant les effets d'un état aléatoire du robot au départ de la tâche. En conséquence, il n'est pas tenu compte de la durée de la première instruction, qui, en effet, partant de cet état aléatoire à une consigne donnée, est ainsi de durée variable. Pour éviter que cette incertitude ne se propage aux autres mouvements, la première consigne est absolue, c'est à dire entièrement déterminée quelle que soit la consigne précédente, par exemple par une position de repli absolue ou par une configuration fixe. De cette façon, l'état du modèle à la fin d'une simulation précédente n'influe en rien sur le calcul de la durée de la tâche.

Calcul du temps de cycle : il est réalisé :

- soit avec l'algorithme théorique présenté par le fabricant de robot comme reflétant correctement l'algorithme de l'armoire de commande réelle, lorsque il a été possible d'obtenir des informations dans ce demaine, ou bien lorsque le fabricant du robot propose un programme de ce type,

- soit par une loi de vitesse trapézoïdale, comme dans le chapitre 2.

Résultats : cette méthode permet de visualiser le volume d'implantation du robot sous la forme d'une collection de volumes élémentaires de la taille des cellules. Pour chaque cellule envisagée la durée de la tâche est considérée être celle correspondant à l'implantation du robot au centre de la cellule. Ainsi, l'approximation n'est elle acceptable que dans la mesure où la taille des cellules correspond à la précision d'implantation demandée. Comme dans toute méthode discrétisant les phénomènes, il est en toute rigueur très difficile d'assurer une accessibilité ou un temps de cycle voisin pour tous les points de la cellule.

4.3.2 Réalisation opérationnelle de nos propositions pour une utilisation industrielle

Cette méthode a été mise en oeuvre sur le système de CAO Catia comme une fonction interactive complémentaire de son module standard de robotique. Les données de description précise des articulations, des caractéristiques des lois de variation des moteurs du robot sont disponibles dans un fichier standard pour toutes les applications robotiques de l'industriel.

Repères : le domaine d'implantation est étudié à partir d'un repère quelconque de l'espace. Les emplacements possibles pour la base du robot sont déduits de la position en cours de celle-ci, par des translations suivant les trois axes de coordonnées. Si cela est utile, une liberté supplémentaire optionnelle de rotation suivant le troisième axe de ce repère est disponible,.

Visualisations : il n'est pas facile de présenter en 3 dimension les résultats de ce problème d'optimisation avec 4 variables. Le volume d'implantation est visualisé pour les 3 translations par des parralélépipèdes. La liberté en rotation est représentée partiellement : lorsque la meilleure implantation n'est pas celle correspondant à l'orientation initiale, la couleur de la cellule est spéciale. En effet, le repère est souvent choisi avec le troisième axe vertical, tandis que le robot est souvent d'un type possédant une première articulation rotoïde d'axe vertical. Dans ce cas, l'orientation est indifférente pourvu que les butées de cette première articulation n'interviennent pas.

Les meilleures implantations sont visualisées chacune par un repère puis le robot est déplacé à l'implantation optimale si cela est désiré.

Observations : si la discrétisation de l'espace est suffisament fine, l'opérateur dispose d'une collection d'implantations les meilleures pour le temps de cycle. Elles se trouvent situées dans une zone, plus rarement plusieurs, de l'espace accessible visualisé, tandis que les implantations les pires sont elles aussi regroupées dans d'autres zones de cet espace. Suivant l'importance des autres facteurs d'optimisation qui n'ont pas été pris en compte, il est possible d'envisager de choisir une implantation dans une zone favorable pour le temps de cycle parce que moins pénalisante pour les autres critères ainsi intégrés "manuellement".

Tests : notre programme est utilisé systématiquement pour les implantations nouvelles de robots ou pour les modifications sur les chaînes flexibles de soudure

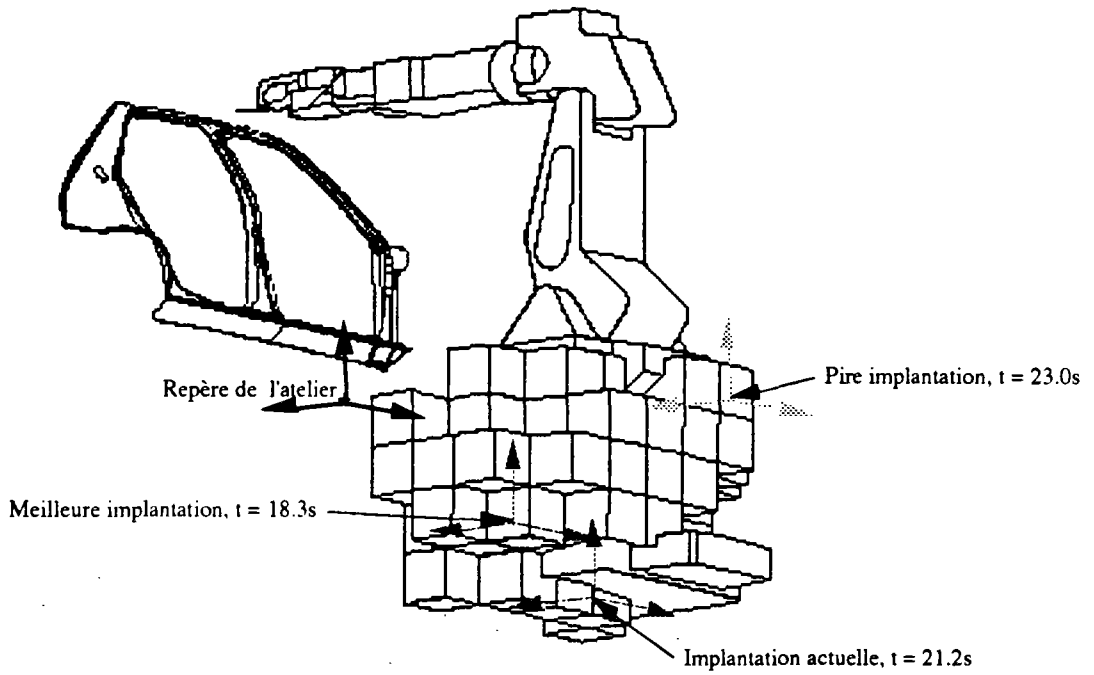


Figure 4.6 : Méthode cellulaire : après un balayage systématique de l'espace, les meilleures implantations possibles sont visualisées par des repères correspondant à celui de la base du robot pour ces implantations.

par points chez l'industriel PSA. Il contribue beaucoup à la diffusion de la CAO pour la robotique, tant les problèmes de durée sont cruciaux pour la production.

Les bénéfices obtenus grâce à l'optimisation de la durée de la tâche par une implantation adéquate du robot sont appréciables [Gom90] : pour des tâches de soudure par points pour lesquelles les poses à atteindre sont assez proches les unes des autres (par exemple des lignes de points de soudure), l'amélioration est de l'ordre de 10 pour cent. Pour des poses à atteindre plus espacées ou disséminées dans le volume accessible par le robot, les gains sont meilleurs, de l'ordre de 15 à 20 pour cent. Il faut ajouter que ce processus fonctionne même si la tâche n'était pas réalisable pour l'implantation initiale du robot ce qui dispense l'opérateur du travail de recherche préliminaire d'une position convenable, recherche parfois malaisée. Enfin, il peut arriver que le changement d'implantation permette d'aboutir à des configurations du robot différentes des

configurations de départ ; lorsque cela a lieu, le gain est encore plus appréciable, puisque à elle seule l'économie d'un retournement de poignet ou de coude change l'ordre de grandeur du temps d'exécution de la tâche.

Nous avons traité à titre d'exemple, l'implantation d'un robot Acma X58 dont la tâche consistait à réaliser dix points de soudures situées homogènement sur le contour d'ouverture de porte avant d'un véhicule 205 Peugeot, avec une implantation de départ ne garantissant pas l'accessibilité à tous les points de soudure. Le volume de recherche était de deux mètres parrallèlement à la ligne de montage de la voiture, de un mètre perpendiculairement à cette ligne, et d'un mètre en hauteur, avec un pas de discrétisation de vingt centimètres, soit 250 tests élémentaires, pour une durée de calculs de 50 secondes (ordinateur central IBM et système d'exploitation VM), la durée du mouvement allant de 23 secondes à 18.3 secondes pour la meilleure implantation, comme le montre le modèle de la cellule robotisée de la figure 4.6, issue d'une visualisation avec le logiciel CATIA.

Chapitre 5

Notre proposition : la "méthode des articulations virtuelles"

Ce chapitre constitue l'aboutissement de nos travaux. Il traite le problème de l'optimisation de l'implantation des robots en tenant compte des obstacles. Dans ce but, la méthode proposée considère les paramètres d'implantation comme s'ils représentaient des articulations virtuelles du robot considéré. La tâche considérée est alors définie de manière discrète dans l'espace des configurations, par des configurations d'arrêt et des configurations de passage, avec une durée du mouvement calculée sur cette trajectoire. Il s'agit de tenter d'optimiser la durée de cette trajectoire, grâce à une modification simultanée de l'implantation du robot et des configurations de passage.
Lors des déplacements du robot, les configurations d'arrêt sont à reconsidérer pour que soit conservés le but désigné dans l'espace ambiant ; les configurations de passage évoluent simultanément dans un sens favorable à l'optimisation, sous réserve de respecter des contraintes d'anti-collision analogues à celles de la méthode des contraintes présentée à la fin de la partie précédente.

Ce traitement suppose qu'il est possible de garantir la sûreté de la trajectoire par une étude en chaque point de passage, ce qui est réalisé en localisant les points de passage aux endroits de risque de collision maximum. A chaque itération, il convient par conséquent de replacer ces points aux lieux de risque maximum, ce qui est réalisé en utilisant la méthode des contraintes. Elle permettra alors si nécessaire les changements de configuration. Il s'agit ainsi

d'une optimisation avec contraintes : celles-ci permettent d'une part de gérer l'évitement de collisions dans l'espace ambiant, d'autre part de garantir la réalisation effective des points d'arrêt dans l'espace ambiant.

Cette méthode n'est pas encore testée dans les cas industriels pour lesquels elle a été conçue, cependant les principales conclusion peuvent être tirées du prototype que nous avons programmé. Cette méthode peut aussi être utilisée pour optimiser d'autres paramètres que l'implantation, comme la forme de l'outil.

5.1 Conditions pour optimiser l'implantation des robots avec leur trajectoire

A quoi peut-on s'attendre pendant l'optimisation de l'implantation et de la trajectoire d'un robot ? Le but de ce paragraphe est d'en illustrer les aspects les plus importants.
Pour cela, il sera pris comme support un exemple pour lequel le passage de l'espace ambiant à l'espace des configurations est simple : un robot plan cartésien qui possède deux axes de translation perpendiculaires.

Le premier but est de rendre accessible chaque configuration de la tâche. C'est ainsi à réaliser dans l'espace des configurations. Le déplacement du robot dans l'atelier provoque une modification des configurations à atteindre ainsi qu'une transformation des zones interdites par les collisions, car ces configurations doivent correspondre aux mêmes positions et orientations de l'organe terminal du robot, dans l'espace ambiant.

Sur l'exemple des figures 5.1 et 5.2, le point B n'est pas accessible mais le devient après un déplacement du robot.

Ceci ne suffit pas, puisque des obstacles peuvent découper l'espace des configurations en zones non connexes, ce qui rend alors impossible l'existence d'une trajectoire depuis une configuration d'une zone jusqu'à une configuration d'une autre zone. En de tels cas, s'il est possible de choisir une autre configuration correspondant à la même position dans l'espace cartésien, les zones interdites de l'espace des configuration deviennent autres : le déplacement du robot doit

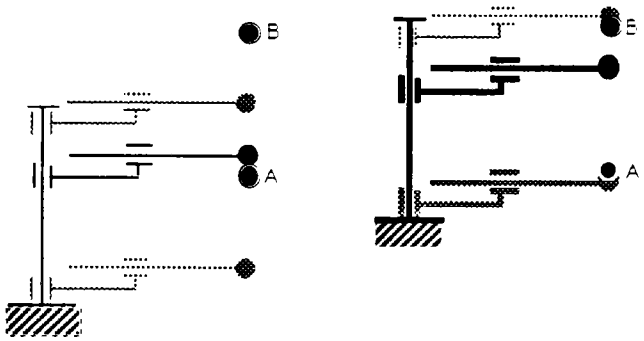


Figure 5.1 : Espace ambiant : positionnement du robot pour que les deux points A et B ne soient pas hors de portée à cause des butées.

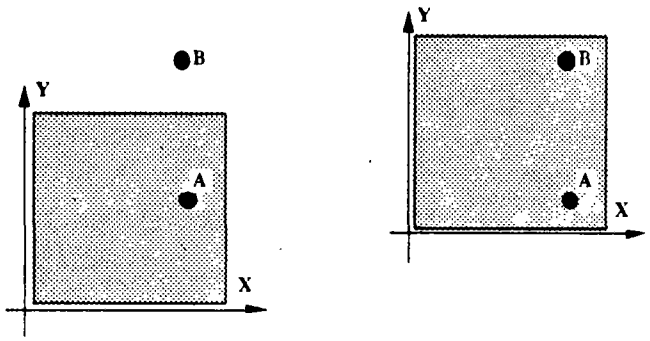


Figure 5.2 : Espace des configurations : positionnement du robot pour que les deux points A et B ne soient pas hors de portée à cause des butées.

alors tenter de faire en sorte que toutes les configurations de la tâche se trouvent dans une même composante connexe de l'espace des configurations, soit parce que les configurations ont pu être regroupées dans la même composante connexe, soit parce que deux composantes non connexes ont pu être réunies par une modification des zones correspondant aux collisions grâce à un déplacement opportun du robot. S'il est possible de réaliser une telle modification, alors la tâche correspondante est exécutable.

Sur l'exemple des figures 5.3 et 5.4, le robot ne peut trouver une trajectoire réalisable reliant les points A et B, car une zone dans laquelle se produiraient des collisions les sépare. En déplaçant le robot, l'espace libre se trouve réunifié, un contournement de l'obstacle est alors possible.

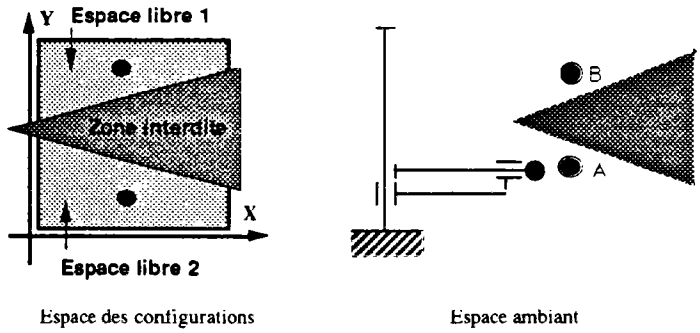


Figure 5.3 : Les configurations A et B sont dans deux composantes non connexes : pas de trajectoire possible.

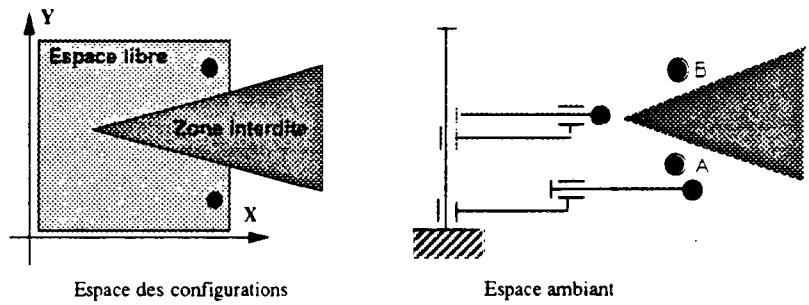


Figure 5.4 : par déplacement du robot, les configurations A et B sont venues dans une même composante connexe : une trajectoire est possible.

Le déplacement du robot induit des modifications sur la définition, et parfois même l'existence, des points de passage de la tâche.

L'implantation du robot peut quelquefois être indifférente :

Par exemple pour un robot cartésien, son déplacement peut parfois simplement provoquer une modification dans l'espace des configurations, qui correspond à une translation d'ensemble de toutes les configurations : dans ce cas, le déplacement modifie la position des configurations de contrôle par rapport aux butées du robot, mais ne change pas les variations de coordonnées entre deux configurations successives de la tâche. Ainsi, chacun des axes du robot réalisant la même variation, le temps mis pour chaque mouvement reste le même.

Dans ce cas, l'implantation du robot est indifférente, pourvu que chaque configuration de la tâche soit accessible.

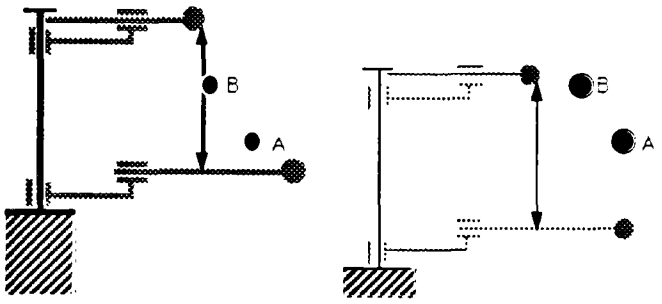


Figure 5.5 : Espace ambiant : positionnement indifférent du robot.

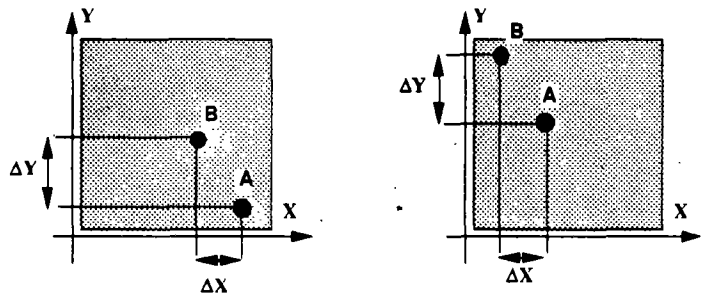


Figure 5.6 : Espace des configurations : positionnement indifférent.

L'exemple des figures 5.5 et 5.6, montre que pour un robot à déplacement selon les coordonnées cartésiennes tel un portique, la modification d'implantation, si elle ne met en jeu que des translations, ne peut améliorer la durée de la tâche, sous réserve que les points de la trajectoire restent dans l'espace accessible et qu'aucun problème de collisions ne se pose.

L'optimisation de la durée de la tâche se traduit par des déplacements des configurations de la tâche dans l'espace des configurations de façon à équilibrer, pour chaque déplacement entre deux configurations successives de la tâche, les variations sur chaque axe du robot. En fonction des méthodes de calcul des durées de déplacement, en particulier des vitesses de déplacement de chaque articulation, ou des temps d'accélération et de décélération, il faut essayer de diminuer les déplacements sur l'axe le plus chargé, éventuellement en augmentant les déplacements sur les autres axes.

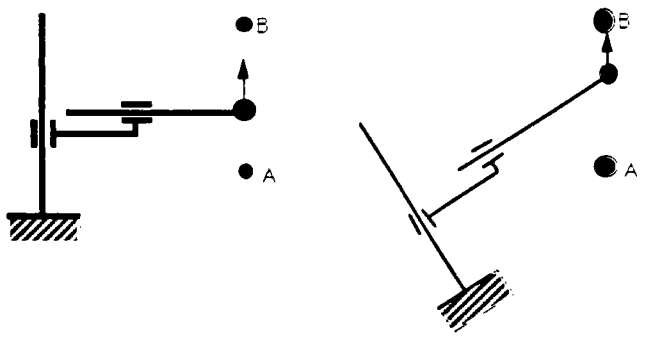


Figure 5.7 : Espace ambiant : répartition équilibrée de la tâche pour chacun des axes du robot.

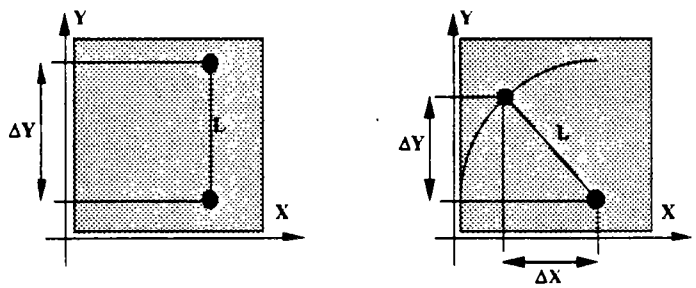


Figure 5.8 : Espace des configurations : répartition équilibrée de la tâche pour chacun des axes du robot.

L'exemple des figures 5.7 et 5.8montre comment optimiser le temps d'exécution de la tâche en implantant le robot avec un angle de 45 degrés par rapport à son orientation initiale : à droite, la distance L est parcourue uniquement en faisant évoluer l'articulation Y, à gauche, la distance L est parcourue en faisant travailler de manière semblable les axes X et Y. Ainsi, les articulations effectuent l'une et l'autre une variation $\sqrt{2}$ fois moins importante que celle qu'aurait accomplie le seul axe Y avant la rotation.

Pour une tâche comportant plusieurs actions, ce raisonnement doit être mené en cumulant chaque mouvement élémentaire de la tâche.

S'il y a des obstacles, la trajectoire est plus ou moins déviée en fonction de l'implantation envisagée pour le robot.

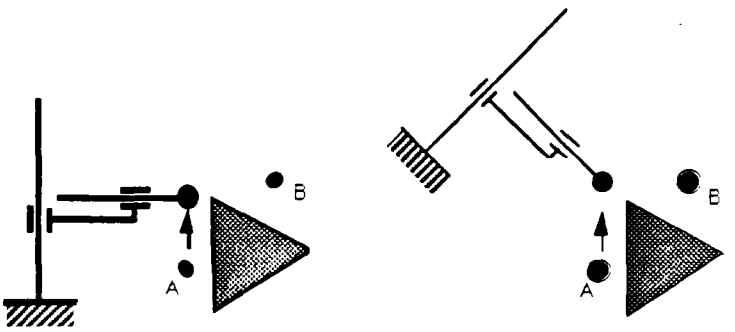


Figure 5.9 : Espace ambiant : positionnement du robot effectué pour équilibrer la repartition de la tâche sur toutes les articulations du robot, par rapport aux obstacles.

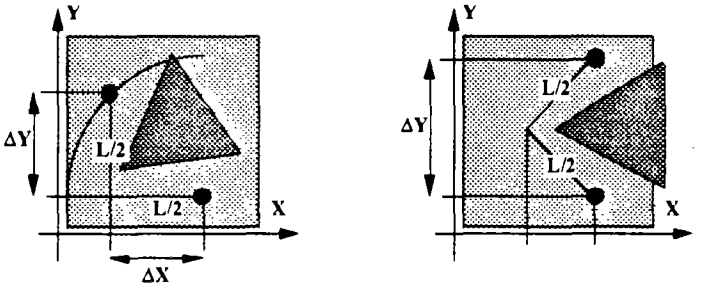


Figure 5.10 : Espace des configurations : positionnement du robot effectué pour équilibrer la repartition de la tâche sur toutes les articulations du robot, par rapport aux obstacles.

L'exemple figure 5.10, montre l'optimisation avec obstacle : dans le cas de gauche, la variation sur chacune des articulations est de $\frac{L}{2}$,mais chaque articulation travaille l'une après l'autre, ce qui donne une longueur totale de L. tandis que dans le cas de droite, la variation sur chaque articulations est de $\frac{L}{\sqrt{2}}$ (ce qui est superieur à $\frac{L}{2}$), mais les mouvements sont exécutés simultanément pour chacune des articulations, ce qui donne une longueur totale de $\frac{L}{\sqrt{2}}$, meilleure que L.

5.2 Intérêt de diverses méthodologies

5.2.1 Optimisations alternées pour la durée de la tâche et pour l'implantation du robot

Il paraît naturel d'imaginer un algorithme qui, d'abord pour une implantation donnée, à partir d'une trajectoire possible mais non optimale, construirait systématiquement une trajectoire optimale. Il chercherait ensuite une meilleure implantation pour que la durée de la tâche diminue.
Ainsi par approximations successives, à chaque fois totalement abouties, un optimum serait-il atteint.

Cependant, si cela présente les avantages de la clarté du processus et d'une simplicité - relative - pour chaque étape, on peut considérer à juste titre qu'il est inutile d'avoir optimisé à grand frais la trajectoire pour une implantation provisoire du robot, pour en chercher une autre au pas suivant en modifiant l'implantation.

Plus grave, cette optimisation de trajectoire pour chaque implantation envisagée tend la trajectoire qui passe de ce fait au plus près des obstacles. Elle passe ainsi beaucoup trop près des obstacles pour que l'optimisation de l'implantation à l'étape suivante du processus ne risque pas d'en être gênée, pouvant même être bloquée par un environnement si contraint, car alors une petite modification de l'implantation du robot peut provoquer une collision au voisinage de nombreux points.

La méthode des articulations virtuelles proposée est un processus qui évite d'avoir à ainsi excessivement préciser les stades intermédiaires.

5.2.2 Autres méthodes possibles

L'implantation optimale *sans tenir compte des collisions* est réalisable, pour un coût en temps d'exécution assez élevé cependant, notamment par une méthode cellulaire. Il peut être envisagé de perfectionner ce processus. Mais comment lier ce processus global d'implantation, avec un processus local de génération de trajectoires sûres ?

En effet, il apparaît que la génération d'une trajectoire optimale *sans collisions*, pour une implantation donnée du robot, est un processus généralement long.

Il peut être imaginé aussi de prolonger le processus global de recherche de trajectoires sans collisions par discrétisation de l'espace, en discrétisant un espace de dimension plus importante, formé à partir des variables d'implantation qui décrivent le problème d'implantation, et des variables articulaires du robot qui décrivent le problème de génération de trajectoire. Mais les méthodes globales deviennent inefficaces avec de telles extensions de l'espace à considérer.

Ainsi, entre les méthodes d'implantation ne prenant pas en compte les collisions et les méthodes de génération de trajectoires sans collisions, la coordination est-elle certes possible, mais le coût en temps d'un tel procédé est prohibitif.

5.3 Esquisse de la méthodologie proposée

5.3.1 Optimisation simultanée de l'implantation et de la trajectoire : principes généraux

Il apparaît comme potentiellement intéressant de chercher une méthode qui traiterait du problème de l'implantation optimale du robot pour une tâche donnée, sans découpler systématiquement la génération de trajectoire de l'implantation du robot. Il conviendrait alors de contrôler dans l'espace ambiant les distances entre les éléments, le problème étant traité dans l'ensemble formé par les variables décrivant la trajectoire et par celles décrivant l'implantation du robot.
Dans un tel ensemble, le processus envisagé peut être fondé sur une étude locale du problème. Ainsi l'optimisation de la trajectoire d'une part et celle de l'implantation d'autre part, pourraient-elles être réalisées conjointement et simultanément par des modifications apportées à une implantation et à une trajectoire qui évoluent simultanément au cours du processus.

5.3.2 Les étapes du processus

Le processus envisagé est itératif. Il s'applique au départ à une trajectoire fournie par l'utilisateur, définie par des configurations de passage et d'arrêt.

Chaque trajectoire en cours de traitement est à traiter successivement en plusieurs étapes.

- Première étape : positionnement convenable des points de contrôle, en vue d'une surveillance correcte des collisions éventuelles.
- Deuxième étape : lissage de la trajectoire, en vue d'éliminer les points de contrôle qui ne sont pas indispensables.
- Troisième étape : optimisation d'ensemble du problème, dans les limites de validité des petits déplacements.

A ce stade, la trajectoire et l'implantation du robot ont l'une et l'autre évolué dans le sens d'une amélioration de la durée de la tâche, en garantissant qu'il n'y a pas de collisions. Mais les changements effectués ont engendré des imperfections dues au processus local utilisé. De plus, le problème a évolué, exigeant une remise au point de la trajectoire : le processus reprend sur les nouvelles bases à la première étape.

Lorsqu'il n'est plus possible d'améliorer la trajectoire et l'implantation du robot, le processus s'achève.

5.3.3 Définition de la tâche en vue d'une méthodologie

Rôle des points de contrôle

La tâche du robot est définie d'une façon qui correspond aux manières de faire habituelles de l'opérateur, telles que définies au chapitre 2.3, mais avec quelques précisions complémentaires. En effet, le choix des points de contrôle est primordial pour le bon déroulement du processus.

L'utilisateur définit une séquence de configurations du robot à atteindre successivement. La tâche se trouve ainsi définie de manière polygonale par une suite de points dans l'espace des configurations.

Les points d'arrêt (définition 8) d'une manière stricte, les points de passage (définition 9) d'une manière plus souple, sont installés au départ par

l'utilisateur, comme points de contrôle permettant de définir la trajectoire de manière discrète par un nombre fini de configurations successives. La trajectoire entre ces points importe assez peu à l'utilisateur qui a placé les points de contrôle de manière telle que le risque de collision ailleurs qu'en ces points soit infime pour peu que le robot suive une trajectoire correctement tendue qui corresponde à l'intuition qu'il en a.

sujétions et contraintes pour les points d'arrêt en cas de déplacement du robot : les configurations d'arrêt étant définies par le process à réaliser par le robot (elles correspondent à des actions à mener dans l'espace ambiant), il n'est question ni d'en ajouter, ni d'en retirer. Les configurations d'arrêt seront seulement mises à jour, en cas de déplacement du robot, de manière à ce que l'organe terminal du robot soit situé exactement dans la position et l'orientation qui avaient été définies au départ par rapport à l'environnement. cette opération est réalisée grâce au modèle géométrique inverse du robot, ce qui permettra éventuellement une reconfiguration du robot pour cette pose.
En adaptant ainsi ces configurations d'arrêt, si l'extrémité du robot reste bien en sûreté, il faut vérifier les collisions éventuelles pour les autres corps du robot : leurs positions auront changé, rendant par conséquent la tâche impossible ;dans ce cas il faudra reconsidérer le déplacement.

Il n'en est pas de même pour les configurations de passage : pour le programme, ces points de passage *doivent* correspondre aux configurations pour lesquelles le risque de collision est localement maximum : il n'est ainsi pas opportun de "bien viser" d'une configuration à une autre, d'exagérément "économiser" les points de passage, en risquant de frôler un obstacle entre les deux, sans avoir garanti le passage par un point déterminé.

De fait, la trajectoire entre deux poses du robot n'est pas toujours aussi rectiligne qu'on s'y attendrait, parce que le suivi de trajectoire réalisé par l'armoire de commande du robot doit tenir compte de paramètres physiques qui ne sont pas systématiquement pris en compte dans la définition de la tâche, par exemple l'inertie. Dans les zones à haut risque, situer un point de contrôle effectif est plus sûr que passer là sans un contrôle précis de la trajectoire, même si une trajectoire rectiligne semble bien passer.

Pour simplifier, nous considérons la tâche élémentaire consistant à aller d'une configuration d'arrêt P_0 à la configuration d'arrêt suivante P_p, en passant successivement par les $p-1$ configurations de passage P_l, $l \in [1, p-1]$. Elle est

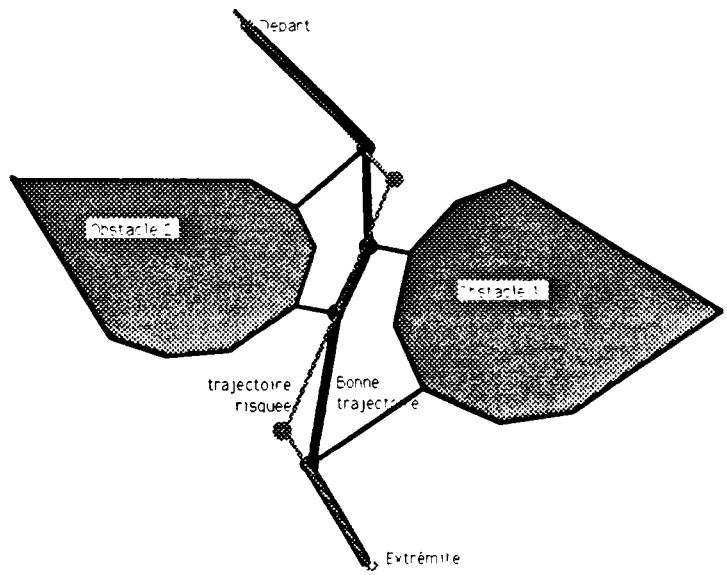


Figure 5.11 : Positionnement des points de passage : un point est mis en place aux endroits où le risque de collision est maximal.

notée par la suite :
$\mathcal{T} = (P_0, P_1, ..., P_l, ..., P_p)$

De toutes les configurations de la tâche élémentaire, seules l'initiale et la finale sont des configurations d'arrêt, toutes les autres sont des configurations de passage. Si la tâche générale du robot comporte plus de deux configurations d'arrêt, elle sera décomposée en autant de tâches élémentaires que nécéssaire.

5.4 Méthode des articulations virtuelles

Une *image physique de cette méthode*, s'il s'agit d'optimiser l'implantation d'un robot dans le plan xy du sol de l'atelier, serait de supposer qu'on ait placé le robot, mobile sur deux rails respectivement parallèles aux axes de coordonnées x et y. Ces deux rails constituent deux "articulations" supplémentaires, au sens de la robotique, qui prend ce terme dans une acception très générale, comme la représentation d'un mouvement simple entre deux corps.

Dans un tel cas, la tâche consiste à aller d'une configuration à une autre, de Q_0 à Q_f, en maintenant évidemment fixes les deux coordonnées x et y, sinon il s'agirait d'un robot mobile.

5.4.1 L'espace des configurations généralisées

Définition 29 *L'espace des configurations EC' à n' composantes généralisées est constitué de l'espace EC des n variables articulaires décrivant le mécanisme du robot réel, augmenté de l'espace EC_v des n_v variables "articulaires" dites virtuelles, décrivant les possibilités de modifications des paramètres de position ou d'efficacité du robot (l'implantation du robot notamment) potentiellement utiles pour la résolution du problème à optimiser.*

$$EC' = EC \oplus EC_v \text{ et } n' = n + n_v$$

Les configurations généralisées sont notées Q^i, i variant de 1 à n'. Les variables "virtuelles" sont notées λ^i, i variant de 1 à n_v. Les variables articulaires réelles sont notées Q^i, i variant de 1 à n.

Remarque 8 *Pendant la recherche de la trajectoire optimale correspondant à des valeurs de ces variables virtuelles, celles-ci conservent les valeurs qui leur ont été attribuées : elles correspondent en effet à la position du robot constante pour toute la trajectoire. Si elles pouvaient varier au cours de la trajectoire, elles ne se distingueraient plus des variables réelles : ceci correspondrait à l'installation physique réelle du robot sur des axes supplémentaires, ce qui est réalisé d'ailleurs dans certains cas (service de plusieurs machines par un robot monté sur un axe de translation par exemple).*

Exemple Si le robot est décrit par 2 variables Q^1 et Q^2 et son implantation par une seule variable λ, l'espace des configurations généralisées est de dimension 3. Pour décrire une tâche réelle, il faut se situer dans un plan orthogonal au troisième axe λ donné, ce qui exprime que la tâche doit être exécutée à variable virtuelle constante.

Lorsque la variable virtuelle varie, la description de la tâche doit changer de plan, tandis que les points de départ et d'arrivée décrivent des courbes dans l'espace.

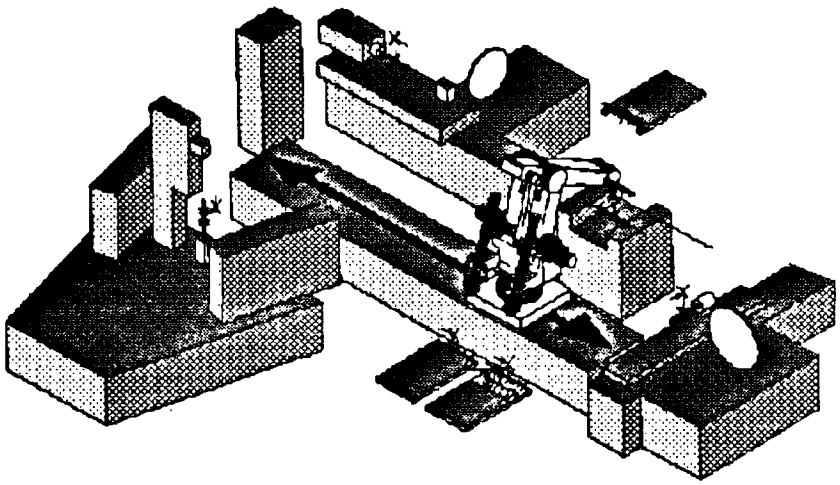


Figure 5.12 : Robot monté sur un axe supplémentaire de translation pour le service de deux meuleuses, d'un tour et de plusieurs tapis d'arrivée et de départ des produits.

5.4.2 La tâche dans l'espace des configurations généralisées

Les configurations d'arrêt y sont telles que leurs coordonnées Q_i sont devenues fonctions des coordonnées virtuelles λ : il convient alors de faire appel au modèle inverse du robot. Ce modèle donne, pour un choix du repère actif de l'outil, les variables articulaires réelles du robot en fonction :

- du repère à atteindre par l'outil,
- du repère de base du robot calculé à partir des variables articulaires virtuelles λ.

Dans l'espace des articulations virtuelles, les configurations de passage sont telles que leurs coordonnées Q_i sont libres, pourvu qu'elles respectent les contraintes engendrées par les obstacles.

Les inconnues du système définissant une telle trajectoire sont ainsi d'une part les composantes du vecteur des variables articulaires réelles du robot en chaque

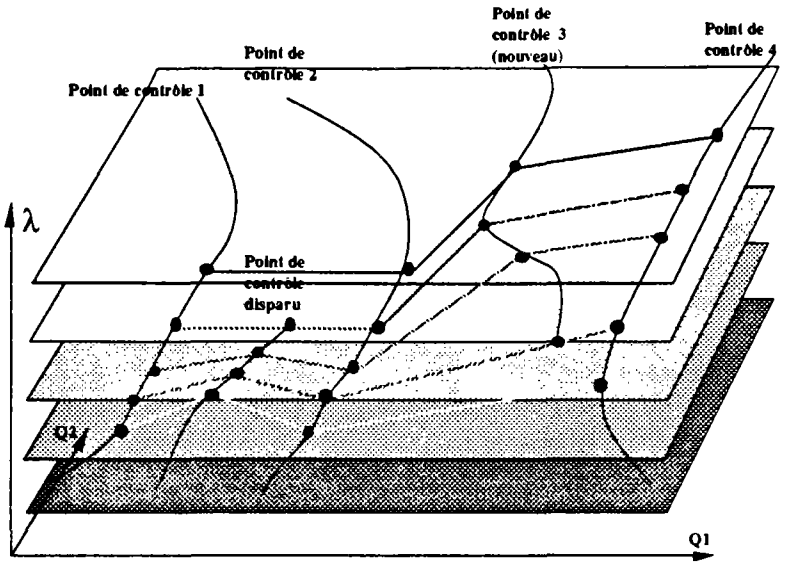


Figure 5.13 : Espace des articulations généralisées : les points de contrôle évoluent, disparaissent ou apparaissent, en fonction des variables virtuelles. La trajectoire doit rester dans des plans à variables virtuelles constantes.

configuration de contrôle, et d'autre part les variables articulaires virtuelles du robot (composantes virtuelles identiques pour tous les points de contrôle).

Le nombre de ces inconnues ν est égal au nombre n_v de variables virtuelles, plus le nombre n_p de points de contrôle multiplié par le nombre n d'axes réels du robot.

$$\nu = n_v + n\, n_p$$

La définition de la "tâche process", c'est à dire des configurations d'arrêt, dépend des coordonnées virtuelles, puisque ces configurations sont définies par une transformation dans l'espace ambiant.

$$\text{pour les configurations d'arrrêt, } Q_l = Q_l(\lambda)$$

Les configuration *de passage* restent librement évolutives dans les limites de non collision entre le robot, l'objet qu'il transporte, et l'environnement. Cette optimisation en effet tend à les faire évoluer dans un sens favorable.

5.4.3 Concept de durée d'une tâche, de "longueur" d'une trajectoire dans l'espace des configurations

Il est recherché une trajectoire optimale, grâce aux variations possibles des variables ci-dessus. La "longueur" de la trajectoire est fonction des Q_i^j, variables réelles et virtuelles décrivant la configuration du robot pour chaque point de contrôle P_i de la trajectoire polygonale.

Mais la norme euclidienne n'a pas de signification physique dans l'espace des configurations du robot, car dans cet espace il n'y a pas de distance "naturelle" comme en cinématique. Il n'y a pas davantage de dimension physique véritable pour les grandeurs considérées puisque toutes les coordonnées ne sont pas nécessairement de la même dimension physique : longueurs, angles,...

Définition 30 *Mieux que la longueur de la trajectoire, il est souhaité optimiser le* **temps d'exécution de la tâche**. *Le temps mis par le robot pour aller d'un point de contrôle à un autre est égal au temps mis par l'axe le plus chargé (celui qui se déplace au maximum de ses possibilités en vitesse et accélération). Les temps sont mis en évidence par les* **variables articulaires normalisées** *: chaque variable est divisée par sa vitesse maximale, ainsi la dimension physique de toutes ces variables articulaires normalisées est-elle le temps.*

Définition 31 "Durée" de la tâche *: Une même trajectoire peut être parcourue d'une infinité de manières par des mouvements différents de durées différentes. L'optimisation de cette durée, par le choix d'un mouvement adéquat en fonction de la dynamique du robot et de ce qu'il transporte, est un problème de constructeur du robot, d'ailleurs très complexe, que nous prenons ici comme résolu. Dans les cas habituels en résulte une loi du mouvement unique suivant une trajectoire donnée dans l'espace des configurations donnée par des algorithmes dont le résultat n'est plus à améliorer. Puisqu'il n'y a ainsi qu'un seul mouvement possible par trajectoire, la durée du mouvement ne pourra alors être réduite que par une modification de trajectoire.*

Une variation articulaire normalisée est effectuée dans un temps mesuré par la norme du maximum des valeurs absolues des variations de chacune des variables normalisées, qui correspond à la durée du mouvement pour l'axe le plus chargé si les effets des accélérations et décélérations sont faibles.

Le problème devient : minimiser la durée L dans un voisinage de la trajectoire initiale, c'est à dire rendre les variations de L les plus négatives possibles, les variables étant les variables articulaires Q_l^i pour chaque point de contrôle P_l. Ces variables dépendent des variables virtuelles λ et des obstacles. Les contraintes sont d'une part celles qui résultent des limites de la morphologie du robot, d'autre part celles résultant du domaine de validité de l'approximation par les petits déplacements.

5.4.4 Les classes d'homotopie des trajectoires

La topologie de l'espace des configurations revêt une importance particulière dans la recherche d'une solution. En effet, les trajectoires qui peuvent être obtenues en déformant continuement une trajectoire type sont celles qui appartiennent à la même classe d'homotopie dans l'espace considéré.

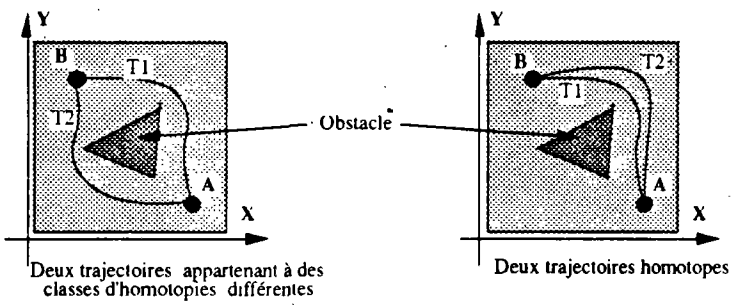


Figure 5.14 : Trajectoires non homotopes ou trajectoires homotopes (espace des configurations)

Puisque les méthodes locales d'optimisation supposent au départ une trajectoire donnée, la solution proposée ne peut qu'être homotope à cette trajectoire initiale.

L'espace libre $EL \subset EC$ dépend des coordonnées virtuelles : $EL = EL(\lambda)$. La connexité de cet espace est fonction de λ, ce qui rend les classes d'homotopie variables.

Dans ces conditions, certains obstacles peuvent disparaître ou apparaître pour certaines implantations du robot.

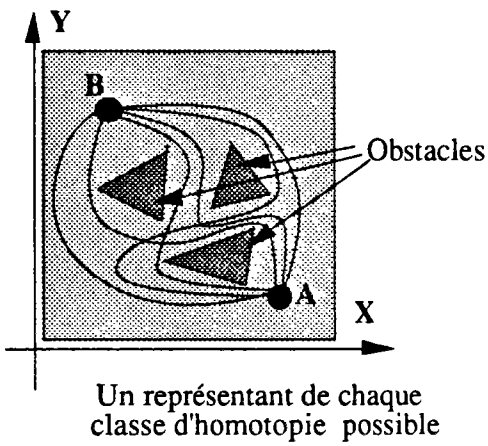


Figure 5.15 : Les classes d'homotopie de trajectoires (espace des configurations) : Les trajectoires sans boucles sont les seules représentées.

S'il est toléré à un stade particulier du processus que les articulations virtuelles de chaque configuration de la tâche varient indépendamment d'une configuration à l'autre, par exemple pour surmonter une difficulté passagère. Alors l'unification de classes d'homotopie peut être envisagée par contournement d'un obstacle dans l'espace généralisé, et non plus seulement dans un de ses sous-espaces à variables virtuelles constantes.

Lorsque des degrés de liberté virtuels sont mis en jeu, trois façons d'initialiser le processus sont possibles :

- initialisation par une trajectoire réelle "faisable" : c'est la méthode la plus sûre, mais elle impose un travail important pour obtenir cette trajectoire. Elle garantit d'obtenir un résultat.
- Initialisation par une trajectoire réelle "infaisable" : la trajectoire peut être non faisable, soit du fait de configurations hors d'atteinte. Le processus d'optimisation est alors chargé d'améliorer cette situation, s'il le peut. La garantie de converger sûrement vers une solution réellement faisable n'existe évidemment pas. Dans de telles conditions la recherche d'une trajectoire optimale doit être réalisée en deux phases :
  - construction puis optimisation d'une fonction de pénalisation d'autant plus importante qu'il y a pénétration entre objets le long

de la trajectoire, avec comme contraintes l'anti-collision entre les objets déjà disjoints : le processus correspondant tend à faire sortir la trajectoire de l'objet.

- une trajectoire sûre ayant été trouvée, optimisation de la durée de la tâche.

- initialisation par une trajectoire virtuelle "faisable", trajectoire le long de laquelle les variables virtuelles varient bien que cela soit illicite : cette méthode est envisageable si l'opérateur a réussi à trouver une bonne trajectoire, mais en déplaçant le robot (illégitimement) au cours de la trajectoire. C'est en général beaucoup plus facile de trouver une telle trajectoire, que d'en trouver une réellement faisable. L'algorithme doit alors s'efforcer d'optimiser une fonction de pénalisation d'autant plus élevée que les non-variations des variables virtuelles sont transgressées, un résultat convenable n'étant pas obtenu à coup sûr.

Parallèlement à la notion d'homotopie de deux trajectoires dans l'espace des variables articulaires réelles, il existe une notion d'homotopie pour deux trajectoires dans l'espace des variables articulaires généralisée. Le processus d'optimisation, parce qu'il est local, fait évoluer la tâche au sein d'une même classe d'homotopie de l'espace des configurations généralisées. Ce processus aboutit à une tâche optimale qu'il faut considérer ensuite dans l'espace articulaire réel. La classe d'homotopie de cette tâche finale est déterminée au cours du processus. En conséquence, il est possible de trouver la même tâche optimale alors que ce processus est initialisé par des tâches non homotopes.

Soit, par exemple, une tâche initiale consistant à monter le poste de conduite d'une voiture en passant par la baie du pare brise, le robot étant placé d'un côté de la voiture. Si la solution optimale absolue consistait à de passer par l'ouverture de la porte, le processus simple envisagé ici ne permettrait pas de trouver cette solution. Si la solution optimale absolue consistait à placer le robot d'un côté particulier d'un obstacle, l'algorithme envisagé ici pourrait permettre de trouver cette solution, s'il est possible de déplaçer le robot continuement autour de cet obstacle.

L'obtention d'une telle trajectoire pour la tâche initiale réalisable est déjà un exploit pour l'expert de la programmation des robots, tant les marges de manœuvres sont faibles. De ce fait, il peut être particulièrement intéressant de

ne pas ignorer la part d'optimisation qui existe dans la recherche de la solution initiale.

Il existe souvent en effet une part d'optimisation -plus ou moins spontanée- dans la recherche d'une trajectoire de départ : en effet, pour l'exemple ci-dessus, une trajectoire presque bonne est assez facile à trouver, grâce à l'intuition ou à la logique de l'opérateur : orientation convenable du tableau de bord dans le même sens que le pare-brise, basculement pour "chausser" le volant, engagement du tableau de bord par une de ses extrémités à travers le pare-brise. Ce sont des choix de stratégie qu'il est difficile de laisser entièrement à la charge du processus d'optimisation, en absence d'intelligence artificielle. En revanche, une fois construite une tâche simple comportant quelques collisions peu profondes, il est plus délicat pour l'opérateur d'éviter "manuellement" tous les petits chocs ici ou là : c'est bien là le rôle d'une optimisation que de remettre hors collisions une telle tâche.

Il est ainsi assez logique d'engager l'optimisation de l'implantation du robot et celle de la tâche simultanément..

Cette part d'optimisation peut prendre un deuxième aspect : en effet, la trajectoire initiale "presque bonne" est construite manuellement à partir d'une implantation a priori intangible du robot. De fait, cette implantation est "presque convenable" : au cours de la construction d'une trajectoire presque convenable, il n'est par rare de s'apercevoir que pour éviter telle ou telle collision, ou tel blocage imprévu, l'implantation aurait dû être légèrement différente. Deux stratégies sont alors possibles : soit recommencer, soit soupçonner que la modification d'implantation n'aura pas de conséquences remettant en cause la partie de trajectoire déjà établie, auquel cas il est envisageable de continuer à définir la trajectoire "presque faisable" jusqu'au but final, en réglant ensuite les rétroactions des changements d'implantation. Si ces rétroactions n'entraînent comme prévu aucune remise en cause notable, l'amélioration des données de départ peut être conduite par l'ordinateur simultanément avec l'optimisation de trajectoire et celle de l'implantation, optimisation dont il viendra à bout si la stratégie de l'opérateur était bonne et si tous les choix stratégiques au long de la trajectoire étaient corrects.

Ainsi, les classes d'homotopies à l'aboutissement de l'optimisation sont-elles le reflet de choix très difficiles pour les processus dépourvus d'intelligence artificielle.

Le processus s'avérant souvent capable d'aboutir à une optimisation correcte à partir de conditions initiales "presque bonnes", c'est à dire imparfaites, entachées ici ou là de "petites" collisions, il se trouve susceptible parfois de passer d'un côté à l'autre d'un obstacle suffisamment étroit.

5.5 Les outils d'étude et de contrôle

5.5.1 Processus général avec itérations

5.5.1.1 Contrôle des distances cartésiennes au cours du processus

Ce contrôle est réalisé comme dans la **méthode des contraintes** (chapitre 3). Il permet au cours de l'optimisation, de refuser les petits déplacements des variables virtuelles qui entraîneraient une collision locale au voisinage des noeuds de la trajectoire. Apparaît ici manifestement l'importance primordiale de la construction initiale de la trajectoire par ses points de contrôle : il ne faut pas qu'il y ait de risque de collision "proche" (c'est à dire pour de petits déplacements) ailleurs qu'au voisinage des points de contrôle.

Etude en un point de contrôle

Comme dans la méthode des contraintes, ce calcul conduit à exprimer la contrainte de non collision dans l'espace des configurations : il y a autant d'inéquations que de couples de solides en interaction.

Pour une configuration de contrôle P_l, la distance entre le corps du robot et l'obstacle ne doit pas diminuer, pour le "petit déplacement", de plus que leur distance au pas précédent, ceci en considérant l'ensemble des paramètres réels et virtuels, ce qui donne en explicitant la variation des paramètres réels Q_l et celle des paramètres virtuels λ dans la variation des configurations généralisée $\Delta\mathcal{Q}$:

$$\overrightarrow{grad_{Q_l}}D_l \,.\, \overrightarrow{\Delta Q_l} + \overrightarrow{grad_{\lambda}}D_l \,.\, \overrightarrow{\Delta\lambda} + K_d(D_l - D_S)\delta t \geq 0$$

$$\overrightarrow{grad}D_l \,.\, \overrightarrow{\Delta Q_1} + K_d(D - D_S)\delta t \geq 0$$

$$\text{avec } \overrightarrow{grad}D_l = \mathcal{J}^{\sqcup}\overrightarrow{N_l}$$

$0 \geq K_d \geq 1$ est un scalaire permettant de contrôler la convergence vers le contact : si $K_d = 1$, l'algorithme est autorisé à aller au contact en une seule itération, ce qui est dangereux à cause des imprécisions dûes à la linéarisation des contraintes. Si K_d est nul, le robot ne se rapproche pas du tout de l'obstacle.

$\overrightarrow{gradD}$ est le gradient de D généralisé aux variables réelles et virtuelles. Les paramètres virtuels interviennent dans la matrice jacobienne généralisée $\mathcal{J}$ dans laquelle ils ajoutent à la jacobienne réelle J autant de colonnes que de paramètres virtuels. Ils interviennent aussi dans le vecteur $\overrightarrow{\Delta Q}$ qui représente la variation des paramètres réels et virtuels.

Si le robot est légèrement déplacé dans l'atelier, tout se passe dans l'espace ambiant comme si les obstacles étaient en fait déplacés par rapport au robot, tout comme les repères qu'il devait atteindre impérativement. Les composantes de la matrice jacobienne qui correspondent aux variations des variables virtuelles sont ainsi simples à calculer.

Ceci est à considérer pour chaque couple (corps $\mathcal{C}_R$ du robot, corps $\mathcal{C}_E$ de l'environnement) susceptible d'entraîner une déviation sur la trajectoire (configurations de contrôles). Ce processus augmente la dimension des jacobiennes d'une ligne supplémentaire par variable virtuelle, ce qui alourdit un peu les calculs correspondant, mais n'intervient pas, bien sûr, sur le nombre ou sur les caractéristiques de ces contraintes.

5.5.1.2 Contrôle des distances cartésiennes après une minimisation de la durée de la tâche dans l'espace des configurations

Cette étape est sensiblement différente de la précédente : ici, il s'agit de "remettre à jour " la trajectoire, en déplaçant certains points de contrôle, en supprimant certains ou en ajoutant d'autres, suivant la proximité des obstacles.

A partir d'une trajectoire polygonale obtenue par déformations continues depuis une précédente trajectoire, déformations continues ayant conservé les points de contrôle précédents en modifiant leur position, il s'agit d'expliciter un algorithme qui replace les points de contrôle à leur meilleur emplacement, c'est à dire au lieu le plus critique, celui pour lequel la distance cartésienne entre les deux corps est vraiment localement minimale.

Le déplacement des points d'arrêt, réalisé de manière à ce que l'outil du robot atteigne le même point que précédemment ne suffit pas à garantir la sûreté de la trajectoire pour d'autres corps que ceux qui sont fixes ou liés au dernier axe du robot ; par ailleurs, pour une trajectoire optimale, le programme a modifié les configurations de passage dans le sens qui raccourcit le mieux la durée. L'algorithme proposé devra corriger les configurations de passage, de façon qu'elles permettent encore de contrôler efficacement les distances (voir section 5.3.4).

L'algorithme proposé doit aussi détecter entre deux points de contrôle, l'apparition d'un obstacle qui, peu dangereux auparavant, le devient après réalisation des petits déplacements. Dans ce cas, il faut ajouter un point de contrôle en ce nouveau lieu critique.

Enfin, si l'optimisation se déroule bien, la trajectoire doit se "tendre" ; certains segments s'alignent : le point de contrôle correspondant s'il n'est plus ainsi critique, doit être supprimé.

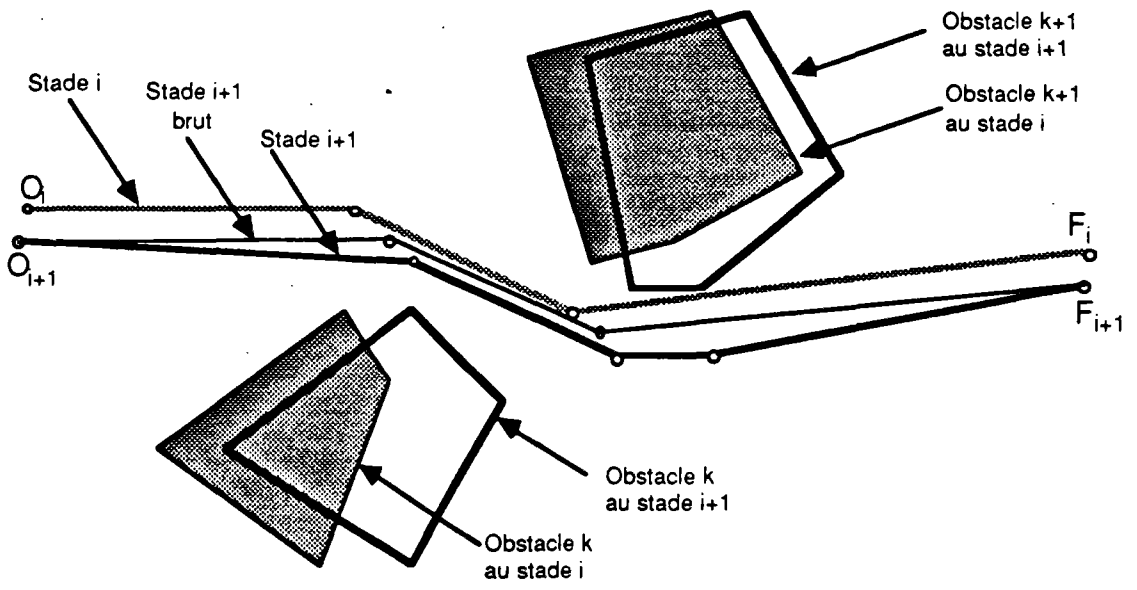


Figure 5.16 : Ancienne et nouvelle trajectoire (avec la trajectoire intermédiaire), obtenue après une itération complète de l'algorithme.

La figure 5.16montre la trajectoire brute obtenue après optimisation, dans laquelle les points de passage ne sont plus très bien placés aux minimums

de distances aux obstacles, puis la trajectoire corrigé par un lissage et un repositionnement correct des points de passage. Cette dernière trajectoire sert de départ à l'itération suivante.

L'algorithme parcourt la trajectoire d'une configuration d'arrêt à la suivante. Il suit entre points de contrôle une droite dans l'espace des configurations ; il contrôle les distances dans l'espace ambiant, à chaque pas et crée un point de contrôle chaque fois que la distance passe par un minimum inférieur à une distance donnée à l'avance. Le parcours est réalisé à l'aide de l'algorithme de la **méthode des contraintes**. Ainsi, si la distance entre éléments a diminué jusqu'à atteindre la distance de sécurité, le mécanisme de contournement d'obstacle fonctionne en créant des points de contrôle supplémentaires.

Les courbures inutiles qu'il formerait éventuellement auront à être éliminées au pas suivant de la minimisation. Ce processus de reconstruction de la trajectoire pour replacer les points de contrôle aux bons endroits risque d'allonger la trajectoire, pour contourner un nouvel obstacle. Aussi le test d'arrêt de l'implantation optimale doit-il se baser sur la longueur de la trajectoire **avant** cette réorganisation : les itérations suivantes rectifient en les optimisant les trajectoires réorganisées qui suivraient peut être trop fidèlement les obstacles.

5.5.2 Implantation des points de contrôle par la méthode des contraintes

Définition 32 Distance de sécurité stricte *: c'est dans l'espace ambiant la distance par rapport à un obstacle, en-deçà de laquelle les organes du robot et les pièces manipulées ne doivent pas se trouver.*

Définition 33 Distance d'approche large *: c'est dans l'espace ambiant la distance par rapport à un obstacle, supérieure à la distance de sécurité stricte, au-delà de laquelle il est souhaitable de tenir éloignés les organes du robot et les pièces manipulées, tout particulièrement aux points de contrôle.*

Cette distance d'approche large sera réduite en fin d'optimisation pour permettre une trajectoire plus tendue, mais elle empêche au début de l'optimisation de suivre de trop près les obstacles (inutilement et à grands frais).

Implantation des points de contrôle : entre deux configurations de passage de la tâche élémentaire $\mathcal{T}$, cette méthode décrit une trajectoire standard, par exemple une droite dans l'espace des configurations : avançant à pas de temps constants, elle place un point de passage à chaque fois. Lorsque la trajectoire ainsi décrite impose aux corps de se rapprocher en deçà d'une certaine distance d'influence, la trajectoire est déviée, les corps ne pouvant se rapprocher à une distance moindre que la distance de sécurité imposée à l'avance. En présence d'obstacle, la trajectoire ainsi déviée de proche en proche, converge vers la configuration d'arrivée de la tâche élémentaire en cours de traitement.

Cette mise en conformité des configurations de contrôle est réalisée dans deux cas très différents : la première fois, à partir de la trajectoire au départ fournie par l'utilisateur, les autres fois en cours de traitement par l'algorithme général, à partir de tâches obtenues par de petits déplacements successifs.

Utilisation en cours de traitement

Les cas d'échec de cette méthode sont ceux des méthodes locales de génération de trajectoire. Pour l'utilisation envisagée ici, la trajectoire douteuse, servant de point de départ à cette méthode, a été obtenue par de petits déplacements autour d'une trajectoire sûre, avec des contraintes d'anticollisions linéarisées autour de chaque configuration de passage. S'il y a collision, il est ainsi raisonnable de penser qu'il s'en faut de peu, et qu'une déviation minime y remédiera. Ainsi une telle méthode locale devrait-elle réussir dans la quasi-totalité des cas.

En première utilisation pour la trajectoire de départ

La trajectoire de départ n'offre pas les mêmes garanties de collisions minimes qu'une trajectoire certes douteuse, mais issue d'une trajectoire possible. L'opérateur peut avoir donné une trajectoire avec des collisions importantes. Dans ce cas, si l'intuition de l'opérateur n'avait pas été convenable, l'algorithme pourrait échouer, ne trouvant pas de trajectoire qui aboutisse au but recherché.

En cours de fonctionnement

Cette méthode met en place un grand nombre de configurations intermédiaires. Cependant en cas d'alignement, il est facile d'éliminer les configurations intermédiaires qui sont ainsi devenues inutiles. Un traitement de lissage est aussi nécessaire, la trajectoire obtenue présentant en effet généralement des courbes excessives induites par le contournement des obstacles. Ces courbes, si elles

subsistaient, ne gêneraient pas l'algorithme, qui saurait les redresser petit à petit par des petits déplacements optimaux, mais ralentiraient inutilement le calcul.

5.5.3 Lissage rapide

Les processus tendant à contourner les obstacles ont tendance à engendrer de nouveaux points de contrôle. Il faut en éviter la prolifération ; c'est pourquoi un processus de lissage simple et même sommaire est bénéfique, en remplaçant une ligne brisée par une ligne droite, ce même s'il n'aboutit pas à la véritable longueur minimale pour la situation considérée.

Méthode simplifiée mais rapide

Un tel processus peut consister à considérer dans l'espace des configurations. les segments de droite de Q_0 jusqu'aux points de passage successifs, soient Q_0Q_1, Q_0Q_2, Q_0Q_3... jusqu'à ce que l'un d'entre eux Q_0Q_m porte un ou plusieurs points qui dans l'espace ambiant, interfèrent avec un volume d'approche large, un volume de sécurité majeure, ou avec un obstacle ; il est alors possible de remplacer la ligne brisée $Q_0Q_1Q_2...Q_{m-1}$ par le segment de droite Q_0Q_{m-1}, le dernier dont aucun des points ne correspondait à une interférence ; ce segment Q_0Q_{m-1} devient Q_0Q_1.

En continuant à partir de ce nouveau Q_1 de la même manière la ligne brisée $Q_1Q_2...Q_{m'-1}$ sera remplacée par le segment de droite $Q_1Q_{m'-1}$ qui deviendra Q_1Q_2.

Et ainsi de suite à partir de ce nouveau Q_2 jusqu'à arriver en Q_f.

Ce processus n'aboutit pas nécessairement au véritable optimum car cet optimum peut, si le volume de sécurité majeure se présente très obliquement par rapport au plan (Q_O, Q_{m-1}, Q_m), avoir à considérer des points de passage très sensiblement écartés de ce plan. Mais il est simple à mettre en oeuvre tout en étant capable d'éviter la prolifération de points de contrôle engendrés par un contournement très "courbé".

En considérant ensuite sur chacun des nouveaux segments, les points représentant les configurations en lesquelles la distance cartésienne avec

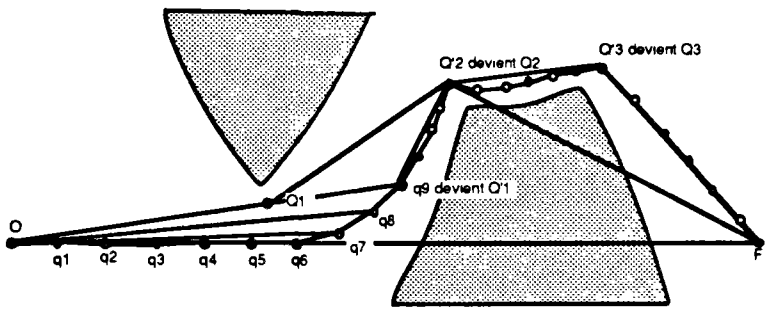


Figure 5.17 : Lissage d'une trajectoire

l'obstacle est la plus courte, une meilleure trajectoire peut être obtenue. Ces points peuvent en effet devenir points de contrôle à partir desquels le processus de lissage peut à nouveau être entrepris.

Par ce procédé, une trajectoire convenable $\mathcal{T}$, allant de Q_{p-1} à Q_p pour l'état λ des variables virtuelles se trouve ainsi établie. La longueur de la trajectoire est $L = L(\lambda, Q_p)$.

5.6 Processus complet

Soit un robot possèdant n articulations.
Soit une tâche $\mathcal{T} = (P_0, P_1, \cdots, P_l, \cdots, P_{p-1}, P_p)$ à réaliser par ce robot. P_0 et P_p sont des points d'arrêt, $P_1, \cdots, P_l, \cdots, P_{p-1}$ sont des points de passage. Il y a un point de passage à chaque fois que la distance minimale du système robot-obstacles passe par un minimum. Dans l'espace des configurations, la tâche $\mathcal{T}$ est exprimée par $(Q_0^i, Q_1^i, \cdots, Q_l^i, \cdots, Q_{p-1}^i, Q_p^i)$, L'indice $i \in [1, n]$ correspondant à la i-ème articulation du robot.

5.6.1 Initialisation

Soit $\overrightarrow{\lambda}$ une implantation définie du robot, soit $\mathcal{T}$ une trajectoire définie. Cette trajectoire n'est pas nécessairement optimale ni même faisable.

Une première vérification est menée : la tâche est exécutée, la trajectoire entre chaque point d'arrêt ou de passage y étant assurée par la méthode des contraintes. Elle permet :

- de vérifier l'absence de collisions au cours de l'exécution de la tâche ;
- de vérifier que chaque fois que la distance entre les différents corps en présence passe par un minimum, une configuration de passage soit en place.

Dans les cas où cette vérification détecte une anomalie par rapport à cette situation, elle déclenche les actions suivantes :

- en cas de collision : modification, par la méthode des contraintes, du parcours de la trajectoire. Ceci permet de tolérer des collisions au sein de la tâche d'origine, à condition que la méthode des contraintes sache en déduire une trajectoire sûre ;
- mise en place de points de passage aux lieux "de distances minimales" : ceci permet des tâches de départ quelconques. Il est en effet possible de placer des configurations de passage aux lieux où elles sont certainement nécessaires, par la méthode décrite plus haut pour l'implantation des points de contrôle ;
- suppression des points de passage inutiles par la méthode de lissage décrite plus haut : ceci permet d'accepter une tâche initiale comportant de nombreux points de passage.
  - ces trop nombreux points de passage peuvent être le reflet des difficultés qui ont pu être rencontrées pour la définition de cette tâche initiale,
  - ils peuvent aussi être la conséquence d'une correction de trajectoire à cause d'une collision intervenant sur cette trajectoire ; en effet le processus de contournement des obstacles, même par la méthode des contraintes qui est employée, crée un nombre considérable, souvent excessif, de points de passages dans la zone perturbée par l'évitement des obstacles.

5.6.2 Construction de la nouvelle trajectoire

La trajectoire initiale ainsi formée sert de point de départ à un processus itératif décomposé en deux étapes.

- **Première étape :** approximation par un modèle valide pour de petits déplacements.

  Le processus d'optimisation itératif employé est une optimisation avec contraintes par la méthode de la tolérance flexible (si on souhaite que l'optimisation accepte des collisions dans la tâche initiale et fasse sortir progressivement des obstacles sans modification de la fonction d'objectif) ou bien un algorithme standard en bibliothèque.

  Ceci permet en général d'établir une nouvelle trajectoire meilleure en ce qui concerne sa durée, par de petits déplacements des variables réelles et virtuelles définissant la trajectoire. Cette optimisation est formalisée dans la section suivante.

  Dans ces conditions aucun des nouveaux points Q_l, $l \in [1,(p-1)]$ n'est en position de collision, mais, à cause des approximations linéaires effectuées, il n'est pas impossible que des configurations sur les segments $Q_0Q_1, Q_lQ_{l+1}, Q_{p-1}Q_p$ soient telles que les obstacles s'y révèlent plus proches qu'aux configurations de passage voisines. Il est ainsi indispensable de remettre à jour le modèle, en parcourant la trajectoire à pas de temps constant.

- **Deuxième étape :** remise à jour du modèle à partir du résultat de cette approximation, par un processus identique à celui utilisé pour traiter la tâche initiale fournie par l'opérateur. Il doit remettre aux normes la trajectoire trouvée, afin de replacer les points de passage aux lieux de distance minimale.

Pour de petits déplacements du robot **les points de passage évoluent de façon continue**, hormis si un **changement de nature** du problème apparaît. La nature du problème change si quelque chose d'important vient de se passer :

- l'apparition ou la disparition d'un point de contrôle.
- le changement d'axe le plus chargé

Ces "changements brusques" ne sont pas pris en compte dans le processus d'optimisation dans la limite des petits déplacements, mais dans le processus d'étalonnage (deuxième étape).

Le lissage réalisé, le processus d'optimisation par petits déplacements est réitéré.

5.6.3 Formalisation du processus d'optimisation par les petits déplacements

Minimisation avec contraintes

L'étude est menée dans l'espace des configurations généralisées.

Un processus itératif doit tendre à diminuer la durée L de la tâche dans l'espace des configurations généralisées, exprimée en fonction de l'axe le plus chargé pour chaque segment de la tâche. Cet objectif est réalisé en cherchant à rendre la variation δL de cette durée la plus négative possible :

$$L = \sum_{l=1}^{p} \max_{i=1}^{n} |Q_l^i - Q_{l-1}^i|$$

$$L + \delta L = \sum_{l=1}^{p} \max_{i=1}^{n} |Q_l^i - Q_{l-1}^i + \delta Q_l^i - \delta Q_{l-1}^i|$$

$$\delta L = \sum_{l=1}^{p} [\max_{i=1}^{n} |Q_l^i - Q_{l-1}^i + \delta Q_l^i - \delta Q_{l-1}^i| - \max_{i=1}^{n} |Q_l^i - Q_{l-1}^i|]$$

Les variations δQ restent petites par rapport à la distance entre deux points de contrôle. Ceci permet de supposer que le maximum de $|Q_l^i - Q_{l-1}^i + \delta Q_l^i - \delta Q_{l-1}^i|$ et de $|Q_l^i - Q_{l-1}^i|$ est atteint pour la même composante $i(l)$, cette composante étant différente, dans le cas général, pour chaque segment de trajectoire l.

Puisqu'il s'agit de petits déplacements, cette simplification est valide. En effet, si au cours d'un petit déplacement, l'articulation du robot qui est la plus chargée peut changer, l'erreur est faible, dans la logique de l'approximation des petits déplacements. Elle n'empêche pas de changer d'articulation la plus chargée au cours du processus d'optimisation envisagé, puisque après une minimisation limitée aux petits déplacements, une correction des dérives du modèle est effectuée : l'optimisation par les petits déplacements à l'itération suivante se fera sur ce nouveau modèle, avec les axes les plus chargés correspondants.

Ainsi :

$$\text{si } i(l) \text{ maximise } |Q_l^i - Q_{l-1}^i|, i \in [1, \cdots, n]$$

$$\delta L = \sum_{l=1}^{p} (|Q_l^{i(l)} - Q_{l-1}^{i(l)} + \delta Q_l^{i(l)} - \delta Q_{l-1}^{i(l)}| - |Q_l^{i(l)} - Q_{l-1}^{i(l)}|)$$

Si on note

$$\varepsilon_l = Sign\epsilon(Q_l^{i(l)} - Q_{l-1}^{i(l)})$$

alors

$$\delta L = \sum_{l=1}^{p} \varepsilon_l(\delta Q_l^{i(l)} - \delta Q_{l-1}^{i(l)})$$

Cette fonction δL constitue la fonction d'objectif de l'optimisation envisagée, qu'il faut à chaque itération rendre la plus négative possible.

Les variables du système sont :

- les $\delta Q_l^{i(l)}, \delta Q_l^{i(l+1)}$ correspondant à chacune des configurations de passage ou d'arrêt Q_l, $l \in [0, p]$,
- les $\delta\lambda_j, j \in 1, n_v$, n_v représentant le nombre de variables d'implantation,
- les $\delta Q_0^i, \delta Q_p^i$, $i \in [1, n]$ correspondant aux deux configurations d'arrêt Q_0 et Q_p.

Ces deux derniers types de variables n'apparaissent pas dans l'expression de la fonction d'objectif, mais dans les contraintes.

$$
\textit{variables} : \left[\begin{array}{ll}
\begin{pmatrix} \delta Q_0^1 \\ \vdots \\ \delta Q_0^i \\ \vdots \\ \delta Q_0^n \end{pmatrix} & 1^{er} \textit{ point d'arret} \\
\begin{pmatrix} \delta Q_1^{i(1)} \\ \delta Q_1^{i(2)} \end{pmatrix} & 1^{er} \textit{ point de passage} \\
\vdots & \\
\begin{pmatrix} \delta Q_l^{i(l)} \\ \delta Q_l^{i(l+1)} \end{pmatrix} & l^{eme} \textit{ point de passage} \\
\vdots & \\
\begin{pmatrix} \delta Q_{p-1}^{i(p-1)} \\ \delta Q_{p-1}^{i(p)} \end{pmatrix} & \textit{dernier point de passage} \\
\begin{pmatrix} \delta Q_p^1 \\ \vdots \\ \delta Q_p^i \\ \vdots \\ \delta Q_p^n \end{pmatrix} & \textit{dernier point d'arret} \\
\begin{pmatrix} \delta \lambda_1 \\ \vdots \\ \delta \lambda_j \\ \vdots \\ \delta \lambda_{n_v} \end{pmatrix} & \textit{variables d'implantation}
\end{array} \right]
$$

soit moins de $2(p-1)+2n+n_v$ variables.

Les contraintes de l'optimisation :

- Pour chacune des $l \in [0,p]$ configurations de passage ou d'arrêt de la trajectoire, le couple d'éléments en interaction qui détermine la distance minimale est considéré. Il représente une contrainte qui s'écrit de la même façon que dans la méthode des contraintes, en considérant à la fois les variations des variables réelles et virtuelles dans un même vecteur $\overrightarrow{\delta Q}$, le robot étant considéré avec ses axes virtuels :

$$\overrightarrow{grad} D_l \,.\, \overrightarrow{\delta Q_l} + K_d(D_l - D_S)\delta t \geq 0$$

avec

$$\overrightarrow{grad}D_l = \mathcal{J}_l^t \overrightarrow{N_l}$$

- $0 \geq K_d \geq 1$ est un scalaire contrôlant la rapidité avec laquelle l'algorithme rapproche les points de passage vers les obstacles,
- δt est l'incrément de temps utilisé,
- D_l est la distance entre les deux objets pour l'interraction au point de passage l,
- D_S est la distance minimale de sécurité à respecter entre deux corps.

Pour chaque configuration de passage ou d'arrêt, il y a une inéquation correspondant au couple d'objets en interaction réalisant la distance minimale.

dans cette inéquation, pour les points de passage :

$$\overrightarrow{\delta Q_l} = \begin{pmatrix} \delta Q_l^{i(l)} \\ \delta Q_l^{i(l+1)} \\ \delta\lambda_1 \\ \vdots \\ \delta\lambda_j \\ \vdots \\ \delta\lambda_{n_v} \end{pmatrix} \quad \text{et} \quad \mathcal{J}_l^t = \begin{pmatrix} \frac{\partial N_l^x}{\partial Q_l^{i(l)}} & \frac{\partial N_l^y}{\partial Q_l^{i(l)}} & \frac{\partial N_l^z}{\partial Q_l^{i(l)}} \\ \frac{\partial N_l^x}{\partial Q_l^{i(l+1)}} & \frac{\partial N_l^y}{\partial Q_l^{i(l+1)}} & \frac{\partial N_l^z}{\partial Q_l^{i(l+1)}} \\ \frac{\partial N_l^x}{\partial \lambda_1} & \frac{\partial N_l^y}{\partial \lambda_1} & \frac{\partial N_l^z}{\partial \lambda_1} \\ \cdots & \cdots & \cdots \\ \frac{\partial N_l^x}{\partial \lambda_j} & \frac{\partial N_l^y}{\partial \lambda_j} & \frac{\partial N_l^z}{\partial \lambda_j} \\ \cdots & \cdots & \cdots \\ \frac{\partial N_l^x}{\partial \lambda_{n_v}} & \frac{\partial N_l^y}{\partial \lambda_{n_v}} & \frac{\partial N_l^z}{\partial \lambda_{n_v}} \end{pmatrix}$$

dans cette inéquation, pour les points d'arrêt :

$$\overrightarrow{\delta Q_0} = \begin{pmatrix} \delta Q_0^1 \\ \vdots \\ \delta Q_0^i \\ \vdots \\ \delta Q_0^n \\ \delta\lambda_1 \\ \vdots \\ \delta\lambda_j \\ \vdots \\ \delta\lambda_{n_v} \end{pmatrix} \quad \text{et} \quad \mathcal{J}_l^t = \begin{pmatrix} \frac{\partial N_0^x}{\partial Q_0^1} & \frac{\partial N_0^y}{\partial Q_0^1} & \frac{\partial N_0^z}{\partial Q_0^1} \\ \cdots & \cdots & \cdots \\ \frac{\partial N_0^x}{\partial Q_0^i} & \frac{\partial N_0^y}{\partial Q_0^i} & \frac{\partial N_0^z}{\partial Q_0^i} \\ \cdots & \cdots & \cdots \\ \frac{\partial N_0^x}{\partial Q_0^n} & \frac{\partial N_0^y}{\partial Q_0^n} & \frac{\partial N_0^z}{\partial Q_0^n} \\ \cdots & \cdots & \cdots \\ \frac{\partial N_0^x}{\partial \lambda_1} & \frac{\partial N_0^y}{\partial \lambda_1} & \frac{\partial N_0^z}{\partial \lambda_1} \\ \cdots & \cdots & \cdots \\ \frac{\partial N_0^x}{\partial \lambda_j} & \frac{\partial N_0^y}{\partial \lambda_j} & \frac{\partial N_0^z}{\partial \lambda_j} \\ \cdots & \cdots & \cdots \\ \frac{\partial N_0^x}{\partial \lambda_{n_v}} & \frac{\partial N_0^y}{\partial \lambda_{n_v}} & \frac{\partial N_0^z}{\partial \lambda_{n_v}} \end{pmatrix}$$

$$\overrightarrow{\delta Q_p} = \begin{pmatrix} \delta Q_p^1 \\ \vdots \\ \delta Q_p^i \\ \vdots \\ \delta Q_p^n \\ \delta\lambda_1 \\ \vdots \\ \delta\lambda_j \\ \vdots \\ \delta\lambda_{n_v} \end{pmatrix} \quad \text{et} \quad \mathcal{J}_p^t = \begin{pmatrix} \frac{\partial N_p^x}{\partial Q_p^1} & \frac{\partial N_p^y}{\partial Q_p^1} & \frac{\partial N_p^z}{\partial Q_p^1} \\ \cdots & \cdots & \cdots \\ \frac{\partial N_p^x}{\partial Q_p^i} & \frac{\partial N_p^y}{\partial Q_p^i} & \frac{\partial N_p^z}{\partial Q_p^i} \\ \cdots & \cdots & \cdots \\ \frac{\partial N_p^x}{\partial Q_p^n} & \frac{\partial N_p^y}{\partial Q_p^n} & \frac{\partial N_p^z}{\partial Q_p^n} \\ \cdots & \cdots & \cdots \\ \frac{\partial N_p^x}{\partial \lambda_1} & \frac{\partial N_p^y}{\partial \lambda_1} & \frac{\partial N_p^z}{\partial \lambda_1} \\ \cdots & \cdots & \cdots \\ \frac{\partial N_p^x}{\partial \lambda_j} & \frac{\partial N_p^y}{\partial \lambda_j} & \frac{\partial N_p^z}{\partial \lambda_j} \\ \cdots & \cdots & \cdots \\ \frac{\partial N_p^x}{\partial \lambda_{n_v}} & \frac{\partial N_p^y}{\partial \lambda_{n_v}} & \frac{\partial N_p^z}{\partial \lambda_{n_v}} \end{pmatrix}$$

- Les contraintes des butées mécaniques aux points de passage : ce sont des contraintes intervalles qui garantissent que toutes les configurations de la trajectoire vont rester dans l'espace accessible. Elles ne concernent que les $\delta Q_l^{i(l)}$ et $\delta Q_l^{i(l+1)}$, $l \in [1, p-1]$:

$$\begin{aligned} Q_{min}^{i(l)} \leq Q_l^{i(l)} + \delta Q_l^{i(l)} \leq Q_{max}^{i(l)} \\ Q_{min}^{i(l+1)} \leq Q_l^{i(l+1)} + \delta Q_l^{i(l+1)} \leq Q_{max}^{i(l+1)} \end{aligned}$$

- Les contraintes de fermeture de la chaîne cinématique du robot aux points d'arrêt :

Remarque 9 *Le problème des points d'arrêt : en ces points l'organe terminal du robot doit être dans une position et une orientation déterminées,alors que la base du robot doit tre fixée par les paramètres d'implantation. Il faut donc exprimer la fermeture de la chaîne cinématique du robot. Ceci peut être réalisé de deux manières :*

- *soit en exprimant* $\overrightarrow{\delta Q_0}$ *et* $\overrightarrow{\delta Q_p}$ *en fonction des* $\overrightarrow{\delta\lambda}$*, ce qui ferait intervenir des pseudo inverses exprimant la redondance, et permet de ne pas consideérer* $Q_0^{i(1)}$ *et* $\delta Q_p^{i(p)}$ *comme de vraies variables,*
- *soit plus simplement sous la forme de contraintes d'égalité liant les* $\overrightarrow{\delta Q_0}$ *et* $\overrightarrow{\delta Q_p}$ *aux* $\overrightarrow{\delta\lambda}$*.*

Une notion de jacobienne dans la relation $\phi : \ q \mapsto \lambda$ est utilisée en supposant que l'outil est fixe, en position et en orientation, sur le repère $\mathcal{R}_0$ pour le premier point d'arrêt et sur $\mathcal{R}_p$ pour le dernier, d'ou les $2n_v$ contraintes :

$$\overrightarrow{\delta\lambda} = J_{\phi_{\mathcal{R}_0}} \overrightarrow{\delta Q_0},$$
$$\overrightarrow{\delta\lambda} = J_{\phi_{\mathcal{R}_p}} \overrightarrow{\delta Q_p},$$

si la cinématique en $\overrightarrow{Q_0}$ et $\overrightarrow{Q_p}$ est donnée par des relations :

$$\overrightarrow{\lambda} = \overrightarrow{\phi_{\mathcal{R}_0}}(\overrightarrow{Q_0}),$$
$$\overrightarrow{\lambda} = \overrightarrow{\phi_{\mathcal{R}_p}}(\overrightarrow{Q_p}).$$

- Les contraintes de butées mécaniques pour les points d'arrêt : pour chaque point d'arrêt, il faut le même type de garantie que pour les points de passage. Cependant, puisque les contraintes de fermeture de la chaine cinématique du robot mettent en jeu toutes les variables articulaires du robot, les contraintes de butées mécanique sont à considérer pour chacune des n variables articulaires décrivant la chaine cinématique du robot aux points d'arrets Q_0 et Q_p :

$$Q^i_{min} \leq Q^i_0 + \delta Q^i_0 \leq Q^i_{max},\ i \in [1,n]$$
$$Q^i_{min} \leq Q^i_p + \delta Q^i_p \leq Q^i_{max},\ i \in [1,n]$$

Le nombre de contraintes est ainsi au plus de $p+1$ contraintes d'inégalité plus $2(p-1)+2n$ contrantes intervalles plus $2n_v$ contraintes d'égalité.

Formalisation par l'introduction de pseudo inverses

Lors de l'étude de la fermeture de la chaine cinématique du robot entre sa base et l'organe terminal tous les deux fixés dans l'espace ambiant, nous avons préféré exprimer cette fermeture sous la forme de contraintes d'égalité liant les Q_0 et δQ_p aux $\overrightarrow{\delta\lambda}$. En utilisant une notion de pseudo inverse à droite des relations $\phi_{\mathcal{R}_0}$ et $\phi_{\mathcal{R}_p}$, on obtient :

$$\overrightarrow{\delta Q_0} = J^{-1+d}_{\phi_{\mathcal{R}_0}} \overrightarrow{\delta\lambda} = J^t_{\phi_{\mathcal{R}_0}} \overrightarrow{\delta\lambda},$$
$$\overrightarrow{\delta Q_p} = J^{-1+d}_{\phi_{\mathcal{R}_p}} \overrightarrow{\delta\lambda} = J^t_{\phi_{\mathcal{R}_p}} \overrightarrow{\delta\lambda},$$

En remplaçant $Q_0^{i(1)}$ et $\delta Q_p^{i(p)}$ par leurs expressions dépendant des variables virtuelles $\delta\lambda^j$, l'expression finale de la fonction d'objectif devient :

$$\begin{array}{c} \delta L = \varepsilon_1(\delta Q_1^{i(1)} - \delta Q_0^{i(1)}) \\ + \sum_{l=2}^{(p-1)} \varepsilon_l(\delta Q_l^{i(l)} - \delta Q_{l-1}^{i(l)}) \\ + \varepsilon_p(\delta Q_p^{i(p)} - \delta Q_{p-1}^{i(p)}) \end{array}$$

avec

$$\delta Q_0^{i(1)} = \sum_{j=1}^{n_v} \mathbf{j}^{(i(p),j)}_{\phi_{\mathcal{R}_0}} \delta\lambda_j$$
$$\delta Q_p^{i(p)} = \sum_{j=1}^{n_v} \mathbf{j}^{(i(p),j)}_{\phi_{\mathcal{R}_p}} \delta\lambda_j$$

Il y a alors moins de $2*(p-1)+n_v$ variables d'optimisation.

Les contraintes d'égalité, liées à la fermeture de la chaîne cinématique, n'ont plus lieu d'être puisqu'elle est déjà prise en compte.

Les contraintes de butées mécaniques aux deux points d'arrêt deviennent des contraintes sur les $\delta\lambda_j$.

Les contraintes d'anticollision aux points de passage restent identiques, tandis que celles aux points d'arrêt s'écrivent de manière équivalente. Cependant, pour les points d'arrets, le vecteur des variables concernées n'est plus formé que des variables virtuelles $\delta\lambda_j$, ce qui diminue de $2n$ le nombre de contraintes et diminue très sensiblement le matrice $\mathcal{J}^t$. En revanche, chaque dérivées partielles (à déterminer selont le robot) $\frac{\partial N_0^x}{\partial \lambda_j}$ doit tenir compte de l'équation de fermeture de la chaine cinématique du robot, ce qui peut être assez compliqué en l'absence d'approximation :

$$\overrightarrow{\delta Q_0} = \begin{pmatrix} \delta\lambda_1 \\ \vdots \\ \delta\lambda_j \\ \vdots \\ \delta\lambda_{n_v} \end{pmatrix} \quad \text{et} \quad \mathcal{J}_0^t = \begin{pmatrix} \frac{\partial N_0^x}{\partial \lambda_1} & \frac{\partial N_0^y}{\partial \lambda_1} & \frac{\partial N_0^z}{\partial \lambda_1} \\ \cdots & \cdots & \cdots \\ \frac{\partial N_0^x}{\partial \lambda_j} & \frac{\partial N_0^y}{\partial \lambda_j} & \frac{\partial N_0^z}{\partial \lambda_j} \\ \cdots & \cdots & \cdots \\ \frac{\partial N_0^x}{\partial \lambda_{n_v}} & \frac{\partial N_0^y}{\partial \lambda_{n_v}} & \frac{\partial N_0^z}{\partial \lambda_{n_v}} \end{pmatrix}$$

$$\overrightarrow{\delta Q_p} = \begin{pmatrix} \delta\lambda_1 \\ \vdots \\ \delta\lambda_j \\ \vdots \\ \delta\lambda_{n_v} \end{pmatrix} \quad \text{et} \quad \mathcal{J}_p^t = \begin{pmatrix} \frac{\partial N_p^x}{\partial \lambda_1} & \frac{\partial N_p^y}{\partial \lambda_1} & \frac{\partial N_p^z}{\partial \lambda_1} \\ \cdots & \cdots & \cdots \\ \frac{\partial N_p^x}{\partial \lambda_j} & \frac{\partial N_p^y}{\partial \lambda_j} & \frac{\partial N_p^z}{\partial \lambda_j} \\ \cdots & \cdots & \cdots \\ \frac{\partial N_p^x}{\partial \lambda_{n_v}} & \frac{\partial N_p^y}{\partial \lambda_{n_v}} & \frac{\partial N_p^z}{\partial \lambda_{n_v}} \end{pmatrix}$$

Cette façon de présenter le problème permet, en compliquant l'écriture de la fonction d'objectif et des contraintes faisant intervenir les variables virtuelles, de diminuer le nombre de variables du problème de $2n$ et de diminuer le nombre de contraintes d'anticollisions au points d'arrets de $2n$.

5.6.4 Conditions de reprise de l'itération

L'itération est menée en deux phases successives.

- Premier temps : la distance de sécurité stricte est fixée au double de celle demandée, ou à la plus petite distance entre éléments au cours de la tâche de départ, si celle-ci est inférieure.

Lorsque l'amélioration de la longueur de la trajectoire pour trois itérations successives n'est plus que le dixième de la meilleure des trois premières itérations, l'algorithme considère :

- que le résultat commence à être cerné,
- qu'au contraire les conditions de blocage semblent se manifester.

- Deuxième temps : la distance de sécurité est prise égale à sa valeur nominale.

Ceci a pour but, en augmentant la taille apparente des obstacles, de diminuer l'effet des concavités, ainsi de passer outre un minimum local, ce qui augmente les probabilités d'accéder à une trajectoire correspondant au minimum absolu de longueur.

L'itération reprend jusqu'à ce qu'il ne soit plus possible de trouver une trajectoire $\mathcal{T}_{k+1}$ meilleure, plus courte, que $\mathcal{T}_k$ qui la précédait. Le critère d'arrêt considéré est classique :

- amélioration relative de la longueur de la trajectoire par rapport à sa longueur précédente inférieure à un seuil donné : $\delta L \leq \epsilon$,
- nombre maximal d'itérations.

Bien sûr, le processus ayant un caractère itératif local, la tâche résultat est proposée sans garantir qu'elle correspond à un minimum global.

Chapitre 6

Un prototype de validation de la "méthode des articulations virtuelles"

Ce chapitre a pour but de valider la méthode des articulations virtuelles. Un programme mettant en application cette méthode a été programmé et testé. Il permet d'utiliser cette méthode avec des performances raisonnables : pour la clarté de l'exposé, il a été choisi un robot avec peu de variables articulaires et peu de variables d'implantation. Le fonctionnement du programme est discuté à partir de l'exemple d'un robot portique deux axes dont l'orientation dans l'atelier est à optimiser conjointement à une trajectoire donnée comportant quelques points de passages. L'espace ambiant est encombré par quelques obstacles en forme de disques.

6.1 Présentation : le programme prototype

Le programme réalisé reproduit l'algorithme présenté au chapitre précédent. Il dispose d'un programme d'optimisation avec contraintes utilisant le gradient de la durée du mouvement en fonction des variables du problème. Lorsque les contraintes de type intervalle correspondant aux butées du robot sont transgressées dès la trajectoire de départ, ou lorsque éventuellement la trajectoire prise comme trajectoire initiale transgresse tels obstacles, l'algorithme

modifie la direction de progression, pour les variables concernées, dans le sens de la réduction maximale possible de la distance de transgression des butées ou des obstacles concernés. Ainsi est-il permis de proposer une trajectoire de départ qui percute les obstacles.

La simulation envisagée dispose d'autre part d'un programme permettant, par un simple parcours de la trajectoire, d'ajouter des configurations de passage aux lieux où la distance aux obstacles est minimum, permettant ainsi de garantir, pour les petits déplacements, la sécurité de la trajectoire. Ce programme n'ajoute des points de passages que si la distance aux obstacles est suffisament petite, correspondant à un majorant du déplacement engendré par les petites variation des variables. La méthode utilisée pour ce prototype ne permet pas, comme la méthode des contraintes, de contourner les obstacles en cas de collision, c'est pourquoi le seul système permettant la progression vers la non-collision est celui intégré dans l'optimisation générale. La méthode des contraintes sera une des bases du programme finalement industrialisé.

La simulation proposée dispose enfin d'un dispositif de lissage permettant de lutter contre la prolifération des points de passage. Il élimine les points de passage centraux des alignements de points de passage, si ceux-ci ne sont pas aux points de distance minimale. Il élimine aussi les points de passage ayant proliféré lorsque la trajectoire se rapproche vraiment près des obstacles.

Il conserve donc la qualité essentielle des trajectoires qu'il lisse, en ne modifiant l'allure de la trajectoire que dans la limite des petits déplacements, et en conservant les points de passage qui permettent de garantir la sécurité de trajectoire.

Pour la clarté de l'exposé, le robot considéré permet une transformation très simple de l'espace ambiant vers l'espace des coordonnées articulaires, ce qui permet de considérer aisément les variations possibles des variables du problème. Le programme final devra réaliser cela dans le cas général, en considérant la matrice jacobienne du problème au voisinage du cas considéré, et traduire les contraintes d'anticollision dans l'espace des variables articulaires. C'est une des justifications de la limitation aux petits déplacements à chaque étape du processus.

6.2 Les tests proposés

Pour illustrer la validité de cet algorithme, le robot utilisé est un robot portique de 6 mètres de course sur le grand axe et de 3 mètres de course pour le chariot.

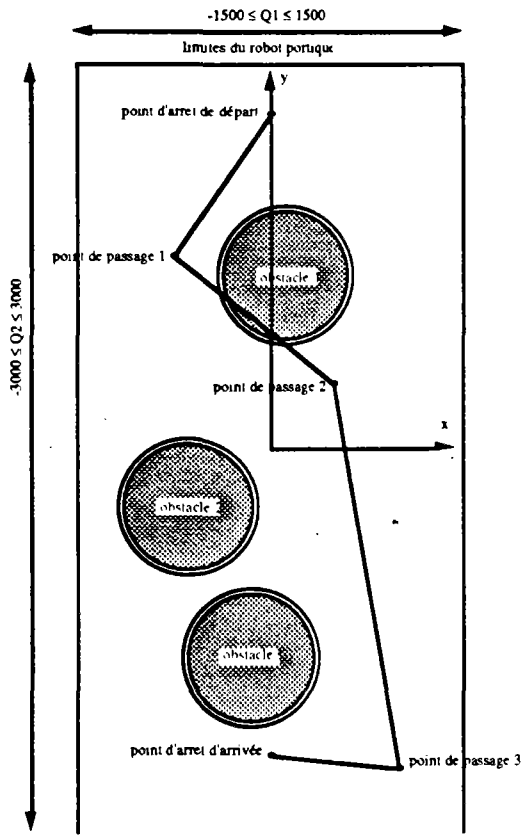


Figure 6.1 : le poste robotisé du test (trajectoire de départ avec collision).

L'espace est encombré, suivant la figure 6.1, de trois réservoirs cylindriques suffisament hauts pour interdire leur survol par l'outil du robot, tout en permettant le passage du portique : dans ces conditions, le robot dispose de deux axes de translation perpendiculaires.

La tâche consiste à aller d'un point de départ à un point d'arrivée. Les trajectoires envisagées doivent être sans collision avec les réservoirs, éventuellement au prix de points de passage intermédiaires. La durée du mouvement sur ces trajectoires est calculée d'après la vitesse maximale de l'axe le plus chargé du robot,

ceci pour chaque tronçon rectiligne de trajectoire. Les temps d'accélération et de décélération des moteurs du robot sont supposés négligeables devant le temps de parcours des tronçons de trajectoires à vitesse constante.

Pour la scène de la figure 6.1, L'optimisation porte sur la variable d'implantation *lambda*, angle de rotation à l'origine du portique par rapport aux axes fixes, et les 2 coordonnées articulaires de chacun des 3 points de passage, soit en tout 7 variables. Ce nombre est susceptible d'augmenter ou de diminuer au cours de l'exécution du processus d'optimisation simultanée de l'implantation et de la trajectoire, en fonction du nombre de points de passage nécessaires.

6.2.1 Cas d'une trajectoire de départ avec collisions ou dépassements des limites

Cette section décrit le fonctionnement de l'algorithme puis ses résultats.

La trajectoire de départ est celle de la figure 6.1, où il y a collision sur le deuxième tronçon de la trajectoire. La première étape consiste donc à placer un point de contrôle au point de distance minimale, là où la collision est "maximale". puis le processus de lissage est inutile car il y a encore trop peu de points de contrôle ;

enfin intervient le processus d'optimisation avec contraintes.

Le programme final, utilisant un algorithme du même type que celui de la méthode de la tolérance flexible, trouvera des déplacements optimaux pour tous les points, en déplaçant le point de contrôle en collision de manière à réduire le viol de la contrainte d'anticollision. Ce viol sera provisoirement consenti suivant une tolérance de plus en plus faible. Comme seuls les petits déplacements sont autorisés, ceci conduira d'abord à des solutions mauvaises s'il y a encore collision enfin bonnes si l'algorithme a réussi à sortir la trajectoire des obstacles.

Le programme d'optimisation avec contraintes proposé ici fonctionne de manière plus simple. Chaque fois qu'une contrainte est violée, il l'accepte, mais remplace pour les variables articulaires réelles et virtuelles des points de contrôle en cause, les composantes du gradient de la fonction d'objectif par

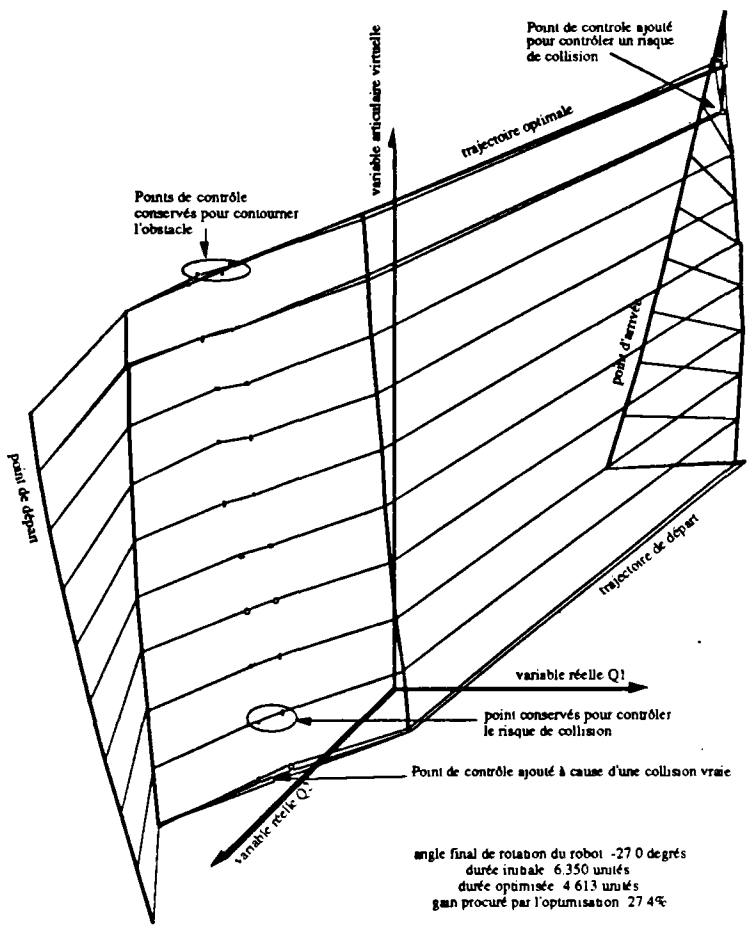


Figure 6.2 : L'optimisation dans l'espace des configurations virtuelles.

celle de la sortie la plus directe de ce point de contrôle de l'obstacle. Dans notre cas, ce calcul est particulièrement aisé puisque les obstacles sont des cercles.

Les étapes du processus, exposées figure 6.3, montrent que des points de contrôle sont ajoutés au droit de l'obstacle en collision, puis que la trajectoire évolue pour éviter la collision, tandis que conjointement les autres points de contrôle tendent la trajectoire. Ensuite, lorsque cette trajectoire passe suffisamment près, des points de contrôle sont encore ajoutés pour surveiller l'évolution de la distance aux obstacles et les contourner.

Le processus s'arrête avec une trajectoire possible optimisée figure 6.4. Il est remarquable que, puisqu'il s'agit de calculer la durée du mouvement suivant

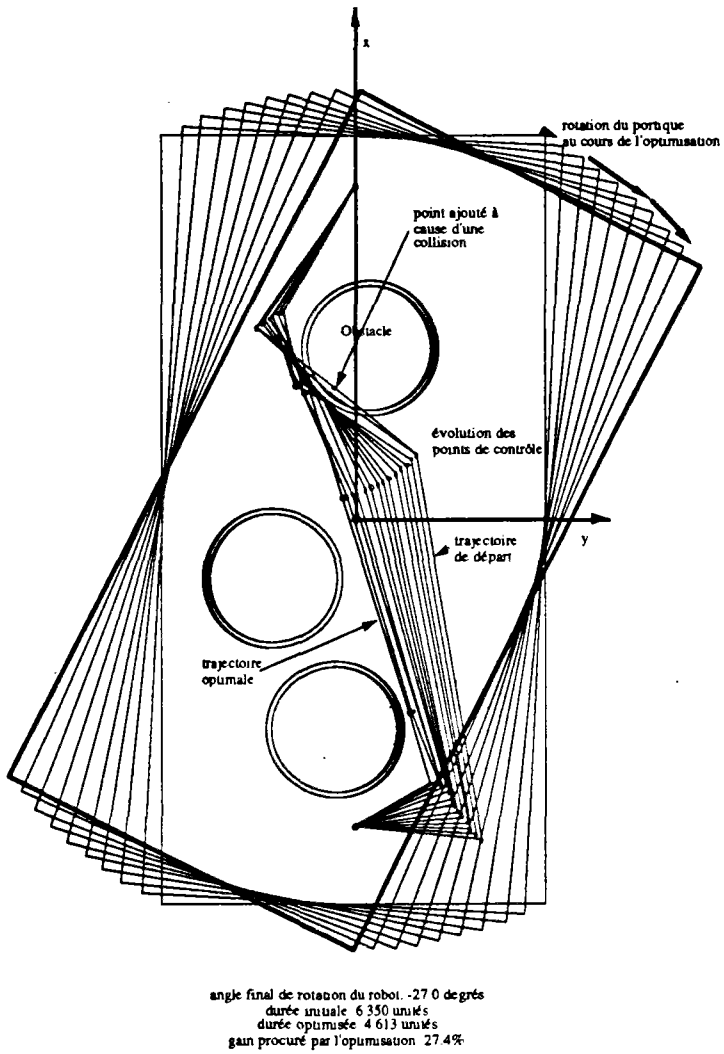


Figure 6.3 : Les étapes de l'optimisation.

l'axe le plus chargé de chaque tronçon, le robot et la trajectoire évoluent pour répartir au mieux le travail sur les deux axes, en orientant la direction "globale" de la trajectoire à 45 degrés par rapport aux axes du robot. la rotation de l'implantation correspondante par rapport à l'atelier est de -27 degrés.

L'algorithme, dans le cas étudié, a permis d'optimiser la durée de la trajectoire de 6350 unités au départ à 4613 unités à l'arrivée, ce qui représente une amélioration de 27,4 pour cent, mais surtout a permi de sortir des collisions.

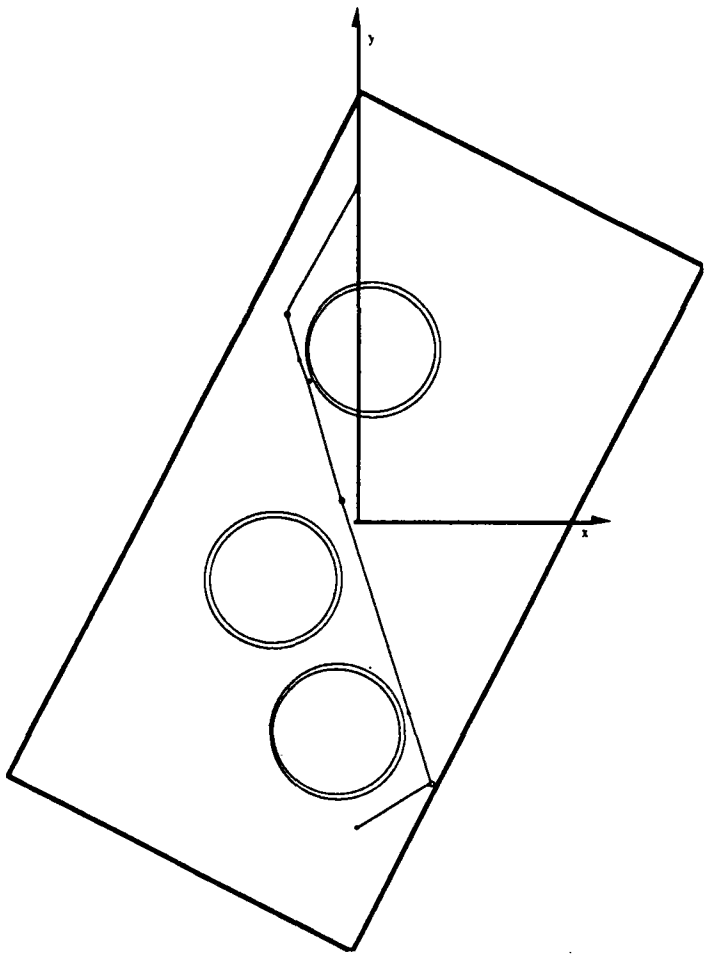


Figure 6.4 : L'implantation et la trajectoire optimisées.

La trajectoire optimisée peut paraître surprenante à première vue : en effet, puisque l'on optimise la durée suivant les axes les plus chargés, le programme n'optimise pas la longueur réelle de la trajectoire : il n'y a donc pas de raison pour améliorer le travail des axes les moins chargés puisque ceux ci n'influent pas sur dans le calcul de la durée de la trajectoire. Ceci donne des trajectoires optimales en temps comme il est bien sûr logique, mais pas nécessairement rasant les obstacles au plus près comme il serait plus intuitif.

6.2.2 cas d'une trajectoire de départ faisable

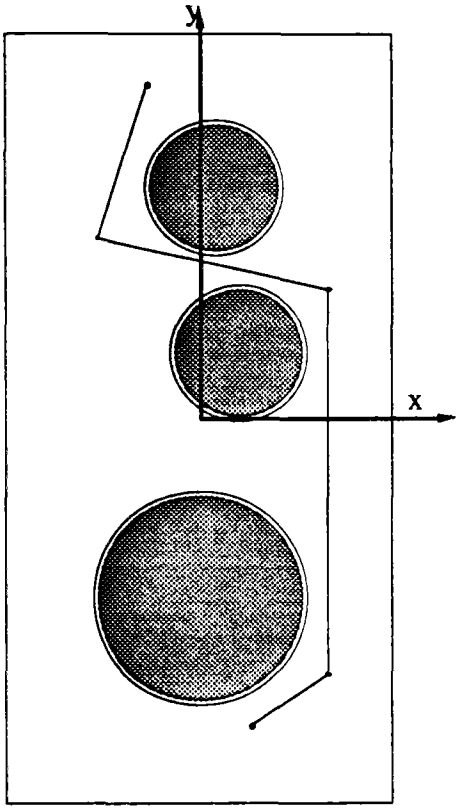


Figure 6.5 : L'implantation et la trajectoire de départ faisables.

la trajectoire et l'implantation de départ sont celles de la figure 6.5.

Le processus d'optimisation est visualisé figure 6.6.

Le résultat final apparait figure 6.7, dans l'espace des configurations virtuelles.

Le résultat final apparait figure 6.8. Dans ce cas, la trajectoire et l'implantation du robot ont été optimisées pour, partant d'une durée de 6600 unités, aboutir à une durée optimisée de 5225 unités, soit un gain de 21 pour cent. L'implantation correspondante du robot fait alors un angle de 21 degrés par rapport à l'orientation de départ.

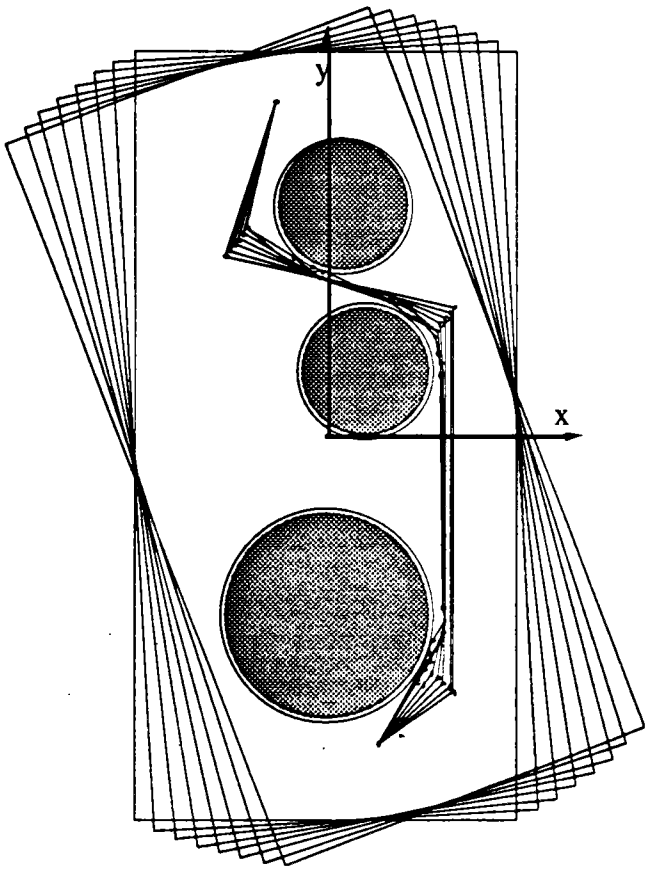


Figure 6.6 : processus d'optimisation de l'implantation et la trajectoire.

optimisation de la durée comparée à celle de la longueur de la trajectoire : Bien que la norme du maximum des valeurs absolues corresponde beaucoup mieux au comportement réel du robot, la figure 6.9 montre la différence entre le résultat d'une optimisation de la durée, calculée avec comme norme le maximum des valeurs absolues, par rapport à une optimisation de la longueur de la trajectoire calculée avec la distance cartésienne. Notons que dans ce dernier cas, les vitesses selon les deux axes du robot étant identiques, l'optimisation de la longueur cartésienne ne fait bien sûr pas intervenir la rotation du robot par rapport à l'atelier.

Il est envisagé, dans le développement du programme industriel définitif, d'intégrer le modèle du comportement du robot le plus réaliste possible.

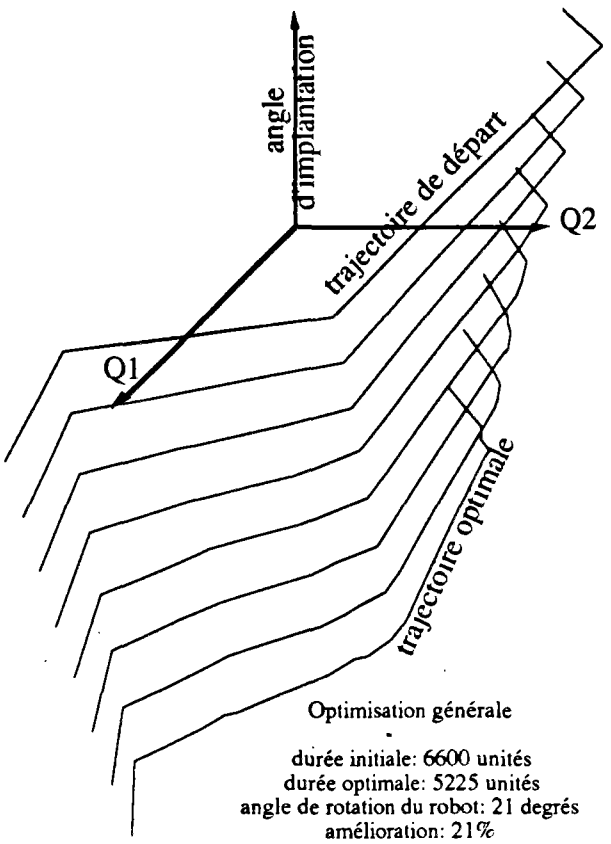


Figure 6.7 : processus d'optimisation de l'implantation et de la trajectoire, exprimé dans l'espace des articulations virtuelles.

Ainsi, le calcul de durée sera éffectué avec une approximation encore meilleure, par exemple par une loi de vitesse trapézoïdale.

6.2.3 Intérêt d'une telle optimisation

Quels peuvent être les gains escomptés d'une optimisation simultanée de l'implantation et de la trajectoire d'un robot, par rapport à l'optimisation de l'implantation seule ou à celle de la trajectoire seulement ? Est-ce même bien utile ?

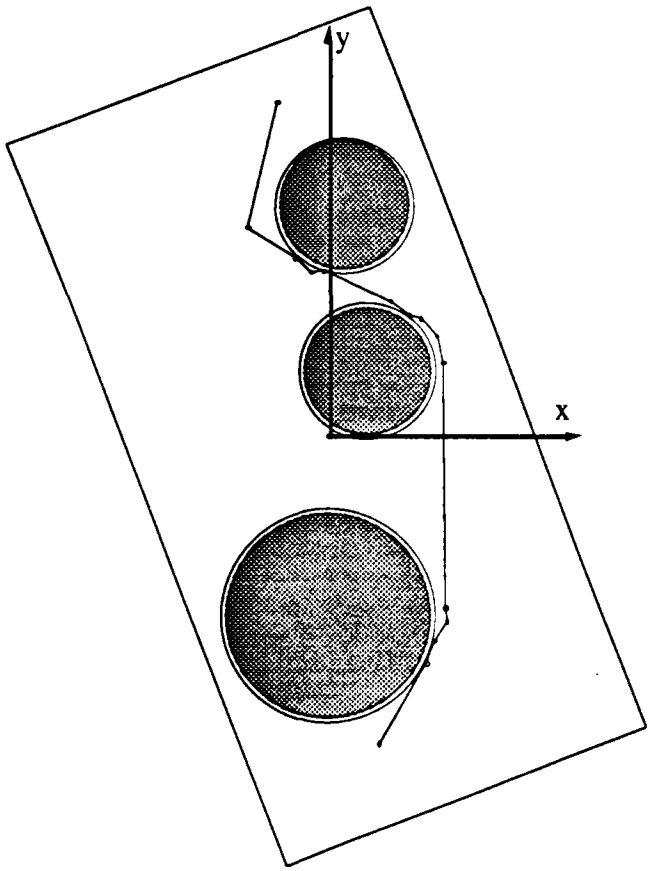
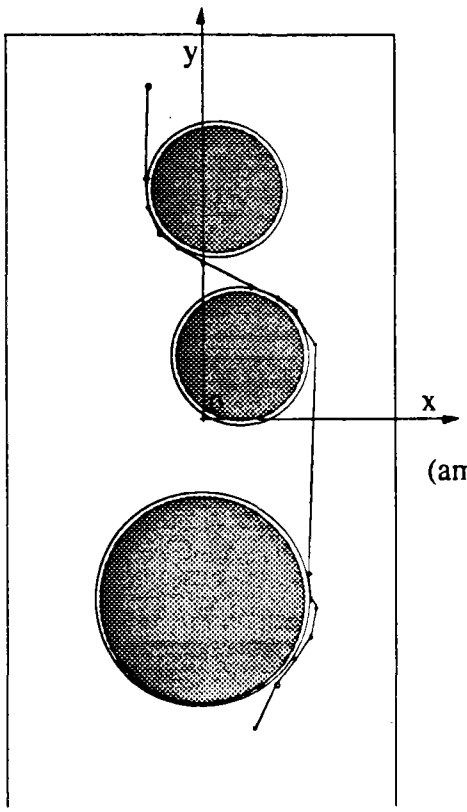


Figure 6.8 : processus d'optimisation de l'implantation et dela trajectoire.

Nous avons testé le programme en bloquant l'implantation ou en n'optimisant que celle-ci. Pour la trajectoire de référence de la figure 5, de durée 6600 unités, on obtient suivant la figure 6.10, une durée de 5541 unités si seule l'implantation est optimisée, ce qui représente un gain de 16 pour cent.

On obtient suivant la figure 6.11, une durée de 5471 unités si seule la trajectoire est optimisée, ce qui représente un gain de 17 pour cent, alors que l'optimisation simultanée procure une durée plus favorable de 5225 unités, soit un gain de 21 pour cent.

Nous avons déjà discuté de l'intérêt qu'il y avait à optimiser l'implantation et la trajectoire simultanément, par rapport à une optimisation indépendante de l'une puis de l'autre. Si l'optimisation est menée successivement, comme sur la



Optimisation générale de la longueur
de la trajectoire (distances cartésiennes)

durée initiale: 6600 unités
durée finale: 5449 unités
amélioration de 17%
(amélioration lorsque la durée est optimisée: 21%)
longueur de trajectoire réduite de 15%

Figure 6.9 : processus d'optimisation de l'implantation et dela trajectoire.

figure 6.12, pour l'implantation seule puis pour la trajectoire, la durée minimale ainsi obtenue est de 4995 unités, alors qu'elle est de 5225 unités si l'optimisation est simultanée. Ce résultat étrange s'explique par une implantation de départ trop différente de l'implantation optimale, ce qui conduit la trajectoire à converger trop vite vers un minimum relatif.

Pour corroborer ceci, nous avons réalisé deux autres essais :

Le premier consiste à optimiser la trajectoire d'abord, puis à tenter d'optimiser l'implantation. Dans ce cas, l'optimisation de l'implantation seule n'aboutit pas, car le système est parvenu à un minimum relatif : les segments de trajectoires sont placés de manière optimale, suivant les directions les plus favorables, certains segments étant même plaçés à 45 degrés par rapport aux axes du robot. L'optimisation de l'implantation ne peut perturber cet équilibre.

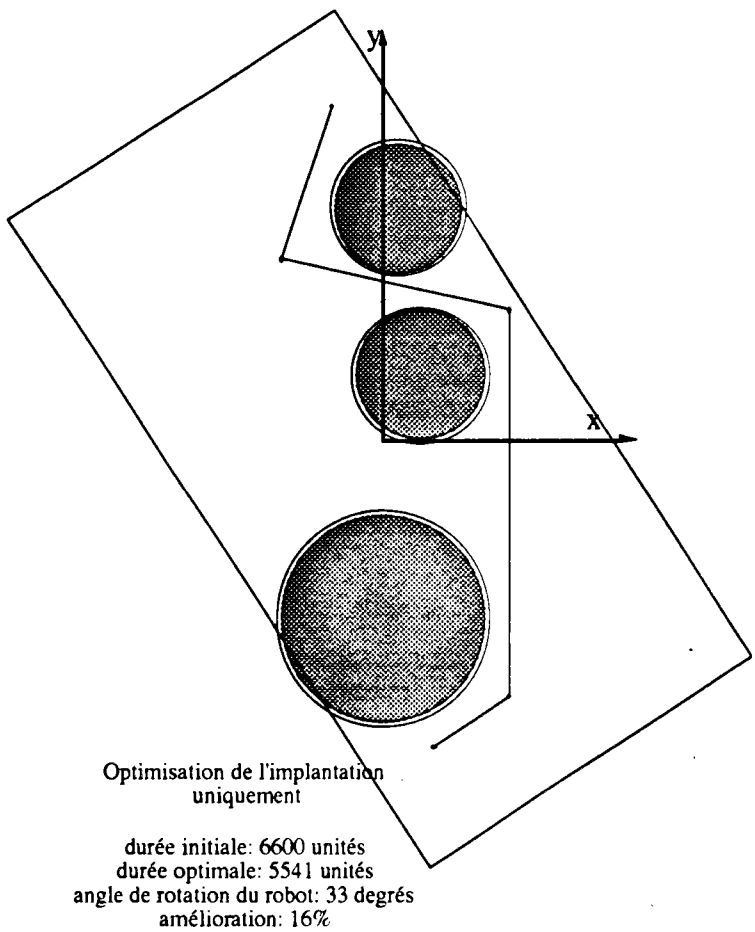
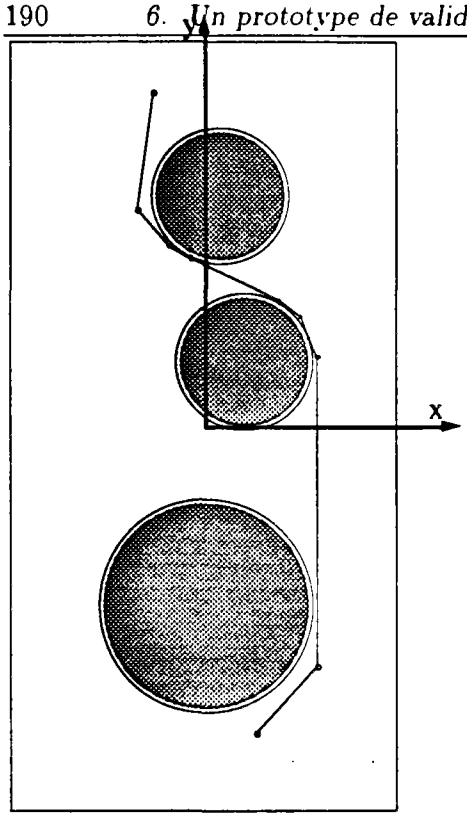


Figure 6.10 : processus d'optimisation de l'implantation seule.

Le deuxième essai consiste à partir de l'implantation trouvée lors d'une optimisation précédente, mais avec la trajectoire de départ initiale, pour être plus proche de la bonne implantation sans que la trajectoire soit trop optimale. Ainsi, le système ne se trouve pas situé à un minimum relatif dès le départ, ce qui nous permet d'atteindre le résultat de la figure 6.13, d'une durée optimale de 4990 unités, ce qui est le meilleur résultat.



Optimisation de la trajectoire uniquement

durée initiale: 6600 unités
durée optimale: 5471 unités
amélioration: 17%

Figure 6.11 : processus d'optimisation de la trajectoire seule.

6.2.4 Conclusion des tests

Nous avons rencontré deux problèmes lors de ces tests :

Le premier concerne les poids à donner aux variables virtuelles, celles d'implantation, par rapport aux variables réelles, celles du robot. comme il fut remarqué pour la méthode des contraintes, l'utilisation des variables articulaires sans normalisation conduit le processus à optimiser préférentiellement les variables qui évoluent le plus.

Dans la méthode des articulations virtuelles, toutes les variables réelles sont normalisées (voir chapitre sur la méthode des contraintes). Ainsi leurs poids dans l'optimisation sont rendus équivalents en divisant chaque variable par sa vitesse maximale, et la trajectoire évolue homogènement.

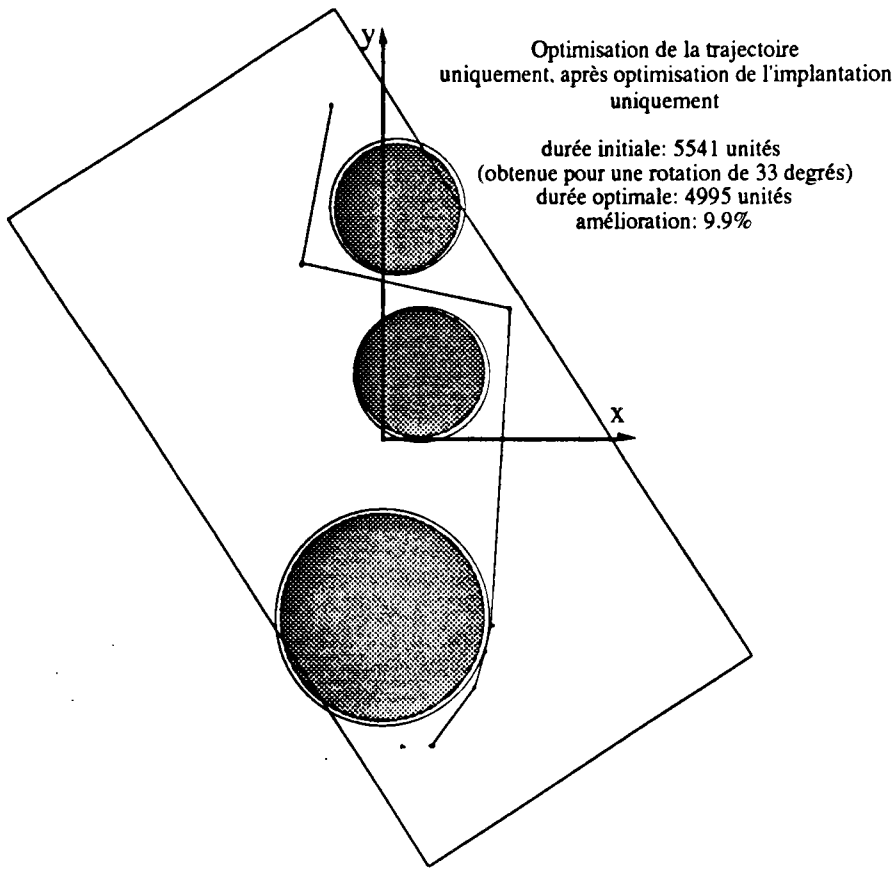


Figure 6.12 : processus d'optimisation de l'implantation puis de la trajectoire séparément.

De cette manière, l'optimisation ne progresse pas préférentiellement suivant l'une des variables plutot qu'une autre.

Il est envisagé, puisque le principe de la méthode proposée consiste à traiter le robot et les libertés d'implantation comme un robot généralisé ayant des axes supplémentaires correspondant à ces libertés, de choisir des vitesses d'évolution de ces variables pour que le robot virtuel ainsi formé soit raisonnable. Dans le test effectué, par exemple, la "vitesse" de rotation du robot par rapport à l'atelier doit être d'autant plus rapide que l'on estime être loin de l'implantation optimale.

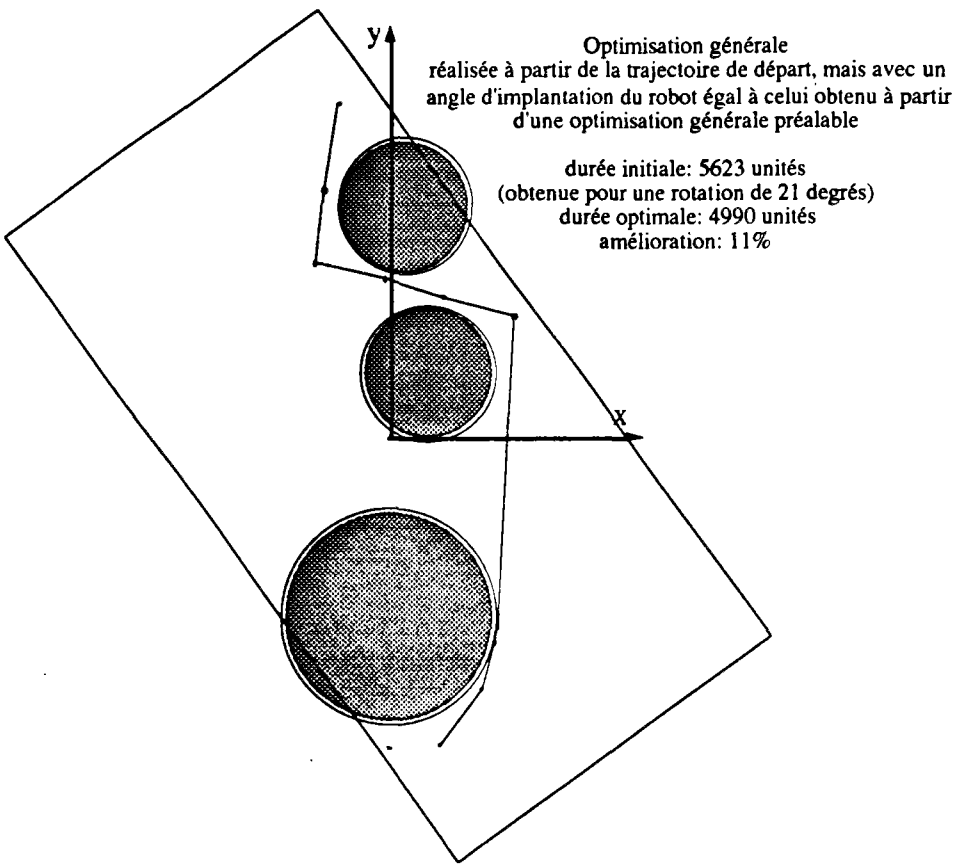


Figure 6.13 : processus d'optimisation de l'implantation et la trajectoire, réitéré.

Trop grande, l'optimisation générale privilégiera trop l'implantation au début, et elle n'apportera pas beaucoup par rapport à une optimisation séparée de l'implantation puis de la trajectoire. Trop petite, la trajectoire convergera avant que l'implantation ait suffisament évolué.

Le second problème apparait lorsque, pour l'un des segments de trajectoire, au moins deux axes du robot sont aussi chargés les uns que les autres : Dans ce cas, le gradient ne peut suffire pour déplacer les deux points de contrôle extrêmités de ce segment, puisque un petit déplacement du premier point n'engendre pas de gain car le même axe est le plus chargé pour le segment considéré et le segment précédent, alors qu'un petit déplacement du second point n'engendre pas de gain non plus puisque un autre axe est le plus chargé à la fois pour le segment considéré et le segment suivant. Si bien que seul un

système considérant des petits déplacements simultanés de plusieurs points, (qui n'est pas le cas de l'optimisation par la méthode du gradient) permettrait de résoudre ce problème.

Nous proposons une solution détournée à ce problème, en intégrant dans le processus de lissage, normalement destiné à éviter la prolifération des points de contrôle, un dispositif de lissage de trajectoire agissant sur trois segments de trajectoire consécutifs. Ce dispositif, consiste à déplacer le segment central parrallèlement à lui même, de manière à réduire la longueur cartésienne de la trajectoire. Ce dispositif traite le cas de blocage rencontré.

Il permet aussi de réduire les déplacements inutiles des axes du robot qui ne sont pas les plus chargés : il est en effet dommage que la trajectoire optimale comporte des déplacements des axes les moins chargés, qui n'ont pas d'influence sur la durée, mais qui ne sont pas toujours justifiés par un obstacle.

Pour terminer, il ne faut pas perdre de vue que le processus est fondamentalement itératif. En conséquence, les résultats peuvent dépendre sensiblement de l'implantation et de la trajectoire de départ.

L'optimisation générale apporte un gain notable, de 24,3 pour cent lors de nos tests, ce qui serait énorme pour un cas industriel. Cette optimisation est meilleure que si chaque problème, trajectoire ou implantation, était traité successivement.

Chapitre 7

Conclusions de l'étude

7.1 Implications de ce projet

Les travaux décrits dans ce mémoire ont permis la construction d'un noyau prototype de conception optimale de cellules robotisées, en intégration avec le système de CAO d'un grand industriel, en vue d'applications opérationnelles dont l'industrialisation est réalisée ou en cours. Ils sont fondés sur une étude préalable précise des besoins industriels dans les domaines de l'implantation des robots, du contournement des obstacles, et des objectifs, critères et contraintes nécessaires à leur optimisation.

Ce prototype propose un premier niveau d'**optimisation d'implantation** de robots. Ce module est utilisable pour implanter le robot de sorte que la durée du mouvement soit minimale, sous réserve que les problèmes de collisions soient traités séparément, ce qui est le cas pour un grand nombre de tâches de soudures par points. Son utilisation est actuellement généralisée en situation industrielle, en liaison avec le logiciel de CAO Catia.

Nous proposons dans son noyau deux outils géométriques fondamentaux aux multiples utilisations industrielles, même en dehors des domaines de la robotique. Un outil de calcul et de contrôle rapide des distances sur l'ensemble d'une cellule robotisée, permet de gérer opportunément le contournement d'obstacle. Il fonctionne de manière hiérarchique, évitant ainsi des calculs systématiques. Il met en oeuvre des volumes simples contenant les objets, évitant ainsi des

calculs précis et longs lorsque la distance est suffisante. L'autre outil permet de transformer des surfaces aux contours quelconques, avec des trous éventuels, en polyèdres, par une méthode de triangularisation originale. En particulier, pour préparer l'usinage ou préparer le calcul de résistance des pièces, le module de polyédrisation de surfaces limitées autorise, grâce à son intégration avec le système de CAO, des bénéfices dont les conséquences industrielles sont très importantes.

Les outils du niveau de la **construction de trajectoires sûres** sont en cours d'industrialisation pour les tâches de montage. Nous les avons réalisés à partir de la méthode des contraintes proposée par B. Faverjon ; ils mettent en oeuvre une méthode locale de construction de trajectoires sans collisions fondée sur l'optimisation d'une fonction d'objectif formée à partir d'une trajectoire directe, sous contraintes d'anti-collisions. Cet outil fonctionne efficacement dans le contexte industriel de l'industrie automobile. Mais beaucoup de conceptions initiales associées les unes aux autres précèdent encore aujourd'hui la construction des tâches de montage. Cette construction, pour être raisonnablement efficace, devrait remettre en cause les choix *a priori* qui s'avèrent trop restrictifs. L'outil proposé permet en cas d'échec, d'analyser les choix initiaux qui en sont la cause. Il fournit ainsi des indications pour les aménager dans un sens favorable. Ces aménagements doivent cependant être encore réalisés manuellement par l'opérateur.

Enfin, au niveau de **l'optimisation conjointe de l'implantation et de la trajectoire sous contrainte d'anti-collision**, qui termine cette étude, correspondent nos propositions aux implications industrielles encourageantes. En effet, en envisageant le problème de l'optimisation multiple en robotique, elles permettent notamment de trouver plus facilement des solutions optimisées viables. Avec cet objectif, en considérant des tâches jalonnées de points d'arrêts imposés et de points de passages ajoutés pour éviter des collisions, nous proposons la méthode des articulations virtuelles, ainsi nommée car les variables d'implantation du robot sont traitées comme des articulations additionnelles du robot assujeties à des contraintes typiques. Cette méthode permet de minimiser la longueur de la trajectoire, en déplaçant le robot et en modifiant la position des points de passages, avec contraintes d'anticollisions. Dans les cas où le mouvement suivant une trajectoire donnée ne peut varier, l'optimisation peut porter sur la durée du mouvement correspondant à la tâche.

7.2 Extension à d'autres travaux futurs :

Mais surtout, par une **ouverture de l'optimisation à d'autres variables du problème**, ce niveau d'optimisation conjointe de l'implantation du robot et de la trajectoire permettra, au fur et à mesure qu'on saura les quantifier, d'optimiser les tâches de robotique sous leurs multiples aspects réels. En effet le traitement des variables d'implantation qui a lieu dans l'application de la méthode des articulations virtuelles est facilement généralisable à d'autres variables pour d'autres problèmes d'optimisation conjointe.

Les aspects réels excessivement multiples des tâches robotisées sont souvent à l'origine des échecs industriels des simulations robotiques ou des déceptions actuelles. Comment optimiser efficacement la tâche d'un robot, sans entendre cela au sens le plus large ? Pourquoi s'étonner qu'une optimisation de trajectoire et d'implantation d'un robot pour introduire une planche de bord dans l'habitacle d'une automobile fasse seulement gagner une faible part du temps d'éxécution de la tâche, alors qu'il n'a pas été envisagé de changer quoi que ce soit à la définition géométrique du préhenseur ?

Forme des pièces

Cette étude a abordé les problèmes liés à une implantation optimale d'un robot couplée à une optimisation de sa tâche. Cette optimisation est contrainte par la nécessaire prise en compte des obstacles.

Ceci est effectif à deux niveaux : celui des causes des collisions potentielles qu'il a été possible d'éviter et celui des causes des blocages qui ont pu intervenir.

Il est facile en effet de noter les lieux de la trajectoire qui sont concernés par la contrainte de non-collision, de connaître en ces lieux les couples d'éléments en cause. Au lieu d'essayer d'éviter la collision, et de faire un détour, il est intéressant d'observer ou de rechercher la cause de la collision potentielle et d'y remédier, dans la mesure du possible, si elle n'est pas structurelle. De même, s'il n'a pas été possible de trouver une trajectoire acceptable sans collisions, il est intéressant de rechercher la cause du blocage, d'étudier les éléments correspondants et d'apprécier comment s'établissent les zones de conflit.
Il est possible par exemple d'envisager -un peu brutalement- de soustraire l'intersection des éléments en collision au volume du corps transporté par le robot : ce serait une manière d'optimiser la forme du corps transporté par le

robot (certes de manière simpliste), mais ceci permettrait de répondre à des questions très importantes telles que par exemple : quelles doivent être les formes du tableau de bord pour que celui-ci puisse être installé dans la voiture en passant par la baie du pare-brise ? Tel élément gênant ne pourrait-il vraiment pas être monté plus tard pour permettre tout de suite un montage robotisé ?

Réadaptation du processus de fabrication, des structures et des formes des machines

La base implicite du raisonnement précédent suppose que l'atelier est impossible à transformer. Au contraire, il est l'élément le plus facile à modifier, car une modification locale a généralement peu d'implication sur les autres éléments du processus de fabrication : si telle pièce de maintien, tel outil ou tel préhenseur gêne le mouvement, on peut souvent envisager de la déplacer ou de modifier sa forme pour que la collision soit supprimée.

Comme ont été introduites des articulations virtuelles pour modéliser les possibilités d'implantation du robot, il peut aussi être envisagé de gérer de cette manière certains paramètres du préhenseur ou de l'outil. Ceci permettrait, si nécessaire, de modifier la forme de ces pièces pour optimiser plus largement la tâche ; par exemple, en cas de blocage pour le montage d'une planche de bord par la baie du pare-brise, il pourrait être utile d'orienter le préhenseur de manière légèrement différente, de modifier quelques dimensions du préhenseur, de changer un peu la position de la planche de bord par rapport à la platine d'extrémité du robot, dans des directions favorisant le déblocage. Ceci peut aussi bien être envisagé de manière systématique pour diminuer le temps d'exécution de la tâche. Mais il est en fait nécessaire de contrôler les modifications de ce type, afin que le préhenseur reste réaliste : là aussi, des systèmes d'aide à la conception permettraient de respecter une partie des règles de construction des préhenseurs...

Modélisations complémentaires utiles à la CAO Robotique

La définition géométrique de l'atelier en CAO ne peut être tout à fait complète : les pièces souples ou déformables ne sont pas modélisées parce que leur position dans l'espace dépend de paramètres qui sont négligés, ou parce que leur comportement physique ou cinématique [Che88], n'a pas encore pu être simulé correctement : ainsi les flexibles d'alimentation des outils du robot sont-ils les éléments qui passent souvent en effleurant certains obstacles, alors même qu'ils sont négligés, dans l'état actuel des choses, dans les études de collisions. La

définition de la trajectoire ne tient aucun compte des efforts qu'ils subissent, d'où des dégradations précoces et surtout imprévues, il conviendra d'envisager une prise en compte même partielle.

"Pousser l'obstacle"

Une autre façon d'envisager l'optimisation pourrait être de répartir ses effets d'une manière beaucoup plus générale : au lieu de détourner le robot de sa trajectoire, et de s'en tenir à cela, il pourrait être envisageable de "faire faire un bout de chemin aux obstacles" : ainsi, le robot serait détourné de sa trajectoire nominale de manière moindre, les obstacles étant déplacés. Ceci implique de prendre en compte correctement les possibilités réalistes de déplacement de tel ou tel obstacle, c'est à dire de tenir compte adroitement du savoir-faire de ceux qui avaient initialement placé les obstacles. Par exemple, si un dispositif de serrage gêne pour souder deux tôles, il est sûrement possible de le déplacer un peu, mais il est difficile de le faire sans un minimum de connaissances et de règles. Celles-ci pourraient être prises en compte par des systèmes experts d'aide à la conception des cellules robotisées.

Plusieurs tâches à traiter

Il arrive de plus en plus qu'un robot ne soit pas affecté à une tâche unique, mais à un ensemble de tâches : par exemple, il peut traiter un ensemble de points de soudures différents suivant le véhicule à assembler, ce qui est le cas pour les ateliers de production flexibles. Il peut se voir affecté une tâche modifiée en cas de panne d'un des autres robots de l'atelier afin d'assurer une partie de ce travail.

Dans ces conditions, il est possible d'optimiser l'implantation du robot pour une tâche particulière indépendament des autres tâches, tout en maintenant ces autres tâches réalisables, mais probablement non optimales. Ceci peut être réalisé en optimisant la tâche principale, et en vérifiant simplement que les modifications d'implantations ne créent pas de collisions pour les autres tâches. Si les modifications d'implantations provoquent une collision, la tâche en cause est modifiée en ajoutant des points de passages pour contourner l'obstacle. Le processus s'arrête lorsque il est impossible d'améliorer la tâche principale, ou lorsqu'il est impossible de maintenir toutes les tâches secondaires sûres, même en ajoutant des points de passages.

En réalité, il faudrait réaliser l'optimisation de l'implantation du robot selon un critère souple : Le plus important étant d'atteindre les objectifs fixés,

il faut commencer par la tâche qui dépasse le plus les objectifs de durée initialement prévus, puis, lorsqu'elle est au niveau des autres tâches, optimiser alternativement les tâches. Le critère d'optimisation n'est plus alors seulement la durée des tâches, mais une durée globale pondérée par la fréquence de ces tâches et par le gain de productivité espéré.

Bibliographie

[ABF88] F. Avnaim, J. D. Boissonnat et B. Faverjon
A practical exact motion planning algorithm for polygonal objects admist polygonal obstacles.
In Proceedings of IEEE Intenational Conference on Robotics and Automation, page 1656-1661, Philadelphie, avril 1988.

[Ala90] H. Alameh
Modélisation mathématique des cables. Cahiers du cerma, Ecole Nationale des Ponts et Chaussées, Paris, septembre 1990.

[AH88] J. Ashton et D. Hoang
An algorithm for finding optimal path between points while avoiding polyedral obstacles .
in proceedings of the International Symposium and Exposition on Robots, 19th ISIR of the International Federation of Robotics, Sidney, Australia, Novembre 1988.

[Bas87] M. Basseville
Distance measures for signal processing and patern recognition.
Rapport de Recherche INRIA numéro 899, Rennes, septembre 1988.

[Bel87] C. Bellevaux
Diplôme d'Etudes Approfondies d'Intelligence Artificielle, Reconnaissance des Formes et Algorithmique Graphique. Cours d'Informatique Graphique, Module d'Initiation à la Recherche numéro 7
Université de Paris VI et Ecole Nationale Supérieure des Techniques Avancées, 1987.

[BFM88] J.D. Boissonnat, B. Faverjon et J.P. Merlet
Techniques de la Robotique : Architecture et Commandes.
Traité des Nouvelles Technologies, série Robotique, Hermès, février 1988.

[Bob88] J.E. Bobrow
Optimal path planning using the minimum-time critérion,
in International Journal of Robotics and Automation, Volume 4, Pages 443-450, 1988.

[Bob89] J.E. Bobrow
A direct minimization approach for obtaining the distance between convex polyedra,
in International Journal of Robotics Reseach, Volume 8, numéro 3, Pages 65-76, juin 1989.

[Bor85] P. Borrel
Modèle de robots pour programmation en CAO.
CAO-Robotique, état de l'art, Hermès 1985.

[Bou86] R. Boulic
Conception assistée par ordinateur de boucles de commandes avec capteurs en robotique et en téléopération.
Thèse de Docteur-Ingénieur, Université de Rennes 1, novembre 1986.

[BSNA89] Bauer,Schaeffer, Novikoff , J.-M. Allée (ParisVI)
Modélisation conceptuelle de l'environnement d'un robot mobile autonome.
Actes de la Convention sur l'Intelligence Artificielle 90, pages 701-710, 1990

[Cia90] P. G. Ciarlet
Introduction à l'analyse numérique matricielle et à l'optimisation
Masson, Paris, 1990.

[Che84] D. Chevallier
Groupes de Lie et mécanique des systèmes de corps rigides.
Séminaire de Modélisation Mathématique, Kassel (RFA), octobre 1984.

[Che88] D. Chevallier
Flexibilités et inerties résiduelles des structures amorties.
Annales des ponts et chaussées, numéro 48, pages 3-19, 1988.

[Cra86] J.J. Craig
Introduction to robotics : manipulation and control.
Adisson-Weshley, Reading, USA, 1986.

[Cro87] A. Crosnier
Integration d'une fonction de perception dans un système de CAO Robotique.
Thèse de Doctorat, Université des Sciences et Techniques du Languedoc, Montpellier, 1987.

[CRW91] P.Chedmail, F Reynier et P. Wenger
Positionnement optimal d'un robot en présence d'obstacles mobiles.
Actes du Micad 91, Paris, pages 113-132, 1991.

[CW87] P. Chedmail et P. Wenger
Domaine atteignable par un robot : généralisation de la notion d'aspect à un environnement avec obstacles, application aux robots articulés plans.
Huitième Congrès Français de Mécanique, Nantes, septembre 1987.

[CW89] P. Chedmail et P. Wenger
Design and positioning of a robot in an environment with obstacles using optimal research.
in IEEE conference on Robotics and Automation, Scottdale, Arizona, may 1989.

[Das89] Dassault Systèmes
Catia, Catgeo and Catmsp libraries : Reference manuals
International Business Machines, Decembre 1989.

[Del87] S. Delignières
Choix de morphologies de robot.
thèse de Doctorat, université de Nantes, Novembre 1987.

[Dia87] A. Diaz
Interactive solution to multiobjective optimization problems.
International Journal for Numerical Methods in Engineering, Volume 24, 1987.

[Dom90] E. Dombres
Conception et programmation hors ligne complète de cellules robotisées sur un système de CFAO.
Projet du Ministère de la Recherche et de la Technologie, Laboratoire d'Automatique êt de Microélectronique de Montpellier, Juillet 1990.

[Don83] B. R. Donald
Hypothesizing channels through free space in solving the find path problem.
Technical memo number 736, Massachussetts Institute of Technology, Artificial Intelligence Laboratory, 1983.

[Erd84] M. A. Erdmann
On motion planning with uncertainty.
Technical repport number 810, Massachussetts Institute of Technology, Artificial Intelligence Laboratory, aout 1984.

[ESP86] B. Espiau
Commande de systèmes redondants et évitement d'obstacles.
Rapport de recherche INRIA IRISA numéro 495, Rennes, mars 1986.

[Esp88] B. Espiau et C. Samson
Le point sur la commande proximétrique.
Techniques de la Robotique, tome 2, Hermès, 1988.

[Fav84] B. Faverjon
Obstacle avoidance using an octree in the configuration space of manipulator.
In Proceedings of IEEE International Conference on Robotics and Automation, Atlanta, mars 1984.

[FR89] M. Fayet et M. Renaud
Quasi-minimal computation under an explicit form of the inverse dynamic model of a robot manipulator.
in Mechanism and Machine Theory, volume 24, number 3, pages 165-174, EUA, 1989.

[FT86] B. Faverjon et P. Tournassoud
A hierarchical CAD system for multi robot coordination.
In Proceedings of the NATO Advanced Research Workshop on Languages for Sensor Based Control in Robotics, Pise, Italie, septembre 1986.

[FT87a] B. Faverjon et P. Tournassoud
A local based approach for path planning of manipulators with a high number of degrees of freedom.
In Proceedings of IEEE International Conference on Robotics and Automation, pages 1152-1159, Raleigh, avril 1987 , et aussi Rapport de Recherche INRIA numéro 621, février 1987.

[FT87b] B. Faverjon et P. Tournassoud
The mixed approach for motion planning : learning global strategies from a local planner.
In Proceedings of the International Journal on Artificial Intelligence, pages 1131-1137, Milan, aout 1987.

[FT88] B. Faverjon et P. Tournassoud
A practical approach to motion planning for manipulators with many degrees of freedom.
Rapport de recherche de l'INRIA numéro 951, Sophia Antipolis, 1988.

[FOU90] J.Y. Fourquet
Mouvement en temps optimal pour les robots manipulateurs en tenant compte de leur dynamique non linéaire.
Thèse de doctorat, Université de Toulouse 3, .

[Gas87] P. Gaspart
Langages de programmation de la robotique.
Traité des nouvelles technologies, série robotique, Hermès,1987.

[GH87] D. Georges et Y. Hamam
Optimal trajectory planning for manipulator robot.
Automatique Productique et Informatique Industrielle, volume 21, numéro 2, pages 129-150, 1987.

[Gha87] M. Ghallab
Optimisation de processus décisionnels pour la robotique.
Thèse de doctorat, Université Paul Sabatier de Toulouse,1987.

[GJK88] E. G. Gilbert, D.W. Johnson, S.S. Keerthi
A fast procedure for computing the distance beetween complex objects in three-dimentional space.
IEEE Journal of Robotics and Automation, volume 4 numéro 2, pages 193-203, Avril 1988.

[Gio87] M. Giordano (Institut Universitaire de Technologie d'Annecy, Laboratoire d'Automatique et de Microinformatique Iindustrielle)
Méthodes d'obtention d'un modèle dynamique de robots en chaîne complexe.
Publications de l'Association Française pour la Cybernétique Economique et Technique numéro 21, page 151-173, 1987.

[Gom86] O. Gomart
Some results in Computer Aided Robot Simulation
In Procedings of 16th International Symposium on Industrial Robots, pages 785-795, Bruxelles, IFS Publications Ltd, Octobre 1986.

[Gom87] O. Gomart
Software for Robotic Workcell Design, Simulation and Optimization.
European Catia Users Association Conference, Milan, septembre 1987.

[Gom88] O. Gomart
Sensor Simulation for Robotic Workcell Design
in Proceedings of the third International Conference CAD-CAM-CAE Integration Technologies in the Automotive Industry, pages 511-521, ATA, Turin, novembre 1988.

[Gom90] O. Gomart
Programmation hors ligne de robots en CAO
Actes de la conférence Exporobot 90, Session "Optimisation de l'expolitation des robots", Association Française de robotique industrielle, pages 19-42, Paris, mars 1990.

[Gou84] L. Gouzènes (CNRS-LASS)
Strategies for solving collision-free trajectories problem for mobile and manipulator robots.
International Journal of Robotics Research, volume 3, numero 4, hiver 1984.

[Gup86] K. C. Gupta
On the nature of the robot workspace.
in International Journal of Robotics Reseach, numéro 2, 1986.

[HP87] A. Haurat et J. L. Perrard
Panorama des langages de programmation des robots industriels.
Axes Robotique n26, Septembre 1987.

[HT88] T. Hasegawa et H. Terasaki
Collision avoidance : Divide and Conquer approach by determining intermediate goals :
in proccedings of IEEE Transactions on Systems, Man and Cybernetics, volume 18, numero 3, pages 337-347,1988.

[HW86a] J. Hopcroft et G. Wilfong
Motion of objects in contact.
in International Journal of Robotics Reseach, numéro 4, 1986.

[HW86b] J. Hopcroft et G. Wilfong
Reducing multiple object motion planning to graph searching.
in SIAM Journal of Computing, volume 15 numéro 3, pages 768 à 785, 1986.

[IMS89] IMSL inc.
I.M.S.L. Math library : Fortran subroutines for Mathematical Applications.
User's manual, Edition 9.2,
IBM Problem-Solving Software Systems, Decembre 1989.

[IO85] Y. Ichikawa et N. Ozani
A heuristic planner and an executive for mobile robot control.
In IEEE transaction on Systems, Man and Cybernetics, volume 15 numero 4, 1986.

[ISO88] ISO
Manipulating industrial robots : vocabulary.
Association Française de Normalisation, Rapport technique ISO/TR 8373, 1988.

[JG85] D. Johnson and E. Gilbert
Minimum time robot path planning in the presence of obstacles.
Proceedings of the 24th conference on decision and control, pages 1748-1753, Ft Lauderdale, decenber 1985.

[Jos89] D. Josipovici
Audit préparatoire à un cahier des charges sur la simulation des cables de robots pour Peugeot S.A.,
Ecole Polytechnique Féminine,Rapport de fin d'étude, Paris, septembre 1989.

[Kha85] W. Khalil et J.F. Kleinfinger
A new geometric notation for open and closed loops robots.
In IEEE International Conference on Robotics and Automation, San Francisco, USA, Avril 1986 .

[Kha86] O. Khatib
Real time obstacle avoidace for manipulators and mobiles robots.
in International Journal of Robotics Reseach, volume 5 numéro 1, pages 90-99, primptemps 1986.

[Kle86] J.F. Kleinfinger
Modélisation dynamique de robots à chaine cinématique simple, arborescente ou fermée, en vue de leur commande.
Ecole Nationale Supérieure de Mécanique de Nantes, Thèse de doctorat, Nantes, mai 1986.

[KL86] J. Koplik et M. C. Leu
Computer generation of robot dynamics equations and the related issues.
in Journal of Robotic System, volume 3(3),page 301-319, mars 1986.

[Kod87] D. Koditscheck
Exact robot navigation by means of potential functions : some topological considerations.
in Proceedings of IEEE International Conference on Robotics and Automation, pages 1 à 6, Raleigh, april 1987.

[Lau87] J.P. Laumont (LASS-CNRS)
Le raisonnement géométrique et spatial en robotique.
Bulletin de l'Association Française pour la Cybernétique Economique et Technique, Interfaces, numéro 56, pages 3-11, juin 1987.

[Leg80] J. Legras
Algorithmes et programmes d'optimisation non linéaire avec contraintes. Application au contrôle optimal.
Masson. Paris, 1980.

[Ler87] J. Lerbet
Mécanique des systèmes articulés rigides comportant des boucles fermées.
Thèse de doctorat, Université de Paris VI Pierre et Marie Curie, Juin 1987.

[Ler88] J. Lerbet
Cinématique des mécanismes.
Ecole Nationale des ponts et chaussées, Cahiers du CERMA n8, Juin 1988.

[LG87] C. Laugier et F. Germain
An adaptative collision-free trajectory planner.
In Proceedings of the International Conference on Advanced Robotics, Tokyo, septembre 1987.

[LMJ*87] T. Lozano-Pérez, J. Jones, E. Mazer, P. O'Donel, W. Grimson, P. Tournassoud, A. Lanusse
Handey : a robot system that recognizes, plans and manipulates.
In Proceedings of the fourth international Symposium on Robotics Research, pages 29 à 36, Santa-Cruz, aout 1987.

[LMT84] T. Lozano-Pérez, M. Masson, R. Taylor
Automatic synthesis of fine motion strategies for robots.
In International Journal of Robotics Research, volume 3 numero 1, pages 3 à 24, printemps 1984.

[Lo 87] S.H.Lo (Laboratoire Central des Ponts et Chaussées)
A new mesh generation scheme for arbitrary planar domains,
in International Journal for Numerical Methods in Engineering, volume 21, pages 1403-1426, 1985.

[Loz86] T. Lozano-Pérez
A simple motion planning algorithm for general robot manipulators.
In Proceedings of the 5th AAAI, Philadelphie, 1986, et aussi ISRR, MIT Press, Gouvieux 1986.

[Lu 88] Y. Lu
Etude du volume de travail des robots : Enveloppe, atteignabilité.
Thèse de l'Ecole Nationale Supérieure des Arts et Métiers, Laboratoire de Robotique de Paris, février 1988.

[LWP 83] Luh, Walker, Paul
Newton-Euler formulation of manipulators dynamics for computer control.
In International IFAC/IFIC Journal, Purdue University,1983.

[Mar88] P. Martin et D. Martin (Université de Nantes, Département de Mathématiques et d'Informatique, GRIN)
Les algorithmes de calcul de l'intersection de solides définis par leurs bords.
Revue de CFAO et d'Infographie, V3(2) pages 30-53, 1988.

[Meg84] S. M. Megahed
Contribution à la modèlisation geometrique et dynamique des robots manipulateurs à structure de chaine cinematique simple ou complexe.
Thèse de doctorat, Université Paul Sabatier de Toulouse, juillet 1984.

[NHT88] T. Nagata, K. Honda, Y. Teramoto
Multirobot plan generation in a continuous domain : planning by Use of Plan Graph and Avoiding Collisions Among Robots.
In IEEE journal of Robotics and Automation, volume 4, number 1, février 1988.

[Pap85] C. H. Papadimitriou
An algorithm for shortest path motion in three dimensions.
Information Processing Letters, volume 20, pages 259 à 263, 1985.

[Per86] J. Pertin-Troccaz
Modélisation du raisonnement géométrique pour la programmation des robots.
Thèse de doctorat, Institut National Polytechnique de Grenoble, 1986.

[Pre77] F. P. Preparata et S. J. Hong
Convex hulls of finite sets of points in 2D and 3D.
Communications of the Association for Computing Machinery, volume 20, numéro 2, février 1977.

[PS85] F. P. Preparata et M. Shamos
Computational Geometry : an Introduction.
Springerr-Verlag, New York, 1985.

[Ren87] M. Renaud
Quasi-minimal computation of the dynamic model of a robot manipulator utilizing the Newton-Euler formalism and the notion of augmented body.
in IEEE Conference on Robotics and Automation, Raleigh, EUA, 1987.

[RD87] J. Rastergar et P. Deravi
Method to determine workspace, its subspaces with different numbers of configurations and all the possible configurations of a manipulator.
Mechanic and Machinery Theory, numéro 22/4 1987.

[RS85] J. Reif et J. Storer,
Shortest paths in Euclidian Space with polyedral obstacles.
Brandeis University, Computer Science Department, Technical Report number 121, 1985.

[Sam87] C. Samson
Une approche pour la synthèse et l'analyse de la commande des robots manipulateurs rigides.
In International Journal of Robotics Research, volume 5 numero 4, pages 56-68, Printemps 1986.

[SC86] R. Smith et P. Cheeseman
On the representation and estimation of spatial uncertainty.
INRIA, rapport de recherche numéro 669, Mai 1987.

[SD86] S. Dubowski, M. Norris and Z. Shiller
Time optimal planning for robotic manipulator with obstacle avoidance : a CAD approach.
Proceedings of IEEE International Conference on Robotics and Automation, pages 1906-1912, San Francisco, avril 1986.

[SD87] Z. Shiller and S. Dubowski
Time optimal path planning for robotic manipulator in the presence of obstacles with actuator, gripper and payload contraints.
Technical repport number 12-42, Massachussetts Institute of Technology, Industrial Liaison Program, 1987.

[SF88] M. Sampei et K. Furuta
Robot control in the neighborhood of singular points.
IEEE Journal of Robotics and Automation, V4N3 pages 303-309, juin 1988.

[Shi87] Z. Shiller
Time optimal motion of robotic manipulators.
Doctoral thesis, Massachussetts Institute of Technology, Cambridge MA, may1987.

[SS88] S. Suh and K. G. Shin
A variational dynamic programming approach to robot path planning with a distance safety criterion.
In IEEE Journal of Robotics and Automation, Juin 1988.

[TLM87] P. Tournassoud, T. Lozano-Perez et E. Mazer
Regrasping,
in IEEE International Conference on Robotics and Automation, Raleigh, 1987.

[Ton84] B. Tondu
Génération de mouvements point à point optimaux en robotique : application à l'évaluation des performances et à la CAO des robots.
Thèse de doctorat, Université des Sciences et Techniques du Languedoc, Montpellier, 1984.

[Tou85] P. Tournassoud
On motion coordination,
INRIA, rapport de Recherche 549, juillet 1986.

[Tou88] P. Tournassoud
Planification de trajectoire en robotique, complexité et approche pratique.
Thèse de doctorat, Université de Paris-Sud, centre d'Orsay, mars 1988.

[Tou88] P. Tournassoud
Géométrie et intelligence artificielle pour les robots.
Traité des Nouvelles Technologies, série Robotique, Hermès, septembre 1988.

[Ver87] L. Vergely
Implantation et comparaison d'algorithmes de génération de trajectoires sur simulateur de robot mobile.
Diplôme d'Etudes Approfondies, Université des Sciences et Techniques du Languedoc, Montpellier, juillet 1987.

[Ver89] L. Vergely
Modélisation de capteurs : développement d'outils d'aide à la conception de cellules robotisées.
Thèse de doctorat, Université des Sciences et Techniques du Languedoc, Montpellier, 1989.

[VW86] Vijaykumar et Waldron
Geometric optimisation of serial chain manipulator structure for working volume and dexterity.
in International Journal of Robotics Reseach, numéro 2, 1986.

[WC90] P. Wenger et P.Chedmail (Ecole Nationale Supérieure de Mécanique de Nantes)

Géométrie de l'espace de travail d'un manipulateur et conception de sites robotisés.
Actes du Micad 90, Paris, pages 392-407, 1990.

[WEN89] P. Wenger
Aptitude d'un robot a parcourir son espace de travail en présence d'obstacles.
Thèse de doctorat, CNRS, Nantes 1989.

[WF86] E. K. Wong and K. S. Fu
A hierarchical orthogonal space aproach to 3D path planning.
In IEEE Journal of Robotics and Automation, mars 1986.

[Yan87] Y. Yang
Contribution à l'amélioration de la flexibilité des robots d'assemblage. Cas de la commande par vision active 3D.
Laboratoire de Mécatronique de l'Institut Supérieur des Matériaux et de la Construction Mécanique, décembre 1987.

[YL84] D.C.H. Yang et T.W. Lee
Heuristic combinatorial optimization in the design of manipulator workspace.
in proccedings of IEEE Transactions on Systems, Man and Cybernetics, number 14.4, Aout 1984.

[YD90] P.A. Yvars et J. Doulcier
Poste de travail du roboticien pour l'étude et la simulation de lignes de production flexibles.
Actes du colloque international Productique et Intégrations, Bordeaux 12-14 juin 1990, Teknea, 1990.

[YD92a] P.A. Yvars et J. Doulcier
Contribution to the study of automatic positioning of robot manipulators in the design of flexible systems for manufactoring industry.
In International Journal of Robotic and Artificial Intelligence, Robotica, février 1992.

[Yva90] P.A. Yvars
Poste de travail du roboticien pour l'étude, la simulation et la reconfiguration des lignes de production flexibles dans l'industrie manufacturière
Thèse de doctorat, Université des Sciences et Techniques de Lille Flandres Artois, Lilles, Novembre 1990.

Liste des Figures

Troisième Partie

Annexes

Troisième Partie

Annexes

La première annexe fait référence à la première partie sur la problématique de l'optimisation de l'implantation des robots, et d'autre part à celle de la génération de trajectoires sans collisions par des méthodes locales d'optimisation sous contraintes. Elle a pour but de présenter et d'analyser les caractéristiques des méthodes d'optimisation avec contraintes disponibles, afin de permettre le choix le plus efficace dans le cas de l'optimisation de l'implantation d'un robot avec contraintes d'anticollisions. Il débouche sur le choix de la méthode de la tolérance flexible, particulièrement apte à résoudre les problèmes très contraints. Bien entendu, la programmation de cette méthode n'a pas été envisagée puisque tel n'était pas le but de nos travaux, et qu'il existe de nombreuses bibliothèques de programmes disposant de multiples méthodes d'optimisation avec contraintes.

La seconde annexe fait référence au chapitre présente notre proposition d'une méthode mixte géométrique et analytique pour déterminer le domaine d'implantation du robot. Elle débouche sur un algorithme rapide aux implications particulières pour l'intelligence artificielle. Elle fait l'objet d'un article et d'une conférence proposés et acceptés, [YD92a] et [YD92b].

Annexe A

Méthodes d'optimisation ou d'investigation

Cette annexe a pour but de présenter les principes de base des méthodes d'optimisation avec contraintes, afin de proposer le choix convenant pour les applications particulières que sont l'optimisation d'implantation sous contraintes d'anticollisions d'une part, et la génération de trajectoires sans collision d'autre part. cette annexe présente les solutions existantes pour trouver des moyens de prendre en compte correctement les contraintes. En effet celles-ci péanalisent toujours lourdement le calcul. C'est pourquoi nous suggérons d'utiliser pour nos travaux la méthode de la flexible tolérance, puisqu'elle autorise d'outrepasser les contraintes d'autant plus largement au cours du calcul qu'il semble loin de la solution. Cette méthode n'est pas programmée par nos soins puisque des bibliothèques de programmes mathématiques d'optimsation proposant de nombreuses familles de méthodes sont disponible : c'est une méthode de ce type proposée par l'une d'elle que nous avons utilisé pour nos programmes.

L'idéal serait d'envisager une méthode d'optimisation qui, parce qu'elle tiendrait compte des particularités du problème de l'optimisation de trajectoire et d'implantation de robots, serait particulièrement efficace pour résoudre ces problèmes spécifiques que sont :

- L'implantation
- Le calcul des trajectoires

Les méthodes d'optimisation mathématique sont de deux natures :

- Les méthodes globales
- Les méthodes locales

A.1 Les méthodes globales

Elles considèrent nécessairement tout l'espace éventuellement possible. Elles divisent l'espace de définition suivant une maille, en cellules. Le calcul est ainsi réalisé à partir d'une discrétisation de cet espace. Les processus proposés permettent à coup sûr de trouver, si elle existe, la meilleure solution possible à partir du maillage envisagé, ou permet de déduire l'absence de solution au problème sur ce maillage.

Trois difficultés caractérisent systématiquement la mise en oeuvre de telles méthodes :

- Celles liés à la finesse du maillage : en effet, à supposer qu'il n'y ait aucune difficulté à définir si une maille est accessible ou non à la trajectoire, la rapidité d'obtention de la solution diminue lorsque la dimension de l'espace de définition s'étend et bien sûr lorsque augmente la finesse du maillage. Mais avec un maillage trop grossier, le calcul risque soit de ne pas trouver de solution, soit d'en trouver une dont la précision insuffisante apporterait une solution valide, mais dont l'optimisation est à la mesure du maillage.

- Celles liées à l'information disponible pour une maille, qui doit représenter de manière synthétique, sous une forme utile à la méthode, les difficultés de construction d'une trajectoire passant par cette cellule de la maille. S'il est interdit de pénétrer dans une cellule dont un point au moins engendre une collision, des "détroits" risquent de se trouver bouchés, empêchant ainsi de trouver une solution. S'il est autorisé de pénétrer dans une cellule contenant au moins un point libre, la trajectoire trouvée n'est plus absolument sûre, un blocage par collision est possible. Il est également possible d'affecter à chaque cellule une ou plusieurs valeurs représentatives, par exemple une valeur représentant la probabilité d'obtenir une trajectoire faisable sans collisions et passant par cette cellule.

- Celles liées à l'obtention des informations synthétiques pour chaque cellule de la maille : en effet, si cette information doit être obtenue a priori pour l'ensemble de la maille, le pré-traitement préalable à toute optimisation ou investigation est très lourd à obtenir. Si il n'y a pas de pré-traitement, l'optimisation commence avec des informations non significatives ou périmées, ce qui laisse présager de nombreux retours en arrière, leur nombre devant être suffisant pour assurer une pertinence raisonnable aux informations de synthèse des cellules mises en cause.

A.2 Les méthodes locales

Les systèmes évoqués par les ouvrages sont nombreux, mais aucune méthodologie générale n'est présentée. Les problèmes d'optimisation non linéaire peuvent être fort complexes, il n'existe pas encore de technique mathématique générale de solutions pour de tels problèmes, car la forme des équations tient un rôle considérable dans la difficulté de résolution. Avec des hypothèses restrictives, par exemple des hypothèses de convexité, des résultats partiels peuvent être obtenus.

Pour rendre optimaux les systèmes complexes [Cia90] [Leg80], il est proposé, sous conditions de contraintes d'égalités et d'inégalités notées respectivement h_i et g_i , de miniminiser telle fonction f, la "fonction d'objectif".

$$\text{minimiser } f(x), x \in E^n$$

avec les contraintes

$$h_i(x) = 0 \qquad i = 1, 2, ..., m$$
$$g_i(x) \geq 0 \qquad i = (m+1), ..., p$$

Les **conditions suffisantes de convergence** sont :

- 1. f d'une part, $h_1, ..., h_m$ et $g_{m+1}, ..., g_p$ d'autre part sont continues et dérivables
- 2. f est convexe, c'est à dire, $\forall \theta \in]0,1[$
  $f(\theta x_1 + (1-\theta)x_2) < \theta f(x_1) + (1-\theta) f(x_2)$
- 3. Σh_i^2 est convexe
- 4. les fonctions $g_{m+1}, ..., g_p$ sont concaves, c'est à dire, $\forall \theta \in]0,1[$
  $f(\theta x_1 + (1-\theta)x_2) > \theta f(x_1) + (1-\theta) f(x_2)$
- 5. il existe un domaine du possible $\mathcal{R}$ non-vide
- 6. le domaine du possible $\mathcal{R}$ est convexe et fermé
- 7. les fonctions représentant les contraintes ne deviennent pas infinies, c'est à dire $\exists M > 0$ tel que
  $|h_i(x)| \leq M$ pour $i = 1, 2, ..., m$
  et $g_i(x) \geq -M$ pour $i = (m+1), ..., p$

De fait la plupart des algorithmes peuvent atteindre un **optimum local**, même quand ces conditions ne sont pas remplies, si la nature concrète du problème s'y prête.

En outre, des **bibliothèques** de programmes mathématiques proposent un grand nombre de méthodes qui chacune conviennent à des cas particuliers [IMS89].

Certaines méthodes apparaissent a priori comme les mieux adaptées à la résolution des problèmes qui se posent au cours des travaux qui exigent constamment l'optimisation de fonctions subissant souvent des discontinuités ou se trouvant très souvent à la limite des domaines de définition.

Pour être véritablement opérationnel, dans la quasi-totalité des cas les fonctions elles-mêmes ne peuvent être considérées. C'est notamment le cas en robotique où n'est connue qu'une représentation au premier ordre des relations liant les mouvements d'un point du robot à ceux de ses articulations, par le moyen des matrices jacobiennes.

Ces méthodes locales éprouvent très souvent des **difficultés au voisinage des limites** formées par les contraintes, avec des risques sinon de blocage, tout au moins de progression extrêmement lente.

Remarque 10 *L'algorithme d'optimisation est souvent utilisé comme une "boite noire", c'est à dire que pour l'étude qui demandait l'optimisation de telle ou telle fonction, les solutions intermédiaires envisagées par l'algorithme d'optimisation ne sont pas être utilisées, puisque seule la solution finale importe. Ceci justifie l'utilisation de méthodes acceptant une tolérance permettant de violer momentanément certaines contraintes. Au contraire, pour la génération automatique de trajectoires, où le processus d'optimisation fait progresser pas à pas vers le but, chaque solution intermédiaire est utilisée pour construire une trajectoire : en conséquence, cette méthode ne peut être utilisée à cette fin puisqu'il y aurait alors des collisions.*

C'est pourquoi, il est intéressant de leur accorder une **tolérance**, généralement une "tolérance flexible" adaptée aux circontances, permettant de transgresser un peu les contraintes, momentanément. Ceci permet de conserver une rapidité de convergence acceptable lorsque l'algorithme d'optimisation est encore loin d'avoir cerné la solution.

Quoi qu'il en soit, si l'expression des contraintes est linéarisée autour du point considéré, ce qui est le cas des contraintes d'anticollisions dans l'espace des configurations, il ne sert à rien de se montrer excessivement rigoureux "loin" de ce point, où la linéarisation est moins pertinente.

Remarque 11 *Puisque plusieurs objectifs peuvent être assignés à une optimisation (chapitre 1.4), il est possible d'envisager comme [Dia87] une optimisation multi-critères. En dehors des cas extraordinaires, tous les critères ne peuvent atteindre leur optimum simultanément. Il convient de décider dans ce cas d'une stratégie nouvelle lorsqu'il ne devient plus possible d'optimiser tous*

les critères simultanément. Si l'optimisation est menée sur une fonction des différents critères, il conviendra soit de modifier cette fonction en fonction de la situation rencontrée, soit de choisir une fonction d'objectif reflétant parfaitement la manière dont les critères sont hérarchisés.

A.3 Méthodologies

cette section aborde les éléments principaux des deux familles de méthodes possibles. En premier lieu, les extremums et leur recherche forcément discrète dans le domaine du possible, puis les tests d'arret, obligatoirement numériques, permettant de présumer de la qualité de la solution obtenue.

Cette section aborde ensuite les outils de base de la recherche, gradient et approximatoins de la fonction d'objectifs, recherche partielle, qui donnent des directions de recherche de la solution. Enfin des artifices de calcul permettant d'obtenir des contraintes du même type ou de partir d'une solution initiale impossible.

Les deux familles de méthodologies

- Celles qui font appel aux dérivées du premier ou du second ordre
- Celles qui n'y font pas appel

Ces dernières sont certes moins rapides pour la résolution des problèmes mathématiquement les plus simples mais sont plus intéressantes dès que la difficulté ou le temps de la préparation de la gestion mathématique des problèmes sont grands par rapport à ceux du calcul lui même, ou bien lorsque la solution du problème est encore trop loin pour que des approximation aussi coûteuses soient significatives.

Les cheminements vers la solution optimale

Le point optimal est le vecteur $x^* = [x_1^*, x_2^*..., x_n^*]$ qui répond aux conditions.

La valeur correspondante $f(x^*)$ de la fonction objectif est dite valeur optimale.

La fonction d'objectif est dite "unimodale" si elle n'a qu'un extremum, "multimodale" si elle en présente plusieurs, auquel cas le problème devient vraiment très difficile à organiser car il peut y avoir des optima locaux différents de l'optimum global.

La recherche de la solution optimale doit passer par celle d'un cheminement unidimensionnel, autant que possible à l'intérieur du domaine défini par les contraintes, ainsi le concept de convexité ou de concavité se trouve-t-il être extrêmement important aussi bien pour la "forme" du champ du possible que pour celle de la fonction d'objectif.

Le domaine du possible :

C'est l'ensemble $\mathcal{R}$ de tous les points x qui satisfont aux contraintes éventuelles. L'optimum est dit un "optimum contraint" s'il se trouve sur une des frontières du domaine du possible : alors une contrainte par inégalité est dite "active" si pour ce point optimum elle se trouve être exprimée par l'égalité stricte. Si toutes les contraintes s'expriment par des égalités alors le point résultat x^* doit se trouver, s'il existe, au point d'intersection de toutes les hypersurfaces $h_j(x) = 0$ représentant les contraintes.

Test d'arrêt

Il faut aussi s'interroger sur la nature, éventuellement la rigueur, de ce test. Comme il s'agit d'un test numérique, il ne constitue pas vraiment un critère de convergence, tout au plus une présomption de convergence.

Dans beaucoup de cas le processus est réputé avoir abouti lorsque pour quelques valeurs successives de x le module de la différence entre les $f(x)$ successifs est inférieur à une précision donnée à l'avance. En toute rigueur ce critère n'est pas suffisant puisque par exemple la série $1 + \frac{1}{2} + \frac{1}{3} + ... + \frac{1}{n} + ...$ n'est pas convergente bien que la différence entre deux termes successifs devienne aussi petite que l'on veut. Mais pour les calculs, il n'y a pas de différence entre zéro et le plus petit nombre pris en compte par l'ordinateur.

Le programme s'arrête à l'itération i :

- soit quand sur un point x , aucun des essais prévus ne propose une direction susceptible d'amener à un résultat meilleur.

Dans ce cas de blocage brutal, où les conditions locales apparaissent comme défavorables, il est intéressant de tenter de débloquer la situation en cherchant une dernière fois une issue par un procédé très différent de ceux du processus ayant conduit la recherche : un essai selon des directions qui sont au hasard par rapport aux critères locaux mais qui balaient l'espace systématiquement et régulièrement peut conduire à une solution plus intéressante ; il semble bon pour une telle tentative de prendre un pas assez grand.

- soit quand il y a convergence numérique en un point x tel que $||x_i - x_{i-1}|| < \epsilon_x$ et $||F(x_i) - F(x_{i-1})|| < \epsilon_F$

  auquel cas x est considéré comme une bonne solution numérique.

Le Gradient :

C'est le vecteur-colonne formé par les dérivées partielles premières de la fonction d'objectif $f(x)$: $\nabla f(x^{(k)}) = \begin{pmatrix} \frac{\partial f(x^{(k)})}{\partial x_1} \\ \ldots\ldots \\ \frac{\partial f(x^{(k)})}{\partial x_n} \end{pmatrix}$

Le Gradient est dans la direction de la plus forte rampe (ascendante) et orthogonal aux lignes de niveau de la fonction $f(x)$, son opposé est dans la direction de la plus grande pente (descendante).

Approximations locales de la fonction d'objectif :

Elles résultent du développement en série de Taylor :

Approximation linéaire (hyperplan tangent) :

$$f(x) \approx f(x^{(k)}) + \nabla^t f(x^{(k)})(x - x^{(k)})$$

Approximation quadratique (hyper quadrique tangente) :

$$f(x) \approx f(x^{(k)}) + \nabla^t f(x^{(k)})(x - x^{(k)}) + (\tfrac{1}{2})(x - x^{(k)})^t \nabla^2 f(x^{(k)})(x - x^{(k)})$$

$\nabla^2 f(x^{(k)})$ est dite "matrice hessienne", matrice carrée formée par les dérivées partielles secondes de la fonction d'objectif $f(x)$:

$$\nabla^2 f(x^{(k)}) = Hf(x^{(k)}) = \begin{pmatrix} \frac{\partial^2 f(x^{(k)})}{\partial x_1^2} & .... & \frac{\partial^2 f(x^{(k)})}{\partial x_1 \partial x_n} \\ .... & .... & .... \\ \frac{\partial^2 f(x^{(k)})}{\partial x_n \partial x_1} & .... & \frac{\partial^2 f(x^{(k)})}{\partial x_n^2} \end{pmatrix}$$

Recherches partielles :

Par ailleurs il peut être très intéressant de pouvoir insérer des recherches partielles dans des recherches d'optimisation de plus grande complexité, notamment par le nombre de variables : un cas particulier est fréquent, la relaxation cherchant l'optimisation d'une variable après l'autre par reprises successives.

Variables d'écart

Les variables d'écart $u_i \geq 0$ transforment les contraintes d'inégalités en contraintes d'égalités :

$$\sum_{u=1}^{n} \frac{\partial g_i(x^{(k)})}{\partial x_u} \Delta^+ x_u^{(k)} - \sum_{u=1}^{n} \frac{\partial g_i(x^{(k)})}{\partial x_u} \Delta^- x_u^{(k)} - u_i^{(k)} = -g_i(x^{(k)})$$

$$\text{avec } i = (m+1), ..., p \text{ et } u_i^{(k)} \geq 0$$

où

$\Delta^+ x_u^{(k)} = (x - x^{(k)})$ si $(x - x^{(k)}) \geq 0$ sinon $\Delta^+ x_u^{(k)} = 0$

$\Delta^- x_u^{(k)} = (x - x^{(k)})$ si $(x - x^{(k)}) \leq 0$ sinon $\Delta^- x_u^{(k)} = 0$

tandis que d'autres variables d'écart $v_u^{(k)}$ transforment les contraintes additionnelles pour rester dans le domaine du possible

$$p_u^{(k)} \Delta^+ x_u^{(k)} + q_u^{(k)} \Delta^- x_u^{(k)} + v_u^{(k)} = m_u^{(k)}$$

$$\text{pour } u = 1, 2, ..., n$$

avec :

$p_u^{(k)} = \max 1, (m_u^k/(U_u - x_u^{(k)}))$ U_u =limite supérieure de x_u
$q_u^{(k)} = \max 1, (m_u^k/(x_u^{(k)} - L_u))$ L_u =limite inférieure de x_u
$m_u^{(k)}$ = pas maximum dans la coordonnée de rang u au stade k .

Variables artificielles

Par ailleurs dans le cas où le point de départ ne satisfait pas toutes les contraintes, des variables artificielles $w_i \geq 0$ sont introduites éventuellement pour obtenir un point initial possible dans le cas où le point initial de base se trouverait hors du domaine (pour les contraintes d'inégalité ces deux sortes de variables interviennent simultanément) :

$$\sum_{u=1}^{n} \frac{\partial h_i(x^{(k)})}{\partial x_u} \Delta^+ x_u^{(k)} - \sum_{u=1}^{n} \frac{\partial h_i(x^{(k)})}{\partial x_u} \Delta^- x_u^{(k)} + w_i^{(k)} = -h_i(x^{(k)})$$

$$\text{avec } i = 1, ..., m$$

$$\sum_{u=1}^{n} \frac{\partial g_i(x^{(k)})}{\partial x_u} \Delta^+ x_u^{(k)} - \sum_{u=1}^{n} \frac{\partial g_i(x^{(k)})}{\partial x_u} \Delta^- x_u^{(k)} - u_i^{(k)} + w_i^{(k)} = -g_i(x^{(k)})$$

$$\text{avec } i = (m+1), ..., p$$

Les variables artificielles sont non-nulles pour chaque contrainte qui se trouve transgressée.

La méthode du simplexe [Cia90] elle-même est utilisée pour ramener ces variables artificielles à zéro en minimisant $\sum_{i=1}^{p} w_i$.

Remarque 12 *Tant que à chaque stade k une solution est trouvée à l'intérieur du domaine du possible la méthode progresse rapidement.*

Mais quand, le processus trouve des solutions intermédiaires à la frontière de ce domaine, alors la progression devient extrêmement lente : l'algorithme tente d'améliorer le résultat concernant la fonction d'objectif en passant beaucoup de temps à satisfaire les conditions de validité correspondant aux contraintes. La convergence en suivant la frontière est ralentie car il faut remettre toutes les solutions explorées dans le domaine du possible (ce qui demande quelquefois une optimisation spécifique), ou bien limiter les recherches à un sous-espace défini par la contrainte limitante qui peut avoir une définition complexe.

A.4 Méthodes faisant appel aux dérivées premières ou deuxièmes

Méthode par approximations linéaires

Il peut être envisagé de ne linéariser que l'expression des contraintes en laissant non-linéaire la fonction d'objectif.

La linéarisation est généralement faite en remplaçant les fonctions par leur développement de Taylor au premier ordre au voisinage du point concerné $x^{(k)}$.

- Une première linéarisation à partir des conditions initiales en $x^{(0)}$ aboutit au résultat $x^{(1)}$.
- Une seconde linéarisation à partir des conditions en $x^{(1)}$ aboutit au résultat $x^{(2)}$.
- Et ainsi de suite jusqu'à "dans les cas convenables" converger vers la solution d'optimisation x^*.

Méthode de la programmation linéaire d'approximation

Le problème, en recherchant une programmation linéaire d'approximation autour du point $x^{(k)}$ consiste à

$$\text{minimiser } f(x) - f(x^{(k)}) = \sum_{u=1}^{n} \frac{\partial f(x^{(k)})}{\partial x_u} \Delta x_u^{(k)}$$

avec les contraintes

$$\sum_{u=1}^{n} \frac{\partial h_i(x^{(k)})}{\partial x_u} \Delta x_u^{(k)} = -h_i(x^{(k)}),\ i = 1, 2, ..., m$$

$$\sum_{u=1}^{n} \frac{\partial g_i(x^{(k)})}{\partial x_u} \Delta x_u^{(k)} \geq -g_i(x^{(k)}),\ i = (m+1), ..., p$$

$$\text{où } \Delta x_u^{(k)} = (x_u - x_u^{(k)})$$

en ajoutant éventuellement une condition d'approximation empêchant le nouveau point soit d'être excessivement éloigné du précédent soit de sortir des limites du domaine du possible.

ainsi :

$$\delta_u^{(k)} - |x_u^{(k+1)} - x_u^{(k)}| \geq 0 \qquad \text{pour } u = 1, 2, ..., n$$

où $\delta_u^{(k)} > 0$ est une valeur positive généralement petite qui limite l'ampleur du pas, ce qui est souvent très utile car les hyperdroites de l'approximation linéaire risquent de se rencontrer très "loin" du point $x^{(k)}$, rendant ainsi une telle approximation sans signification. $x^{(k+1)}$ étant ainsi calculé, la séquence entière est reprise avec une ampleur possible du pas $\delta_u^{(k+1)}$ graduellement décroissante jusqu'à ce que la variation dans la valeur de $f(x)$ soit inférieure à une valeur petite donnée à l'avance,

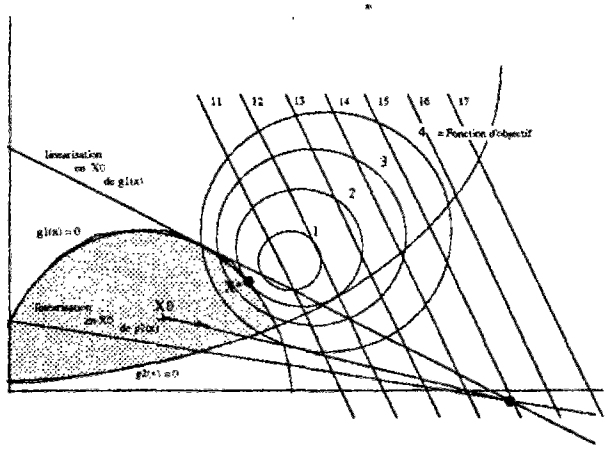


Figure A.1 : Programmation linéaire d'approximation : la direction de déplacement proposée par l'approximation linéaire peut être bonne mais le pas proposé peut être excessif.

ou, plus précisément, en tenant compte de la "distance" de $x^{(k)}$ par rapport aux limites du domaine du possible, comme l'indique la figure A.1,

$$p_u^{(k)}\Delta^+ x_u^{(k)} + q_u^{(k)}\Delta^- x_u^{(k)} \leq m_u^{(k)} \text{ pour u = 1,2,...,n}$$

où

$\Delta^+ x_u^{(k)} = (x - x^{(k)})$ si $(x - x^{(k)}) \geq 0$ sinon $\Delta^+ x_u^{(k)} = 0$

$\Delta^- x_u^{(k)} = (x - x^{(k)})$ si $(x - x^{(k)}) \leq 0$ sinon $\Delta^- x_u^{(k)} = 0$

$p_u^{(k)} = \max 1, (m_u^k/(U_u - x_u^{(k)}))$ U_u =limite supérieure de x_u
$q_u^{(k)} = \max 1, (m_u^k/(x_u^{(k)} - L_u))$ L_u =limite inférieure de x_u
$m_u^{(k)}$ = pas maximum dans la coordonnée de rang u au stade k .

ainsi $x^{(k+1)}$ ne risque-t-il pas de se trouver hors du domaine du possible.

Les difficultés rencontrées dans la programmation linéaire d'approximation

La progression est lente si les points $x^{(k)}$ successifs doivent rester dans le domaine du possible ou en être très proches. Il est difficile, voire impossible, de comparer deux points successifs si une ou plusieurs contraintes éventuellement différentes sont outrepassées par l'un ou l'autre de ces points. L'approximation linéaire peut aboutir à une direction médiocre pour la recherche soit près des arêtes formées par les limites de contraintes soit dans les zones où la fonction d'objectif présente des crêtes ou des vallées. Il est ainsi très difficile avec l'approximation linéaire d'organiser un processus d'optimisation systématiquement performant.

Programmation quadratique

L'algorithme est beaucoup plus compliqué mais nombre d'écueils sont évités notamment si la forme quadratique est définie positive.

Il peut être intéressant de rendre les contraintes linéaires et de rendre quadratique la fonction d'objectif.

Les méthodes qui utilisent les dérivées au second ordre progressent avec un rythme de convergence locale quadratique, fournissant à chaque stade à la fois la direction et le pas. Certes mais il faut calculer en chaque point, outre la valeur de la fonction, les n valeurs des dérivées premières et les n^2 valeurs des dérivées secondes lesquelles sont parfois d'expression moins simple que celle

de la fonction, ou alors les estimer par différences finies, ce qui est souvent hasardeux, puis. qui plus est, vérifier que la matrice hessienne correspond bien à une forme définie positive car sinon la configuration ne ressemble pas à celle d'une quadrique ayant un véritable minimum.

Ces méthodes trouvent leur épanouissement quand la nature des problèmes se traduit par de "belles" fonctions ou par des fonctions dont les dérivées s'expriment aisément. C'est le cas lorsque certaines variables interviennent au premier ou au second degré ou par des lignes trigonométriques sinus et cosinus au premier degré ou lorsque les variables sont au moins partiellement séparées. Mais s'il y a ici ou là quelques discontinuités dans quelques dérivées alors ce devient bien difficile.

Méthode par des fonctions de pénalisation

Ces méthodes ramènent les problèmes de minimisation avec contraintes à une succession de problèmes sans contraintes *en combinant à la fonction d'objectif une ou plusieurs fonctions des contraintes, ce qui permet de supprimer celles-ci en tant que telles.*

Dans les méthodes utilisant des paramètres, un ou plusieurs paramètres donnent plus ou moins de poids à ces fonctions de pénalisation. Dans celles n'utilisant pas de paramètres, la fonction d'objectif classique est traitée comme les contraintes additionnelles . Elle est progressivement resserrée à partir de l'information recueillie au cours de la résolution du problème.

Il est ainsi possible d'établir un véritable mur autour du domaine du possible, mais il faut cependant énormément diminuer le poids de la contrainte lorsque la solution s'approche.

A.5 Méthode par une tolérance flexible

Le concept de "proche du possible"

Souvent une très grande part de la difficulté, et ainsi du temps de calcul, a pour origine la satisfaction des conditions de contraintes pour rester rigoureusement à l'intérieur du domaine du possible. C'est pourquoi il peut être intéressant dans la marche vers la solution de pouvoir utiliser également des points certes hors

du domaine de respect strict des contraintes, mais restant cependant "proches du possible" : cette tolérance peut être choisie à chaque pas et progressivement diminuée au fur et à mesure du déroulement du processus jusqu'à n'accepter à la conclusion que des points x possibles.

Ainsi :

$$\text{minimiser } f(x), x \in E^n$$

$$\text{avec les contraintes } h_i(x) = 0, i = 1,2,....,m \; g_i(x) \geq 0, i = (m+1),....,p$$

devient :

$$\text{minimiser } f(x),\ x \in E^n \text{ avec la contrainte unique } \Phi^{(k)} - T(x) \geq 0$$

$T(x)$ est une fonction positive de toutes les fonctions représentant les contraintes prise comme mesure de l'ampleur de leur violation, $\Phi^{(k)}$ est le critère de tolérance flexible, fonction positive non-croissante, des points sommets du polyèdre flexible utilisé dans E^n au k-ième stade de la recherche ; de nombreuses définitions sont possibles pour $\Phi^{(k)}$, une très commode parmi elles est $\Phi^{(k)} = min\{\ \Phi^{(k-1)}, \theta^{(k)}\ \}$

avec $\Phi^{(0)} = 2(m+1)t$
m = nombre de contraintes d'égalité t = nombre de sommets du polyèdre initial
et
$\theta^{(k)} = \frac{m+1}{r+1} \sum_{u=1}^{r+1} ||x_u^{(k)} - x_{(r+2)}^{(k)}||$

où $r = n - m$ est le nombre de degrés de liberté de $f(x)$
$||x_u^{(k)} - x_{(r+2)}^{(k)}|| = [\sum_{v=1}^{n} (x_{u,v}^{(k)} - x_{(r+2),v}^{(k)})^2]^{\frac{1}{2}}$

$x_{u,v}^{(k)}, v = 1,2,....,n$ sont les coordonnées du u-ième sommet du polyèdre flexible dans E^n

$x_{(r+2)}^{(k)}$ est le centre de gravité correspondant à tous les sommets (les coefficients sont identiques pour tous les sommets) hormis celui pour lequel la valeur de f(x) est la plus grande $x_{(r+2)}^{(k)} = (\frac{1}{r})((\sum_{u=1}^{r+1}(x_{u,v}^{(k)}) - x_{h,v}^{(k)}), v = 1,2,...,n$

h est défini comme le numéro du sommet $x_h^{(k)}$ tel que :

$$f(x_h^{(k)}) = max\{\ f(x_1^{(k)})\ ,\ f(x_2^{(k)})\ ,...,\ f(x_{(n+1)}^{(k)})\ \}$$

de même : l est défini comme le numéro du sommet $x_l^{(k)}$ tel que :

$$f(x_l^{(k)}) = min\{\ f(x_1^{(k)})\ ,\ f(x_2^{(k)})\ ,...,\ f(x_{(n+1)}^{(k)})\}$$

En fait $\theta^{(k)}$ représente la distance moyenne des $r+1$ points $x_u^{(k)}$ au centroïde $x_{(r+2)}^{(k)}$ du polyèdre qui peut s'étendre ou se contracter. Ce polyèdre se contracte généralement quand le processus s'approche du but, mais $\Phi^{(k)}$ reste une fonction positive qui ne croît jamais

$$\Phi^{(0)} \geq \Phi^{(1)} \geq \Phi^{(2)} \geq ... \geq \Phi^{(k)} \geq 0$$

quand approche la solution du problème $\theta^{(k)}$ et $\Phi^{(k)}$ tendent l'une et l'autre vers zéro.

Quand il n'est plus possible de trouver une valeur meilleure pour $f(x)$, les sommets du polygone flexible se resserrent de plus en plus proches de ce sommet correspondant à la meilleure valeur trouvée. A la limite la concentration en un seul sommet de tous les sommets du polyèdre flexible s'effectue à la solution là où $f(x)$ devient stationnaire.

Dans ces conditions, à l'approche de la solution la valeur de $\theta^{(k)}$ devient de plus en plus petite puisque la distance moyenne des sommets au centroïde tend vers zéro, alors, comme à chaque stade, au k-ième, $\Phi^{(k)}$ est égal à la plus petite des valeurs $\Phi^{(k-1)}$ et $\theta^{(k)}$, le critère de tolérance flexible tend également vers zéro quand x tend vers la solution x^*.

Le critère de tolérance quant à l'ampleur de la violation globale des contraintes $T(x)$ peut être opportunément pris égal à

$$T(x) = +[\sum_{j=1}^{m} h_j^2(x) + \sum_{j=m+1}^{p} U_j g_j^2(x)]^{\frac{1}{2}}$$

où U_j est l'opérateur de Heaviside

$U_j = 0$ si $g_j(x) \geq 0$

$U_j = 1$ si $g_j(x) < 0$

$T(x)$ représente en quelque sorte une distance généralisée par rapport aux limites signifiant les contraintes. $T(x) \geq 0 \ \forall x \in E^n$

Si $\sum_{j=1}^{m} h_j^2(x)$, est convexe et si toutes les fonctions $g_j(x)$, pour $j = 1, 2, ..., p$, sont concaves alors $T(x)$ est une fonction convexe avec un minimum 0 atteint en tous les points du domaine du possible.

Dans ces conditions le point $x^{(k)}$ sera dit :

possible si $T(x^{(k)}) = 0$
proche du possible si $0 \leq T(x^{(k)}) \leq \Phi^{(k)}$
impossible si $T(x^{(k)}) > \Phi^{(k)}$

La statégie de l'algorithme admettant une tolérance flexible

La stratégie générale consiste à réduire $\Phi^{(k)}$ dans le déroulement des stades du processus en resserrant ainsi la région de proximité du possible par une minimisation ne prenant en compte que cette contrainte unique :

$$\Phi^{(k)} - T(x) \geq 0$$

Au stade k, $\Phi^{(k)}$ étant alors une valeur acceptée, seront acceptés les points $x^{(k+1)}$ tels que $T(x^{(k+1)}) \leq \Phi^{(k)}$, points possibles ou proches du possible, seront refusés ceux pour lesquels $T(x^{(k+1)}) > \Phi^{(k)}$ auquel cas il faudra chercher un autre point.

Tant que le nouvel $x^{(k+1)}$ n'est pas devenu celui $x_l^{(k+1)}$ pour lequel $f(x_i^{(k+1)})$ est minimum la taille du polyèdre se réduit, mais dès que un $x^{(k+1)}$ devient le nouvel $x_l^{(k+1)}$ alors reprend le processus d'expansion possible du polyèdre flexible. Le polyèdre ne se concentre en un seul point qu'au voisinage d'un point rendant $f(x)$ stationnaire.

C'est un avantage important de cette stratégie que d'admettre au début de la recherche une tolérance relativement grande, ce qui évite d'avoir dès ces stades à exagérément calculer, mais de la serrer de plus en plus jusqu'à tendre vers une tolérance nulle lorsque le recherche s'approche de la solution.

D'ailleurs cette tolérance flexible $\Phi^{(k)}$ elle-même peut être opportunément prise comme critère d'aboutissement de la recherche puisqu'elle mesure la distance

moyenne des sommets au centroïde.
Si le polyèdre est sensiblement régulier $f(x_l^{(k)}) \leq f(x^* \pm \epsilon)$ et si $\Phi^{(k)} \leq \epsilon$ alors, comme :
$\Phi^{(k)} - T(x) \geq 0$, $T(x_l^{(k)}) \leq \epsilon$, la distance généralisée par rapport aux limites étant inférieure à ϵ alors sûrement chacune des contraintes ne peut être outrepassée de plus de ϵ.

Une difficulté se présente néammoins lorsque un ou plusieurs des sommets du polyèdre flexible qui étaient dans la zone de proximité du possible au stade k_1 se trouvent encore conservés au cours du processus de remplacement des sommets (ce peut être le cas d'un sommet se trouvant très "bien" placé par rapport à la valeur de $f(x)$ mais éloigné de la limite juste en-deçà de la marge admise au début du processus). Alors au stade k_2, la tolérance ayant été resserrée, ils ne sont plus dans la nouvelle zone du possible ; il convient de reprendre ces points pour les remplacer par d'autres plus proches des points de la zone de possibilité.

En effet lorsque dès le début de la recherche, quand la tolérance est encore grande, un sommet se trouve être un point proche du possible avec une intéressante valeur de la fonction d'objectif, il devient ensuite difficile de s'en débarrasser : il semble ainsi tout à fait souhaitable de construire le premier polyèdre avec des sommets tous à l'intérieur du domaine du possible.

Peuvent même se produire des cas, par exemple lorsque la ligne de plus grande pente est presque orthogonale à une limite active du possible, où le processus se "cale" prématurément sur cette limite, comme s'il existait un frottement dû à la pression du vecteur opposé au Gradient, ne pouvant ainsi aboutir.

Il serait dans ces conditions très intéressant, comme le montre la figure A.2, de "débloquer" le polyèdre flexible en le faisant évoluer non point tant en fonction de la valeur de la fonction d'objectif mais en fonction de la fonction mesurant la violation des contraintes : une cote d'alerte pourrait être le passage du centroïde hors de la zone du possible strict.

En considérant la contribution de chacun des points à la formation de $T(x)$ soit

$$T_j(x^{(k)}) = +[h_j^2(x) + U_j g_j^2(x)]^{\frac{1}{2}}$$

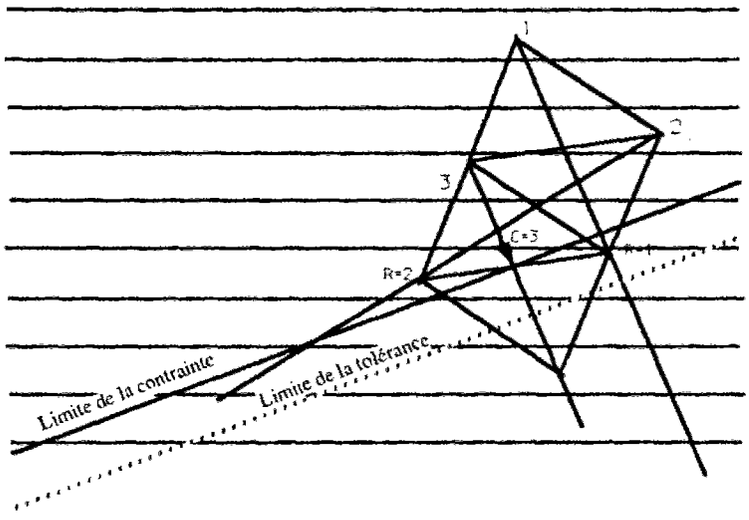


Figure A.2 : Déblocage du polyèdre flexible

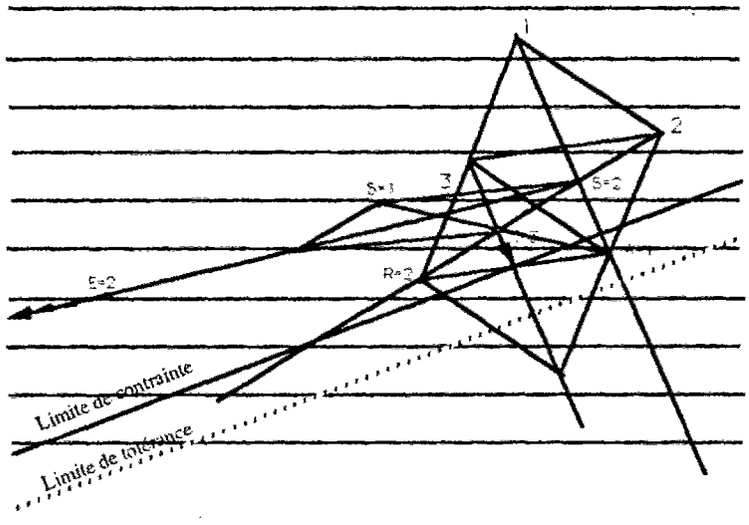


Figure A.3 : Déblocage du polyèdre flexible par réflexion

$x_m^{(k)}$ est le point, ou l'un d'entre eux s'il en existe plusieurs, pour lequel $T_m(x^{(k)})$ est le plus petit : en faisant, ce que montre la figure A.3, réflexion sur ce point de tous les sommets du polyèdre, hormis de ceux qui ainsi sortiraient du domaine de possibilité ou de proche-possibilité, le nouveau polyèdre reviendrait à une position à partir de laquelle il pourrait à nouveau évoluer.

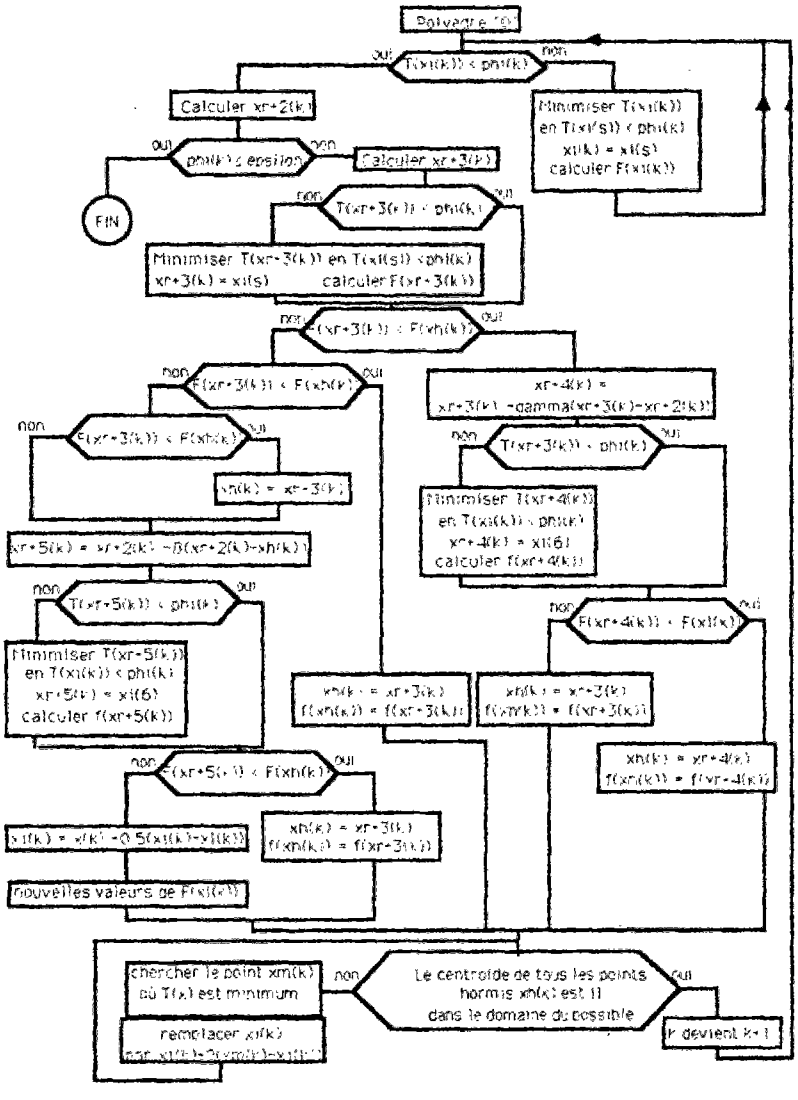


Figure A.4 : Tolérance flexible : organigramme

L'algorithme trouve une fin :

soit si $\Phi^{(k)} \leq \epsilon$ auquel cas c'est un succès de la recherche,

soit si aucun point possible ou proche du possible ne peut être trouvé, auquel cas c'est un échec : il faut essayer d'autres points de départ ou d'autres paramètres de conduite d'un stade à un autre.

la figure A.4montre le fonctionnement général de cet algorithme.

A.6 Observations sur les processus

Les particularités qui apparaissaient comme spécifiques ont été soulignées lors de l'évocation de chacune des méthodes, de chacun des processus. Aucune méthode n'apparaît vraiment comme universelle, il faut sans nul doute beaucoup d'expérience et une grande part d'intuition pour devant un problème entreprendre sa résolution sans hésitation quant à la méthode générale et plus encore quant aux processus particuliers.

Programmation faisant appel aux fonctions dérivées

Les méthodes utilisant la ligne de plus grande pente, vers la bas, ont des inconvénients qui se manifestent surtout lorsque la "forme" de la fonction d'objectif se trouve être celle d'une vallée dont il serait souhaitable de suivre la ligne de thalweg alors que dans leur quasi-totalité les lignes de plus grande pente des versants sont presque orthogonales à cette ligne de thalweg.

C'est pourquoi un trajet d'apparence hésitante voire heurtée en oscillations d'un versant sur l'autre a de fait de meilleures chances d'aboutir. Cependant ces oscillations, même si elles vont s'amortissant ce qui est le cas général, allongent très sensiblement la trajectoire. De plus elles obligent à chaque stade de faire le point d'une manière très complète sans pouvoir aisément tenir compte du sens du cheminement antérieur.

Il existe une sorte de pas optimum qui va diminuant d'ampleur au fur et à mesure du déroulement du processus car un pas trop grand exacerbe ce phénomène d'oscillations mais un pas trop petit est bien lent. Si un tel phénomène est pressenti, il peut s'avérer intéressant de rechercher avec précision, par plusieurs longueurs du vecteur pas notamment, l'intersection du pas en cause avec la ligne de fond de thalweg, ce supplément d'information n'exige pas le calcul des fonctions dérivées, il n'est ainsi pas très onéreux.

Malgré ces événements locaux, cette ligne brisée ne s'écarte pas pourtant très "volontiers" de la direction globale prise, l'exploration de l'espace se fait préférentiellement dans une sorte de cône.

Pour remédier à celà il faut pouvoir aisément "prendre des virages", les méthodes dont le pas peut facilement être adapté aux circonstances sont à cet égard les meilleures.

L'intérêt d'adapter les échelles de chacune des coordonnées

Le vecteur Gradient est affecté par un changement d'échelle pour les coordonnées mais la solution du problème ne l'est pas. L'homogénéité dans l'ordre de grandeur des diverses variables et si possible dans leur nature physique, est d'ailleurs une qualité pour tous les modèles qui représentent le monde concret.

Comme les formes étroites en vallée de la fonction d'objectif sont celles où les méthodes achoppent alors qu'elles fonctionneraient parfaitement pour une calotte sphérique, il est intéressant, encore faut-il que le problème s'y prête, de changer les échelles de représentation des facteurs pour "écarter" les rives des vallées.

Mais c'est difficile, voire impossible, pour certaines configurations : une vallée s'enroulant dans une spirale pourrait encore faire l'objet d'une homothétie mais une vallée évoluant à la manière d'un S, se prêterait moins aisément à une transformation de ce genre, par exemple pour une fonction d'objectif comportant des sinus.

Programmations ne faisant pas appel aux fonctions dérivées

La méthode par investigation directe ou par relaxation en considérant cycliquement et successivement chacune des variables est extrêmement commode pour le calcul, mais elle est comme aveugle quant à la direction pouvant probablement être privilégiée au début du calcul ; il faut cependant remarquer que loin de la solution en fait rien n'est vraiment connu, toutes les hypothèses sont hasardeuses, celles concernant les dérivées encore davantage que celles concernant la fonction d'objectif elle-même : il est ainsi intéressant d'utiliser ces méthodes "économiques" dans les premiers stades de la recherche puis de faire appel aux méthodes plus lourdes ou plus complexes quand le processus s'approche d'une solution possible.

Il pourrait également être intéressant d'utiliser la relaxation d'une manière périodique pendant l'utilisation des autres processus, ce pourrait éviter parfois telle ou telle forme de blocage à laquelle aucun n'échappe tout à fait. Pour les méthodes qui cherchent à retrouver plus ou moins des directions analogues aux directions conjuguées des systèmes quadratiques le cheminement pour aboutir à de telles directions apparaît souvent comme très compliqué. C'est encore le processus tendant, sur une direction de recherche prise à partir d'un point

donné, à bien trouver le point sur la ligne de thalweg qui semble le plus fructueux.

Peut-être pour les programmations sans contraintes la méthode utilisant un polyèdre flexible n'est-elle pas la plus performante mais elle paraît très bien adaptée aux cas de programmation avec contraintes, notamment avec contraintes d'inégalité.

A.7 Considérations plus générales

L'importance et la taille d'un problème ne peuvent être normalisés : dimension de l'espace, difficulté d'expression de la fonction d'objectif, nombre de contraintes et difficulté de leur expression, sont certes des bases principales mais interviennent également des situations très contingentes, par exemple nul ne peut vraiment savoir à l'avance si telle ligne de plus grande pente sera ou non presque orthogonale à une limite du domaine du possible.

Ceci étant les performances d'un algorithme peuvent être appréciées selon quelques critères :

- Précision du résultat par rapport à celle souhaitée : en effet certaines manières de faire parfois capables d'apporter une solution rapide mais sans précision pourraient apparaître comme plus intéressantes que d'autres moins rapides mais de plus grande précision tant pour la valeur des variables de base que pour celles des fonctions ou des contraintes. Les comparaisons sont parfois rendues très difficiles par la diversité des critères de fin de recherche.

- Nombre d'évaluations de fonctions, difficultés de ces évaluations et par là temps d'exécution.

- Simplicité de mise en place : temps exigé pour introduire les données et les fonctions dans le programme et pour faire les calculs préliminaires : en effet les programmations qui exigent de longues et difficiles préparations sont sujettes à des erreurs humaines ; c'est souvent le cas de la mise en calcul des dérivées premières et encore davantage des dérivées secondes. Par ailleurs le coût du temps de préparation est parfois excessivement

important par rapport à celui du calcul lui-même, ainsi un problème résolu par un système moins efficient mais aisé à préparer coûte en fait moins cher que celui - en principe mieux - résolu avec la haute efficacité d'un système ayant requis de nombreuses heures de préparation.

- Simplicité du programme lui-même : en effet un programme simple et clair, même un peu long, peut être facilement compris et utilisé par un autre opérateur, peut également aisément être modifié pour suivre l'évolution des problèmes posés.

Enfin aucun algorithme ne paraît capable d'aboutir systématiquement à un excellent résultat. Il est d'ailleurs toujours plus ou moins possible de fabriquer des problèmes "sur mesure" pouvant mettre en échec tel ou tel système envisagé.

Ceci étant, il est souvent intéressant qu'un système de programmation puisse s'adapter aux problèmes réels avec un certain aspect "figuratif" car c'est également une forme d'aide à la conception.

En ces sens, les programmes d'investigation par un polyèdre flexible

- sensiblement moins compliqués que les programmes utilisant les fonctions dérivées,
- mieux dirigés vers le but que les programmes d'investigation au hasard,

en restant encore assez simples, en donnant à chaque étape une sorte de point de la situation, paraissent pouvoir assez bien suivre les lignes de contraintes avec une tolérance flexible, laquelle diminue quand le processus avance, et ainsi déterminer l'optimum avec des précisions successives bien adaptées aux exigences à chacun des stades d'évolution du processus d'optimisation.

Annexe B

Proposition d'une méthode mixte géométrique et analytique pour la construction du domaine d'implantation d'un robot

Cette méthode a été développée conjointement avec le concepteur du projet industriel *Sartre*, [Yva90], qui envisage la "reconfiguration" de lignes de production flexibles. Ce projet a pour but de rép artir les tâches élémentaires (telles la réalisation d'un point de soudure) entre les robots de la ligne robotisée. De ce fait, la tâche de chaque robot évolue à mesure que les tâches élémentaires sont affectées à tel ou tel robot. l'évolution permanente de la répartition des tâches élémentaires exige un calcul rapide de l'implantation du robot de manière à rendre accessible chacune d'elle [YD90].

Dans ces conditions, il faut un algorithme qui propose des stratégies pour déplacer le robot afin de rendre accessible telle pose donnée de l'espace. D'autre part, puisque le volume d'implantation est fonction des caractéristiques de l'outil, il est souhaitable que la structure utilisée pour définir le volume d'implantation du robot permette de mesurer les latitudes encore disponibles dans la définition de cet outil.

Notre approche trouve son originalité dans l'utilisation d'une argumentation mixte (analytique + géométrique) qui assure un coût de calcul plus que raisonable allié à une flexibilité élevée du modèle résultat.

La méthode proposée n'est pas générale. Elle présuppose qu'un système de coordonnées bien adapté soit utilisé. Ce système doit prendre en compte simplement à la fois les limites de débattement du poignet et celles du porteur. *Pour illustrer ceci, le cas choisi est celui des robots à porteurs plans et à poignets à axes concourants.*

B.1 Processus fondamental

De manière imagée l'outil serait mis en place sur la pose à atteindre. Le robot y serait attaché par la platine située à son extrémité. Le corps du robot "pendrait" ainsi, les libertés de mouvement de la base faisant alors l'objet de l'étude.

Notations

Pour l'étude de tels mécanismes, il est commode de définir un certain nombre de référentiels qui nous serviront de base durant toute la suite de ce papier.

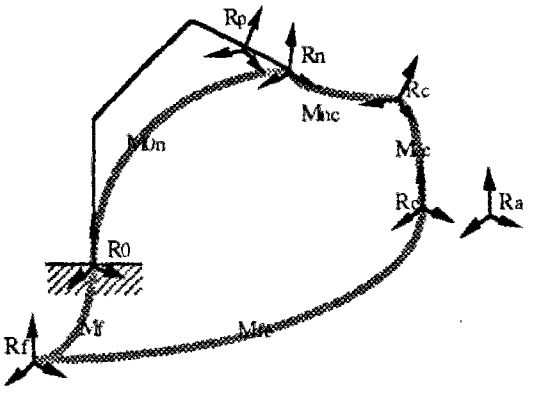


Figure B.1 : Les différents repères et matrices de transformations associées

Nous noterons (figure 4) :

- $\mathcal{R}_f$ le référentiel lié à l'atelier.

- $\mathcal{R}_0$ le repère lié à la base du manipulateur à étudier.

- $\mathcal{R}_i$ le repère lié au i^{eme} axe du robot.

- $\mathcal{R}_n$ le repère lié à l'extrémité du dernier axe du manipulateur.

- $\mathcal{R}_p$ le repère lié au centre du poignet du robot.

- $\mathcal{R}_c$ le référentiel lié à l'extrémité de la réhausse d'outil.

- $\mathcal{R}_e$ le référentiel lié à l'extrémité de l'effecteur ou repère outil.

- q_i la i^{eme} articulation de la chaine considérée.

- $\mathcal{R}_a$ le référentiel lié à la pose à atteindre.

Lorsque l'outil est en situation, le repère actif de l'outil coïncide avec le repère à atteindre. Le repère à la base de l'outil coïncide avec le repère de fixation des outils sur le robot : la position du centre du poignet est ainsi connue par rapport au repère à atteindre.

Choix du système de coordonnées :

Il dépend de la morphologie du robot. La quasi-totalité des robots industriels pour la construction automobile sont des robots dont le porteur se déplace dans un plan. Tels seront ceux qui sont étudiés par la suite.

L'outil étant en position sur un point de soudure donné, le centre du poignet reste fixe quelle que soit l'implantation du robot : il est le centre du repère utilisé.

Le système de coordonnées tient compte des symétries du problème d'orientation des corps du porteur par rapport au centre du poignet : le premier axe de rotation du robot fait un angle constant avec la verticale à cause des contraintes technologiques (influence de la gravité). Ainsi cet axe sera-t-il choisi pour axe des coordonnées cylindriques. De telles coordonnées conviennent bien à ce type de robots.

Pour le robot Acma X58, le repère est cylindrique d'axe vertical passant par le poignet, tandis que les coordonnées des points de l'espace sont repérées par (r, θ, z) comme le montre la figure B.2.

Dans ce système de coordonnées, les possibilités d'évolution du porteur sont aisément identifiables et mesurables.

Les résultats de cette étude seront contraints par la prise en compte des limitations suivantes :

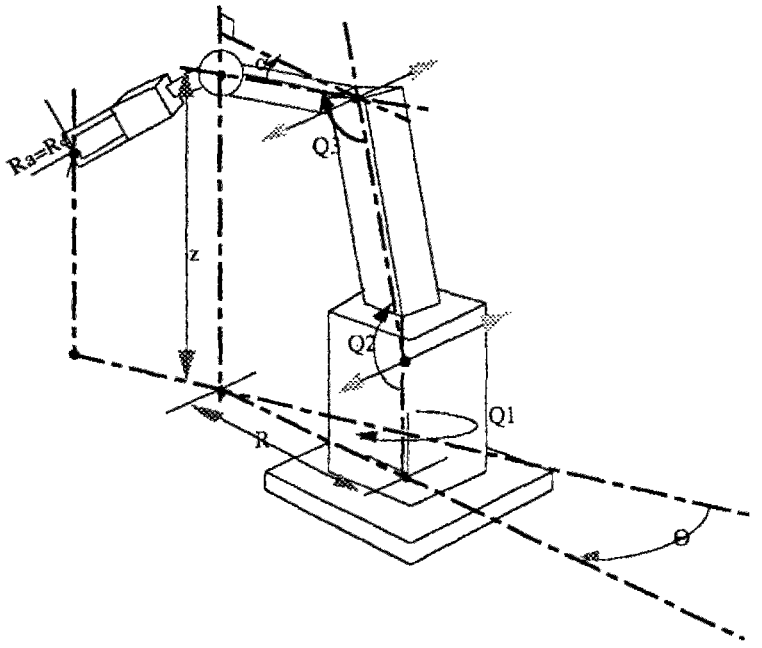


Figure B.2 : Le repère utilisé

- Les limites de débattement des articulations du poignet.
- Les limitations induites par les boucles cinématiques éventuellement présentes dans cinématique du porteur.

Pour chaque pose à réaliser, le domaine d'implantation est ainsi déterminé avec précision.

Equations du porteur :

La chaîne cinématique principale du porteur, est caractérisée par le système d'équations suivant :

$$r = f(q_i)$$
$$z = g(q_i)$$

pour un porteur à trois articulation comme celui du robot X58, on obtient (figure B.3) :

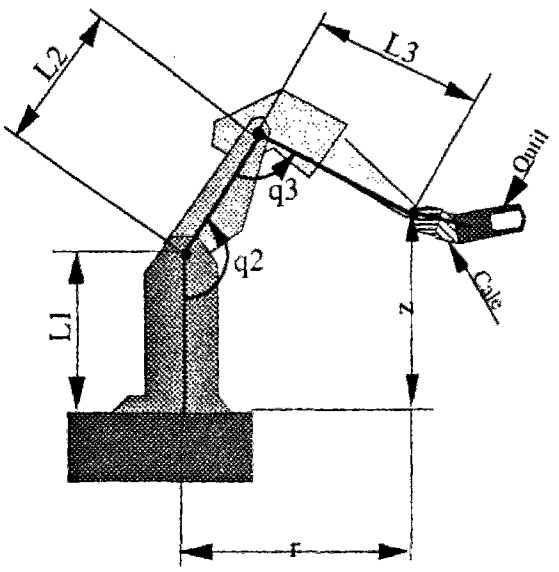


Figure B.3 : robot du type X58

$$r = L_2 \sin(q_2) - L_3 \sin(q_2 + q_3)$$
$$z = L_1 - L_2 \cos(q_2) + L_3 \cos(q_2 + q_3)$$

L'articulation q_1, rotation autour d'un axe vertical constituant la première articulation du robot, interviendra indépendamment, de manière très simple, lorsqu'il faudra faire la synthèse des résultats obtenus sur chaque pose de la tâche.

La figure B.4 représente les familles de courbes $z = g(q_2, q_3)$, où z est le paramètre de la famille, représentant physiquement l'altitude du centre du poignet par rapport à la base du robot.

Etude des rayons extrêmes

Dans le cas du robot X58, la fonction $r = f(q_1, q_2)$ est continue et différentiable. Lorsque z est donné, l'équation $z - g(q_2, q_3) = 0$ représente une contrainte.

Les extrema sont obtenus pour $q_3 = 0 \bmod \pi$. Dans le cas des robots usuels, les contraintes technologiques imposent $q_3 \in]o, 2\pi[$, la seule solution est $q_3 = \pi$. Cependant, dans le cas du robot Acma x58 qui sert d'exemple à nos propos, ces

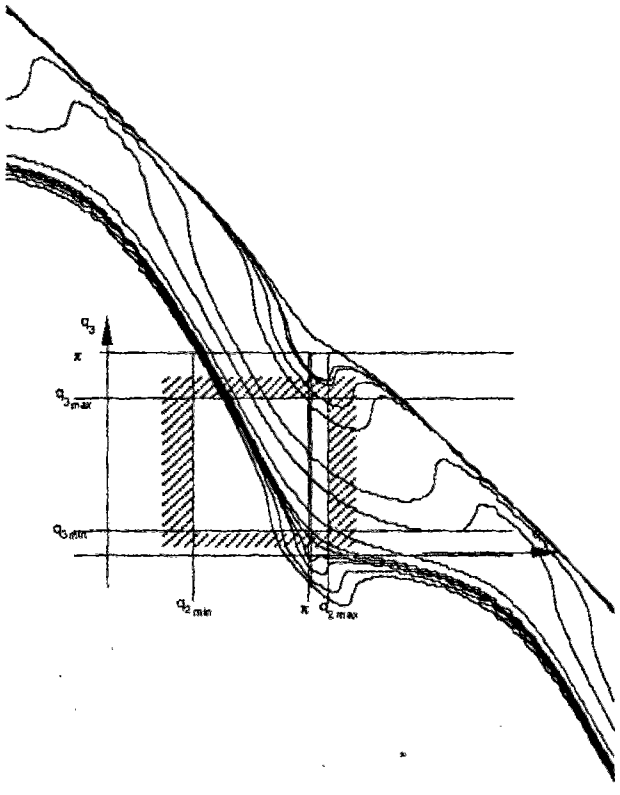


Figure B.4 : Domaine de validité et courbes paramétrées en z

extrêma de la fonction ne sont pas atteint à cause des limites de débatement de l'articulation Q^3, comme le montre la figure B.4+4.

B.2 prise en compte des contraintes

Le domaine de validité du couple (q_2, q_3) est réduit par la prise en compte des contraintes, comme le montre la figure B.5

B.2.1 débattements du porteur

Chaque articulation de la chaîne cinématique principale du porteur possède des butées. Cela s'exprime par les quatre contraintes linéaires suivantes :

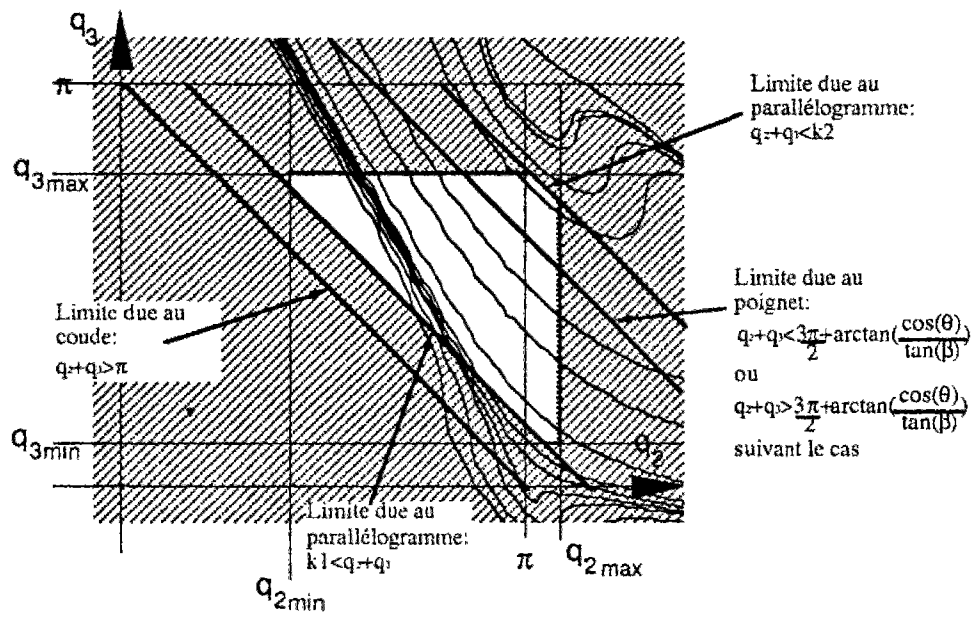


Figure B.5 : prise en compte des contraintes : courbes à z = constante

$$q_2 \in [q_{2min}, q_{2max}]$$
$$q_3 \in [q_{3min}, q_{3max}]$$

B.2.2 débattements des boucles cinématiques du porteur

Ces contraintes sont étroitement liées à la nature de la boucle fermée. Dans le cas d'un parallélogramme, comme pour le robot X58, la contrainte s'écrit :

$$q_2 + q_3 \in [q_{p_{min}}, q_{p_{max}}]$$

B.2.3 contraintes de configurations du porteur

Le repère actif de l'outil du robot peut atteindre une pose de plusieurs manières (jusqu'à 16 en théorie) comme il a été montré dans la première partie. Il est

alors possible de choisir et de contrôler la configuration du robot. Par exemple, si l'on souhaite prendre en compte uniquement les configurations "coude en haut", comme c'est toujours le cas pour le robot X58, les deux contraintes linéaires s'écrivent :

$$\pi < q_2 + q_3 < 2\pi$$

B.2.4 contraintes de débattement du poignet

L'étude est menée ici pour un poignet à axes concourants. Ce type de poignet est presque généralisé sur les robots industriels. S'il n'en était pas ainsi, l'étude suivante devrait alors être menée en fonction des caractéristiques particulières du poignet considéré.

Cette étude a pour but de décrire l'influence des limites de débattements du poignet sur les variables articulaires du porteur, comme le montre la figure B.5.

L'articulation centrale du poignet est la seule qui se trouve limitée à moins d'un tour.

Pour éviter la multiplication des différents cas possibles, qui conduiraient à écrire plusieurs fois des équations presque identiques en fonction des intervalles de valeurs considérés, il sera supposé ici que $q_{5max} = -q_{5min} = \frac{\pi}{2}$

en suivant la figure B.6 :
le vecteur directeur de l'articulation 6 est
$u = (0, \cos(\beta), -\sin(\beta))$
le vecteur directeur de l'articulation 4 est
$v = (\sin(\theta)\cos(\alpha), -\cos(\theta)\cos(\alpha), -\sin(\alpha))$

la contrainte s'écrit :

$\overrightarrow{u}.\overrightarrow{v} < ||\overrightarrow{u}||||\overrightarrow{v}||\cos(q_{5max})$

ce qui donne, puisque $\alpha = q_2 + q_3 - \frac{3\pi}{2}$:

$\forall\beta \in [0, \frac{\pi}{2}]$ $\quad q_2 + q_3 \in [\pi, \frac{3\pi}{2} + \arctan(\frac{\cos(\theta)}{\tan(\beta)})]$ l'outil pointant vers le haut.

$\forall\beta \in [\frac{-\pi}{2}, 0]$ $\quad q_2 + q_3 \in [\frac{3\pi}{2} + \arctan(\frac{\cos(\theta)}{\tan(\beta)}), \pi]$ l'outil pointant vers le bas.

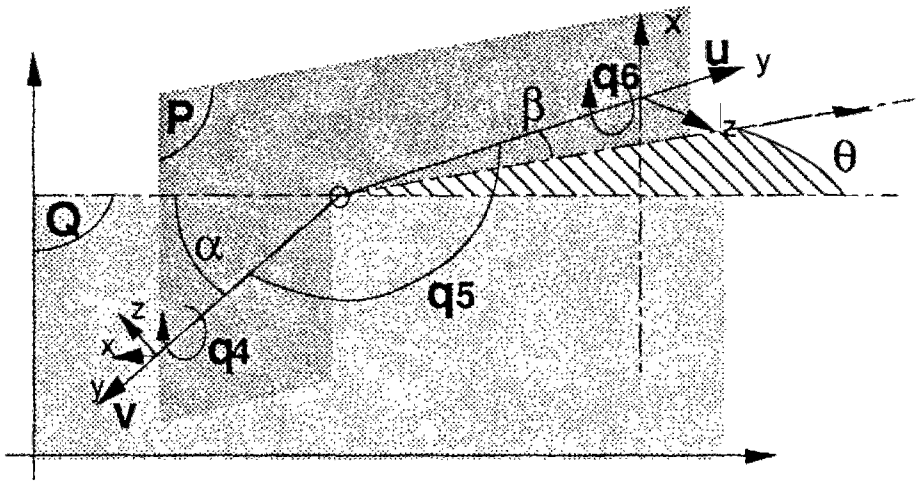


Figure B.6 : Angles caractéristiques du poignet

B.3 Etude du rayon en fonction de la hauteur

Les limites du rayon en fonction de la hauteur sont données par les contraintes prises en compte :

D'après les équations :
$r = f(q_2, q_3)$ et $z = g(q_2, q_3)$:

le domaine de validité de R en fonction de z est limité par des cercles $r = f(q_2$ ou q_3 ou $q_2+q_3, z)$ paramètrés par q_2, q_3 ou q_2+q_3 suivant la contrainte limitante.

Ainsi, en décomposant les sinus et cosinus de $q_2 + q_3$ puis en éliminant q_2 on trouve :
$r^2 + (z - l_1)^2 = l_3^2 + l_2^2 - 2l_2l_3\cos(q_3)$
C'est un cercle centré en $(r-0 = 0, z-0 = l_1)$ et de rayon $[l_3^2+l_2^2-2l_2l_3\cos(q_3)]^{\frac{1}{2}}$
le minimum est obtenu pour $q_3 = \pi$ ou $q_3 = q3max$
le maximum est obtenu pour $q_3 = 0$ ou $q_3 = q_3min$.

En conservant $q_2 + q_3$, on trouve l'équation :
$[r + l_3\sin(q_2 + q_3)]^2 + [z - l_1 - l_3\cos(q_2 + q_3)]^2 = l_2^2$ ce sont des cercles de rayon l_2 et de centres $r_c = -l_3\sin(q_2 + q_3)$ et $z_c = +l_3\cos(q_2 + q_3)$. Les centres de ces cercles sont sur un arc de cercle $r_c^2 + z_c^2 = l_3^2$, limité par les valeurs de $q_2 + q_3$.

En conservant q_2, on trouve l'équation :
$[r - l_2 \sin(q_2)]^2 + [z - l_1 + l_2 \cos(q_2)]^2 = l_3^2$ ce sont des cercles de rayon l_3 et dont les centres ont pour coordonnées $r_c = l_2 \sin(q_2)$ et $z_c = -l_2 \cos(q_2)$. Les centres de ces cercles sont sur un arc de cercle $r_c^2 + z_c^2 = l_2^2$, limité par les valeurs de q_2
.

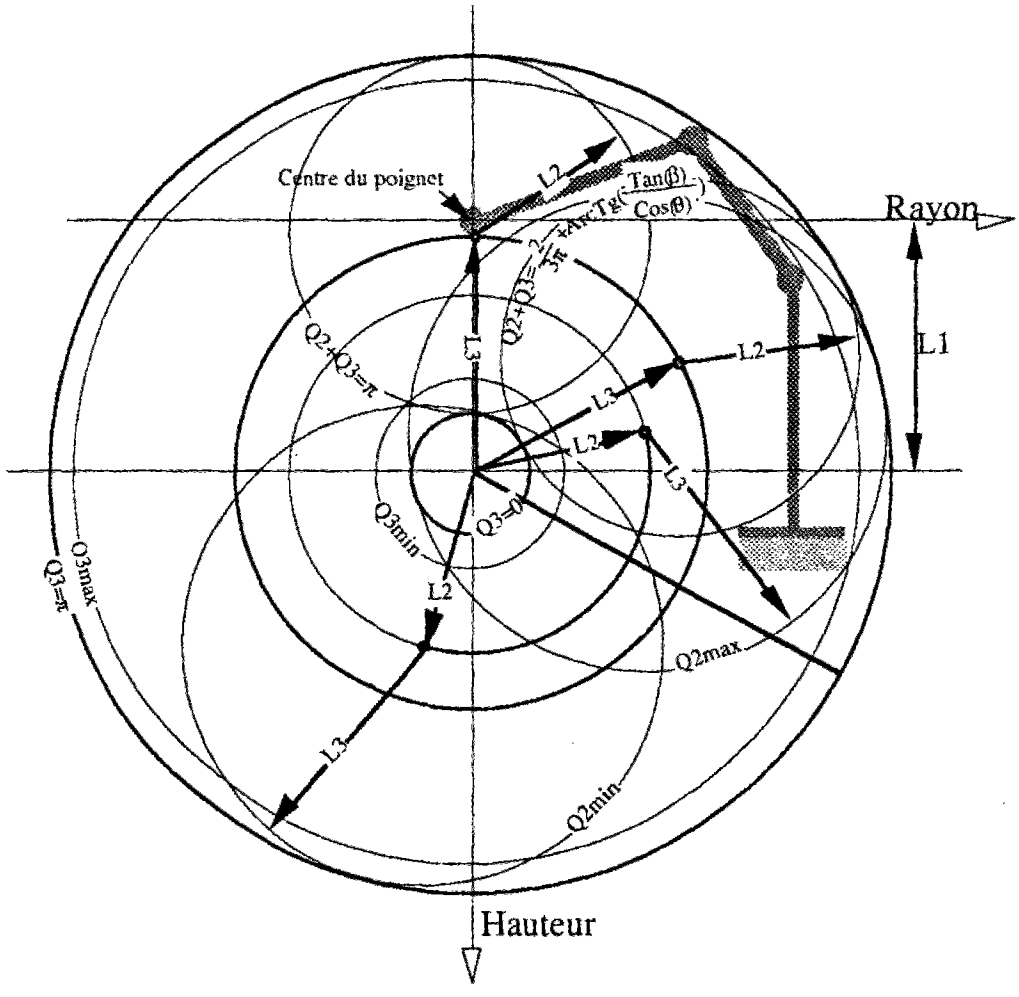


Figure B.7 : zone de validité du rayon r en fonction de la hauteur z

La figure B.7 montre graphiquement ces contraintes : la zone de validité de r est limitée par les arcs de cercles énoncés ci-dessus, le robot y est dessiné dans une position plausible.

La présence de trous dans le domaine peut être détectée facilement : dans le cas du robot X58, pour une configuration donnée du poignet, on connaît les angles θ et β, ce qui permet de connaitre la zone valide de (r, z). Ainsi, si $\beta \in [0, \frac{\pi}{2}]$, la zone valide du rayon r en fonction de la hauteur z est décrite

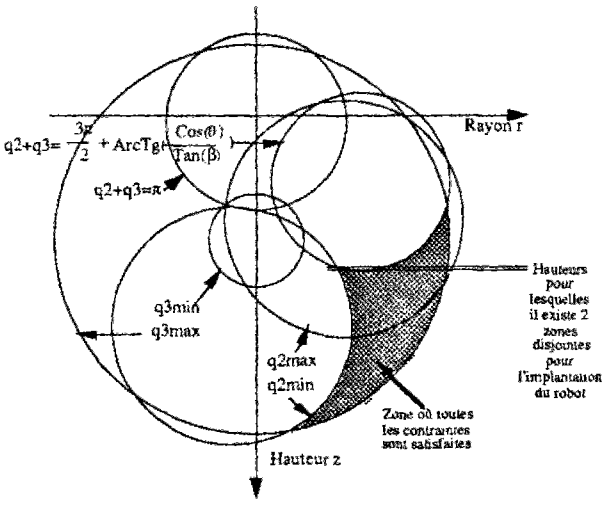


Figure B.8 : Cas ou $q_2 + q_3 \in [\frac{3\pi}{2} + \arctan(\frac{\cos(\theta)}{\tan(\beta)}), \pi]$: zone de validité du rayon r en fonction de la hauteur z

Ainsi, si $\beta \in [0, \frac{\pi}{2}]$, la zone valide du rayon r en fonction de la hauteur z est décrite figure B.9-1, ce qui correspond pour les variables articulaires à la figure B.9.

Dans ce cas, pour le robot acma x58, le domaine n'est pas connexe pour tout z donné. (pas de continuité en $R = h(z)$).

Tandis que si $\beta \in [\frac{-\pi}{2}, 0]$, la zone valide du rayon r en fonction de la hauteur z est décrite figure B.11-1, ce qui correspond pour les variables articulaires à la figure B.11.

Dans ces conditions, on constatera que le domaine d'implantation possible de notre robot est entièrement connexe quelquesoit z donné. (il n'y a pas de "trous" à l'intérieur du domaine) et est tracé pour$z$ fixé et pour chaque angle thêta en calculant les intersections entre $R = f(q_2, q_3)$ et les équations des droites délimitant le domaine de définition de (q_2, q_3) (y compris les droites engendrées par la contrainte poignet).

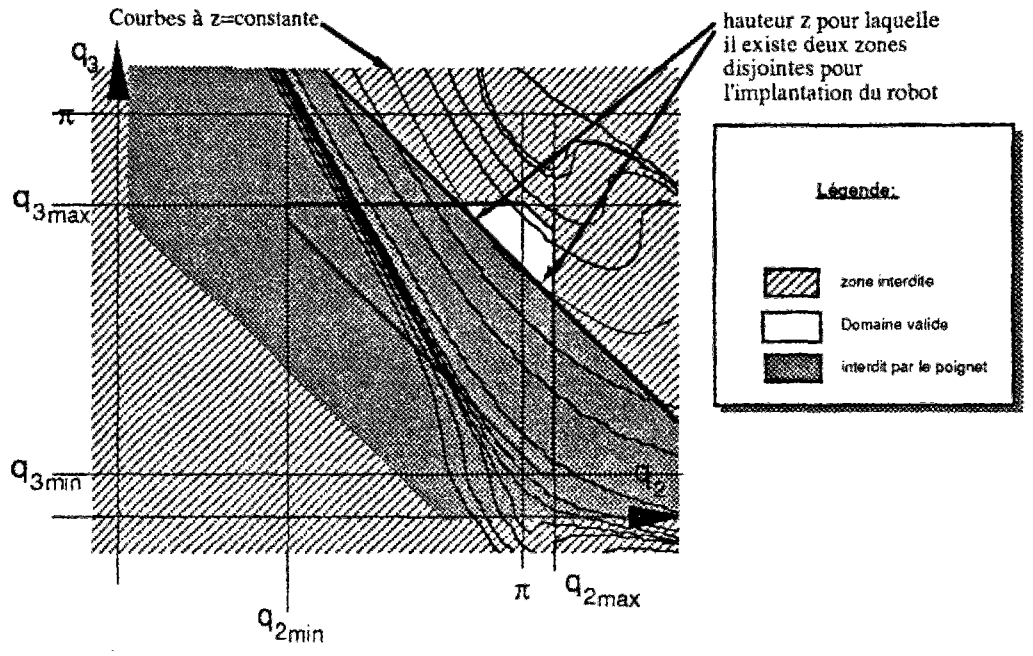


Figure B.9 : Cas ou $q_2+q_3 \in [\frac{3\pi}{2}+\arctan(\frac{\cos(\theta)}{\tan(\beta)}, \pi]$: zone de validité des variables articulaires

B.3.1 Exploitation et mise en oeuvre de l'approche

implantation automatique de robots

Cette application est surtout très utile dans le cas où le volume du domaine d'implantation est très faible ou très découpé, lorsque les méthodes de discrétisation classiques exposées précédemment ne peuvent trouver d'implantation possible qu'au prix d'une discrétisation excessivement fine impliquant un trop grand nombre de calculs /YVA 90b/.

Nous appliquons notre méthode pour l'ensemble des poses constituant une tâche discrète comme la réalisation de points de soudure (figure B.12).

Le volume obtenu permet de visualiser la zone d'implantation possible pour un robot manipulateur donné, en tenant compte de l'orientation de la base de ce dernier autour de $Z0$ (ceci n'est pas pénalisant en pratique).

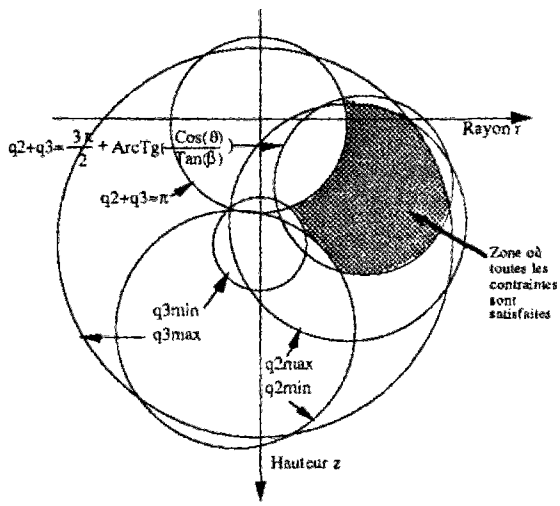


Figure B.10 : Cas ou $q_2 + q_3 \in [\pi, \frac{3\pi}{2} + \arctan(\frac{\cos(\theta)}{\tan(\beta)}]$: zone de validité du rayon r en fonction de la hauteur z

Le robot sera alors implanté à l'intérieur de ce domaine par l'opérateur (figure B.13).

Etude d'accessibilité

Nous supposons disposer d'un robot manipulateur implanté et équipé de son organe terminal. Ce robot doit effectuer une gamme composée d'un certain nombre de poses. Chacune de ces poses est caractérisée par le quadruplet (r, z, θ, β).

Pour valider l'accessibilité d'une pose, il suffira de vérifier que chaque pose repérée par (r, z, θ, β) appartient à l'un des deux domaines étudiés suivant la valeur de β.

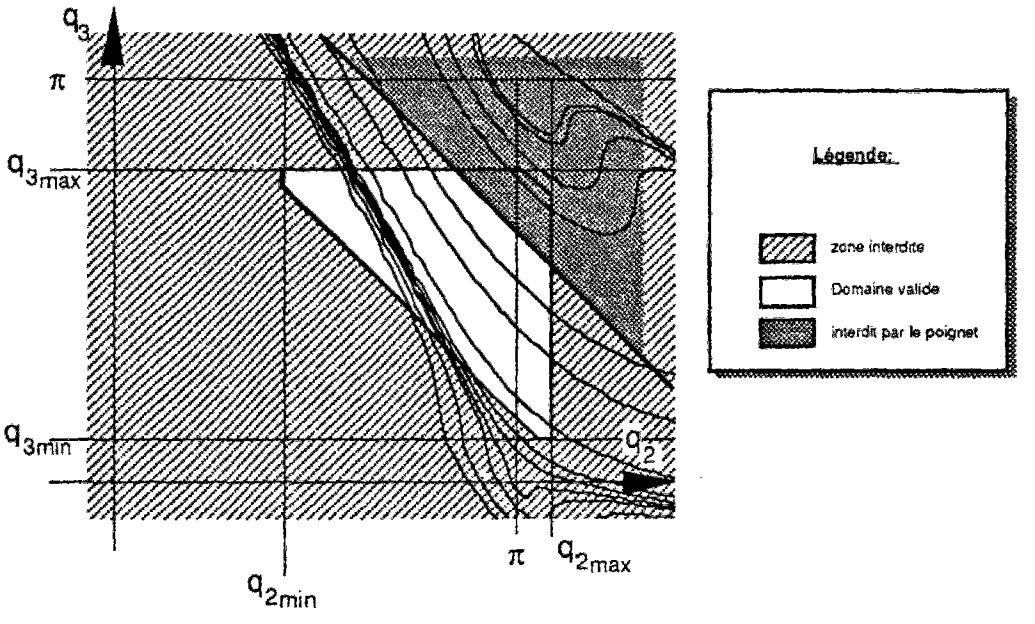


Figure B.11 : Cas ou $q_2 + q_3 \in [\pi, \frac{3\pi}{2} + \arctan(\frac{\cos(\theta)}{\tan(\beta)}]$: zone de validité des variables articulaires

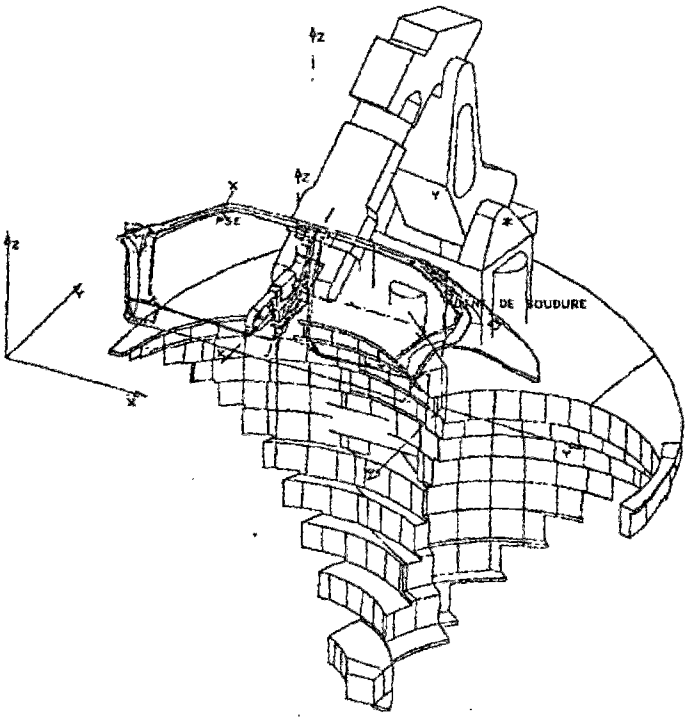


Figure B.13 : Implantation de robots : cas de deux poses

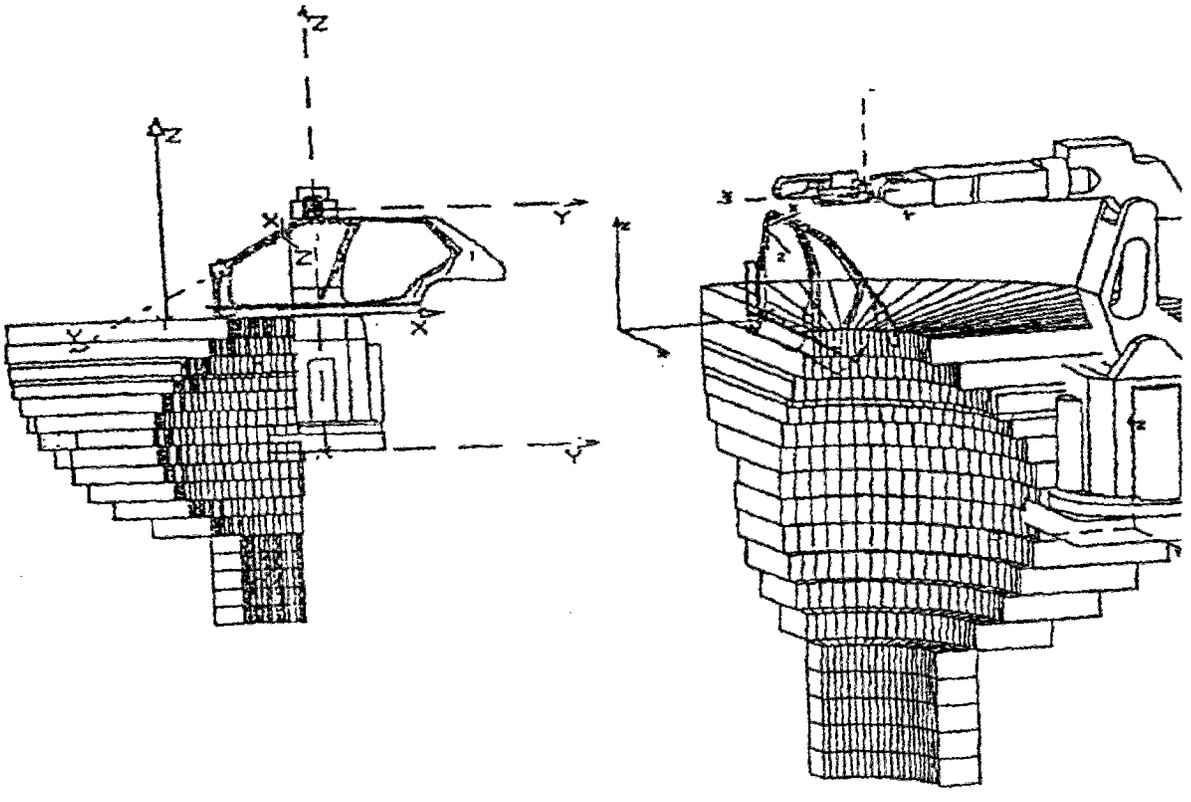


Figure B.12 : Implantation de robots : cas d'une pose