50376
1987
107

THESE

présentée à

L'UNIVERSITE DES SCIENCES ET TECHNIQUES DE LILLE FLANDRE ARTOIS

pour obtenir le titre de
DOCTEUR INGENIEUR
spécialité: AUTOMATIQUE
par
CASSAR Jean-Philippe
INGENIEUR ENSAM

POSTE DE PROGRAMMATION GRAPHIQUE DE CELLULE DE SOUDAGE ROBOTISEE.

Soutenue le 3 juillet 1987 devant la commission d'Examen:

MM	P. VIDAL	Président
	A. OSORIO	Rapporteur
	M. STAROSWIECKI	Examinateur
	E. DEFFONTAINES	Examinateur
	D. ELIOT	Examinateur

RESUME

La réalisation d'un poste de programmation graphique de cellule de soudage robotisée, objectif du travail présenté, s'inscrit dans le cadre plus vaste de la programmation des systèmes robotisés et plus généralement encore dans l'évolution de la robotique vers une plus grande fléxibilité.

Décision, programmation, perception sont trois fonctions importantes d'un système robotique. Leursobjectifs propres et leurs niveaux d'intervention dans le système sont les éléments qui permettent de définir la robotique de troisième génération. Les moyens que ces fonctions utilisent, intelligence artificielle, langages, animation graphique...,sont développés ensuite tout au long de la démarche de programmation. Celle-ci inclut les étapes de validation, détection de collision, simulation temporelle...; et de contrôle de l'exécution , sécurité, compliance...

Une analyse des causes d'erreurs géométriques dans le processus de programmation introduit la présentation de divers travaux sur la maitrise de la précision: identification des paramètres d'une structure mécanique articulée, correction automatique de programme...Une méthode particulière est ensuite développée pour déterminer la situation (position et orientation) de l'objet de travail dans l'espace de référence du robot. Cette méthode est discutée à partir d'un calcul d'erreur et de quelques essais de validation de la démarche de palpage de la pièce par le robot.

Le poste de programmation graphique, situé par rapport à d'autres travaux similaires, intègre la démarche de programmtion,les acquis d'une première réalisation et les contraintes de la soudure à l'arc pour fournir un poste de travail spécialisé à ce domaine.

Motsclé: robot, simulation, programmation, soudure, graphique, collision, précision.

SUMMARY

The achievement of a graphic work-station for arc-welding robotised cell programming, subject of this work, takes place in the largest field of robotised systems programming, and more widely in the robotic's evolution towards more flexibility.

Decision, programming and perception are three important components of a robotic system. The third generation robotics is definedthrough the goals of these elements, and their intervention levels. These functions use some means wich are presented in the programming process: artificial intelligence, languages, graphic devices...This process includes validation step, with collision detection, trajectories simulation..., and execution control step, with security, compliance...

The analysis of geometrical error causes during the programming process, are studied before the presentation of several works carried out on error limitation: geometrical parameters identification, programmed location automatic correction .... A specific method is then developed in order to determine the pose (position and orientation) of the object in the robot reference frame. This method is discussed from error estimation calculation and some object locating trials.

The programmation work-station is presented with other similar products. It takes into account the programming process, the result of a previous realisation and the arc-weldind process to provide a specialized work-station for this industrial field.

à mon père,
Le souvenir de la rigueur dont ses travaux étaient
emplis m'a soutenu dans mes efforts.

à Bernadette,
ma compagne de tous les jours. Sans son bon sens et
sa présence, il n'est pas sûr que ce travail aurait
été entrepris.

à Jérémie et Simon,
s'ils n'ont pas beaucoup vu leur père ses derniers
temps peut être comprendront-ils quand si sauront
lire ce travail. S'ils le lisent un jour...!

à ma mère,
pour tout ce qu'elle m'a transmis.

à mes amis,
présence indispensable.

aux membres du jury,

au professeur P.Vidal,
pour le soutient efficace de nos travaux et pour
la présidence de ce jury.

à Monsieur A.Osorio,
qui m'a accueilli dans son laboratoire et dont
j'apprécie l'entousiasme.

à Etiennes Déffontaines,
qui m'a permis de dégager le temps nécéssaire
à ce travail.

à Monsieur Staroswiecki,
pour sa présence dans ce jury et l'accueil chaleureux
dont nous avons plusieurs fois bénéficié.

à Messieur E.Eliot et J.L.Bréat,
pour la collaboration que nous avons entreprise et
leur disponibilité.

à Luis Peralta,
sa présence et son sens des réalités ont
accompagné mes travaux parisiens.

à mes collègues, Philippe, J.Paul, Gérard, J.Yves,
ils ont su me supporter et même souvent m'aider:
qu'ils en soit ici remerciés.

aux membres du laboratoire du LIMSI:
Daniel, Dond-mei, Eric, Francis, Jaccques, J.Claude,
Kumar, Mohamed, Philippe, Roger, Serge, ainsi que
tous les stagiaires pour leur accueil et leur gentillesse.

à toutes les personnes avec lesquelles je travaille
et particulièrement les personnels de l'ISEN et de
l'ICAM

TABLE DES MATIERES

LISTE DES FIGURES

1. INTRODUCTION.

PROLOGUE

"Au pays des aveugles, les borgnes sont rois".

Lorsque, frais émoulu des Arts et Métiers, j'ai été embauché dans une école d'ingénieurs en électronique, l'Institut d'Electronique du Nord, ISEN, mes quelques connaissances en mécanique faisaient de moi un "spécialiste". Comme tant d'autres dans le début des années 80, j'y ai goûté les joies du développement de microprocesseur. Une voie royale s'ouvrait à ma "double compétence" : la robotique !

Cette voie, à peine empruntée, se révèle déjà bien encombrée, menacée de bouchons même. Tout le monde fait ou veut faire de la robotique: cette discipline porteuse d'avenir et très à la mode, bien vue des pouvoirs publics et objet de diverses subventions.

Mais comment fait-on de la robotique ? Les avis recueillis à droite et à gauche laissent une bonne marge de manoeuvre : "on ne peut pas faire de robotique sans construire son propre robot" ou encore "la robotique est l'art d'assembler des composants existants" en passant par "la robotique doit considérer le robot comme un composant de l'automatisation dont il faut tirer le maximum".

"Qui trop embrasse mal étreint".

Toutes les pistes semblent bonnes. Alors pourquoi ne pas saisir toutes les opportunités qui se présentent ? Et l'on achète un robot industriel, d'autre part on s'imprègne de grands maîtres et l'on décide de faire de la génération de trajectoire. On aborde la modélisation par éléments finis avant de se lancer dans la conception d'un robot et de faire de la programmation hors ligne. On est presque sûr de ne passer à côté de rien. Bien que, la reconnaissance de forme...!

C'est alors que commencent les vraies découvertes :

. plusieurs mois d'étude pour un préhenseur et des moyens de découpe pour l'aluminium et, enfin !, le beau robot est programmé en deux jours.

. il peut être écrit dans un ouvrage, par ailleurs très intéressant, $A = f^{-1}(X)$. Ce f^{-1} peut coûter plusieurs jours de calculs, lorsque f est le modèle géométrique direct du robot.

. pour appliquer le merveilleux outil qu'est la méthode des éléments finis au modèle d'un robot, il faut passer par l'étude de la déformée d'une poutre en rotation. Misère !

Bien d'autres "vraies découvertes" doivent faire à la longue ce qu'on appelle l'expérience.

Apercevant, dans le même temps, les produits qui sortent des laboratoires, et devinant les moyens nécessaires aux applications mises en oeuvre dans l'industrie, le doute parfois s'installe. Y-a-t-il encore quelque chose à apporter en robotique ? Que peut faire un laboratoire qui démarre avec des moyens plus que limités ?

"On ne peut courir qu'un lièvre à la fois".

Pour faire passer des interrogations à l'évidence, nul n'est mieux placé qu'une personne qui n'est pas impliquée dans le quotidien de l'institution et à laquelle une compétence certaine est reconnue. Monsieur OSORIO du LIMSI a joué ce rôle "d'expert" il y a presque deux ans... Après le coup à l'estomac des réponses cruement données à des questions à peine formulées il a fallu modestement tout remettre à plat .... et choisir !

Choisir une branche d'activité, la soudure, choisir un domaine, la programmation hors ligne, choisir un marché les "PMI, PME".

Dans cette période délicate, le rapprochement avec le laboratoire d'automatique de l'université de Lille du professeur VIDAL fut un soutien et une source d'éclaircissements sérieux.

D'autre part, l'Institut de Soudure nous passe contrat de la réalisation d'un poste de programmation hors-ligne de cellule de soudage robotisée. Dans une perspective plus lointaine, la région Nord Pas de Calais soutient le projet d'atelier pilôte de mécano-soudure que nous avons présenté avec l'Institut Catholique des Arts et Métiers.

La route serait-elle enfin tracée?... Ou ne s'agit-il toujours que d'une piste?

INTRODUCTION.

Je ne sais si la démarche présentée dans le prologue a été celle d'autres laboratoires ou départements qui se sont "lancés" dans la robotique. Il me paraissait cependant important de situer ma thèse comme une composante de l'histoire de la création d'une activité en robotique. Car, et c'est tout l'intérêt de la recherche, le fait d'être obligé d'une part de réaliser, de proposer des solutions, d'autre part de formaliser ces solutions, de les situer parmi d'autres, amène une ouverture de vue qui permet de regarder plus loin que les besoins immédiatement formulés.

1.1 Objectifs du travail présenté.

Comme cela a été dit dans le prologue, les activités en robotique du département automatique-robotique de l'ISEN sont centrées sur le domaine du soudage à l'arc robotisé. Ce choix faisait suite à une première réalisation pour une société de la région lilloise, d'un poste de simulation graphique de scènes tridimensionnelles, suivi d'un contrat avec l'Institut de soudure pour la réalisation d'un poste de programmation hors ligne de cellule de soudage robotisée.

Ces deux travaux servent de support à cette thèse. Ils ont servi de fils directeurs pour situer la programmation hors ligne dans le domaine beaucoup plus vaste de la robotique. Leur analyse détaillée nous a permis de mettre en évidence que l'un des points critiques de la démarche de programmation sur la base de modèles était la maîtrise de la précision.

Nos objectifs sont ainsi définis : présenter les travaux réalisés ; les situer dans leurs divers domaines, robotique, soudure à l'arc, simulation graphique... ; approfondir du point de vue théorique la maîtrise de la précision.

1.2 Présentation de la démarche.

La présentation qui suit se décompose donc en trois parties :

1. Les chapitres 2 et 3 s'efforcent de faire le point sur l'état de la robotique et les pistes ouvertes pour son évolution. L'aspect programmation y tient une part importante en rapport avec le sujet du travail mené.

   Le chapitre 2 décompose le système robotique en différents modules, décision, programmation, perception. Il insiste sur les buts recherchés à travers ces fonctions et leurs niveaux d'intervention dans le système.

   Le chapitre 3 reprend, vus sous l'angle programmation, les mêmes éléments en s'attachant à les définir et en présentant les moyens qu'ils mettent en oeuvre.

2. Le chapitre 4 insiste sur un aspect important de la programmation des robots : la maîtrise de la précision du résultat. Il traite donc des causes d'imprécision, de leur remède, et propose une méthode de limitation d'une cause d'erreur adaptée à la programmation hors ligne: la détermination de la position de la pièce. Le début de ce chapitre en présentant les aspects géométriques de la programmation définit un certain nombre de notions de base de la modélisation en robotique, auxquelles on pourra se référer en premier lieu si elles ne sont pas acquises.

3. Le chapitre 5 situe et présente le travail réalisé dans le domaine de la programmation hors ligne de site de soudure robotisé. Outre le poste de programmation proprement dit, il décrit d'autres systèmes de programmation de robot par simulation graphique, le soudage robotisé, les sites expérimentaux sur lesquels sont réalisés les tests.

4. Le chapitre 6 conclut ce travail en le critiquant et ouvre des perspectives par la présentation d'un projet d'atelier flexible de pièces mécano-soudées.

1.3 Présentation des laboratoires directement concernés.

Avant d'entreprendre la lecture des chapitres suivants, il faut présenter les trois laboratoires qui supportent cette oeuvre : le département automatique-robotique de l'ISEN, la cellule robotique de l'Institut de Soudure, le groupe robotique du LIMSI.

I. Département automatique-robotique de l'ISEN.

   Parti d'une activité en mini et micro-informatique, le département automatique-robotique de l'ISEN, Institut Supérieur d'Electronique du Nord, a intégré la composante robotique en 1983. Il regroupe 4 personnes : 3 ingénieurs, un technicien. Son activité principale reste l'enseignement : de l'automatique linéaire et non linéaire et de de la robotique.

   Parallèlement (voir le prologue) se développent des activités de recherche en robotique. En collaboration avec l'Institut Catholique des Arts et Métiers (ICAM), se met en place

progressivement un atelier pilote de mécano-soudage qui sert de site expérimental aux travaux en cours.

* programmation hors ligne de cellule robotisée
* gestion de production
* pilotage d'un ensemble robotisé (simulation de fonctionnement...)
* coopération des robots

A l'heure actuelle, cet ensemble dispose de deux robots ASEA, un IRB 6.2 de charge nominale 6 kg et un IRB 90 de charge nominale 90 kg. La réalisation d'une cellule de soudage sur la base de ces deux robots est prévue pour fin 1987.

Depuis un an, la participation du département au Laboratoire d'Automatique de l'USTL a permis déjà des contacts fructueux, en gestion de production ou en coopération de robots. Les activités en robotique de ce laboratoire (développement d'un atelier flexible robotisé pour l'industrie textile) abordent des thèmes très proches des orientations choisies par le département. Cette collaboration et cette intégration ne peuvent que se renforcer.

Les contacts réguliers avec plusieurs équipes, dans le même domaine mais avec des approches, des contraintes, des objectifs différents, est une source d'enrichissements, que j'ai essayé de traduire dans cette thèse. Au lecteur de juger ...!

II. Cellule robotique de l'Institut de Soudure

L'Institut de Soudure est un organisme para-public dont la tâche est de conseiller, d'animer, de former les personnels d'entreprises dans les domaines du soudage.

La cellule robotique regroupe 2 ingénieurs et 5 techniciens sur deux lieux, Paris et Metz. Ses activités sont variées et sont réalisées sur la base de contrats avec les industriels.

1. Etude de faisabilité du soudage robotisé sur pièces réelles
2. Mise au point des modes opératoires de soudage
3. Conseil sur le choix de matériel
4. Formation inter ou intra entreprise (technique de soudage, programmation de robot...)
5. Développements d'activités particulières :
   . programmation hors ligne (langage LM et simulation graphique)
   . suivi de joint - en collaboration avec la CGE (Labo de Marcoussi)
   - animation d'un groupe travail entre laboratoires et industriels
   . adaptativité des paramètres de soudage

* Université des Sciences et techniques de Lille.

Pour cela, elle dispose de 6 robots de marque différente (Commercy, ACB, HOBART, ACMA...) et de type différent (cartésien ou polaire).

III. Le groupe robotique du LIMSI

Le Laboratoire d'Informatique pour la Mécanique et les Sciences de l'Ingénieur (LIMSI) est né en 1945 sur le campus d'Orsay. Il est composé de quatre groupes de recherche : mécanique, informatique graphique, communication parlée, robotique.

L'orientation principale du groupe robotique est le développement d'un environnement de programmation pour la robotique. Ce thème se divise en trois activités :

* perception avec le développement d'algorithmes d'extraction de contour, de reconnaissance, et l'intégration d'une auto-adaptativité aux conditions de prise d'image.
* la programmation avec la participation au développement d'une librairie de contrôle de robots pour le langage C (RCCL) et le développement d'outil de programmation de haut niveau associant LISP, et d'autres langages (SIMIR).
* une activité simulation à partir d'un logiciel de modélisation tridimensionnel MOVIE.

Une cellule flexible d'assemblage sert de site expérimental. Elle comprend un robot TH8 (ACMA) et un robot cartésien développé dans le laboratoire RTL. Le développement des applications liées à ces trois activités est réalisé sur des machines fonctionnant sous UNIX (HP 9000 sous Hp-ux et SM90 sur SPIX).

Les modules exécutables développés sur SM90 peuvent être transférés par une interface parallèle, vers une structure informatique décentralisée bâtie autour de microprocesseurs 68000 sur Bus VME. Sont ainsi réalisées la commande des robots, l'acquisition et le traitement d'image...

En 1987, l'intégration du groupe robotique du LIMSI dans le Centre de Robotique Intégrée de l'Ile de France (CRIIF) par le rapprochement qu'il permet avec d'autres laboratoires (LM Paris VII, ENSAM...) enrichit encore l'expérience acquise.

1.4 Avertissement.

Dans les chapitres qui suivent, je n'ai pas voulu détailler toutes les notions abordées. Les présentations seront donc des synthèses, orientées vers la réalisation finale de chaque domaine. Celles-ci seront forcément restrictives dans leur présentation, la bibliographie proposée a pour but de permettre au lecteur un approfondissement des thèmes qui l'intéressent.

2. LA ROBOTIQUE DE TROISIEME GENERATION.

Le premier robot est né en 1961 de l'apparition "des composants électroniques et des techniques digitales". Fruit de cinq années de travail de Georges C Deud et Joe Engelberger [ENG 86], le premier UNIMATE, robot hydraulique, a débuté "au service d'une machine à mouler en matrice pour General-Motor". De nombreuses applications de service de machines allaient suivre cette première : service de machine-outils, de presse... Puis vint la palétisation, dépalétisation. Les cycles restaient figés car les moyens de programmation étaient rudimentaires.

La robotique de deuxième génération est née de l'amélioration des moyens de calcul due aux progrès de la micro-électronique. Le robot peut prendre en compte des informations externes, les programmes et les trajectoires deviennent plus complexes et plus souples. Les applications intègrent la vision, des capteurs par apprentissage : finition, ébavurage, polissage, soudure, insertion.

La robotique de troisième génération vise la flexibilité de l'ensemble robotisé. Le système robotique se substitue au composant robot pour assurer la réalisation d'objectifs de production.

Quatre fonctions principales peuvent être dégagées d'un système robotique : action, perception, communication, décision. Chacune de ces fonctions est en liaison avec les autres mais il peut être utile de leur attribuer des buts.

- Les fonctions "action" : réalisent les déplacements des structures articulées par l'asservissement en position, vitesse ou couple de chaque articulation. Elles commandent aussi certaines actions tout ou rien (verin, ...).
- Les fonctions "perception" : elles fournissent des informations sur le monde réel dans lequel doivent évoluer les actions à accomplir par le robot.
- Les fonctions "communication" : gèrent les échanges d'informations avec l'opérateur, communication homme-machine, ou avec d'autres systèmes informatiques,(de gestion de production par exemple) communication machine-machine.
- Les fonctions "décision" : autorisent la prise en compte des fluctuations de l'état de l'environnement , des objectifs de production de l'ensemble robotisé. Cette prise en compte se fait à l'initialisation des actions ou en cours d'exécution.

Ce chapitre présente trois de ces fonctions dans l'ordre suivant : décision, programmation, perception. Il présente les objectifs de chacune, les niveaux de mise en oeuvre que l'on peut rencontrer en robotique. Les moyens utilisés par les fonctions décision et programmation sont détaillé s au chapitre 3. La perception n'est pas reprise par la suite et fait l'objet dès ce chapitre d'une présentation plus complète.

2.1 DECISION.

2.1.1 Besoin de décision.

L'objet d'un système robotisé est la modification d'un état des choses. Cela peut se traduire par un déplacement d'objet ou d'outil, un assemblage, un usinage... Les modifications à apporter sont autant de tâches pour le système qui peuvent elles-mêmes être décomposées en sous-tâches.

La réalisation de ces tâches impose de "déterminer ce qu'il faut faire" pour passer d'un état initial à l'état modifié. Ce qui est la définition même de la décision.

La question ne sera donc pas de savoir s'il faut ou non de la décision dans un système robotique. Elle existe de fait. Il faut par contre se demander :

- quel est l'opérateur de la décision ?
- quel lien établir entre information et détermination des actions ?
- comment décomposer la fonction décision ?

2.1.2 Les opérateurs de la décision.

Dès qu'il y a action, il y a eu décision. L'homme a toujours organisé son activité et déterminé la décomposition de la réalisation de son ouvrage. C'est l'étape de l'artisan.

L'industrialisation a amené l'automatisation, source de productivité. Les séquences d'exécution des tâches sont alors définies par la conception du produit et des machines.

La programmation des machines autorise un séquencement plus riche et plus souple que celui de la conception mécanique en électromécanique. Le décideur est le programmeur plus que le concepteur.

L'homme a donc toujours été, jusque dans les premières étapes de la robotique, l'opérateur de la décision : le concepteur ou le programmeur mais aussi l'opérateur d'exécution auquel incombe souvent un rôle important et souvent méconnu de contrôle des opérations !

La recherche d'une flexibilité encore plus grande, d'une fiabilité dans les opérations de programmation amène à envisager de "remplacer l'homme dans son activité intelligente" [FOX86]. L'intelligence artificielle s'est fixée pour objectif l'étude du raisonnement humain et sa formalisation en vue de sa reproduction par un système informatique. Celui-ci pourrait alors devenir opérateur de décision.

Le conditionnel employé ici signifie que les outils de l'intelligence artificielle (voir chapitre IV) ne sont à même de résoudre que quelques problèmes particuliers en robotique (reconnaissance, séquences de mouvements finis...) mais pas de prendre en compte l'ensemble d'un système robotique.

2.1.3 Objectifs.

Le but alloué au système robotique est le succès de la tâche. L'état final doit correspondre à l'ordre passé. L'intégrité de l'ensemble du système de production doit être conservée (pas de casse !). Pour y parvenir, des séquences d'action doivent être exécutées, et la fonction décision doit intervenir pour :

- définir les actions nécessaires à l'accomplissement de la tâche ;

- ordonner leur exécution ;
- contrôler leur déroulement ;
- éventuellement modifier les séquences d'action prévues.

Elle doit pour cela tenir compte :

- de l'état du système (actions en cours, position, ...) ;
- des ressources disponibles à l'instant donné ;
- de connaissances sur les conditions d'exécution des tâches, des actions (priorité, moyens à mettre en oeuvre...) ;
- de l'évolution des requêtes adressées au système robotisé.

Le traitement de ces informations sera d'autant plus souple que le système sera capable d'apprendre, c'est-à-dire de faire le lien entre l'exécution d'une action et la modification des informations.

2.1.4 Mise en oeuvre de la décision.

Nous avons vu que la décision peut inclure plusieurs objectifs : réalisations d'une tâche, maintien de l'intégrité de l'ensemble. Elle peut donc intervenir à des niveaux différents qui vont faire intervenir les facteurs temps, moyens disponibles, ...

2.1.4.1 Niveau tactique.

Des moyens matériels sont nécessaires pour arriver à accomplir des actions. Le choix des moyens mis à la disposition du système robotique constitue une première étape de décision qui conditionne les possibilités des suivantes. Parmi les dispositions tactiques, on peut citer :

+ le nombre de robots, leur capacité
+ les capteurs utilisés
+ les outils : préhenseurs, moyens de découpe...
+ .....

2.1.4.2 Niveau stratégique.

Les moyens étant déterminés, la stratégie constitue, si nous reprenons sa définition dans la théorie des jeux, "l'ensemble des décisions prises en fonction d'hypothèses de comportement dans une conjoncture déterminée". Dans ce cas, l'opérateur de décision doit connaître les méthodes qui permettent d'aboutir à une séquence d'action en fonction du but à atteindre, des hypothèses de comportement (modèle des différents composants du système robotique), de la conjoncture (informations sur l'état du système robotique). La résolution d'un problème donné implique à la fois une stratégie à long terme et des stratégies à court terme.

1. Approche logistique:

   Elle développe dans un horizon à moyen ou long terme les opérations matérielles nécessaires pour arriver au résultat. Elle est fortement déterminée par le but à atteindre et présente l'intérêt de pouvoir prévoir les moyens nécessaires à un instant donné, et d'éventuels conflits entre sous-buts.

2. Approche comportementale

   Elle réagit à une sollicitation externe ou interne de manière immédiate. La réaction dépend moins du but à atteindre que de la nature et de l'amplitude de la sollicitation. Sa

rapidité d'action permet des actions de contrôle et de sécurité... Pour pouvoir agir de manière efficace et rapide, l'approche comportementale spécialise l'opérateur de décision en établissant un lien très étroit entre sollicitation et action.

2.1.4.3 Communication.

La fonction décision va faire intervenir des opérateurs variés. Des hommes dans les phases de conception, de mise en oeuvre d'exploitation vont intervenir au niveau tactique (conception, choix de moyens) et stratégique (ordonnancement, surveillance...). Ils peuvent être aidés ou remplacés dans certaines de ces interventions par des supports informatiques plus ou moins spécialisés.

Le problème de la communication se pose pour la transmission d'informations d'un niveau à l'autre (par exemple conception et exploitation) mais aussi d'un opérateur à l'autre.

- L'aspect communication homme-machine prend une place d'autant plus importante que la décision et le contrôle peuvent être répartis entre un opérateur humain et un système automatisé. Les moyens de cette communication sont : les simulateurs, les affichages graphiques, la synthèse vocale...
- Il ne faut pas non plus négliger, bien qu'elle ne soit pas l'objet de ce travail, la communication dans l'entreprise, facteur de motivation, d'échanges souvent précieux d'information entre les différents niveaux de décision humaine et technique.

De la communication entre moyens informatiques va dépendre aussi la performance des systèmes de décision. Réseaux locaux, bus, mémoire partagée les différentes techniques peuvent cohabiter pour répondre plus précisément aux besoins.

Dans tous les cas, plusieurs questions doivent se poser :

a. Comment représenter l'information ? Cette représentation fait appel à un modèle de la réalité. Est-il connu du destinataire ?

b. Quels sont les destinataires de l'information ? Les opérateurs de décision qui en ont besoin.

c. Quelle est l'information pertinente à transmettre ? Pour ne pas saturer les moyens de communication et les noeuds de décision, l'information transmise doit être concise et suffisamment riche.

La réponse à ces questions, et donc le traitement des communications, constituera donc un élément à part entière de la fonction décision

2.2 PROGRAMMATION.

2.2.1 Objectifs de la programmation.

La programmation doit permettre de définir une séquence d'actions à éxécuter par l'ensemble robotisé, le plus souvent par la commande du robot, et de les enregistrer sous une forme permettant l'éxécution future du programme.

Les divers types de programmation vont donc se distinguer par :

- les moyens fournis pour définir les actions ;
- la nature des actions qui peuvent être prise en compte ;
- le mode de stockage et les traitements nécéssaires avant l'éxécution.

Ces différents points seront détaillés au chapitre III mais nous pouvons déjà établir un critère qui rendra un système plus ou moins opérant à utiliser: la souplesse. Celle-ci peut se traduire par :

- la facilité (souvent exprimée en temps) de la reprogrammation d'une tâche;
- la fléxibilité possible de l'outil informatique lui-même pour la prise en compte des nouvelles tâches;
- le niveau de compétence et de spécialisation nécessaires pour maîtriser la programmation.

2.2.2 Les niveaux de programmation des robots.

Par les objectifs de sa programmation, un ensemble robotisé ne se distingue pas des autres systèmes programmés (ordinateur, automate programmable). Les "actions" générées par un programme robot sont cependant plus variées et il est possible de décomposer les concepts qu'elles représentent en quatre niveaux d'abstraction.

a. Le niveau actionneur

   Ce niveau manipule directement des consignes d'asservissement (déplacement ou efforts), ou des informations logiques sur des actionneurs binaires (vérins, pince...).

b. Le niveau effecteur

   Les consignes lient un repère associé à l'extrémité du manipulateur (effecteur) à un repère cartesien de référence. Il impose l'existence d'un transformateur de coordonnées exprimant les relations entre l'attitude de l'effecteur (position et orientation) et les coordonnées articulaires du manipulateur.

c. Le niveau objet

   L'action définie n'est plus la position de l'effecteur mais la ou les relations qui doivent être réalisées entre des objets donnés. La description des objets, de leurs positions, et la définition d'une séquence de relations permet la génération des actions.

d. Le niveau objectif

   A ce niveau, la programmation fournit la description de l'état initial (objets et positions) et celle de l'état final (objectif). Le système génère une séquence d'actions à partir de règles qui dépendent de la tâche à accomplir, la définition de ces règles fait partie du processus initial de programmation.

Cette classification n'est pas rigide et un même programme pourra faire intervenir les différents niveaux. Elle permet cependant de mettre en évidence le passage de modes de programmation directement accessibles (niveau a et b) mais peu souples à des outils plus performants, mais qui nécessitent une maîtrise d'un plus grand nombre de connaissances (relations, règles) et de leur formulation en vue de la programmation.

2.3 PERCEPTION.

Le système de perception fournit, à d'autres sous-systèmes de l'ensemble robotisé (décision, programmation, sécurité...), les informations sur l'état de l'environnement.

Il est possible d'envisager que le système de perception, ou éventuellement le processus contrôlant un capteur particulier fournissent l'information à la demande. Cette activation par "l'extérieur" de la perception impose des délais de réponse dûs au temps de traitement des capteurs.

Comme M. O. Shneier [SHN86], nous pouvons fixer au système de perception l'objectif de maintenir à jour un modèle de l'environnement qui corresponde à l'évolution du monde physique réel. Ce modèle est directement accessible aux autres systèmes. Il est donc possible de dégager quatre processus qui constitueront le système de perception.

- *le processus de prédiction :*
  génère, à partir d'un modèle de l'environnement, les informations que l'on est en droit d'attendre de capteurs qui l'observent;
- *l'analyse:*
  active des processus liés aux capteurs en fonction des informations attendues et analyse les signaux qui en sont issus ;
- *le processus de comparaison:*
  établit les différences entre informations reçues et informations attendues ;
- *le processus de description:*
  actualise le modèle de l'environnement en fonction des différences et erreurs établies.

2.3.1 Nature des informations.

Le système de perception répond aux exigences du système de décision en maintenant, grâce à un aspect décisionnel propre, la représentation de l'environnement. Il est possible de définir la nature des informations qui peuvent être extraites de l'environnement en répondant à quelques questions de base :

1. *Y a-t-il un objet ?*

   L'information présence d'un objet indique qu'une zone de l'espace est occupée. Les notions de zone et d'objet doivent être comprises dans un sens large. La zone peut aller d'une droite à un angle solide. L'objet peut être une pièce, un bras de robot ou un opérateur humain.

2. *Quel est l'objet ?*

   L'information sur l'identité de l'objet suppose que l'on sait le reconnaître. Cette tâche de reconnaissance implique la recherche d'une correspondance entre un modèle de l'objet à identifier et les informations dont on dispose à son sujet.

3. *Où se trouve l'objet ?*

   L'information de position d'un objet est nécessaire dès que l'on veut exécuter une action sur cet objet. La précision de la mesure de cette position, définie par une translation et une orientation dépend de la précision nécessaire à l'éxécution de la tâche.

4. *Quelle interaction de l'objet avec d'autres objets ?*

   Cette information sur les efforts de contact est utile pour certaines applications qui mettent en oeuvre soit la présence d'un effort, soit la nécessité de son absence. L'effort de

contact est défini par un torseur.

2.3.2 Les moyens de perception.

Les informations que nous venons de définir peuvent provenir de divers moyens de perception. Il est possible de concidérer que certaines informations(identité de l'objet, présence...) soient connues à priori ou fournies par un système amont de la cellule (gestion de production, ordonnancement...). Dans ce cas, les capacités d'adaptation de l'ensemble du système robotique seront réduites.

Les moyens de perception peuvent être classés en deux catégories à partir de la richesse des données disponibles à partir de ces moyens, et du temps de traitement de ces données.

1. Capteurs d'informations particulières.

   Ces capteurs, capteurs efforts,cellules photo-électrique, proximètres..., fournissent un nombre limité de valeurs numériques souvent liées à une information particulière. Ils fournissent leurs données rapidement et celles-ci peuvent être exploitées en première priorité par les dispositifs de sécurité ou des systèmes utilisant les boucles perception-commande (compliance active - voir chapitre III, commande ou arrêt de processus).

   Ils peuvent être utiles au système de perception tant par la redonnance qu'ils introduisent dans les informations que pour "diriger" le processus d'analyse sur telle ou telle zone de l'environnement.

2. Capteur d'informations globales.

   Ces moyens, caméra vidéo ou CCD, revêtement tactile, capteur 3D..., fournissent des blocs de données de taille importante. De cette masse d'informations très riche, il faut extraire les renseignements pertinents. Ces processus d'extraction demandent de grandes puissances de calcul si l'on veut obtenir des temps de réponse de l'ordre de la seconde (pour les problèmes les plus complexes). Ce délai est souvent compensé par la densité de l'information obtenue (identité, position...).

L'intégration dans un système robotisé de l'aspect perception va nécéssiter le choix de moyens de perception variés. Ces choix se feront en fonction d'impératifs suivants :

- recherche de la redondance d'information ;
- possibilité d'utiliser un même moyen de perception pour des fonctions variées ;
- rapidité du traitement nécéssaire.

2.4 CONCLUSION.

La présentation que nous venons de faire, par l'aspect didactique de la décomposition en fonctionalités séparées, peut masquer l'importance de leur intégration en un seul système. Cela signifie que, s'il est important que dans chacun des domaines évoqués une recherche propre soit menée, et elle l'est sans doute, il faut par ailleurs éviter qu'un ensemble robotisé soit la juxtaposition pure et simple de résultats de chacune de ces activités.

Le rôle de la robotique est de raisonner sur le fonctionnement global de l'ensemble vers un but donné. Les performances de l'ensemble dépendent autant de la coopération à établir entre les différents systèmes que des performances propres de chacun d'eux [OSO85].

Il faut répartir les attributions de chaque fonction. Nous avons tenté de le faire en leur allouant des objectifs, mais il peut y avoir certains recouvrements entre décision et programmation, décision et perception, programmation et perception. Ces recouvrements peuvent apporter une redondance intéressante dans le traitement ou impliquer des niveaux de traitement différents de la même information. Cependant, ces recouvrements ne doivent pas amener à des ordres d'actions contradictoires de la part des différents modules. Ceci peut être assuré par une fonction décision de haut niveau par exemple.

A cette répartition des attributions doit être couplée une étude des communications entre les modules. Il faut d'une part définir la nature des informations à communiquer: l'information pertinente pour tel module de décision ne le sera peut-être pas pour le module de perception, et les moyens informatiques de cette communication d'autre part.

La robotique devient donc de plus en plus l'art d'envisager une configuration matérielle performante (robot, peri-robotique...) et de la coupler à une configuration informatique adaptée. Un défi !

3. PROGRAMMATION HORS LIGNE ET EN LIGNE.

Le développement de la robotique et son extension à des activités industrielles variées passe par la flexibilité de l'installation robotisée. Cette flexibilité est déjà présente dans l'architecture physique des robots eux-mêmes et de leur commande : on parle même, injustement sans doute, de "robots universels". "Mais cette souplesse ne peut être exploitée convenablement que si le système peut être programmé facilement. Le coût de programmation d'une nouvelle application est en effet une fraction importante du coût total [OSO85].

Le moyen traditionnel de programmation des robots est l'apprentissage. Cependant, le développement des outils de programmation en informatique, langages de haut niveau, intelligence artificielle... ouvrent à la robotique les perspectives d'une plus grande souplesse de programmation. Une différence importante existe entre les deux domaines : la programmation en robotique manipule des entités physiques, robot, capteurs... dont il faut prendre en compte les capacités. Par exemple, on peut toujours programmer d'un point de vue informatique n'importe quel déplacement dans l'espace, encore faut-il que le robot soit capable de le réaliser!... Une étape de validation de la programmation hors ligne est donc indispensable. Cette étape présente aussi l'intérêt de pouvoir estimer les performances du programme réalisé.

L'utilisation de modèle, tant pour la programmation que pour la validation exige que lors de l'exécution, un contrôle soit réalisé pour vérifier le déroulement correct du programme sur le site réel. Le développement de toutes ces étapes de la programmation hors ligne, après une présentation rapide de l'apprentissage, est aussi l'occasion d'aborder les moyens utilisables dans chacune d'elles, et les conditions de leur mise en oeuvre en robotique.

3.1 L'APPRENTISSAGE.

Le moyen le plus utilisé pour programmer les robots consiste à "guider" leur structure mécanique pour enregistrer la séquence des points qui constituera la trajectoire à réaliser. Chaque point est affecté d'attributs (vitesse, chemin entre deux points, contrôle) qui permettrons à l'unité de commande de générer les mouvements du robot. Cette méthode a l'avantage:

- de la simplicité pour l'opérateur ;
- de réaliser un contrôle implicite du mouvement programmé qui se trouve de fait dans le domaine atteignable, qui évite, au moment de l'apprentissage les obstacles.

Elle présente aussi certaines limites :

- lorsque la tâche à réaliser demande une certaine précision, de nombreuses retouches sont nécéssaires ;
- lorsque le mouvement est constitué de nombreux points répétitifs, palétisation,insertion..., leur programmation est longue et pourrait être aventageusement remplacée par un calcul.
- L'apprentissage présente des inconvenients qui limitent l'utilisation des robots dans de nombreux domaines :
  * La détermination des performances d'un programme nécéssite son exécution. L'amélioration de celles-ci dépend de l'intuition du programmeur qui n'a aucun élément "objectif" pour l'aider.
  * la programmation sur le site de production empêche toute utilisation de l'outil productif ;
  * Chaque contructeur de robots propose sa méthode d'apprentissage, plus ou moins facile à utiliser, plus ou moins astucieuse ; la présence d'un parc de robots d'origines diverses pose des problèmes de formation du personnel.
  * les études sur la sécurité en robotique mettent en évidence le risque encouru par le programmeur lors des phases de programmation.

Malgrè ces inconvénients, ce mode de programmation restera sans doute prédominant dans les années qui viennent. Et il serait illusoire de prédire la programmation complête et totale de robots par d'autres moyens dans un avenir proche.

3.2 PROGRAMMATION.

L'apprentissage peut rester un moyen de programmation pour de nombreuses applications et les performances offertes par certaines armoires de robot (instructions de contrôle, prise en compte de capteurs...) renforcent d'ailleurs cette voie. Des raisons oeuvrent cependant pour la programmation hors ligne:

- Certaines tâches sont beaucoup plus simples à programmer lorsque les positions peuvent être calculées (palétisation) ou recalculées (recalage, poursuite) .
- La possibilité de calculs rend plus souple la prise en compte de capteurs.
- Les raisons économiques amènent à utiliser la programmation hors ligne pour augmenter le temps d'engagement des moyens de production (flexibilité de reprogrammation). Elles plaident aussi pour une meilleur utilisation des moyens de calcul : l'unité de programmation du robot n'est pas utilisée pendant la production .

Programmation en ligne et hors ligne.

- Les performances de l'outil de programmation seront de plus en plus fonction de sa possibilité de communication qui amène :
  . l'indépendance souhaitée bien qu'encore relative des programmes par rapport au robot;
  . la possibilité d'utiliser les moyens de la CAO pour la programmation et de communiquer les résultats de la fabrication aux systèmes de gestion de production;
  . l'intégration dès la programmation des divers composants d'un système robotisé.

L'intérêt réel de la programmation hors ligne a amené le développement de divers moyens de programmation. Certains sont opérationnels d'autres encore au stade expérimental.

3.2.1 Les langages.

Les premiers langages de programmation de robots ont été développés dans les années 197O par des laboratoires soucieux de disposer d'outil logiciel pour la mise en oeuvre de manipulations génériques : coordination vision-manipulateur (WAVE)[PAU76], capteur d'effort (AL)[FIN75]. Leur développement semble cependant plus rapide que leur utilisation et Barrus Iroin Soraks [SOR83] propose quelques raisons à cet état de fait qui rejoignent les caractéristiques, développées par L. Henninger [HEN85], nécéssaires à un langage en robotique.

3.2.1.1 Caractéristiques des langages.

1. La facilité d'utilisation.

   Cette notion dépend évidemment des compétences que l'on attend du progammeur mais il faut mettre en évidence d'autres éléments plus objectifs :

   - temps d'apprentissage du langage;
   - le type de modalité (langage textuel, possibilité de menu...);
   - les instructions de branchement (si, tant que...);
   - les utilitaires disponibles (éditeur de textes, macro, gestion de fichiers, fichiers de commande, fonction de débogage, simulateur);
   - les outils de mise au point (pas à pas, point d'arrêt, trace ...);
   - la manière dont sont définis les positions dans l'espace (repère, point, angles...);

2. Le niveau de modélisation.

   Les langages robotiques doivent manipuler des entités variées dans un environnement donné.

   Les objets peuvent être décrits dans le langage :

   - soit comme des repères auxquels ils sont associés,
   - soit comme des volumes : dans ce cas, plusieurs représentations sont possibles (limites, forme analytique, octree...) chacun ayant des avantages et des inconvénients par rapport au traitement souhaité (visualisation, détection de collision, calcul d'inertie...)

   Outre la représentation liée aux objets, les modèles des ensembles articulés peuvent prendre en compte :

- la cinématique de l'ensemble,
- la dynamique,
- les modèles des actionneurs.

3. Le mode de déplacement.

   Le langage possède-t-il le moyen de générer des trajectoires de chemin linéaire, circulaire (ou de fonction quelconque : spline, polygonale) et d'en contrôler la loi de vitesse? Les modes de déplacements possibles vont être étroitement liés au contrôleur de mouvement du robot. Il faudra cependant conserver une certaine portabilité des programmes.

4. L'interaction avec l'environnement.

   Deux aspects pourront être développés sous la notion d'interaction :

   * la capacité de dialogue par divers canaux avec l'environnement (liaison avec d'autres calculateurs, gestion de signaux logiques, ...) ;
   * l'aptitude offerte par le langage d'intégrer des capteurs spécialisés (vision, force, proximètre...).

5. L'aptitude au parallélisme.

   Il doit être possible, par exemple grâce aux fonctions du langage et du système d'exploitation, de gérer plusieurs manipulateurs évoluant ensemble et d'activer en même temps un processus de détection d'objet. Cet aspect est fondamental pour les développements d'application en robotique.

3.2.1.2 Différentes approches.

Une étude menée par A. Gruver [GRU83] sur des langages pour la robotique met en évidence diverses conceptions de ces langages.

1. Nouveau langage ou bibliothèque de sous-programmes.

   L'approche la plus courante a été le développement de langages "robotique" parfois à partir de langages existants (ALGOL pour AL, BASIC pour VAL [SHI79]). Une approche semblable a été realisée en France pour ROL (dérive du BASIC), LM, LMAC.

   D'un autre côte, nous trouvons les bibliothèques de sous-programmes appelables par un programme développé dans un langage standart (JARS,RPL). Une telle approche est aussi celle du laboratoire du LIMSI avec RCCL (Robotic Control C Librairy) pour le langage C [HAY86].

2. Compilé ou interprété.

   Le débat entre ces deux termes est souvent celui qui oppose rapidité d'exécution et souplesse de mise au point. Des solutions intermédiaires entre tout compilé et tout interprété peuvent être envisagées (précompilation en un code lui-même interprété par le contrôleur, compilation de module critique prend un temps d'éxécution et langages interprétés permettant leur appel (voir SIMIR [HEN 85]).

3. Temps réel

La notion de temps réel peut recouvrir des réalités très différentes suivant le contexte dans lequel elle est employée. S'il faut définir ce terme, on peut dire qu'un système répond en temps réel à une application si le temps qui s'écoule entre une solicitation externe et le résultat de l'action liée à cette solicitation est compatible avec la rapidité d'évolution de l'ensemble contrôlé.

Pour un système rapide (asservissement, contrôle de processus ...), il doit être possible de gérer prioritairement les actions initialisées sur un signal exterieur. Dans le cas de processus à évolution plus lente, il est possible ne pas assurer l'exécution immédiate de telles actions, et le système devra simplement fournir les résultats des actions avant qu'ils ne soient utiles pour le contrôle suivant.

En robotique, les deux cas se présentent ils sont souvent traités par des processeurs différents. Dans les armoires de robots, les fonctions asservissements sont gérées en temps réel par des unités différentes de celle qui assure le contrôle global des fonctions du robot.

3.2.1.3 Conclusion.

L'une des limites de l'emploi de langage en robotique est leur aspect procédural. Celui rend délicate la programmation d'une application dans laquelle les séquences d'évenements peuvent varier de façons multiples. Une approche déclarative peut dans ce cas présenter une plus grande souplesse.

3.2.2 Les sytèmes de décision.

Le but de ce chapitre n'est pas de développer longuement les diverses techniques de l'intelligence artificielle, mais d'en situer les méthodes et de montrer comment elles peuvent constituer des moyens de programmation pour la robotique.

3.2.2.1 Les techniques de l'Intelligence Artificielle."[NIL80] [SHI87]

Le terme Intelligence Artificielle commence à être utilisé après la parution en 1961 de l'article de M. Minsky du MIT "Steps toward artificial intelligence". Depuis, les recherches dans ce domaine ont eu pour but de remplacer l'homme dans la résolution de problèmes pour lesquels aucune séquence précise d'opération n'a été déterminée.

1. L'approche générative

   Dans cette approche, une situation donnée est décrite par un état. Un état est un ensemble d'informations qui définissent la configuration des entités pouvant être traitées. La configuration peut être une position, une relation, une condition logique...

   Des opérateurs permettent de passer d'un état à un autre état si des préconditions sont satisfaites. Ils sont donc définis par les préconditions nécessaires à leur activation, les transformations de l'état réalisées. Le logiciel STRIP associe par exemple trois listes de prédicats à chaque action. La définition d'un état est ici une liste d'opérateurs logiques pouvant être vrais ou faux (prédicat).

   Etat ═══ prédicat :

Action ═══ prédicat : PList prédicats vrais pour l'action se réalise
prédicat : AList prédicats vrais après l'action
prédicat : DList prédicats faux après l'action

La PList exprime les préconditions, les AList et DList la définition du nouvel état.

L'approche génératrive va se définir par la recherche d'une solution qui permette de passer d'un état initial à un état final ou état cible. La solution est une séquence d'états, et donc d'activations d'opérateur.

L'espace d'état est l'ensemble de tous les état possibles en relation avec les opérateurs. La solution est donc un sous-ensemble de l'espace d'état. L'espace d'état peut être représenté par un graphe dont les noeuds sont les états et les arcs les opérateurs. La procédure de recherche d'une solution est une exploration de ce graphe. Le but des différents algorithmes existants est de limiter l'étendue de la recherche par différentes méthodes, car le parcours systématique comporte un aspect combinatoire important dans le cas de problèmes réels.

Si l' on affecte à l'opérateur une fonction coût, on peut :

. rechercher la solution optimum en choisissant pour un ensemble d'états donnés, le résultat des parcours précédant l'opérateur dont l'exécution minimise le coût global du parcours.

. estimer le coût final du parcours pour arriver au but et développer un heuristique H qui est l'estimation du coût du chemin restant à parcourir dans le graphe. On choisit de la même façon d'appliquer l'opérateur qui minimise H (algorithmes A, A*, A* modifié).

Une deuxième solution est d'imposer des contraintes liées à l'opérateur et de réaliser le marquage de l'état des noeuds correspondant aux contraintes. Cet aspect propagation de contraintes convient particulièrement aux problèmes de reconnaissance.

2. Les systèmes de production

Un système de production est constitué des éléments suivants :

1. un ensemble de règles de production
2. une mémoire de travail
3. des méthodes de contrôle pour déterminer l'ordre d'application d'une règle.

Une règle est constituée de préconditions et d'une action. L'action n'est exécutée que si les préconditions sont vraies.

La mémoire de travail contient l'état du système. Elle peut être modifiée par le système de production en application des règles, ou par le monde extérieur.

Un cycle de reconnaissance va :

1. comparer le contenu de la mémoire de travail avec les préconditions des règles.
2. appliquer les règles pour lesquelles une correspondance a été établie.

Lorsque plus d'une règle sont applicables après l'étape 1, l'élément 3 du système de production doit intervenir.

A. Meller [MEL85] propose de structurer les ensembles de règles et de mémoires de travail associées par domaines afin de mieux adapter le fonctionnement global du système de production aux modules d'un système robotique. Nous présentons ici la structure de APSIS.

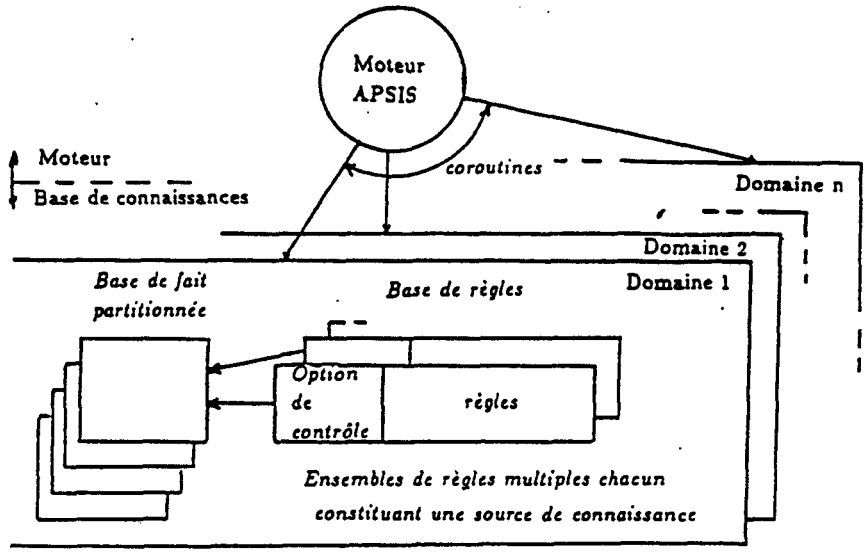


Figure 3. 1. Structure du système APSIS.

3.2.2.2 Utilisation en robotique.

Dans les deux cas, la programmation se fera de manière déclarative :

- en décrivant l'état initial du système
- en décrivant l'état final recherché
- en indiquant les critères ou les règles

La recherche d'une solution conviendra assez bien à un univers figé, et dans lequel l'exécution des actions ne modifie pas l'environnement. La solution trouvée est en effet entièrement générée à partir d'un modèle de comportement de l'univers sous l'effet d'une action, c'est-à-dire entre deux états. Certains événements aléatoires, les différences de comportement entre le modèle et la réalité peuvent rendre caduque en cours d'exécution la solution générée, d'où nécessité d'une nouvelle génération. Le "Find path problem", la recherche de trajectoire libre de collision, correspond cependant à cette catégorie [LOZ83] ainsi que les problèmes de reconnaissance en vision.

Les systèmes de production semblent plus adaptés au monde mouvant et incertain de la robotique. Car l'évaluation se fait à chaque cycle du moteur d'inférence et autorise donc la prise en compte par la modification de la mémoire de travail des variations de l'environnement (prévues et dues à l'action ou imprévues).

3.2.2.3 Limites et intérêts pour la programmation.

I. Limites

La description des informations en robotique est souvent complexe (aspect géométrique, temporel, relationnel, effort...) et parfois mal modélisée.

La définition des règles ou des critères n'assure pas leur validité et donc le succès de l'exécution dans toutes les configurations. Ce problème reste vrai à un autre niveau pour les moyens classiques de programmation.

Les procédures d'action doivent être écrites si elles sont spécifiques de l'application. Ceci limite l'intérêt de l'aspect déclaratif, et demande la maîtrise du système de production et d'un langage de programmation.

Une piste ouverte reste la création d'interfaces logicielles pour aider le programmeur à décrire l'état des systèmes et à définir les règles. La possibilité d'auto-apprentissage des règles à partir de l'expérience de fonctionnement peut à la fois apporter une puissance réelle au système et risquer de le conduire à l'impossibilité d'agir si l'on ne prend pas garde à la génération de règles contradictoires.

II. Intérêts

- La programmation se fait au niveau objectif, elle n'a pas à tenir compte des éléments de la cellule (robot, manipulateur outil...) et peut être exécutée sur des sites différents.
- La démarche procédurale de la programmation classique doit prendre en compte les étapes d'évaluation, d'action, de mise à jour de l'environnement. L'aspect séquentiel rend cette démarche délicate car il faut prévoir quand évaluer et ce qu'il faut évaluer. Le cycle de reconnaissance autorise une évaluation systématique, qui permet de prendre en compte des événements d'occurence aléatoire.

  La démarche déclarative permet de bien distinguer les informations sur l'état, les procédures d'action à mettre en oeuvre et les relations qui les unissent.
- Les techniques de l'intelligence artificielle peuvent s'appliquer à des domaines différents. La programmation des modules du système robotique peut se faire à partir du même environnement de programmation. Le programmeur n'est alors confronté qu'à un seul environnement de programmation.

3.2.3 Les systèmes de simulation graphique.

Ces systèmes sont le plus souvent utilisés comme moyens de validation de la programmation (voir III.3). Ils peuvent cependant être un moyen de programmation à part entière [DIL86] [CAS86]... Ces logiciels intègrent obligatoirement :

- une modélisation géometrique des objets;
- un logiciel de visualisation 3D autorisant des perspectives variées;
- un logiciel d'animation permettant la modification des positions relatives des objets entre eux;

- un simulateur de programmes robot. Certains langages manipulent même la notion d'émulateur de composant (robot, capteur, périphérique...) qui permet une simulation fine de l'exécution du programme[DIL86].

Pour devenir un outil de programmation, ils comprennent :

- soit un interpréteur de langage: dans ce cas, la validation est immédiate ;
- soit une interface logicielle qui autorise le programmeur à piloter les diverses composantes de l'ensemble.

La philosophie, qui tend à montrer immédiatement le résultat prévisible d'une action (aspect simulation) donne aux systèmes de simulation graphique les avantages de l'apprentissage en lui procurant les performances de la programmation hors-ligne.

3.2.4 Conclusion.

La programmation d'un site robotisé doit d'une part, prendre en compte de plus en plus d'éléments (perception, communication...) et d'autre part, être de plus en plus souple. Cette recherche d'une exhaustivité de la programmation et d'une amélioration de l'interface homme-machine n'est pas propre à la robotique et l'on peut même dire que celle-ci bénéficie des apports des recherches en informatique.

Cependant, des outils spécifiques doivent être mis en oeuvre pour prendre en compte l'univers tridimensionnel, sa géométrie, sa dynamique, dans la programmation. Il n'existe donc pas a priori un moyen de programmer efficacement un système robotique mais une palette de moyens dans laquelle il faut choisir pour réaliser le système de programmation le plus adapté à l'utilisation qui doit en être faite.

3.3 VALIDATION.

La validation de la programmation constitue une étape de contrôle avant l'éxécution du programme. La validation peut se faire soit par simulation, soit sur site.

Les contrôles peuvent porter sur la faisabilité de la tâche ou sur une estimation des performances atteintes par le programme [LIE84].

Ces opérations de contrôle offrent alors la possibilité de retouches (sur site) ou de modification de programme (sur site ou à partir de la simulation).

La validation va utiliser des modèles plus ou moins fins des éléments intervenant dans l'exécution du programme.

3.3.1 Volume de travail."[BOR86]

On vérifie que l'ensemble des positions programmées sont accessibles aux éléments articulés (robots...). Cette étape met en oeuvre les modèles géométriques inverses de toutes les

structures articulées en tenant compte des limites d'évolution des coordonnées articulaires et les positions singulières de la structure.

3.3.2 Aspects temporels de l'éxécution.

1. Trajectoires - Temps d'éxécution

   La programmation ne fait pas seulement intervenir des données géométriques, mais elle peut imposer des lois de vitesse sur le chemin de sa trajectoire.

   Deux cas vont se présenter, en vue d'une validation :

   1. la configuration du système robotique autorise le contrôle complet de la trajectoire : on a accès aux consignes d'asservissement sur chaque axe du robot.
   2. la configuration du système robotique impose pour le robot les trajectoires entre les points programmés dans l'armoire de commande.

   Le deuxième cas ne permet que la simulation des trajectoires. Il faut réaliser un simulateur de robot assez fin pour le permettre. Celui-ci doit intégrer tous les modes de définition de la trajectoire disponibles dans l'armoire du robot :

   - chemin imposé, rectiligne, circulaire,... ou libre ;
   - vitesse variable ;
   - précision de point de passage ;
   -

   On pourra en déduire le chemin simulé de la trajectoire et le temps d'éxécution [BLA87].

   Le premier cas n'impose pas de trajectoire. On peut définir des points de passage et rechercher la trajectoire optimale vis-a-vis d'un critère donné (souvent le temps d'éxécution). Il est nécessaire ensuite de vérifier que les accélérations imposées par la loi vitesse sont compatibles avec les capacités des actionneurs (couple maxi) en rapport avec la dynamique du système [LIE84].

2. Actions coordonnées.

   Les temps d'éxécution des trajectoires permettent de vérifier si les déplacements du robot peuvent être réalisés durant l'exécution de tâches de durée d'éxécution donnée. La manipulation d'une pièce pendant un usinage est le cas typique d'actions coordonnées.

   Un autre aspect des actions coordonées se produit lors des déplacements de deux manipulateurs en vue par exemple de leur arrivée simultanée en un lieu donné. Dans ce cas, le mouvement le plus rapide potentiellement devra être ralenti pour s'adapter au temps d'évolution du déplacement le plus lent.

3. Utilité

   Ces aspects temporels vont permettre de déterminer, outre la faisabilité dans certains cas, les performances de l'ensemble d'une cellule. Les temps de cycle obtenus par simulation sont vérifiés sur site dans une marge de ± 10 % [QUE 82]. Ils constituent de bonnes données de base pour un système de planification de l'ensemble de la cellule.

3.3.3 Détection de collisions.

Les trajectoires en robotique font se déplacer dans l'espace des objets ayant un volume donné. Il est important de vérifier que ces volumes n'entrent pas en interférence entre eux ou avec d'autres volumes de l'environnement. En effet, une interférence géométrique en simulation se traduit dans le système réel par une collision avec les dommages matériels qu'elle entraîne.

I. Recherche d'interférence

S'il n'y a qu'un seul objet en mouvement, la recherche des interférences peut se faire :

- par comparaison de points particuliers de l'objet (sommets, extrêmité de l'effecteur...) avec les volumes des objets de l'environnement en des points particuliers de la trajectoire.
- par comparaison du volume de l'objet avec les volumes des objets de l'environnement en des points particuliers de la trajectoire. Cette comparaison peut se faire de volume à volume en chaque point ou en réalisant "l'expansion"des objets de l'environnement par le volume de l'objet à déplacer et en appliquant la démarche au cas précédants sur les objets "expansés" [LOZ83].
- par comparaison du volume engendré par l'objet au cours de son déplacement avec les objets de l'environnement.
- par comparaison de l'enveloppe de l'objet et des éléments mobiles du manipulateur engendrés par le mouvement avec les objets de l'environnement.

Dans le cas ou plusieurs objets sont en mouvement, il n'est plus possible d'utiliser les enveloppes engendrées. Un séquencement temporel des déplacements doit être réalisé grâce aux simulations de trajectoire. Les volumes en mouvement sont comparés dans leurs positions à chaque période de la simulation de la même façon que pour les cas 1 et 2.

II. Moyen de détection de collision

La détection de collision est un problème de décision. Il faut en effet décider si deux objets interfèrent ou non. Cette décision peut se faire de manière automatique après calcul ou être prise par l'opérateur humain sur la base d'une simulation graphique.

1. Détection automatique

Les traitements à éffectuer vont dépendre des modèles disponibles des objets. On peut en distinguer trois : modèle analytique, modèle polyhédrique, les octrees. Dans les trois cas, ce traitement est important parce qu'il nécessite soit des calculs nombreux (modèle analytique), soit une recherche qui est combinatoire (modèle polyhédrique,octrees)

2. La simulation graphique

Le modèle le plus souvent utilisé est la modélisation polyhédrique car elle se prête bien à la représentation sur un écran graphique. La scène est simulée en 3D et il revient à l'opérateur de déterminer au vu de l'image dont il dispose s'il y a collision ou non.

Comme l'on a à faire à une projection en deux dimensions d'un modèle en trois dimensions, le critère à appliquer est "pour décider que deux objets sont distincts, il faut et il suffit que leur projection soit distincte". On se rend compte que les projections doivent pouvoir être choisies par l'opérateur pour déterminer "le point de

vue" assurant l'absence de collision entre deux objets.

Un logiciel de faces cachées améliore aussi les performances de la simulation graphique en rendant l'image plus lisible et en indiquant certaines collisions par l'intersection entre les objets.

3.3.4 Conclusion.

L'étape de validation présente plusieurs intérêts du point de vue de la programmation.

- comme tout système de mise au point de programme, elle permet de tester les algorithmes utilisés sans prendre le risque de leur exécution sur site réel.
- d'un point de vue géométrique, elle oblige à définir l'environnement du programme et les éléments avec lesquels il va être exécuté, et teste la faisabilité du programme avec ses éléments.
- en fournissant au programmeur des informations sur les performances, elle permet une optimisation "objective" des programmes au regard de celles-ci.

Pour autoriser toutes les fonctionalités différentes, divers modèles des robots devront être mis en place (cinématique, dynamique...), la cohabitation de ces modèles peut rendre l'ensemble lourd à gérer et des choix peuvent être faits en fonction des performances que l'on estime importantes.

3.4 CONTROLE DE L'EXECUTION.

Quelque soit le système de programmation utilisé et les validations réalisées, l'éxécution de la séquence de déplacements générés ne peut être réalisée sans contrôle. Il s'agit de vérifier que la tâche s'effectue correctement, malgré les modifications de l'environnement de la tâche et l'imprécision des modèles utilisés, et de modifier la commande lorsque l'on s'écarte du but recherché. Cet aspect "asservissement" est déjà présent dans les commandes actuelles de robots en ce qui concerne la position de l'effecteur dans l'espace robot. De nombreuses tâches exigent cependant la prise en compte d'autres paramètres, leur mouvement sera dit "contraint". Dans un programme, des mouvements contraints et non contraints pourrons cohabiter.

3.4.1 Mouvement non contraints.

Un mouvement est dit non contraint lorsque la définition de sa trajectoire permet seule d'assurer l'accomplissement de la tâche. C'est le cas pour les applications de manipulation simple, de peinture, ... Certains contrôles sont cependant nécessaires, soit pour des raisons de sécurité, contrôle de la présence d'obstacles dans l'espace de travail du robot, ou de synchronisation avec les processus automatisés amont ou aval: présence de pièce, ouverture de la porte d'un tour, état de fonctionnement d'une machine.

A ces informations de type logique correspondent des instructions de contrôle de l'interface d'apprentissage ou du langage. Quelle que soit la richesse de ces instructions, elles ne modifient que peu la trajectoire. Leur action va de l'arrêt d'urgence à la selection d'une trajectoire prédéfinie, par exemple pièce bonne ou mauvaise, en passant par l'arrêt de la production avec émission d'un signal à l'opérateur pour signaler l'absence d'une pièce...

3.4.2 Mouvements contraints.

Lorsque la trajectoire programmée amène l'extrêmité du robot à proximité ou en contact d'un objet, il y a mouvement contraint. Dans le premier cas, le mouvement est soumis à des tolérances. Celles-ci sont plus contraignantes que les erreurs cumulées dues aux modèles, aux systèmes de génération de trajectoire, et aux asservissements, auxquels s'ajoutent les éventuelles variations géométriques de l'objet. Dans le cas du contact, une faible variation de position implique une variation importante des efforts de contact. Cette constatation impose une prise en compte de la notion d'effort. Le système doit être compliant.

1. Contrainte d'effort ou de contact : la compliance

   La compliance passive utilise les capacités de déformation, dues à son élasticité propre sous l'action d'un torseur d'efforts, d'un dispositif particulier spécialisé. Dans le domaine de l'assemblage, le plus souvent on trouve des poignets compliants, parfois des tables compliantes qui "corrigent" les erreurs de position en tendant, par leur déformation, à annuler les efforts non souhaités.

   Les applications d'usinage, de polissage robotisées ont amené le développement de support d'outils compliants. Les limites de tels systèmes sont leur spécialisation dès la conception en vue de l'éxécution d'une tâche donnée (annulation de la composante d'effort paralléle à l'axe d'insersion en assemblage par exemple). D'autre part, ils ne tolèrent que de faibles variations de positionnement relatif.

   La compliance active utilise des informations sur les efforts pour modifier la commande en position. A partir de la mesure des composantes du torseur d'effort, il s'agit de générer des mouvements de faible amplitude qui prennent en compte les impératifs du positionnement et le contrôle des efforts de contact. Il s'agit donc de systèmes asservis, et les critères de précision et de stabilité de ces systèmes pourront être appliqués aux systèmes compliants. Ces systèmes tolèrent en général des écarts plus grands que les systèmes passifs, mais leur amplitude reste limitée.

2. Contrainte de proximité

   Certaines tâches imposent un écart précis entre la pièce et l'outil (soudage, encollage). La notion de proximité implique qu'il n'y a pas contact entre l'extrêmité du robot et l'objet. La compliance passive ne peut donc être appliquée.

   Les méthodes des systèmes compliants actifs pourront être appliquées en prenant comme composantes mesurées les informations de distance objets/outil fournies par des proximètres. Les différentes approches seront détaillees dans le chapitre 4 sur la prise en compte des erreurs.

3. Adaptativité

   Certaines armoires de robot dispose d'un fonction dite d'adaptativité. La correction du mouvement en fonction des indications d'un ou plusieurs capteurs, est réalisée suivant une ou des directions fixes définies par apprentissage.

   Aucune correction en orientation n'est possible, les directions de corrections sont fixées dans le repère de référence. Ces deux points limitent fortement l'intérêt de l'adaptativité dans le cas de procédés nécéssitant le contrôle de l'orientation de l'effecteur.

3.5 CONCLUSION.

Nous retiendrons de ce chapitre l'intérêt de la programmation hors ligne, pas seulement pour la souplesse qu'elle apporte, mais aussi pour les possibilités de simulation, de contrôle, d'optimisation qu'elle offre.

Tout effort de programmation hors ligne n'empêche pas un contrôle de l'exécution du programme, car la réalité n'est jamais entièrement conforme aux modèles dont on dispose. L'intelligence artificielle à l'aide des systèmes de production, a l'ambition de traiter les deux étapes de la mise en oeuvre d'un programme. La question que l'on peut se poser est celle des moyens et de leur répartition. Faut-il un système de contrôle central qui réalise programmation, contrôle, simulation ? Ou faut-il plutôt concevoir des systèmes répartis avec chacun leur part de décision dédiée, contrôlée par un système central de planification? La structure informatique de l'ensemble sera différente :

* un "gros" ordinateur de process, avec un exécutif temps réel performant dans le premier cas.
* des moyens de traitements répartis avec des moyens de communication et d'échanges performants (permettant téléchargement de programmes, échanges de requêtes et de comptes rendus) dans la seconde hypothèse.

La place de l'homme dans le processus de programmation ne peut être négligée, car il semble encore illusoire en robotique de se passer de son expertise. Un poste de travail interactif doit permettre à l'opérateur de contrôler par la simulation les conséquences des ordres qu'il passe, et d'être en liaison avec le site réel pour les aspects contrôle, sécurité... Pour fournir une aide efficace à la formalisation des problèmes en trois dimensions, il doit être graphique, et intégrer au maximum des utilitaires issus de la connaissance des modèles nécessaires pour la programmation hors ligne.

Nous verrons dans le chapitre 5 comment le poste de programmation graphique répond partiellement à cette définition dans le cas de la soudure.

4. PROGRAMMATION MODELE ET PRECISION.

Lorsque la robotisation d'un procédé est envisagée, le cahier des charges de l'application précise les résultats à atteindre en fonction du procédé : l'aspect du polissage, l'assemblage à réaliser (tolérance, respect des états de surface), profondeur de la soudure... La qualité du produit final dépendent de ces critères. Pour le roboticien, ils vont se décomposer en deux aspects :

- l'aspect proprement robotique qui contrôle les trajectoires. Le procédé impose alors la précision du chemin, les lois de vitesse.
- l'aspect lié au procédé qui impose d'autres consignes spécifiques au processus : vitesse de rotation de l'outil, intensité de l'arc et vitesse de défilement du fil de soudage, débit de colle...

Les liens entre ces deux aspects sont très importants car des couplages physiques existent entre eux pour obtention du résultat souhaité. La démarche la plus courante est le développement de logiciels de contrôle spécifiques au procédé qui modèlisent ces relations. Pour qu'ils puissent se dérouler correctement, il faut que les aspects robotiques soient déja bien maîtrisés.

La maitrise de la précision que nous allons aborder dans ce chapitre, impose que l'on définise d'abord le modèle géométrique de la démarche de programmation. Cette modélisation permet la mise en évidence des origines des erreurs lors de l'éxecution d'une tache.

L'analyse des erreurs, de leur nature, de leurs origines amène à définir les performances en précision d'un système robotisé. Pour améliorer ces performances plusieurs démarches complémentaires peuvent prendre en compte les erreurs.

Suite à la présentation de ces méthodes, nous nous attacherons à l'amélioration de la précision par la détermination de la position de la pièce. Divers moyens existent: une méthode de palpage de la pièce par le robot est proposée dans la dernière partie de ce chapitre puis évaluée par un calcul d'erreurs et des essais sur site réel.

4.1 ASPECTS GEOMETRIQUES DE LA PROGRAMMATION.

4.1.1 Définition des notations utilisées.

- R : définit un repère:
- $\mathbf{v}$: définit un vecteur;
- $\mathbf{T}$: définit une matrice;
- a : définit un scalaire.

Les indices seront explicités au fur et à mesure de leur utilisation.

Dans la suite de ce chapitre nous utiliserons la notion de transformation: nous allons la définir ici. Une transformation permêt de passer d'un repère à un autre repère. Si $\mathbf{v}_o$ est un vecteur défini dans le repère d'origine R_o et $\mathbf{v}_f$ le même vecteur défini dans le repère final R_f. On peut écrire :

$$\mathbf{v}_o \overset{T}{\rightarrow} \mathbf{v}_f$$

Les modes de calculs pour passer de $\mathbf{v}_o$ à $\mathbf{v}_f$ se ramènent tous à la combinaison d'une rotation et d'une translation. La représentation de T par une matrice homogène [FOL84][LEE83] a l'aventage de de réaliser la combinaison des transformation par un produit de matrices et de pouvoir ainsi utiliser les formalismes liés à celles-ci.

Une matrice homogène s'écrit: $\begin{bmatrix} \mathbf{R} & \mathbf{d} \\ 0 & 1 \end{bmatrix}$.

Où $\mathbf{R}$ est la matrice 3×3 de rotation définie par les cosinus directeurs du repère R_f dans le repère R_o et $\mathbf{d}$ exprime les coordonnées de l'origine du repère R_f dans R_o. .

Les vecteurs sont de dimension 4. On passe de $\mathbf{v}$ à $\mathbf{v}'$ par $\mathbf{v} \rightarrow \mathbf{v}' = \begin{bmatrix} \mathbf{d} \\ 1 \end{bmatrix}$.

Si nous appelons $\mathbf{T}$ la matrice homogène associée à la transformation T, nous obtenons:

$$\mathbf{v}_f = \mathbf{T} \cdot \mathbf{v}_o$$

On peut aussi écrire: $\mathbf{v}_o = \mathbf{T}^{-1} \cdot \mathbf{v}_f$ avec

$$\mathbf{T}^{-1} = \begin{bmatrix} \mathbf{R}^T & \mathbf{R}^T(-\mathbf{d}) \\ 0 & 1 \end{bmatrix}.$$

Dans cette expréssion $\mathbf{R}^T$ est la matrice transposée de $\mathbf{R}$.

Trois notions reviennent régulièrement en robotique lorsqu'on aborde la programmation: la position, l'orientation, la situation d'un objet.

- La position: définit la composante translation de la transformation qui lie le repère d'un objet situé dans l'espace au repère de base.
- L'orientation: est déterminée par la composante en rotation de la même transformation.
- La situation: exprime à la fois la position et l'orientation de l'objet.

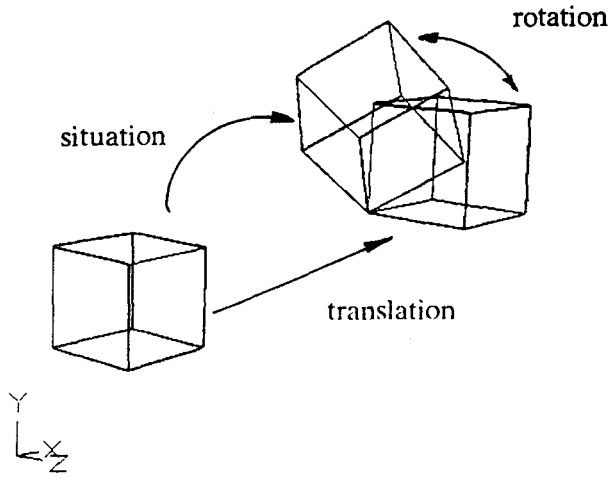


Figure 4.2. Situation d'un objet

4.1.2 Condition géométrique de réalisation d'une tâche.

Si nous considérons la programmation de la composante géométrique d'une tâche, nous pouvons établir que celle-ci revient à établir les conditions successives dans le temps de réalisation d'une (de) chaîne(s) cinématique(s) fermée(s). Dans le cas d'une seule chaîne, la figure 1 présente les éléments de la chaîne cinématique.

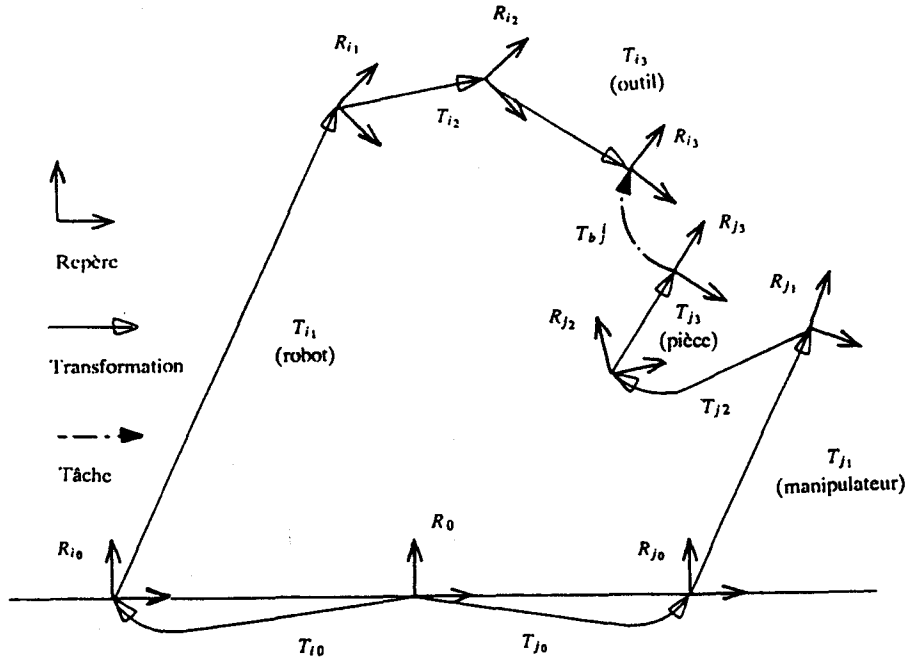


Figure 4.3. Description d'une chaine cinématique

Une chaîne cinématique fermée va donc être composée :

- de transformations préfixées qui représentent les positions relatives des différents éléments dans l'espace $\mathbf{T}_f$.
- d'une transformation de fermeture qui exprime les conditions de positions relatives des extrêmités des chaînes cinématiques ouvertes pour lesquelles la chaîne cinématique est fermée $\mathbf{T}_b$.
- de chaîne(s) cinématique(s) ouverte(s) (manipulateurs) qui définissent des transformations géométriques variables $\mathbf{T}_v$. Les transformations variables sont définies par la structure mécanique des chaines cinématiques. Les articulations (rotoïdes ou prismatiques) sont liées à des bras.

  Les coordonnées articulaires sont les amplitudes prises par les variables articulaires (angle ou distance) par rapport à une position de callage. *Les coordonnées cartésiennes* définissent la situation d'un objet situé à l'extrémité de la chaine cinématique. Le passage des coordonnées cartésiennes aux coordonnées articulaires est l'opération *d'inversion de la chaine cinématique* : elle est réalisée à partir d'une connaissance théorique de l'organisation des articulations (position relatives des axes, types d'articulation..) qui constitue *le modèle cinématique* de la chaine.

Soit T la composition de ces transformations sur une chaîne cinématique ouverte, nous

pouvons écrire l'équation de la chaîne cinématique fermée:

$$\mathbf{T}_i \ (EQ)\ \mathbf{T}_j \cdot \mathbf{T}_b{}_j{}^i \qquad (4.1)$$

Avec $\mathbf{T}_b{}_j{}^i$ la relation entre l'extrêmité de la chaîne j et celle de la chaîne i .

Dans la relation (4.1), (EQ) exprime le fait que $\mathbf{T}_b{}_j{}^i$ ne contraint pas toujours tous les degrés de liberté dans l'espace. Les contraintes peuvent fixer de 0 à trois rotation Rj, de 0 à trois translations Ti [TAK81].

(EQ) peut être de type égalité ($T3R3$) ou de type $TiRj$ ($T2R2$, $T1R3$...).

Nous allons maintenant étudier les différents modes de programmation en ligne et hors ligne pour déterminer par la suite quels sont les termes "sensibles" de l'équation cinématique qu'ils mettent en évidence. On pourra se référer au chapitre 2.2.2 pour la définition des termes.

4.1.3 Programmation en ligne.

Dans le cas de la programmation en ligne, la relation (4.1) est toujours une égalité dans la mesure où elle exprime une configuration physique du système.

Les transformations fixes sont définies, et les transformations variables sont modifiées par l'opération de programmation qui se déroule au niveau effecteur. La programmation en ligne définit donc la relation $\mathbf{T}_b$ qui représente la tâche à accomplir dans le repère objet par la relation suivante :

$$\mathbf{T}_b = \mathbf{T}_j{}^{-1} \cdot \mathbf{T}_i$$

On mémorise soit les coordonnées articulaires de $\mathbf{T}_i$ et $\mathbf{T}_j$ en supposant les $\mathbf{T}_f$ constants, soit les $\mathbf{T}_i$ calculés à partir du modèle du robot et les coordonnées articulaires de $\mathbf{T}_j$.

4.1.4 Programmation hors ligne.

1. Niveau effecteur

   On se trouve dans le même cas que pour la programmation en ligne. La différence principale provient du fait que l'on manipule les modèles des éléments de la chaîne cinématique, tant pour les transformations variables que pour les transformations fixes. Les modèles de transformation variable doivent inclure la notion de volume de travail des manipulateurs pour n'autoriser à la programmation que des valeurs des coordonnées

articulaires qui seront effectivement accessibles aux manipulateurs réels(butées articulaires, orientation valide pour un robot 5 axes).

2. Niveau objet

La programmation niveau objet définit les relations entre les extrémités de chaînes cinématiques ouvertes (effecteur/objet). La relation peut définir :

- une transformation complète liant les deux repères, les trois paramètres de translation ($T3$) et les trois rotations ($R3$) sont alors fixés.
- une transformation incomplète $Ti \;\; Rj$ *avec* $i \leq 3$ *et* $j \leq 3$.

Si le nombre de degrés de liberté contrôlés par $\mathbf{T}_i$ et $\mathbf{T}_j$ est supérieur à l'ordre de la relation (EQ), il y aura un nombre infini de configurations possibles. S'ils sont égaux, le nombre de configurations sera fini. S'ils sont inférieurs, il n'y a pas de configuration possible à priori. Les configurations ainsi définies doivent ensuite vérifier les limites d'évolution sur les axes (butées articulaires).

Le premier cas offre la plus grande souplesse et permet :

. à l'utilisateur de choisir arbitrairement les valeurs de certaines variables articulaires ;

. au système d'optimiser la configuration sur la base de critères définis (minimisation des mouvements,minimisation de l'energie du mouvement..) ;

. d'ajouter des contraintes sur les chaînes cinématiques pour diminuer le nombre de degrés de liberté contrôlables. On peut ainsi bloquer un mouvement, ou imposer une contrainte sur le repère final d'une chaine (mouvement avec l'axe z vertical, arête d'une pièce horizontale...).

4.2 NATURE ET ORIGINE DES ERREURS.

La maîtrise de la précision de la programmation impose une analyse fine des erreurs qui peuvent se produire dans la chaîne cinématique. Ces erreurs se traduisent par une différénce (un écart) entre la transformation souhaitée

Pour affiner l'analyse de ces erreurs, nous allons d'abord en définir la nature pour passer ensuite en revue les composants de la chaîne cinématique.

4.2.1 Nature des erreurs.

Nous allons distinguer pour la nature des erreurs deux cas distincts :

- l'erreur d'exécution qui définit l'écart entre les situations relatives de deux objets, théorique d'une part, réalisée de l'autre ;
- les erreurs de mesures qui devront être prises en compte dès que l'on réalisera une mesure à l'aide d'un dispositif spécialisé.

En mode statistique, pour chacun de ces deux cas, l'erreur inclut deux parts :

- une part déterministe, l'erreur systématique, qui correspond à l'écart entre la moyenne des valeurs obtenues et la valeur théorique ;
- une part aléatoire, l'erreur aléatoire, qui exprime les fluctuations des valeurs obtenues autour de l'erreur systématique.

Dans le cas de l'éxécution, nous pouvons écrire ΔT, l'erreur sur une transformation T définissant la situation relative de deux repères.

$$\Delta T = T^r \cdot (T^t)^{-1} \qquad (4.2)$$

. où T^t est la transformation théorique

. et T^r la transformation réelle.

Par exemple, si T^t exprime un déplacement entre deux situations de l'extrêmité du robot, ΔT sera la différence entre le déplacement commandé et le déplacement réalisé. Ce dernier doit être mésuré, pour connaître l'erreur, par un dispositif externe (laser, comparateur...).

Dans le cas de la mesure, l'erreur systématique n'est déterminable que si l'on utilise deux moyens de mesures (le moyen le plus précis servant de référence à l'autre). Les résultats des mesures sont souvent utilisés pour calculer les valeurs de paramètres non directement mesurables de l'environnement. Il faut veiller à la propagation des erreurs de mesure dans les calculs.

4.2.2 Origine des erreurs.

Si nous reprenons à titre d'exemple la chaîne cinématique de la figure 1, nous allons pouvoir définir les transformations suivantes.

1. Les transformations fixes

   - T_{i_o}, T_{j_o} : définissent les positions des repères de référence des chaines cinématiques ouvertes dans le repère de base R_0. Elles sont constantes et fixées par l'implantation de

la cellule.

- T_{i_2}, T_{i_3} : respectivement position de l'outil à l'extrémité du robot et transformation propre de l'outil. Elles sont constantes dans la mesure où l'outil reste le même et où il n'y a pas de démontage de celui-ci. En cas de démontage, remontage, la qualité du "centrage de l'outil" assurera l'invariance de T_{i2}.

- T_{j_2} : la position de la pièce sur le manipulateur peut varier car la pièce est démontée à chaque opération.

- T_{j_3} : dépend des dimensions de la pièce. Ses variations peuvent être dues aux variations dimensionnelles de celles-ci.

Dans le cas de l'apprentissage, les transformations fixes pourront donc être considérées constantes ou dans le cas contraire, les écarts seront de type aléatoire.

La programmation hors ligne manipule des modèles des objets, et donc des transformations fixes qui leur sont associées. Les écarts entre la transformation issue du modèle et moyenne de la transformation liée aux objets vont générer des erreurs systématiques.

2. Transformations variables

Elles sont liées aux configurations des robots et manipulateurs. Dans notre exemple, T_{i_1} et T_{j_1} sont imposées par les valeurs des coordonnées articulaires et les dimensions de différents "entre axes" de la structure.

L'étude systématique des performances des robots en positionnement est présentée dans [ACK85] et [COL85]. Ces deux études distinguent deux types de performances: la répétabilité et la précision.

La précision indique la différence entre la situation désirée d'un objet et la moyenne des situation réelles obtenues *lorsqu'il n'y a pas eu de mémorisation préalable de la tache.*

La précision représente donc une erreur systématique. La dispersion des situations réelles fournit une indication sur la composante aléatoire du positionnement.

La répétabilité est analogue à la précision lorsque la situation désirée a été mémorisée lors du phase d'apprentissage.

Les causes d'erreurs peuvent être:

- l'erreur de digitalisation:

  elle provient du fait que la commande en coordonnées articulaires est continue alors que le capteur mesurant la position articulaire est discrêt;

- l'erreur de calibration:

  elle est issue de la différence entre les valeurs des coordonnées articulaires réelles et celles qui sont prises en compte dans le modèle cinématique du robot pour la position de calage;

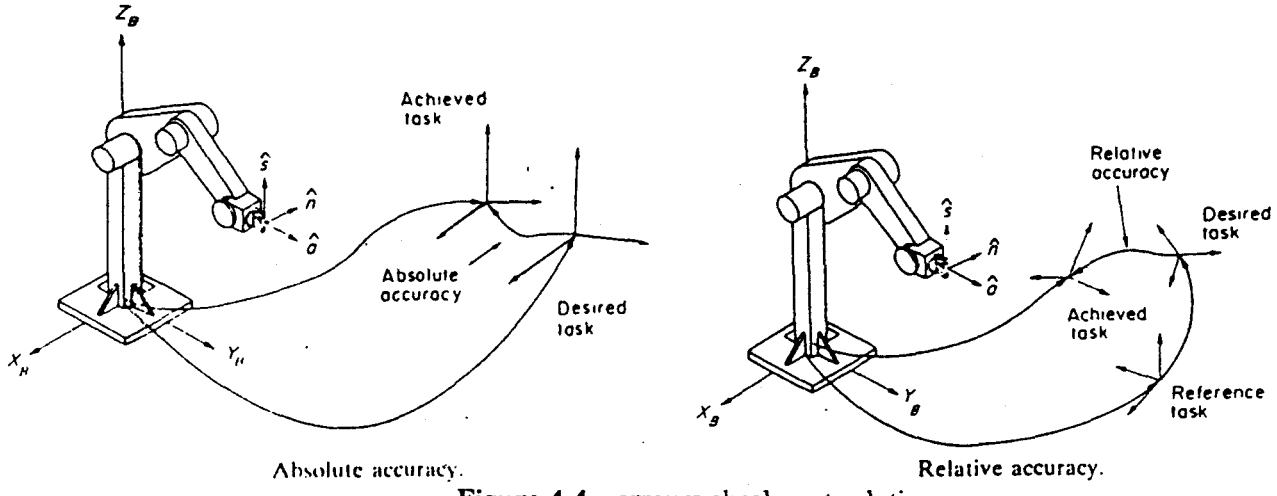
- l'erreur sur la structure cinématique:

  elle possède une composante systématique qui résulte de la modélisation imparfaite de chaque bras.

  La composante aléatoire est due pour chaque articulation, à l'existance d'un débatement qui ne provoque pas de correction de l'asservissement. Cette " zone morte " cumule les jeux, la résolution des capteurs, l'erreur résiduelle de l'asservissement.

  Dans le cas de l'apprentissage, les erreurs sur la structure vont être primordiales, principalement dans leurs composantes aléatoires. Les dilatations de la structure, la charge... peuvent provoquer une erreur systématique.

  En programmation hors ligne, l'ensemble des causes d'erreurs vont intervenir. Les principales sont les erreurs systématiques, dans lesquelles les erreurs de modélisations prennent une part importantes. L'erreur absolue est due à l'écart entre les coordonnées cartésiennes de consignes et les coordonnees réellement atteintes. Ces dernières sont difficiles à mesurer avec précision puisque le repère de référence est immatériel. On peut par contre mesurer l'erreur relative sur un déplacement de longueur donnée.



Absolute accuracy. Relative accuracy.

Figure 4.4. erreurs absolue et relative.

D.Scott Ackerson [ACK85] propose comme standart pour un robot électrique, dont le porteur a des bras de longueur 38 centimètres, les performances suivantes:

. précision absolue: 4 mm , 0.3 deg;

. précision relative: 1mm , 0.03 deg;

. répétabilité: 0.2mm

. influence de la charge: 0.05 mm/lb , 0.04 deb/lb.

Dans le cas de manipulateurs programmés en coordonnées articulaires, positionneur par exemple, il faut que le système de programmation hors ligne possède un modèle correct du manipulateur afin de limiter les erreurs systématiques sur la chaine cinématique et les erreurs dues au calage (voir [ISH86] présenté en 4.3.2.3).

4.3 PRISE EN COMPTE DES ERREURS.

Vues les multiples causes d'erreurs que nous venons de présenter, il serait illusoire de concidérer que le résultat d'une tache programmée hors ligne puisse être correct sans qu'un certain nombre d'efforts soient entrepris pour améliorer la précision ou prendre en compte les ereurs..

Deux approches vont donc se présenter: la première prend en compte les erreurs à l'exécution grace à des dispositifs spécialisés, la seconde vise à améliorer les modèles de tous les termes qui entre en jeu dans l'exécution de la tache.

Ces deux approches se complêtent. Nous alons même voir que la seconde est parfois nécessaire pour que la première puisse être utilsée.

4.3.1 La compliance.

Les erreurs cumulées dans la chaîne cinématique vont modifier la distance entre les deux objets, représentée par la composante translation de $\mathbf{T}_b$. Si celle-ci est plus grande que prévue, le procédé mis en oeuvre n'aura pas le résultat escompté.

Il peut se produire d'un point de vue mathématique que la distance entre les objets devienne négative. Ce cas, impossible physiquement, se traduit par un contact entre les deux objets. L'effort de contact sera d'autant plus important que l'amplitude de l'écart négatif est grande. La loi de variation de cet effort en fonction de l'erreur est fortement non linéaire et dépend de la "raideur" des asservissements des manipulateurs.

4.3.1.1 La compliance passive .

La compliance passive va ajouter dans la chaîne cinématique un terme dont la transformation est variable suivant les efforts auxquels il est soumis. De la relation (4.1) et de la relation (4.2) on peut déduire l'expresssion de ΔT_b :

$$\Delta T_b = (T_b{}^t)^{-1} \cdot \Delta(T_j)^{-1} \cdot T_b{}^t \cdot \Delta T_i$$

Pour que ΔT_b soit nul, c'est-a-dire que les conditions géométriques de l'éxécution de la tâche soient réalisées, il faut que le système compliant puisse compenser, par l'ajout d'un terme T_c, les erreurs cumulées sur la chaîne cinématique.

Si l'on raisonne sur les transformations complêtes, il est possible de déterminer T_c à partir de la mesure de ΔT_b. Le système compliant est alors placé entre l'extrémité du robot et l'outil de tranformation T_o.

$$T_c = T_o \cdot \Delta T_b{}^{-1} \cdot T_o{}^{-1}$$

En effet, si l'on appele T_r la tranformation avant la système compliant en abscence de compliance ($T_i = T_r \cdot T_o$). Comme T_b comporte une erreur:

$T_b = T_b{}^t \cdot \Delta T_b = (T_j)^{-1} \cdot T_i$ La compliance permet de passer de T_i à $T_i{}'$ de telle façon que ΔT_b s'annule.On a donc: $T_i{}' = T_r \cdot T_c \cdot T_o$ *et* $T_b = (T_j)^{-1} \cdot T_i{}'$

Ce qui nous donne:

$(T_j)^{-1} \cdot T_i{}' \cdot \Delta T_b = (T_j)^{-1} \cdot T_i$

$T_r \cdot \Delta T_c \cdot T_o \cdot \Delta T_b = T_r \cdot T_o$

$$T_c = T_o \cdot (\Delta T_b)^{-1} \cdot (T_o)^{-1}$$

Si l'on ne raisonne qu'en translation on trouve, le produit des transformations étant alors commutatif, $T_c = \Delta T_b{}^{-1}$. La difficulté de réalisation d'un système compliant passif réside dans la définition de la structure et des raideurs de celui-ci afin que les efforts de contact dus à ΔT_b produisent une tranformation ΔT_c.

La formulation dans le cas général montre l'intérêt du concept RCC (Remote Center Compliance) [WHI79] puisque dans celui-ci la correction s'effectue en un point extérieur au système compliant (l'extrémité de l'outil par exemple) ce qui équivaut à $T_o = identité$. La correction répond donc directement à la sollicitation en effort.

Les systèmes compliants passifs ayant une amplitude de correction limitée, il ne pourront compenser que des erreurs de faible amplitude. Une autre approche consiste à modifier l'un des termes variables de la chaîne cinématique pour annuler les erreurs. Les robots de type scara, conçus pour l'assemblage, constituent ainsi en eux-mêmes un système compliant. La configuration du robot se modifie sous l'effet des efforts d'insertion, grâce à la "raideur" programmable des asservissements sur les deux axes principaux.

4.3.1.2 La compliance active.

La compliance active part d'informations exprimant l'erreur de position relative (efforts, mesure de distance...) pour modifier la transformation d'un des termes variables de la chaîne cinématique. La mesure d'effort amène une déformation de faible amplitude du dispositif de mesure que nous négligerons par la suite.

1. Le robot

   Dans le cas de la compliance active à partir du robot, deux approches ont été proposées :

   1. une commande hybride effort et position
      [PAU76] [RAI81]

   2. la définition d'une matrice de raideur pour modéliser le comportement de la structure du robot en présence d'efforts

   De la même façon que pour la compliance passive on trouve: $\Delta \mathbf{T}_r = \mathbf{T}_o \cdot \Delta \mathbf{T}_b^{-1} \cdot \mathbf{T}_o^{-1}$

2. Le manipulateur

   Il peut s'agir d'une table instrumentée ou d'un manipulateur par mouvements fins [BID85] qui génèrent les mouvements nécéssaires pour annuler les efforts dus aux erreurs. Dans ce cas, $\Delta \mathbf{T}_m$ la variation de la transformation du manipulateur s'écrit : $\Delta \mathbf{T}_m = (\mathbf{T}_p)^{-1} \cdot \Delta \mathbf{T}_b \cdot \mathbf{T}_p$ où $\mathbf{T}_p$ est la relation pièce-positionneur.

La compliance ne peut agir correctement que si $\mathbf{T}_b$ se trouve dans un voisinage réduit autour de sa position théorique. Cela impose de limiter les erreurs dues aux termes de la chaine cinématique.

4.3.2 Identification des termes de la chaine cinématique.

La compliance permet, comme nous l'avons vu, de prendre en compte les erreurs de faible amplitude. Elle s'applique donc parfaitement aux erreurs aléatoires et dans une moindre mesure aux erreurs absolues. Ces dernières peuvent être importantes en programmation hors-ligne, et une démarche parallèle ou concurrente est l'amélioration de la précision des modèles.

1. Qualification de robot [RAN 84]

   - Le principe de la méthode consiste à mettre en correspondance un cube porté par le robot avec trois fois trois capteurs analogiques qui définissent un trièdre de référence. Les informations des trois capteurs par rapport à une face permettent de définir l'orientation et la position de la face par rapport au trièdre de référence. Le même calcul sur trois faces donne la position et l'orientation du cube.

   - L'avantage de cette méthode est sa simplicité: on la retrouve d'ailleurs dans de nombreux laboratoires.

   - Pour le problème qui nous intéresse, elle permet uniquement d'évaluer les erreurs de repositionnement (aléatoires) mais ne donne aucune information sur les erreurs absolues. Nous retiendrons cependant le principe de la détermination del'orientation et de la position d'une face.

2. Cartographie des erreurs [PAY 85]

   - Une mesure des positions atteintes par un point de l'extrémité du robot est réalisée grâce à un dispositif externe de précision meilleure que celle du robot. Les mesures sont réparties dans l'ensemble de l'espace de travail du robot. En comparant les résultats des mesures soit aux valeurs calculées à partir des consignes articulaires, soit aux valeurs commandées on peut dresser une cartographie des erreurs de positionnement. A une zone de l'espace on associera une valeur moyenne d'erreur, dont il pourra être tenu compte lors de la définition de la tâche.

   - Cette méthode a l'avantage de fournir des erreurs de positionnement absolu par rapport au repère de mesure.

   - Elle ne fournit cependant aucune information sur les origines des erreurs et le temps de recherche de la valeur correspondant à une zone de l'espace n'autorise son utilisation qu'en programmation hors ligne.

3. Système de calibration [ISH86] [ALD86]

   - Cette méthode a pour but d'affiner le modèle géométrique d'un manipulateur commandé en coordonnées articulaires.

     Pour définir complétement les erreurs sur une articulation, il a été montré, [ALD86], qu'il faut au plus 5 paramètres (4 fixes et 1 variable). Si l'on estime (N * L) variables articulaires (L est le nombre de positions réparties sur chaque axe) et x*N paramètres constants ($x \leq 4$) ainsi que la position du système de mesure dans le repère de

référence (4 paramètres), il faut réaliser m mesures qui vérifient les conditions d'identifiabilité $3*m > (x+L)*N + 4$. Les choix des mesures assurent l'indépendance de celles-ci.

Si l'on ne cherche pas à estimer que les calages des axes, [ISH86], la première condition se ramène à $3*m > L*N$.

Une méthode des moindres carrés permet de déterminer les valeurs des erreurs sur les paramètres recherchés en fonction des écarts entre les points mésures et les points calculés.

Les mesures sont effectuées par le même système que dans [PAY85] pour [ALD86] et par un manipulateur cartésien de haute réolution qui vient palper des point particuliers de l'extrémité du robot pour [ISH86].

- Les résultats obtenus montrent une nette amélioration en ne tenant compte que du calage des axes [ISH86] et la possibilité [ALD86] d'obtenir un modèle complet du robot qui, associé à un algorithme de compensation d'erreur, réduit d'un facteur 4 les erreurs de position du Robot MA 23.
- Nous pourrons la retenir pour l'identification du positionneur. Il faudra alors choisir un système pour réaliser les mesures.

4. Conclusions et applications

- Les méthodes de mesures des erreurs permettent de mieux connaitre les performances en précision des systèmes utilisés. Leur exploitation en vue d'une amélioration des performances semble difficile.
- Par contre, les méthodes d'identification proposées montrent la possibilité d'affiner le modèle du robot en vue de l'amélioration de la précision de sa commande hors ligne.

  Par la suite, nous ferons l'hypothèse, à vérifier sans doute par les méthodes sus citées, que les armoires de commande des robots que nous utilisons possèdent un modèle suffisemment précis de la géométrie du robot. Nous considérerons que l'erreur absolue due au robot est pratiquement constante dans le volume de travail de la tâche.
- Pour éviter des erreurs absolues importantes dues au positionneur, l'identification de la transformation variable $\mathbf{T}_{j_1}$ (le manipulateur) est nécessaire car son modèle géométrique n'est pas inclu dans l'armoire de commande et demande donc sa gestion par le système de programmation.

- Un autre élément de la chaine cinématique doit être identifié: la transformation $T_{i_2} \cdot T_{i_3}$ qui définit l'outil.

4.4 SITUATION DE LA PIECE DE TRAVAIL.

L'une des transformations fixe qui ont été définies en 4.5.1 n'a pas été traitée par les méthodes qui viennent d'être présentées. Il s'agit de la transformation que fait passer du repère terminal du positionneur au repère de la pièce(T_{j2}).

Cette position est redéfinie à chaque pièce et ne peut donc faire partie des "paramètre systèmes". La recherche de la situation de la pièce dans l'espace du robot permet d'obtenir à partir du modèle du positionneur la transformation recherchée. L'expression de ces tranformation dans l'espace robot permet d'utiliser le fait que la précision relative des robots est meilleure que leur précision absolue. En créant un référentiel local dans l'espace du robot les déplacements programmés en réference à celui-ci ne seront soumis qu'à l'erreur relative.

La recherche de la situation d'une pièce n'est pas l'objet principal des études menées pour intégrer des capteurs dans un ensemble robotisé. Ceci, sans doute pour plusieurs raisons:

- la reconnaissance d'une pièce pose beaucoup plus de problème théoriques que le calcul de la position. Celle-ci est souvent un résultat annexe des algorithmes de reconnaissance car elle est l'un des critère qui permêt de réaliser la correspondance entre les éléments d'une pièce issus de la mesure d'une part du modèle d'autre part. En effet en supposant la pièce indéformable, tous les éléments d'une même pièce sont issus de la même transformation (orientation, position) à partir du modèle [GRI83][FAU86].
- Les travaux ménés en reconnaisance de forme, en imagerie, sont souvent indépendants des besoins de la programmation hors-ligne des robots.
- Les couplages robot-capteurs viennent souvent en amont d'une opération de manipulation dans laquelle le préhenseur assure la précision du recalage de la pièce.

Dans notre cas, le problème de reconnaissance ne se pose pas et nous ne pouvons faire appel au procéde de soudage pour recaler la pièce qui par conception est fixe.

4.4.1 Localisation de la pièce à partir de caméras.[PER84]

Un système de vision fournit une image de la scène qui est fonction:

- des caractéristiques électroniques de la caméra et de son électronique associée (résolution,nombre de niveaux de gris,linéarité...);
- des caractéristiques de l'optique(distance focale,axe optique...);
- de la position de la caméra par rapport à la scène;

Les systèmes de vision, et leurs algorithmes associés (transformée de Fourier du contour,segmentation...), fournissent la position (souvent celle du centre de gravité) et l'orientation de la projection pièce dans le plan de l'image.

Pour ramener cette information dans l'espace de la scène, la détermination de la transformation entre le plan image de vecteurs $(x\ y\ 1)^T$ et le plan de la scène $(u\ v\ w)^T$ est indispensable. Elle dépend de la distance focale et de la position de la caméra.

L.Peralta [PER84] propose une méthode de calcul de cette transformation pour calibrer une caméra sans faire d'hypothèse sur la position de son axe optique par rapport à la scène. Il montre que la transformation peut s'exprimer de la manière suivante:

$$\begin{matrix} wu \\ wv = \\ w \end{matrix} \begin{bmatrix} a_1 & a_2 & a_3 \\ a_4 & a_5 & a_6 \\ a_7 & a_8 & 1 \end{bmatrix} \begin{matrix} x \\ y \\ 1 \end{matrix}$$

Le calcul des huit coéficients est possible par la calibration de 4 points dont les coordonnées sont connues dans le plan de la scène.

- Cette méthode présente l'intérêt d'une calibration directe à partir d'une seule image et sans faire d'hypothèse sur la position de la caméra. Elle fournit la position d'un objet dans le repère de la scène qui peut servir de repère de référence.
- L'étude de la chaine cinématique complête que nous avons pris pour base de l'étude montre qu'il faut alors déterminer la transformation repère de la scène - repère de base du robot pour pouvoir réaliser la programmation hors-ligne.
- Cette méthode est très bien adaptée à des pièces quasiment planes, situées sur un plan fixe. Ces deux caractéristiques la rendent inapplicable dans notre cas: la pièce est définie en volume, le positionneur peut avoir des attitudes variées.

4.4.2 Correction des positions programmées."[ELC84] [ELC86]

* La méthode proposée se base sur la mesure de la position de l'extrêmité du robot dans un repère de références. Cette mesure est réalisée par un système de mesures tridimensionnelles (SEL COM) constitué de deux caméras très haute résolution. La position d'une ou plusieurs LED'S infrarouges est obtenue par triangulation.

* La correction des positions programmées se fait sur les écarts entre les déplacements relatifs programmés et les déplacements relatifs mésurés. Le point de départ de l'exécution se fait en un point où l'on est "sûr" qu'il n'y a pas d'erreur (déterminé manuellement par exemple).

* Pour que la méthode puisse s'appliquer l'auteur fait les hypothèses suivantes:

  + Les repères objet et robot restent stationnaires;

  + Le robot est capable de fournir les coordonnées cartésiennes du point de son extrémité;

  + Le repère robot est identifiable;

  + Le robot possède une bonne résolution.

  Bien que les résultats n'aient pas été présentés, cette méthode présente un certain intérêt dans la mesure où elle permet la correction d'un programme de manière automatique.

Elle a pour limites :

. le coût des capteurs ;

. le fait qu'elle n'assure pas une position relative de l'outil par rapport à la pièce. La correspondance n'est assurée en effet qu'au point de départ de la correction. Si la position de la pièce ne correspond pas à sa position théorique les déplacements théoriques n'assureront pas la position relative outil/pièce.

Nous retiendrons cependant l'idée de correction automatique de programme qui peut être complémentaire du repositionnement dans le cas d'un robot peut précis dans un volume donné.

4.4.3 Positionnement à partir de données de palpage." [GRI83]

4.4.3.1 Principe.

Le palpage est un procédé d'évaluation de l'environnement qui se base sur un "contact" entre un élément tactile et l'objet. L'élément tactile détecte la position de contact et éventuellement les forces de contact avec l'objet.

Des travaux menés au MIT [GRI83] [FEA84] [GAS84] ont largement utilisé la notion de contact tactile pour la reconnaissance d'objet dont un modèle polyhédrique est connu. Les résultats de simulation montrent l'efficacité des critères proposés pour focaliser l'aspect combinatoire de la recherche. L'influence d'incertitudes de mesures est analysée par Grimson [GRI86].

La notion d'information tactile peut être étendue à des dispositifs de mesure de la topographie de la pièce. O.D.Faugeras propose par exemple un balayage laser associé à une triangulation [FAU86]. La quantité d'informations recueillie nécéssite alors un traitement pour extraire des élément singuliers de l'enveloppe (point, segment de droite, face plane) qui serviront de base au processus de repositionnement.

L'une des retombées des méthodes proposées est la possibilité de déterminer la situation de l'objet reconnu dans l'espace.

4.4.3.2 Détermination de la position de la pièce.

Pour déterminer la position de la pièce, nous disposons :

- d'un modèle théorique polyhédrique de la pièce.
- d'information sur la position d'élements géometriques de la piece issues du palpage: plans,segments de droite ou points.

On peut définir des couples d'éléments (M_i, S_i) liant un élément du modèle à une information issue de la scène.A partir de plusieurs couples d'éléments (3 plans, 4 droites, 6 points) il est possible de calculer la transformation de passage du modèle à la scène. Ce calcul peut être direct [GRI83] ou utiliser une méthode des moindres carrés [FAU86]. La transformation obtenue n'est valable que pour les éléments déjà appariés.

L'opération de " reconnaissance pendant la localisation" consiste à soumettre les éléments non appariés du modèle à la transformation et à rechercher les éléments de la scène qui correspondent. Chaque nouvel appariement permet le calcul d'une nouvelle valeur de la transformation.

4.4.3.3 Intérêts du palpage.

- Le palpage permet la détermination de la situation du volume de la pièce et pas seulement de sa projection dans un plan comme dans le cas de la vision. Cette méthode semble donc

bien adaptée au problème posé.

- Les moyens de palpage proposés dans la littérature sont choisis en fonction de la reconnaissance et non du positionnement. Cette première nécessite un grand nombre de données pour permettre des appariements significatifs. Les moyens de mesure sont donc assez complexes.

- L'opération d'appariement débute par une recherche sur un petit nombre d'éléments (souvent de faces planes). La première transformation trouvée va dépendre du choix de ces faces et la validité de l'ensemble de l'appariement découle de la justesse de ce résultat. Grimson propose des critères basés sur la géométrie de faces (taille normale, distancè.) pour définir les couples de candidats (M_i, S_i). Faugeras, quant à lui, propose plusieurs critères :

  * taille des faces suffisante

  * normale aux faces orthogonales

  Deux démarches partent donc de la connaissance du modèle pour déterminer les couples candidats du début de l'algorithme de reconnaissance.

- Dans le cas du calcul proposé par Grimson, les calculs sont simples et donnent un résultat immédiat sur la position de la pièce.

  On a donc un moyen de détermination de la position d'une pièce relativement peu sensible aux erreurs de mesure, mais qui nécessitent un grand nombre de résultats. Si la première partie de la proposition est intéressante, la seconde pose problème pour la démarche que nous voulons proposer.

4.5 METHODE DE DETERMINATION DE LA SITUATION D'UNE PIECE.

4.5.1 Objectif de la méthode.

La démarche de palpage correspond bien à la détermination de la situation de la pièce dans l'espace. Il semble donc intéressant d'exploiter ses résultats, tout en les adaptant à notre problème d'amélioration de la précision.

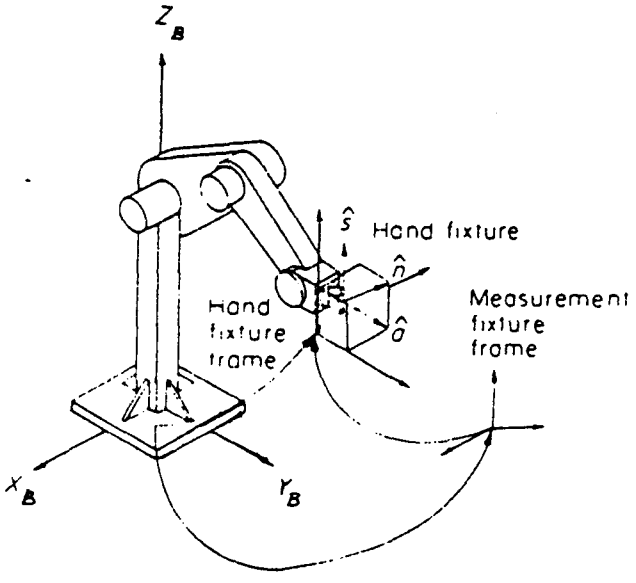
Ces adaptations vont être de deux ordres :

- se libérer du référentiel de mesure

- limiter le nombre de mesures nécessaires.

1. Référentiel de mesure

Les trois approches que nous venons de présenter fournissent leurs informations sur la situation de la pièce dans un système de coordonnées lié au dispositif de mesure (plan de l'image, position des caméras,...). L'information utile pour la programmation est la position dans le repère du robot. Il faut donc calibrer la position du référentiel de mesure dans l'espace du robot pour obtenir cette information. Si des méthodes automatisées ont été proposées dans le domaine de la vision par exemple, il semble plus judicieux de s'affranchir de cette étape. G.Gaston [GAS84] a proposé dans le cas d'un problème plan de réaliser le palpage par la main tridactile de préhension manipulée par un robot.



Hand measurement fixture frames.

Figure 4.5. Référenciel de mesure

2. Limitation du nombre de mesures nécessaires

L'utilisation pour un calcul suffisamment précis de la situation d'un objet, d'un grand nombre d'informations exclut l'emploi de capteurs de palpage discrets qui ne fournissent que peu d'informations.

L'objectif de la méthode peut donc se définir ainsi : méthode de palpage permettant

. de définir la situation d'un objet directement dans le repère de base du robot,

. la position relative robot/positionneur ;

. la relation entre l'extrémité de l'outil et le robot.

La dernière hypothèse suppose que l'on prenne quelques précautions quant au centrage de l'outil par rapport au poignet pour éviter les dispersions lors des montages démontages de l'outil. Un calibrage de l'outil peut être fait après chaque montage.

4.5.2.3 Choix de l'élément tactile.

L'élément tactile peut être varié, la main proposée par Peter C. Gaston [GAS84], micro-manipulateur avec retour d'effort [BID86], ou capteur discret , proximètres analogiques (potentiomètre linéaire, paramètre optique), détecteurs de contact (switch, capteur optique, tout ou rien). Plus l'information fournie par l'élément tactile sera riche, moins il y aura besoin de mesures et donc de déplacements, et plus la méthode sera rapide. Pour des raisons de généralité, nous partirons des capteurs les plus simples. La méthode reste valable si l'on a des informations plus riches.

Dans la démarche proposée dans la suite, l'élément tactile est un capteur de proximité. Le signal logique de sortie indique la présence d'une surface à une distance fixée du capteur suivant la direction de palpage.

Ce type de capteur peut être un switch électromécanique, un détecteur de proximité inductif ou optique. Nous avons choisi pour nos essais un détecteur de proximité optique car il n'impose pas de contrainte sur la nature de la pièce, et réalise une détection sans contact. La répétitivité de déclanchement est de ± 0,1 mm dans une gamme de 5 à 20 cm.

Le choix de ce type de capteur impose d'accorder beaucoup de soin à l'orientation relative du plan à palper et de la direction de palpage. A l'erreur de paralaxe vient en effet s'ajouter la sensitivité de la détection à l'angle entre le capteur et la normale à la surface.

. d'obtenir une précision correcte,

. d'utiliser des capteurs simples et de traitement immédiat.

4.5.2 Principe de la méthode.

4.5.2.1 Principe du palpage.

Pour répondre au premier objectif de la méthode, le robot sera utilisé comme moyen de mesure de son environnement. Les hypothèses que cela suppose sont détaillées dans le paragraphe suivant.

La connaissance du modèle de la pièce permêt, à partir de la définition approximative de la situation de celle ci, de piloter le robot pour amener le dispositif de palpage porté par celui-ci au contact de la pièce.

De plusieurs contacts il est possible d'extraire les informations nécéssaires au calcul de la situation de la pièce.

4.5.2.2 Hypothèses.

Pour déterminer la situation de la pièce dans l'espace robot, les hypothèses suivantes sont posées :

* on dispose d'un modèle théorique polyhédrique de la pièce ;

* la position théorique approchée de cette pièce dans l'espace robot est connu ;

* l'armoire du robot peut donner la position, dans le repère de référence du robot, d'un point de l'extrémité de celui-ci. La précision attendue sur cette information est du même ordre que la résolution, soit $\approx$ .06mm.

* l'erreur absolue de positionnement du robot sera concidérée constante dans le volume de travail englobant la pièce.

Les deux dernières hypothèses rejoignent celles rapportées en 4.4.2.

Pour déterminer la position pièce/positionneur, l'identification des éléments fixes peut être réalisée une fois pour toutes. Nous supposerons donc connu :

. le modèle du positionneur ;

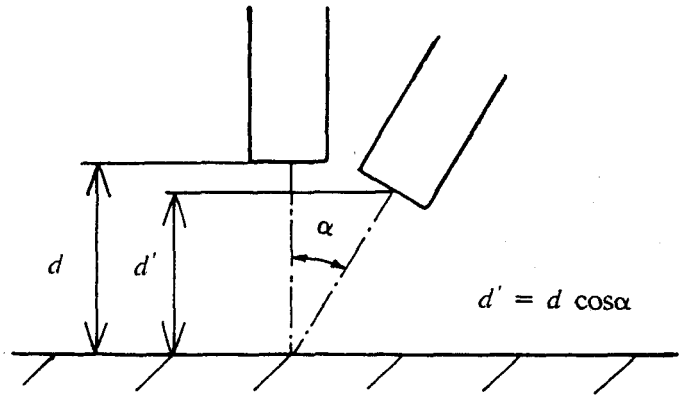


Figure 4.6. Erreur de paralaxe sur le palpage.

Ce capteur ne fournit pas d'informations sur la normale au plan palpé la démarche de palpage devra donc en tenir compte.

4.5.2.4 Déroulement du palpage.

I. Exploitation du modèle et de la position théorique de la pièce.

On choisit (les critères de choix sont explicités dans 4.1.5) 3 faces de la pièce à palper. Le repère de référence de la pièce est confondu avec le repère de base du robot. Pour chacune des faces, on détermine le centre de gravité et la normale.

Ces vecteurs permettent de définir la transformation $\mathbf{T}_{j3}$ qui dépend de la géométrie de la pièce et de la chaîne cinématique. Cette transformation est incomplète puisqu'elle ne fixe pas complètement l'orientation du repère. On peut imposer que l'axe **z** du repère R_{i3}, l'outil, soit confondu avec la normale. Les axes x et y sont fixés arbitrairement. Les coordonnées du centre de gravité fournissent les composantes en translation.

Le pilotage du robot en vue du palpage est rendu possible grâce à la connaissance de la position théorique de la pièce. La combinaison de deux transformations, la position de la pièce et $\mathbf{T}_{j3}$ (donnée par la situation du pont de palpage sur la pièce) fournit un $\mathbf{T}_j{}'$ qui permet de générer le mouvement de palpage $\mathbf{T}_i$.

Bien souvent dans le cas du palpage, $\mathbf{T}_b$ traduit une identité ou éventuellement une translation suivant l'axe **z** du repère R_{i3}.

II. Mesure de la normale à une face et de sa position.

Pour pouvoir déterminer après palpage la normale mesurée à partir de positions, nous avons besoin de 3 points de palpage situés sur la face. La normale sera obtenue par la moyenne des produits vectoriels de p_1, p_2 et p_3 (voir figure). Pour chaque face, nous situons ces trois points sur un triangle équilatéral centré au centre de gravité de la face.

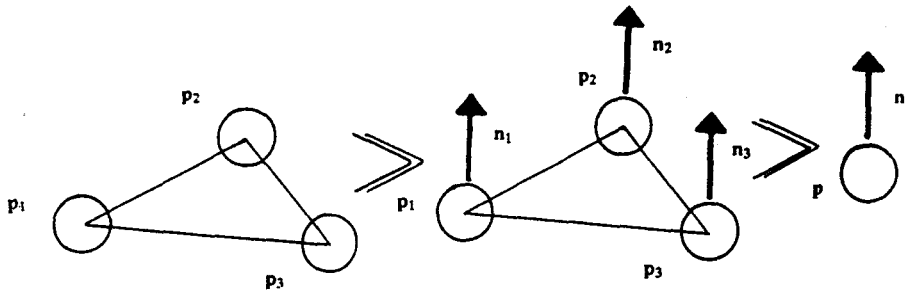


Figure 4.7. Palpage d'une face.

En chaque point palpé, le robot nous fournit la position du centre d'outil.

Trois positions palpées sur une même face nous donnent les coordonnées d'un point sur la face et la normale mesurée par la moyenne des produits vectoriels. Si la normale mesurée est très différente de la normale théorique, on reprend le palpage en prenant pour consignes du mouvement de palpage l'orientation de la normale mesurée et la position mesurée.

III. Calcul de la position de la pièce

Les positions et les normales sur trois faces permettent de déterminer la situation de la pièce dans l'espace robot en prenant, dans le calcul présenté dans le paragraphe suivant, comme normales théoriques et positions théoriques les normales et la position des faces du modèle dans une situation confondue avec le repère de base.

La valeur de **T** vient remplacer la situation théorique de la pièce dans le modèle de la cellule. Deux cas peuvent se présenter :

1. les T_j sont des transformations fixes (pas de positionneur). **T** est alors constante et mémorisée telle qu'elle.

2. la présence d'un positionneur amène une transformation variable. Il faut alors calculer $\mathbf{T}_{j2}$ position de la pièce sur le positionneur.

$$\mathbf{T}_{j2} = \mathbf{T}_{pos}{}^{*-1} \quad \mathbf{T}.$$

ou T_{pos}^{*} est la position estimée du positionneur. Le mouvement du positionneur amène un nouveau calcul de la position de la pièce. Une bonne caractérisation du positionneur est importante.

IV. Critique de la méthode

a. Avantages par rapport aux autres méthodes.

- la détermination de la position se fait directement dans l'espace robot.
- la précision des mesures dépend de la résolution de l'armoire de commande du robot et de la répétitivité du capteur. Les performances classiques de ces systèmes sont de l'ordre du 1/10 de millimètre. Pour obtenir des performances comparables avec des systèmes de vision, il faudrait atteindre pour une scène de 70 cm des résolutions de caméra de 1400 sur 1400 en considérant qu'il faut 2 pixels pour définir un contour.
- La mesure est réalisée uniquement en position : le contrôle de l'orientation est moins important.
- La méthode est entièrement en 3D et tolère donc les rotations de la pièce dans l'espace.

b. Inconvénients.

- le temps d'exécution du palpage est le principal inconvénient de la méthode proposée. Ce facteur impose pratiquement que la répétitivité de la position de la pièce soit assurée. Le palpage n'est alors utilisé que pour "mettre à jour" la position de la pièce dans le modèle de la scène.
- il est peu vraisemblable que la transformation outil et la transformation capteur soient identiques. Cela impose une détermination précise de la transformation capteur.

4.5.2.5 Mise en oeuvre sur robot réel.

La méthode telle que nous l'avons décrite ne fait aucune hypothèse sur le type de robot utilisé, ni sur le type de capteur. Cependant, il nous faut présenter ici les problèmes à résoudre lors de la mise en oeuvre sur un matériel réel.

Le robot utilisé est un ASEA IRB 6-2. Une présentation complète en est faite chapitre 5. Nous n'insisterons ici que sur deux aspects importants de la commande de ce robot et qui interviennent dans le palpage :

Programmation modèle et précision.

. la notion de centre d'outil

. les contrôles de l'orientation.

1. Commande du robot.

   La commande de l'IRB 6-2 autorise la définition d'un centre d'outil (CDO) dans le repère lié à la bride du robot (annexe 3). Une fois un centre d'outil validé, les positions programmées dans le repère robot deviennent celles du CDO. Ceci évite la prise en compte de la composante translation de la transformation outil dans la commande. En orientation cependant, la transformation outil n'est pas déterminée et l'orientation programmée définit toujours l'orientation du repère lié à la bride.

   Il faut donc découpler pour la commande translation et rotation.

2. Position du capteur

   Le robot est un robot 5 axes. Toutes les valeurs $\mathbf{T}_r$, situations de l'extrémité du robot, ne sont pas valides. Le palpage n'est qu'un problème à 5 degré de liberté. Une solution est possible à condition de ne pas mettre l'axe du capteur parallèle au dernier axe du robot (perte d'un degré de liberté). Nous le choisirons donc incliné par rapport à l'axe 5 et dans le même plan. La détermination de la valeur valide de la situation du capteur est réalisée par le premier module de l'algorithme d'inversion du robot (voir chapitre V).

4.5.3 Détermination de la situation d'une pièce.

Nous partons d'une pièce connue dont on connait le modèle polyhédrique. La situation théorique de la pièce est connue et correspond à la transformation $\mathbf{T}^t$ de composante de rotation $\mathbf{R}^t$ et de translation $\mathbf{v}_0{}^t$.

Nous recherchons la position réelle de la pièce de transformation $\mathbf{T}$ différente de $\mathbf{T}^t$.

Pour le calcul de $\mathbf{T}$, nous reprenons l'algorithme proposé par L.W.Grimson et T.Lozano Perez [GRI83]. Cet algorithme part de la mesure pour trois faces, d'un point de la face $\mathbf{p}_i$ et de la normale à la face $\mathbf{n}_i$. Une face f_i correspond à un triplet $(\mathbf{p}_i, \mathbf{n}_i, f_i)$.

La normale théorique à la face i est appelée $\mathbf{m}_i$ et d_i l'offset de la face par rapport à l'origine est $d_i = \mathbf{P}_i \cdot \mathbf{m}_i$.

1. Détermination de la rotation.

   Toute rotation peut être représentée par une direction autour de laquelle s'effectue la rotation et d'un angle de rotation autour de cet axe. Soit $\mathbf{r}$ la direction de la rotation. Toute rotation conserve l'angle entre l'axe de rotation et un vecteur. Il est donc possible

d'écrire que $\mathbf{r}$ doit respecter la relation suivante pour la face i :

$$\mathbf{r} \cdot \mathbf{m}_i = \mathbf{r} \cdot \mathbf{n}_i$$

que l'on peut écrire

$$\mathbf{r} \cdot (\mathbf{m}_i - \mathbf{n}_i) = 0 \qquad (4.3)$$

La relation (4.3) exprime que $\mathbf{r}$ doit être perpendiculaire à $(\mathbf{m}_i - \mathbf{n}_i)$. La même relation peut être formulée pour une face j. L'objet étant supposé inderformable, le même vecteur $\mathbf{r}$ représente l'axe de rotation dans les deux cas. $\mathbf{r}$ est donc colinéaire au produit vectoriel des vecteurs aux quels il doit être perpendiculaire: $(\mathbf{m}_i - \mathbf{n}_i) \times (\mathbf{m}_j - \mathbf{n}_j)$

$\mathbf{r}$ est défini à 180° près. Cette indétermination n'a pas d'importance dans la mesure où le signe de θ lié à l'orientation de $\mathbf{r}$.

L'amplitude de la rotation θ est calculée à partir de l'expression de $\mathbf{m}_i$ dans un repère normé formé par $\mathbf{r}$ et $\mathbf{n}_i$.

$$\mathbf{m}_i = \cos\theta\, \mathbf{n}_i + (1-\cos\theta)(\mathbf{r}\cdot\mathbf{n}_i)\mathbf{r} + \sin\theta(\mathbf{r}\times\mathbf{n}_i).$$

Ce qui permêt de calculer :

$$\cos\theta = 1 - \frac{1-(\mathbf{n}_i\cdot\mathbf{m}_i)}{1-(\mathbf{r}\cdot\mathbf{n}_i)(\mathbf{r}\cdot\mathbf{m}_i)} \qquad \sin\theta = \frac{(\mathbf{r}\times\mathbf{n}_i)\cdot\mathbf{m}_i}{1-(\mathbf{r}\cdot\mathbf{n}_i)(\mathbf{r}\cdot\mathbf{m}_i)}$$

Dans le cas où $\sin\theta = 0$ seule la solution $\theta = \Pi$ est valide.

2. Composante en translation.

Un plan peut être défini comme un ensemble de points dont l'offset par rapport à l'origine suivant la normale est constant. Nous écrirons cet ensemble :

$$\{\mathbf{v} \mid \mathbf{m}_i \cdot \mathbf{v} = d_i\}$$

Où $\mathbf{v}$ est le vecteur représentant un point du plan.

Pour déternimer les trois composantes du vecteur $\mathbf{v}_0$, il est nécéssaire de posséder trois triplets de palpage indépendants, $(\mathbf{p}_i, \mathbf{n}_i, f_i), (\mathbf{p}_j, \mathbf{n}_j, f_j), (\mathbf{p}_k, \mathbf{n}_k, f_k)$, tels que le produit mixte $\mathbf{m}_i \cdot (\mathbf{m}_j \times \mathbf{m}_k)$ est non nul.

Pour chacune des faces i,j,k , on peut calculer l'offset après rotation: $d_i = \mathbf{R}\mathbf{m}_i \cdot \mathbf{v}$. Cet offset doit correspondre à celui obtenu à partir du palpage sans la composante translation: $d_i = \mathbf{n}_i \cdot (\mathbf{p}_i - \mathbf{v}_0)$.

Nous obtenons donc un système de trois équations indépendantes:

$$\mathbf{n}_i \cdot \mathbf{v}_0 = \mathbf{n}_i \cdot \mathbf{p}_i - d_i$$

$$\mathbf{n}_j \cdot \mathbf{v}_0 = \mathbf{n}_j \cdot \mathbf{p}_j - d_j$$
$$\mathbf{n}_k \cdot \mathbf{v}_0 = \mathbf{n}_k \cdot \mathbf{p}_k - d_k$$

Le produit mixte est associatif. Le scalaire que son calcul fournit, exprime le volume du parallélépipède géné à partir des trois vecteurs. Pour déterminer $\mathbf{v}_0$ on cherche à le faire apparaitre dans les équations ci-dessus multiplié par le produit mixte $\mathbf{n}_i \cdot (\mathbf{n}_j \times \mathbf{n}_k)$ qui est égal à $\mathbf{m}_i \cdot (\mathbf{m}_j \times \mathbf{m}_k)$ car le produit mixte est concervé par la rotation.

$$\left[\mathbf{m}_i \cdot (\mathbf{m}_j \times \mathbf{m}_k)\right] \mathbf{v}_0 = (\mathbf{n}_i \cdot \mathbf{p}_i)(\mathbf{n}_j \times \mathbf{n}_k) + (\mathbf{n}_j \cdot \mathbf{p}_j)(\mathbf{n}_k \times \mathbf{n}_i) + (\mathbf{n}_k \cdot \mathbf{p}_k)(\mathbf{n}_i \times \mathbf{n}_j)$$

La vecteur translation est donc déterminé. On s'aperçoit, sans anticiper sur l'étude des erreurs et le choix des faces traitées dans le sous-chaptitre suivants, que le choix des faces devra tendre à rendre le produit mixte suffissament grand pour que le calcul conserve une précision suffisante.

4.5.4 Estimation des erreurs

Comme dans toute méthode qui utilise les résultats de mesures en vue du calcul de certains paramètres, il nous faut étudier comment les erreurs de mesures peuvent se propager dans les calculs, et altérer le résultat final. Deux approches sont possibles pour étudier cette propagation :

* une approche statistique qui prenne en compte à la fois l'aspect aléatoire des erreurs et les éléments géométriques du calcul. Cette méthode a l'avantage de tenir compte de l'amélioration qu'apporte la présence de moyennes dans les calculs. L'écart type est alors divisé par le nombre d'éléments utilisés pour réaliser la moyenne.

* une approche purement géométrique basée sur les volumes ou surface engendrées par le résultat d'un calcul lorsque les variables mesurées varient dans leurs limites.

L'objectif de l'étude des erreurs est non seulement d'avoir une estimation de l'erreur maximum sur un résultat, mais aussi d'étudier les paramètres qui influent sur cette erreur. La seconde approche, si elle amène des erreurs estimées supérieures aux erreurs réelles, offre un bon moyen d'étude de ces paramètres.

4.5.4.1 Erreurs sur le palpage d'une face.

Lors du palpage d'un point sur la face d'un objet, le résultat obtenu va être entaché de plusieurs erreurs:

. ϵ_r : erreur aléatoire due au robot.

. ϵ_{s1} : erreur systématique due au robot.

. ϵ_{s2} : erreur systématique due au capteur.

. ϵ_v : erreur aléatoire due au capteur.

Dans la zone de travail nous avons fait l'hypothèse que l'erreur systématique due au robot est constante. On poura donc concidérer l'erreur systématique globale ϵ_s constante.

$$\epsilon_s = \epsilon_{s1} + \epsilon_{s2}$$

La norme de ϵ_r est appellée ϵ.

Si $\mathbf{n}$ est la normale à la face palpée, $\epsilon_v = \pm \alpha_v \mathbf{n}$.

La normale à la face est calculée à partir du palpage de trois points situés sur un triangle équilatéral de coté d. L'annexe 1 donne pour expréssion de l'erreur sur la normale ϵ_n: $|\partial \mathbf{n}| = \frac{2(\alpha_v + \epsilon)}{d \sin\frac{\theta}{2}}$. Avec une valeur de θ, l'angle entre les cotés du triangle, égal à 60^o le résultat devient:

$$\epsilon_n = \frac{4(\alpha_v + \epsilon)}{d}.$$

La position de la face est calculée sur la moyenne des trois positions obtenues lors du palpage. Les erreurs systématiques apparaissent donc dans ce résultat, à l'inverse du résultat précédant, les erreurs aléatoires seront diminuées . L'erreur maximum sur la position sera obtenu lorsque $\partial \mathbf{p} = \epsilon_r + \epsilon_s + \epsilon_v$ sera la somme de trois vecteurs colinéaires et de même sens.

$$|\partial p| = \epsilon + \alpha_v + |\epsilon_s|.$$

4.5.4.2 Erreurs sur l'orientation.

Nous allons nous intéresser ici à l'erreur sur l'orientation d'un vecteur. L'angle du à un vecteur d'erreur $\partial \mathbf{v}$ est égal à $2 arcsin \frac{\partial \mathbf{v}}{2|\mathbf{v}|}$ ce peut être assimilé, si l'erreur reste faible, à $arctan \frac{\partial \mathbf{v}}{\mathbf{v}}$.

Une bonne manière d'estimer l'erreur sur l'orientation est la valeur de la norme de la projection de $\partial \mathbf{v}$ dans le plan orthoganal à $\mathbf{v}$ soit $|\mathbf{v} \times \partial \mathbf{v}|$. On estimera donc qu'un vecteur parcourt un cône d'erreur et le vecteur d'erreur $\partial \mathbf{v}$ pourra s'exprimer à partir d'un vecteur de référence $\partial \mathbf{v}_0$ et d'un angle α.

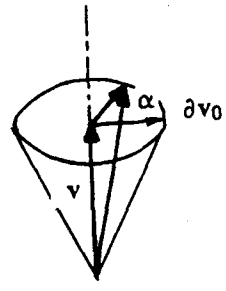


Figure 4.8. Erreur sur un vecteur.

4.5.4.3 Erreur sur l'orientation de l'axe de rotation. L'axe de rotation R est obtenu par le calcul de $(\mathbf{m}_i\text{-}\mathbf{n}_i)\times(\mathbf{m}_j\text{-}\mathbf{n}_j)$. Nous allons donc rechercher le maximum de $|\mathbf{r}\times dr|$ lorsque les vecteurs $\mathbf{n}_i$, $\mathbf{n}_j$ parcourent leur propre cône d'erreur. On exprime $|\mathbf{r}\times\partial\mathbf{r}|$ en fonction des α_i et α_j et l'on dérive le résultat par rapport à ces deux paramètres pour obtenir la valeur maximale.

$$\epsilon_{rot} = \max\left(\frac{\mathbf{r}\times\partial\mathbf{r}}{|\mathbf{r}|}\right) = \frac{\epsilon_n}{16\sqrt{2}\,(1-\cos^2\frac{\beta}{2})\cos\frac{\beta}{2}\sin^3\frac{\theta}{2}\sin\beta_i\sin\beta_j}$$

θ est l'amplitude de la rotation. β est l'angle entre $\mathbf{m}_i\text{-}\mathbf{n}_i$ et $\mathbf{m}_j\text{-}\mathbf{n}_j$. β_i et β_j expriment respectivement les angles entre $\mathbf{n}_i$, $\mathbf{n}_j$ et r .

4.5.4.4 Erreur sur l'amplitude de la rotation.

On applique la même méthode de calcul à $\mathbf{m}_i$ exprimé dans le repère $\mathbf{r}$, $\mathbf{n}_i$, $\mathbf{r}\times\mathbf{n}_i$. L'extrémité de $\mathbf{m}_i$ balaye une elipse étendue dont l'axe maximum exprime la variation de $\partial\theta$ et vaut:

$$max\,(tg\,(\partial\theta)) = \frac{\epsilon_n}{\sin\beta_i} + 2\epsilon_{rot}\cos\beta_i\sin\frac{\theta}{2}.$$

4.5.4.5 Erreur sur la translation.

On dérive la relation de base trouvée en 5.5.3 par rapport à $\mathbf{n}_i$, $\mathbf{n}_j$, $\mathbf{n}_k$. On obtient le résultat

$$\begin{aligned}\mathbf{n}_i\cdot(\mathbf{n}_j\times\mathbf{n}_k)\,\partial v_0 = -b\,v_0 + {} & (\partial\mathbf{n}_i\cdot\mathbf{p}_i + \mathbf{n}_i\cdot\partial\mathbf{p}_i)(\mathbf{n}_j\times\mathbf{n}_k) + \\ & (\partial\mathbf{n}_j\cdot\mathbf{p}_j + \mathbf{n}_j\cdot\partial\mathbf{p}_j)(\mathbf{n}_k\times\mathbf{n}_i) + \\ & (\partial\mathbf{n}_k\cdot\mathbf{p}_k + \mathbf{n}_k\cdot\partial\mathbf{p}_k)(\mathbf{n}_i\times\mathbf{n}_j)\end{aligned}$$

La norme de ce vecteur est surtout fonction des orientations entre les normales aux faces. *b* s'annule si elles sont toutes perpendiculaires et dans ce cas la somme des trois vecteurs $(\mathbf{n}_j \times \mathbf{n}_k), (\mathbf{n}_k \times \mathbf{n}_i), (\mathbf{n}_i \times \mathbf{n}_j)$ a aussi une norme minimum.

4.5.5 Le choix des faces mesurées.

Le but de l'étude qui suit est la détermination de critères qui puissent constituer une métrique pour le choix des trois faces mesurées. En effet, si l'on veut rendre ce choix automatique, le nombre de triplets parmis lesquels il faudra choisir est C_3^n dans lequel n est le nombre de faces de la pièce. Comme nous l'avons montré au chapitre 3, la connaissance d'une métrique permet de limiter l'exploration de l'arbre tout en concervant la possibilité d'aboutir à une solution satisfaisante. Le résultat de l'estimation des erreurs fournit des renseignements précieux sur les paramêtres qui influent sur les erreurs.

4.5.5.1 La taille des faces. L'erreur sur la normale à la face est inversement proportionnel à d , la distance entre les points palpés. Plus la face sera grande, plus d pourra être grande. L'ensemble des résultats sur l'orientation dépendent de l'erreur sur la normale: il est donc important de limiter celle-ci en choisissant les plus grandes faces. Ce critère ne dépend pas de la situation de la pièce: la taille des faces, exprimée par exemple par le diamêtre du cercle inscrit peut donc être précalculée. Les faces trop petites sont éliminées dès cette étape.

4.5.5.2 L'orientation des faces. Les expressions de l'incertitude sur **r** fait apparaitre $\sin\beta_i$ et $\sin\beta_j$ au dénominateur pour la direction et $\cos\beta_i$ et $\cos\beta_j$ au numérateur pour les amplitudes.

Les normales au faces doivent être aussi proches que possible du vecteur orthogonal à l'axe de la rotation.

L'orientation relative des deux faces est donnée par l'angle β qui est présent au dénominateur sous la forme: $(\cos\frac{\beta}{2} - \cos^3\frac{\beta}{2})$. Cette expression est maximale lorsque $\cos\frac{\beta}{2} = \frac{\sqrt{3}}{3}$ soit $\beta \approx 2rd$.

D'autre part, dans l'expression de ∂v_0, l'orientation relative des faces intervient par les termes $\cos a_{ij}$, $\cos a_{jk}$, $\cos a_{ik}$ au dénominateur et $\cos\rho_{i,j,k}$ au numérateur. Lorsque les faces sont orthogonales les premiers termes sont égaux à 1 tandis que les seconds sont nulls.

L'angle souhaitable entre deux faces est donc compris entre 90° et 100°. La correspondance à ce critère pourra être établie à l'avance.

4.5.5.3 Application des critères. L'application des trois critères définis précédemment doit se faire pour sélectionner l'une des meilleures combinaisons de trois faces vis à vis de la précision du résultat. Nous proposons la démarche suivante:

1. Sélection des faces dont la taille est supérieure à un seuil qui peut être exprimé comme une fraction des dimensions maximales de la pièce (exemple 1/5 de la plus grande dimension).

2. Sélection des couples de faces qui réponde au critère d'orientation mutuelle. Il est à noter que le nombre de couples faisant l'objet de cette étape est C_2^m où m est le nombre de faces retenues à l'étape précédente.

3. Les résultats des deux première étapes peuvent être précalculés pour chaque pièce. La connaissance de l'axe de rotation théorique permêt l'application du critère d'orientation par rapport à celui-ci. On recherchera le couple de face qui offre le produit $\sin\beta_i \cdot \sin\beta_j$ maximum.

4. Le choix de la troisième face pour le calcul de la translation se fera sur des critère de perpendicularité au couple précédemment choisi. Si son orientation par rapport à l'axe de rotation est correcte, les deux nouveaux couples créés par ce choix peuvent servir de base à d'autres calculs de $\mathbf{r}$ et θ. La valeur retenue est alors la moyenne des trois valeurs.

La démarche proposée ci-dessus suppose que tout choix permet un choix à l'étape suivante. Cela n'est pas toujours possible et il faudra envisager la possibilité de retour arrière.

4.5.6 Essais et discussion de la méthode.

4.5.6.1 Présentation des essais.

Les essais dont les résultats sont résentés en annexe 2, ont pour but:

* d'une part, d'estimer les erreurs aléatoires sur une opération de palpage. On vérifie par la même occasion que les informations fournies par l'armoire du robot correspondent bien aux variations de la position de l'extrémité du robot.

* d'autre part, de mettre en oeuvre concrétement le palpage dans un cas simple pour valider le principe de la méthode et estimer les peformances que l'on peut en attendre en précision relative.

1. Premier essai:

   Pour le premier essais, une face horizontale a été palpée 70 fois. Chaque palpage a donné lieu à une mesure de la variation de la position de l'extrémité du robot suivant l'axe vertical à l'aide d'un comparateur, et simutanément à l'acquision de la position

fournie par l'armoire du robot. Le palpage est réalisé par la fonction RECHERCHE du robot qui arête le mouvement de palpage au moment du changement d'état du capteur et revient chercher la position de commutation par un léger retour arrière.

Les résultats montrent un très bon couplage des deux séries de mesures, exprimé par le coéficient de régréssion $r = 0.985$. La résolution du robot est donc suffisante pour exprimer les variations de la position de l'extrémité.

La dispersion des mesures montre que la répartition n'est ni uniforme, ni normale: il existe des positions privilégiées pour l'arrêt du robot dans la zone concidérée. On peut cependant estimer la dispertion des positions grâce à la valeur de l'écart type. Sa valeur (0.2 mm) est supérieure à celle qu'on pouvait attendre des caractéristiques du robot et du capteur (0.2 pour le premier, et 0.1 pour le second: caractéristiques qui correspondent à un écart type équivalent global de 0.1 mm).

2. La seconde série d'essais.

Ses essais cherchent à déterminer la situation d'un parallépipède rectangle dans le repère de référence du robot. Le choix d'un parallélépipède rectangle offre une situation favorable car elle permet le palpage de trois faces perpendiculaires et donc la limitation de la propagation des erreurs.

La situation théorique place un coin du parallélépipède parallèlement au repère de base conformément à la figure 4.9. L'orientation de base à mesurer correspondont à une rotation d'axe vertical et d'amplitude importante. Les divers essais consistent à lui composer diverses rotation d'axe horizontaux et de plus faible amplitude. On pourra mettre ainsi en évidence la sensibilité de la mesure des rotations de faibles amplitudes.

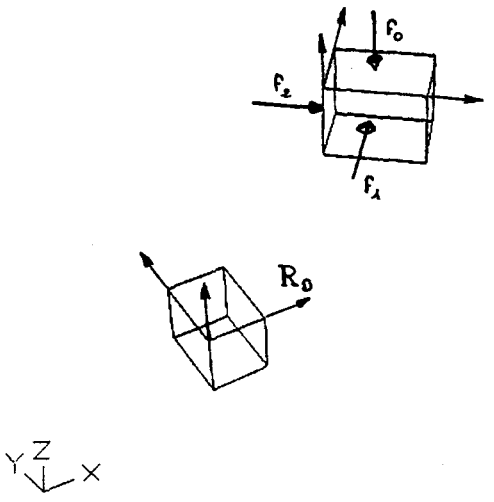


Figure 4.9. Situation de base du palpage.

4.5.6.2 Résultats des essais.

Les résultats sur une série de six palpages différents, et dont les conditions d'expérimentation sont présentées en annexe 2, montrent que l'on obtient une erreur de deux à trois degrés sur l'orientation de l'axe de rotation. Les projections du vecteur **r** sur le plan horizontal sont présentées ci-dessous. On peut remarquer que pour les tests B1, V1, V2 qui correspondent tous les trois à une rotation sur le plan de la table support, les résultats obtenus pour **r** sont très proches. Ceci nous a incité à rechercher les causes d'une telle erreur systématique, le même décalage se retrouve pour les autres essais. Trois hypothèse peuvent être émises:

1. la surface qui sert de support à la pièce n'est pas horizontale. Cette hypothèse n'est pas vérifiée car l'angle entre la normale à la face horizontale et la verticale est de 0.6°.

2. les faces verticales ne sont pas parfaitement perpendiculaires à la face horizontale. Le calcul des angles entre les faces sur la base des normales mesurées montrent que les deux faces verticales sont perpendiculaires entr'elles mais qu'elles font un angle de 87° et88° avec la face horizontale. Une vérification sur l'objet a bien confirmé ces résultats.

3. l'erreur est due aux imprécisions sur les mesures et à leur propagation dans les calculs. Cette composante ne peut être totalement évacuée mais son influence est limitée (sensiblement inférieure au degré) sauf peut-être dans le cas du test HHV2 (erreur de 5.2°).

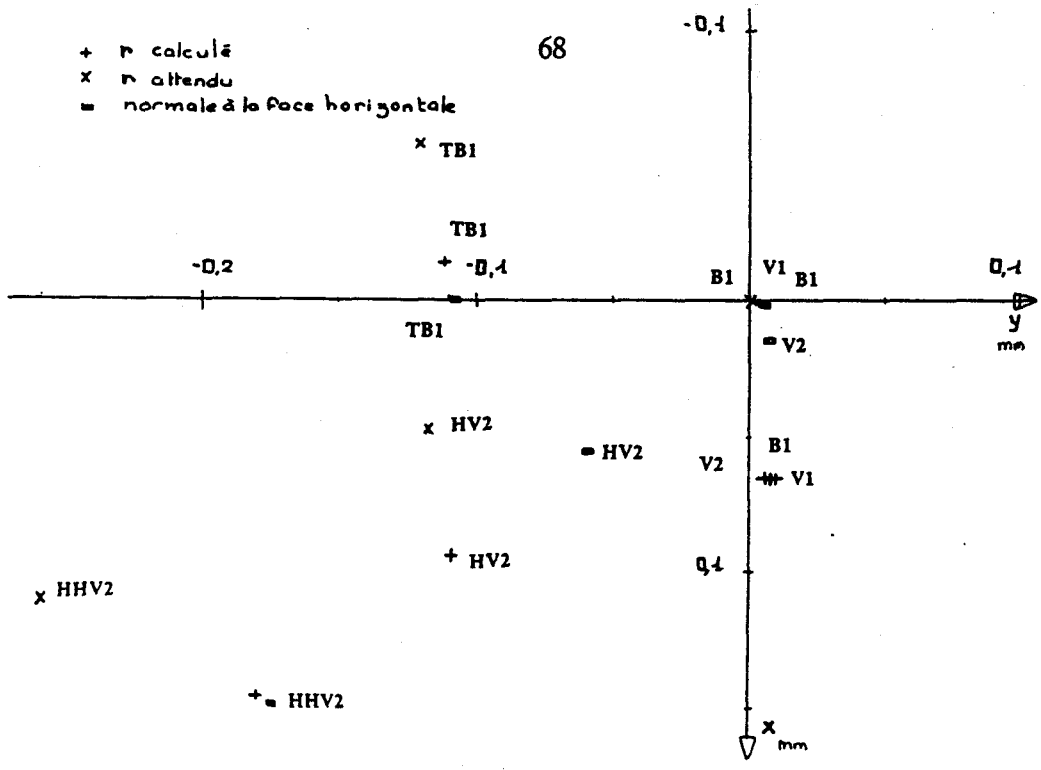


Figure 4.10. Essais: projection des axes de rotation sur le plan horizontal.

Les variations de la projection de HV2 et TB1 par rapport aux orientations de référence V2 et B1 sont conformes à la variation de l'axe de rotation théorique ou aux calculs qui peuvent être éffectués par combinaison de deux quaternions (voir [FAU86]).

Les amplitudes obtenues pour la rotation correspondent par contre parfaitement aux valeurs attendues: moins d'un degré d'ecart.

Les résultats sur l'orientation sont donc très satisfaisant. Le test de la conformité géométrique de la pièce est même un résultat annexe que l'on pourra en tirer. Ces résultats sont d'autant plus intéressant qu'ils montrent que l'on peut détecter de faibles variations angulaires à conditions qu'elles soient couplées à des rotation d'amplitude importante. En effet le calcul de l'orientation de V2 en prenant comme angle de rotation 10° donne une erreur de 16° sur l'orientation de **r**.

Les erreurs sur les translations sont beaucoup plus importantes de l'ordre de 10mm, comme l'indique le tableau ci-dessous. Cela est probablement du au fait que les essais ont été réalisés à orientation constante des capteurs. A cette limitation s'ajoute une autre contrainte due au volume de traivail du robot qui interdit le palpage d'une face verticale avec l'inclinaison du capteur choisie. D'autre part la détermination de la référence de l'origine du repère sur la feuille de mesure est beaucoup moins précise que le dégauchissage en orientation.

En appliquant la démarche proposée au paragraphe 5.5.2.3 sur le choix du capteur ces résultats devraient être fortement améliorés.

ESSAI	X	Y	Z
B1	-6.7	-3.8	-
V1	-6.8	-3.65	-3.11
V2	-5.7	-5.3	-0.7
HV2	-4.6	+6	-
HHV2	-0.5	+6	-
TB1	-15.6	-1	-

Figure 4.11. Erreurs en translation.

Les résultats obtenus ont été réalisé avec un capteur qui amène une dispersion des mesures de l'ordre de $0.6mm$. Un capteur autorisant une recherche plus précise, potentiomètre linéaire par exemple, peut amener des résultats meilleurs notament en position.

4.5.6.3 Discution de la méthode.

La méthode proposée répond bien aux objectif qui lui ont été fixés.

1. On est capable de déterminer la situation absolue d'une pièce dans le repère de base du robot. Les déplacements programmés à partir de cette position seront donc uniquement entachés de l'erreur relative de positionnement.

2. Ce résultat est obtenu à partir de neuf données de palpage ce qui correspond à une durée de palpage de 40 secondes, les trajectoires de palpage n'étant absoluement pas optimisées.

   Le fait de calculer les moyennes (position,normale) en éliminant les résultats qui peuvent être entachés d'erreur importante, améliore sensiblement les résultats en limitant leur dispersion. Ce fait n'avait pas été pris en compte dans les calculs d'erreur.

   L'estimation des erreurs fournit de bon critères pour estimer la validité du résultat obtenu. Le terme multiplicatif de ϵ_n dans l'erreur sur la normale inclue tous les paramètres du problème en orientation.

   - amplitude de la rotation

   - angle entre les faces;

   - angle entre les normales aux faces et l'axe de rotation.

   La connaissance de la situation théorique de la pièce permet de calculer ce terme et si sa valeur est trop élevée de modifier l'orientation de référence pour limiter l'influence du

terme $\sin^3\frac{\theta}{2}$.

Le produit mixte des trois normales offre pour la position le même type de critère qui dépend uniquement du choix des faces.

Quelques soient les solutions adoptées, il y aura toujours des pièces dont la situation sera difficile à mesurer: un exemple en est donné ci-dessous. L'intérêt des critères proposés est de savoir que le resultat sera entaché d'incertitude. On pourra alors:

* prendre un capteur plus précis
* augmenter le nombre de mesures afin de pouvoir moyenner les résultats.

4.6 CONCLUSION.

L'évaluation des erreurs tout au long de la chaine cinématique mêt en évidence tant de causes d'erreur, jeux, discrétisation, modélisation... que l'on pouvait se demander si la programmation hors ligne permettait d'atteindre une bonne précision à l'éxécution.

Les travaux présentés, sur la base de documents bibliographiques, ouvrent diverses voies pour obtenir cette précision: identification des modèles, prise en compte lors de l'exécution.

Les essais que nous venons de présenter montre que l'armoire de commande du robot IRB6-2 inclue déjà un modèle précis du robot et peut fournir une indication de la situation de son extrémité avec une bonne précision. La méthode proposée pour la détermination de la situation de la pièce dans l'espace est validée par ces mêmes essais malgrè la mise en oeuvre imparfaite du palpage.

Le chapitre suivant inclut cette étape de recalage de la situation de la pièce dans un module spécifique du logiciel de programmation par simulation graphique.

5. POSTE DE PROGRAMMATION GRAPHIQUE.

Présenter un logiciel représente toujours un défi : il faut éviter l'accueil de la notice d'utilisation qui en donne une approche trop proche de l'utilisation sans se perdre pour autant dans les détails de l'analyse et de la programmation. Tout ne sera donc pas dit du poste de programmation graphique. L'attention sera portée sur deux approches différentes : une description de l'organisation du logiciel, d'une part, une analyse plus détaillée des fonctionalités mises en oeuvre, d'autre part.

Cette réalisation est l'aboutissement de trois années de travail dans le domaine de la simulation graphique et de la soudure robotisée. Ce temps a permis de situer le produit en cours d'élaboration par rapport à d'autres produits similaires, de l'enrichir aussi de ses confrontations. Le soudage à l'arc, d'autre part, demandait à être approfondi pour adapter les fonctionalités proposées à ce domaine. Le cadre de travail, avec les sites expérimentaux, tiennent une grande place dans notre connaissance pratique de la robotique et de l'informatique.

Ce chapitre, qui constitue en fait la seconde partie de cette thèse, abordera donc :

* les systèmes de simulation de robots existant à partir de quatre produits;
* la soudure à l'arc robotisée, avec la présentation du procédé de ses contraintes et de son application actuelle à la robotique.
* les sites expérimentaux, description des matériels et des logiciels.
* l'application proprement dite, avec la présentation et la critique d'une première étude et la présentation du poste de programmation graphique de robot.

5.1 SYSTEMES DE SIMULATION DE ROBOTS.

Les quatre systèmes présentés ici ne peuvent à eux seuls représenter l'ensemble des travaux, menés dans de nombreux laboratoires, sur la simulation graphique de robot (on pourra aussi se référer à [HOR86] [WEC84] [NEM85][SJO83]). Ils présentent l'intérêt d'être opérationnels et de représenter des approches diverses du problème.

5.1.1 Catia-Robotique." [DOM84] [LIE84]

1. Présentation

   Le module robotique de CATIA a été développé par le Laboratoire d'Automatique et de Micro-mécanique de Montpellier (LAMM).

   Il utilise la base de données tridimensionnelles du système de CAO. Celle-ci utilise des tables qui représentent des éléments géométriques (points, ..., surfaces, volumes...) de manière géométrique et graphique. Des pointeurs définissent les relations entre différents éléments, ou ensemble d'éléments.

   Le module robotique a nécessité l'ajout d'un élément articulation et d'un élément tâche. A partir du premier, n'importe quel robot peut être défini. Le second permet de définir un programme comme un ensemble de tâches. Une tâche définit la liaison entre un élément du robot et un élément de l'environnement. Les éléments peuvent être des points, les lignes, ou des plans.

2. Points à retenir

   * La notion de structuration des données géométriques et son utilisation cohérente pour définir tant les robots que les tâches à accomplir.
   * La souplesse offerte par le mode d'inversion itérative à partir du modèle variationnel. Cette méthode de calcul permet aussi bien de trouver les positions articulaires que les couples aux articulations d'une structure de robot quelconque.
   * La représentation graphique utilise un modèle polyhédrique affiché avec une visualisation en faces cachées.

3. Configuration matérielle.

   CATIA ne tourne que sur des matériels IBM de grosse taille (4341, 3033) associés à une console graphique IBM 3250. L'implantation de CATIA sur la machine à architecture RISC 6150 permet d'envisager le module robotique sur cette machine dans un avenir proche. Cette "gourmandise" en moyen de calculs s'explique par les choix faits tant pour la visualisation (faces cachées) que pour l'inversion (méthode itérative).

4. Programmation hors ligne.

   Ce n'est pas la principale utilisation de CATIA-Robotique qui est plus utilisée pour la conception de postes robotisés. Les utilisateurs sont principalement les sociétés d'engeneering qui peuvent utiliser à fond la souplesse offerte par CATIA.

   Le Centre Technique de Robotique Industrielle de la Générale de Mécanique Aéronautique (Filiale des Avions Marcel Dassault) utilise cependant CATIA robotique avec succès pour la programmation des robots. Cela a imposé, comme l'explique A. Risbourg [RIS85], des interfaces logicielles et matérielles pour prendre en compte

automatiquement les erreurs, et recaler le programme.

5.1.2 Robotics."[SOL 84]

1. Présentation

Robotics est un logiciel spécifique de simulation et de programmation de cellule robotisée proposé par Mc Donnell Douglas Automation. Il est composé de quatre modules indépendants :

. Build pour créer de nouveaux robots

. Place pour la conception et l'évolution d'une cellule robotisée

. Command pour créer des programmes robot hors ligne

. Adjust sert à ajuster le modèle graphique au monde réel.

Une séquence de programme est une suite de répères définis par l'opérateur directement sur l'écran. Des fonctions du type GOTO... permettent de commander les mouvements. Place prend les modèles de la bibliothèque de robots (d'éléments articulés) permet de les placer les uns par rapport aux autres. La manipulation des représentations des volumes de travail autorise la recherche d'un placement optimum par rapport à la tâche à accomplir.

2. Points à retenir

* L'aspect modulaire du logiciel
* L'inversion des robots fait appel à des algorithmes spécifiques paramétrés contenus dans la bibliothèque de robots.
* La représentation graphique est filaire et ne comprend que des cercles et des droites.

3. Configuration matérielle

Robotics est supporté par un VAX 11/750 (DEC) couplé à un terminal graphique rapide de Evans et Sutherland, le PS 300. Cette configuration autorise une simulation très rapide des mouvements proche des temps d'exécution sur site.

4. Programmation hors ligne

La programmation hors ligne requiert deux des quatre modules de Robotics.

Command comporte le post-processeur qui "traduit" les points sauvegardés par Place dans le format adapté à l'armoire de commande du robot, il active les protocoles nécessaires à la transmission du programme.

Adjust prend en compte les modifications apportées sur site pour les intégrer dans le programme. E.SOL, dont l'article nous a servi pour cette présentation, souligne que le temps le plus long n'est pas la programmation du robot mais la saisie du modèle de la pièce.

5.1.3 Robographix.

1. Présentation

Robographix est le logiciel de programmation hors ligne interactive graphique de la société Computer Vision. Il utilise les primitives de modalisation géométrique en trois dimensions et la base de donnés graphique du logiciel de CAO CADDS-4X.

L'environnement est un ensemble d'objets classiques de la base de données, seuls les robots et les effecteurs sont traités de manière spécifique.

2. Points à retenir

   * Les trajectoires sont des éléments de la base de donnée graphique. Elles sont définies par déplacement interactif de l'effecteur dans le repère de base.

   * L'inversion des robots se fait sur la base de modèle mathématique propre à chaque robot.

   * La visualisation est filaire, la modélisation est non polyhédrique.

3. Configuration matérielle

   Computer Vision propose ses logiciels sur ses propres stations de travail graphiques, la série CDS 4000, qui peuvent être assimilés à des mini-ordinateurs spécialisés.

4. Programmation hors ligne

   Les données en code robot sont générées directement à partir de la base de donnée graphique par des post-processeurs spécifiques aux robots. Le module CAMACS assure la communication des données entre le robot et le système Computer Vision.

   Un utilisateur important du Robographix en France est le groupe PSA. Jean-Guy Queromes [QUE 82] présente les intérêts de l'utilisation de la CAO en robotique sur la base de cette expérience. L'aspect programmation se heurte, d'après lui, "à un problème technologique. Deux robots ne réalisent pas le même travail. Cela provient principalement de fabrication et des difficultés très importantes liées au calage des capteurs de déplacement des différentes articulations".

   Il propose trois solutions :

   - connaissance parfaite des caractéristiques de tous les robots pris individuellement. Ceci nécessite l'élaboration d'un cahier des charges des mesures à faire dans les laboratoires de métrologie, afin de personnaliser les programmes en fonction des robots.
   - phase d'apprentissage très succinte durant laquelle l'opérateur affine sur le site la position des points préalablement programmés.
   - utilisation de capteurs sur les robots permettant de se recaler sur les points (sens tactiles et de la vision par exemple). Ce sont des robots en "boucle fermée".

   Les deux premières sont celles qui ont été retenues par PSA qui a constitué une véritable base de données des caractéristiques individuelles de chaque robot.

   Le soudage par point des carosseries de voitures (ferrage) est l'application principale de la programmation hors ligne des robots chez PSA. Cette programmation est réalisée directement à partir de la description des carosseries issues de leur conception sur le logiciel de CAO.

   Marc Bourez [BOU83] a utilisé le même système pour la programmation d'un robot de pulvérisation mis en oeuvre pour le dégraissage de pièces dans un atelier flexible. Il affirme quant à lui que "les mesures réalisées sur le robot montrent que les coordonnées générées par le système de CAO sont reproduites par le robot avec une précision meilleure que 0,5 mm et qu'aucune procédure de calibration n'a été nécessaire".

   La différence d'approche provient sans doute de la nature de données de programmation. Dans le premier cas, le traitement est réalisé en coordonnées articulaires, dans le

second directement en coordonnées cartésiennes. La fixation sur palette dans le cas de l'atelier flexible assure d'autre part une position beaucoup plus précise de la pièce que dans le cas du ferrage.

5.1.4 Simulateur de l'Université de Tokyo." [SAT 82]

Parmi les nombreux systèmes de simulation développés dans des laboratoires de robotique [DIL 86] [SJO83] [HOR86], celui de l'Université de Tokyo est intéressant à plusieurs titres :

- il sert de référence à la plupart des autres travaux
- il présente la démarche complète de la programmation.

En effet, les systèmes de simulation sont souvent utilisés pour présenter les résultats d'algorithmes, pour montrer la validité d'une trajectoire générée mais rarement pour la programmation réelle d'un robot.

I. Présentation.

Le logiciel est décomposé en trois modules :

* la modélisation géométrique qui fournit un modèle de l'environnement
* la simulation de programme robot qui produit des données de mouvements
* la génération de programme de contrôle du robot réel.

La modélisation géométrique est réalisée par le logiciel de modélisation solide GEOMAP qui fournit en plus des fonctions classiques de type de logiciels :

. une fonctionnalité de détection d'interférence entre objets

. un classement des modèles générés en vue de leur réutilisation.

La programmation peut se faire du niveau actionneur au niveau objectif. Dans ce dernier cas, des macro spécifiques définissent les informations à extraire des données géométriques (position ...) et génèrent la séquence d'actions (niveau effecteur) associées. Les informations de mouvements sont ensuite traduites en langage VAL et transférées sur le PUMA.

II. Points à retenir

- La détection d'interférence est automatique
- La modélisation géométrique se fait à partir d'objets standards (cylindre, cube...)
- La visualisation est réalisée avec faces cachées
- Un langage de haut niveau, avec la notion de macro associée est utilisée dès la simulation.

III. Configuration matérielle

Aucun élément n'est donné sur ce point.

IV. Programmation hors ligne

Les résultats des mouvements générés sous VAL et validés par la simulation ont été testés pour deux types d'applications : empilage de cubes, simulation de peinture autour d'un cylindre (orientation normale à la surface, distance constante à la pièce).

Poste de programmation graphique

Il est à noter que ces applications ne nécessitent que des précisions limitées pour réussir.

5.1.5 Conclusions et perspectives pour un poste de travail.

De ces quatres exemples et de la lecture d'autres articles (une présentation comparative de plusieurs systèmes commercialisés est présentées dans [DET87]), voici les points qui peuvent être retenus.

1. Performances et moyens de calcul

   Les performances recherchées ne sont pas toujours les mêmes : souplesse et face cachée dans CATIA, rapidité de l'affichage et représentation filaire dans ROBOTICS.

   Un affichage filaire avec une rapidité moyenne, à partir d'un module d'inversion paramétré, doit permettre de limiter les moyens de calcul nécessaires sans trop pénaliser l'interactivité.

2. La modélisation solide

   Si celle-ci peut être plus ou moins riche, opération sur les solides dans [SAT82] et CATIA, juxtaposition d'éléments dans ROBOTICS, le point de départ est toujours un ensemble d'éléments standards (cylindre, parallélépipèdes,...) que l'on manipule.

3. Structuration modulaire

   L'ensemble des logiciels présente à peu près tous la même structure.

   * modélisation des composantes de l'environnement (robot, pièces,...)
   * organisation de l'environnement en indiquant les positions relatives des composantes
   * animation de l'environnement (simulation de programme, commande spécifique d'animation...)
   * transcription et transfert

   Seul ROBOTICS propose une phase d'adaptation à la situation réelle de l'environnement.

4. Détection d'interférences

   Elle est réalisée :

   . par le logiciel de faces cachées

   . par l'opérateur en utilisant de manière interactive la rapidité d'affichage de la console graphique

   . de manière automatique.

   Chaque système utilise pour ce point les atouts de la configuration choisie.

5. Modélisation de la pièce

   Nous retenons, pour l'avoir aussi expérimentée (voir V.2) la remarque de [SOL 84] sur le temps de création de la pièce. Les outils de modélisation doivent offrir des utilitaires qui permettent de diminuer ce temps. Dans le cas d'utilisation de logiciel de CAO (CATIA, CADDS'4X) la modélisation est réalisée à l'aide du logiciel. Pour un poste de travail

autonome, cette fonction est à développer.

Une autre approche peut être aussi l'extraction des données de la pièce à partir d'une modélisation pré-existante sur un système de CAO.

6. Conclusion

Le développement d'un poste de programmation de robot par simulation graphique n'est possible que si l'on accepte de faire des choix quant aux performances recherchées.

Outre les aspects communs à tous les systèmes de simulation graphique de robot, celui-ci doit utiliser un maximum de possibilités de la connaissance de l'environnement par la modélisation et de la spécialisation du poste à un type de tâche particulière.

5.2 SOUDAGE A L'ARC ET ROBOTIQUE.

Le soudage à l'arc est un secteur de l'industrie qui représente en France 15 % du parc de robots installés. Les entreprises utilisatrices de robots dans ce domaine sont de tailles variées : de l'industrie automobile aux entreprises de sous-traitance, de la PME de machinisme agricole [AXE 86] à l'artisan qui travaille seul avec un seul robot [JRO85].

La soudure à l'arc est un domaine exigeant (nous étudierons les contraintes que posent les procédés dans le paragraphe 5.2) et de nombreuses études ont été menées pour réaliser un mode de compliance active adaptée au procédé : le suivi de joint [BJO86] [STE86]. Celui-ci à pour buts principaux de limiter la précision requise pour la programmation et de prendre en compte les variations dimensionnelles des pièces à assembler.

Nous nous intéressons pour le travail qui suit à l'aspect programmation de la soudure robotisée qui est un aspect complémentaire du suivi de joint. Mais rappelons d'abord le principe du soudage à l'arc et les différents procédés qui existent.

5.2.1 Le soudage à l'arc.

5.2.1.1 Principe du soudage à l'arc.

Le soudage est la liaison rigide de deux pièces par un métal d'apport. Lorsque le métal est de même nature que celui des pièces, la soudure est dite autogène.

Le principe du procédé de soudage à l'arc est la création d'un arc généré par la tension entre la source d'apport du métal de polarité positive et la pièce de polarité négative. Dans les procédés, MIG et MAG, le métal fondu par l'arc est protégé par un flux gazeux, soit inerte (MIG), soit légèrement oxydant (MAG).

L'électrode d'apport de métal est un fil nu de 0,8 à 1,6 mm de diamètre poussé à vitesse constante dans un tube-contact en cuivre formant amenée de courant. L'arc jaillit entre la pièce et le fil qui fond au fur et à mesure de son avance.

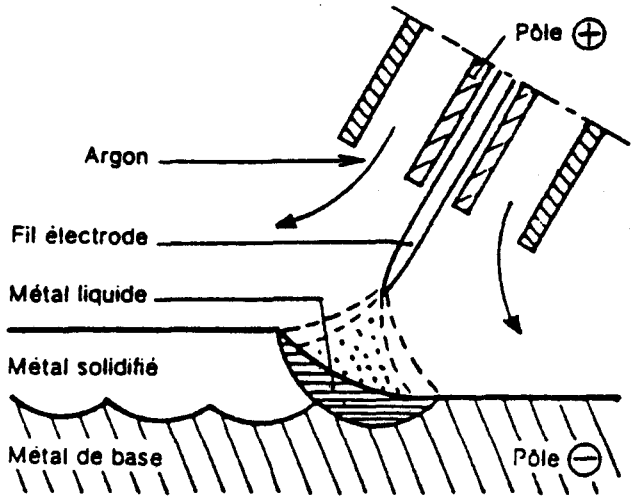


Figure 5.12. Principe du soudage MIG

Il existe plusieurs types de transfert du métal fondu (voir annexe 3) selon le type de gaz et l'intensité du courant (l'annexe reprend les figures représentant les différents types :

* par court-circuit : l'intensité du courant est faible, l'arc est court, tension 14 à 22 V.
* par pulvérisation axiale : l'intensité du courant est forte, l'arc est long, le métal forme de fines goutelettes qui sont guidées par le champ dans l'arc. tension 28 - 36 V.
* régimes intermédiaires : correspond à des règlages intermédiaires dus à des flux gazeux différents ou au type de soudage.
* le régime à arc pulsé : le transfert s'effectue à raison de une goutte de métal par impulsion de courant. Ce procédé plus récent est plus souple d'emploi.

L'équipement (annexe 3) comprend :

- - un générateur courant continu
- - un dévidoir de fil à vitesse régulée
- - une torche d'amenée de fil, de gaz et du courant qui est refroidie par circulation d'eau
- - une gaine contenant les arrivées de courant, fil et gaz et la circulation d'eau.
-

Ce procédé récent, première application industrielle en 1948 aux USA, a connu et connaît encore beaucoup d'évolutions qui devraient lui permettre de se maintenir face à d'autres procédés de soudage (faisceaux d'électrons, laser...).

5.2.1.2 Les contraintes.

Le résultat d'une soudure va être évalué par rapport au bain de fusion, c'est-a-dire à la zone de métal qui aura été fondue entre les deux pièces. La profondeur du bain, sa largeur, l'épaisseur et la forme du cordon qui en résultent sont autant de paramètres à obtenir.

Ils vont dépendre du réglage de la source de soudage (intensité, vitesse d'avance du fil...) mais aussi de paramètres de positionnement entre la torche et la pièce, et de la vitesse d'éxécution

de la trajectoire. On définit la distance torche-pièce qui est plus exactement la distance anneau d'amenée du courant/pièce (voir figure) comme paramètre de position. L'orientation de la torche sera fixée par deux angles (avance, inclinaison) en référence à la pièce. La vitesse devra être la plus réguliere possible (de l'ordre de 100 mm/s) et la trajectoire doit être linéaire ou circulaire entre deux points.

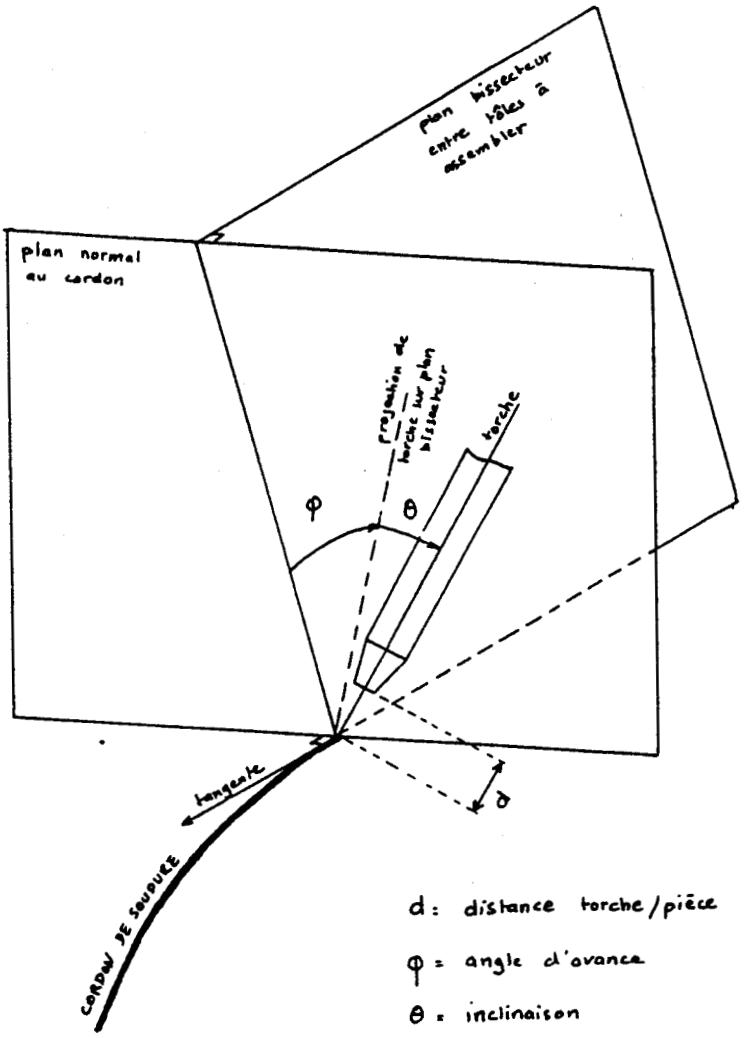


Figure 5.13. Position torche-pièce

1. La distance torche-pièce

   La distance torche-pièce varie suivant le type de régime employé (12-15 mm en court-circuit, 18-25 mm en pulvérisation axiale et 12-25 mm en arc pulsé).

   Pour un régime donné, si d diminue, la pénétration de la soudure augmente. La précision requise est de l'ordre de ± 2 mm.

2. L'angle d'avance ϕ

   ϕ peut être positif (soudage tiré) ou négatif (soudage poussé) par rapport au sens d'avance de la soudure.

   ϕ est de l'ordre de 5°.

   Le choix entre poussé ou tiré dépend du matériau, de la position du cordon de soudure (horizontal, incliné...).

   ϕ négatif donne un cordon large, peu bombé et peu profond, il y a risque de collage ou de soudure mauvaise.

   ϕ positif produit un cordon plus étroit et plus profond, mais bombé. En soudage automatique, ϕ est souvent nul.

3. L'angle d'inclinaison latérale θ .

   On recherche le plus souvent le plan bisecteur des deux tôles. Il est possible de s'éloigner de cette règle dans des cas particuliers (soudure tôle sur tôle) ou pour modifier la forme du cordon.

   Dans l'ensemble, la précision recherchée sur les angles sera de quelques degrés.

   L'ensemble de ces paramètres sont liés et leur détermination automatique en vue d'un résultat donné n'est pas encore possible bien que des travaux soient en cours, à L'Institut de Soudure, dans le domaine de la soudure plane en vue de la constitution d'une base de donnée de paramètres de soudage. On retiendra que pour un réglage donné et une vitesse donnée, le paramètre de position est déterminant pour l'obtention d'une bonne soudure. Ceci explique les efforts réalisés pour maîtriser le positionnement relatif de la torche et de la pièce.

5.2.1.3 Le maintien de la géométrie de la pièce.

A cause les différences de température auxquelles la pièce soudée est soumise, des contraintes internes se développent. Elles provoquent des déformations indésirées de la pièce.

Pour limiter ces déformations, plusieurs approches sont possibles :

1. jouer sur la séquence de soudage en soudant successivement à des endroits différents de la pièce afin de mieux répartir et de diminuer les gradients de température par une température plus homogène de la pièce. L'expertise dans ce domaine revient au soudeur.
2. pointer la pièce, c'est-a-dire réaliser de petits cordons de soudure liant les éléments de la pièce afin de rigidifier celle-ci. Cette opération est bien souvent manuelle.
3. brider les éléments de la pièce : des fixations empêchent les déformations et les mouvements mutuels des tôles. Les dispositifs utilisés (gabarits, ...) sont encombrants et doivent être étudiés pour chaque pièce.

5.2.1.4 Application au soudage robotisé.

Pour définir la tâche de soudage en vue de sa robotisation, nous pouvons dire :

- qu'elle évolue réellement dans l'ensemble de l'espace (en position et en orientation) ;

- qu'elle impose une précision qui dépend du procédé et de l'épaisseur de la tôle. Une précision de l'ordre du millimètre semble cependant nécessaire ;
- qu'elle évolue dans un espace encombré par la pièce qui peut être de forme complexe, par le gabarit de maintien des tôles;
- qu'une expertise importante revient au soudeur.

5.2.2 La robotisation du soudage.

5.2.2.1 La cellule de soudage robotisée.

La cellule classique en soudage à l'arc robotisé est constituée comme suit:

. un robot qui porte la torche de soudage ;

. un positionneur qui porte la pièce. Le positionneur à souvent 2 plateaux afin de permettre le chargement-déchargement sur l'un pendant que le robot soude sur l'autre.

Le positionneur peut être indexé (incrément de rotation fixé) ou non indexé à 1, 2 ou 3 axes de rotation asservis afin de permettre l'orientation de la pièce dans l'espace.

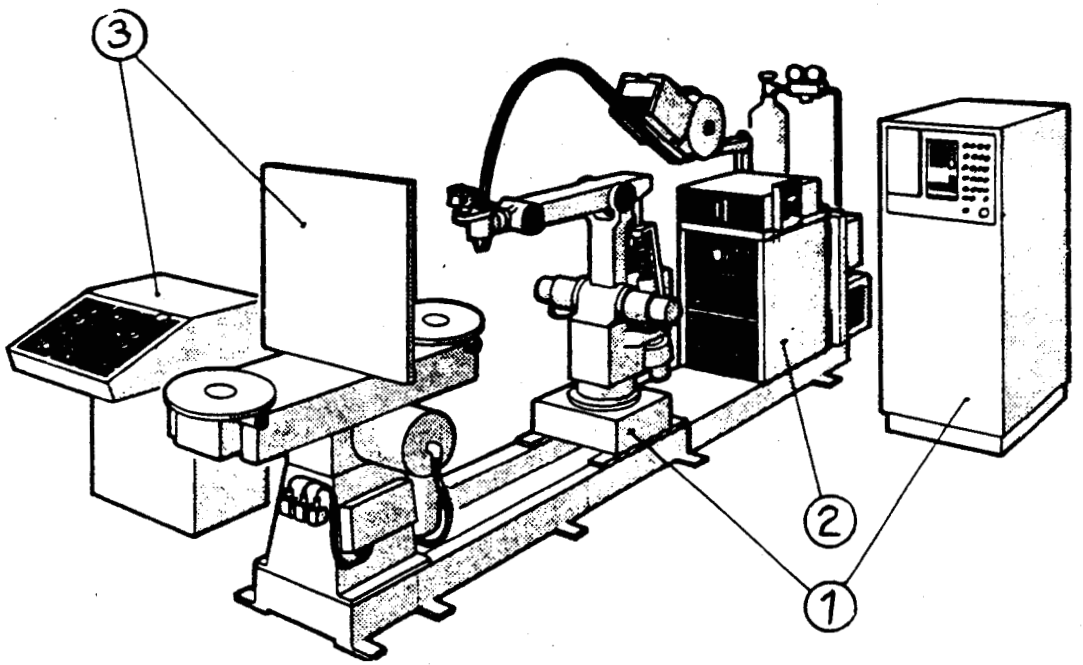


Figure 5.14. Cellule de soudage

Les robots utilisés sont de structures variées, cartésiens cylindriques ou sphériques. Ils peuvent présenter 4, 5 ou 6 axes. L'utilisation d'un suivi de joint nécéssite pratiquement un robot 6 axes car ce dispositif impose la dernière orientation de l'extrémité du robot. Cette tendance au 6ème axe est renforcée par la recherche d'une plus grande souplesse dans le cas où le suivi de joint n'est pas utilisé.

5.2.2.2 La programmation.

Tous les fournisseurs de robots offrent un logiciel de programmation adapté au soudage. En plus des fonctions classiques de programmation des trajectoires du robot par apprentissage, celui-ci intégre les possibilités de contrôle du poste de soudage, de trajectoires particulières,

Poste de programmation graphique

appelées balayage, de suivi de joint (soit par contrôle des parametres d'arc, soit par capteur externe) et de contrôle du positionneur. On trouve aussi parfois des fonctions de recherche de joint ou de palpage de la pièce.

L'ensemble de programmation est donc conçu pour fournir au soudeur un environnement complet de contrôle de la cellule de soudage et apporte une certaine souplesse à la programmation.

5.2.2.3 Le mode d'exploitation.

L'exploitation classique d'une cellule de soudage impose un opérateur pour charger et décharger les pièces. Ce qui amène à dire qu'en soudage, le soudeur a été remplacé par un manutentionnaire [AXE 86].

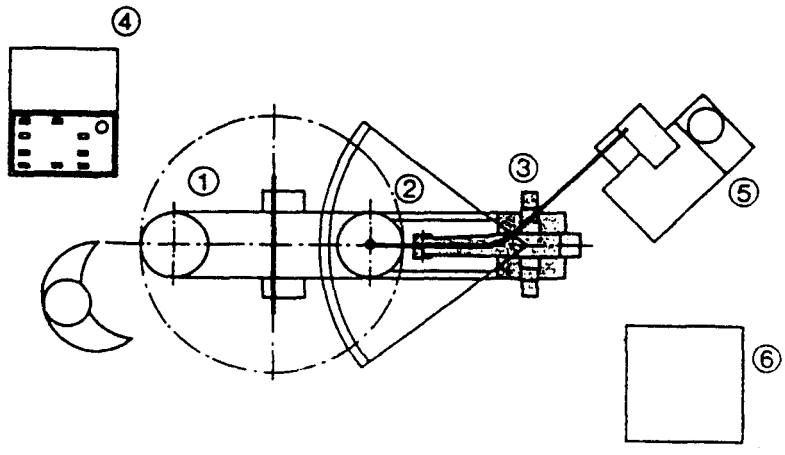


Figure 15. Configuration typique d'un poste travail.

Le mode d'exploitation est alors le lancement par lot de taille importante. Entre chaque lot, la cellule doit être programmée et reconfigurée.

5.2.3 Conclusion.

Si le soudage représente un secteur important de la robotique, le mode d'implantation des robots à travers le concept de cellule de soudage robotisée n'est pas satisfaisant à plusieurs titres:

* le rôle de l'opérateur est fortement déqualifié ;
* le mode d'exploitation est rigide et ne permet l'implantation de robot que pour des séries de pièces importantes.

Les industriels qui les utilisent signalent pourtant des avantages à la robotisation :

* qualité constante du produit permettant une reproductibilité des caractéristiques de la pièce (résistance mécanique, géométrie...) ;
* maintien des cadences de fabrication et donc, gestion prévisionnelle simplifiée.

Il semble cependant (et certaines implantations vont dans ce sens [AXE86]) que le développement de la robotique de soudage passe :

Poste de programmation graphique

* par la robotisation de la manutention des tôles et des pièces ;
* par le pointage automatique hors référence ;
* par le développement d'outils de conception et de programmation hors ligne des robots adaptés au soudage.

Si le travail rapporté dans cette thèse concerne exclusivement le dernier point, nous présentons en conclusion les travaux en cours dans ce domaine dans le département d'automatique-robotique de l'ISEN.

5.2.4 LES SITES EXPERIMENTAUX.

Le travail présenté dans cette thèse n'a pas pour support un site expérimental donné mais plusieurs. La multiplicité des sites provient des différents laboratoires qui sont impliqués dans cette recherche (voir présentation Chapitre I). Les matériels sont différents, un certain nombre de points communs et de choix permettent cependant l'articulation d'un travail sur ces trois sites.

5.2.4.1 Les sites robotiques.

Le poste de programmation graphique est conçu pour être implanté dans un premier temps sur le site de l'ISEN et celui de l'Institut de Soudure. Les caractéristiques du produit (voir V.4.2) permettent bien sûr l'adaptation à d'autres installations.

Aucun des deux sites expérimentaux n'est équipé à l'heure actuelle de positionneur, les équipements principaux sont donc les robots : un ASEA IRB 6-2 pour l'ISEN, un robot COMMERCY CY800 avec armoire ITMI pour l'Institut de Soudure.

I. Le robot ASEA IRB 6-2.

a. Structure mécanique.

Le robot IRB 6-2 possède 5 degrés de liberté organisés suivant une structure sphérique. La structure mécanique en aluminium est animée par des servomoteurs à courant continu. La charge maximum est 6 kg avec une répétitivité meilleure que 0,2 mm. Le volume de travail du modèle non rallongé est limité comme le montre l'annexe 3.

b. Programmation.

Le robot se programme par apprentissage. Aux fonctions de positionnement s'ajoutent des fonctions de contrôle assez nombreuses qui permettent la réalisation de programmes déjà complexes. Nous citerons parmi les plus importantes :

- ATTENTE sur événement externe ou une durée
- SAUT sur condition externe ou état d'un registre
- APPEL de sous-programme

Ces fonctions peuvent être complétées par un ensemble de commandes gérant des capteurs : les fonctions d'adaptativité. Les capteurs peuvent être digitaux ou analogiques. Un niveau de référence est fixé par apprentissage. On peut :

* rechercher un niveau donné dans une direction donnée (RECH)
* corriger une trajectoire proportionnellement à l'écart entre le niveau de référence et le niveau réel du capteur. La direction de correction est fixée par

apprentissage.

c. Liaison informatique.

Le robot de l'ISEN est équipé d'une liaison informatique permettant le dialogue entre l'armoire du robot et un ordinateur.

La liaison intègre les niveaux 0 à 4 de la norme OSI sur les protocoles de transmission.

Le niveau 0 est un protocole RS232 géré directement par le composant de sérialisation.

Le niveau 1 est un protocole de communication ADLP10. Celui-ci peut être représenté par deux automates déterministes, l'un pour le robot, l'autre pour l'ordinateur.

Le maître est la station qui a été à l'origine de la liaison, l'esclave est l'autre station. Le caractère de contrôle RVI permet l'inversion de cet échange.

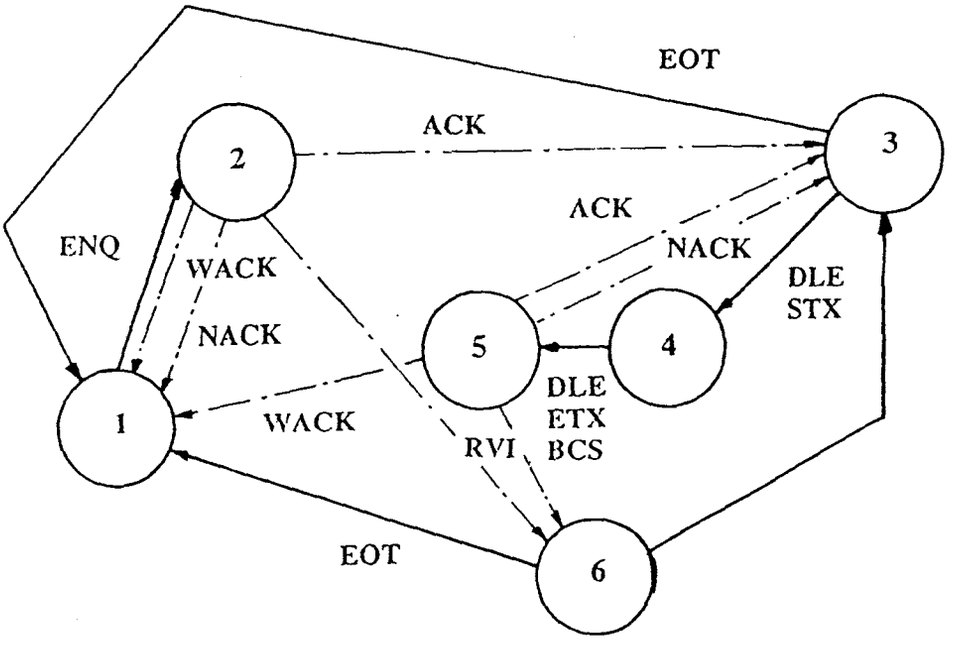


Figure 5.16. Protocole ADLP10

Les niveaux 2 à 4 sont asurés par le protocole d'application ARAP. Celui-ci définit l'entête des messages échangés par la liaison. Les messages sont découpés en blocs. Ils sont de deux types :

* les requêtes à l'initiative du poste de contrôle informatique
* les messages à l'initiative du robot.

La fiabilité des résultats est assurée par le calcul d'un "check-sum" qui permet la validation d'un bloc de données.

Les requêtes permettent d'accéder à toutes les fonctions, à tous les états du robot (voir en annexe la liste des requêtes), principalement elles offrent la possibilité de

manoeuvrer le robot et de transférer des programmes.

Les messages spontanés correspondent à des événements survenus sur le robot (arrêt d'urgence, fin d'une fonction recherche, ...). L'occurence aléatoire de ces messages et la nécessité de les traiter rapidement a, pour des raisons de sécurité, amené lors de l'implantation sous UNIX à faire gérer la liaison par deux processus (un en lecture, l'autre en écriture) qui communiquent par des pipes.

L'ensemble des requêtes est accessible par l'appel de procédures écrites en C qui rendent leur utilisation aisée.

d. Conclusion.

L'ensemble adaptativité, liaison série, donne à cette configuration de l'IRB 6-2 des performances qui sont indispensables pour la programmation hors ligne.

II. Le robot CY800.

Le robot CY800 a été développé par la société COMMERCY qui est spécialisée dans la fabrication et la vente de matériel pour le soudage à l'arc. C'est un robot qui a donc été conçu pour la soudure avec son armoire de base (annexe 1). Il est équipé dans la version dont dispose l'Institut de Soudure d'une armoire ITMI ACOR.

a. La structure mécanique

Les 5 axes du robot CY 800 sont organisés suivant la même structure que le robot IRB 6-2. Seuls quelques choix technologiques distinguent les deux structures. Vis à billes remplacées par des billes sur les axes 2 et 3, transmission par chaîne par les axes 4 et 5 à la place de biellettes). La répétabilité est $\pm 0{,}2$ mm. Son volume de travail est légèrement supérieur à celui de l'IRB-6 non rallongé.

b. Armoire de commande.

L'architecture de l'armoire ACOR est articulée autour d'un HP 1000 (voir Annexe 3). Celui-ci réalise la fonction commande et interface avec les fonctions asservissements par un bus HPIB.

Le système temps réel RTEA permet l'exécution simultanée du développement de programmes en langage LM et leur exécution. Cette seconde tâche est bien entendue prioritaire.

Un boîtier de commande et de télé-opération autorise le pilotage direct du robot et la prise en compte de positions réelles dans les programmes LM.

c. Programmation.

Le langage LM [MAZ84] permet la programmation hors ligne du robot. Le langage LM a été développé de 1979 à 1981 par l'équipe intelligence artificielle du Laboratoire d'Informatique Fondamentale et d'Intellignece Artificielle de Grenoble (LIFIA). C'est un langage de programmation à part entière (possibilité de manipuler, flottant, entier, réel) qui possède en outre des instructions propres à la robotique. Nous reprenons ici les différents points définis au chapitre III (3.2.1.1) pour caractériser les langages.

1. Modélisation et déplacement

La modélisation de l'espace et des relations est réalisée par la définition des repères. Un déplacement est défini comme une succession de repères, les

chemins entre ces repères peuvent être libres, rectilignes, ou circulaires. L'existence de "fichiers trajectoires" permet même de définir des chemins plus complexes.

2. Facilité de programmation

   LM est un langage déclaratif. Il permet de définir le type des données. Il possède d'autre part les instructions permettant une programmation structurée. Les fonctions de manipulation des vecteurs, repères... sont très utiles pour définir un déplacement. L'existence d'un interpréteur interactif permet une mise au point plus facile avant la compilation définitive du programmme.

3. L'interaction avec l'environnement

   L'interaction avec l'environnement est réalisée soit par les fonctions capteurs (Force, Moment, Vision) ou par les entrées-sorties du système.

4. Aptitude au parallélisme

   Implantation LM à partir d'un exécutif temps réel autorise l'exécution parallèle de plusieurs tâches.

5.2.4.2 Les sites informatiques.

Il faut distinguer dans les sites informatiques les sites de développement du logiciel et le matériel cible de l'application. La compatibilité entre les deux sites de développement (HP 9000 du LIMSI robotique, Intel 310 à l'ISEN) et le poste de programmation final est assuré par le choix d'un système d'exploitation commun UNIX (XENIX pour l'ISEN) lié au langage C.

I. Le choix de UNIX et C.

   UNIX est un système d'exploitation développé par la BELL TELEPHONE COMPAGNY dans les années 70. Depuis, diverses licences ont permis son installation sur des matériels variés, des mini-ordinateurs aux micro. On se trouve donc en face d'une gamme étendue de matériels pouvant supporter un logiciel développé sous UNIX. Au point de vue développement de programme, UNIX offre de très nombreux utilitaires d'aides qui se révèlent précieux lors l'étude de la faisabilité, du développement, de la mise au point.

   D'autre part, le couple UNIX-C autorise l'exécution à partir d'un langage évolué des fonctions de base du système UNIX. Particulièrement, cela autorise l'exécution de processus en parallèle (multi-tâche) et la communication entre processus (largement utilisée dans la gestion du protocole IRB 6-2 / Intel 310, voir V.5.1).

   Ces avantages compensent les faiblesses du langage C quant aux contrôles effectués lors de la compilation.

   La portabilité des logiciels développée en C sous UNIX entre les deux sites de développement n'a pas encore été mise en défaut. Cela n'est cependant possible que parce que l'on prend soin de n'utiliser les fonctions des bibliothèques communes aux deux systèmes et que l'on n'utilise pas directement les possibilités de modification des paramètres systèmes offertes par le langage C.

II. Le poste de programmation graphique.

   La configuration du poste de travail qui suit fait l'objet de quelques compromis et impératifs déjà présentés :

Poste de programmation graphique

- contrainte de prix sur l'ensemble du poste de travail
- nécessité d'un graphique haute résolution incluant la couleur pour la lisibilité de l'image
- puissance de calcul nécessaire pour manipuler les objets en trois dimensions tout en gardant des temps de réponse compatibles avec l'aspect intéractif du programme.

Deux options étaient possibles quant aux deux derniers points :

. l'apparition de processeurs graphiques qui incluent des fonctions de modélisation géométrique en trois dimensions autorise une puissance de calcul moindre sur le processeur principal.

. les processeurs 32 bits autorisent en monoposte les puissances de calcul jusqu'ici réservées aux mini-ordinateurs et permettent un processeur graphique plus simple.

Dans les deux cas, la "base" d'accueil est un compatible IBM-PC, car le standard qu'il représente a amené la création de cartes d'extension dans les deux domaines. La solution Bus VME offre aussi des avantages mais nous ne l'avons pas retenue pour de raisons de coût de la version de base.

La contrainte de prix nous a amenés à choisir la seconde solution quant au graphique (50 KF contre 100 KF).

Nous proposons donc comme station de travail :

- une base IBM PC AT III - 30 Mo
- une carte graphique offrant une résolution 1024 x 1024 par 16 couleurs.
- une carte processeur 32 bits 2Mo de mémoire sous UNIX système V.

III. Le logiciel MOVIE.

Le logiciel MOVIE a été développé par Brigham Young University. C'est un logiciel très général de création de modèles tridimensionnels et de visualisation graphique.

Il présente l'avantage d'être portable sur des petits systèmes, une version est même prévue par les micro-ordinateurs Mini MOVIE.

Son aspect modulaire s'intègre très bien dans la conception du poste de travail.

Trois modules sur les 6 que comporte MOVIE seront utilisés :

- UTILITY génère et édite des modèles polyhédriques deux ou trois dimensions. Des modèles de base, très généraux (parallèlépipède, hexaèdres, modèles de révolution...) sont assemblés en parts dans lesquelles ils sont liés de manière rigide.
- DISPLAY assure la visualisation sur terminal graphique. Ce module autorise les modifications de la pespective utilisée pour la projection (ROTA, TRANS) et le déplacement d'une ou plusieurs parts par rapport au repère de référence (PIVO, EXPLO).
- SECTION réalise la découpe d'un objet par des plans ou des surfaces quelconques. L'objet découpé est constitué de parts dont la numérotation dépend de l'orientation de la normale au plan. La supréssion de point redondant et d'arête inutile est réalisée.

Les commandes envoyées aux différents modules sont saisies à partir du terminal ou d'un

fichier de commande. Les possibilités offertes par UNIX de tâches parallèles et de redirection des entrées-sorties permet la communication entre des programmes d'application et MOVIE par l'intermédiaire de pipes.

5.2.5 Conclusion

La présentation des sites expérimentaux montre que par delà la diversité des matériels, des points communs existent. Ils assurent une cohérence entre les travaux pouvant être menés sur chaque site. La concordance au niveau informatique relève du choix liéau rapprochement avec le LIMSI. La similitude des structures des robots évite par ailleurs le dévellopement d'algorithmes d'inversion différents.

5.3 POSTE DE PROGRAMMATION GRAPHIQUE DE CELLULE DE SOUDAGE ROBOTISEE.

Une première réalisation, telle qu'elle a été présentée au congrès MICAD 86 [CAS 86], a servi de base à la définition d'un poste de travail adapté aux besoins des industriels du soudage.

Cette démarche sera reprise dans la présentation qui suit: description et critique de la première application, définition d'un poste de travail précisant les utilisateurs potentiels et qui utilise les acquis des travaux précédants

5.3.1 PREMIERE REALISATION.

Cette maquette a été réalisée en collaboration avec une société de la région lilloise, Electrification Charpente Levage, pour répondre aux besoins de cette entreprise. Celle-ci était confrontée pour l'étude de ses produits (équipements lourds de manipulation automatisée) ou pour l'étude de ses gabarits de soudage à la nécéssite de détecter les interférences possibles au cours de mouvements de solides dans l'espace.

5.3.1.1 Présentation

I. Objectifs.

L'objectif premier était la réalisation d'un outil de simulation de scènes animées tridimensionnelles. Ce logiciel permet la validation de la conception des éléments et de leur séquence d'animation. L'aspect programmation de la soudure est arrivée en seconde instance pour aider l'opérateur dans sa tâche particulière de simulation de la soudure.

II. Le logiciel d'animation.

Ce logiciel a été développé autour de la bibliothèque graphique AGP3 (Advence Graphic Package 3D de Hewlett Packard) en Fortran sur un HP 1000. AGP3 offre toutes les fonctionalités graphiques nécéssaires aux développements de telles applications :

* déplacement autour de la scène ;
* projection en perspective de la scène ;
* agrandissements (ZOOM) ;

* curseur graphique ;
*

Ces opérateurs s'appliquent sur une représentation filaire de la scène à réaliser. Le principal développement a été la mise en place d'une modélisation de cette scène et son animation.

5.3.1.2 La modélisation et l'animation de la scène.

I. Les représentations internes.

Pour permettre une création rapide et précise d'un objet, ainsi que sa modification, nous avons utilisé une représentation proche de l'arbre de construction [GAR 85] dont nous avons gardé l'aspect fonctionnel. Un objet est ainsi représenté par une arborescence d'éléments paramétrés. Chaque élément est relié à son élément de référence par une transformation géométrique de déplacement et un opérateur implicite de sommation.

Ce choix présente l'avantage de fournir un mode de construction de l'objet à souder proche du processus de mécanosoudure (ajout successif d'éléments) mais laisse à l'opérateur la responsabilité de la cohérence de la représentation qui n'est pas assurée implicitement.

Pour éviter les temps de calcul, la représentation par l'arborescence est, après vérification de sa cohérence, convertie en une représentation par les limites. Le modèle "fil de fer" utilisé amène des ambigui'tes d'interprétations de la scène. Certaines sont levées par la connaissance implicite qu'a l'opérateur de la géométrie de la scène, d'autres, particulièrement pour la recherche de collision, par l'utilisation d'un "angle de vue" adapté et par les couleurs attribuées à chaque pièce .

II. L'animation de la scène.

L'animation de la scène nécéssite la décomposition de l'environnement en modules indéformables, articulés entre eux.

Un module est défini par :

1. une arborescence d'éléments paramétrés (représentation de sa géométrie) ;
2. la nature de l'articulation (rotation ou translation) qui le lie à un module de référence et l'axe d'application de cette liaison (éléments de la cinématique) ;

Pour une plus grande commodité de traitement, les modules sont regroupés par pièce (robot, positionneur...). Chaque pièce a une position fixe dans l'environnement et l'objet à souder est l'un des modules du positionneur sur lequel il se trouve. On obtient donc une représentation de l'environnement sous la forme d'un arbre.

Pour réaliser l'animation, le parcours de l'arborescence permet d'obtenir la position de chaque module dans le repère de base défini par un opérateur matriciel.

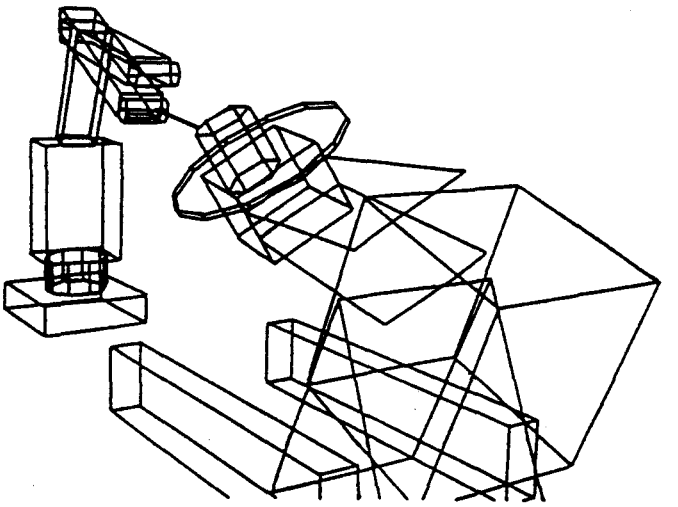


Figure 5.17. Représentation graphique d'un environnement.

III. Simulation de soudure robotisée.

- La soudure à réaliser est définie en référence au modèle de la pièce (niveau objet). Le cordon de soudure est l'un des segments de la représentation polyhédrique choisi par l'opérateur. La torche peut être orientée autour du segment (angle d'inclinaison et d'avance) et son extrémité peut être déplacée sur le segment. Les déplacements, orientation et position, sont incrémentaux et d'incréments ajustables.
- Le passage de la situation de la torche à la configuration du robot nous a amené à développer un logiciel d'inversion spécifique au robot 5 axes que nous utilisons et à la définition d'une tâche de soudure. Cette inversion géométrique a pour principe la recherche de la position du point invariant du poignet à partir de la connaissance de la situation de la torche dans l'espace et de la géométrie de celle- ci. Le cas le plus général d'une torche décalée et inclinée par rapport à l'axe de rotation est présenté en annexe (Annexe 4). La configuration du robot nous permet par comparaison avec les butées articulaires de définir si la situation de la torche est accessible.
- La détection de collision telle qu'elle est définie dans le chapitre III nous a amené à concevoir un fenêtrage dynamique de l'image. Le but de ce fenêtrage est de fournir directement un point de vue à l'utilisateur qui lui permette de lever toute ambigui'te quant à la collision, et qui le décharge de ce fait de la recherche fastidieuse de ce point de vue. En nous inspirant d'une solution utilisée en télé-opération assistée par un système de vision [COI82], nous fixons donc l'extrémité de la torche au centre de l'écran et la normale à l'écran est calculée en référence aux orientations du segment à souder et de la torche.

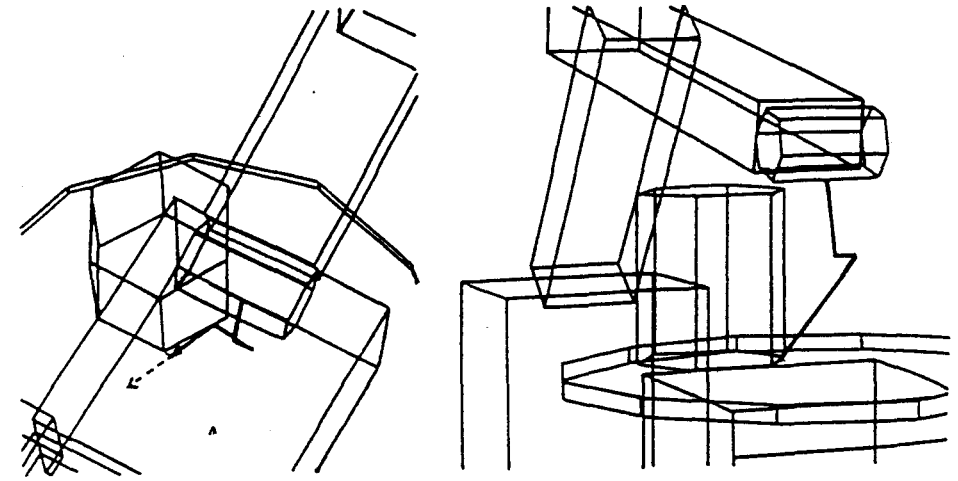


Figure 5.18. Fenêtrage dynamique.

IV. la programmation.

La programmation n'a pas été menée à son terme sur cette implantation. Les points de soudage sont simplement mémorises en coordonnées effectives et la simulation d'une séquence enregistrée est possible.

5.3.1.3 Critique des solutions retenues.

1. Les limites

   La modélisation uniquement hiérarchique d'une application empêche toute souplesse dans la définition d'une nouvelle pièce, d'une nouvelle cellule... En effet, les objets n'existent qu'en référence à une application donnée.

   Le type de modélisation filaire n'autorise pas d'opération sur les solides de type intersection, union ou différence. Ces opérations peuvent pourtant se révéler très utiles pour la définition d'un cordon de soudure (souvent à l'intersection de deux volumes).

   La programmation de la soudure d'une pièce se fait obligatoirement en référence à une application donnée (conséquence de la hiérarchie de la représentation et du mode de mémorisation des points). Ceci ne permet pas le transport d'un programme d'une cellule à une autre. Les mouvements hors référence ne sont pas gérés.

2. Les points positifs

   Nous retiendrons de cette expérience :

   - le fenêtrage dynamique et couleurs différentes pour chaque pièce qui se sont révélés très utiles lors de la validation du logiciel sur une pièce réelle (Annexe 5).
   - la méthode d'inversion du robot est rapide et fournit l'ensemble des solutions possibles, ce qui est intéressant dans un volume encombré. Son extension à d'autres robots implique une bibliothèque de sous-programmes paramétrés.

- l'intérêt de la construction des pièces à partir d'éléments standards [GAR85].
- la souplesse qu'apporte la programmation hors ligne : la connaissance du modèle de la pièce autorise une programmation de la tâche de soudure (paramètres géométriques, vitesse) plutôt que la programmation de points de soudure. Cet élément est important par rapport à l'apprentissage pour lequel la maîtrise de l'orientation dans l'espace n'est pas immédiat.
- le mode de définition de la soudure n'est possible que si la pièce a été modélisée. La saisie d'une pièce est souvent longue et pénible car aucun outil n'a été développé pour améliorer l'interactivité de cette tâche (deux jours et demi de saisie pour la pièce de l'Annexe 5). La durée de cette étape indispensable est englobée dans le temps de programmation et limite l'efficacité du système.

5.3.2 DEFINITION D'UN POSTE DE TRAVAIL.

La conception d'un produit, informatique ou non, part des utilisateurs potentiels. Elle ne doit cependant pas s'enfermer dans une réponse étroite aux demandes de ceux-ci sous peine d'être vite obsolète.

5.3.2.1 Les utilisateurs.

Le public visé est celui des PMI, PME du soudage à l'arc. Les utilisateurs seront donc des professionnels du soudage possédant une bonne expertise du procédé. Il est fort probable d'autre part qu'ils n'aient peu ou pas de connaissances informatiques. L'informatisation de ces entreprises en CAO ou en CFAO est par ailleurs très limitée, la diversité des matériels pouvant être proposés n'autorisant pas le développement de fonctionnalité de soudure robotisée à chacun.

Un poste de travail graphique autonome est l'option que nous avons retenue. Ce poste ne doit cependant pas représenter plus de 10 à 15 % de l'investissement global de la robotisation.

L'accent devra être mis sur la qualité de la communication homme-machine. Cela sera possible grâce à des fonctionnalités

. spécifiques à la soudure (pour la programmation)

. de création de pièces adaptées au type de pièce soudée

. de gestion de données...

. de représentation graphique.

Les fonctions de décision relatives à la programmation seront dans un premier temps laissées à l'expertise de l'opérateur, car il n'existe pas de travaux assez avancés en soudure pour générer automatiquement une séquence de soudage. La détection de collision reste plus simple à décider à l'aide d'une représentation graphique. Cela implique que l'opérateur pourra disposer des éléments lui permettant de choisir (vue pertinente, données géométriques...) et qu'il pourra faire des choix, modifier des paramètres en conséquence.

5.3.2.2 Le produit.

La qualité principale du produit, outre la réponse aux contraintes posées ci-dessus, est la recherche de l'ouverture. Celle-ci peut s'exprimer sous deux formes :

- extensivité du produit.

  Le produit doit pouvoir s'adapter pour accueillir des fonctionnalités nouvelles, pour prendre en compte des configurations matérielles différentes ou d'autres procédés que le soudage (manutention ...). Cet aspect nous amènera à un développement modulaire du produit.

- la portabilité du produit.

  Vu la rapidité d'évolution des matériels informatiques, nous devons assurer l'indépendance la plus complète possible du logiciel par rapport au matériel. Nous avons fait le choix d'un développement en langage de haut niveau en lien avec un système d'exploitation (C et UNIX, voir sites expérimentaux).

5.3.3 MODULES ET FONCTIONNALITES.

Après avoir défini les objectifs du poste de programmation graphique, il est possible de décrire, d'une part, les enchaînements de modules nécessaires à la programmation de la soudure d'une pièce, d'autre part les fonctionnalités auxquelles peut faire appel chaque module.

5.3.3.1 Programmation de la soudure.

La figure ci-dessous montre un enchaînement possible de différents modules. Un module est une unité de programmation cohérente utilisant des données en entrée et fournissant, après traitement, des données en sortie. La décomposition en module autorise l'ajout, le remplacement d'un module sans remettre en cause l'ensemble de l'application.

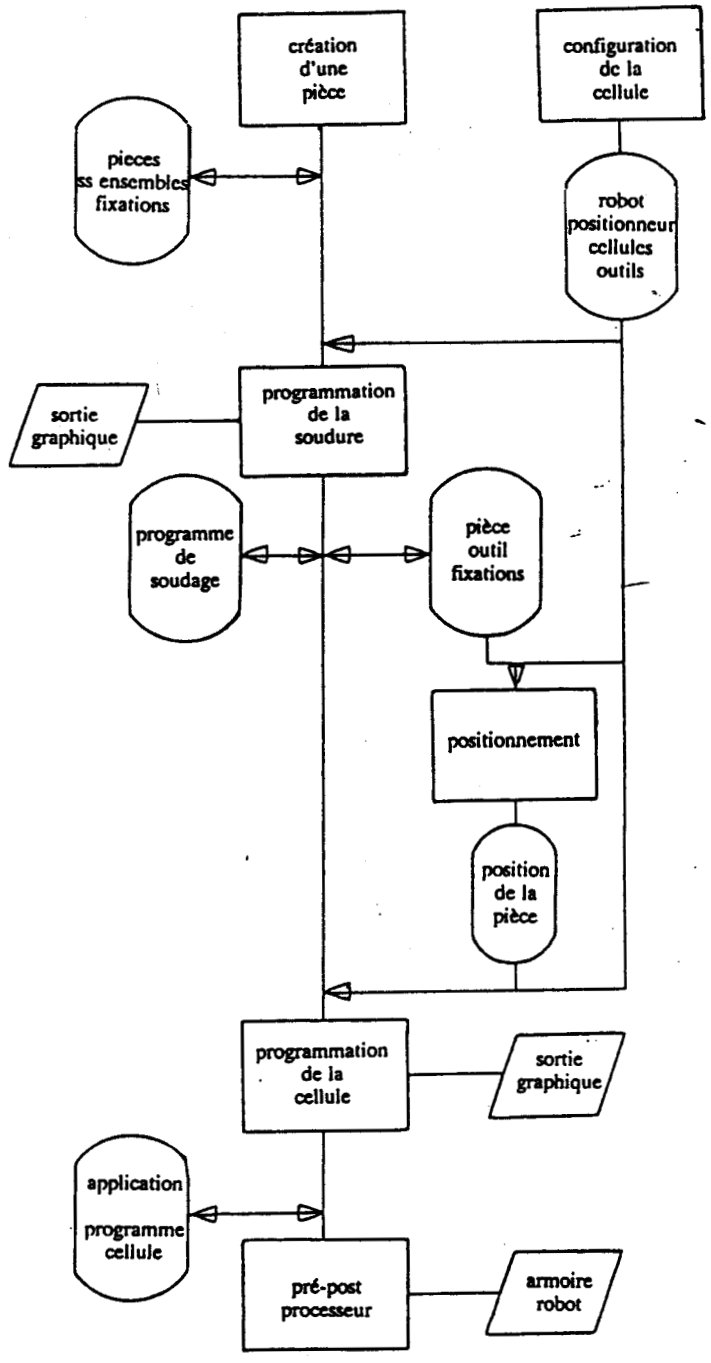


Figure 5.19. Enchainement des modules.

La programmation s'effectue en deux étapes. Une étape de définition de la trajectoire dans le repère de la pièce (niveau objet) puis la validation dans un environnement donné (cellule de soudage) du programme avec le calcul des "programmes" des différents éléments de la chaîne cinématique (robot(s), positionneur...) au niveau effecteur. Entre ces deux étapes, un module de positionnement de la pièce peut avoir assuré la conformité de la position théorique avec la position réelle de la pièce (voir Chapitre IV). Un certain nombre de données sont nécessaires à chaque étape :

- géométrie de la pièce, de l'outil pour la première phase de programmation.
- constitution de la cellule, position de la pièce pour la deuxième phase.

D'autre part, un post-processeur se charge de convertir les programmes en coordonnées effecteur ou articulaires en instructions compatibles avec l'armoire du robot choisi.

5.3.3.2 Les modules.

Chaque module constitue un traitement particulier sur les données qui aboutissent à la programmation de la soudure. Il nécessite donc des informations en entrée et fournit des données en sortie.

I. Configuration de la cellule.

- Ce module va permettre la constitution de la cellule de soudage. Il est intéractif bien que son utilisation soit peu fréquente.
- La cellule est constituée d'un environnement fixe, de manipulateurs (robot, positionneur) qu'il faut placer dans l'espace. Les éléments de la cellule sont désignés par leur nom et l'opérateur peut choisir parmi les éléments existants.
- Les éléments sont saisis par l'opérateur sur la base d'une structure de données préfixée.
- Ce module permet d'enrichir la base d'éléments d'une nouvelle cellule ou d'un nouveau robot.

II. Création d'une pièce.

L'opérateur de saisie d'une nouvelle pièce doit être la plus simple possible dans la mesure où le temps de saisie doit être compté dans le temps de programmation global de la soudure[SOL84]. Les données de base sont des modules élémentaires qui sont assemblés pour former la pièce. La création d'une pièce peut faire appel à des sous-ensembles déjà définis. Ce module fournit la représentation de la pièce dans un format compatible avec la visualisation par MOVIE et l'arbre de construction de la pièce en vue de modifications éventuelles.

III. Programmation de la soudure.

A partir de la géométrie de la pièce, la définition d'un programme de soudure revient à placer le repère de la torche en référence au repère pièce. Une torche, choisie parmi le catalogue d'outils, ainsi qu'éventuellement des éléments de fixation, sont d'abord associés au programme. L'ensemble pièce, torche, fixations constitue avec les points programmés le programme de soudage.

a. Programme de soudage.

Un point programme contient les informations de position et d'orientation de la torche dans le repère pièce. Il mémorise d'autre part des informations qui devront être envoyées à la source de soudage et la vitesse d'exécution du mouvement. Les paramètres programmés sont les suivants, ils sont définis dans le chapitre 5.2.1 (figure 5.2) :

- distance torche-pièce
- angle d'avance

- angle d'inclinaison

Leur référence est un segment de la pièce assimilé au cordon de soudure. La définition de l'angle d'inclinaison impose la connaissance des faces qui délimitent le cordon de soudure. Nous devons donc substituer pour ce traitement à la représentation face/sommet F(S) de MOVIE une organisation Arete/Face A(F). On passe donc de la représentation de la figure 5.3(a) à celle de la figure 5.3(b). La numérotation des arêtes se fait par adressage calculé sur les numéros des points constituant l'arête.

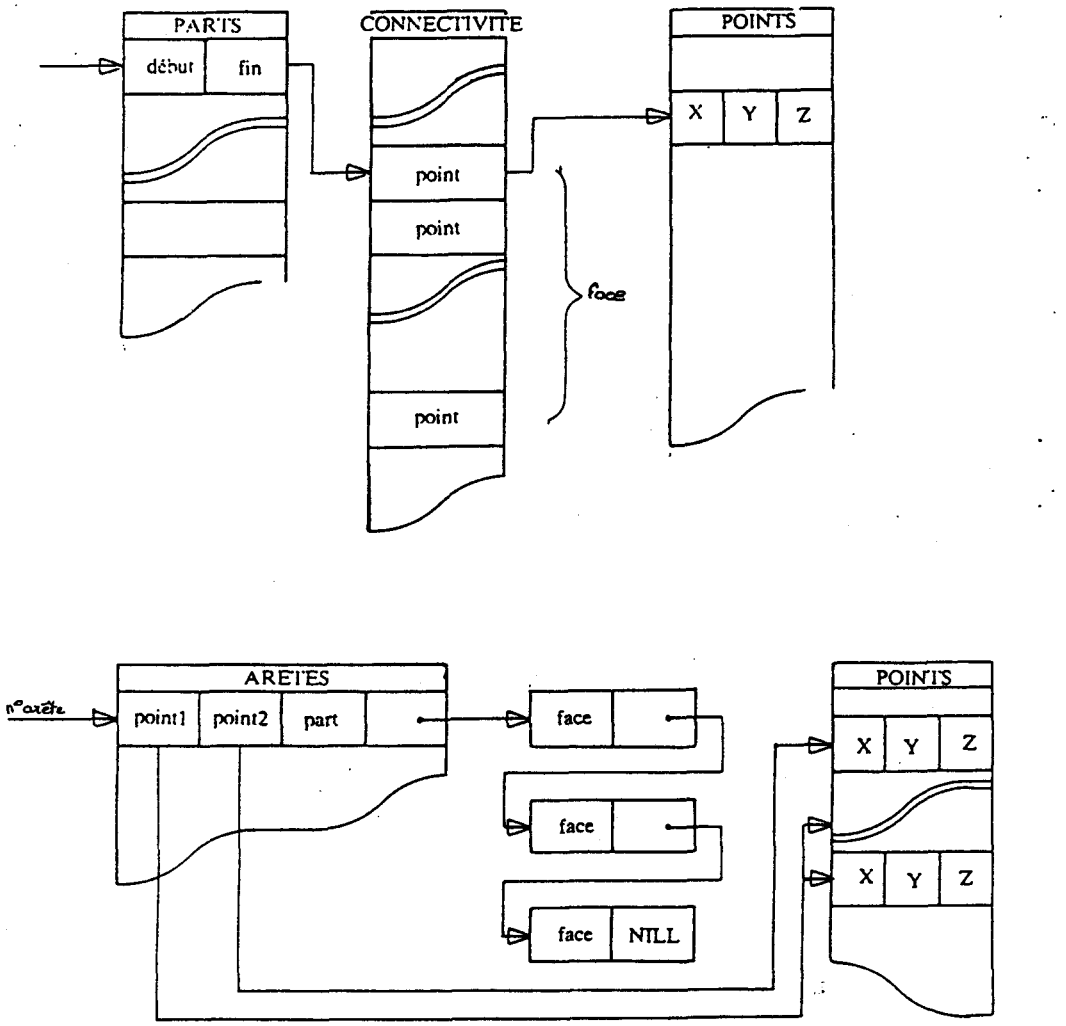


Figure 5.20. Représentations des objets

Pour calculer l'orientation du vecteur torche dans le repère de la pièce, on définit d'abord le repère cordon défini par :

* le vecteur cordon **c** assimilé au segment (axe **x**)
* le vecteur **v** (axe **z**) produit vectoriel de **c** avec **n** la plus horizontale des normales aux faces définissant le cordon.
* le vecteur torche est l'axe **z** du repère défini ci-dessus après une rotation ϕ/x et θ/y. L'origine du repère peut se déplacer sur le cordon suivant un pas ajustable ∂x.

b. Points de dégagement

La soudure d'une pièce nécessite la réalisation de cordon à divers endroits de la pièce. Des mouvements d'approche, de dégagement, et de parcours libre sont nécessaires. Ces mouvements hors référence de cordon de soudure peuvent se faire soit dans le repère de base de la pièce, soit parallèlement au vecteur torche. Dans ce cas, un déplacement $\partial \mathbf{v}$ incrémental de valeur ∂x sera calculé, suivant l'axe choisi **a** où **a**=**x**,**y**, ou **z**, par $\partial \mathbf{v} = \partial x\ \mathbf{R}\ \mathbf{a}$. **R** est la matrice de rotation donnant l'orientation de la torche dans le repère de base.

c. Edition du programme.

L'ensemble des points associés à des paramètres annexes (vitesse, soudage,...) est chaîné par un chaînage avant et arrière pour faciliter :

* l'édition
* la modification
* la simulation de programme

Les instructions de contrôle ne sont pas prévues dans le programme de soudure. L'ensemble des déplacements de la torche est visualisé au fur et à mesure de leur commande. La saisie d'un point est réalisée en validant la position affichée. Un retour arrière d'un point autorise une reprise en cas d'erreur. Une fois le programme terminé, il est possible d'en simuler l'exécution, de modifier, d'ajouter des points à l'aide d'un éditeur spécialisé.

Le module programmation fournit donc à partir de la définition d'une pièce, une suite de positions de la torche constituant le programme.

IV. Positionnement

Nous avons vu dans le chapitre IV qu'un programme défini en référence objet imposait de définir avec précision la situation de la pièce dans l'espace. Pour la simulation, nous avons besoin d'une situation théorique proche de la situation réelle. Elle sera demandée à l'opérateur.

Cette position théorique servira de base à l'algorithme de positionnement présenté au chapitre IV. La définition du processus de palpage est intéractive par la définition des faces à palper, d'une part, la simulation du palpage pour éviter les collisions,d'autre part.

Ce module définit l'application. Elle comprend la cellule qui va réaliser le soudage et la position de la pièce dans cette cellule.

V. Programmation de la cellule.

Ce module constitue, à partir de la description de la cellule dans la base de données et des élements contenus dans le programme de soudure,la représentation graphique qui va être animée.

Sa fonctionnalité principale est la validation du programme réalisé par le module " programmation de la soudure " par rapport au site réel. L'opérateur a toute latitude pour se déplacer dans le programme et pour demander la visualisation de l'exécution d'un point particulier ou de toute une séquence. Il peut ainsi vérifier si le point de soudure est accessible au robot. Dans le cas contraire le logiciel d'inversion indique le premier axe en butée articulaire. La visualisation permêt de rechercher les collisions éventuelles.

Poste de programmation graphique

Les axes externes aux robots peuvent être commandés individuellement. Chaque point du programme source est transformé en un point du programme cellule qui indique la situation de l'extrémité du robot et les coordonées articulaires des axes externes.

La simulation peut ensuite se faire en continu ou en pas par pas sur la base du programme cellule. Toute modification de ce programme est traduite immédiatement dans le programme de soudure.

VI. Pré et post processeurs.

L'adaptation du poste de programmation aux robots industriels impose que les programmes qu'ils fournissent soient traités pour les adapter au format du simulateur. Dans l'autre sens de la transmission, les positions issues du poste de programmation doivent être transcrites pour former un programme exécutable par l'armoire du robot.

A chaque type d'armoire va donc être associé un pré et post processeur de traitement des programmes. Ces informations sont transmises ou reçues par un protocole spécialisé.

Il y aura donc un module pré-post processeur par robot contenu dans la base de données robot.

5.3.3.3 Fonctionnalités.

Chaque module (mis à part le traitement spécifique qu'il doit réaliser) met en oeuvre des fonctions qui sont communes à plusieurs modules. Pour rendre systèmatique le traitement de ces fonctions, des bibliothèques fonctionnelles gèrent les opérations spécifiques à chacune d'elles.

I. Dialogue.

La notion de dialogue dans un programme intéractif nécessite la visualisation, le choix, la saisie d'information. Comme le présente très bien le chapitre IV de "Fundamental of interactive computer graphics" [FOL84], le traitement de ces différentes fonctions doivent être systèmatiques afin de rendre l'apprentissage plus facile et l'utilisation moins fatigante.[FOL84] définit par ailleurs trois niveaux de retour d'information qui doivent être donnés à l'opérateur :

- le niveau lexical retourne immédiatement l'effet d'une action primaire. Par exemple : affichage d'une touche tapée au clavier, déplacement d'un curseur graphique.
- le niveau syntaxique indique à l'opérateur le résultat d'une sélection (touche fonction, objet...) ou la nature d'une commande.
- le niveau sémantique présente la compréhension par le système d'une succession de commandes et donne un compte rendu de leur exécution.

Chaque niveau doit pouvoir faire l'objet d'une correction et d'un retour à l'état précédant l'action engagée.

La fonctionnalité dialogue permet donc de décrire de manière systématique un module ou une procédure comme un graphe. Les noeuds de graphe représentent un ensemble de choix possibles à l'utilisateur (un menu). A chaque choix est associé :

* un texte de présentation du choix
* le noeud suivant
* une saisie

* une action

La saisie est décrite dans un langage simplifié.

```
<saisie>    :=   <nombre> {<texte><type>}
<texte>     :=   {<lettre>}
<lettre>    :=   [ASCII]
<type>      :=   /<lettre>/<borne>/<borne>/
<borne>     :=   <entier> // <flottant> // <lettre>
<nombre>    :=   <entier>
```

Le type renvoie à une procédure de saisie ,soit standard (acquisition d'entier, de flotant....) ,soit spécialisée en fonction de l'application.

Le contrôle du niveau lexical et syntaxique est assuré par le menu, et les procédures de saisie. Le niveau sémantique (retour d'information et correction) est assuré par l'application elle-même.

L'action est définie par un nom qui correspond à une procédure à exécuter. Les transferts de données (saisie, résultat d'action...) se font par l'intermédiaire d'une pile de données. Des utilitaires permettent aux programmes d'application d'utiliser cette pile.

Le graphe et les saisies associées sont décrits dans des fichiers au nom du module généré.

La visualisation est constituée par une bibliothèque de procédures spécialisées qui dépendent du type d'affichage souhaité et du terminal. On peut, si l'on dispose du terminal adéquat, utiliser un logiciel de multi-fenêtrage ou se contenter d'un menu de type "touche fonction" sur un terminal normal.

L'aspect systématique de l'enchaînement des commandes, des saisies et des contrôles associés (niveau lexical et syntaxique) offre l'avantage de décharger l'application de ses tâches, d'offrir à l'utilisateur un environnement de dialogue unique et donc d'en faciliter l'utilisation et surtout l'apprentissage.

II. Gestion des données.

La spécificité des données liées à la robotique vient de la profondeur hiérarchique des liaisons entre les entités manipulées. La description d'une application complète (cellule, outil, pièce, programme...) fait appel à des entités elles-mêmes composées d'autres entités (cellule, pièce) et pouvant comporter de nombreux éléments (pièces, programme).

La fonctionnalité de gestion de données décrit donc les données sous la forme d'une base de donnée hiérarchique (en réseau). La base est décrite sous la forme d'un langage.

```
<Entite>    :=   <saisie> // <Entite> // [<Entite>]
<base>      :=   <nom> <entite>
<saisie>    :=   /<type>
<type>      :=   <lettre>/<borne>/<borne>/
<borne>     :=   <entier> // <Flotant>
<lettre>    :=   [ASCII]
<nom>       :=   10{<lettre>}
```

La description est stockée dans un fichier. A chaque base correspond une directorie qui contient la base. La base est décomposée en un fichier entête et des fichiers contenus. Le fichier entête contient les informations permettant la gestion des données, les fichiers contenus sont les fichiers de données spécialisées (géométrique, cinématique, programme...)accessible à partir du nom de l'entité sélectionnée.

Les opérations classiques de bases de données sont possibles.

- créer un enregistrement
- sélectionner un enregistrement
- modifier un enregistrement
- détruire un enregistrement.

La sélection peut être multi-critères. Il est aussi possible de décrire une base, de lister une base.

La description de la base de données associée au poste de travail est présentée en annexe accompagnée d'un exemple de manipulation sur une base simplifiée.

III. Modélisation solide.

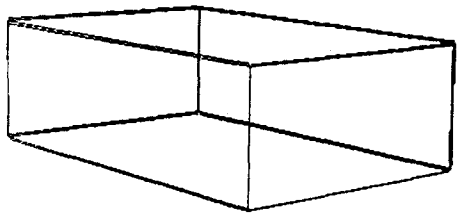
Nous avons vu que l'un des objectifs de la simulation graphique est la validation de la trajectoire quant aux collisions.

Le but de la soudure à l'arc est l'assemblage de volumes. Ces volumes sont soudés à leur intersection. Les intersections pouvant être complexes, il est important d'offrir à l'utilisateur la définition automatique de ces intersections.

Ces deux objectifs imposent que la représentation des objets, principalement de la pièce à souder, fasse l'objet d'une modélisation volumique.

1. Opération sur les solides.

Le logiciel MOVIE que nous utilisons met en oeuvre une modélisation solide par les limites. Une pièce est modélisée par son enveloppe polyhédrique, stockée sous forme de faces polygonales orientées. La représentation de ces faces est une liste orientée de sommets (face-sommet). se terminant par un numéro de sommet négatif. La figure ci-dessous montre la visualisation d'un objet et sa représentation.




```
                1   8   6   24
part            1   6
points  { 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 1.00000E+02 0.00000E+00
          2.00000E+02 0.00000E+00 0.00000E+00 2.00000E+02 1.00000E+02 0.00000E+00
          0.00000E+00 0.00000E+00 3.00000E+02 0.00000E+00 1.00000E+02 3.00000E+02
          2.00000E+02 0.00000E+00 3.00000E+02 2.00000E+02 1.00000E+02 3.00000E+02
faces   {  1   2   4  -3   6   5   7  -8   5   1   3  -7   2   6   8  -4
           5   6   2  -1   3   4   8  -7
```

Figure 5.21. Représentation d'un objet

MOVIE permet de décomposer un objet en plusieurs sous ensembles rigides. Chaque sous-ensemble peut être déplacé, visualisé ou pas... Nous verrons dans animation l'intérêt de ces fonctionnalités.

L'une des limites de MOVIE est l'absence d'opération sur les solides. Deux solides peuvent s'inter-pénétrer sans qu'il y ait de possibilité d'obtenir leur intersection. Il est donc nécéssaire de compléter les possibilités de MOVIE en autorisant des popération sur les solides.

a. Les opérations de base.

Les opérations de base qui peuvent être réalisées sont :

- l'intersection de deux solides
- la différence de deux solides
- l'union de deux solides.

Ces opérations doivent être consolidées, c'est-à-dire qu'une opération sur deux solides doit générer un solide [GAR85]. Ceci n'est pas toujours la cas, comme le montre la figure.

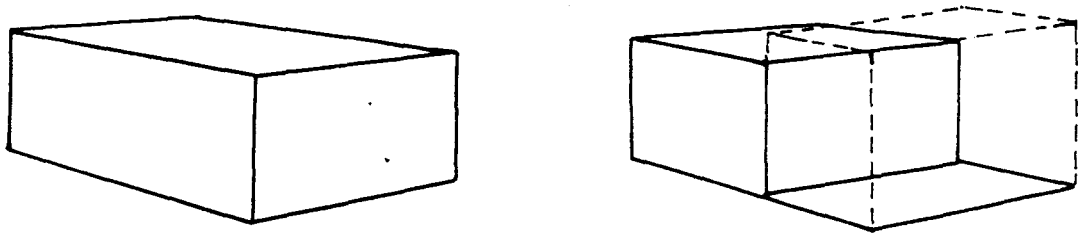


Figure 5.22. Différence non consolidée de deux solides.

b. Les opérateurs réalisés.

Si MOVIE n'offre pas d'opération sur les solides, il permet cependant de découper un solide en deux suivant un plan (CLIPPING)(Voir SECTION dans la présentation de MOVIE).

Il est possible, à partir de cette fonction, de réaliser les opérations de base recherchées. Il nous faut d'abord réaliser des opérateurs de base.

- découpe S1 S2 : génère les plans des faces de S2 et découpe S1 par ces plans. Découpe est l'opérateur de base, l'objet obtenu est décomposé en parts. La part 1 est toujours, si elle existe, l'intersection de 1 dans 2. Après chaque découpe, une procédure supplémentaire doit valider la découpe pour d'une part éviter les découpes inutiles résultant de face qui

n'est pas en intersection réelle avec l'objet, d'autre part détecter en face finale si les objets sont disjoints.

Cela nécessite la mise en oeuvre :

. d'un algorithme pour détecter l'appartenance ou non d'un point à une face.

. le test d'appartenance de tous les points générés par la découpe à la face génératrice du plan de découpe. Si aucun point n'appartient à la phase, l'opération de découpe est annulée.

A la fin de la découpe, l'objet contient des segments inutiles, et donc des faces inutiles car elles sont dans un même plan (figure). Il faut donc reconstituer les faces de l'objet en tenant compte des points générés par le clipping.

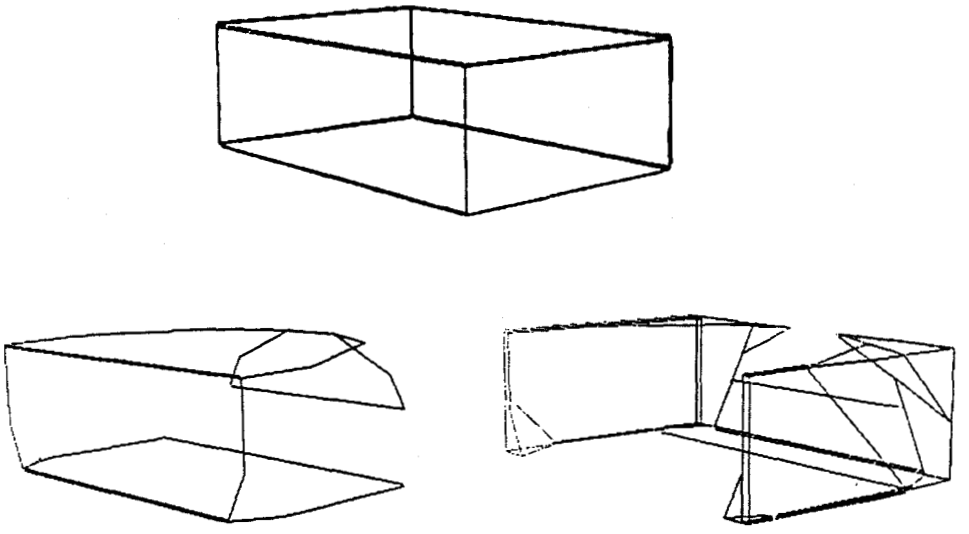


Figure 5.23. Pièce après clipping

- éclate S1: permet d'obtenir certaines parts de S1. La part 1 donne l'intersection réduite, les autres la différence réduite. Il faut parler ici de "solides" réduits car on obtient des enveloppes non closes qui ne sont pas des solides.

- invert S1 : change l'orientation des faces de S.

- union S1 S2 : réalise l'union de deux solides et élimine les points éventuellement en plusieurs exemplaires.

c. Utilisation.

A partir de ces opérateurs, il est possible de réaliser les opérations voulues.

Soit deux solides S1 et S2, on réalise :

découpe S1 S2	→	S3
éclate S3	→	int1
	→	diff1

invert int1	→	invert1
découpe S1 int	→	S4
éclate S4	→	int2
	→	diff2
invert int2	→	invert2

On réalise :

UNIR S1 S2	par	union diff1 diff2
INTER S1 S2	par	union int1 int2
DIFF S1 S2	par	union diff1 invert2
DIFF S2 S1	par	union diff2 invert1

L'annexe 6 montre la séquence de réalisation de cet ensemble d'opération sur deux objets simples.

2. Construction d'un solide.

Un solide complexe est souvent l'assemblage de formes simples liées par des opérations sur les solides. Cette notion d'assemblage introduit une structure hiérarchique de représentation des solides par une décomposition en modules élémentaires. Une première approche situe les modules élémentaires par rapport à un repère de référence. Ceci donne la structure arborescente suivante.

Repère de base

module 1 module 2 module 3 ....

Chaque arc de l'arbre représentent la transformation géométrique qui définit la position du module par rapport au repère de base. Cette représentation a pour inconvénient principal qu'elle ne garde pas trace d'un éventuel mode de construction de la pièce. On ne peut indiquer par exemple que le déplacement d'un module entraine le déplacement d'autres modules puisqu'il n'y a pas d'indication de lien entre modules.

Une deuxième étape consisite à décrire les relations entre objets par une arborescance plus profonde les liaisons.

Repère de base
module 1 module 4

module 2 module 3

module 5

Les transformations entre modules sont fixes ce qui interdit toute modification simple d'un objet solide. Par exemple, si le module 2 est "posé" sur le module 1. La modification d'une dimension du module 2 ne change en rien la position du module 1, la cohérence du solide n'est plus respectée. L'expérience des différentes maquettes montre par ailleurs que la composante orientation de la situation absolue d'un objet dans l'espace est difficile à définir.

Pour tenir compte de ces deux contraintes, nous allons définir des opérateurs de placement relatif de deux objets liés à une structure de représentation d'un solide.

Ces opérateurs définissent la manière de calculer la transformation entre deux objets à partir de la nature et des dimensions de l'objet de base. On va donc définir qu'un module peut être:

- posé ou imbriqué: le premier opérateur indique que l'on aura une liaison plan sur plan entre deux faces; pour le second, une face du module de base sert de référence aux calculs de la position.
- sur ou contre: permêt le choix de la face du module de base qui sert de référence;
- face ou axe: indique l'élément de référence du module à placer;
- centré ou décentré: centrè fait coîncider les centres des deux faces dans le cas d'un opérateur "posé" ou fait passer l'axe par le centre de la face de référence.

  décentré décale ce point de référence dans le plan de la face.

L' ordre de construction de l'objet est mémorisé avec les opérateur de positionnement et d'opération sur les solides.

IV. Visualisation.

Le but de cette fonctionnalité est de permettre le contrôle visuel de l'exécution d'actions commandées. Ce contrôle "sémantique" n'est pas indispensable mais représente un apport sérieux qui permet de ne pas travailler en "aveugle" sur toutes les commandes incluant la manipulation d'objets dans l'espace.

- position d'un module pour la création d'un objet
- position de la torche en programmation
- configuration de la cellule en simulation d'exécution de programme
-

Elle est d'autre part indispensable pour la détection de collision.

1. Visualisation

   Le logiciel MOVIE offre dans le module DISPLAY toutes les possibilités de représentation en perspective d'une scène en trois dimensions (voir MOVIE dans V.5). Nous n'utilisons pas pour des raisons de matériels et de temps d'exécution les possibilités d'ombrage (shading). La visualisation avec faces cachées (VIEW) est accessible à l'opérateur à la demande . La visualisation de type "fil de fer" (DRAW) est le type de visualisation standard à cause de son temps d'exécution réduit.

2. Curseur graphique

   Pour maintenir l'aspect intéractif du programme dialogue, la visualisation doit comporter une partie autorisant les requêtes au système graphique. MOVIE autorise une numérotation des points qui peut être un premier niveau de dialogue. Une fonction "curseur graphique" devra retourner l'orientation de la normale à l'écran et la position du curseur dans le plan de l'écran. Cette fonction permet de retrouver le point de la pièce pointé par l'opérateur par la recherche de la distance minimum entre la droite définie par le curseur et les points.

3. Fenêtrage dynamique

Le contrôle par l'opérateur des paramètres de visualisation (Zoom, orientation de la normale à l'écran, centre de l'écran....) est une tâche annexe qui interfère avec la tâche principale de création d'objet ou de programmation.

De plus, la notion d'orientation dans l'espace et donc la référence à un repère fixe à partir d'une image n'est pas facile à réaliser. Ce problème s'aparente à celui rencontré en télé-manipulation assistée par vision [COI82], où il a été montré qu'à partir de la connaissance de la tâche à effectuer (en télé-opération centre de l'action), il était possible de piloter automatiquement l'axe optique de la caméra pour que l'image soit toujours centrée sur la tâche. Pour éviter l'effet désagréable d'un changement trop fréquent du point de vue les modifications de l'orientation et de la position du plan de l'écran ne sont réalisées que lorsque ces deux paramètres varient de valeurs supérieures à des limites prédéfinies.

En simulation graphique, la "caméra" n'est plus fixée. Nous pouvons donc déplacer l'observateur fictif autour de la scène à l'aide de deux paramètres, la normale au plan de l'écran, et la position du centre de l'écran défini par un point de l'espace et la distance à l'écran . La normale à l'écran donne l'orientation de la perspective et nous avons vu que cet élément est déterminant pour la détection de la collision. L'image est centrée sur l'extrémité de la torche.

Le fenêtrage dynamique propose donc deux directions pour la normale :

- Une combinaison linéaire du vecteur torche et du vecteur représentation le segment à souder.
- Le produit vectoriel du vecteur précédent avec le vecteur torche.

Les deux images appartiennent donc à des plans perpendiculaires et la détection de collision peut se faire dans l'un ou l'autre plan. S'il y a ambigui'té, un léger de déplacement peut être obtenu par modification des paramètres de la combinaison linéaire.

Le fenêtrage dynamique, associé aux couleurs liées aux différents éléments de la cellule (robot, pièce, positionneur...), offre une souplesse d'utilisation qui permet à l'opérateur de se concentrer sur les opérations de programmation et de contrôle (voir utilisation dans annexes 5 et 7).

V. Animation

Pour simuler le fonctionnement de la cellule, il est nécessaire de pouvoir déplacer les objets dans l'espace et de les représenter. Nous avons vu que MOVIE propose une décomposition d'un fichier géométrique en "parts" indéformables. Chaque part peut être déplacée en fournissant sa position et son orientation dans le repère de référence de la scène.

Le premier objectif du logiciel d'animation est donc de fournir, en vue de leur représentation graphique, les situations de chaque solide dans l'espace pour l'exécution d'une tâche donnée.

1. Cas d'une chaîne cinématique

Nous avons vu au chapitre IV que la réalisation d'une tâche d'un point de vue cinématique se ramenait à la résolution d'une chaîne cinématique. Cette résolution

part de la relation T_i EQ T_j T_b. T_b étant connue, la résolution nécessite qu'un et un seul élément de la chaîne soit inconnu. Son calcul permet d'assurer la réalisation de la condition EQ de fermeture de la chaîne.

Le cas le plus classique est à partir de la connaissance de T_b (tâche) dans le repère de la pièce et de la position de la pièce dans l'espace, la détermination de la situation de l'extrémité du robot (ou de l'outil porté par le robot).

Mais il peut être utile, lors de la conception de la cellule, de calculer la position relative du robot et du positionneur pour que la tâche à réaliser soit dans le volume de travail du robot, ou de déterminer la transformation outil permettant d'atteindre telle ou telle situation lorsque le robot est proche de ses butées. Ces différents cas amènent à généraliser la résolution de la cinématique en permettant de choisir la transformation à calculer.

2. Cas de plusieurs chaînes cinématiques

   Une cellule peut comporter plusieurs robots ou plusieurs positionneurs intervenant en même temps ou séquentiellement pour l'exécution de la tâche. Nous devrons donc prévoir que chaque tâche puisse être liée à une chaîne cinématique donnée. L'animation assure la résolution de chaque chaîne à chaque instant. La synchronisation des tâches devra être réalisée au moment de la programmation ou par un module de séquencement de l'ensemble. Ce module n'est pas prévu dans la première version du programme.

3. Traitement des éléments articulés.

   La résolution des chaînes cinématiques ne fournit, comme nous venons de le voir, que la transformation globale des éléments articulés. A chacun de ces éléments doivent donc être associées des fonctions paramétrées qui autorisent d'une part le calcul de la situation de chaque solide pour une transformation globale donnée (inversion), et d'autre part la modification intéractive d'une coordonnée articulaire.

   Le calcul d'inversion d'un robot sphérique 5 axes plan est présenté en annexe 4. Il fournit les coordonnées articulaires du robot à partir desquelles peuvent être calculées les situations de chaque élément du robot. $S_i = \prod_{i} T_{f_j} T_{v_j}$ avec

   . S_i situation de l'élément i

   . T_{f_j} transformation fixe de l'élément j donnée par le modèle géométrique du robot.

   . T_{v_j} transformation variable entre j et j + 1, calculée à partir du type de transformation et de la valeur de la variable articulaire.

   Les transformations fixes sont définies de telle façon que l'axe **z** soit toujours l'axe de référence de la transformation variable (rotation ou translation).

5.4 CONCLUSION.

Dans un récent article, Michel CARRARD, de l'IUT de Cachan, faisait le point sur la CAO-DAO pour les PMI-PME. Il s'y dégage une idée forte : l'outil pour les PME "doit être parfaitement ciblé", c'est-à-dire "spécialisé". Le poste de programmation graphique, par sa

spécialisation à la soudure, rentre tout à fait dans cette démarche, et s'il conseille de se limiter à des petits systèmes traitant le 2D, force est de constater que la soudure à l'arc, donc la spécialisation, impose le travail en 3 dimensions.

S'il reste quelques développements nécessaires dans le domaine de l'ouverture (interfaces pour systèmes de CAO,...), les fonctionalités que nous venons de présenter en font un outil qui devrait correspondre aux nécessités de la programmation de la soudure à l'arc robotisée. En effet, par rapport à des systèmes plus généraux, le poste de programmation graphique tire beaucoup d'informations de la connaissance du modèle de la pièce et des contraintes géométriques du procédé. Pour la programmation en évitant l'aspect fastidieux de la définition d'une attitude dans l'espace, par le calcul automatique de l'attitude de la torche à partir du choix d'un segment. La création de la pièce est déjà un élément de la programmation.

Pour la visualisation, en fournissant une projection de la scène qui limite les recherches, pour la recherche de la position de la pièce, les connaissances des faces et de la position théorique doivent être exploitées au maximum. Cette utilisation des données n'est possible que grâce à la spécialisation.

* le choix que nous avons fait pour la visualisation impose un outil de révolution.
* l'orientation de cet outil n'est possible que par la définition des angles ϕ et θ.

Il a fallu cependant, à ces apports du point de vue communication homme- machine, ajouter des fonctionalités plus classiques sans lesquels un système ne saurait être convivial, la gestion des informations, et le dialogue interactif.

Ayant l'ambition limitée de répondre à des besoins spécifiques d'un secteur de l'industrie, le travail qui vient d'être présenté a des limites quant aux fonctionalités proposées ou à la configuration matérielle, bien que le résultat doive être celui escompté. La décomposition de la programmation en deux étapes permet dans la première de s'attacher au procédé avec moins de données à traiter, dans la seconde de tester la robotisation.

La limitation d'un nombre du type de robots disponibles permet le développement d'algorithmes spécifiques rapides. Les divers travaux, présentés pour situer le produit, suscitent le désir de quelques extensions qui en augmenteraient les possibilités. Celles ci sont abordées dans la conclusion.

6. CONCLUSION.

Les grandes lignes de la robotique de troisième génération et son application dans le domaine de la programmation ont été rapidement brossées dans le chapitre II et III. Le travail présenté dans les deux chapitres suivants n'a pas la prétention de couvrir tous les domaines présentés. Aussi est-il utile de faire le point sur les apports de la démarche proposée, d'une part sur les limites, d'autre part sur les extensions souhaitables.

6.1 APPORT ET LIMITES.

Situer le poste de programmation graphique de cellule de soudage robotisé (P.G.C.S) dans le domaine de la programmation de site robotisé, amène à proposer la notion d'apprentissage hors ligne.

6.1.1 L'apprentissage hors ligne.

Le but recherché à travers cette notion est de fournir à la fois les avantages de l'apprentissage et de la programmation hors ligne. De l'apprentissage, nous avons retenu l'aspect validation immédiate des positions et mouvements programmés, de la programmation hors ligne, l'utilisation de modèles, la libération maximum du site de production pour la programmation.

6.1.1.1 Une validation immédiate.

Cet aspect de l'apprentissage hors ligne utilise pleinement les ressources graphiques du poste de travail. La principale validation concerne les collisions, torche-pièce pour la définition des trajectoires de soudage, robot-environnement dans la phase de simulation de la cellule.

Dans les deux cas, la communication homme-machine est améliorée par le fenêtrage dynamique qui produit une visualisation proche du comportement d'un opérateur lors de la programmation sur site.

La simulation de la cellule assure aussi, par l'algorithme d'inversion du robot que les positions programmées se situent dans le volume de travail du robot.

Une des limites de cette validation est son aspect purement géométrique, puisque les aspects temporels des trajectoires ne sont absolument pas traités (voir 3.3.2).

L'aspect validation et optimisation par contrôle des couples ne concernait guère les configurations expérimentales sur lesquelles nous travaillons, car les commandes n'autorisent pas le contrôle des couples aux articulations.

L'absence de coordination temporelle des mouvements, et éventuellement de simulation temporelle de ceux-ci est beaucoup plus gênante car elle limite la programmation au robot au lieu de l'appliquer à toute la cellule.

La structure des données et l'organisation des modules de programmation et de simulation peut permettre de lever cette limitation.

6.1.1.2 Programmation.

L'objectif de la programmation par simulation graphique n'est pas de proposer un nouveau langage ou même une bibliothèque de contrôle de robot. Il fournit à ces outils existants, langage ou armoire de commande les informations validées qui sont le plus difficiles à déterminer hors ligne, à savoir les situations dans l'espace des repères liés aux points programmés.

Ces informations peuvent ensuite être traitées soit par un langage supporté par l'armoire de commande (LM par exemple), soit par des instructions de contrôle. L'interface entre les positions programmées et leur intégration dans un programme complêt fait l'objet du post-processeur.

L'aptitude au parallélisme des langages imposera cependant que dès la simulation des instructions puissent assurer l'aspect synchronisation de mouvement. On retrouve la même limitation sur l'aspect temporel.

6.1.2 Maîtrise de la précision.

Si la structure de données intégrées au produit développé autorise la conception de nouvelles cellules et leur test, cette fonction ne doit pas masquer le but essentiel à atteindre : la programmation hors ligne. Le chapitre IV a démontré que la maîtrise de la précision devait porter sur tous les composants de la chaîne cinématique impliqués dans l'exécution de la tâche.

6.1.2.1 Positionnement.

L'aspect repositionnement a été principalement développé car il nous est imposé par le procédé. En effet, contrairement au domaine de l'usinage où les pièces sont fixées avec précision sur des palettes elles-mêmes soigneusement bridées sur les tables de machines-outils, la position de la pièce en soudage n'est imposée qu'avec très peu de précision au départ. La détermination automatique de cette position par palpage à l'aide du robot évite soit d'imposer une précision (usinage, calage) inutile et coûteuse, soit une mesure manuelle longue et difficile.

La méthode proposée vise le nombre minimum de points de palpage. Le calcul des erreurs a mis en évidence que de nombreux paramètres peuvent nuire à la précision du résultat. Les essais réalisés ont confirmé ces calculs. Des solutions existent pour limiter ces erreurs, elles sont proposées en 4 . Leur mise en oeuvre nécessite l'application d'un procesus décisionnel qui n'a pas été développé. Ce processus tirera ses critères et ses actions de l'étude réalisée.

6.1.2.2 Identification.

D'autre part, une identification des dispositifs articulés commandés en coordonnées articulaires (positionneur par exemple) améliorerait la précision de la programmation de l'ensemble de la chaîne cinématique. L'application des travaux de M.J. Aldon (voir 4.3.2), aux mesures assurées par le robot en utilisant la méthode de palpage doivent pouvoir assurer cette identification.

6.1.2.3 Précision et suivi de joint.

La recherche de la précision de programmation n'exclut absolument pas le contrôle de l'exécution, et particulièrement dans le cas de la soudure à l'arc, l'utilisation d'un suivi de joint car ce sont deux démarches complémentaires vers le même but, la qualité du produit final.

6.2 ETAT D'AVENCEMENT ET PERSPECTIVES.

6.2.1 Etat d'avancement.

La première maquette du logiciel de simulation nous a aidé à formuler le cahier des charges d'un poste de programmation complet, sans aller toutefois jusqu'à la validation de la programmation sur un robot, le protocole entre l'ASEA et un ordinateur n'étant pas encore mis en oeuvre.

La phase d'analyse issue du cahier des charges a été présentée dans le chapitre V. Toutes les fonctionalités présentées ont été développées. Leur intégration dans les différents modules est en cours. Ces développements sont réalisés sur le HP 9000 du LIMSI avant implantation sur le matériel cible. Celle-ci devra être réalisée pour la fin Novembre 1987.

6.2.2 Perspectives.

Si l'on veut concerver ces performances et son rôle de programmation, le poste de travail ne peut pas intégrer toutes les possibilités offertes par les recherches sur la programmation hors ligne, bien que cela soit tentant. Il est pourtant deux points qu'il semble important de développer dans l'avenir.

La prise en compte de plusieurs chaînes cinématiques pour pouvoir traiter des ensembles comportant plusieurs robots ou positionneurs. Si ce point est déjà intégré dans la structure de données, le module de simulation ne l'intègre pas encore. Il nécessitera la présence d'instructions dans le programme pour signaler l'exécution de tâches en parallèle.

La simulation temporelle de l'exécution est fortement lié au développement précédent. Une première étape peut être l'échantillonnage des mouvements à période fixe en tenant compte des vitesses programmées.

6.3 ATELIER PILOTE DE MECANOSOUDAGE ROBOTISE.

La connaissance progressive du soudage robotisé, par l'intermédiaire de cette étude, a progressivement fait naître le projet de réaliser un atelier pilote de mécano-soudage.

Ce projet commun à l'ISEN et l'Institut Catholique des Arts et Métiers est parti de plusieurs constatations issues des contacts que nous avions eus avec des industriels du soudage.

1. Le mode d'exploitation des robots de soudure, et nous l'avons déjà expliqué au début du chapitre V, n'est pas satisfaisant par la place qu'il impose à l'opérateur.
2. L'évolution technologique que suppose le soudage robotisé doit être suivi par une amélioration de la qualité (précision, ...) des moyens de préparation des éléments à souder, ..., ronds, profilés,...
3. La productivité d'éléments automatisés doit être améliorée par la diminution des stocks intermédiaires, et l'intégration dans un ensemble plus vaste incluant la manipulation des produits intermédiaires ou finaux.

De ces constatations est née la conception de l'atelier flexible de mécano-soudage Il intègre l'ensemble de la chaîne de fabrication de la préparation des tôles au soudage et à l'évacuation. Après bien des évolutions, le schéma d'implantation est défini et des simulations de fonctionnement sont en cours pour la valider. Il comprend - voir figure-:

1. Un poste de découpe plasme à commande numérique.
2. Un poste de sciage à commande numérique des ronds et profilés.
3. Une cellule de soudage autour du robot ASEA IRB 6-2 équipé d'une source de soudage en MIG pulsé.
4. Un robot ASEA IRB 90 assure l'ensemble des manipulations. Les premières simulations montrent que cela est possible car les postes critiques en temps sont le soudage et la découpe.
5. Un réseau local LAC de la Société COMPEX assure la connexion entre le calculateur de contrôle de l'ensemble, le poste de programmation graphique, et les diverses commandes.

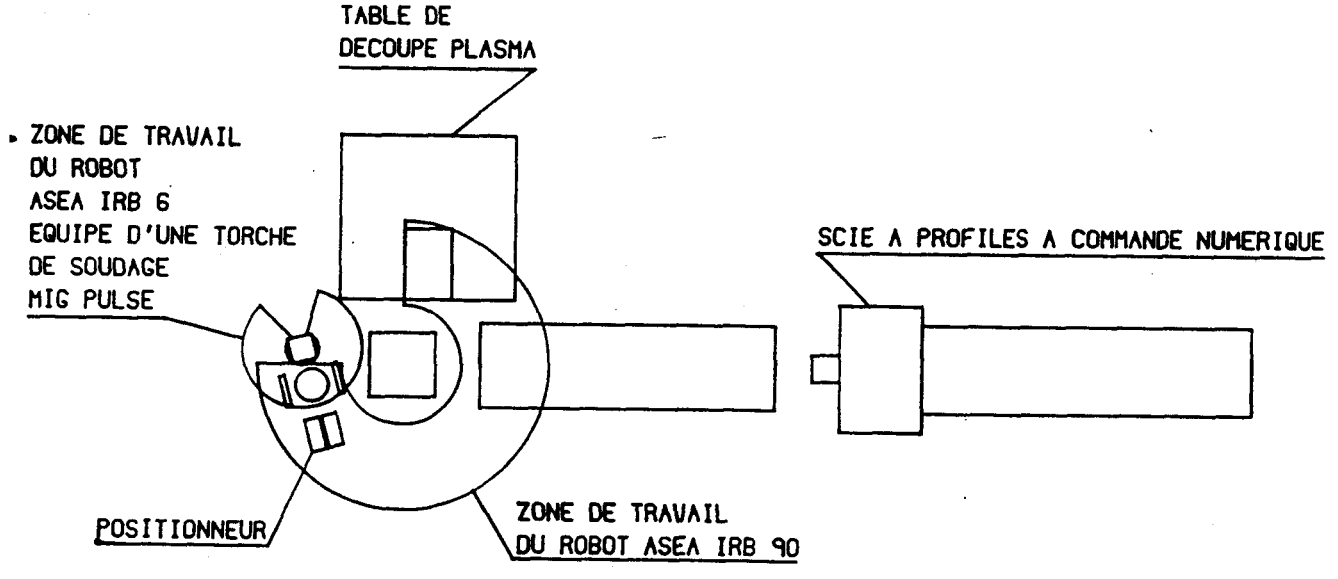


Figure 6.24. Atelier flexible de soudage.

Les objectifs principaux recherchés à travers la mise en place d'un tel atelier sont :

- la validation sur un site de production de la programmation hors ligne en soudage à l'arc ;
- le développement de la collaboration entre robot pour limiter les fixations des tôles par un pointage automatique ;
- maîtrise d'un ensemble complexe de production à travers un réseau local hétérogène.

Bien sûr, ce projet obligera le poste de programmation à évoluer encore, pour prendre en compte un robot 6 axes, l'IRB 90, pour s'adapter aux tâches de manutention. Une étape est à peine entrevue sur notre route qu'il faut déjà songer aux suivantes ! Passionnant !

BIBLIOGRAPHIE.

BIBLIOGRAPHIE

[ACK 85] DS. ACKERSON, D.R HARRY

"Theory, experimental results, and recommended standards regarding the static positionning and orienting precision of industrial robots".
Robotics & computer - integrated manufacturing - Vol 2 nx 314 1985.

[ALD 86] M.J ALDON

"Identification des param tres structuraux des robots manipulateurs"
Colloque ATP - Outil mathématique pour la modélisation et la commande des Robots"
Paris - 15-16 Septembre 1986.

[BJO 86] M. BJORKELUND

" A True Seamtracker for Arcwelding".
Proceedings of the 16 th. International Symposium on Industrial Robot.
Brussels. 30 September 2 October 1986.

[BLA 87] J.L BLANCHON

"Generation et identification des Lois de mouvements en robotique - Application à l'identification et à la correction de trajectoires du manipulateur PUMA 560".
Thèse à l'Université des Sciences et Techniques du Languedoc - 17 Mars 1987.

[BOR 86] G. BORREL

"Configurations multiples et volume de travail".
Colloque ATP - Outil mathématique pour la modélisation et la commande des robots".
Paris 15-16 Septembre 1986.

[BOU 83] M. BOUREZ - F. LEGRE - M. PARENT - J.G QUEROMES - A. RENAULT.

"Automatic programming of a spray robot"
Congrès Autofact
Genève - September 1983

[CAS 86] JEAN-PHILIPPE CASSAR

" Un Simulateur Graphique pour la programmation de Cellule de Soudage Robotise : l'Apprentissage Hors Ligne".
Proceeding of the 5 th. European Conference on CAD/CAM and Computer Graphics.
Paris. February 24-28 1986.

[COI 82] P.COIFFET M.CHIROUZE P.KUSPIYANTO J.VERTU P.MARCHAL G.CLEMENT

"Système de coordination automatique télévision-télémanipulation."
Etat de la robotique en France - Tome 2 - HERMES 1982.

[COL 85] J.C COLSON - N.D PERREIRA.

"Quasi static performance of robots"
Robotics computer - integrated manufacturing Vol 2 nx 314 1985

[DET 87] J.M DETRICHE

"Robot de soudage - commande et programmation"
HERMES 1987.

[DIL 86] R. DILLMAN - B. HORNUNG - M. HUCK

"Interactive Programming of Robots using textual Programming and Simulation technique"
16th ISIR Brussel 86.

[DOM 84] E.DOMBRE P.BORREL A.LIEGEOIS

" A CAD system for programming and evaluating robot actions."
Digital system for automation - Vol 2 - 1984 - p 201-226.

[ELZ 84] EL-ZORKANY

"Automatic Location Correction in off-line programming of industrial Robots"
14th International Symposium on Industrial Robots",
Gotheburg, Sweden, October 1984.

[ELZ 86] EL-ZORKANY - R. LISCANO - B. TONOU - G. SOWATKY.

"Sensor-based location and trajectory specificatino and correction in Robot programming
16th ISIR - Bruxelles 1986.

[ENG 85] J.ENGELBERGER

" La naissance d'une industrie."
Le journal de la robotique - septembre 1985 - nr 11.

[FAU 86] O.D FAUGERAS, M. HEBERT

"Representation, recognition, and locating of 3D objects"
The international Journal of Robotics Research Volume 5 Number 3 1986.

[FEA 84] R.S FEARING - J.M HOLLERBACH

"Base solid Mechanics for tactiles sensing"
MIT - A.I meno 771 - March 1984

[FIN 75] R. FINKEL

"An overview of AL, a programming system for automation".
Proc of the fourth International joint conference on Artificial Intelligence
1975 MIT Press.

[FOL 84] J.D FOLLEY - A. VAN DAM

"Fundamentals of Interactive Computer Graphics"
ADDISON - Wesley Publishing Company 1984.

[FOX 86] M.S.Fox

"Industrial application of artificial intellegence"
Robotic. Vol.2 1986.

[GAR 85] YVON GARDAN

" Modelisation geometrique- Seminaire d'introduction".
Proceedings of the 4 th. European Conference on CAD/CAM and Computer Graphics.
Paris. 1985.

[GAS 84] PETER C.GASTON and TOMAS LOZANO-PEREZ

" Tactile Recognition and Localization Using Object Models: The Case of Polyhedra on a plane".
IEEE Transaction on Pattern Analysis and Machine Intelligence. PAMI-6, # 3 - May 1984. pp 257-266.

[GRI 83] W. ERIC L. GRIMSON, THOMAS LOZANO-PEREZ

" Model-Based Recognition and Localization from Sparse or Tactile Data"
Massachusetts Institute of Technology Artificial Intelligence Laboratory. A. I. Memo 738. August 1983.

[GRI 86] W. ERIC L. GRIMSON

" The Combinatorics of Local Constraints in Model-Based Recognition and Localization from Sparse Data". Journal of the Association for Computing Machinery. Vol. 33, # 4, October 1986. pp 658-686.

[GRU 83] W.A GRUVER - B.I SOROKA - JOJ GRAIG? - T.L TUNER

"Evaluation of commercialy available Robot Programming Languages"
13th ISIR - April 83 - Chicago.

[HAV 86] S.HAVLIK

" A Three Axis Tactile Sensor-Probe for Robotic Use".
Robotica. 1986. Vol. 4. pp 229-235.

[HAY 86] V. HAYWARD - R.P PAUL

"Robot Manipulation Control under Unix RCCL : A Robot Control "C" Libary"
The international journal of Robotics Research. Vol 5 nx 4 Winter 1986.

[HEN 85] L.HENNINGER

" SIMIR-L: Un interpréteur pour l'intégration en robotique."
Thèse de Docteur Ingénieur - Université de Paris Sud
13 décembre 1985.

[HOF 86] REGIS M.HOFFMAN

" Applications of High-Performance Graphics Workstations in Robotics Simulation".
Proceedings of the 16 th. International Symposium on Industrial Robot.
Brussels. 30 September 2 october 1986.

[HOR 86] M.L HORNICK - B. RAVANI

"Computer-Aided off-line programming of Robot Motion"
The international Journal of Robotics Research Vol 4 nx 4 - Winter 1986.

[ISH 86] M. ISHII, S.SAKANE and M. KAKIKURA and Y. MIKAMI

" A New Calibration System for Improving Absolute Positioning Accuracy of Robot Manipulators".
Proceedings of the 16 th. International Symposium on Industrial Robot.
Brussels. 30 September 2 October 1986.

[KAZ 86] H. KAZEROONI, P.K HOUPT

"Robust Compliant Motion For Manipulators"
IEE Journal of Robotics and Automation Vol RA 2, nx 2, June 86.

[KOD 86] N. KODAIRA, M. OSHITA and H. MARUYAMA and M. UMETSU and K. MIURA

" An Off-Line Programming System for Spot Welding Robots".
Proceedings of the 16 th. International Synposium on Industrial Robot.
Brussels. 30 September 2 October 1986.

[LEE 83] C.S G LEE

"Robot Arm Kinematics"
IEE Tutorial on robotics pp 47-65 1983.

[LIE 84] A. LIEGEOIS - P. BORREL - E. DOMBRE.

"Programming, simulating and evaluating Robot Action".
2nd Int Symposium of Robotics Research.
Kyoto, Japan, August 1984.

[LIE 84] A. LIEGEOIS - B. TONOU - P. TOURON

"Simulation de Trajectoires et contrle automatique des efforts aux articulations d'un robot".
Proc ASME Congress Modeling and simulating
Minneapolis August 84.

[LIE 84] C - A. LIEGEOIS

"Analyse de performances et CAO"
Les robots - Tome 7 - HERMES 1984.

[LOZ 83] T.LOZANO-PEREZ

"Spatial planning: a configuration space approach."
IEE Trans. Computer (C-32) 108-120 1983.

[MAZ 84] E.MAZER J.F. MIRIBEL

"Le langage LM: manuel de référence."

TAI- édition CEPADUES 1984.

[MEL 85] A.MELLER

" APSIS: un système pour l'expression de connaissances décisionelles en robotique."
Thèse de Docteur Ingénieur - Université de Paris Sud
30 mai 1985.

[NEM 85] B. NEMEC J. LENARCIC

"A robot simulation system based on kinematic ananlyse."
Robotica (1985) Vol. 3, p 79-84.

[NIL 80] J. NISSON

"Principles of artificial intelligence."
Tiage Publishing Compagny - Palo Alto - 1980.

[OSO 85] A.OSORIO L.HENNINGER A.MELLER

"Programmation d'une cellule flexible d'assemblage."
Actes des journées SM90
Versailles - décembre 1985.

[PAN 85] J.PANISET

" Le point sur le soudage robotisé."
Le journal de la robotique - octobre 1985 - nr 12.

[PAU 79] R. PAUL B.SHIMANO

"Compliance control"
Joint autocontrol conférence - 1976.

[PAU 76] R. PAUL

"Wave: a model based language for manipulator control."
SME Tecnical paper MR 76-615 - 1976.

[PAY 85] D. PAYANNET M.J. ALDON A. LIEGEOIS

"Identification and compensation of mecanical errors for industrial robots."
15th ISIR
Tokyo, Japan - 11-13 september 1985.

[PER 84] L. PERALTA A. OSORIO K. BEN RHOUMA

"Vision system for advenced automation and robotics."
IASTED IA'84
Innsbruck, Autriche - février 1984.

[QUE 82] J.G.QUEROMES

" Computer aided design and robotics: a full promise collaboration."
Proceedinng of 12th ISIR
Paris - June 1982.

[RAI 81] M.H. RAIBERT J.J.CRAIG

"hybrid position / force control of manipulators."
Trans. Dynamic system, measurement, control of ASME - Vol 102 June 1981.

[RAN 84] P. G. RANKY

"Test method and software for robot qualification."
The industrial robot - june 1984.

[RIS 85] A. RISBOURG S. DA COSTA

"Programmation hors ligne: vers une mise une oeuvre industrielle."
Cao et robotique: Etat de l'art.
Actes du symposium AFRI-MICADO - HERMES 1986.

[SAL 80] K. J. SALISBURY

"Active stiffness control of manipulator in cartesian coordonates."
Proceeding of 19th IEE conference decision, control - december 1980.

[SAT 81] T.SATA F.MIMURA A.AMANO

" Robot simulation system as a task programming tool."
Proceeding of 11th ISIR
Tokyo - October 1981.

[SHI 79] B. SHIMANO

"VAL: a versatile robot programming and control system."
Proceeding of COMPSAC79 p 878-883 IEEE - 1979.

[SHI 87] Y. SHIRAI J.TSUJII

"Intelligence artificielle concept technique et application."
Eyrolles - 1987.

[SJO 83] P.SJOLUND M.DONATH

"Robot Task planning: Programming using interactive computer graphics."
Proceeding of 13th ISIR - April 1983.

[SOL 84] E.J.SOL J.A.T.M.VAN DEN BROEK

" Progress in CAD-tools for robot based flexible automation systems."
Proceeding of 14th ISIR
Gothenburg, Sweden - October 1984.

[SOR 83] BARRY-IRWIN SOROKA

" What Can't Robot Languages do ? ".
Proceeding of 13 th. International Symposium on Industrial Robot.
Chicago. April 1983.

[SWE 85] L. M. SWEET

"Sensor-based control system for arc welding robots."
Robotics & computer-integrated manufacturing - Vol.2 nr 2 - 1985.

[TAK 81] K. TAKASE R. P. PAUL

"A structured approach to robot programming and teaching."
IEE Trans. on system, man and cibernetics - Vol. SMC 11 nr4 - april 1981.

[WEC 84] M. WECK T. NIEHAUS

"Off-line programming via standardized interfaces."
The industrial robot - september 1984.

[WHI 79] D.E. WHITNEY J.L. NEVINS

"What is the Remote Center Compliance (RCC) and what can it do?"
9th ISIR
Washington - 1979.

Divers.

[AXE 86] " les classique de la robotique : HUARD."

Axes robotique - octobre 1985 - nr 20.

[JRO 86] " L'atisan et le robot."

Le journal de la robotique -septembre 1986 - nr 23.

ANNEXE 1

ESTIMATION DES ERREURS

annexe 1.

ESTIMATION DES ERREURS.

erreur sur l'orientation de la face.

L'erreur sur les positions comporte trois composantes $\epsilon_r, \epsilon_s, \epsilon_v$ *avec* $\epsilon_s \approx$ *constant et* $\epsilon_v = -\alpha N$
ϵ_r est l'erreur aléatoire due au robot, ϵ_s l'erreur systématique due au robot et au capteur, ϵ_v l'erreur aléatoire due au capteur.

$$\epsilon^p = \epsilon_r + \epsilon_s + \epsilon_v$$

$\mathbf{p}_1$, $\mathbf{p}_2$, $\mathbf{p}_3$ sont trois vecteurs qui représentent les points mesurés.
$|(\mathbf{p}_1 - \mathbf{p}_2)| = |(\mathbf{p}_3 - \mathbf{p}_2)| = d$ est la distance entre les points mesurés sur la face.

$$\mathbf{n} = (\mathbf{p}_1 - \mathbf{p}_2) \times (\mathbf{p}_3 - \mathbf{p}_2)$$

Par dérivation on obtient:

$\partial\mathbf{n} = \partial(\mathbf{p}_1 - \mathbf{p}_2) \times (\mathbf{p}_3 - \mathbf{p}_2) + (\mathbf{p}_1 - \mathbf{p}_2) \times \partial(\mathbf{p}_3 - \mathbf{p}_2)$
$\partial(\mathbf{p}_1 - \mathbf{p}_2) = (\epsilon_{r_1} - \epsilon_{r_2}) + (\epsilon_{v_1} - \epsilon_{v_2})$

Le maximum du deuxième terme est obtenu lorsque les vecteurs sont opposés: $\epsilon_{v_2} = -\epsilon_{v_1}$

$\partial(\mathbf{p}_1 - \mathbf{p}_2) = (\epsilon_{r_1} - \epsilon_{r_2}) + 2\alpha\mathbf{n}$

De même pour $(\mathbf{p}_3 - \mathbf{p}_2)$:

$\partial(\mathbf{p}_3 - \mathbf{p}_2) = (\epsilon_{r_3} - \epsilon_{r_2}) + 2\alpha\mathbf{n}$

$$\partial\mathbf{n} = \partial\mathbf{n}_r + \partial\mathbf{n}_v$$

Calcul de $\partial\mathbf{n}_v$:

$\partial\mathbf{n}_v = 2\alpha(\mathbf{n} \times (\mathbf{p}_3 - \mathbf{p}_2) + (\mathbf{p}_1 - \mathbf{p}_2) \times \mathbf{n})$

Si θ est l'angle entre $(\mathbf{p}_1 - \mathbf{p}_2)$ et $(\mathbf{p}_3 - \mathbf{p}_2)$:

$$\begin{aligned} |\partial\mathbf{n}_v|^2 &= (2\alpha)^2 (d^2 + d^2 + 2\cos\theta d^2) \\ &= (2\alpha)^2 d^2 (2 + 2\cos\theta) \\ &= 16\,\alpha^2 d^2 \cos^2\frac{\theta}{2} \end{aligned}$$

Car $\mathbf{n}$ est perpendiculaire à $(\mathbf{p}_3 - \mathbf{p}_2)$ et $(\mathbf{p}_1 - \mathbf{p}_2)$: la norme de $\mathbf{n} \times (\mathbf{p}_3 - \mathbf{p}_2)$ et $\mathbf{n} \times (\mathbf{p}_1 - \mathbf{p}_2)$ est égal à d.

$$|\partial\mathbf{n}_v| = 4\alpha\, d \cos\frac{\theta}{2}$$

$$\frac{|\partial\mathbf{n}_v|}{|\mathbf{n}|} = \frac{4\alpha\, d \cos\frac{\theta}{2}}{d^2 \sin\theta}$$

$$\frac{|\partial\mathbf{n}_v|}{|\mathbf{n}|} = \frac{2\alpha}{d \sin\frac{\theta}{2}}$$

Calcul de $\partial \mathbf{n}_r$:

$$\partial \mathbf{n}_r = (\epsilon_{r_1} - \epsilon_{r_2}) \times (\mathbf{p}_3 - \mathbf{p}_2) + (\mathbf{p}_1 - \mathbf{p}_2) \times (\epsilon_{r_3} - \epsilon_{r_2})$$

On trouve une borne supérieure à $\partial \mathbf{n}_r$ en remplaçant $(\epsilon_{r_1} - \epsilon_{r_2})$ par $2\epsilon_{r_1}$ et $(\epsilon_{r_3} - \epsilon_{r_2})$ par $2\epsilon_{r_3}$. Ceci est bien une borne supérieure car il est peu probable que ϵ_{r_2} soit opposé à la fois à ϵ_{r_1} et ϵ_{r_2}.

$$\partial \mathbf{n}_r = 2[\epsilon_{r_1} \times (\mathbf{p}_3 - \mathbf{p}_2) + (\mathbf{p}_1 - \mathbf{p}_2) \times \epsilon_{r_3}]$$

ϵ_{r_1} et ϵ_{r_3} balayent un cercle dans le plan orthogonal respectivement à $(\mathbf{p}_1 - \mathbf{p}_2)$ et $(\mathbf{p}_3 - \mathbf{p}_2)$. On exprime ϵ_{r_i} dans un repère normé constitué à partir de $\mathbf{p}_i - \mathbf{p}_2$ et $\mathbf{n}$. L'angle α_i indique la position de ϵ_{r_i} dans le cercle.

$$\epsilon_{r_i} = [\cos\alpha_i \, (\mathbf{p}_i - \mathbf{p}_2) \times \mathbf{n} + \frac{\sin\alpha_i}{d}((\mathbf{p}_i - \mathbf{p}_2) \times \mathbf{n}) \times (\mathbf{p}_i - \mathbf{p}_2)] \frac{\epsilon}{|\mathbf{n}| d}$$

On appele $\epsilon_{r_{i0}}$ le vecteur $(\mathbf{p}_i - \mathbf{p}_2) \times \mathbf{n} / \frac{\epsilon}{|\mathbf{n}| d}$.

Les erreurs sont faibles et l'on peut assimiler $\partial \mathbf{n}_r$ à sa projection dans le plan orthogonal à $\mathbf{n}$: $\mathbf{n} \times \partial \mathbf{n}_r$.

$$\mathbf{n} \times \partial \mathbf{n}_r = 2[\mathbf{n} \times \epsilon_{r_1} \times (\mathbf{p}_3 - \mathbf{p}_2) + \mathbf{n} \times (\mathbf{p}_1 - \mathbf{p}_2) \times \epsilon_{r_3}]$$

On rappelle que $\mathbf{v}_1 \times (\mathbf{v}_2 \times \mathbf{v}_3) = (V_1 \cdot \mathbf{v}_3)\mathbf{v}_2 - (V_1 \cdot \mathbf{v}_2)\mathbf{v}_3$.

$$\mathbf{n} \times \partial \mathbf{n}_r = 2[(\mathbf{n} \cdot (\mathbf{p}_3 - \mathbf{p}_2))\epsilon_{r_1} - (N \cdot \epsilon_{r_1})(\mathbf{p}_3 - \mathbf{p}_2) + (N \cdot \epsilon_{r_3})(\mathbf{p}_1 - \mathbf{p}_2) - ((\mathbf{p}_1 - \mathbf{p}_2) \cdot N)\epsilon_{r_3}].sp$$

$\mathbf{n}$ est perpendiculaire à $(\mathbf{p}_i - \mathbf{p}_2)$:

$$\mathbf{n} \times \partial \mathbf{n}_r = 2[(\mathbf{n} \cdot \epsilon_{r_3})(\mathbf{p}_1 - \mathbf{p}_2) - (N \cdot \epsilon_{r_1})(\mathbf{p}_3 - \mathbf{p}_2)]$$

$$(N \times \partial \mathbf{n}_r) = \frac{2}{d}[\sin\alpha_3 (N \cdot [\epsilon_{r_{30}} \times (\mathbf{p}_3 - \mathbf{p}_2)](\mathbf{p}_1 - \mathbf{p}_2) - \sin\alpha_1 (N \cdot [\epsilon_{r_{10}} \times (\mathbf{p}_3 - \mathbf{p}_2))(\mathbf{p}_3 - \mathbf{p}_2)]$$

On appelle $a_3 = \mathbf{n} \cdot (\epsilon_{r_{30}} \times (\mathbf{p}_3 - \mathbf{p}_2))$ et $a_1 = \mathbf{n} \cdot (\epsilon_{r_{10}} \times (\mathbf{p}_1 - \mathbf{p}_2))$.

$$a_1 = a_3 = \frac{\epsilon}{|\mathbf{n}| d}$$

$$\frac{|\mathbf{n} \times \partial \mathbf{n}_r|^2}{4} = [\sin^2\alpha_3 a_3^2 d^2] + [\sin^2\alpha_1 a_1^2 d^2] - 2\cos\theta \, d^2 \sin\alpha_1 \sin\alpha_3 a_1 a_3$$

En dérivant par rapport aux α_i :

$$\frac{\partial}{\partial \alpha_i}\left[\frac{|N \times \partial \mathbf{n}_r|^2}{4}\right] = 2\cos\alpha_3 \sin\alpha_3 a_3^2 + 2\cos\alpha_1 \sin\alpha_1 a_1^2 - 2\cos\theta \, a_1 a_3 (\cos\alpha_1 \sin\alpha_3 + \cos\alpha_3 \sin\alpha_1)$$

Le dernier terme est égal à $-2\cos\theta \, a_1 a_3 \sin(\alpha_1 + \alpha_3)$.

La dérivée s'annule dans les conditions suivantes:

$$\sin(\alpha_1 + \alpha_3) = 0 \quad soit \quad \alpha_3 = -\alpha_1 \pm k\Pi$$

et

$$2\cos\alpha_1 \sin\alpha_1 = 0 \quad soit \quad \alpha_1 = \pm k\Pi \quad ou \quad \alpha_1 = \frac{\Pi}{2} \pm k\Pi$$

La norme de ce vecteur est maximale lorsque: $\alpha_3 = -\alpha_1 + k\Pi$ et $\alpha_1 = \frac{\Pi}{2} + k\Pi$.

On obtient alors:

$$\max|\mathbf{n}\times\partial\mathbf{n}_r|=4|\mathbf{n}|d\epsilon\cos\frac{\theta}{2}$$

$$\boxed{\max\frac{|\mathbf{n}\times\partial\mathbf{n}_r|}{|\mathbf{n}|}=2\frac{\epsilon}{d\sin\frac{\theta}{2}}}$$

annexe 1.
erreur sur la direction de l'axe de rotation.

$$\mathbf{r}=\frac{(\mathbf{m}_i-\mathbf{n}_i)\times(\mathbf{m}_j-\mathbf{n}_j)}{|(\mathbf{m}_i-\mathbf{n}_i)|\,|(\mathbf{m}_j-\mathbf{n}_j)|}$$

Nous appellons k_r $\frac{1}{|(\mathbf{m}_i-\mathbf{n}_i)|\,|(\mathbf{m}_j-\mathbf{n}_j)|}$

$$\partial\mathbf{r}=[-\partial\mathbf{n}_i\times(\mathbf{m}_j-\mathbf{n}_j)+(\mathbf{m}_i-\mathbf{n}_i)\times\partial\mathbf{n}_j]\,k_r$$

$$\mathbf{r}\times\partial\mathbf{r}=k_r{}^2\,[\mathbf{r}\cdot\partial\mathbf{n}_i(\mathbf{m}_j-\mathbf{n}_j)-\mathbf{r}\cdot\partial\mathbf{n}_j(\mathbf{m}_i-\mathbf{n}_i)] \qquad (1)$$

$\partial\mathbf{n}_i$ peut être exprimé en fonction d'un angle α_i

$$\partial\mathbf{n}_i=\frac{\epsilon_n}{|\mathbf{n}_i\times\mathbf{r}|}[\cos\alpha_i\,\partial\mathbf{n}_{i0}+\sin\alpha_i(\mathbf{n}_i\times\partial\mathbf{n}_{i0})]$$

ϵ_n est l'amplitude de l'erreur sur la normale à la face.

On choisit $\partial\mathbf{n}_{i0}$ pour que le produit mixte que représente dans (1) $\mathbf{r}\cdot\partial\mathbf{n}_i$ s'annule : $\partial\mathbf{n}_{i0}=(\mathbf{n}_i\times\mathbf{r})\,k_i$ *avec* $k_i=\frac{\epsilon}{|(\mathbf{n}_i\times\mathbf{r})|}$.

On obtient:

$$\partial\mathbf{n}_i=\frac{\epsilon_n}{|\mathbf{n}_i\times\mathbf{r}|}[\cos\alpha_i(\mathbf{n}_i\times\mathbf{r})+\sin\alpha_i(\mathbf{n}_i\times(\mathbf{n}_i\times\mathbf{r}))] \qquad (2)$$

Pour remplacer (2) dans (1) on calcule $\mathbf{r}\cdot\partial\mathbf{n}_i$:

$$\begin{aligned}\mathbf{r}\cdot\partial\mathbf{n}_i&=\sin\alpha_i\,(\mathbf{n}_i\times\partial\mathbf{n}_{i0})\cdot\mathbf{r}\\&=k_i[(\mathbf{n}_i\cdot\mathbf{r})^2-\mathbf{r}^2]\sin\alpha_i\\&=k_i[\cos^2\beta_i-1]\sin\alpha_i \qquad \cos\beta_i=r\cdot\mathbf{n}_i\\&=-\epsilon_n\sin\beta_i\sin\alpha_i=a_i\end{aligned}$$

β_i et β_j sont les angles entre $\mathbf{r}$ et respectivement $\mathbf{n}_i$ et $\mathbf{n}_j$. L'équation (1) devient alors :

$$\mathbf{r}\times\partial\mathbf{r}=k_r{}^2\,a_i\,(\mathbf{m}_j-\mathbf{n}_j)-aj\,(\mathbf{m}_i-\mathbf{n}_i)) \qquad (3)$$

$$|\mathbf{r}\times\partial\mathbf{r}|^2=k_r{}^4[|(\mathbf{m}_j-\mathbf{n}_j)|^2\,a_i{}^2+|(\mathbf{m}_i-\mathbf{n}_i)|^2\,aj^2-2\cos\beta\,a_i\,aj\,|(\mathbf{m}_i-\mathbf{n}_i)|\,|(\mathbf{m}_j-\mathbf{n}_j)|$$

Où β est l'angle entre $(\mathbf{m}_i-\mathbf{n}_i)$ et $(\mathbf{m}_j-\mathbf{n}_j)$.

La norme est maximum dans les mêmes conditions que pour le calcul précédent : $\alpha_j=-\alpha_i\pm k\Pi$ et $\alpha_i=\frac{\Pi}{2}+k\Pi$.

On peut calculer que $|(\mathbf{m}_i - \mathbf{n}_i)| = 2\sin\frac{\theta}{2}\ \sin\beta_i$ et de la même façon $|(\mathbf{m}_j - \mathbf{n}_j)| = 2\sin\frac{\theta}{2}\ \sin\beta_j$ avec θ l'amplitude de la rotation.

Il est alors possible de calculer k_r : $k_r = \dfrac{1}{4\sin\beta\ \sin^2\frac{\theta}{2}\ \sin\beta_i\ \sin\beta_j}$

L'équation (3) devient alors:

$$\begin{aligned}\max|\mathbf{r}\times\partial\mathbf{r}|^2 &= \epsilon_n^2 k_r{}^4[\,|(\mathbf{m}_j - \mathbf{n}_j)|^2\sin^2\beta_i + |(\mathbf{m}_i - \mathbf{n}_i)|^2\sin^2\beta_j + 2\cos\beta\sin\beta_i\sin\beta_j\,|(\mathbf{m}_i - \mathbf{n}_i)|\,|(\mathbf{m}_j - \mathbf{n}_j)| \\ &= \frac{\epsilon_n^2\,(2+2\cos\beta)}{64\sin^4\beta\ \sin^6\frac{\theta}{2}\ \sin^2\beta_i\ \sin^2\beta_j} \\ &= \frac{\epsilon_n^2\cos^2\frac{\beta}{2}}{64\sin^4\beta\ \sin^6\frac{\theta}{2}\ \sin^2\beta_i\ \sin^2\beta_j}\end{aligned}$$

Ce qui donne:

$$\max\frac{|\mathbf{r}\times\partial\mathbf{r}|}{|\mathbf{r}|} = \frac{\epsilon_n}{16\sqrt{2}(1-\cos^2\frac{\beta}{2})\cos\frac{\beta}{2}\sin^3\frac{\theta}{2}\sin\beta_i\sin\beta_j}$$

annexe 1.
erreur sur l'amplitude de la rotation.

On cherche à exprimer les variations de $\mathbf{m}_i$ dans le repère fixe $\mathbf{r}$, $\mathbf{n}_i$ et $\mathbf{r}\times\mathbf{n}_i$, lorsque $\mathbf{r}$ et $\mathbf{n}_i$ varient. Les erreurs sur $\mathbf{r}$ et $\mathbf{n}_i$ sont respectivement $\partial\mathbf{r}$ et $\partial\mathbf{n}_i$.

$$\mathbf{m}_i = \cos\theta\mathbf{n}_i + (1-\cos\theta)(\mathbf{r}\cdot\mathbf{n}_i)\mathbf{r} + \sin\theta(\mathbf{r}\times\mathbf{n}_i) \qquad (1)$$

$$\partial\mathbf{m}_i = \frac{\partial\mathbf{m}_i}{\partial\mathbf{r}} + \frac{\partial\mathbf{m}_i}{\partial\mathbf{n}_i}$$

Si les erreurs sont faibles, la projection de $\partial\mathbf{m}_i$ sur le vecteur perpendiculaire à $\mathbf{m}_i$ et $\mathbf{r}$, soit $\mathbf{r}\times\mathbf{m}_i$,peut etre assimilé à $\partial\theta$, l'erreur sur l'amplitude de la rotation.

La projection dans le plan perpendiculaire à $\mathbf{r}$ de la dérivée de l'équation (1) par rapport à $\mathbf{r}$ donne :

$$\begin{aligned}\mathbf{r}\times\frac{\partial\mathbf{m}_i}{\partial\mathbf{r}} &= \mathbf{r}[(1-\cos\theta)[(\partial\mathbf{r}\cdot\mathbf{n}_i)\mathbf{r} + (\mathbf{r}\cdot\mathbf{n}_i)\partial\mathbf{r}] + \sin\theta(\partial\mathbf{r}\times\mathbf{n}_i)] \\ &= (1-\cos\theta)(\mathbf{r}\cdot\mathbf{n}_i)\mathbf{r}\times\partial\mathbf{r} + \sin\theta\mathbf{r}\times(\partial\mathbf{r}\times\mathbf{n}_i) \\ &= (1-\cos\theta)(\mathbf{r}\cdot\mathbf{n}_i)\mathbf{r}\times\partial\mathbf{r} + \sin\theta(\mathbf{r}\cdot\mathbf{n}_i)\partial\mathbf{r}\end{aligned}$$

Car $\partial\mathbf{r}$ est perpendiculaire à $\mathbf{r}$. On pose $|\partial\mathbf{r}| = \epsilon_r$

$$\begin{aligned}\left|\mathbf{r}\times\frac{\partial\mathbf{m}_i}{\partial\mathbf{r}}\right|^2 &= (1-\cos\theta)^2(\mathbf{r}\cdot\mathbf{n}_i)^2\epsilon_r{}^2 + \sin\theta^2(\mathbf{r}\cdot\mathbf{n}_i)^2\epsilon_r{}^2 \\ &= \epsilon_r{}^2(\mathbf{r}\cdot\mathbf{n}_i)^2[(1-\cos\theta)^2 + \sin\theta^2) \\ &= 4\epsilon_r{}^2(\mathbf{r}\cdot\mathbf{n}_i)^2\sin^2\frac{\theta}{2}\end{aligned}$$

$$\left|\mathbf{r}\times\frac{\partial\mathbf{m}_i}{\partial\mathbf{n}_i}\right| = 2\epsilon_r\cos\beta_i\sin\frac{\theta}{2}$$

Avec $\cos\beta_i = \mathbf{r}\cdot\mathbf{n}_i = \mathbf{r}\cdot\mathbf{m}_i$.

Ce résultat signifie que l'extrémité de $\mathbf{m}_i$ parcours un cercle dans le plan perpendiculaire à $\mathbf{r}$ lorsque $\mathbf{r}$ varie.

$$\begin{aligned}(\mathbf{r}\times\mathbf{m}_i)\cdot\frac{\partial\mathbf{m}_i}{\partial\mathbf{n}_i} &= \frac{1}{\sin}\beta_i\,[\cos\theta(\mathbf{r}\times\mathbf{m}_i)\cdot\partial\mathbf{n}_i + \sin\theta(\mathbf{r}\times\mathbf{m}_i)\cdot(\mathbf{r}\times\partial\mathbf{n}_i)\\ &= \frac{\partial\mathbf{n}_i}{\sin\beta_i}\,[\cos\theta(\mathbf{r}\times\mathbf{m}_i) + \sin\theta(\mathbf{r}\times(\mathbf{r}\times\mathbf{m}_i))]\\ &= \frac{\partial\mathbf{n}_i}{\sin\beta_i}\,\mathbf{v}\end{aligned}$$

$\partial\mathbf{n}_i$ est définie de la manière suivante $\partial\mathbf{n}_i = \frac{\epsilon_n}{\sin\beta_i}\,[\cos\alpha\ \partial\mathbf{n}_{i0} + \sin\alpha\ \mathbf{n}_i\times\partial\mathbf{n}_{i0}]$

On choisit $\partial\mathbf{n}_{i0} = \frac{\epsilon_n}{|\mathbf{n}_i\times\mathbf{v}|}\,(\mathbf{n}_i\times\mathbf{v})$

$$\begin{aligned}(\mathbf{r}\times\mathbf{m}_i)\cdot\frac{\partial\mathbf{m}_i}{\partial\mathbf{n}_i} &= \frac{\sin\alpha}{\sin\beta_i}\,\frac{\epsilon_n}{|\mathbf{n}_i\times\mathbf{v}|}\,(\mathbf{n}_i\times(\mathbf{n}_i\times\mathbf{v}))\cdot\mathbf{v}\\ &= \frac{\sin\alpha}{\sin\beta_i}\,\frac{\epsilon_n}{|\mathbf{n}_i\times\mathbf{v}|}\,(\mathbf{n}_i\times\mathbf{v})^2\end{aligned}$$

Le calcul de $|\mathbf{n}_i\times\mathbf{v}|$ donne 1. On peut en déduire l'expression de $\partial\theta$.

$$\boxed{max\,(tg\,(\partial\theta)) = \frac{\epsilon_n}{\sin\beta_i} + 2\,\epsilon_r\,\cos\beta_i\sin\frac{\theta}{2}}$$

annexe 1.
erreur sur la translation.

[GRI83] nous donne:

$$\begin{aligned}\mathbf{n}_i\cdot\mathbf{v}_0 &= \mathbf{n}_i\cdot\mathbf{p}_i - d_i\\ \mathbf{n}_j\cdot\mathbf{v}_0 &= \mathbf{n}_j\cdot\mathbf{p}_j - d_j\\ \mathbf{n}_k\cdot\mathbf{v}_0 &= \mathbf{n}_k\cdot\mathbf{p}_k - d_k\end{aligned}$$

En dérivant chaque équation on obtient:

$$\begin{aligned}\mathbf{n}_i\cdot\partial\mathbf{v}_0 + \partial\mathbf{n}_i\cdot\mathbf{v}_0 &= \partial\mathbf{n}_i\cdot\mathbf{p}_i + \mathbf{n}_i\cdot\partial\mathbf{p}_i\\ \mathbf{n}_j\cdot\partial\mathbf{v}_0 + \partial\mathbf{n}_j\cdot\mathbf{v}_0 &= \partial\mathbf{n}_j\cdot\mathbf{p}_j + \mathbf{n}_j\cdot\partial\mathbf{p}_j\\ \mathbf{n}_k\cdot\partial\mathbf{v}_0 + \partial\mathbf{n}_k\cdot\mathbf{v}_0 &= \partial\mathbf{n}_k\cdot\mathbf{p}_k + \mathbf{n}_k\cdot\partial\mathbf{p}_k\end{aligned}$$

$$\begin{aligned}\mathbf{n}_i\cdot(\mathbf{n}_j\times\mathbf{n}_k)\,\partial\mathbf{v}_0 + (\partial\mathbf{n}_i\cdot(\mathbf{n}_j\times\mathbf{n}_k) + \partial\mathbf{n}_j\cdot(\mathbf{n}_k\times\mathbf{n}_i) + \partial\mathbf{n}_k\cdot(\mathbf{n}_i\times\mathbf{n}_j))\mathbf{v}_0 = {}& (\partial\mathbf{n}_i\cdot\mathbf{p}_i + \mathbf{n}_i\cdot\partial\mathbf{p}_i)(\mathbf{n}_j\times\mathbf{n}_k) +\\ & (\partial\mathbf{n}_j\cdot\mathbf{p}_j + \mathbf{n}_j\cdot\partial\mathbf{p}_j)(\mathbf{n}_k\times\mathbf{n}_i) +\\ & (\partial\mathbf{n}_k\cdot\mathbf{p}_k + \mathbf{n}_k\cdot\partial\mathbf{p}_k)(\mathbf{n}_i\times\mathbf{n}_j)\end{aligned}$$

Si ϵ est l'erreur de positionnement :$max\,(\mathbf{n}_i\cdot\partial\mathbf{p}_i) = \epsilon$

On pose $\partial\mathbf{n}_i = \frac{\epsilon_n}{\sin\gamma_i\,|\mathbf{p}_i|}\,[cos\,\alpha(\mathbf{n}_i\times\mathbf{p}_i) + \sin\alpha\mathbf{n}_i\times(\mathbf{n}_i\times\mathbf{p}_i)]$ avec $\cos\gamma_i = \frac{\mathbf{n}_i\cdot\mathbf{p}_i}{|\mathbf{p}_i|}$

$\partial \mathbf{n}_i \cdot \mathbf{p}_i$ est maximum quand $\alpha = \frac{\Pi}{2} + k\Pi$ et est egal à $\epsilon_n \, |\mathbf{p}_i| \sin\gamma_i$

$a_i = \epsilon_n \, |\mathbf{p}_i| \sin\gamma_i + \epsilon$ *idem pour j,k*

$$|\max \partial \mathbf{v}_0|^2 = \frac{a_i^2}{\cos^2\rho_i} + \frac{a_j^2}{\cos^2\rho_j} + \frac{a_k^2}{\cos^2\rho_k} + 2a_i a_j \frac{\cos a_{ij}}{\cos\rho_i \cos\rho_j} + 2a_k a_j \frac{\cos a_{kj}}{\cos\rho_k \cos\rho_j} + 2a_i a_k \frac{\cos a_{ik}}{\cos\rho_i \cos\rho_k}$$

Avec $\cos\rho_i = \frac{\mathbf{n}_i \cdot (\mathbf{n}_j \times \mathbf{n}_k)}{|\mathbf{n}_j \times \mathbf{n}_k|}$ et $\cos a_{ij} = \frac{(\mathbf{n}_j \times \mathbf{n}_k) \cdot (\mathbf{n}_k \times \mathbf{n}_i)}{|(\mathbf{n}_j \times \mathbf{n}_k|\,|\mathbf{n}_k \times \mathbf{n}_i|}$

annexe 1.

ANNEXE 2

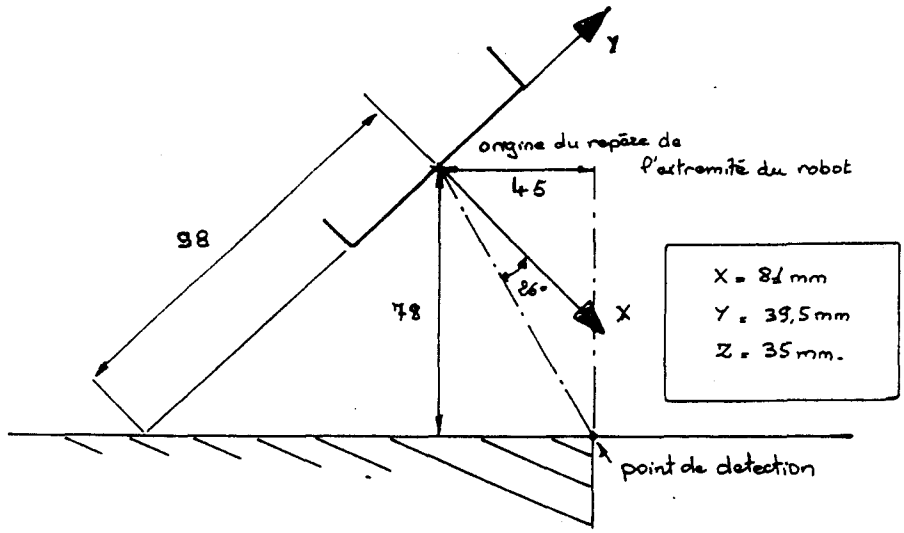
ESSAIS DE REPOSITIONNEMENT

Condition d'essais.

Une boite parallélépipédique (220 mm, 305 mm, 245 mm) est posée sur une table dans espace de travail du robot.

On dégauchit au départ les axes x et y de la feuille de mesure (présentés ci joint) par rapport aux axes du robot. Les axes de la feuille pourra donc servir de référence pour le calcul de l'orientation absolue de la pièce.

Le dispositif tactile est fixé à l'extrémité du robot. On calcule le point de détection par quelques mesures présentés dans le schéma ci-dessous



calcul du centre d'outil.

Des essais en matérialisant ce point ont montré la bonne définition du centre d'outil par la quasi invariance de ce point (inférieure au mm) lors de mouvements de rotation.

Le palpage est réalisé par un programme issu de l'apprentissage et comportant de ce fait des orientations et des positions figées. Les positions de palpage sont transmises à l'ordinateur par l'intermédiaire de la liaison série.

Trois essais B1,V1,V2 sont enregistrés avec une rotation d'axe vertical.

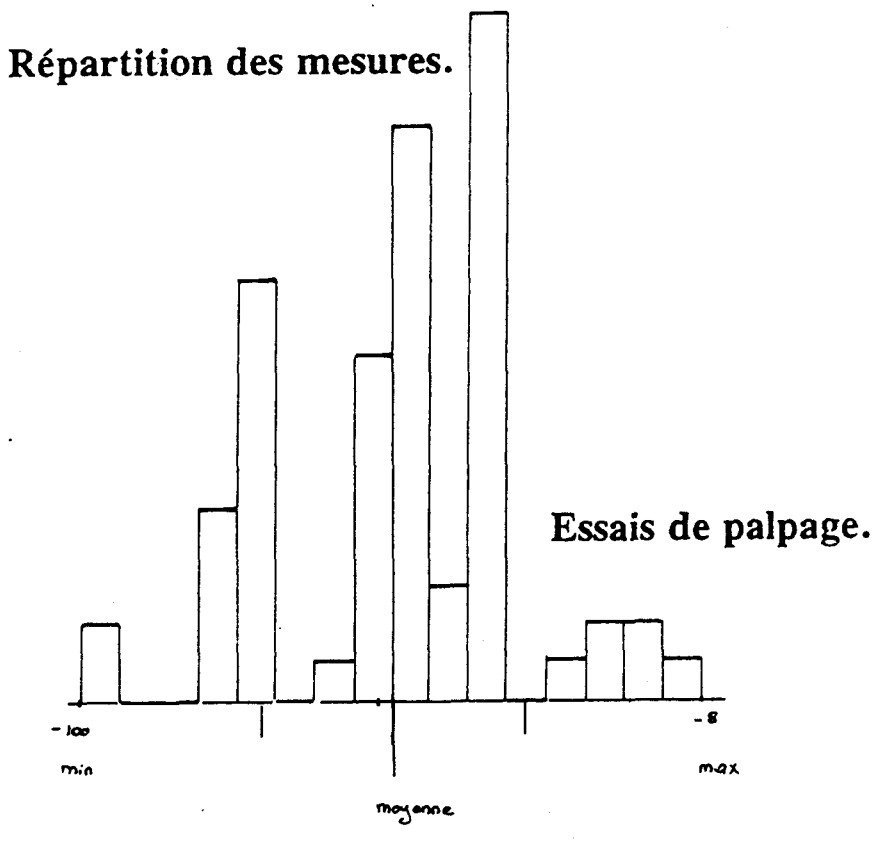
HV2 et HHV2 ajoute à la rotation de V2 une rotation d'axe paralèlle à une arêre de la face posée sur la table. Cette face est élvée respectivement de 20 et 50 mm à 210 mm de l'axe de rotation.

TB1 reprend l'orientation de B1 en la composant à une rotation d'axe Y passant par un sommet posé sur la table: élévation de 20 mm à 149 mm de l'axe de rotation.

Les essais sont consignés sur la feuille où sont tracés les différentes situations de la pièce. Les résultats des calculs sont donnés dans les tableaux suivant.

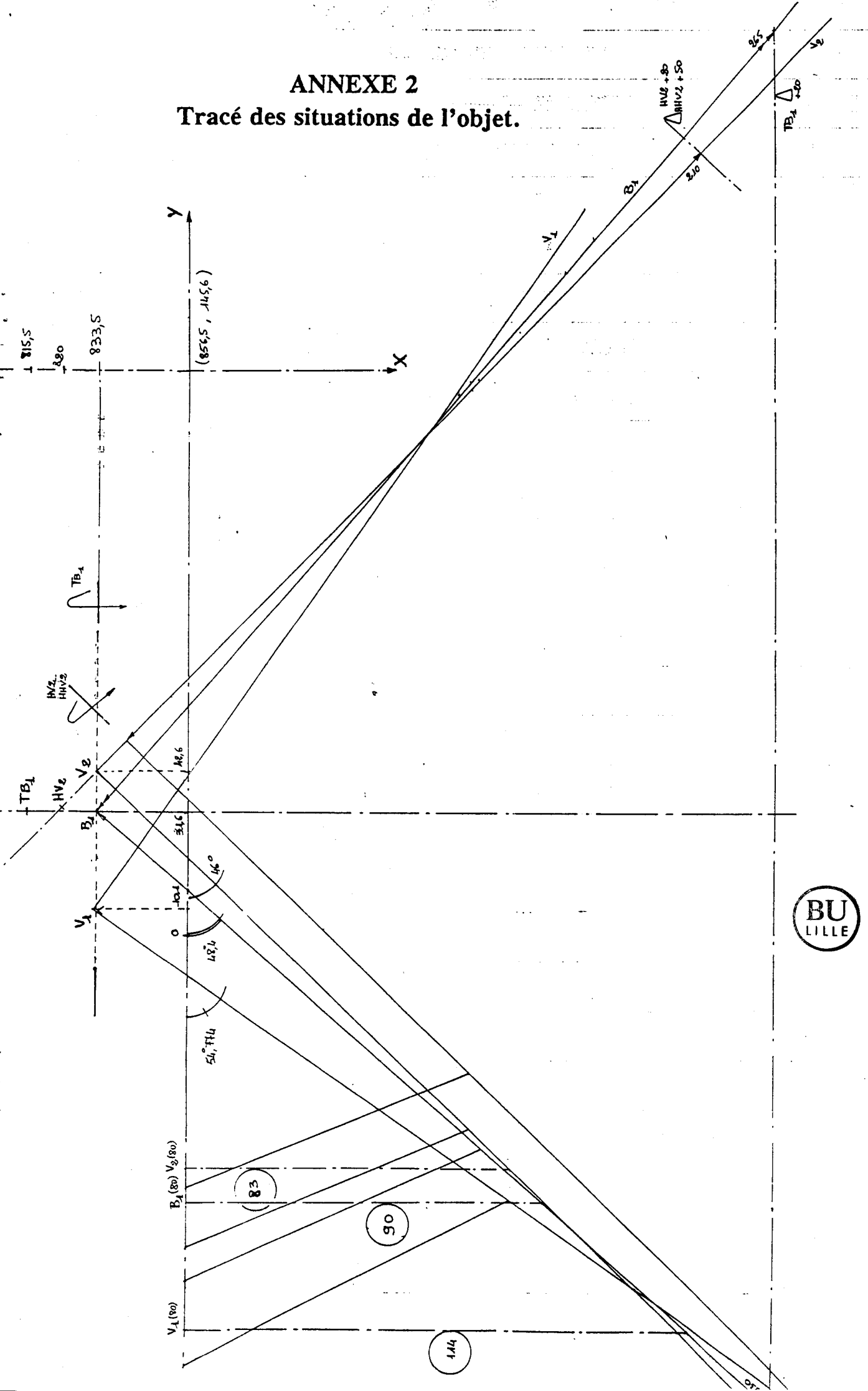
Essais de palpage.

Données robot:	70 points	
max = 769.375000	min = 768.500000	moyenne 768.930359
Valeurs comparateur:	70 points	
max = -8.000000	min = -100.000000	moyenne -50.271427
regression 0.985910		
Données robot:	ecart type 0.201430 soit 71.428574 % des valeurs	
Valeurs comparateur:	ecart type 19.489135 soit 64.285713 % des valeurs	



ANNEXE 2

Tracé des situations de l'objet.



Test B1								
Normales theoriques:								
0.00000	0.00000	1.00000	-1.00000	0.00000	0.00000	0.00000	1.00000	0.00000
Essai 0:			position			normale		
FACE 0:			999.33331	10.37500	727.00000	0.01479	0.00219	0.99989
FACE 1:			935.52081	-65.45834	701.89581	-0.66199	-0.74945	-0.00964
FACE 2:			897.75000	109.77084	691.54169	-0.74747	0.66300	0.04145
axe de rotation :					0.05083	0.01009	0.99866	
angle:					48.544083			
position:					827.59406	28.96436	728.95862	
Essai 1:			position			normale		
FACE 0:			999.33331	10.43750	727.00000	0.01479	0.00219	0.99989
FACE 1:			935.20831	-66.45834	701.89581	-0.66364	-0.74783	-0.01812
FACE 2:			897.66669	109.75000	691.50000	-0.74670	0.66352	0.04663
axe de rotation :					0.06088	0.00321	0.99814	
angle:					48.474369			
position:					826.73938	28.27417	728.68201	
Essai 2:			position		normale			
FACE 0:			999.33331	10.41667	727.02081	0.01656	0.00521	0.99985
FACE 1:			935.27081	-66.27084	701.85419	-0.66167	-0.74969	-0.01261
FACE 2:			897.70831	109.75000	691.50000	-0.74651	0.66377	0.04615
axe de rotation :					0.05757	0.00919	0.99830	
angle:					48.536270			
position:					826.73212	28.23749	728.98590	
Normales attendues:								
0.00000	0.00000	1.00000	-0.66436	-0.74741	0.00000	-0.74741	0.66436	0.00000
valeurs attendues:								
axe de rotation					0.00000	0.00000	1.00000	
angle:			48.366344					

Test V1								
Normales theoriques:								
0.00000	0.00000	1.00000	-1.00000	0.00000	0.00000	0.00000	1.00000	0.00000
Essai 0:			position			normale		
FACE 0:			999.31250	10.37500	724.06250	0.01352	0.00193	0.99991
FACE 1:			934.02081	-72.54166	701.97919	-0.57887	-0.81529	-0.01417
FACE 2:			899.85419	109.04166	691.52081	-0.81391	0.57927	0.04471
axe de rotation :					0.05098	0.00897	0.99866	
angle:					54.654179			
position:					825.81311	3.34100	725.76990	
Essai 1:			position			normale		
FACE 0:			999.33331	10.41667	726.93750	0.01295	0.00067	0.99992
FACE 1:			934.02081	-72.43750	701.95831	-0.58034	-0.81430	-0.01090
FACE 2:			899.83331	109.04166	691.47919	-0.81605	0.57635	0.04340
axe de rotation :					0.04791	0.01133	0.99879	
angle:					54.701267			
position:					826.57129	3.33713	728.63159	
Essai 2:			position		normale			
FACE 0:			999.27081	10.41667	726.91669	0.01603	0.00323	0.99987
FACE 1:			934.02081	-72.52084	701.95831	-0.57799	-0.81590	-0.01545
FACE 2:			899.89581	109.02084	691.52081	-0.81443	0.57842	0.04622
axe de rotation :					0.05314	0.00858	0.99855	
angle:					54.718254			
position:					825.91736	2.95404	728.98718	
Normales attendues:								
0.00000	0.00000	1.00000	-0.57443	-0.81856	0.00000	-0.81856	0.57443	0.00000
valeurs attendues:								
axe de rotation :					0.00000	0.00000	1.00000	
angle:			54.940445					

Test V2								
Normales theoriques:								
0.00000	0.00000	1.00000	-1.00000	0.00000	0.00000	0.00000	1.00000	0.00000
Essai 0:			position			normale		
FACE 0:			999.33331	10.37500	727.14581	0.01258	0.00656	0.99990
FACE 1:			935.18750	-65.52084	701.87500	-0.69809	-0.71596	-0.00849
FACE 2:			898.04169	109.39584	691.47919	-0.71281	0.70017	0.04081
axe de rotation :					0.05356	0.01074	0.99851	
angle:					45.678444			
position:					828.63556	37.27887	728.44257	
Essai 1:			position			normale		
FACE 0:			999.31250	10.41667	725.89581	0.01740	0.00203	0.99985
FACE 1:			935.20831	-65.45834	701.87500	-0.69809	-0.71595	-0.00922
FACE 2:			896.89581	109.95834	691.45831	-0.71140	0.70133	0.04525
axe de rotation :					0.05961	0.01229	0.99815	
angle:					45.640884			
position:					827.82202	38.17319	728.18378	
Essai 2:			position			normale		
FACE 0:			999.29169	10.39583	727.14581	0.01513	0.00713	0.99986
FACE 1:			935.27081	-65.35416	701.81250	-0.69869	-0.71532	-0.01248
FACE 2:			896.97919	109.91666	691.47919	-0.71177	0.70127	0.04001
axe de rotation :					0.05478	0.00565	0.99848	
angle:					45.611027			
position:					827.78485	38.30997	728.80792	
Normales attendues:								
0.00000	0.00000	1.00000	-0.69398	-0.72000	0.00000	-0.72000	0.69398	0.00000
valeurs attendues:								
axe de rotation					0.00000	0.00000	1.00000	
angle:	46.054298							

Test HV2								
Normales theoriques:								
0.00000	0.00000	1.00000	-1.00000	0.00000	0.00000	0.00000	1.00000	0.00000
Essai 0:			position			normale		
FACE 0:			999.31250	10.39583	735.52081	-0.05425	-0.06209	0.99659
FACE 1:			932.77081	-73.66666	701.91669	-0.69492	-0.71217	-0.09951
FACE 2:			895.79169	110.47916	691.47919	-0.71277	0.70017	0.04148
axe de rotation :					0.09385	-0.09465	0.99108	
angle:					45.985477			
position:					820.13171	32.01837	726.54272	
Essai 1:			position			normale		
FACE 0:			999.29169	10.41667	735.45831	-0.05605	-0.06353	0.99640
FACE 1:			932.89581	-73.04166	701.85419	-0.69666	-0.70930	-0.10747
FACE 2:			895.79169	110.52084	691.47919	-0.71297	0.69995	0.04192
axe de rotation :					0.09864	-0.10430	0.98964	
angle:					45.959110			
position:					820.63403	32.62033	726.16968	
Essai 2:			position		normale			
FACE 0:			999.31250	10.37500	735.58331	-0.05246	-0.06056	0.99679
FACE 1:			932.77081	-73.58334	701.91669	-0.69483	-0.71165	-0.10372
FACE 2:			895.77081	110.52084	691.47919	-0.71225	0.70090	0.03798
axe de rotation :					0.09293	-0.10185	0.99045	
angle:					45.974236			
position:					819.91858	32.16739	726.88867	
Normales attendues:								
-0.06579	-0.06826	0.99550	-0.69084	-0.71677	-0.09481	-0.72001	0.69396	0.00000
valeurs attendues:								
axe de rotation					0.04717	-0.11097	0.99270	
angle:	46.358109							

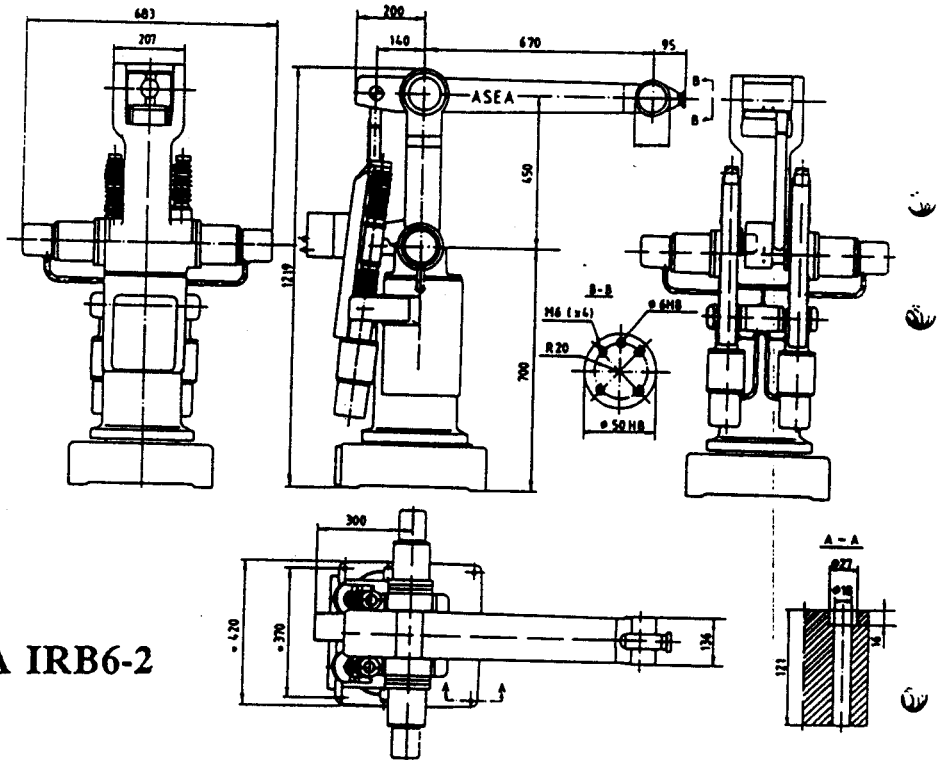
Test HHV2						
Normales theoriques:						
0.00000	0.00000	1.00000	-1.00000	0.00000	0.00000	
Essai 0:			position			normale
FACE 0:			999.31250	10.41667	752.95831	-0.15217 -0.17226 0.97323
FACE 1:			923.91669	-104.60416	702.04169	-0.69565 -0.69198 -0.19294
FACE 2:			895.62500	110.60416	691.47919	
axe de rotation :			0.14590	-0.18271	0.97228	
angle:			47.439579			
position:			799.12451	13.41907	720.92621	
Normales attendues:						
-0.16074	-0.16677	0.97281	-0.67509	-0.70043	-0.23162	-0.72001 0.69396 0.00000
valeurs attendues:						
axe de rotation			0.11245	-0.26457	0.95779	
angle:			47.861042			

Test TB1						
Normales theoriques:						
Essai 0:			position			normale
FACE 0:			999.29169	10.41667	747.41669	-0.10744 0.00320 0.99421
FACE 1:			930.52081	-82.47916	701.97919	-0.66608 -0.74109 -0.08439
FACE 2:			881.91669	117.20834	691.45831	-0.73723 0.67426 -0.04326
axe de rotation :			-0.01438	-0.11343	0.99344	
angle:			48.077824			
position:			800.44440	31.06125	725.31482	
Essai 1:			position			
FACE 0:			999.29169	10.37500	747.35419	-0.10940 0.00164 0.99400
FACE 1:			930.47919	-82.68750	701.97919	-0.66499 -0.74166 -0.08792
FACE 2:			881.91669	117.18750	691.45831	-0.73633 0.67550 -0.03884
axe de rotation :			-0.00885	-0.11477	0.99335	
angle:			48.072384			
position:			799.97522	30.73184	724.69592	
Essai 2:			position			
FACE 0:			999.33331	10.39583	747.39581	-0.10924 0.00023 0.99402
FACE 1:			930.47919	-82.62500	701.97919	-0.66438 -0.74243 -0.08602
FACE 2:			881.95831	117.14584	691.43750	-0.73596 0.67598 -0.03760
axe de rotation :			-0.00831	-0.11206	0.99367	
angle:			48.069668			
position:			799.95355	30.65366	724.80389	
Normales attendues:			-0.11438	0.00135	0.99344	
valeurs attendues:						
axe de rotation			-0.05796	-0.12620	0.99031	
angle: 48.786						

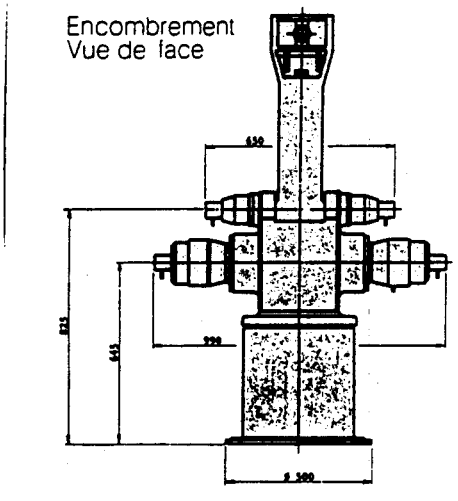
ANNEXE 3

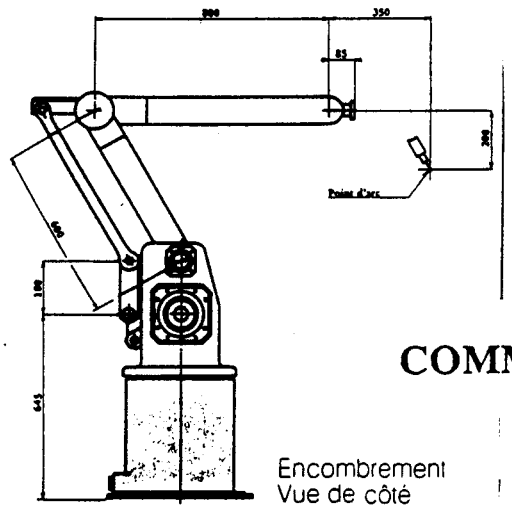
DOCUMENTATION TECHNIQUE.

Robots
Description



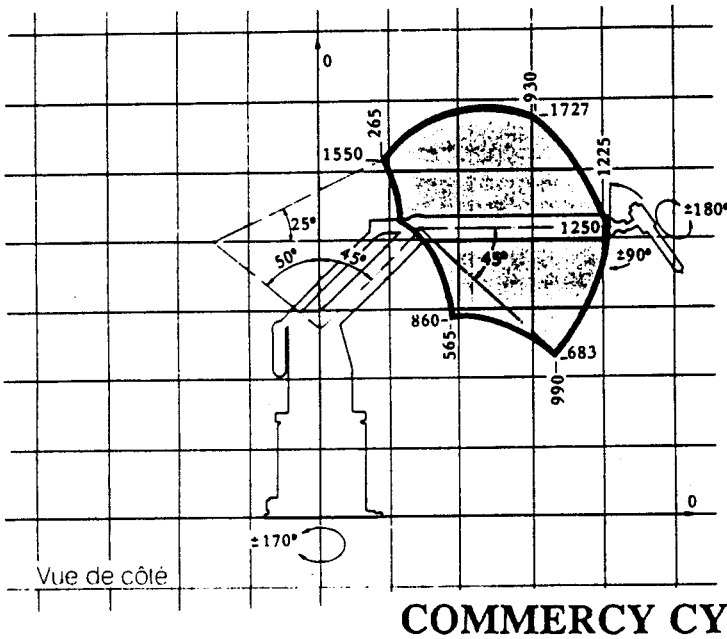
ASEA IRB6-2



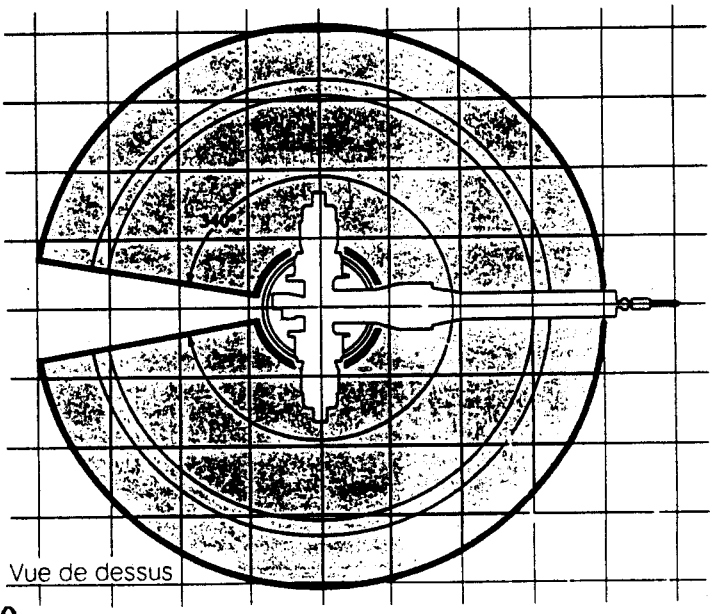


COMMERCY CY800

Volume de travail

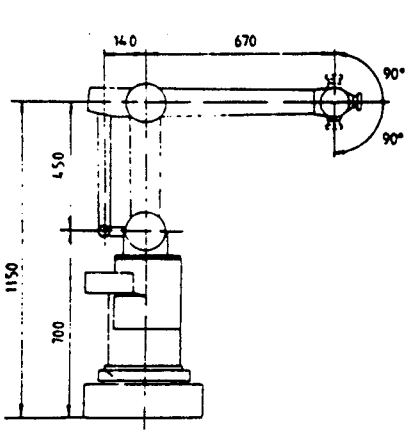


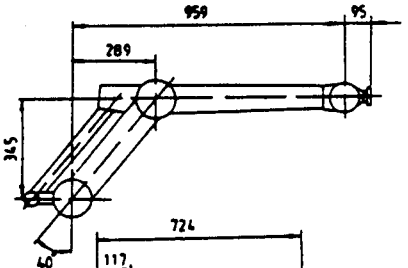
Vue de côté

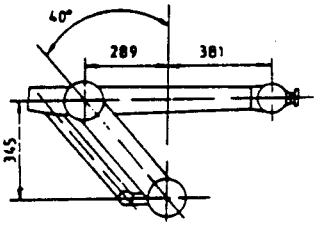


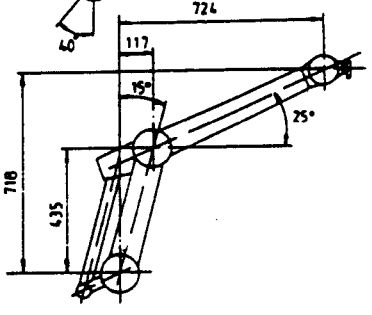
Vue de dessus

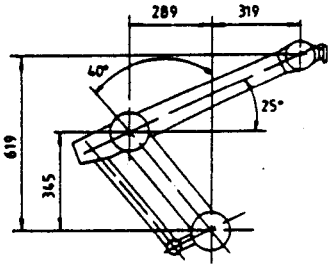
COMMERCY CY800

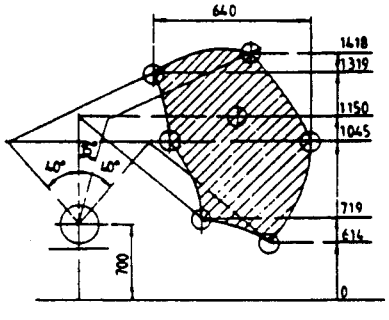


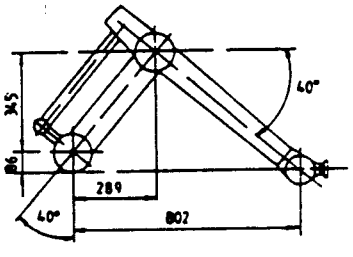


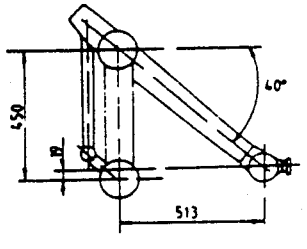








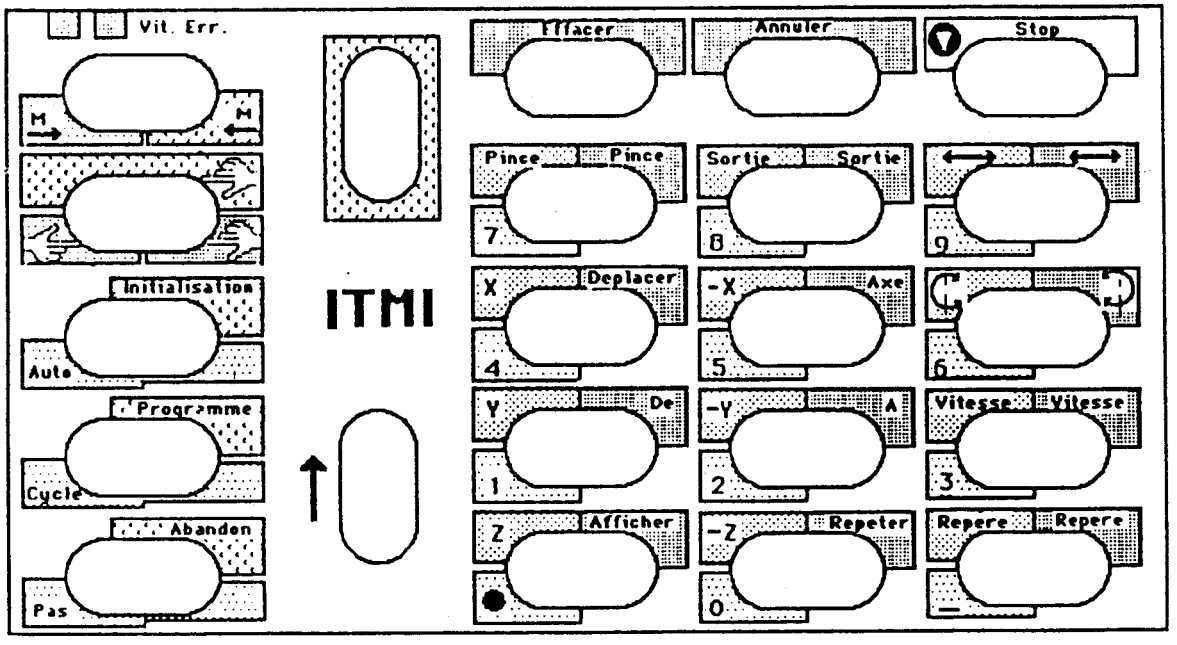




ASEA IRB6-2

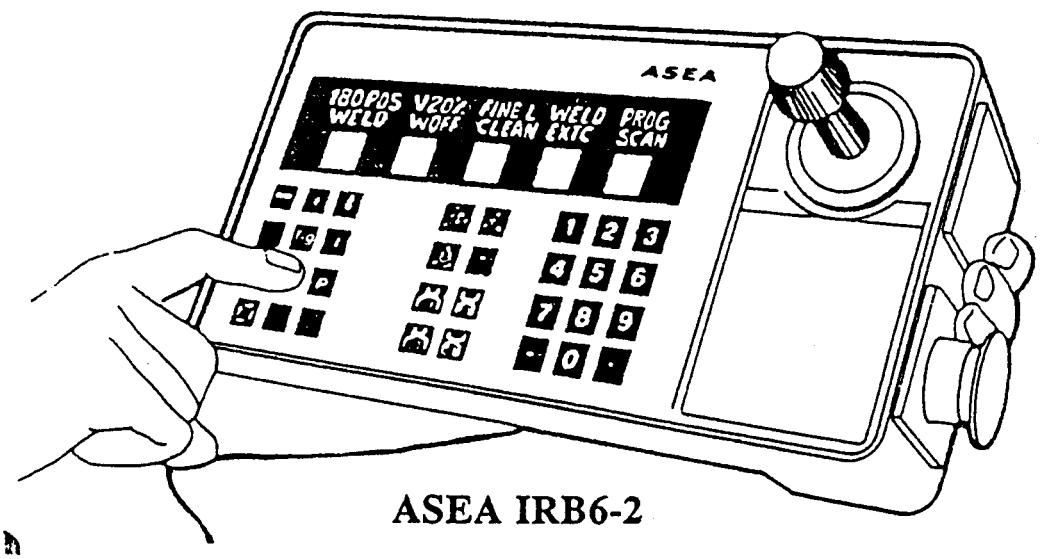
Robots

Boitiers d'apprentissage



BOITIER DE COMMANDE ET DE TELEMANIPULATION

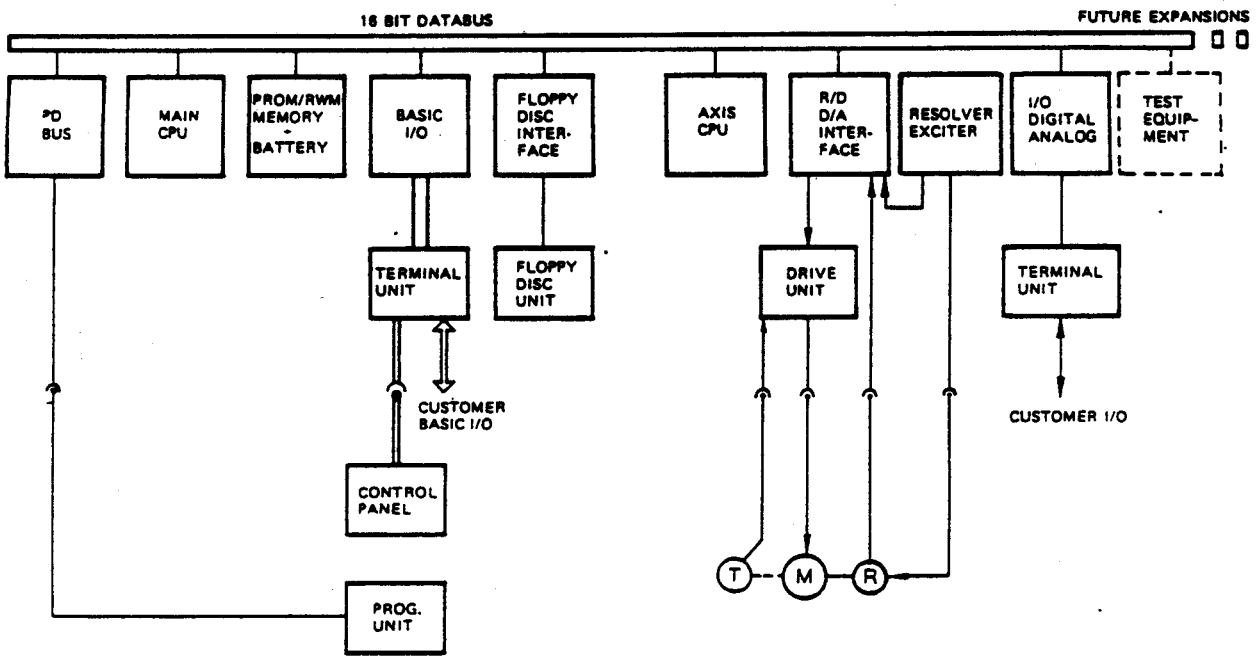
COMMERCY CY800



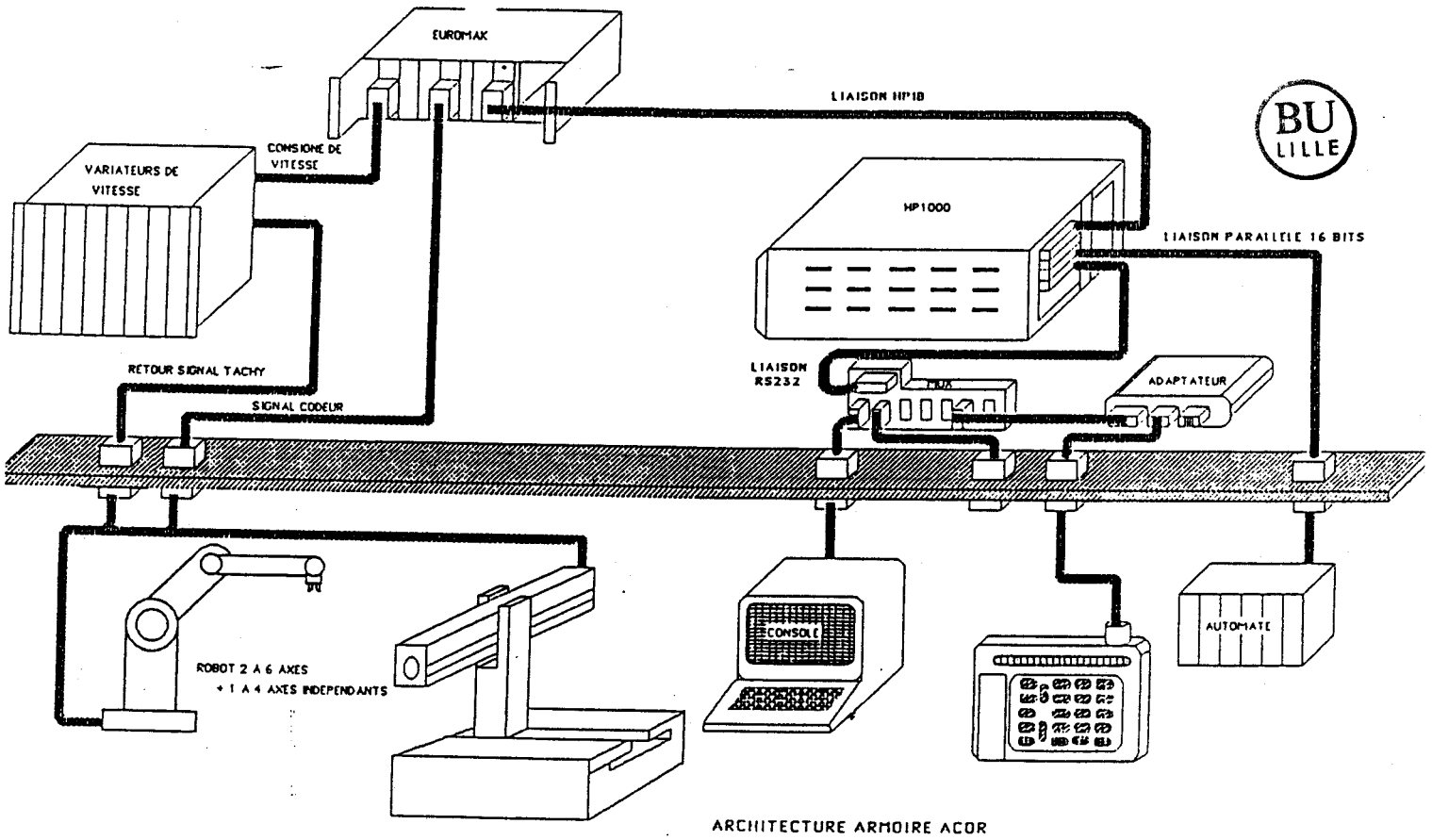
ASEA IRB6-2

Robots

Commandes



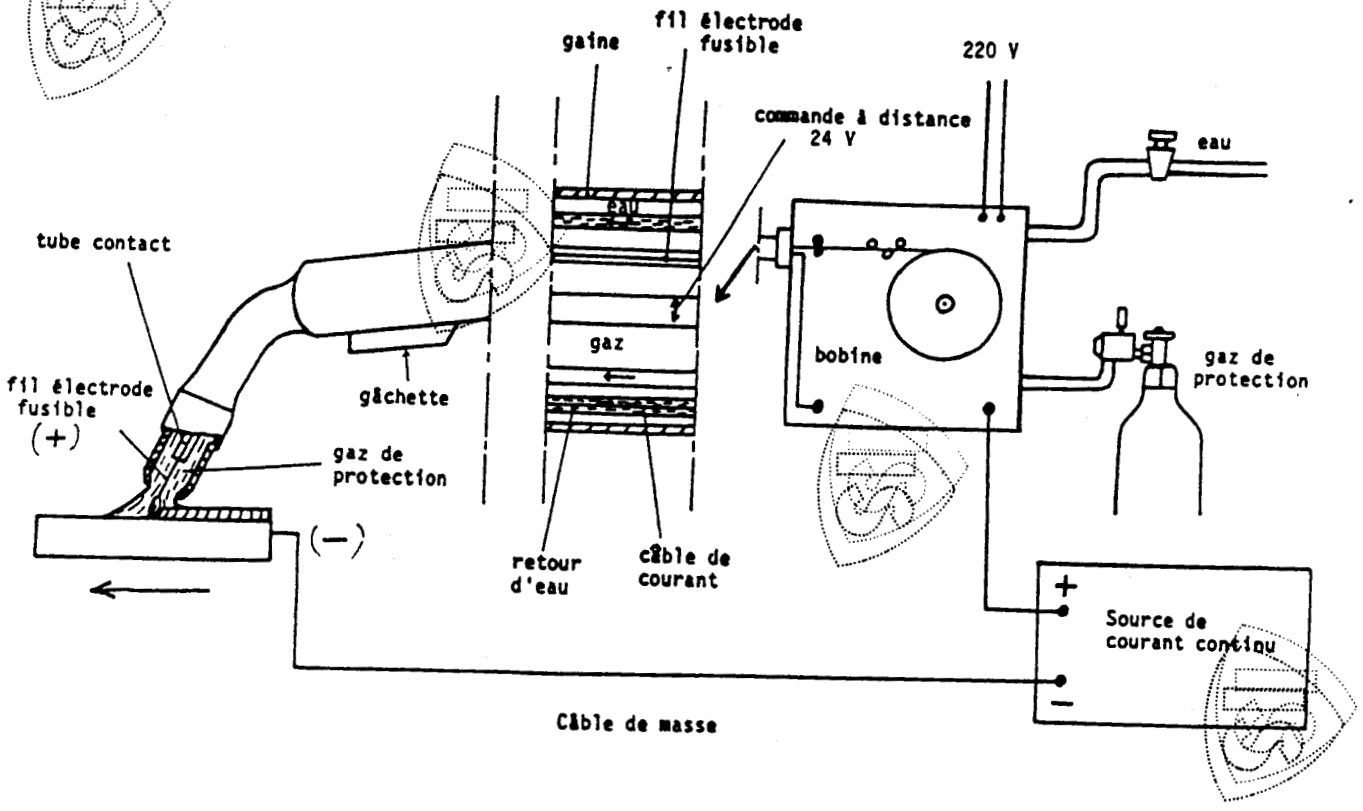
ASEA IRB6-2



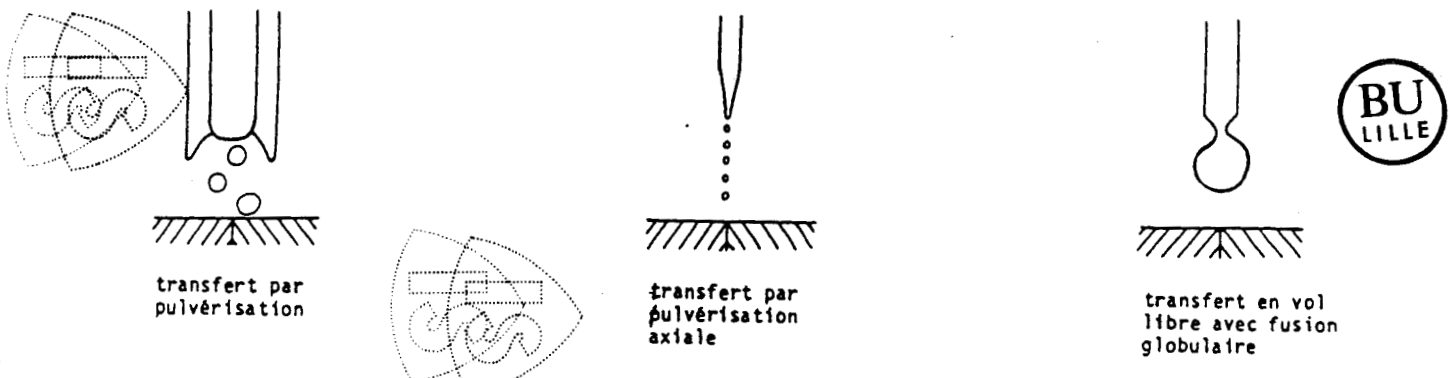
COMMERCY CY800

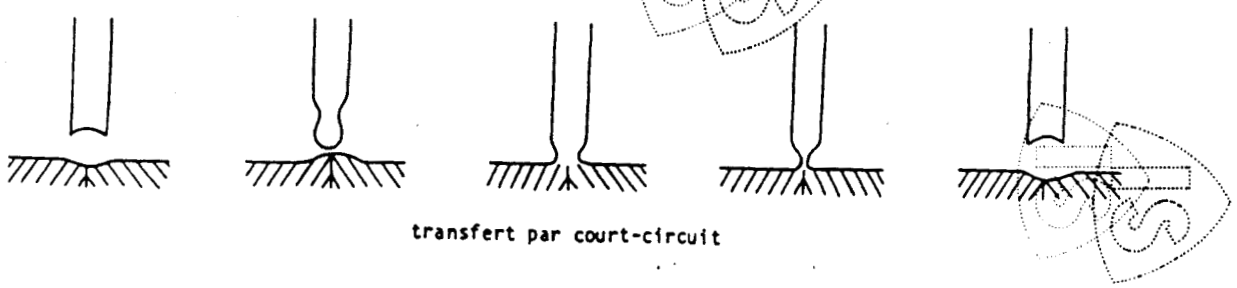
SOUDAGE

SCHEMA D'UNE INSTALLATION MIG OU MAG

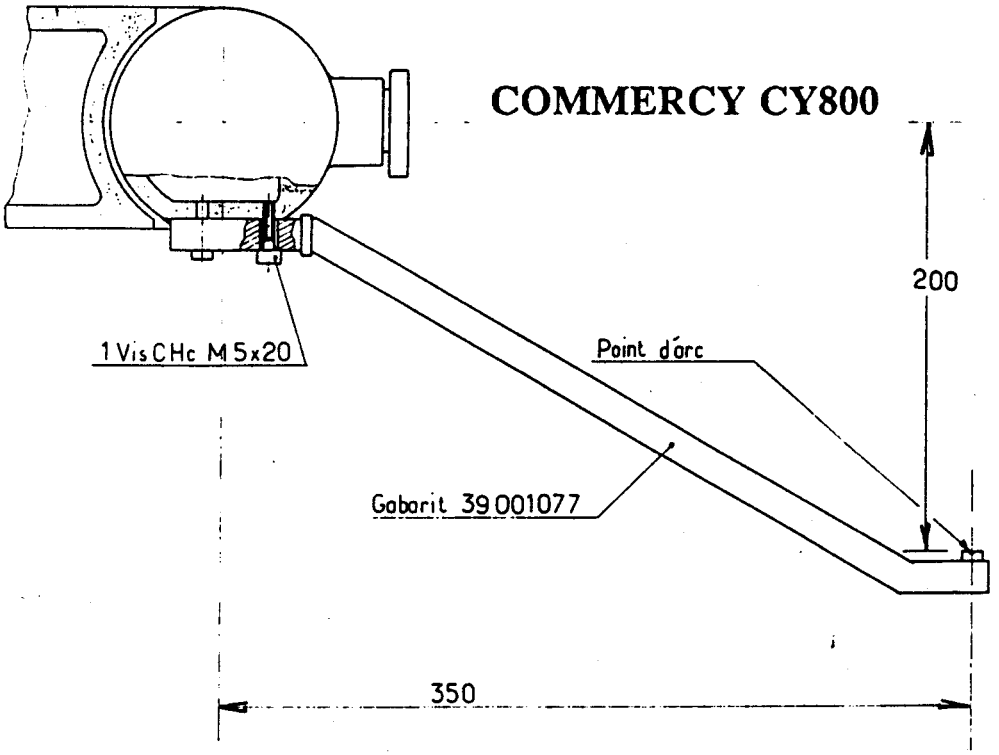


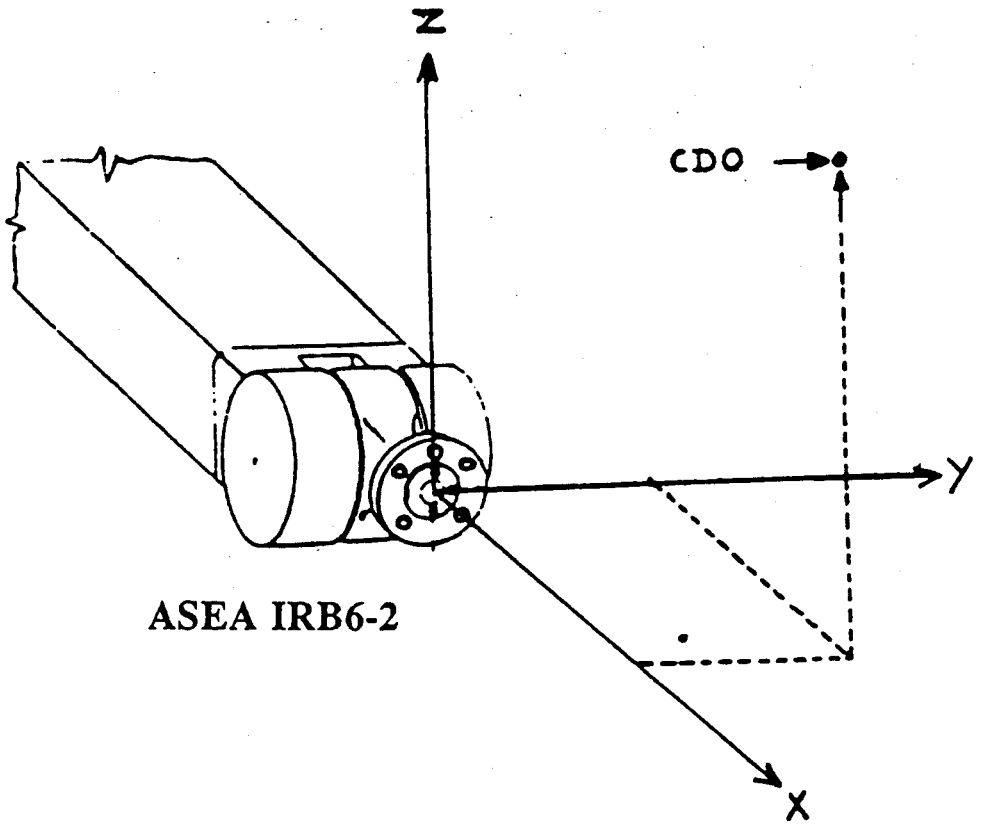
DIFFERENTS MODES DE TRANSFERT DU METAL DANS L'ARC





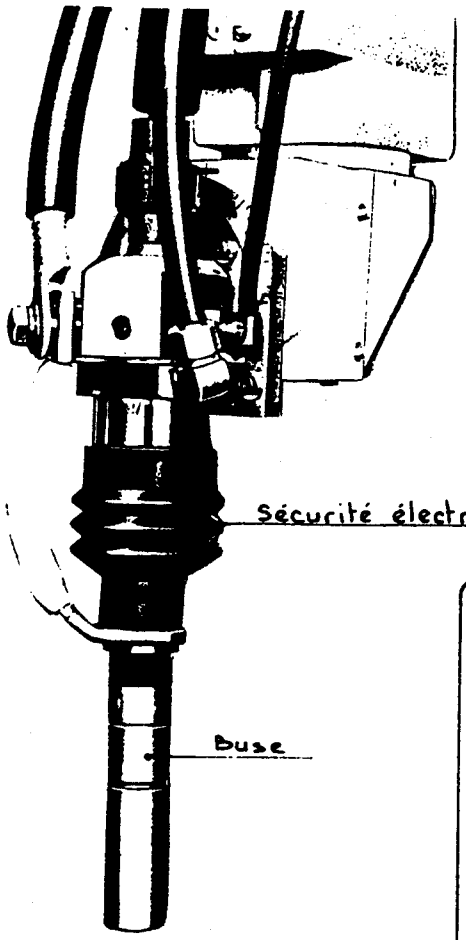
Robots
Centres d'outil



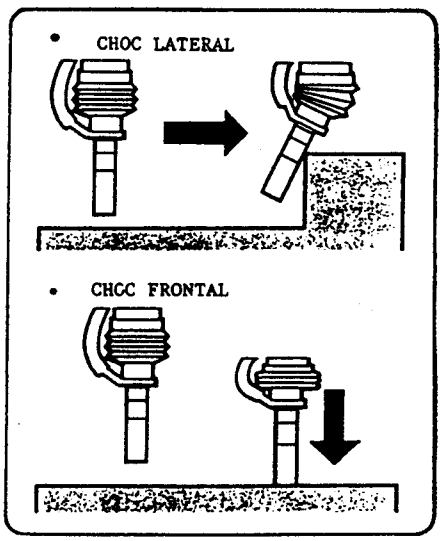


ASEA IRB6-2

TORCHE ANTI-COLLISION



- MIG - MAG
- 350 Ampères
- ⌀ fil : 0,8 à 1,6
- Refroidissement naturel
- Poids : 600gr.



ALSTHOM ATLANTIQUE

acb

ANNEXE 4

INVERSION DE ROBOTS 5 AXES.

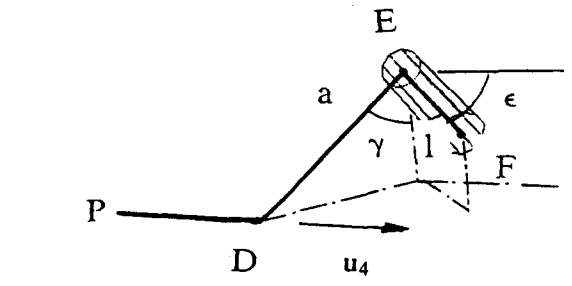
Inversion d'un robot 5 axes plan

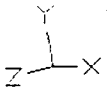
L'inversion du robot va se décomposer en trois parties: la recherche du point invariant du poignet à partir de la position et de l'orientation de la torche, l'inversion des trois premiers axes du robot, détermina tion des angles du poignet.

Nous définirons d'abord la géométrie de la torche que nous avons choisie la plus générale possible.

I. Géométrie de la torche.

La torche est inclinée par rapport au dernier axe de rotation du robot. Elle est aussi décalée par rapport à cet axe. Cela nous donne la géométrie suivante:





géométrie de la torche.

$$ED \cdot EF = a\ l \cos\gamma = A \quad (1)$$
$$|ED|^2 = a^2 \quad (2)$$
$$\mathbf{u}_4 \cdot EF = |\mathbf{u}_4|\ l \sin\epsilon = |\mathbf{u}_4| B \quad (3)$$
$$ED \cdot \mathbf{u}_4 = 0 \quad (4)$$

II. géométrie du robot.

Le fait que le robot soit un robot plan impose que l'axe $\mathbf{u}_4$ soit toujours dans son plan vertical. Cela peut se traduire par:

$\mathbf{u}_4 = OD + \beta \mathbf{z}$ *Avec* $\mathbf{z}$ *l'axe vertical du repère de base.* (5)

III. Recherche du point invariant.

Le point invariant peut être calculé à partir de la connaissance de OD et $\mathbf{u}_4$.

a. calcul de β

$$(5) \rightarrow |\mathbf{u_4}|^2 = OD^2 + \beta^2\mathbf{z}^2 + \beta OD \cdot \mathbf{z} \quad (6)$$
$$(2)\ et\ (6) \rightarrow |\mathbf{u_4}|^2 = a^2 - OE^2 + 2OD \cdot OE + \beta^2 \mathbf{z}^2 + 2\beta OD \cdot \mathbf{z} \quad (7)$$

On cherche à éliminer $OD \cdot OE$ et $OD \cdot \mathbf{z}$ qui sont inconnus.

$$(4)\ et\ (5) \rightarrow (OD + \beta\mathbf{z}) \cdot ED = 0$$
$$soit\ -OE^2 + OE \cdot OD + ED^2 + \beta EO \cdot \mathbf{z} + \beta OD \cdot \mathbf{z} = 0 \quad (8)$$
$$(7) - 2(8) -> |\mathbf{u_4}|^2 = a^2 + OE^2 - 2ED^2 + \beta^2\mathbf{z}^2 - 2\beta\mathbf{z} \cdot EO \quad (9)$$

On cherche $\mathbf{u_4}$ indépendamment de OD pour trouver β.

$$(3) -> |\mathbf{u_4}| = \frac{1}{B}(\mathbf{u_4} \cdot EF)$$
$$= \frac{1}{B}(OD + \beta\mathbf{z}) \cdot EF$$
$$= \frac{1}{B}(OE + ED + \beta\mathbf{z}) \cdot EF$$
$$|\mathbf{u_4}| = \frac{1}{B}(OE \cdot EF + A + \beta\mathbf{z} \cdot EF) \quad (10)$$

On pose:

$$b = [OE \cdot EF + A]$$
$$c = \mathbf{z} \cdot EF$$
$$d = B\ OE \cdot \mathbf{z}$$
$$e = B(OE^2 - a^2)$$

$$(9)\ et\ (10)^2 -> (c\beta^2 + b) = e + 2d\beta + \beta^2\mathbf{z}^2$$
$$(c^2 - \mathbf{z}^2)\beta^2 + (2cb - 2d)\beta + (b^2 - e) = 0$$
$$\Delta' = (cb - d)^2 + (b^2 - e)(c^2 - \mathbf{z}^2)$$

$$\beta_1^2 = \frac{-(cb - d) + -\sqrt{\Delta'}}{(c^2 - \mathbf{z}^2)}$$

avec $\mathbf{z}^2 = 1$.

b. norme de $\mathbf{u_4}$.

$$|\mathbf{u_4}|^2 = -a^2 + OE^2 + \beta^2\mathbf{z}^2 - 2\beta(\mathbf{z} \cdot EO)$$

c. calcul de OD.

$$(5) -> OD^2 = |\mathbf{u_4}|^2 - \beta^2\mathbf{z}^2 - 2\beta\mathbf{z} \cdot OD$$

on pose:

$$f = |\mathbf{u_4}|^2 - \beta^2 = (OE^2 - a^2)$$
$$on\ appelle\ y = \mathbf{z} \cdot OD$$

On cherche à exprimer OD en fonction de y.

$$(5) \rightarrow OD \cdot EF = \mathbf{u}_4 \cdot EF - \beta \mathbf{z} \cdot EF = g \quad (11)$$
$$(8) \rightarrow OD \cdot OE = OE^2 - ED^2 - \beta EO \cdot \mathbf{z} - \beta \mathbf{z} \cdot OD = h - \beta y \quad (12)$$
$$(11) \text{ et } (12) \rightarrow OD \times (EF \times OE) = -gOE + (h - \beta y)EF \quad (13)$$

Pour faire apparaitre OD et $\mathbf{z} \cdot OD$:

$$\mathbf{z} \times [OD \times (EF \times OE)] = [\mathbf{z} \cdot (EF \times OE)]OD - y(EF \times OE)$$
$$= (h - \beta y)\mathbf{z} \times EF - g(\mathbf{z} \times OE)$$

Ce que l'on peut écrire:

$$i\ OD = H - y\ G$$

Si i différent de zéro:

$$OD^2 = \frac{1}{i^2}[H^2 + y^2G^2 - 2H \cdot Gy]$$
$$= f - 2\beta y$$

$$y^2 \frac{G^2}{i^2} + 2y(\beta - \frac{H \cdot G}{i^2}] + (H^2 - f) = 0$$

Que l'on pose:

$$y^2 j + 2y\ k + m = 0$$
$$\Delta' = k^2 - jm$$

$$\boxed{y_1^2 = \frac{-k + -\sqrt{\Delta'}}{j}}$$

On calcule y pour chaque valeur de β et l'on obtient quatres valeurs possibles pour OD

$$\boxed{OD = \frac{1}{i}(H - yG)}$$

d. point invariant.

Le point invariant est donné par $OP = OD - e_l \frac{\mathbf{u}_4}{|\mathbf{u}_4|}$ avec $\mathbf{u}_4 = OD + \beta \mathbf{z}$.

IV. Invertion du porteur.

L'inversion du porteur va consister à calculer les coordonnée articulaires pour atteindre le point invariant P.

On dispose pour cela des longueurs des différents bras données par le modèle de celui-ci. Le schéma ci dessous explicite les notations choisies pour le type de structure des deux robots ASEA et COMMERCY.

structure du robot

Les angles $\theta_1,\theta_2,\theta_3$, constituent les variables articulaires du porteur que nous devons trouver. θ_4, θ_5, sont les coordonnées articulaires du poignet.

a. Calcul de θ_1.

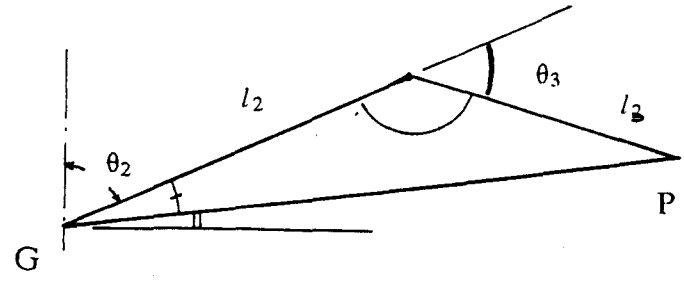
θ_1 est donné directement par la projection de P dans le plan horizontal, car P appartient au plan du robot qui contient l'axe de rotation vertical.

$$\cos\theta_1 = \frac{(OP \times \mathbf{z})\cdot \mathbf{x}}{|OP \times \mathbf{z}|} \qquad \sin\theta_1 = \frac{(OP \times \mathbf{z})\cdot \mathbf{y}}{|OP \times \mathbf{z}|}$$

$$\theta_1 = arctan\frac{\sin\theta_1}{\cos\theta_1}$$

b. Calcul de θ_2 θ_3 .

Ce calcul se ramène à la détermination des angles d'un triangle dont on connait la longueur des trois cotés.



deuxième et troisième axe.

On pose: $GP = OP - l_1\mathbf{z}$.

$$\theta_2 = -\frac{\Pi}{2} + arctan\left(\frac{(GP\cdot \mathbf{z})}{|OP \times \mathbf{z}|}\right) + arcos\left(\frac{l_2^2 + GP^2 - l_3^2}{2\, l_2\, GP}\right)$$

$$\theta_3 = -\Pi + arcos\left(\frac{l_2^2 + l_3^2 - GP^2}{2\, l_2\, l_3}\right)$$

Les arc cosinus dans le calcul de θ_2 et de θ_3 donnent deux solutions qui correspondent aux positions " coude haut "," coude bas " du porteur. Dans les cas des deux robots choisis, seule la solution coude haut est valable.

V. coordonnées articulaires du poignet.

$$\theta_4 = \frac{\Pi}{2} - \theta_2 - \theta_3 - arctan\left(\frac{\mathbf{u_4}\cdot\mathbf{z}}{|\mathbf{u_4}\times\mathbf{z}|}\right)$$

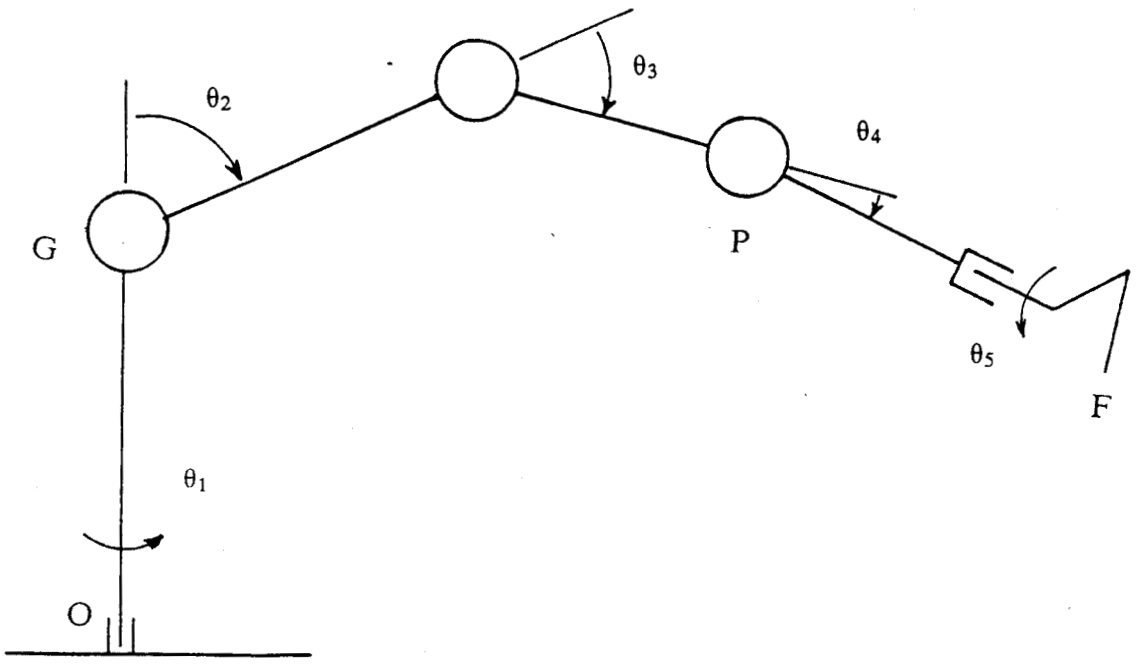
Le calcul de θ_5 impose que l'on prenne une référence pour la rotation. Cette réference est le plan vertical du robot. On définit donc un vecteur $\mathbf{w_1}$ tel que:

$$\mathbf{w_1} = \frac{\mathbf{u_4}\times\mathbf{z}}{|\mathbf{u_4}\times\mathbf{z}|} \qquad si\ |\mathbf{u_4}\times\mathbf{z}| \neq 0$$

$$\mathbf{w_1} = \frac{\mathbf{u_4}\times\overrightarrow{OP}}{|\mathbf{u_4}\times\overrightarrow{OP}|} \qquad si\ non.$$

$$\theta_5 = arctan\left(\frac{(\mathbf{u_4}\times\mathbf{w_1})\cdot\overrightarrow{DE}}{(\mathbf{w_1}\cdot\overrightarrow{DE})\ |\mathbf{u_4}|}\right) + \theta_{5_0}$$

Où θ_{5_0} est l'angle entre DE est l'axe y du repère lié à l'outil.

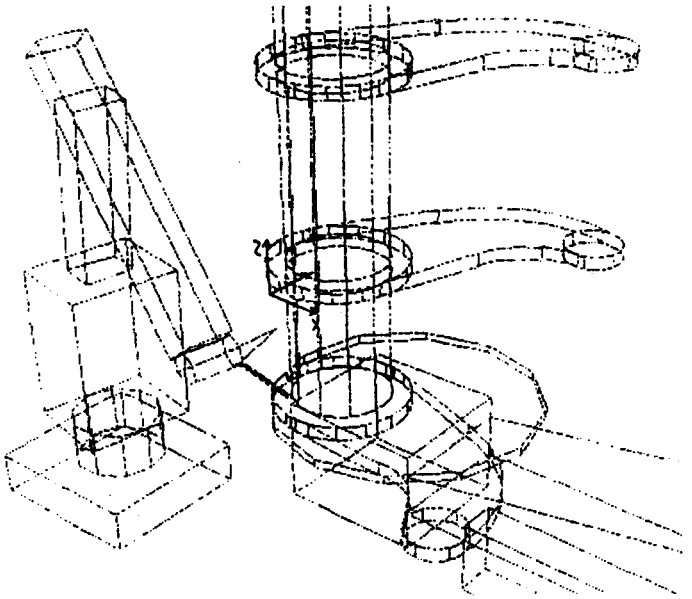


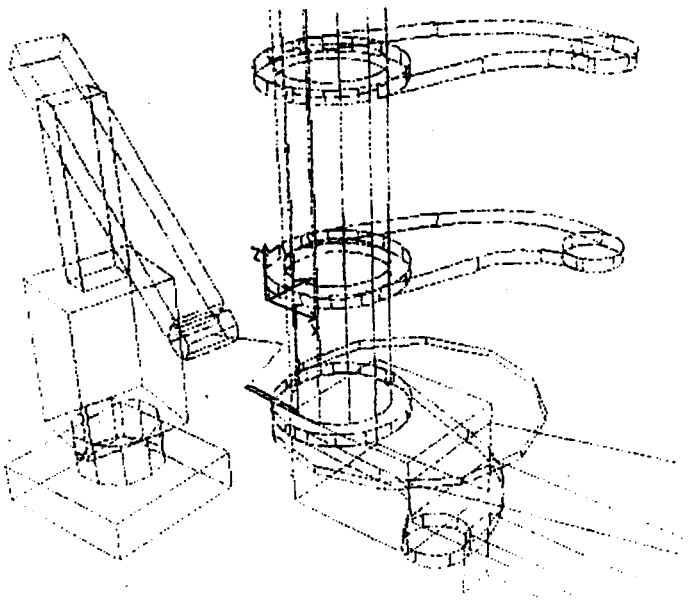
ANNEXE 5

PREMIERE REALISATION.

Animation E.C.L.

TRAJECTOIRE : deux points de soudure

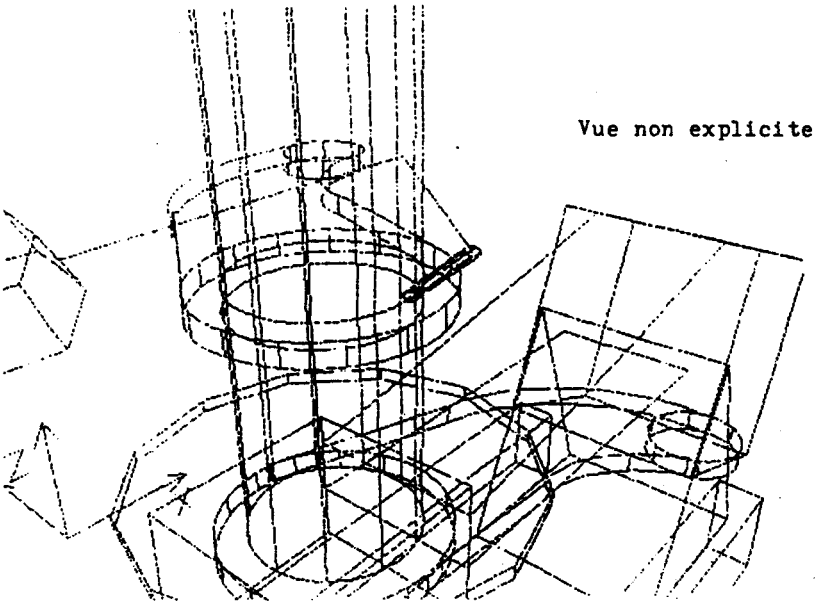


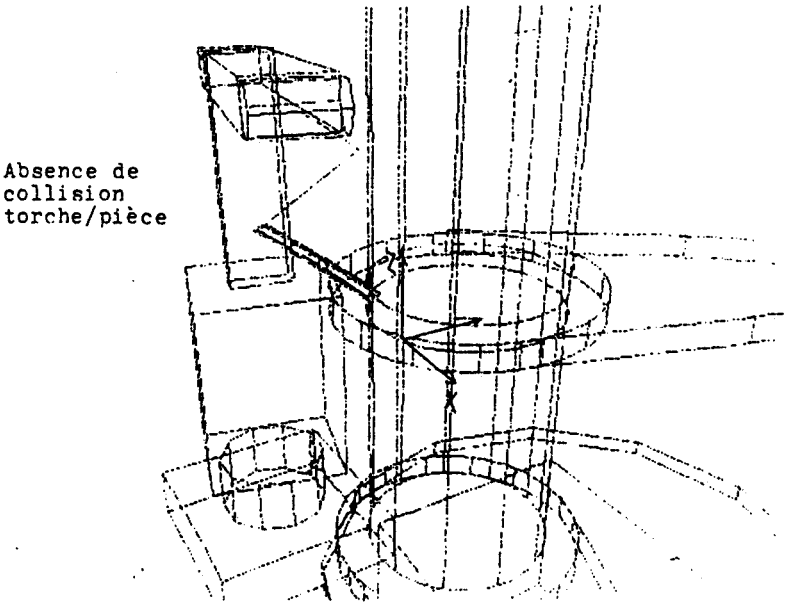


Premiere réalisation.

Animation E.C.L.

VISUALISATION OPTIMALE POUR DETECTER LES COLLISIONS

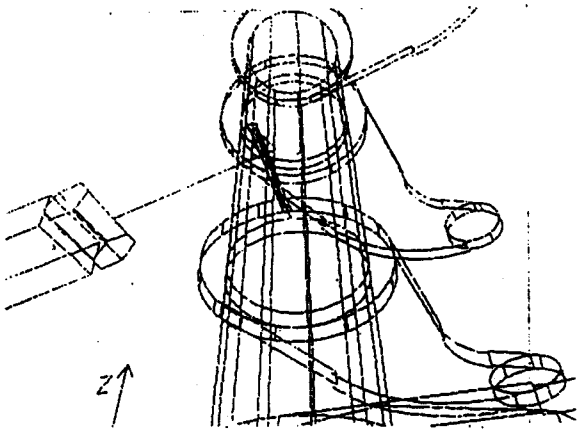




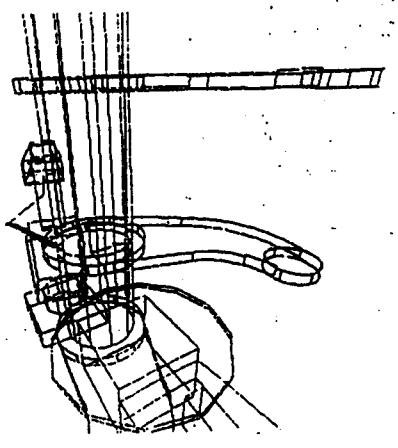
Premiere réalisation.

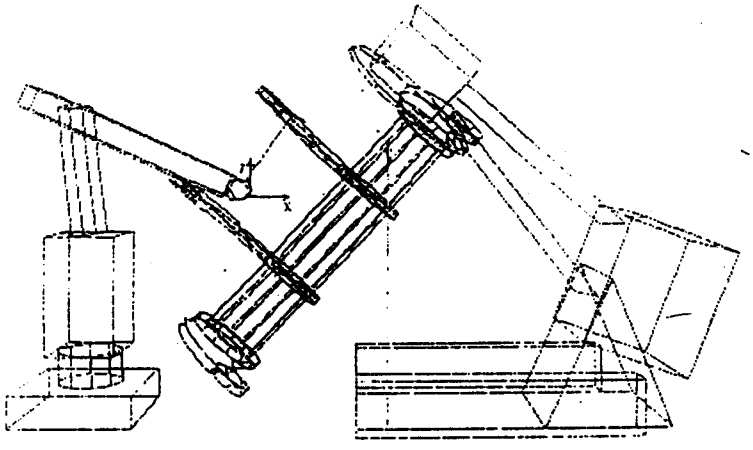
Animation E.C.L.

Normale au plan de vue = a.SEC + b.DIR

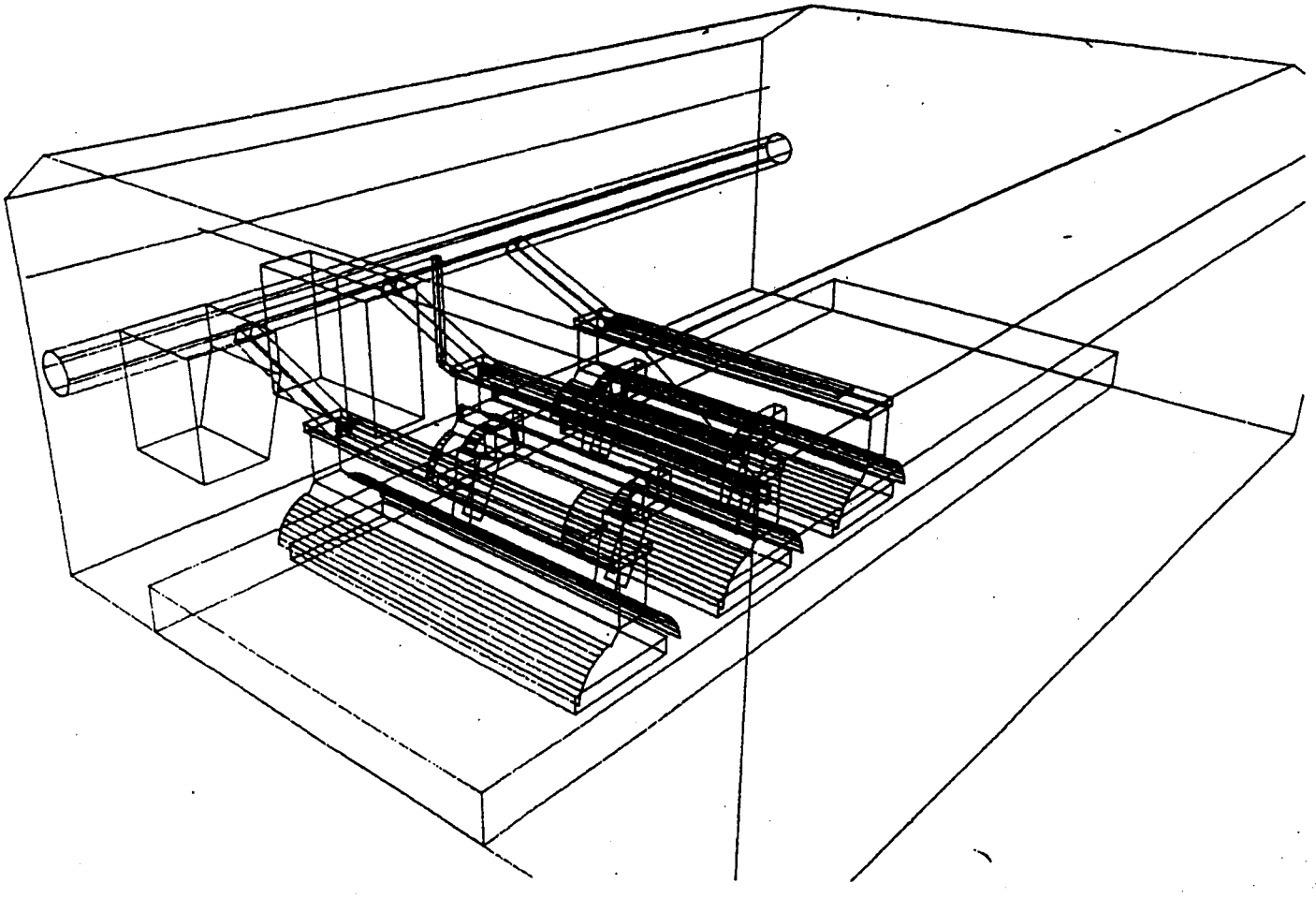


Changement de coefficient par rapport à la vue précédente (plus explicité)





Premiere réalisation.



Premiere réalisation.

ANNEXE 6

MODELISATION SOLIDE.

objets de référence.

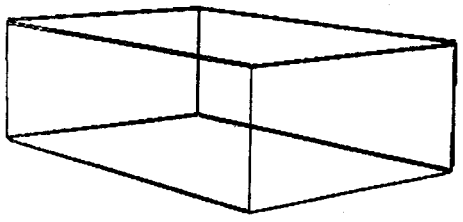


figure 1: parallélépidède

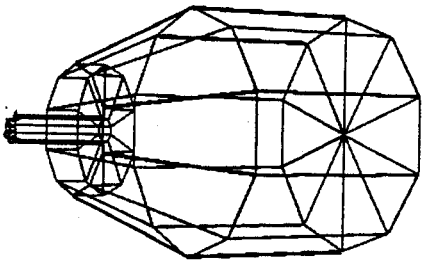


figure 2: torche

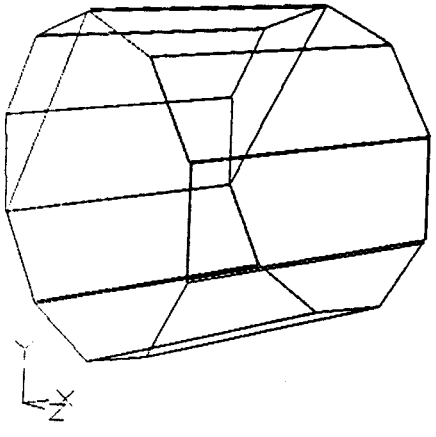


figure 3: cylindre

manipulation d'objet.
parrallélépipède union parrallélépipède
tourné de 90 degrés

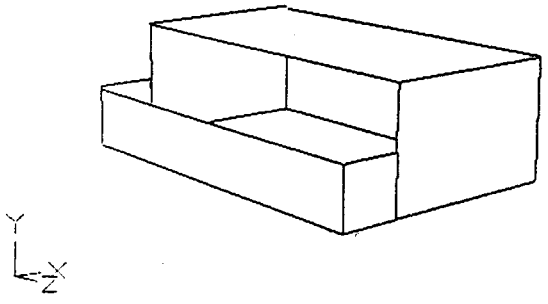


figure 5: différence 1 par 2

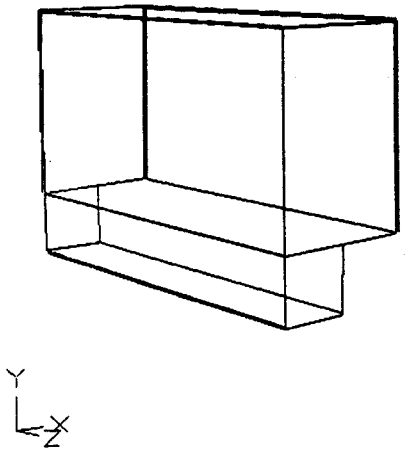


figure 6: différence 2 par 1

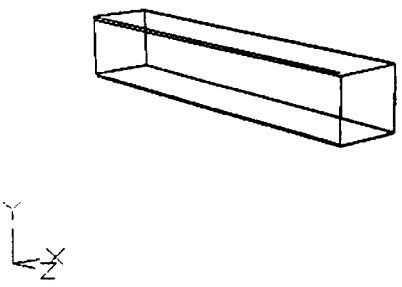


figure 7: intersection 1 et 2

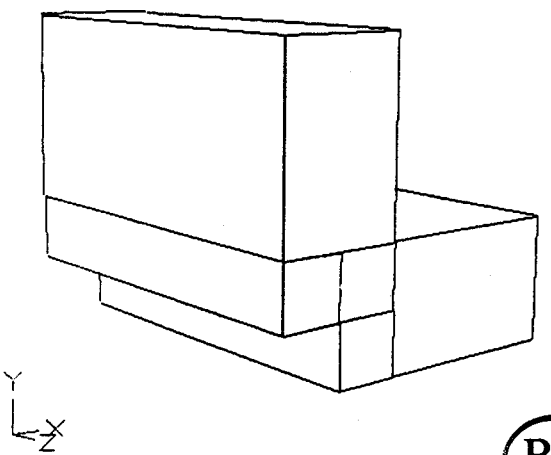


figure 8: union 1 et 2

manipulation d'objet.
parrallélépipède union cylindre
tourné de 45 degrés

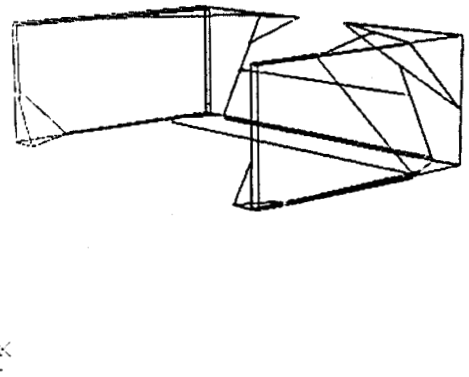


figure 9: différence 1 par 2

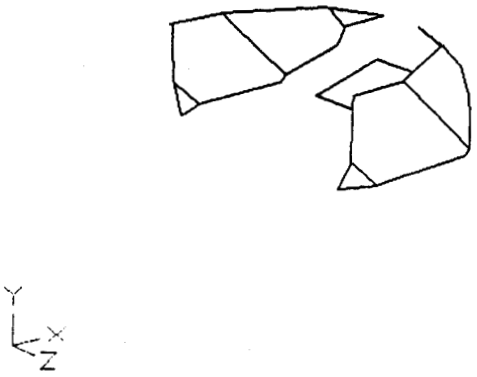


figure 10: intersection 2 par 1

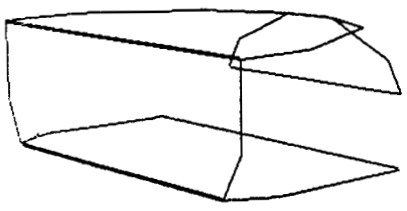


figure 11: intersection 1 par 2

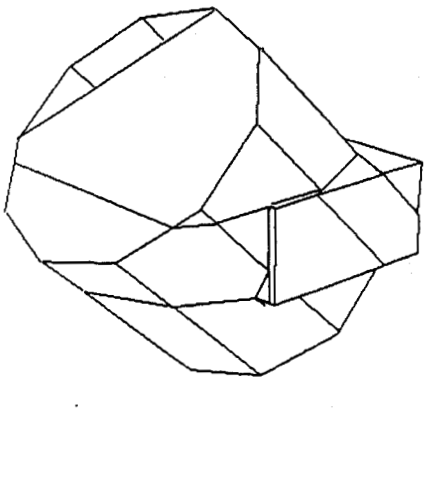
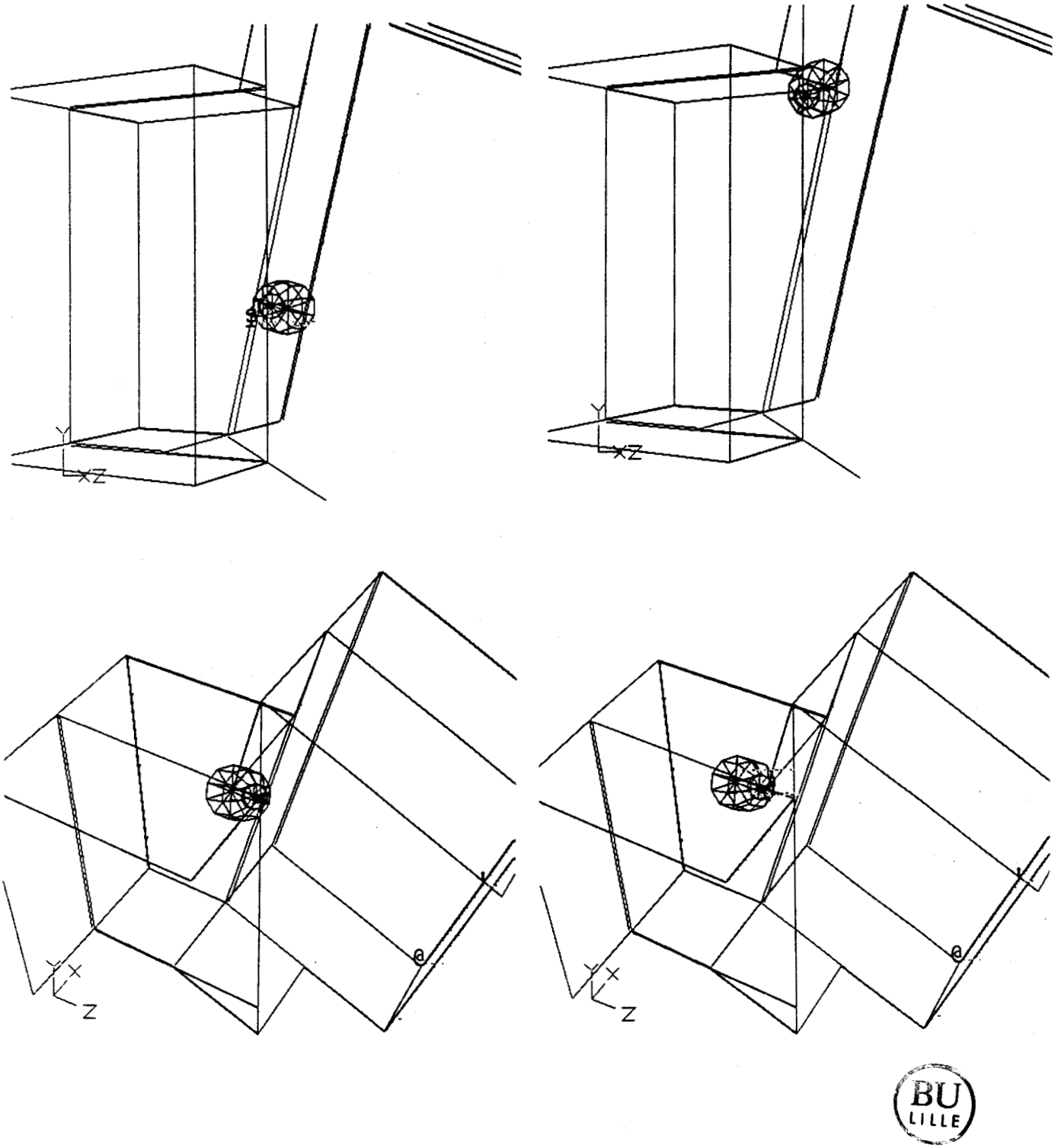


figure 12: union 1 et 2

ANNEXE 7

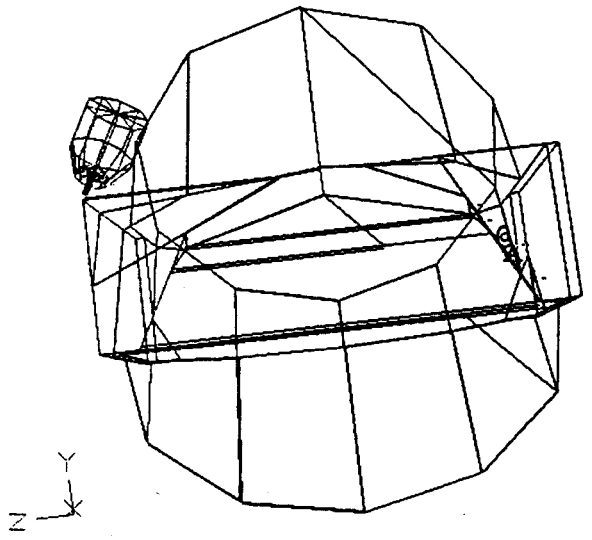
ANIMATION SOUS MOVIE.

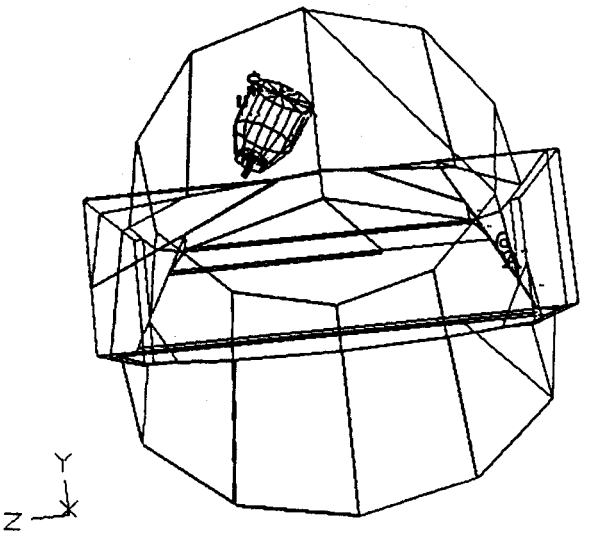
Animation sous movie.

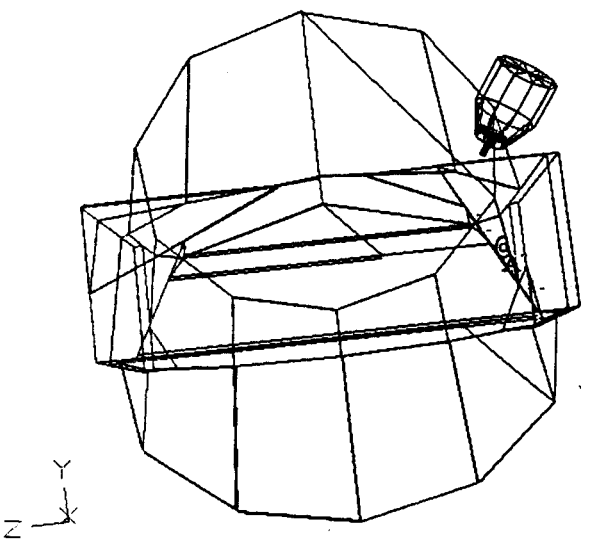


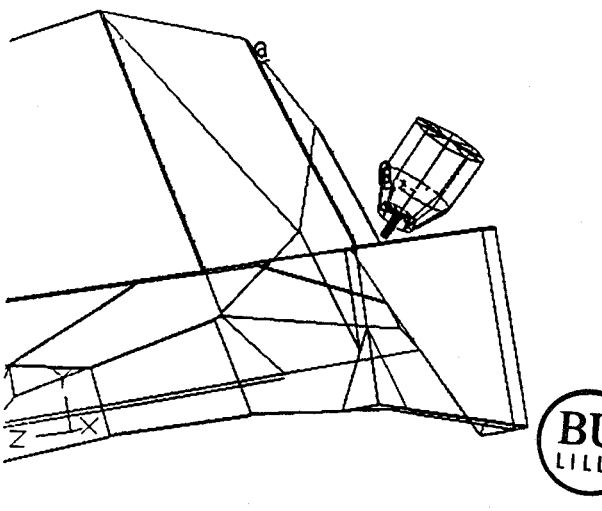
Animation.

Animation sous movie.



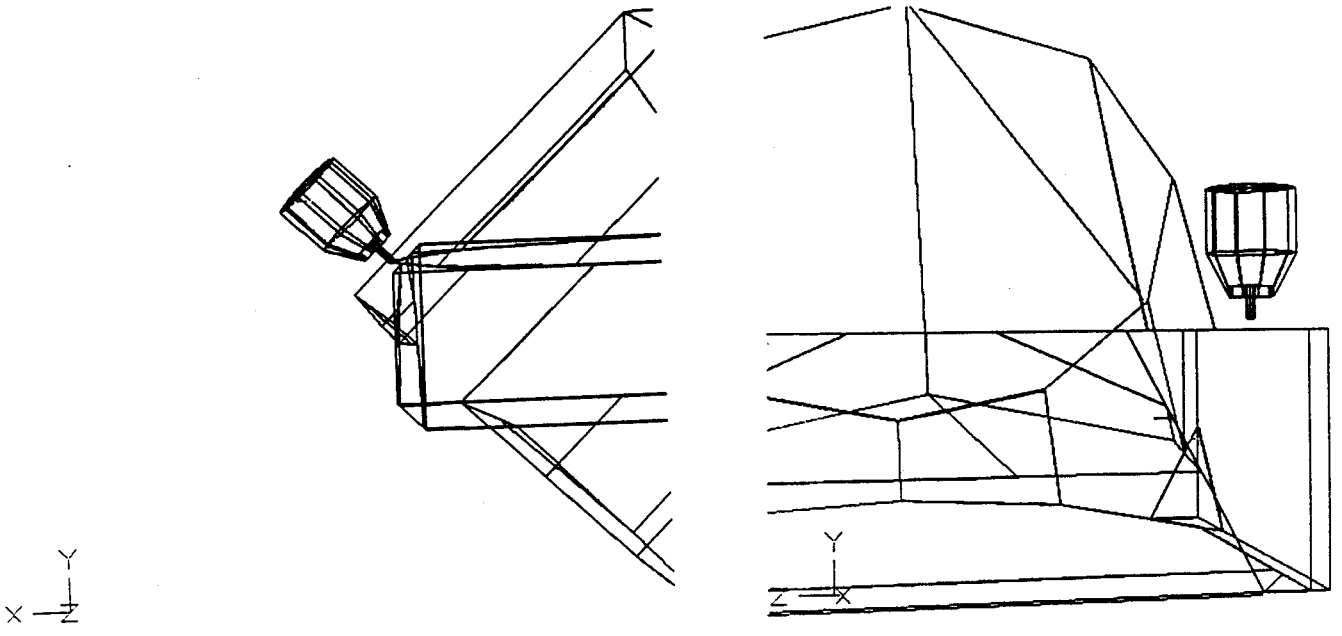


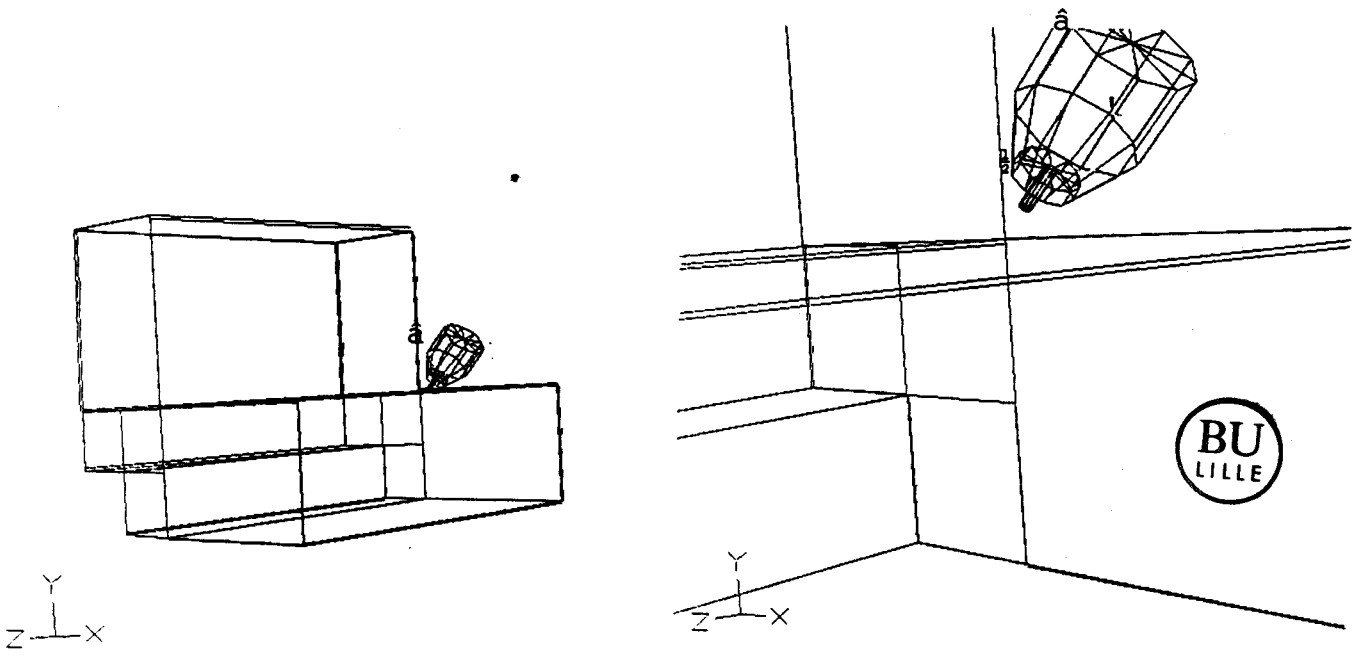




Animation.

Animation sous movie.





Animation.

ANNEXE 8

MANIPULATION DES DONNEES.

Structure des données
du
poste de programmation

applicat :	nom	créateur	date	programme	manipulat	cellule
programme :	nom	créateur	date	pièce	outil	
cellule :	nom	créateur	date	environnem	[chaine]	
environnem:	nom	créateur	date			
chaine :	robot	manipulat | robot	position			
robot :	nom	créateur	date	cinematiq		
cinematiq :	algorithm	nbr_axe	[axe]			
manipulat :	nom	créateur	date	nbre_axe	[axe]	
axe :	/a					
nbre_axe :	/i/1/6/					
position :	/f	/f	/f			
outil :	nom	créateur	date			
piece :	nom	créateur	date	[ss-ensembl]		
ss-ensembl:	nom	créateur	date			
créateur :/c						
date :/d						

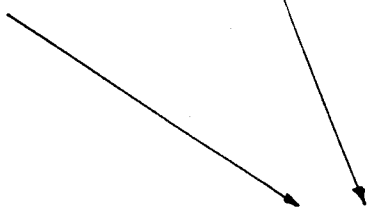
Manipulation des données

Description de la base.

```
cellule   :nom date createur [robot]
robot     :nom date createur
date      :/d/
createur  :/x/
```

base cellule.

```
atelier 87-09-01 jpc scara puma .
bidon 86-03-31 JPC commercy .
criif 87-01-20 osorio TH8 .
icam 87-01-09 jpg irb6-2 puma .
isen 87-02-28 moi irb6-2 .
limsi 86-07-08 jpc irb6-2 .
```



base robot.

```
IRB90 86-02-28 pva
TH8 81-09-20 osorio
commercy 86-01-20 jpc
irb6-2 86-03-31 jpc
puma 87-03-31 JPc
scara 87-09-01 jpc
toto 87-04-02 jpc
```