# THESE

Présentée à

# L'Ecole des Mines de Paris

Pour obtenir

Le Titre de Docteur

en

INFORMATIQUE TEMPS RÉEL - ROBOTIQUE - AUTOMATIQUE

par

ROGER PISSARD-GIBOLLET

# Conception et Commande par Asservissement Visuel d'un Robot Mobile

Soutenue le 3 Décembre 1993 devant la commission d'examen :

| | | | |
|---|---|---|---|
| Messieurs | Claude | Laurgeau | Président |
| | Emmanuel | Mazer | Rapporteur |
| | Thierry | Viéville | Rapporteur |
| Madame | Marie-Claude | Thomas | Examinateurs |
| Messieurs | Rachid | Alami | |
| | Jean-Charles | Bonin | |
| | Richard | Etéki | |
| | Patrick | Rives | |

# Remerciements


Mes remerciements vont à :

> Claude Laurgeau, professeur à l'Ecole des Mines de Paris, qui me fait l'honneur de présider ce jury,
>
> Emmanuel Mazer, chargé de recherche du CNRS au LIFIA à Grenoble, et Thierry Vieville, chargé de recherche à l'INRIA Sophia-Antipolis, pour avoir accepté d'être les rapporteurs de ce travail,
>
> Rachid Alami, chargé de recherche du CNRS au LAAS à Toulouse, pour avoir bien voulu participer à ce jury.

Je tiens à exprimer ma gratitude, sans les dissocier, à Jean-Charles Bonin, maître-assistant à l'ISIA, à Patrick Rives, chargé de recherche à l'INRIA Sophia-Antipolis et à Bernard Espiau, directeur de recherche à l'INRIA Sophia-Antipolis, qui m'ont chacun beaucoup apporté.

Je remercie également Marie-Claude Thomas, professeur à l'ESSI, pour m'avoir si bien aiguillé et aidé à la sortie de l'école.

Je tiens à remercier tout le personnel de la société ECA qui a participé au projet plus particulièrement Richard Eteki, ingénieur recherche et développement, Alain Thouvenin, directeur du service recherche et développement, et Gérard Falcou, sous-directeur études.

Merci à Jean-Jacques Borrelly, ingénieur de recherche à l'INRIA de Sophia-Antipolis, d'avoir mis à ma disposition les cartes de traitement d'images mais surtout sa compétence et son aide pour la mise au point du système expérimental.

Merci à Konstantinos Kapellos, doctorant à l'ISIA, pour notre collaboration fructueuse qui m'a permis de disposer d'un environnement logiciel de tout premier ordre.

Finalement je remercie tout les membres de l'ISIA et de l'équipe ICARE de l'INRIA qui m'ont apporté leur soutien.

# Résumé

Les techniques, désormais classiques, d'asservissement visuel, sont utilisées pour la commande de bras manipulateurs ou la vision active mais il n'existait pas encore d'applications en robotique mobile.

Le problème principal de l'utilisation de ces techniques pour un robot mobile est dû à la présence de liaisons mécaniques non-holonomes qui limitent les mouvements du robot. Nous proposons une approche originale et générale utilisant la coopération des mouvements d'une base mobile muni d'un bras manipulateur qui porte une caméra. Le fait de contrôler les mouvements de la caméra plutôt que ceux de la base permet de contourner les limitations des contraintes non-holonomes en considérant l'ensemble mécanique base et bras. Cette modélisation permet d'utiliser le formalisme de la commande référencée vision destinée en premier lieu aux bras manipulateurs rigides.

En utilisant la notion de liaisons virtuelles, nous énumérons et illustrons en simulation un ensemble de tâches élémentaires référencées vision destiné à guider un robot mobile placé dans un environnement structuré d'intérieur de "type bureau". Pour la réalisation de tâches robotiques plus complexes composées d'un enchaînement de tâches élémentaires, nous donnons un cadre formel pour gérer les problèmes de transition entre tâches référencées vision.

Pour une meilleure validation de ces travaux, nous désirions maîtriser toute la chaîne : des spécifications à l'expérimentation. Nous avons participé à la conception et à la réalisation d'un robot mobile ; nous avons adapté un environnement de programmation (MINISIM, ORCCAD) qui puisse associer à une loi d'asservissement un comportement logique pour la prise en compte des événements.

Finalement nous proposons les résultats expérimentaux validant toute cette approche sur une tâche de positionnement vis à vis d'un cône.

# Abstract


Visual servoing technics are commonly used for manipulators control or active vision. However, they have never been applied yet to control mobile robots.

The main problem to the use of such technics for mobile robots is linked to the non holonomic mechanical constraints which restrict the robot displacements. We propose a general and original approach based on the cooperation between a mobile robot and an eye-in-hand manipulator system attached to the mobile base. Controling the motions of the camera rather than the motions of the cart, allows to disregard the limitations due to the non holonomic constraints. Thanks to this modeling, the use of the vision based control primarly dedicated to rigid manipulators can be successfully extended and applied to mobile systems.

Using the notion of virtual link, we mention and illustrate through simulations, a whole set of elementary visual based tasks dedicated to mobile robot guidance in an office-like structured environment.

In order to perform more complex robotic tasks made of a sequenced set of elementary tasks, we define a formalizing approach in order to manage the transitions between vision based tasks.

For a better validation of this work, we wanted to master the whole development system from specifications to experimentation. Therefore, we took part to the conception and hardware realization of the mobile robot; we also developped and adapted a programming environment (MINISIM, ORCCAD). It is now possible, in this environment, to associate a local behaviour to a servoing loop in order to take into account some events relevant to the control laws execution.

Finally, in order to validate the overall approach, we implemented a simple example of task-positionning in front of a target cone. We provide the reader with the corresponding experimental results.

# Resumi [1]


Li teinico desenant classico d'asservimen visuau soun utilisado pèr la coumando de bras manipulaire vo la visioun ativo, mai noun esistavo encaro d'aplicacioun pèr la rouboutico móuvedisso.

Lou maje proubleme de l'usanço d'aquèli teinico pèr un automato móuvedis es la presènci de liesoun mecanico "non-holonomes" que restregnon li mouvamen d'aquelo eisino, prepausan un aproche ouriginau e tambèu generau gaubejant la cououperacioun di mouvamen d'uno baso moblo alestido d'un bras manipulaire pourtaire de la camera. Lou fa de countroula li mouvamen de la camera, pulèu qu'aqéu de la baso douno la poussibleta de trevira la limitacioun di countrencho "non-holonomes" en counsiderant l'ensen mecanico de baso e de bras. L'establimen de moudele fourmalisa permet alor d'utilisa lou fourmalisme de la coumando referido visioun destinado d'en premié: bras manipulaire rege.

En utilisant la noucioun de liesoun virtualo enumerant e enlusissen en simulacioun un ensèu d'obro elementàri referido visioun destina à mena un automate mouvedis bouta dins un enroudamen dòu dintre de "type bureau". Pèr realisa d"obro "robotiques" mai coumpleisso facho d'un encadenamen d'obro elementàri dounan un encastre fourmau pèr geri li proubleme de transicioun entre lis obro referido visioun.

Pèr uno mai bono validacioun d'aquèli travai, souvetan mestreja touto la cadeno: l'espicificacioun de l'esperimentacioun. Aven participa au councebemen e à la realisacioun d'un automate móuvedis. Aven asata un enroudamen de prougramacioun (MINISIM, ORCCAD) que pousquesse aparia uno lei d'asservimen a-n-un coumpourtamen lougi pèr la preso en comte dis evinamen.

Fin finalo prepausan li resulto esperimentalo validant touto aquesto acerco sus un travai de pousiciounamen visioun-viéu d'un cone.


[1]Gramaci a Rougiè e Alberto Rey

# Avant-Propos

Le travail présenté dans ce document a pu être réalisé dans le cadre d'une convention CIFRE entre trois laboratoires : l'ISIA-CMA (Ecole des Mines de Paris), ICARE de l'INRIA Sophia-Antipolis, l'Université de Nice Sophia-Antipolis (ESSI-I3S) et la société ECA comme partenaire industriel.

# Table des Matières

# Table des Figures

*Je ne sépare plus l'idée d'un temple de celle de son édification... ce qui fait, ce qui est fait sont indivisibles.*
*(P. Valéry, 1921)*

*Aie l'habitude de tenir à la main un fil à plomb, pour pouvoir apprécier l'inclinaison des choses.*
*(L. De Vinci, Traité de la peinture)*

# Introduction Générale

Le développement des capacités sensorielles des robots dans divers domaines, en particulier au niveau des systèmes de vision artificielle, est une des voies pour les faire sortir de leur état actuel d'infirme. Pourtant la technologie dispose depuis longtemps des moyens d'égaliser, sinon de surpasser, les capacités de nos organes sensoriels ; le vrai problème ne réside plus dans le fait de collecter des informations mais plutôt dans l'utilisation qui en est faite.

L'important est non seulement que nos sens nous fournissent l'information nécessaire pour que nous comprenions le monde extérieur, mais aussi que nous puissions traiter cette information et réagir de manière adéquate. Donner des capteurs aux robots présente peu d'intérêt si nous ne les rendons pas aptes à réagir aux informations. Il ne suffit donc pas que les robots puissent percevoir et comprendre le monde extérieur comme des observateurs passifs. Il importe également que, grâce à leurs capteurs, ils soient dotés d'une "rétroaction" sur le monde extérieur (cette notion de monde extérieur se limite à la petite portion du monde qui intéresse le robot).

Le travail que nous présentons consiste en l'étude et l'application de cette boucle de rétroaction sur la vision - autrement nommée asservissement visuel - dans le cadre de la robotique mobile en environnement d'intérieur. Nous désirons, à partir de formalismes bien établis, concevoir un robot mobile expérimental doté de réflexes visuels. Nous porterons notre attention sur le passage entre "la théorie" et la "pratique", étape trop souvent négligée. Pour se convaincre de l'importance de ce passage, on peut citer un commentaire instructif de M. Crosier : *Malgré nos dénégations, nous vivons toujours selon le principe du "one best way", c'est à dire selon l'illusion qu'on peut séparer les moyens et les fins et* *que les techniciens peuvent déterminer le seul meilleur moyen une fois qu'une fin a été clairement fixée... En fait ce raisonnement qui a permis de grand progrès est un raisonnement pauvre,... parce qu'il ne rend compte que d'une partie de la réalité... A côté de la démarche décomposante et hiérarchique impliquant une causalité simple, (on peut...) développer une démarche totalisante prenant en compte les ensembles "fins/moyens".*

En ayant comme soucis ce lien entre "fins" et "moyens", ce document présente la conception et la commande par asservissement visuel d'un robot mobile et se divise en cinq chapitres :

Chapitre 1 : un état de l'art est présenté en ce qui concerne la robotique mobile autonome avec ses problèmes et les différentes réalisations de laboratoires. Nous faisons le point sur les différents travaux en asservissement visuel en remarquant que peu ont été appliqués aux robots mobiles.

Chapitre 2 : nous présentons ensuite des techniques récentes de modélisations cinématiques et de commande des robots mobiles dont la structure mécanique comporte souvent des liaisons non-holonomes. Nous proposons une approche originale utilisant la coopération des mouvements d'une base mobile et d'un bras manipulateur placé dessus. Le fait de contrôler les mouvements de la caméra plutôt que ceux de la base permet de contourner les limitations des contraintes non-holonomes en considérant l'ensemble mécanique base et bras. Cette modélisation permet d'utiliser le formalisme de la commande-référencée-vision destiné en premier lieu aux bras manipulateurs rigides.

Chapitre 3 : Après les rappels sur la commande-référencée-vision, nous définissons et illustrons en simulation les différentes tâches élémentaires envisageables pour assurer le déplacement du robot en interaction avec son environnement visuel. Pour la réalisation de tâches robotiques plus complexes composées d'un enchaînement de tâches élémentaires, nous donnons un cadre formel pour gérer les problèmes de transition entre tâches référencées vision.

Chapitre 4 : L'implémentation de ces travaux a été effectué en ayant le soucis de développer ou d'adapter des outils de spécifications (MINISIM) et de conceptions (ORCCAD) pour disposer d'un environnement de programmation qui tienne compte des liens entre "théorie" et "pratique". Dans cet environnement la réalisation d'une tâche élémentaire d'asservissement prend en compte l'aspect continu (la loi de commande elle-même) et permet d'y associer un comportement logique fonction des événements provenant de l'environnement : c'est le concept de la "Tâche-robot".

Chapitre 5 : Nous avons enfin réalisé un système robotique complet pour répondre aux besoins de nos expérimentations sur les techniques de commande évoluée. Les principaux composants de ce système sont le robot mobile muni "d'une tête" à deux degrés de liberté et la machine de vision dédiée à l'asservissement visuel. La première tâche robotique visuelle réalisée est un positionnement vis à vis d'un objet (un cône). Cette tâche a été réalisée en utilisant toute la chaîne de conception qui va des spécifications aux observations expérimentales.

# Première Partie

# Etude Théorique

# Chapitre Premier

# Etat de l'art : Robotique Mobile et Vision

Avant d'entreprendre l'étude et la réalisation d'un robot mobile au comportement réactif basé sur des asservissements visuels, il est indispensable de faire l'état de l'art sur les travaux connexes à ce sujet, plus précisément :

- la Robotique Mobile Autonome avec les problèmes spécifiques qu'elle soulève et les différentes approches menées dans les laboratoires pour les résoudre.
- les aspects liés à la perception et plus particulièrement les approches visant à utiliser les informations visuelles pour contrôler le mouvement du robot (vision active).

## 1.1 La Robotique Mobile Autonome

### 1.1.1 Les points clés en Robotique Mobile Autonome

Un Robot Mobile Autonome [1] est un système automoteur doté de moyens de perception de son environnement partiellement ou totalement inconnu et de capacités décisionnelles qui lui permettent d'accomplir une mission donnée sans intervention humaine.

Dans les années soixante, c'est avec de grandes espérances que débutèrent les travaux sur la conception de RMA ; malheureusement les nombreuses difficultés rencontrées tempérèrent l'enthousiasme originel. Nous allons évoquer un certain nombre de ces difficultés qui permettront de préciser le cadre de notre étude.

[1] que l'on notera par la suite RMA

D'une façon générale, un RMA doit posséder des fonctions perceptuelles (vision, ultra-son...) lui permettant de construire des modèles de l'environnement dans lequel il évolue. Il doit, de plus, posséder des capacités de raisonnement sur ces modèles pour décider des actions à faire pour accomplir sa tâche. Dans la quasi totalité des cas, la décision se traduit par une spécification de trajectoire dans l'espace cartésien ou l'espace de configuration du robot. Cette trajectoire est ensuite exécutée en utilisant une commande en boucle ouverte sur l'environnement donc peu robuste vis à vis des perturbations de modèles (obstacles mal définis...) et surtout des perturbations de mesures (dans la pratique, l'odométrie d'un robot mobile à roues se révèle peu fiable).

La mise en application d'une telle approche n'est pas suffisante pour assurer une exécution correcte de la tâche avec des conditions de robustesse et de sécurité suffisantes. L'introduction d'une étape intermédiaire visant à traduire les contraintes de planification en modèle d'exécution exploitable par un schéma de commande robuste, se révèle indispensable.

L'architecture de contrôle du RMA doit pouvoir prendre en compte ce lien entre planification et exécution afin de permettre au robot de réagir vis à vis des erreurs de mesures et de modèles liées à sa perception de l'environnement. Nous allons rapidement passer en revue les différentes architecture de contrôle en œuvre en robotique mobile.

#### 1.1.1.1 Architecture de contrôle

Différentes approches ont été mises en œuvre au niveau des architectures de contrôle. On peut tenter de les classer en fonction de l'aspect plus ou moins distribué des processus de perception et de décision qui les constituent.
(Approche centralisée)
Une approche naturelle consiste à décomposer le système selon son fonctionnement et à distinguer plusieurs modules fonctionnels :

- un module de perception pour acquérir des informations sur l'environnement,
- un module de modélisation pour se faire une représentation du monde à partir des informations du module de perception,
- un module de navigation et planification qui décompose la mission en tâches élémentaires,
- un module de pilotage du robot qui assure l'exécution de la tâche élémentaire de déplacement.

Cette approche amène à une représentation centralisée sur laquelle s'appuie les différents niveaux de décisions. Depuis le robot mobile Shakey ([Steele 86]) de nombreuses applications utilisent cette approche en utilisant des bases de plan ([Isik 88]), des blackboard ([Thorpe 91], [Albus 90])...

Cette approche est très contreversée car l'organisation hiérarchique des fonctions du robot pénalise la testabilité, la versabilité et la robustesse de son logiciel.

(Approche réactive ou à base de comportement)
Les travaux de Brooks ([Brooks 86], [Flynn 88], [Brooks 90]) ont ouvert une nouvelle voie en proposant de décomposer le système suivant le comportement désiré c'est à dire la réaction vis à vis de stimuli extérieurs. Chaque sous-système a accès aux capteurs et aux effecteurs. Les comportements élémentaires sont structurés hiérarchiquement de telle sorte qu'un comportement puisse inhiber les entrées ou les sorties d'un comportement inférieur ; c'est la notion d'architecture de "subsumption".

Bien qu'elle résolve certains problèmes inhérents aux architectures centralisées, cette approche en amène d'autres. En effet, la prise de décision est très difficile car il n'existe plus de données centralisées pour représenter l'état de l'environnement. La construction incrémentale du comportement du robot ne permet pas d'analyser ou de prédire les réactions du robot.

(Approche hybride)
Il semble qu'il existe actuellement un certain consensus sur le développement d'architectures hybrides tendant à faire cohabiter une partie réactive et une partie décisionnelle centralisée. Différentes réalisations utilisent cette approche : l'architecture du robot mobile HILARE ([Chatila89], [Giralt 90]), celle appelée RBM pour Rational Behavior Model ([Byrnes 93]), le Symbolic Subsumption Servo ou SSS ([Connel 92]), l'approche ORCCAD ([Simon 93])...

Des études sont également menées pour proposer des architectures logicielles moins "rigides" ; par exemple il existe des systèmes réactifs avec des modèles du monde ([Mataric 90],[Tigli 93]) des blackboards distribués ([Occello 93]). On peut cependant constater que les travaux sur ces architectures logicielles manquent encore de méthodologie et d'outils pour la spécification, la représentation et la sémantique des tâches robotiques. De plus on dispose de peu de moyens pour évaluer et comparer toutes ces architectures.

Le degré d'autonomie d'un robot mobile sera beaucoup fonction de sa faculté d'exploitation des informations perceptuelles au niveau de la navigation. Nous allons maintenant présenter les approches développées dans ce domaine.

#### 1.1.1.2 Navigation et Perception

La navigation se définit de manière générale comme l'art de se diriger vers une destination finale en déterminant sa route. Or le robot se déplace dans un environnement partiellement ou imparfaitement connu à l'avance et de nature dynamique. Ainsi la planification des actions ne peut pas être rigide et doit pouvoir être remise en cause à chaque instant.

De plus, durant ses déplacements, le robot devra être capable d'assurer la mise à jour de sa connaissance sur l'environnement en utilisant :

- une représentation sémantique des lieux et des objets (ex: couloir, route, arbre...),
- une représentation topologique qui interconnecte ces lieux,
- une représentation géométrique pour permettre au robot de se localiser géographiquement.

Certains travaux utilisent simplement une carte géométrique ([Buffa 93]) ou une carte topologique ([Kuipers 91]) ou combinent les deux ([Thorpe 91]). On remarque que l'utilisation d'une carte topologique est mieux adaptée à l'utilisation d'approches locales : chaque arête de passage entre chaque noeud de la carte correspond à un déplacement, éventuellement à partir d'un asservissement visuel, sur un amer de l'environnement (suivi de murs, positionnement...). En revanche la reconstruction d'une carte géométrique nécessitent l'utilisation de capteurs fournissant des mesures 3D (stéréovision, caméra et laser, vision dynamique).

La perception est donc un élément essentiel à la navigation et doit permettre, entre autres, la localisation du robot. Cette localisation peut se faire à deux niveaux : relative par rapport à l'environnement local ou absolue par une carte globale des lieux. En robotique mobile, la localisation relative est utilisée essentiellement pour des raisons de sécurité ; nous montrerons que, par le biais des techniques d'asservissement visuel, elle est également capable d'assurer le déplacement du robot en interaction avec son environnement local.

Pour obtenir des informations sur son environnement afin de réaliser une tâche, un robot peut utiliser différents types de capteurs ([Durrant-Whyte 88], [Walker 93], [Makela 93]) ; une autre approche, que nous adopterons, consiste à utiliser la vision comme capteur principal voire unique ([Buffa 93]).

La résolution de tous les points durs en robotique mobile nécessite de nombreuses expérimentations qui ont donné jour à divers robots mobiles de laboratoires. Sans être exhaustif, nous présentons certains systèmes qui nous paraissent représentatifs des travaux effectués en robotique mobile.

### 1.1.2 Les systèmes les plus représentatifs

Un des premiers robots mobiles fût le robot *Shakey* de l'institut de recherche de Stanford dans les années soixante dix ([Moravec 77], [Marsh 86], [Steele 86]). Ce robot a été développé dans le but d'étudier des applications de l'intelligence artificielle. La commande du robot mobile et la vision étaient secondaires et ont été peu expérimentées.
A l'inverse, le robot *Cart* de l'université de Stanford était destiné à l'étude de la commande des robots mobiles par la vision. Relié à un gros ordinateur, il permettait dans les années 80 de :

- suivre une ligne blanche au sol avec une caméra,
- suivre une série de segments rectilignes en se basant sur des points de référence à l'horizon,
- atteindre un point situé à une distance de 20 m.

La technologie de l'époque permettait au robot de traverser une pièce en cinq heures. Dans le prolongement des études effectuées sur le robot *Cart*, l'université de Carnegie-Mellon a réalisé plusieurs robots :

- Le robot *Pluton* est doté de 3 roues motrices et directrices. Il a été créé pour l'étude de la commande par la vision. Ce programme n'a pu aboutir du fait de la complexité de la commande du robot avec une telle structure mécanique.
- Le robot *Neptune* est un tricycle muni de deux caméras et d'une ceinture de 24 capteurs à ultra-son. La stéréoscopie est utilisée pour la localisation du mobile et lui permet de parcourir une pièce en une heure ; avec les capteurs à ultra-son, les calculs sont moins précis mais dix fois plus rapides.
- Le robot *Uranus* ou *Ilonator* ([Feng 89]) est doté de 4 roues omnidirectionnelles. Sa structure mécanique a été étudiée pour éliminer les contraintes non-holonomes. Ce robot est destiné à étudier les problèmes de commande.
- Le robot *Terragator* est doté d'un moteur à essence. Il parcourt de courtes distances sur route en se guidant visuellement sur le bord de la chaussée.

Au MIT, ont été réalisé des robots réactifs ([Flynn 88], [Brooks 86], [Hartley 91], [Brooks 91]) qui ont des comportements complexes tels que se promener en évitant des obstacles comme *Allen* et *Herbert*. Des recherches sont également effectuées sur la miniaturisation des robots mobiles comme *Tom* and *Jerry* qui

sont des voitures télécommandées dont le radio-contrôle a été remplacé par des PAL. *Seymour* est le robot le plus évolué, il permet de tester les capteurs passifs (caméra, barrette CCD...) pour effectuer de la stéréoscopie, de l'analyse de mouvement, de l'asservissement en vitesse par la mesure du temps de collision ...
Le robot de Meystel ([Isik 88]) utilise une caméra pour la navigation et une caméra pour le pilotage. La fonction de pilotage permet dans un cône de l'image de détecter des obstacles. Sous l'hypothèse d'un terrain plan, il peut déterminer la distance mobile-obstacle.
En France, on peut citer comme exemple de robots mobiles de laboratoires :

- *Hilare* du LAAS à Toulouse ([Giralt 90], [Chatila89]) qui utilise la stéréoscopie pour localiser un objet mais qui a comme axes de recherche principaux la programmation des robots mobiles et la planification.
- le robot mobile du projet ROBOTVIS de l'INRIA de Sophia-Antipolis qui sert de support expérimental à l'étude de la vision 3D (avec 3 caméras) ([Matthieu 89], [Buffa 93]).

Les robots mobiles dotés de caméras et destinés à une application industrielle utilisent souvent des balises visuelles artificielles pour se recaler tels que par exemples des codes barres sur les murs ([Kabuka 87]) où des marques au sol ([Frohn 89]).
Seuls les robots de laboratoire peuvent se repérer sans balise dans un environnement structuré, en utilisant la vision. Ils ont souvent comme tâche d'estimer un modèle de l'environnement dans lequel ils évoluent grâce à des techniques de reconstruction 3D. La plus répandue est celle de la stéréoscopie classique avec deux ou trois caméras ou un couple caméra-laser. Ces approches sont employées par les deux laboratoires français cités ci-dessus par exemple. L'utilisation de la vision dynamique est également utilisée mais elle se heurte à des difficultés quant à l'estimation correcte du mouvement de la caméra. Une approche moins générale consiste à "filtrer" l'image avant son enregistrement par la caméra ([Hong 91], [Takeno 91]). Les données 3D calculées sont ensuite exploitées par le planificateur du robot pour commander son déplacement. Ce déplacement local est souvent effectué sans asservissement visuel sur l'environnement. Ceci génère une erreur importante au niveau du suivi de la trajectoire. Cela a conduit les chercheurs à étudier diverses approches permettant d'estimer plus précisément la position du robot vis à vis de sa trajectoire.

- [Atiya 91] ou [Crowley 91] utilise une carte pré-enregistrée de segments verticaux de l'environnement que l'on compare avec celle que le robot a reconstruit,

- une connaissance a priori sur les segments extraits de l'image fournit des informations 3D et facilite l'estimation de la position du robot mobile (ex: écart latéral et orientation dans un couloir). Cette approche est décrite dans [Lebègue 92a] ou [Zhang 93].
- A l'université d'Osaka ([Barth 91]) la vision active est utilisée en dotant la caméra embarquée sur le robot d'un degré de liberté en lacet. Le principe consiste à fixer un segment vertical par la caméra tandis que le robot se déplace le long d'une trajectoire décrivant un arc de cercle ou une droite. L'angle de rotation de la caméra permet d'estimer l'orientation du robot.

L'intérêt d'étudier la réalisation et la commande d'un manipulateur mobile [2] est apparu depuis peu. Les travaux les plus nombreux portent sur la commande d'un point terminal de l'effecteur :

- [Hootsmans 91] utilise un modèle dynamique incluant les suspensions du véhicule pour contrôler une trajectoire de référence de l'effecteur.
- [Campion 93] emploie un modèle plus simple dans le cadre d'une approche réactive par champ de potentiel où les obstacles génèrent des forces répulsives et les buts à atteindre des forces attractives.
- [Mishck 92] ou [Wang 93] prennent en compte des bras manipulateurs à axes redondants et incluent dans la commande la minimisation d'une tâche secondaire sur la trajectoire de la base.
- [Yamamoto 93] utilise une commande à retour d'effort pour suivre une surface en mouvement.

Après avoir sommairement présenté certains problèmes que l'on rencontre en robotique mobile autonome notamment celui du passage de la perception vers l'action, il nous semble utile de faire un bref panorama de l'utilisation de la vision en robotique. Nous nous intéresserons plus particulièrement à son utilisation dans le cadre de la commande de robots.

## 1.2 Utilisation de la vision en robotique

### 1.2.1 Les capteurs extéroceptifs

Les capteurs proprioceptifs (odométrie) fournissent des renseignements sur l'état interne du robot. Ils ne tiennent pas compte des problèmes dus à l'interaction

[2] bras manipulateur monté sur un robot mobile

avec l'environnement tel que : patinage des roues, glissement latéral du véhicule, détection d'obstacles... Il est donc vain de penser pouvoir situer précisément le robot uniquement à partir de ce type de mesures.
Pour acquérir des informations sur son environnement, les capteurs extéroceptifs sont indispensables. On distingue plusieurs types de capteurs extéroceptifs :

- les capteurs tactiles qui préviennent d'un contact avec un élément extérieur,
- les capteurs proximétriques qui détectent la présence d'un objet situé à une distance inférieure au mètre,
- les capteurs télémétriques pour des objets situés à une distance plus importante,
- les capteurs visuels (caméra ou nappe laser par exemple) qui permettent de percevoir l'environnement.

Ces capteurs délivrent des informations pouvant être utilisées à deux niveaux :

- **au niveau raisonnement:** afin de définir des stratégies de planification de la tâche.
- **au niveau asservissement:** afin de permettre des actions réflexes en intervenant directement au niveau de la boucle de commande du robot.

La vision en robotique est souvent utilisée au niveau raisonnement mais depuis quelques années, un fort intérêt s'est développé pour son utilisation au niveau asservissement.

### 1.2.2 Les différentes approches d'utilisation de la vision

La planification d'une tâche robotique et la localisation du robot nécessite la reconstruction de l'environnement, de manière explicite ou implicite. Différentes techniques de reconstruction à l'aide de la vision existent :

- la stéréoscopie avec deux ou trois caméras ([Faugeras 88]),
- la vision dynamique en utilisant une caméra mobile ([Rives 87], [Espiau 87], [Bouthemy 88], [Vernon 90], [Nagel 86]).

Cette reconstruction 3D, qui nécessite des traitements informatiques importants, n'est pas toujours indispensable à la réalisation de certaines tâches robotiques. Ainsi à partir d'informations visuelles de bas niveau tel que des points, des segments, des droites...extraites de l'image, on peut contrôler le déplacement relatif du robot vis à vis d'éléments caractéristiques de l'environnement. Ce type d'asservissement sur des informations visuelles peut se mettre en oeuvre de deux façons:

- par asservissement en situation,
- par asservissement visuel.

On trouvera dans [Wijesoma 93] une étude expérimentale des deux approches que nous allons maintenant décrire plus précisément.

#### 1.2.2.1 Asservissement en situation

Cette approche consiste à positionner la caméra par rapport à un objet. La situation courante de la caméra vis à vis de l'objet est obtenue en interprétant les informations visuelles sur l'objet extraites dans l'image. Le déplacement du robot est ensuite assuré par un classique asservissement en position dans l'espace cartésien (*figure 1.1*).

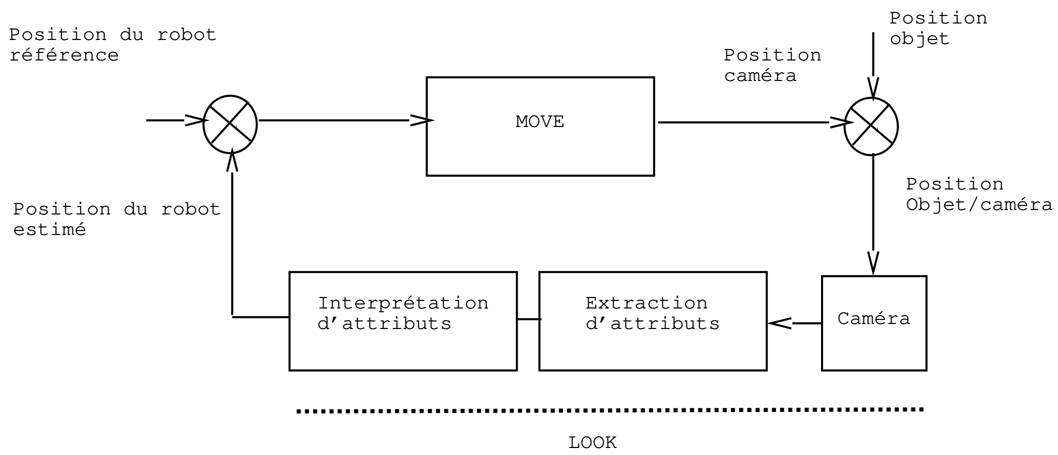



Figure 1.1 : Asservissement en situation

Il existe deux manières de réaliser cet asservissement en situation :

- static look and move : la phase d'acquisition d'images avec déduction de la situation courante est enchaînée séquentiellement avec le déplacement du robot ([Agin 77]).
- dynamic look and move : les deux étapes sont effectuées simultanément ([Allen 91], [Westmore 91], [Agin 79]). Cette technique a l'avantage de pouvoir suivre des objets en mouvements.

Théoriquement, pour atteindre une situation désirée, une seule itération est nécessaire, mais dans la pratique il faut tenir compte des erreurs dues à :

- l'extraction d'attributs dans l'image,
- la modélisation de la caméra,
- la modélisation du robot.

L'interprétation des informations visuelles pour estimer la situation de la caméra nécessite des hypothèses fortes telles que la connaissance des dimensions de l'objet ou de la distance caméra-objet. Si l'on ne peut pas formuler de telles hypothèses, on est dans l'obligation de reconstruire l'espace 3D par les méthodes indiquées précédemment.

#### 1.2.2.2 Asservissement visuel

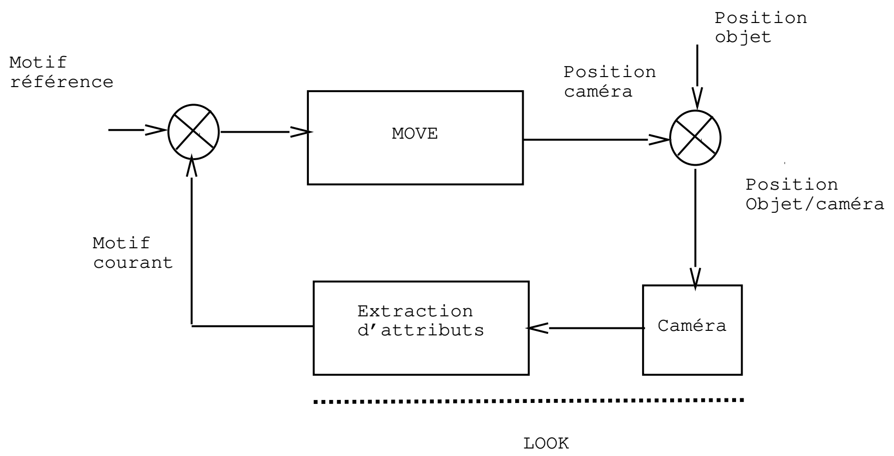



Figure 1.2 : Asservissement visuel

Avec cette technique, on cherche à atteindre une configuration particulière d'un motif dans l'image et non plus à contrôler une situation entre la caméra et l'objet (*figure 1.2*). Ainsi on supprime la phase de reconstruction d'un modèle 3D et on élimine du même coup les erreurs sur l'estimée de la situation. Dans ce type d'approche les calculs se révèlent le plus souvent beaucoup moins complexes que dans l'approche précédente.

### 1.2.3 Les travaux en asservissement sur les informations visuelles

#### 1.2.3.1 Les bras manipulateurs

Pour illustrer cette approche, on peut consulter l'article [Gelbgras 89] où le calcul du contour d'un objet en 3D a nécessité de coupler la caméra à un laser. L'expérimentation consiste ensuite à suivre le contour de cet objet avec un bras manipulateur. D'autres exemples de calculs de position et d'orientation d'objets dans l'espace 3D sont rapportés dans :

- [Allen 92], [Allen 91] où deux caméras fixes sont utilisées pour estimer le mouvement d'un train en modèle réduit. L'estimation de sa trajectoire et de sa vitesse est réalisée à l'aide d'une architecture pipeline et permettant ainsi de saisir le train en mouvement à l'aide d'un bras manipulateur.

- [Kay 91] où deux caméras CCD sont embarquées sur un bras manipulateur. L'estimation de la position 3D de l'objet tient compte du bruit sur la mesure mais aussi du bruit sur le positionnement du robot (notion de jacobienne étendue du robot).

- [Skofteland 91] où l'effecteur est muni de deux caméras pour faire de la stéréovision, de capteurs d'efforts, et de capteurs télémétriques laser, ceci afin d'effectuer la saisie d'un objet polyédrique dans l'espace 3D. Ce travail n'en est qu'à l'étape de la simulation.

Les travaux menés dans le cadre de l'asservissement visuel mettent en lumière différents types de problèmes :

- Dans les travaux présentés dans [Chaumette 90], [Espiau 92] sous le nom de Commande référencée vision, les efforts ont porté sur :

  - la modélisation des variations d'un signal observé dans l'image en fonction des mouvements de la caméra. Cette modélisation est faite par le biais d'un torseur d'interaction dont une forme analytique peut être calculée pour des primitives géométriques paramétrables dans l'image,

- la classification des tâches spécifiées en termes de liaisons virtuelles entre la caméra et la scène,

- l'intégration du problème dans le formalisme des fonctions de tâches appliqué à commande des bras manipulateurs. Ce formalisme général est décrit dans ([Samson 90a]). La Commande référencée vision fait l'objet de nombreux développements. A titre de démonstration, un manipulateur portant une caméra sur l'effecteur utilise les techniques d'asservissement visuel pour pointer un objet en mouvement et réaliser des tâches de positionnement vis à vis d'une cible constituée par des points ([Chaumette 90]) ou des ellipses ([Rives 93a]). Dans [Coste-Manière 93] il est présenté un suivi de contour de pièce manufacturée en faisant coopérer la commande référencée vision pour approcher la pièce et un retour en force pour le suivi de contour. L'équipe Témis de l'IRISA utilise également la commande référencée vision pour reconstruire les informations 3D de primitives visuelles ([Boukir 93]).

- Contrairement aux travaux précédents, Feddema ([Feddema 89]) utilise un torseur d'interaction moins général puisqu'il est exprimé uniquement pour un segment repéré par ses coordonnées du point milieu, son orientation et sa longueur. Ce torseur lui permet de contrôler la position de la caméra et *une* de ses orientations. La prise en compte des images pour calculer le torseur et en déduire la commande se fait de manière asynchrone et discontinu. Un système de fenêtrage sur les régions d'intérêt dans l'image est utilisé pour réduire le temps de calcul. Le problème de la sélection des informations dans l'image a été abordé dans [Feddema 91]. Cette sélection est basée sur des critères de reconnaissance (robustesse, unicité, complexité, temps de calcul) et de contrôle ( observabilité, contrôlabilité, sensibilité). Ce travail montre l'importance du conditionnement de la matrice d'interaction dans le comportement de l'asservissement. Il est à noter que ce problème est comparable à celui de la régularité du jacobien d'un robot. Une matrice bien conditionnée doit être telle qu'un petit changement de cette matrice et du positionnement de l'effecteur induit un petit changement dans l'image.
  La démonstration présentée est la reconnaissance d'une pièce mécanique (joint de carburateur) et la saisie de cette pièce par un robot manipulateur PUMA commandé par un Vax.

- Un autre travail est présenté dans [Weiss 87] où l'on montre la nécessité de relier la variation des informations visuelles dans l'image et le mouvement de la caméra. Une approche adaptative est utilisé pour la commande. L'étude reste limitée à des robots ayant un maximum de trois degrés de liberté. Dans sa conclusion il fait remarquer que l'asservissement visuel est intéressant pour la commande en temps-réel d'un robot mobile au niveau local, mais que l'interprétation globale de la scène peut se faire à une cadence moins élevée.

- Dans l'article de Kabuka [Kabuka 88] l'asservissement visuel est réalisé sans extraction des contours de l'image. Le gradient de l'application, associant la configuration du motif dans l'image à la situation relative entre la caméra et la cible, est estimé directement par une méthode de perturbations. Le choix du paramétrage du motif dans l'image est tel que, chaque variation d'un degré de liberté du manipulateur entraîne seulement la variation d'une composante du vecteur représentant le motif dans l'image. Cela se traduit par un découplage de la commande du manipulateur dans l'espace de l'image. Chaque axe est ainsi contrôlé indépendamment par une commande adaptative. Cette étude a porté sur les mouvements de translation de la caméra qui permettent de contrôler la position et l'échelle d'un objet dans l'image. Par contre, elle ne permet pas de contrôler l'orientation relative de la caméra vis à vis de l'objet.
  La démonstration réalisée consiste à poursuivre un objet en contrôlant sa position et son échelle dans l'image.

- Au CMU [Papanikolopoulos 91] effectue une étude comparative de différents modes de contrôles qui sont :

  la régulation proportionnelle-intégrale (PI),

  une commande boucle fermée par placement de pôles,

  une version modifiée du contrôle LQG (Linear Quadratic Gaussian) où le vecteur de perturbation et le vecteur d'état sont estimés par un filtre de Kalman [Papanikolopoulos 93].

  Les résultats de cette étude nous confirment qu'un contrôle PI est utilisable si les mesures visuelles sont précises, et qu'il vaut mieux utiliser un contrôle stochastique (LQG) pour des données bruitées. Pour la modélisation, il utilise les équations du flot optique auxquelles il ajoute, pour des raisons de précisions, une technique d'appariemment nommée *SSD optical flow* (somme des différences carrées du flot optique). Cette technique de *SSD optical flow* est détaillée par exemple dans [Heel 90]. Une application au suivi d'une trajectoire 2D par un bras manipulateur est présentée .

- [Hashimoto 91] utilise aussi les équations du flot optique pour calculer la matrice d'interaction pour N *points* de l'image. La commande est faite par un retour d'état. L'expérimentation consiste encore à suivre la trajectoire 2D d'un objet. L'inconvénient majeur d'utiliser des points comme amer dans l'image est qu'ils fournissent des informations bruitées en comparaison à des primitives géométriques plus structurées telles que les droites.

- Dans [Garcia 89] la localisation d'un objet en 2D utilise une méthode dérivée de la transformée de Hough présentée dans [MAI 85]. Elle est plus rapide, mais nécessite une meilleure initialisation de la position de l'objet à suivre. La poursuite de l'objet est ensuite réalisée à l'aide d'un filtre de Kalman.

- Dans [Jang 91] les informations visuelles utilisées dans la commande sont l'aire, les moments centraux... Dans le cas où le nombre d'informations visuelles est égal au nombre de degrés de liberté du robot (comme dans [Weiss 87] par exemple) la matrice d'interaction peut posséder des configurations singulières. L'auteur propose d'utiliser des informations redondantes pour résoudre ce problème.

- Les travaux de [Corke 92] montre l'importance des retards, des gains variables dus à la perspective, des vibrations dans la performance d'un asservissement visuel. Le banc d'essai consiste en un pointage par un bras manipulateur de deux LEDs qui s'allument et s'éteignent mutuellement. La simplicité de la primitive visuelle permet d'avoir une période de commande de 20 ms mais en ne contrôlant qu'un axe du robot.

- Le travail de [Nelson 93] combine à l'asservissement visuel un critère de manipulabilité permettant au robot de s'éloigner des singularités cinématiques et des butées mécaniques.

- On peut citer également les travaux de suivi d'objets dans l'image qu'ils soient déformables ([Rehg 91]) ou rigides ([Hutchinson 91]).

- Enfin, le seul travail qui aborde le problème des "liaisons" caméra - environnement est réalisé par [Hutchinson 91] où il définit une surface de contrainte visuelle. Cette surface permet de réaliser par exemple une phase d'approche d'objets (ex: approche d'une vis vers un trou), puis de commuter sur une commande à retour d'efforts pour insérer la vis. Ce travail n'en est qu'à sa phase d'étude.

Une récapitulation des travaux énumérés est présentée dans le tableau "Travaux dans le domaine des asservissements visuels".
Afin de réduire le temps de traitement de l'image, les techniques d'asservissement visuel effectuent souvent le traitement sur une fenêtre contenant les motifs visuels ([Feddema 91]). Le suivi de la fenêtre dans l'image utilise alors un filtre prédictif ([Westmore 91]). Cette méthode peut être aussi appliquée à la stéréoscopie ([Toscani 88]).

Dans les applications décrites précédemment, l'asservissement visuel sert à contrôler un bras manipulateur. Une autre classe d'application de ce type d'asservissement est la conduite de véhicules autonomes.

#### 1.2.3.2 La conduite de véhicules

De nombreux projets ont été lancés afin d'étudier la réalisation de véhicules autonomes. Ces projets nécessitent de rassembler diverses compétences qui vont de l'intelligence artificielle (reconnaissance de formes, raisonnement, communication entre véhicules) à la commande (asservissement visuel, modélisation...). On peut citer certains travaux tels que :

- Le projet NAVLAB ([Singh 91], [Thorpe 91]) qui permet la conduite automatique d'un véhicule à 18 km/h. La vision, dans ce projet, n'est pas utilisée, ceci afin d'éviter les problèmes de sa mise en œuvre dans des scènes d'extérieur (ombres, mauvais contraste ...). Le positionnement est calculé grâce à des capteurs inertiels et l'évitement d'obstacle utilise un capteur laser (CYCLONE) spécialement réalisé pour ce véhicule.

- Par contre le projet ALV utilise la vision pour guider le véhicule. Il a été établi une base de connaissance à l'aide d'amers visuels décrivant une route (carrefour, virage...) [Le Moigne 88]. Une interprétation 3D de la scène est faite par une caméra couplée à un laser ([Dunlay 88]), elle permet de définir la position du véhicule. La commande est du type asservissement en situation.

- Pour des raisons de confidentialité, il est difficile d'avoir des renseignements sur les recherches effectuées par les constructeurs automobiles. Notons que PSA travaille sur un projet de suivi d'autoroute par asservissement visuel ([Jurie 93]).

- Dickmanns utilise des techniques d'asservissement visuel pour guider sur circuit le véhicule *VaMors* jusqu'à une vitesse de 96 km/h. Ces travaux (décrits dans [Dickmanns 91], [Dickmanns 88]) utilisent les notions de fenêtrages pour le suivi d'amers, de sélection des "meilleurs" amers. Les calculs temps-réel sont effectués sur une architecture parallèle dédiée au traitement d'images.

Le tableau "Véhicules autonomes" présentent quelques applications de la vision à la conduite de véhicules autonomes.

| Référence | Modélisation | Commande | Implémentation |
|---|---|---|---|
| [Allen 92]<br>Columbia | Jaco. feature<br>+stéréo | filtre prédictif | PUMA 560<br>pipeline dédié<br>T=200 ms |
| [Corke 92]<br>CSIRO<br>Australie | Jaco. de points | proportionnel | PUMA 560<br>carte 68040<br>1 ddl, T=20 ms |
| [Espiau 92]...<br>IRISA<br>INRIA | Jaco. de primitives<br>géométriques.<br>liaisons virtuelles | Commande-<br>réfé.-capteur<br>Proportionnel | SUN, AID ou BRAS<br>6 ddl<br>T=40 ms |
| [Feddema 91]<br>Purdue U.<br>W. Lafayette | Jacobienne<br>d'un segment | Asynchrone et<br>discontinue | SUN3, VAX<br>PUMA 600<br>T=70 ms |
| [Hashimoto 91]<br>Osaka | Jacobienne<br>de point | par retour<br>d'état | PUMA 560<br>transputer<br>T=250 ms |
| [Hutchinson 91]<br>U. of Illinois | Surface de<br>contraintes visu. | Association<br>avec compliance | non encore<br>implémenté |
| [Jang 91]<br>Corée | Feature sur aire,<br>moment,... | Algorithmique | non temps réel<br>tache robot<br>= Traj de features |
| [Kabuka 88]<br>U. of Miami | Gradient par<br>méthode de<br>perturbation | adaptative.<br>contrôle de<br>l'échelle objet | 1 ddl |
| [Papanikolopoulos 93]<br>CMU | Jacobienne<br>d'un point<br>optical SSD | PI<br>placement-pôles<br>LQG+Kalman | bras manip.<br>suivi traj. 2D<br>T=150 ms |
| [Taalebinezhaad 91] | Jaco. point<br>+eq. luminosité<br>suivi image | Minimisation | non implémenté |
| [Rehg 91]<br>CMU | suivi image<br>SSD | suivi minimun<br>SSD | Simulation<br>Silicon Graphics |
| [Weiss 87]<br>CMU | jacobienne | adaptatif | 3 ddl<br>cartes 68030<br>T=30 ms |
| [Westmore 91]<br>U. of Waterloo<br>Wesmhore | Asservissement<br>en situation | Kalman<br>fenêtre et<br>estimation pos. | IBM-PC<br>Transputer<br>T=16 ms |

Figure 1.3 : Travaux sur l'asservissements visuel

| Référence | type et Nom | Descriptions |
|---|---|---|
| [Dickmanns 88]<br>[Dickmanns 91]<br>Université de Munich | van<br>VaMors | vitesse max = 96 km/h,<br>asservissement visuel,<br>fenêtrage, selection *features* |
| [Thorpe 91]<br>[Singh 91]<br>CMU | van<br>Navlab | max = 18 km/h,<br>position par capteurs inertiels,<br>obstacles détectés par laser |
| [Le Moigne 88], [Davis 91]<br>Martin Marietta's lab.<br>U. of Marynland | ALV | Asservissement en position<br>base de connaissance d'amers |

Figure 1.4 : Véhicules autonomes

#### 1.2.3.3 La vision active

Il est maintenant bien connu que les problèmes de vision classiques (reconstruction 3D, estimation du mouvement ...) sont le plus souvent mal posés et que leur résolution nécessite l'adjonction de contraintes supplémentaires. Dans le cas des méthodes dites "passives", ces contraintes apparaissent le plus souvent par le biais de techniques de régularisation. Plus récemment, une nouvelle façon d'aborder le problème a vu le jour connu sous le nom de "vision active". Dans cette approche, il ne s'agira plus d'imposer des contraintes a posteriori sur l'image mais directement au niveau du processus d'acquisition : contrôle de la vergence dans le cas de la stéréovision, contrôle du mouvement dans le cas de la vision dynamique etc...On peut alors montrer que, sous réserve de bien contrôler les paramètres de l'acquisition de l'image, la majorité des problèmes classiques de la vision par ordinateur, deviennent mieux posés.

Les supports expérimentaux sont le plus souvent des "têtes robotiques" binoculaires ou trinoculaires ([Viéville 92], [Ekhlund 93], [Gros 92], [Wavering 93]). Les techniques d'asservissements visuels sont ici très utiles pour contrôler les conditions d'acquisition de l'image. On peut remarquer que peu de têtes (mis à part [Crowley 92]) sont montées sur un robot mobile et qu'aucune ne propose de coopération tête et base mobile.

#### 1.2.3.4 Conclusion

Les techniques d'asservissements visuels sont utilisés dans le contrôle de bras manipulateurs, la conduite de véhicules ou la vision active. En revanche ces techniques ne sont pas utilisées en robotique mobile où la vision intervient souvent à un haut niveau d'abstraction pour la reconstruction 3D, l'aide à la navigation... Aussi il existe un fossé entre la prise de décision et l'action. Pour assurer l'exécution d'une tâche d'un robot mobile de façon robuste vis à vis des changements de l'environnement, il apparaît intéressant de doter les robots mobiles de réflexes visuels.

# Chapitre 2

# Modélisation et commande des robots mobiles

## 2.1 La commande des robots mobiles

Etudier une commande pour des robots mobiles ne peut se faire sans être confronté à des problèmes spécifiques parmi lesquels nous trouvons :

- l'étude des structures mécaniques comportant des liaisons non-holonomes,
- la commande par retour d'état de ces structures,
- l'utilisation dans une boucle fermée de capteurs extéroceptifs, en particulier la vision.

### 2.1.1 Les contraintes non-holonomes

Les principales difficultés dans la commande d'un robot mobile en comparaison avec un bras manipulateur proviennent de la présence de contraintes mécaniques spécifiques.

L'étude du paramétrage d'un système mécanique permet de définir les liaisons qui sont des contraintes limitant les possibilités de mouvement des solides constitutifs du système. Une liaison se traduit, schématiquement, par des relations de dépendances entre les paramètres du système $q$, les vitesses généralisées $\dot{q}$ et le temps $t$.

Dans le cas des bras manipulateurs, les solides constituant le robot ne sont astreints qu'à des conditions géométriques de contact entre eux. Ces relations de liaison ne font pas intervenir les vitesses. Elles sont dites *liaisons holonomes* et peuvent s'exprimer par des équations de la forme :

$$f(q, t) = 0 \tag{1}$$

En revanche dans le cas des robots mobiles à roues, en plus des conditions géométriques de contact, les solides du système peuvent être astreints à certaines conditions cinématiques de contact. Les relations de liaisons font alors intervenir les vitesses généralisées. Ces liaisons sont dénommées *liaisons non-holonomes* et leur comportement est régi par des contraintes de la forme :

$$f(q, \dot{q}, t) = 0 \tag{2}$$

On remarque qu'une liaison non-holonome nécessite au préalable une liaison holonome traduisant le contact purement géométrique. Pour illustrer ce type de liaisons, considérons une roue de centre $C$ de coordonnées $(x, y)$ dans un repère $R_o$ (figure (2.1)).

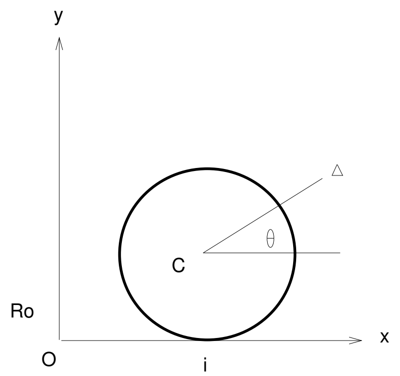



Figure 2.1 : Exemple d'une roue

La condition de contact géométrique correspond à un degré de liaison :

$$y = R \tag{3}$$

La condition cinématique de **roulement sans glissement** s'écrit :

$$\vec{V}(i) = 0 \tag{4}$$

En utilisant le paramétrage $(x, \theta)$ illustré figure(2.1), la relation dite parfois *semi-holonome* s'écrit :

$$\dot{x} + R\dot{\theta} = 0 \tag{5}$$

Bien d'autres exemples simples peuvent se trouver dans [Bamberger 81].

Dans la pratique la présence de liaisons non-holonomes dans un mécanisme se traduit par le fait que pour une configuration donnée du mécanisme, il existe certaines directions dans laquelle le mécanisme ne pourra se déplacer de façon instantanée. Par exemple, dans le cas d'une voiture, cette direction sera donnée par l'axe liant les roues non-directrices. Il est à noter toutefois que l'existence de ces contraintes n'entraîne pas une impossibilité d'atteindre une configuration désirée ; celle-ci pourront être atteinte par le biais de manœuvre. L'existence de ces liaisons non-holonomes pose des problèmes difficiles au niveau du contrôle de tels mécanismes que nous allons maintenant évoquer.

### 2.1.2 La commande par retour d'état des robots mobiles

C'est très récemment que l'on s'est intéressé aux problèmes à la commande en boucle fermée des robots mobiles. Cet intérêt a suscité de nombreuses études sur des lois de commande tenant compte des contraintes non-holonomes ([Laumond 91], [Ait-Abderrahim 93], [Samson 91b], [Andréa-Novel 91], [Tournassoud 88]).

Les travaux effectués au sein de l'équipe Icare de l'INRIA ([Samson 91c], [Ait-Abderrahim 93]) montrent que du fait des liaisons non-holonomes, il n'existe pas de commande à retour d'état stationnaire continu capable de stabiliser une base mobile de type "char" [1] à la fois en position et orientation autour d'une configuration terminale. Une première voie explorée a consisté à reformuler le problème de stabilisation en terme de poursuite d'une trajectoire dans l'espace cartésien. Cela a permis de montrer que ce problème de poursuite peut être traité par des classes de commande à retour stationnaire non linéaire ou linéarisée. Ce type d'approche est réalisé en pratique en utilisant une base mobile virtuelle dont la trajectoire sert de consigne d'asservissement au robot réel.

Une deuxième approche pour la stabilisation des robots mobiles, a été développée plus récemment et a déjà fourni des résultats très prometteurs. Celle-ci concerne la commande par retour d'état instationnaire, dans laquelle la boucle de feedback dépend non seulement de l'état du système mais également du temps. Nous ne développerons pas plus en détail ces techniques et nous renvoyons le lecteur intéressé à ([Samson 91b]). L'idée originale que nous avons mis en œuvre afin de surmonter les difficultés introduites par la présence de contraintes non-holonomes procède d'une autre démarche que nous allons présenter dans le paragraphe suivant.

[1] deux roues motrices et directrices et deux roues stabilisatrices

### 2.1.3 La commande référencée vision pour un robot mobile

La mise en œuvre de techniques d'asservissement visuel appliquées à la robotique mobile montre que l'utilisation d'une caméra rigidement liée à la base mobile limite considérablement les possibilités d'évolution de celle-ci. Cela est dû principalement à la présence de liaisons non-holonomes qui impliquent l'utilisation de manœuvres au niveau de la base mobile pour positionner la caméra vis à vis d'une zone d'intérêt dans la scène. De ce fait, il paraît très intéressant de doter la caméra d'une certaine motricité indépendante de celle fournie par les degrés de liberté de la base mobile. Cette idée peut être réalisée en embarquant sur une base mobile un bras manipulateur dont l'effecteur porte la caméra. Un tel système offre des possibilités intéressantes, en effet :

- le couplage de la base mobile et de son manipulateur embarqué permet de réaliser des tâches robotiques plus élaborées (inspection, rondier,...).
- Le champ de vision du robot est plus important et ne nécessite plus obligatoirement de déplacements de la base pour percevoir son environnement local.

Nous allons voir que cet ensemble mécanique est primordial pour réaliser les tâches robotiques d'asservissement visuel. Cette coopération entre la base mobile et le bras porteur va simplifier les problèmes de commande car elle permet :

- d'utiliser les formalismes des fonctions de tâches et de la commande référencée capteur qui sont en principe destinés aux robots holonomes rigides. Un rappel sur ces formalismes est fait dans le chapitre suivant.
- de stabiliser l'ensemble mécanique (base mobile et bras) par retour d'état stationnaire.

L'ajout des degrés de liberté du manipulateur à ceux de la base mobile va permettre d'exprimer la commande dans le repère caméra plutôt que de contrôler l'orientation et la position du véhicule qui la déplace. Nous allons montrer qu'en raisonnant dans le repère caméra et en choisissant une structure du bras convenable, les contraintes non-holonomes n'interviennent plus dans la commande de l'ensemble mécanique. Evidemment, une telle approche ne permettra pas de contrôler indépendamment le mouvement de la base mobile et celui du bras. Cependant on peut montrer, grâce à des considérations géométriques simples, que le mouvement de la base demeure borné.

Ainsi la modélisation cinématique de l'ensemble va consister à exprimer un modèle global correspondant à la réunion du modèle de la base mobile et de celui du bras embarqué calculés séparément. Pour cela nous avons adopté une méthode de modélisation indépendante de la structure mécanique réelle du robot. Cette méthode nous permet d'obtenir des résultats généraux facilement applicables à différents types de structure. En parallèle de cette description, nous l'appliquerons à un modèle expérimental. Cette modélisation permet le passage du torseur cinématique de la caméra aux vitesses articulaires sur les différents axes de la structure mécanique.

## 2.2 Le modèle cinématique d'une base mobile

Nous calculons le modèle cinématique de notre base mobile en suivant le formalisme décrit dans les travaux récents de [Campion 93]. Cette approche, par l'utilisation d'un paramétrage adéquat, permet de modéliser les robots mobiles à roues d'une manière générique. Après avoir rappelé brièvement ce type de modélisation, nous l'appliquerons au cas d'un robot mobile de “type char”.

### 2.2.1 Modélisation générique des robots mobiles

#### 2.2.1.1 La posture d'un robot mobile

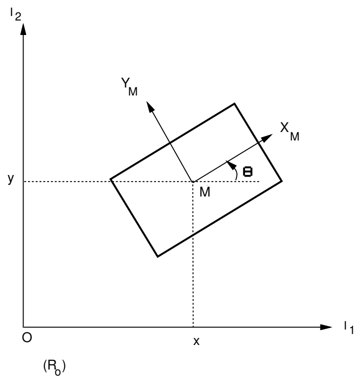



Figure 2.2 : Position d'un mobile dans un repère cartésien

On définit un point de référence M dont les coordonnées sont :

$$\vec{OM} = x\vec{I_1} + y\vec{I_2}$$

L'orientation du repère $(\vec{x_M}, \vec{y_M})$ par rapport au repère $(\vec{I_1}, \vec{I_2})$ est définie par $\theta$. A l'aide de ces paramètres, on définit le vecteur de posture de la base mobile $\underline{\xi}$ et la matrice de rotation $R(\theta)$ :

$$\underline{\xi} = \begin{pmatrix} x \\ y \\ \theta \end{pmatrix} \quad , \quad R(\theta) = \begin{pmatrix} cos\theta & sin\theta & 0 \\ -sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

#### 2.2.1.2 La description des roues

Les contraintes cinématiques au point de contact des roues sont définies pour différents types de roues :

- les roues fixes,
- les roues centrées orientables,
- les roues avec chasse (décentrées) orientables,
- les roues suédoises.

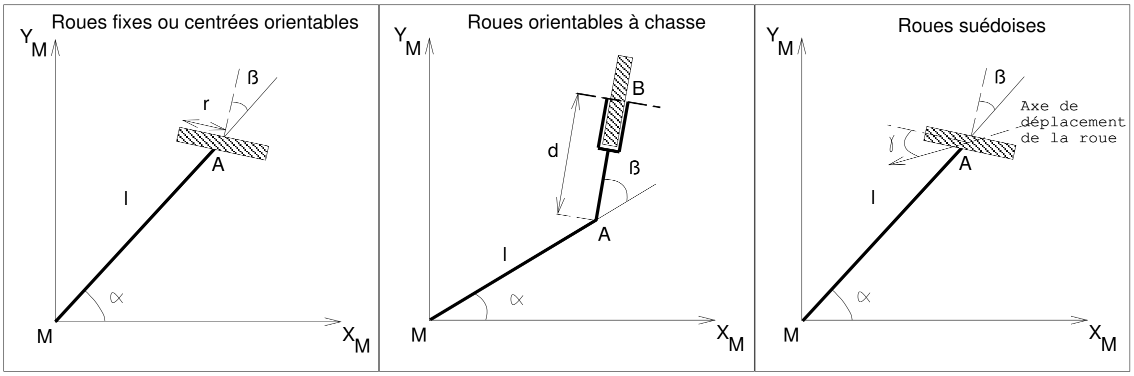



Figure 2.3 : Illustration des différents types de roues

Pour décrire chaque type de roues, il est nécessaire d'utiliser un paramétrage adéquat. Pour une roue fixe, par exemple, on définit quatre constantes $l$, $\alpha$, $r$, $\beta$ où :

- $(l, \alpha)$ sont les coordonnées polaires du point de contact de la roue dans le repère $(\vec{x}_M, \vec{y}_M)$,
- $r$ le rayon de la roue,
- $\beta$ l'orientation de la roue fixe.

L'angle de rotation de la roue $\varphi(t)$ est un paramètre qui dépend du temps. En utilisant cette description, les composantes de la vitesse au point de contact sont calculées et expriment deux contraintes :

- suivant le plan de la roue (condition de rotation pure)

$$\begin{bmatrix} -sin(\alpha+\beta) & cos(\alpha+\beta) & lcos\beta \end{bmatrix} R(\theta)\dot{\underline{\xi}} + r\dot{\varphi} = 0 \qquad (6)$$

- suivant le plan orthogonal à celui de la roue (condition de non-glissement latéral)

$$\begin{bmatrix} (cos(\alpha+\beta) & sin(\alpha+\beta) & lsin\beta \end{bmatrix} R(\theta)\dot{\underline{\xi}} = 0 \qquad (7)$$

Pour une roue centrée orientable, les équations sont les mêmes, mais le paramètre $\beta(t)$ devient lui aussi dépendant du temps. Pour une roue orientable décentrée, une autre constante est introduite pour exprimer la distance de décentrage. Pour une roue suédoise (roue omnidirectionnelle), les contraintes cinématiques sont réduites à une équation du fait de l'absence de la contrainte de glissement latéral. Les résultats sont résumés dans le tableau (2.4). L'illustration de cette classification est donnée dans la figure (2.3). Tout les détails sur ce travail de classification se trouvent bien sûr dans [Campion 93].

| Roues Fixes : | Ref. |
|---|---|
| $\begin{bmatrix} -sin(\alpha+\beta) & cos(\alpha+\beta) & lcos\beta \end{bmatrix} R(\theta)\dot{\underline{\xi}} + r\dot{\varphi} = 0$ | (c1) |
| $\begin{bmatrix} cos(\alpha+\beta) & sin(\alpha+\beta) & lsin\beta \end{bmatrix} R(\theta)\dot{\underline{\xi}} = 0$ | (c2) |
| Roues Centrées Orientables : | |
| $\begin{bmatrix} -sin(\alpha+\beta) & cos(\alpha+\beta) & lcos\beta \end{bmatrix} R(\theta)\dot{\underline{\xi}} + r\dot{\varphi} = 0$ | (c3) |
| $\begin{bmatrix} cos(\alpha+\beta) & sin(\alpha+\beta) & lsin\beta \end{bmatrix} R(\theta)\dot{\underline{\xi}} = 0$ | (c4) |
| Roues Orientables avec Chasse | |
| $\begin{bmatrix} -sin(\alpha+\beta) & cos(\alpha+\beta) & -lcos\beta \end{bmatrix} R(\theta)\dot{\underline{\xi}} + r\dot{\varphi} = 0$ | (c5) |
| $\begin{bmatrix} cos(\alpha+\beta) & sin(\alpha+\beta) & d+lsin\beta \end{bmatrix} R(\theta)\dot{\underline{\xi}} + d\dot{\beta} = 0$ | (c6) |
| Roues Suédoises | |
| $\begin{bmatrix} -sin(\alpha+\beta+\gamma) & cos(\alpha+\beta+\gamma) & lcos(\beta+\gamma) \end{bmatrix} R(\theta)\dot{\underline{\xi}} + rcos\gamma\dot{\varphi} = 0$ | (c7) |

Figure 2.4 : Equations des roues

#### 2.2.1.3 La mobilité et la classification des robots mobiles

Un robot est doté d'un nombre de roues $N = N_f + N_c + N_{oc} + N_{sw}$ où chaque composante de l'addition correspond aux nombres de roues fixes, centrées orientables, décentrées orientables et suédoises.

La configuration du robot est définie par :

- la posture $\underline{\xi}$,
- les coordonnées angulaires $\underline{\beta_c}(t), \underline{\beta_{oc}}(t)$,
- le vecteur de rotation des roues $\underline{\varphi}(t) = \begin{pmatrix} \underline{\varphi_f}(t) \\ \underline{\varphi_c}(t) \\ \underline{\varphi_{oc}}(t) \\ \underline{\varphi_{sw}}(t) \end{pmatrix}$

Le nombre de composantes du vecteur de configuration du robot est donc égal à :

$$N_f + 2N_c + 2N_{oc} + N_{sw} + 3.$$

Les contraintes cinématiques données dans le tableau (2.4) peuvent être réécrites sous la forme générale :

$$\begin{cases} J_1(\underline{\beta_c}, \underline{\beta_{oc}})R(\theta)\dot{\underline{\xi}} + J_2\dot{\underline{\varphi}} = 0 \\ C_1(\underline{\beta_c}, \underline{\beta_{oc}})R(\theta)\dot{\underline{\xi}} + C_2\dot{\underline{\beta_{oc}}} = 0 \end{cases} \tag{8}$$

avec les définitions suivantes :

$$J_1(\underline{\beta_c}, \underline{\beta_{oc}}) = \begin{pmatrix} J_{1f} \\ J_{1c}(\underline{\beta_c}) \\ J_{1oc}(\underline{\beta_{oc}}) \\ J_{1sw}(\underline{\beta_c}) \end{pmatrix}, J_2 = diag(r, rcos\gamma)$$

où $J_{1f}, J_{1c}, J_{oc}, J_{sw}$ sont respectivement des matrices de dimensions $(N_f \times 3)$, $(N_c \times 3)$, $(N_{oc} \times 3)$, $(N_{sw} \times 3)$ qui sont issues directement des contraintes (c1), (c3), (c5) et (c7) du tableau récapitulatif 2.4. $J_2$ est une matrice diagonale constante de dimension $(N \times N)$ où les termes diagonaux sont les rayons $r$ des roues exceptés pour les roues suédoises où le rayon est multiple de $cos\gamma$.

$$C_1(\underline{\beta_c}, \underline{\beta_{oc}}) = \begin{pmatrix} C_{1f} \\ C_{1c}(\underline{\beta_c}) \\ C_{1oc}(\underline{\beta_{oc}}) \end{pmatrix} \quad, C_2 = \begin{pmatrix} 0 \\ 0 \\ C_{2oc} \end{pmatrix}$$

où $C_{1f}, C_{1c}, C_{oc}$ sont respectivement trois matrices de dimensions $(N_f \times 3)$, $(N_c \times 3)$, $(N_{oc} \times 3)$, $(N_{sw} \times 3)$ qui sont issues directement des contraintes (c2), (c4), (c6) du tableau récapitulatif 2.4. $C_{2oc}$ est une matrice diagonale constante où les termes diagonaux sont les termes $d$ des roues à chasse orientables.

En considérant les $(N_f + N_c)$ premières contraintes de la deuxième équation de (8) on obtient alors :

$$\begin{cases} C_{1f}R(\theta)\underline{\dot{\xi}} & = 0 \\ C_{1c}(\underline{\beta_c})R(\theta)\underline{\dot{\xi}} & = 0 \end{cases} \tag{9}$$

Si l'on définit la matrice $C_1^*(\underline{\beta_c})$ telle que :

$$C_1^*(\beta_c) = \begin{pmatrix} C_{1f} \\ C_{1c}(\underline{\beta_c}) \end{pmatrix} \tag{10}$$

Les contraintes (9) impliquent que le vecteur $R(\theta)\underline{\dot{\xi}}$ appartient au noyau de la matrice $C_1^*(\underline{\beta_c})$.

Immédiatement on remarque que : $rang[C_1^*(\underline{\beta_c})] \leq 3$. De plus si l'on a $rang[C_1^*(\underline{\beta_c})] = 3$ alors $R(\theta)\underline{\dot{\xi}} = 0$ c'est à dire qu'aucun mouvement de la base n'est autorisé dans le plan. Les auteurs dans [Campion 93] font remarquer que les équations (9) ont une interprétation géométrique intéressante sur la structure du robot mobile. En effet la condition $rang[C_1^*(\underline{\beta_c})] \leq 2$ est équivalente géométriquement à avoir les axes de rotation horizontaux des roues fixes et centrées concourant au centre de rotation instantanée. Cette condition est illustrée en figure (2.5).

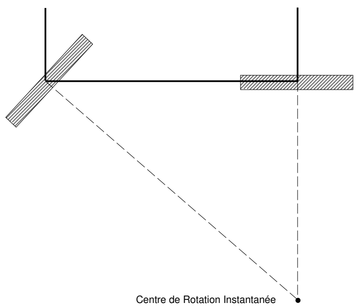



Figure 2.5 : Centre de Rotation Instantanée

Le degré de mobilité du robot introduit dans [Campion 93] est alors défini par :

$$\delta_m = dim\{Ker[C_1^*(\underline{\beta_c})]\} = 3 - rang[C_1^*(\underline{\beta_c})]$$

En ce qui concerne la matrice $C_{1f}$ si son rang est supérieur ou égal à deux, cela implique que le robot possède au moins deux roues fixes mais son centre de rotation est alors fixe dans le repère du robot. En pratique, il faut donc choisir $rang[C_{1f}] \leq 1$. En utilisant toutes les considérations présentées ci-dessus sur la structure d'un robot mobile, on peut dire qu' *un robot mobile est dit non-dégénéré si :*

$$rang[C_1^*(\underline{\beta_c})] = rang[C_{1f}] + rang[C_{1c}(\underline{\beta_c})] \leq 2$$

Ainsi dans [Campion 93], cinq classes de structures mécaniques non-singulières intéressantes sont définies telles que :

- $rang[C_{1f}] = i = 0$ ou $1$,
- $rang[C_{1c}(\beta_c)] = j = 0, 1$ ou $2$,

Les différentes classes de robots mobiles *non dégénérés* sont illustrées et présentées dans le tableau ci-dessous :

| | i = 0 | i = 1 |
|---|---|---|
| j = 0 | robot omnidirectionnel<br>$N_f = 0$ $N_c = 0$ | robot type *char*<br>$N_f \geq 1$ avec 1 axe commun $N_c = 0$ |
| j = 1 | tricycle à 2 roues folles<br>$N_f = 0$ $N_c = 1$ | tricycle à 2 roues fixes<br>$N_f \geq 1$ avec 1 axe commun $N_c \geq 1$ |
| j = 2 | base à roues orientables<br>$N_f = 0$ $N_c \geq 2$ | |

#### 2.2.1.4 Le modèle cinématique

La vitesse de posture $\dot{\underline{\xi}}(t)$ du robot mobile peut être formulée sous la forme d'une équation d'état ([Campion 93]). En effet, d'après (9) on a pu déduire que : $R(\theta)\dot{\underline{\xi}} \in Ker[C_1^*(\underline{\beta_c})]$. Cette condition est équivalente à dire que :*quelque soit t, il existe un vecteur $\underline{\eta}(t)$ tel que* :

$$\dot{\underline{\xi}} = R(\theta)^T \Sigma(\underline{\beta_c})\underline{\eta} \tag{11}$$

où les colonnes de $\Sigma(\underline{\beta_c})$ forment une base de $Ker[C_1^*(\underline{\beta_c})]$.

Dans le cas où le robot possède des roues centrées orientables, on peut augmenter l'équation d'état en définissant un vecteur $\underline{\zeta}$ pour obtenir :

$$\begin{cases} \dot{\underline{\xi}} = R(\theta)^T \Sigma(\underline{\beta_c})\underline{\eta} \\ \dot{\underline{\beta_c}} = \underline{\zeta} \end{cases} \tag{12}$$

Ces équations sont des représentations d'état du modèle cinématique de posture où les coordonnées de la posture $\underline{\xi}$ et, s'ils existent, les coordonnées angulaires $\underline{\beta_c}$ sont les variables d'états tandis que $\underline{\eta}$ et $\underline{\zeta}$ homogènes à des vitesses forment le vecteur de contrôle.

On remarque que la vitesse angulaire $\underline{\dot{\beta}_{oc}}$ et la vitesse de rotation $\underline{\dot{\varphi}}$ n'interviennent pas dans le modèle cinématique de posture. En utilisant (8), il vient directement :

$$\left\{ \begin{array}{l} \underline{\dot{\beta}}_{oc} = -C_{2oc}^{-1} C_{1oc}(\underline{\beta_{oc}}) R(\theta) \underline{\dot{\xi}} \\ \underline{\dot{\varphi}} = -J_2^{-1} J_1(\underline{\beta_c}, \underline{\beta_{oc}}) R(\theta) \underline{\dot{\xi}} \end{array} \right. \tag{13}$$

En combinant ces équations avec le modèle cinématique de posture (12), on obtient les équations d'états pour $\underline{\beta}_{oc}$ et $\underline{\varphi}$ qui s'écrivent :

$$\left\{ \begin{array}{l} \underline{\dot{\beta}}_{oc} = -C_{2oc}^{-1} C_{1oc}(\underline{\beta_{oc}}) \Sigma(\beta_c) \underline{\eta} \\ \underline{\dot{\varphi}} = -J_2^{-1} J_1(\underline{\beta_c}, \underline{\beta_{oc}}) \Sigma(\underline{\beta_c}) \underline{\eta} \end{array} \right. \tag{14}$$

En définissant $\underline{z}$ le vecteur de configuration du robot mobile tel que : $\underline{z} = \begin{pmatrix} \underline{\xi} & \underline{\beta_c} & \underline{\beta_{oc}} & \underline{\varphi} \end{pmatrix}^T$ le modèle cinématique s'écrit :

$$\underline{\dot{z}} = S(\underline{z}) \underline{u} \quad \text{avec} \quad \underline{u} = \begin{pmatrix} \underline{\eta} \\ \underline{\zeta} \end{pmatrix} \tag{15}$$

$$\text{où } S(\underline{z}) = \begin{pmatrix} R^T(\theta)\Sigma(\beta_c) & 0 \\ 0 & I \\ -C_{2oc}^{-1} C_{1oc}(\beta_{oc}) \Sigma(\beta_c) & 0 \\ -J_2^{-1} J_1(\beta_c, \beta_{oc}) \Sigma(\beta_c) & 0 \end{pmatrix} \tag{16}$$

Ainsi ce modèle cinématique général permet de modéliser la plupart des types de robots mobiles que l'on rencontre en pratique. Nous illustrons, dans la partie qui suit, son application pour un robot mobile de type “char”. Ce formalisme nous permettra ensuite de modéliser une base mobile dotée d'un bras manipulateur de manière générique.

### 2.2.2 Application à une base mobile de type “Char”

#### Les notations utilisées pour notre base mobile

Le robot mobile que nous avons réalisé pour valider les approches réactives d'asservissement visuel est de type “char” (cf. figure 2.6). La base mécanique comporte deux roues motrices à l'arrière et deux “boules” stabilisatrices à l'avant. Nous considérons pour la modélisation que les deux roues boules assurent un contact ponctuel parfait sans contraintes cinématiques. Toutes les notations concernant le robot mobile sont illustrées sur la figure (2.6) et présentées en annexe A.1. En utilisant le formalisme décrit auparavant, nous pouvons définir les paramètres de modélisation de notre robot qui comporte deux roues

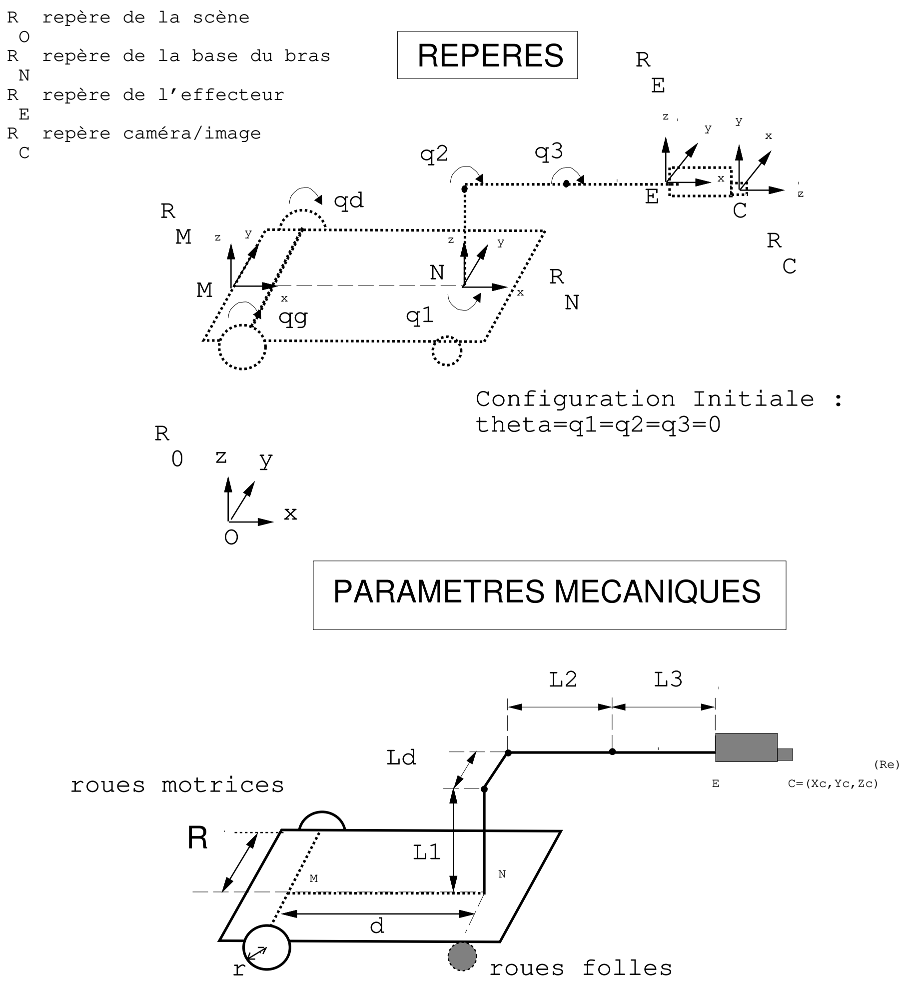



Figure 2.6 : Description du robot mobile

fixes ($N_f = 2$). Le point de référence est situé à l'entre-axe des roues motrices (cf. figure 2.7).

| $l$ | $\alpha$ | $r$ | $\beta$ | $\varphi_c(t)$ |
|---|---|---|---|---|
| $R$ | 0 | $r$ | 0 | $q_d$ |
| $R$ | $\pi$ | $r$ | 0 | $q_g$ |

Les matrices des contraintes cinématiques s'écrivent alors dans notre cas :

$$J_1 = \begin{pmatrix} 0 & 1 & R \\ 0 & -1 & R \end{pmatrix} , \; J_2 = \begin{pmatrix} r & 0 \\ 0 & r \end{pmatrix}$$

$$C_1 = \begin{pmatrix} 1 & 0 & 0 \\ -1 & 0 & 0 \end{pmatrix} , \; C_2 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

On peut vérifier que la classe du robot correspond bien à un type “char” car

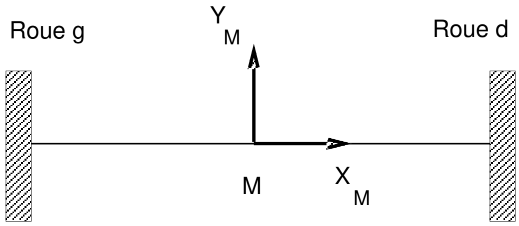



Figure 2.7 : Roues motrices du robot mobile version "char"

on obtient :

$$rang[C_{1f}] = 1 \text{ , } rang[C_{1c}(\underline{\beta_c})] = 0 \text{ et } \delta_m = 2.$$

On choisit un vecteur de commande en vitesse au point de référence $M$ composé d'une vitesse linéaire suivant l'axe $Y_M$ et de la vitesse de rotation $\dot{\theta}$, soit
$\underline{u} = \begin{pmatrix} v & \dot{\theta} \end{pmatrix}^T$. Le vecteur d'état $\underline{z}$ est défini par $\underline{z} = \begin{pmatrix} x & y & \theta & q_d & q_g \end{pmatrix}^T$. Le modèle cinématique de configuration de la base mobile s'exprime ainsi sous la forme :

$$\dot{\underline{z}} = S(\underline{z})\underline{u} \tag{17}$$

La matrice $S(\underline{z})$ a alors pour expression :

$$S(\underline{z}) = \begin{pmatrix} -sin\theta & 0 \\ cos\theta & 0 \\ 0 & 1 \\ \frac{1}{r} & \frac{R}{r} \\ \frac{1}{r} & -\frac{R}{r} \end{pmatrix} \tag{18}$$

Pour la suite de nos calculs, nous nous intéressons à l'état articulaire du robot. A partir du modèle cinématique de configuration (18), on extrait un modèle cinématique où l'état du robot $\underline{q_m}$ est réduit à $\underline{q_m} = \begin{pmatrix} q_d & q_g \end{pmatrix}^T$. Le modèle réduit s'écrit alors :

$$\underline{q_m} = J_m^{-1}(\underline{q_m})\underline{u} \tag{19}$$

avec

$$J_m^{-1}(\underline{q_m}) = \begin{pmatrix} \frac{1}{r} & \frac{R}{r} \\ \frac{1}{r} & -\frac{R}{r} \end{pmatrix} \tag{20}$$

Le modèle direct a pour expression :

$$J_m(\underline{q_m}) = \begin{pmatrix} \frac{r}{2} & \frac{r}{2} \\ \frac{r}{2R} & -\frac{r}{2R} \end{pmatrix} \tag{21}$$

Ce modèle aurait pu être bien sûr obtenu de manière quasi-immédiate en utilisant les relations fondamentales de la cinématique. Mais l'approche décrite, en plus de proposer une classification des différents robots mobiles suivant leur structure mécanique, permet d'envisager l'automatisation de leur modélisation.

## 2.3 Le modèle cinématique du porteur caméra

Après avoir rappelé très brièvement le principe de la modélisation géométrique et cinématique appliqué aux robots manipulateurs, nous décrirons les calculs à effectuer pour modéliser de façon générale un ensemble mécanique constitué par une base mobile et un manipulateur. Nous appliquerons ces résultats à un robot mobile de type "char" muni d'un bras manipulateur à trois degrés de liberté.

### 2.3.1 Les techniques de modélisation

#### 2.3.1.1 Le modèle géométrique

La situation (orientation et position) , notée $\underline{X}$, d'un repère lié à l'effecteur du manipulateur est donnée par son modèle géométrique. Ce modèle, régi par des équations holonomes, dépend des positions articulaires du robot, représentées par le vecteur $\underline{q}$ :

$$\underline{X} = \Phi(\underline{q}) \tag{22}$$

D'une manière désormais classique, le changement de repère entre la base du bras ($R_n$) et l'effecteur ($R_e$) s'obtient en utilisant la méthode de **Denavit-Hartenberg** ([Dombre 88]). Elle permet de passer d'un repère $R_i$ à un repère $R_{i+1}$ par les transformations géométriques suivantes :

- rotation autour de l'axe $z_i$ d'angle $\theta_i$,
- translation suivant l'axe $z_i$ de $d_i$,
- translation suivant l'axe $x_{i+1}$ de $a_i$,
- rotation autour de l'axe $x_{i+1}$ d'angle $\alpha_i$.

La matrice de transformation entre les deux repères est alors :

$$\begin{pmatrix} cos\theta_i & -sin\theta_i cos\alpha_i & sin\theta_i sin\alpha_i & a_i cos\theta_i \\ sin\theta_i & cos\alpha_i cos\theta_i & -sin\alpha_i cos\theta_i & a_i sin\theta_i \\ 0 & sin\alpha_i & cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{23}$$

#### 2.3.1.2 Le modèle cinématique

Le modèle cinématique, que l'on obtient par dérivation du modèle précédent, relie les vitesses articulaires $\underline{\dot{q}}$ au torseur cinématique exprimant la vitesse $\underline{\dot{X}}$ d'un repère attaché à l'effecteur du manipulateur :

$$\underline{\dot{X}} = J(\underline{q})\underline{\dot{q}} \tag{24}$$

### 2.3.2 La modélisation du bras porteur

#### 2.3.2.1 Le modèle géométrique du bras porteur

Dans le cas de la structure mécanique de notre bras (cf. figure 2.6), le changement de repères de $R_e$ à $R_n$, est décomposé en quatre transformations successives correspondant aux paramètres de Denavit-Hartenberg suivant :

$$\begin{cases} \theta_1 = q_1 \\ d_1 = l_1 \\ a_1 = 0 \\ \alpha_1 = \frac{\Pi}{2} \end{cases} \begin{cases} \theta_{1d} = 0 \\ d_{1d} = l_d \\ a_{1d} = 0 \\ \alpha_{1d} = 0 \end{cases} \begin{cases} \theta_2 = q_2 \\ d_2 = 0 \\ a_2 = l_2 \\ \alpha_2 = 0 \end{cases} \begin{cases} \theta_3 = q_3 \\ d_3 = 0 \\ a_3 = l_3 \\ \alpha_3 = -\frac{\Pi}{2} \end{cases} \tag{25}$$

Afin de simplifier l'écriture, posons :

$$\begin{array}{l} sin\theta_i = s_i \\ cos\theta_i = c_i \\ sin(\theta_i + \theta_j) = s_{ij} \\ cos(\theta_i + \theta_j) = c_{ij} \end{array} \tag{26}$$

Soit $\underline{X}^{(R)}$ le vecteur représentant la position d'un point en coordonnées homogènes exprimés dans un repère $R$, on arrive finalement à :

$$\underline{X}^{(R_n)} = MG(q_1, q_2, q_3)\underline{X}^{(R_c)} \tag{27}$$

La matrice homogène $MG(q_1, q_2, q_3)$ s'exprime sous la forme :

$$\begin{pmatrix} -s_1 & -c_1 s_{23} & c_1 c_{23} & c_1(l_2 c_2 + (l_3 + x_c)c_{23} - z_c s_{23}) + s1(l_d - y_c) \\ c_1 & -s_1 s_{23} & s_1 c_{23} & s_1(l_2 c_2 + (l_3 + x_c)c_{23} - z_c s_{23}) + c1(-l_d + y_c) \\ 0 & c_{23} & s_{23} & l_1 + l_2 s_2 + (l_3 + x_c)s_{23} + z_c c_{23} \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{28}$$

Dans la pratique, il est souvent difficile à l'utilisateur de se représenter concrètement le changement de repère en rotation exprimé à partir des positions articulaires $(q_1, q_2, q_3)$. On raisonnera donc, au niveau de la spécification de tâche, au moyen de paramétrages plus "naturels". Parmi ceux-ci, le plus couramment utilisé, est celui en : *Lacet, Tangage, Roulis* qui se définit par la succession des rotations suivantes :

- le lacet qui est une rotation de $\Psi$ autour de l'axe $z$,
- le tangage qui est une rotation de $\Theta$ autour du "nouvel axe" $y$,
- le roulis qui est une rotation de $\Phi$ autour du "nouvel axe" $x$.

Le modèle géométrique direct est alors exprimé par la matrice :

$$\begin{pmatrix} cos\Psi cos\Theta & cos\Psi sin\Theta sin\Phi - sin\Psi cos\Phi & cos\Psi sin\Theta cos\Phi + sin\Psi sin\Phi & x \\ sin\Psi cos\Theta & sin\Psi sin\Theta sin\Phi + cos\Psi cos\Phi & sin\Psi sin\Theta cos\Phi - cos\Psi sin\Phi & y \\ -sin\Theta & cos\Theta sin\Phi & cos\Theta cos\Phi & z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{29}$$

L'équivalence entre les deux représentations nous fournit les relations entre $\Psi$, $\Theta$, $\Phi$ et $q1$, $q2$, $q3$ :

$$\begin{cases} \Psi = q_1 + \frac{\Pi}{2} \\ \Theta = 0 \\ \Phi = \frac{\Pi}{2} - (q_2 + q_3) \end{cases} \tag{30}$$

La position de l'effecteur est simplement donnée par le système d'équations :

$$\begin{cases} x = c_1(l_2c_2 + (l_3 + x_c)c_{23} - z_c s_{23}) + s_1(l_d - y_c) \\ y = s_1(l_2c_2 + (l_3 + x_c)c_{23} - z_c s_{23}) + c_1(-l_d + y_c) \\ z = l_1 + l_2 s_2 + (l_3 + x_c)s_{23} + z_c c_{23} \end{cases} \tag{31}$$

#### 2.3.2.2 Le modèle cinématique du bras porteur

On choisit donc comme vecteur de représentation, exprimé dans $R_n$ le vecteur $\underline{X}_{rtl}$ où:

$$\underline{X}_{rtl} = \begin{pmatrix} \underline{X_p} \\ \underline{\theta}_{rtl} \end{pmatrix} \text{ avec } \underline{\theta}_{rtl} = \begin{pmatrix} \Psi \\ \Theta \\ \Phi \end{pmatrix} \text{ et sa position } \underline{X_p} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} \tag{32}$$

On remarque que d'après (30), l'angle de tangage est toujours nul. Ceci traduit une contrainte mécanique dans la structure du bras.

Le vecteur des vitesses articulaires est $\underline{\dot{q}}_{bras} = \begin{pmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{pmatrix}$. On calcule la matrice jacobienne $J_b$ telle que :

$$\underline{\dot{X}}_{rtl} = J_b \underline{\dot{q}}_{bras} = \begin{pmatrix} J_p \\ J_o \end{pmatrix} \underline{\dot{q}}_{bras} \tag{33}$$

En dérivant la position (31) et l'orientation (30) de l'effecteur, la matrice jacobienne $J_b$ s'écrit :

$$\begin{pmatrix} -s_1(l_2c_2 + (l_3 + x_c)c_{23} - z_cs_{23}) + c1(l_d - y_c) & c_1(-l_2s_2 - (l_3 + x_c)s_{23} - z_cc_{23}) & c_1(-(l_3 + x_c)s_{23} - z_cc_{23}) \\ c_1(l_2c_2 + (l_3 + x_c)c_{23} - z_cs_{23}) - s1(-l_d + y_c) & s_1(-l_2s_2 - (l_3 + x_c)s_{23} - z_cc_{23}) & s_1(-(l_3 + x_c)s_{23} - z_cc_{23}) \\ 0 & l_2c_2 + (l_3 + x_c)c_{23} - z_cs_{23} & (l_3 + x_c)c_{23} - z_cs_{23} \\ 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & -1 & -1 \end{pmatrix} \tag{34}$$

## 2.4 Le modèle cinématique de l'ensemble base mobile et bras porteur

Partant des modèles précédemment calculés pour la base mobile et le bras manipulateur embarqué, nous allons maintenant exprimer le modèle global liant le repère caméra à l'ensemble des variables articulaires de la base mobile et du bras.

### 2.4.1 Le modèle générique

Il est nécessaire de définir un repère lié à l'effecteur du bras porteur. Dans notre cas nous placerons l'origine de ce repère de telle sorte qu'il corresponde avec le centre optique de la caméra fixée sur l'effecteur. Ce point est noté $C$. On utilise les équations fondamentales de la cinématique afin d'exprimer la vitesse de ce repère :

$$\begin{cases} \vec{V}_{absolue} = \vec{V}_{entrainement} + \vec{V}_{relative} \\ V(M) = V(O) + \Omega \wedge \vec{OM} \end{cases} \tag{35}$$

Ces équations appliquées à l'ensemble mécanique s'expriment dans le repère $R_M$ :

$$\begin{cases} \vec{V_C} = \vec{V_M} + \Omega_{R_M/R_O} \wedge \vec{MC} + \vec{V}_{C/R_M} \\ \Omega_{R_C/R_O} = \Omega_{R_M/R_O} + \Omega_{R_C/R_M} \end{cases} \tag{36}$$

Grâce aux calculs antérieurs, on connaît la modélisation cinématique du bras qui peut s'écrire :

$$\begin{pmatrix} \vec{V_M} \\ \Omega_{R_C/R_M} \end{pmatrix} = J_b . \underline{\dot{q}}_{bras}$$

Le modèle de la base mobile est donné par l'équation (18). On choisit le point M de la base mobile comme point de référence. La jacobienne $J_M(\underline{z})$ de dimension $2 \times 2$ peut se décomposer sous la forme :

$$J_M(\underline{z}) = \begin{pmatrix} J_{MV} \\ J_{M\dot{\theta}} \end{pmatrix} \tag{37}$$

Les vitesses en translation et rotation de la base mobile peuvent s'exprimer sous la forme :

$$\vec{V_M} = J_{MV} . \begin{pmatrix} \dot{q}_d \\ \dot{q}_g \end{pmatrix} \ , \ \Omega_{R_M/R_O} = J_{M\dot{\theta}} . \begin{pmatrix} \dot{q}_d \\ \dot{q}_g \end{pmatrix} \tag{38}$$

En définissant un vecteur $\underline{q} = \begin{bmatrix} q_d & q_g & \underline{q_{bras}} \end{bmatrix}$, l'équation (36) peut alors facilement se mettre sous la forme matricielle :

$$\underline{\dot{X}}_{rtl} = \left( \begin{array}{c|c} J_{M\dot{\theta}} \wedge \vec{MC} + J_{MV} & J_b \\ J_{M\dot{\theta}} & \end{array} \right) . \underline{\dot{q}} \tag{39}$$

Cette matrice jacobienne de forme générale est notée $J$.

### 2.4.2 Application de cette modélisation à notre robot

La description de la structure mécanique est faite sur la *figure 2.6*. Pour la simulation, nous avons simplifié le modèle du bras en faisant correspondre l'origine du repère effecteur avec celle du repère caméra ($x_c = y_c = z_c = l_d = 0$). Le modèle cinématique a alors comme expression :

$$J = \begin{pmatrix} -\frac{r}{2R} s_1(l_2 c_2 + l_3 c_{23}) + \frac{r}{2} & \frac{r}{2R} s_1(l_2 c_2 + l_3 c_{23}) + \frac{r}{2} & -s_1(l_2 c_2 + l_3 c_{23}) & c_1(-l_2 s_2 - l_3 s_{23}) & -l_3 c_1 s_{23} \\ \frac{r}{2R} c_1(l_2 c_2 + l_3 c_{23} + d) & -\frac{r}{2R} c_1(l_2 c_2 + l_3 c_{23} + d) & c_1(l_2 c_2 + l_3 c_{23}) & s_1(-l_2 s_2 - l_3 s_{23}) & -l_3 s_1 s_{23} \\ 0 & 0 & 0 & l_2 c_2 + l_3 c_{23} & l_3 c_{23} \\ \frac{r}{2R} & -\frac{r}{2R} & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & -1 \end{pmatrix} \tag{40}$$

On peut supprimer la ligne nulle équivalente à $\dot{\Theta} = 0$ qui permet d'obtenir la matrice carrée $J_c$.

On peut alors en déduire les vitesses articulaires à partir du torseur cinématique de la caméra exprimée dans son repère :

$$\underline{\dot{q}} = J_c^{-1} M_p T_c^{(R_c)} \tag{41}$$

Le calcul de la matrice de passage $M_p$ est donné en annexe A.2.

#### 2.4.2.1 Etude des singularités

Il est important de connaître les configurations singulières d'un robot car elles jouent un rôle primordial dans son contrôle. Il existe une singularité cinématique en $\underline{q_s}$ si la jacobienne du robot en $\underline{q_s}$ n'est pas de rang plein (pour une matrice jacobienne carrée si son déterminant est nul).

En utilisant cette définition générale, dans le cas présent, le déterminant de la jacobienne du robot a pour expression :

$$Det(J_c) = -\frac{dl_2 r^2 c_2}{2R} \tag{42}$$

Les configurations singulières pour le robot sont définies pour : $\theta_2 = \frac{\pi}{2} + k\pi$. Elles correspondent physiquement à des singularités limites : le deuxième segment du bras étant vertical, on ne peut plus commander par exemple une montée verticale du repère caméra. Dans ce déterminant on retrouve des valeurs interdites pour les paramètres mécaniques telles que :

- $d \neq 0$. La base du bras ne doit pas être située sur l'entre-axe du chariot pour que l'angle de rotation du chariot et $q_1$ ne soient pas redondants,
- $l_2 \neq 0$ pour éviter une redondance pour $q_2$ et $q_3$.

Finalement, on remarque que l'influence des contraintes non-holonomes de la base sur les configurations singulières de l'ensemble du robot est nulle. Ces contraintes ne limitent donc pas la commande choisie.

## 2.5 Résultats de simulation

Pour vérifier et expérimenter cette modélisation, nous avons réalisé un simulateur avec le logiciel *Mathematica* (logiciel de calcul formel doté de possibilités graphiques importantes).

Le logiciel simule le déplacement de l'ensemble base mobile et bras sous l'action d'un torseur cinématique spécifié dans le repère caméra par l'utilisateur. Les résultats graphiques de simulation représentent la trajectoire du robot en vues 2D et 3D.

La simulation permet de vérifier que la commande calculée est capable de contrôler les mouvements de la caméra en translation ou rotation suivant un axe quelconque. Par exemple, si le mouvement spécifié est une translation suivant l'optique de la caméra, naïvement on peut dire que le robot mobile effectuera une "manœuvre" pour satisfaire le mouvement caméra spécifié. Cet exemple de trajectoire est illustré sur la figure (2.8). En combinant translation et rotation, on peut définir des trajectoires circulaires pour le mobile ou le bras tel que sur la figure (2.9). La trajectoire spécifiée de la caméra est un cercle dans un plan horizontal avec son axe optique dirigé vers le centre de ce cercle ; cette trajectoire correspond à un torseur cinématique constant $T_C^{R_C} = (-1, 0, 0, 0, 1, 0)^T$ par exemple.

Un ensemble de tests a permis de valider et d'illustrer le modèle cinématique calculé. Les résultats graphiques complets sont donnés en annexe A.3.

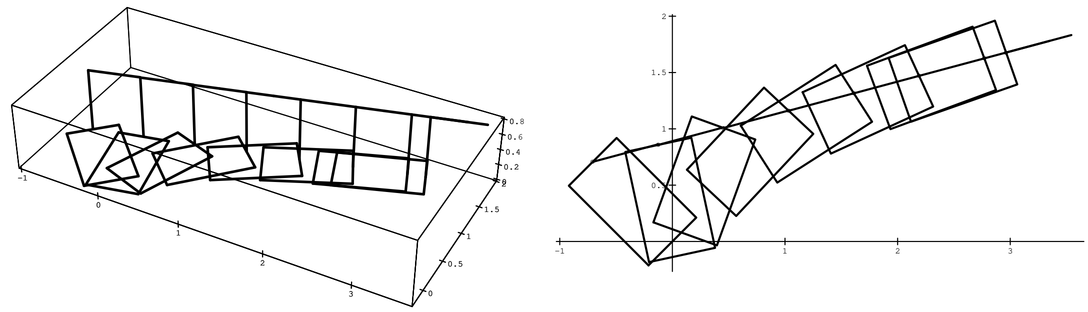


Figure 2.8 : Translation suivant l'axe optique de la caméra

## 2.6 Conclusion

Dans ce chapitre, nous avons montré que la présence de contraintes non-holonomes ne permettait pas d'appliquer de façon directe les résultats obtenus en asservissement visuel dans le cas de bras manipulateurs holonomes. L'adjonction des degrés de liberté supplémentaires fournis par un bras manipulateur embarqué sur la base mobile et portant la caméra permet à la fois de surmonter ces difficultés et d'offrir une souplesse accrue au niveau des tâches référencées vision. Nous montrons qu'en choisissant une structure adaptée du bras, le jacobien de l'ensemble base mobile + bras est de rang plein et ne présentent pas de singularité gênante pour des tâches d'asservissement visuel définies dans le repère caméra.

On constate qu'il n'est pas nécessaire de pré-positionner la base mobile vis à vis du motif visuel dans la scène avant de lancer la tâche référencée vision sur ce

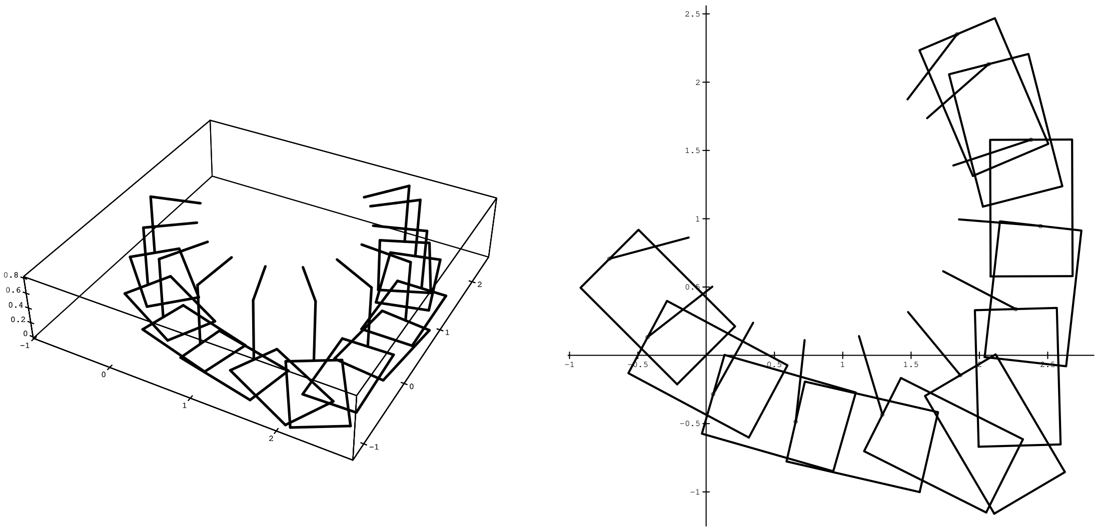


Figure 2.9 : Trajectoire circulaire de la caméra

motif mais qu'il suffit simplement que le motif soit dans l'image. La commande placera elle-même la base mobile pour satisfaire le torseur caméra spécifié par l'asservissement visuel. En simulation, si la trajectoire à suivre pour la caméra est une droite ou un arc de cercle, on remarquera que la base mobile "manœuvre" pour ensuite suivre la trajectoire nominale. Ceci peut constituer un handicap dans un environnement encombré. Il est quand même à noter que le mouvement non contrôlé de la base mobile est toujours inscrit dans un disque centré sur l'axe vertical du repère caméra et de rayon $\rho < \sqrt{(l_2 + l_3 + d + \epsilon)^2 + R^2}$.

Le contrôle effectif de l'orientation de la base durant la tâche référencée vision est un problème qui reste ouvert. De très récents travaux sur la commande des robots mobiles ([Samson 93]) et les "manipulateurs mobiles" ([Mishck 92], [Wang 93]) ouvre deux voies à explorer pour résoudre ce type de problème. L'utilisation d'un bras manipulateur à axes redondants devrait permettre d'inclure le contrôle de l'orientation de la base dans une tâche secondaire. Une deuxième approche plus originale devrait tenter de mixer la commande à retour d'état instationnaire sur la base et l'asservissement visuel sur le bras.

Malgré le travail qui reste à faire, nous avons donc montré et illustré l'utilité de la coopération d'une base mobile et d'un manipulateur pour diriger une caméra dans la réalisation de tâches d'asservissement visuel pour lesquelles on dispose de formalismes (commande-référencée-vision et fonction de tâche) bien établis. Dans la partie suivante, nous allons présenter ces formalismes et les appliquer dans le cadre de notre étude.

# Chapitre 3

# La Commande référencée vision pour un robot mobile

## 3.1 Rappels sur la Commande référencée vision

Nous présentons dans cette partie, les principaux résultats des travaux sur la commande référencée vision qui ont fait l'objet d'une thèse ([Chaumette 90]) et de nombreuses publications ([Espiau 92], [Chaumette 91a], [Chaumette 91b], [Rives 89]). Ces rappels ne pourront être que sommaires et pour une connaissance plus approfondie du sujet, le lecteur devra se rapporter à la bibliographie donnée.

Nous évoquerons donc simplement les principaux aspects de la Commande référencée vision qui sont :

- la phase de modélisation des informations visuelles qui permet de modéliser l'interaction entre la caméra et son environnement local,
- la classification des différentes tâches réalisables par asservissement visuel,
- l'intégration de cette modélisation dans le domaine de la Commande référencée capteur et l'approche fonction de tâche, formalismes détaillés dans l'ouvrage [Samson 90a].

Les rappels de ces travaux sont indispensables puisque notre travail a consisté à étendre et à implémenter ces résultats dans le cadre de la robotique mobile.

### 3.1.1 Modélisation de l'interaction caméra - environnement local

#### 3.1.1.1 Le modèle caméra

On considère un modèle géométrique idéal de la caméra présenté en figure (3.1). Tout point ayant pour coordonnées $\underline{x} = (x, y, z)^T$, dans le repère caméra $(R_c)$, se projette par projection perspective sur le plan image en M : $\underline{X} = (X, Y, f)^T$ avec :

$$\underline{X} = \frac{f}{z}\,\underline{x} \tag{43}$$

Pour la suite de nos calculs, dans le but de les simplifier, nous considérerons $f = 1$.

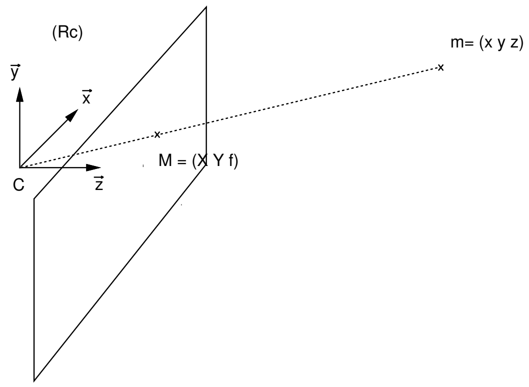



Figure 3.1 : Modèle de la caméra

#### 3.1.1.2 L'interaction entre la caméra et son environnement

Du fait du modèle de caméra utilisé, nous nous intéressons uniquement à ses caractéristiques géométriques sans prendre en compte les phénomènes photométriques liés à la formation de l'image. On considère un repère $(R_c)$ attaché au point C de la caméra qui interagit avec un environnement local associé à un repère $(R_T)$.

La situation du robot portant la caméra, exprimée dans un repère $(R_R)$ est un élément $\underline{r}$ de $SE3$ (Special Euclidean Group) isomorphe au groupe des matrices homogènes $4 \times 4$. La caméra $C$ est liée au système mécanique par plusieurs degrés de liberté dont les coordonnées articulaires $\underline{q}$ constituent une carte de $SE3$.

Sous l'hypothèse du modèle d'une caméra "purement géométrique", son image est totalement caractérisée par la position relative entre $(R_C)$ et $(R_T)$, le type et le positionnement des primitives géométriques 3D présentes dans la scène.

En utilisant cette hypothèse, le signal de sortie du capteur peut être modélisé comme une fonction multidimensionnelle $s(\underline{r}, t)$ qui est entièrement définie par la nature du capteur et des objets dans la scène, la position du robot $\underline{r}$ et le temps $t$ représentant les mouvements propres des objets dans l'environnement local lié au repère $(R_T)$. Cette fonction est une application de l'espace de configuration du robot dans l'espace des signaux capteurs. Pour différents types de capteurs (proximètre, vision, force, ...), une forme analytique de cette fonction peut être calculée. Dans le cas du capteur visuel, le vecteur $\underline{s}$ peut être constitué par un paramétrage des primitives géométriques de l'environnement (points, lignes, cercles...) projetées dans l'image. Du fait des propriétés de l'espace des torseurs (on conseille toujours de se reporter à [Chaumette 90], [Espiau 92] pour plus de détails), la forme différentielle de $s(\underline{r}, t)$ peut s'exprimer sous la forme d'un produit de torseur :

$$\dot{s} = H \bullet T_{CT} \tag{44}$$

où :

- $H$ est un torseur qui caractérise totalement l'interaction entre le capteur et l'environnement. On l'appelle "torseur d'interaction",
- $T_{CT}$ est le torseur cinématique représentant la vitesse relative de la scène par rapport à la caméra,
- $\bullet$ est le produit de torseurs.

En considérant la représentation matricielle $L^T$ du torseur d'interaction $H$ exprimé dans le repère caméra $(R_C)$, on obtient :

$$\dot{s} = L^T \, T_{CT} \tag{45}$$

Pour le cas de primitives géométriques dans l'image pouvant être décrite par une équation paramétrique, une expression analytique de la matrice d'interaction $L^T$ peut être obtenue (cf [Chaumette 90]). On rappellera simplement ici la méthode de calcul d'une matrice d'interaction et l'application aux primitives qui nous intéressent.

#### 3.1.1.3 Méthode de calcul du torseur d'interaction

Soit le motif $p$ de la scène qui se projette en un motif $P$ dans l'image. On note $\underline{p}$ et $\underline{P}$ leurs paramétrages respectifs.

Les primitives géométriques du motif dans la scène satisfont une équation paramétrique de la forme :

$$h(\underline{x}, \underline{p}) = 0 \tag{46}$$

où $h$ caractérise la nature de la primitive et $\underline{p}$ sa paramétrisation. En utilisant le modèle de la caméra (43), on peut écrire :

$$h(z\underline{X}, \underline{p}) = 0 \tag{47}$$

soit :

$$h'(\underline{X}, z, \underline{p}) = 0 \tag{48}$$

Le théorème des fonctions implicites permet alors de dire, sous la condition $\frac{\partial h'}{\partial z} \neq 0$, qu'autour de la solution $\underline{x}_0$ de (46), il existe une fonction unique $\mu$ telle que :

$$z = \mu(\underline{X}, \underline{p}) \tag{49}$$

Pour les différentes types de primitives géométriques considérées (points, droites et primitives planes) on calcule la fonction $\mu$ et la projection dans l'image de (47) pourra alors s'exprimer sous la forme :

$$g(\underline{X}, \underline{P}) = 0 \tag{50}$$

où $\underline{P}$ représente la paramétrisation de la primitive dans l'image.

Le motif $g$ dans l'image est donc la projection du motif $h$ dans la scène. En utilisant l'hypothèse de rigidité des solides 3D de l'environnement, soit $\dot{g} = 0$, on peut dériver (50) et obtenir :

$$\frac{\partial g}{\partial \underline{P}}(\underline{X}, \underline{P})\ \underline{\dot{P}} = -\frac{\partial g}{\partial \underline{X}}(\underline{X}, \underline{P})\ \underline{\dot{X}}\ ,\ \forall \underline{X} \in P \tag{51}$$

La dérivation de (43) conduit aux équations, bien connues, du "flot-optique" :

$$\underline{\dot{X}} = L^T_{of}(\underline{X}, z)\ T_{CT} \tag{52}$$

avec :

$$L^T_{of} = \begin{pmatrix} -1/z & 0 & X/z & XY & -(1+X^2) & Y \\ 0 & -1/z & Y/z & 1+Y^2 & -XY & -X \end{pmatrix} \tag{53}$$

En utilisant (49) dans (53), il vient :

$$L^T_{of}(\underline{X}, z) = L'^T_{of}(\underline{X}, \underline{p}) \tag{54}$$

En utilisant (51), (52), (54), on obtient :

$$\frac{\partial g}{\partial \underline{P}}(\underline{X}, \underline{P})\ \underline{\dot{P}} = -\frac{\partial g}{\partial \underline{X}}(\underline{X}, \underline{P})\ L'^T_{of}(\underline{X}, \underline{p})\ T_{CT} \tag{55}$$

On peut donc exprimer la variation des paramètres de la primitive observée dans l'image en fonction du mouvement de la caméra :

$$\dot{\underline{P}} = -\left[\ \frac{\partial g}{\partial \underline{P}}(\underline{X}, \underline{P})\ \right]^{-1} \frac{\partial g}{\partial \underline{X}}(\underline{X}, \underline{P})\ L'^{T}_{of}(\underline{X}, \underline{p}).T_{CT} = L^{T}.T_{CT} \tag{56}$$

La matrice $L^T$ recherchée est de dimension $m \times 6$. Sous réserve que la représentation de la primitive dans l'image est minimale ($det[\frac{\partial g}{\partial \underline{P}}] \neq 0$), on peut calculer $L^T$ en choisissant $m$ points $\underline{X}_i$ et en résolvant :

$$\left\{ \begin{array}{rcl} \frac{\partial g}{\partial \underline{P}}(\underline{X}_i, \underline{P}) & = & \alpha_i^T(\underline{P}) \\ -\frac{\partial g}{\partial \underline{X}}(\underline{X}_i, \underline{P}) L'^{T}_{of}(\underline{X}_i, \underline{p}) & = & \beta_i^T(\underline{p}, \underline{P}) \end{array} \right. \tag{57}$$

On obtient alors la matrice d'interaction :

$$L^T(\underline{p}, \underline{P}) = \begin{pmatrix} \alpha_1^T \\ \vdots \\ \alpha_m^T \end{pmatrix}^{-1} (\underline{P}) \begin{pmatrix} \beta_1^T \\ \vdots \\ \beta_m^T \end{pmatrix} (\underline{p}, \underline{P}) \tag{58}$$

#### 3.1.1.4 Expressions de la matrice d'interaction

Sachant que notre application finale est l'expérimentation dans des scènes d'intérieur, nous nous intéressons principalement au cas des primitives de type point ou droite. Les calculs de la matrice d'interaction pour d'autres types de primitives (cercle, sphère, cylindre...) peuvent être trouvés dans [Chaumette 90].

**1. Primitive de type point**

Pour un point $m_i$ de coordonnées $\underline{x}_i = (x_i\, y_i\, z_i)^T$ et sa projection $\underline{X}_i = (X_i\, Y_i)$ on obtient facilement :

$$h(\underline{x}, \underline{p}) = \left\{ \begin{array}{l} h_1 = x - x_i = 0 \\ h_2 = y - y_i = 0 \\ h_3 = z - z_i = 0 \end{array} \right. \tag{59}$$

Pour ce cas simple, $\mu$ est donnée à partir de $h_3$ : $z = z_i$ et $g$ par :

$$g(\underline{X}, \underline{P}) = \left\{ \begin{array}{l} g_1 = X - X_i = 0 \\ g_2 = Y - Y_i = 0 \end{array} \right. \tag{60}$$

La représentation matricielle des torseurs d'interaction exprimés au point $C$ dans $R_C$ est :

$$\begin{array}{lcl} L_{X_i}^T & = & [\ -1/z \quad 0 \quad X/z \quad XY \quad -(1+X^2) \quad Y\ ] \\ L_{Y_i}^T & = & [\ 0 \quad -1/z \quad Y/z \quad 1+Y^2 \quad -XY \quad -X\ ] \end{array} \tag{61}$$

**2. Primitive de type droite**

Parmi les nombreuses manières de représenter une droite dans l'espace 3D (cf. [Ayache 88]), nous choisissons celle où la droite est considérée comme l'intersection de deux plans :

$$h(\underline{x}, \underline{p}) = \begin{cases} a_1 x + b_1 y + c_1 z + d_1 = 0 \\ a_2 x + b_2 y + c_2 z + d_2 = 0 \end{cases} \tag{62}$$

Hormis les configurations singulières, une droite 3D se projette sous la forme d'une droite 2D dans l'image tel que :

$$AX + BY + C = 0 \text{ avec } \begin{cases} A = a_1 d_2 - a_2 d_1 \\ B = b_1 d_2 - b_2 d_1 \\ C = c_1 d_2 - c_2 d_1 \end{cases} \tag{63}$$

On choisit une paramétrisation 2D minimale $\underline{P} = (\rho, \theta)$. Nous conseillons de se référer à [Chaumette 90] où il est indiqué comment lever les ambiguïtés de cette représentation. L'équation de la droite devient :

$$g(\underline{X}, \underline{P}) = X \cos\theta + Y \sin\theta - \rho = 0 \tag{64}$$

avec pour changement de paramétrage :

$$\theta = \arctan(B/A)\ ,\ \rho = -C/\sqrt{A^2 + B^2} \tag{65}$$

En utilisant ce paramétrage, l'équation (51) s'écrit :

$$\dot{\rho} + (X \sin\theta - Y \cos\theta)\dot{\theta} = \cos\theta \dot{X} + \sin\theta \dot{Y} \tag{66}$$

La fonction $\mu$ peut s'obtenir à partir de $h_1$ ou $h_2$ :

$$1/z = -(a_i X + b_i Y + c_i)/d_i \text{ avec } i = 1 \text{ où } 2, \text{ et } d_i \neq 0 \tag{67}$$

En utilisant (64), (66), (67), on obtient finalement la représentation matricielle du torseur d'interaction :

$$\begin{array}{lcl} L_\theta^T & = & [\ \lambda_\theta \cos\theta \quad \lambda_\theta \sin\theta \quad -\lambda_\theta \rho \quad -\rho\cos\theta \quad -\rho\sin\theta \quad -1\ ] \\ L_\rho^T & = & [\ \lambda_\rho \cos\theta \quad \lambda_\rho \sin\theta \quad -\lambda_\rho \rho \quad (1+\rho^2)\sin\theta \quad -(1+\rho^2)\cos\theta \quad 0\ ] \end{array} \tag{68}$$

$$\begin{array}{lll} \text{avec} & \lambda_\theta = & (a_i sin\theta - b_i cos\theta)/d_i \\ \text{et} & \lambda_\rho = & (a_i \rho cos\theta + b_i \rho sin\theta + c_i)/d_i \end{array}$$

Le calcul pour d'autres primitives planes (cercles, ellipses..) ou 3D (sphères, cylindres) sont détaillés dans ([Chaumette 90]).

### 3.1.2 La notion de liaison virtuelle

Lorsque la matrice $L^T$ est de rang inférieur à six, il existe des cas où le signal visuel choisi dans l'image est invariant ($\underline{\dot{s}} = 0$) vis à vis d'un ensemble de mouvement de la caméra défini par le torseur cinématique $T^*_{CT}$ tel que :

$$\underline{\dot{s}} = L^T . T^*_{CT} = 0 \tag{69}$$

Ce qui implique directement que :

$$T^*_{CT} = \text{Ker}(L^T) \tag{70}$$

où $Ker(L^T)$ est le noyau de la matrice d'interaction. Le sous espace engendré par $T^*_{CT}$ est donc le sous espace des vitesses caméra laissant invariant le motif image.

Par analogie avec le formalisme des liaisons mécaniques, on dira que l'ensemble des contraintes de la forme $s(\underline{r}) - s^* = 0$ (équivalent à $\underline{\dot{s}} = 0$) définit une **liaison virtuelle** entre le repère associé $(R_T)$ à l'environnement et le repère caméra $(R_C)$.

La dimension $N$ de $T^*_{CT}$ sera appelée **Classe de la liaison virtuelle**. Elle est telle que $N = 6 - \text{rang}(L^T)$.

Le rapprochement avec le formalisme des liaisons mécaniques va nous permettre de spécifier des tâches référencées capteurs en termes de liaisons "mécaniques" entre le robot et son environnement.

Dans les liaisons classiques en mécanique, quelques types de liaisons nous paraissent intéressantes pour un robot mobile interagissant avec son environnement :

- **liaison rigide**

  C'est une liaison de *classe 0* puisqu'aucun degré de liberté n'est laissé libre. Cette liaison correspond à une tâche de positionnement telle qu'il existe une seule situation entre la caméra et son environnement satisfaisant la liaison. Ceci implique qu'aucun mouvement de la caméra n'est possible à l'équilibre : $\underline{s}(\bar{r}) = \underline{s}^*$. Le noyau de $L^T$ est nul :

$$T^*_{CT} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Il est donc nécessaire de disposer d'au moins six informations visuelles $s_j$ telles que la matrice d'interaction soit de rang plein 6. Ainsi par exemple trois points choisis correctement, de façon à éviter des singularités ([Michel 93]), suffisent à fournir les six informations visuelles.

- **liaison prismatique**

  Une liaison prismatique laisse un degré de liberté en translation. C'est donc une liaison de classe 1. Le sous-espace $T^*_{CT}$ correspondant doit avoir la forme :

  $$T^*_{CT} = \begin{pmatrix} a & b & c & 0 & 0 & 0 \end{pmatrix}$$

  où la direction du vecteur de composantes $(a, b, c)$ est la direction de la translation de la caméra suivant laquelle le motif dans l'image est constant.

  Cette liaison nécessite cinq informations visuelles indépendantes, que l'on peut obtenir de diverses façons, par exemple en utilisant trois droites parallèles entre elles (cf. [Chaumette 90]).

- **liaison rotoïde**

  Cette liaison laisse un degré de liberté en rotation. C'est donc une liaison de classe 1. Pour une liaison rotoïde autour de l'axe $\vec{z}$ de la caméra, par exemple, on peut choisir comme motif deux cercles parallèles au plan image, coplanaires, centrés et de rayons différents. Pour cet exemple le sous-espace a comme expression :

  $$T^*_{CT} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

- **liaison "trajectoire circulaire" de classe 1**

  Pour notre application, on peut avoir à utiliser une liaison virtuelle de classe 1 sans analogie avec une liaison mécanique. Une trajectoire circulaire portée par un plan orthogonal au plan image caméra peut être spécifié, par exemple, par le torseur :

  $$T^*_{CT} = \begin{pmatrix} 0 & a & 0 & b & 0 & 0 \end{pmatrix}$$

  On verra que dans notre cas, un segment vertical de la scène peut-être choisi pour constituer le motif qui satisfasse cette liaison virtuelle.

• **liaison appui-plan**

Cette liaison laisse trois degrés de liberté : deux en translation et un en rotation. Sa classe est donc de trois. Par exemple, dans le cas d'un appui-plan constitué par le plan image, il faut trouver un motif tel que l'espace des torseurs invariants associés soit :

$$T^*_{CT} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

La motif nécessaire pour réaliser cette liaison peut être constitué d'un maillage régulier parallèle au plan image et constitué de segments de même longueur. En prenant les longueurs d'au moins trois segments non alignés, ou les quatre d'un maillage carré, on dispose des trois informations nécessaires à la réalisation de l'appui-plan. Cet exemple se retrouve encore dans [Chaumette 90].

La notion de liaison virtuelle peut donc être utilisée pour spécifier des tâches robotiques pouvant s'exprimer par un ensemble de contraintes géométriques entre le repère caméra et le repère de l'environnement local.

### 3.1.3 La commande référencée vision

La méthodologie pour modéliser l'interaction entre une caméra et son environnement, étant sommairement décrite, il est nécessaire d'intégrer cette modélisation au sein de la commande d'un robot.

Cette intégration, utilise un formalisme très général de commande robotique : l'approche fonction de tâches. Nous rappellerons très brièvement les principaux résultats de cette approche et demandons au lecteur de consulter [Samson 90a] pour de plus amples renseignements. On rappelle ensuite l'application de ce type de commande robuste pour un capteur visuel, issue des travaux présentés dans ([Chaumette 90], [Espiau 92], [Rives 89]).

#### 3.1.3.1 La fonction de tâche

Le comportement dynamique d'un robot rigide est décrit par l'équation générale suivante :

$$\Gamma = M(q)\ddot{q} + N(q, \dot{q}, t), \dim(q) = \dim(M) = n \quad (71)$$

où :

- $\Gamma$ est le vecteur des forces extérieures appliquées,
- $M$ est la matrice d'énergie cinétique
- $N$ rassemble les contributions des forces de gravité, centrifuges, de Coriolis, et de frottement.

La tâche robotique à effectuer peut-être spécifiée par l'utilisateur comme une fonction de sortie associée à l'équation d'état (71). Dans ([Samson 90a]) il est montré que satisfaire une tâche peut être exprimé comme une régulation à zéro, sur un horizon $[0, T]$, d'une fonction $C^2$ de dimension $n$ : $e(q, t)$, appelée fonction de tâche.

La fonction de tâche est admissible (le problème de régulation est bien posé) si $e(q, t)$ possède certaines propriétés :

- il existe une trajectoire idéale unique $q_r(t)$ telle que $e\left(q_r(t), t\right) = 0$, $\forall t \in [0, T]$ et $q_r(0) = q_0$.
- le jacobien de la tâche $\frac{\partial e}{\partial q}(q, t)$ est régulier autour de la trajectoire idéale.

Les exemples classiques de fonctions de tâches pour les bras manipulateurs sont :

- la commande en position articulaire sur une trajectoire articulaire désirée $q_d$. On choisit alors : $e\left(q, t\right) = q - q_d(t)$,
- la commande en position cartésienne sur une trajectoire de position $x_d$. La fonction de tâche est de la forme : $e(q, t) = x(q) - x_d(t)$.

A partir de (71) et de la fonction de tâche, il est possible d'obtenir un schéma général de commande robuste qui englobe de nombreux schémas existants (commande dynamique, commande en vitesse, commande adaptative...). Pour la description complète de la commande, il faut bien sûr consulter ([Samson 90a]).

#### 3.1.3.2 Redondance et tâches hybrides

On s'intéresse au cas où une tâche robotique $e$ de dimension $m$ peut être spécifiée comme la réalisation simultanée de deux tâches $e_1$ et $e_2$.

Pour pouvoir construire une fonction de tâche $e$ admissible, il est indispensable que les fonctions $e_1$ et $e_2$ soient compatibles et indépendantes. Intuitivement, on comprend qu'il faut que le deuxième objectif fixé par $e_2$ puisse être atteint grâce à l'ensemble des mouvements réalisables laissés libres par la liaison virtuelle associée à $e_1$.

Pour répondre aux besoins de combinaisons de telles tâches, le formalisme des "tâches redondantes" permet de considérer une tâche $e_1$, de dimension $m < n$ prioritaire et une deuxième tâche $e_2$ dont on minimise un coût secondaire, sous contrainte de la régulation parfaite de $e_1$.

La fonction de tâche globale prend alors la forme :

$$e = W^+ e_1 + \alpha(I_n - W^+ W) g_s^T \tag{72}$$

où :

- $\alpha$ est un scalaire strictement positif,
- $g_s^T = \frac{\partial h_s}{\partial r}$ est le gradient d'une fonction scalaire $h_s$ à minimiser sous la contrainte $e_1 = 0$.
- $(I_n - W^+ W)$ est un opérateur de projection

Pour conserver la régularité de la tâche, il est nécessaire que :

- $J_1 = \frac{\partial e_1}{\partial \underline{r}}$ soit de plein rang le long de la trajectoire idéale,
- $Ker(W^T) = Ker(J_1^T)$

De plus pour s'assurer de la stabilité de la commande (cf [Chaumette 90]), il est nécessaire de satisfaire la propriété de positivité :

$$J_1 W^T > 0 \tag{73}$$

Dans le cas où la tâche principale $e_1$ est la régulation de signaux capteurs et que le coût secondaire à minimiser s'exprime comme un suivi de trajectoire dans $SE3$, la tâche $e$ résultante est appelée **tâche hybride**.

#### 3.1.3.3 Application au cas de l'asservissement visuel

L'asservissement visuel s'intéresse à la régulation du vecteur des informations visuelles s(q,t) autour d'une trajectoire désirée $s^*(t)$ exprimé dans l'espace image.

Soit $D$ une matrice de combinaison telle que la matrice $DL^T$ soit de rang plein le long de la trajectoire idéale. La tâche principale est de la forme :

$$e_1 = D(t)\ (s(\bar{r}, t) - s^*(t)) \tag{74}$$

Le jacobien de la tâche principale s'écrit :

$$J_1^T = \frac{\partial e_1}{\partial \bar{r}} = DL^T \tag{75}$$

En utilisant le formalisme de la redondance, la tâche globale à réguler s'écrit :

$$e = W^+D\,(s(\bar{r},t) - s^*(t)) + \alpha \left(I_6 - W^+W\right) g_s^T \tag{76}$$

W est formé des vecteurs de base de $Ker(L^T)$ de manière à avoir :

$$Ker(W^T) = Ker(L)$$

La condition de régularité s'exprime alors $DL^TW^T > 0$.

Il existe plusieurs choix de la matrice $D$ permettant de satisfaire cette condition de régularité ([Chaumette 90]). Une des solutions les plus simples et qui a donné de bons résultats expérimentaux, est de choisir :

$$D = W\hat{L}^{T+} \text{ avec } \hat{L}^{T+} = L^T(\underline{p}^*, \underline{P}^*) = L^T_{|s=s^*} \tag{77}$$

Pour ensuite réguler cette fonction de tâche, on impose une décroissance exponentielle de l'erreur :

$$\dot{e} = -\lambda e \tag{78}$$

En dérivant la fonction de tâche, on obtient :

$$\dot{e} = \frac{\partial e}{\partial \underline{r}}\,T_{CT} + \frac{\partial e}{\partial t} \tag{79}$$

En supposant que l'on souhaite réaliser une commande en vitesse $T_{CT}$ du déplacement de la caméra, on peut calculer un torseur cinématique désiré à partir de (78) et (79). Dans la pratique, on estime la jacobienne de la tâche et sa vitesse. On obtient alors :

$$T_{CT} = \left(\widehat{\frac{\partial e}{\partial \underline{r}}}\right)^{-1} (-\lambda e - \widehat{\frac{\partial e}{\partial t}}) \tag{80}$$

Ayant choisi $W$ constant, l'erreur de poursuite de la tâche issue de la dérivation de (76), est de la forme :

$$\frac{\partial e}{\partial t} = W^+\frac{\partial e_1}{\partial t} + \alpha \left(I_6 - W^+W\right) \frac{\partial g_s^T}{\partial t} \tag{81}$$

où $\widehat{\frac{\partial e_1}{\partial t}}$ représente la contribution d'un éventuel mouvement de la cible. Il est montré dans [Chaumette 90] que :

- sous réserve que $\alpha$ soit petit (correspondant au "poids" de la tâche secondaire vis à vis de la tâche principale),

- l'on se situe autour de la position d'équilibre où la liaison est réalisée ($s = s^*$),

alors le choix $\frac{\widehat{\partial e}}{\partial \underline{r}} = I_6$ est valide. En utilisant (80) et (81) l'expression du torseur cinématique sera alors :

$$T_{CT} = -\lambda e - W^+ \frac{\widehat{\partial e_1}}{\partial t} - \alpha(I_n - W^+ W) \frac{\partial g_s^T}{\partial t} \tag{82}$$

où :

- $\lambda$ fixe le taux de convergence de la commande,

- $\alpha$ permet de choisir la prépondérance de la tâche principale sur la tâche secondaire ($\alpha < 1$),

- $\frac{\widehat{\partial e_1}}{\partial t}$ estime un éventuel mouvement autonome des cibles dans la scène. Pour des cibles statiques, on prendra évidemment :$\frac{\widehat{\partial e_1}}{\partial t} = 0$.

Le torseur de vitesse, exprimé dans le repère caméra, sera transformé en vitesse de consigne articulaire via le jacobien inverse de l'ensemble base mobile + bras calculé précédemment, soit :

$$\underline{\dot{q}} = J^{-1} T_{CT} \tag{83}$$

## 3.2 Les tâches référencées vision pour un robot mobile

Jusqu'à présent, nous avons montré comment il était possible de commander un robot pour assurer un asservissement des mouvements d'une caméra embarquée à partir de signaux visuels. L'étape suivante de notre travail consiste à spécifier et à illustrer un ensemble de tâches référencées vision dédiées à un robot mobile. Tout d'abord, nous délimitons notre espace d'étude en fixant quelques hypothèses de travail.

### 3.2.1 Les hypothèses de travail

La définition d'une tâche robotique référencée vision doit tenir compte du type de robot utilisé, en particulier des degrés de liberté que le robot confère à la caméra. De plus, l'environnement dans lequel est placé le robot influe évidemment sur le choix des motifs visuels.

Dans notre cas, nous considérons *la famille des robots mobiles à roues dotés d'un manipulateur portant une caméra.* Au cours du chapitre 2, la modélisation cinématique de ce type de robot a été étudiée et illustrée sur un exemple d'un robot mobile de type char muni d'un bras à trois degrés de liberté. Nous avons choisi cette même structure mécanique pour les simulations de tâches référencées vision que nous présenterons ultérieurement.

La scène, dans laquelle se déplace le robot, *est partiellement inconnue.* En effet l'environnement est inconnu mais il est nécessaire de faire des hypothèses sur la dimension 3D des motifs visuels sur lesquels l'asservissement est effectué. Le robot évoluant dans *un environnement structuré d'intérieur de "type bureau"*, on peut énoncer de nombreuses hypothèses $3D$ car les dimensions de la scène sont souvent normalisées (hauteur des murs, largeur de portes...).

Ce type d'environnement structuré, dans lequel se déplace le robot, fixe naturellement le choix des signaux capteurs. Les contours extraits de la projection image de la scène 3D sont, pour une grande partie, constitués par des segments (exemples : arêtes de murs, ouverture d'une porte, meubles...). Les signaux visuels utilisés seront donc constitués de primitives géométriques simples tels que les droites, les segments et les points.

Le motif visuel de consigne caractérise le type de tâche robotique qui sera effectué. A partir du motif choisi, la construction de la tâche robotique peut se décomposer en plusieurs étapes qui sont :

- le calcul de la tâche principale d'asservissement visuel caractérisée par sa matrice d'interaction.
- le calcul de la liaison virtuelle associée, c'est à dire la détermination des degrés de liberté du mouvement caméra qui sont laissés libres par l'asservissement visuel.
- le calcul d'une tâche secondaire, en général un suivi de trajectoire, permettant de contrôler l'espace de commande laissé libre. Si la liaison virtuelle est de classe 0 (liaison rigide) cette étape est bien sûr inutile.
- la construction de la commande (tâche hybride) intégrant la tâche principale et la tâche secondaire

### 3.2.2 Les différentes tâches référencées vision

Sous les hypothèses énoncées ci-dessus et ayant définit le type de robot et d'environnement considéré, nous allons énumérer les tâches référencées vision qui vont permettre au robot de se déplacer dans son environnement local.

**Tâche de Positionnement**
Une tâche de positionnement définit une liaison rigide (classe 0) entre la caméra et le motif visuel ; c'est à dire qu'il n'existe qu'une seule situation entre la caméra et son environnement satisfaisant la liaison. Dans notre type d'environnement, par exemple, une tâche de positionnement doit permettre de positionner le robot, ou plutôt sa caméra embarquée, en face d'une ouverture de porte ou de fenêtre, devant un angle de mur... Notons au passage que le franchissement d'une porte par exemple devra combiner une tâche d'asservissement visuel et le contrôle de l'orientation du mobile ; c'est encore un problème ouvert. Mais le franchissement de porte peut aussi être réalisé en combinant des tâches élémentaires (positionnement de la caméra en face de la porte puis avancée vers la porte pour que le robot se positionne...)

Les hypothèses 3D sur le motif, pour calculer sa matrice d'interaction à la position d'équilibre, sont faciles à obtenir pour nos exemples : une ouverture de porte ou de fenêtre est de dimension standardisée, un coin de mur constitué par l'intersection de trois segments est à une hauteur connue par rapport au sol.

**Tâche de Suivi de cible**
Nous faisons une différence entre une tâche de positionnement où le motif est statique dans la scène et une tâche de suivi de cible où le motif est mobile. Pour un même objet 3D, qu'il soit fixe ou mobile, la matrice d'interaction et le motif visuel peuvent être identiques. Par contre la construction de la commande est différente car, dans le cas du suivi de cible, il est nécessaire d'estimer la vitesse de la cible afin d'annuler l'erreur de traînage dûe au mouvement de la cible. Tout objet dont on connaît les dimensions et dont on peut extraire suffisamment d'informations visuelles peut faire l'objet d'un suivi au niveau local par la caméra embarquée sur le robot.

**Tâche de Suivi de lignes**
Dans une scène de "type bureau", on peut extraire de nombreuses lignes horizontales utilisables pour constituer un motif image dans le but d'assurer une liaison prismatique entre la caméra et son environnement local. Les lignes horizontales de la scène, constituées par les arêtes d'un mur, le carrelage au sol, les contours

horizontaux d'objets... se projettent dans l'image sous forme de droites. Ces informations visuelles peuvent être exploitées dans une tâche d'asservissement visuel laissant non contraint un degré de liberté en translation. Ce degré de liberté sera utilisé dans une tâche hybride afin de réaliser un suivi de ligne.

**Tâche de Rotation autour d'un segment**
Cette famille de tâches laisse, comme la précédente, un degré de liberté non contrôlé par la vision. On utilise un segment vertical de la scène tel un angle de mur ou un montant de porte. On fait encore ici l'hypothèse que l'on connaît des informations 3D a priori sur la scène. Le motif ainsi constitué, doit permettre de contrôler les mouvements de la caméra pour exécuter une trajectoire circulaire autour de l'axe vertical définit par l'orientation du segment.

En utilisant deux segments verticaux différents, on devrait être capable de composer un motif afin de définir une trajectoire ellipsoïde autour des deux axes. L'intérêt de ce type de trajectoire est beaucoup moins évident.

**Tâche de déplacement vers un coin**
Dans une scène structurée, on trouve de nombreux "coins" définis par l'intersection de deux droites. On peut utiliser ce motif visuel pour spécifier une tâche de déplacement vers un angle d'objet (meuble, porte, murs). Cette famille de tâches nécessite d'estimer en permanence la matrice d'interaction car la distance entre le centre optique et le motif varie le long de la trajectoire. Les calculs et les simulations (cf. annexe B.2) montrent que pour ce type de tâche la liaison virtuelle est de classe 1.

**Tâche de Contrôle d'effecteur**
Les tâches référencées vision peuvent être utilisées pour contrôler l'effecteur sur lequel est placé la caméra. On peut, par exemple, contrôler que l'effectuer reste horizontal en utilisant un motif issu d'un carrelage au sol qui définirait une liaison virtuelle d'appui plan de classe 2. Le suivi de conduite (tuyaux) pourrait être aussi assuré par un bras manipulateur placé sur robot mobile. L'application finale à laquelle on s'intéresse étant l'exploration d'une scène structurée, nous n'avons pas approfondi ce type d'utilisation.

**Tableau 1 : Tâches référencées vision**

| Type de la sous-tâche | Liaison | Motif | Hypothèses 3D |
|---|---|---|---|
| Se positionner :<br>devant un "objet"<br>(porte, meuble, angle)<br>Suivre une cible | Rigide<br>Classe 0 | 2 segments ou<br>3 droites non parallèles | Dim objet |
| Suivre une ligne :<br>pan de mur,<br>plinthe,<br>carrelage au sol,<br>un objet. | Prismatique<br>Classe 1 | 2 droites parallèles<br>horizontales<br>dans la scène | Distance<br>entre les<br>2 droites |
| Aller vers<br>un coin | Prismatique<br>$L^T$ varie | 2 droites non parallèles<br>(= coin) | Info 3D sur<br>le coin |
| Tourner autour<br>d'un segment | Circulaire<br>Classe 1 | 1 segment vertical | Longueur 3D |
| Tourner autour<br>de 2 segments | Ellipsoïde<br>Classe 1 | Composition<br>de 2 segments verticaux | Longueurs 3D |

### 3.2.3 Exemples de simulation

La plupart des tâches référencées vision énumérées dans la partie précédente ont été simulées dans le but d'obtenir une première validation de l'approche.

Le modèle du robot simulé est celui qui a été déjà utilisé pour les simulations présentées au chapitre 2 . Pour ces premières simulations, nous ne prenons pas en compte les phases transitoires de l'asservissement visuel; c'est à dire que le motif image courant à l'instant initial est le motif de consigne de l'asservissement. Nous simulons ensuite le déplacement le long de la trajectoire définie par la liaison virtuelle associée au motif utilisé.

Les enchaînements des tâches illustrées en simulation sont réalisées par l'utilisateur. Il se charge d'arrêter la tâche visuelle quand le motif disparaît de l'image, de choisir un autre motif, de relancer la tâche suivante. Nous avons développé un outil de simulation qui permet la spécification et l'illustration des tâches référencées vision. Nous présenterons brièvement cet outil dans la partie d'étude expérimentale.

L'exemple de simulation présenté ici est l'utilisation de tâches de "suivi de lignes" et de "rotation autour d'un segment" pour faire effectuer au robot un demi-tour dans un couloir. D'autres exemples de simulation sont donnés en annexe B.2 :

- déplacement du robot dans un couloir,
- déplacement du robot en utilisant le carrelage,
- déplacement du robot en se dirigeant vers des “angles”.

L'exemple de simulation du “demi-tour” est illustré par la figure (3.2). Le robot utilise le motif constitué par les deux droites de la plinthe du mur pour réaliser une première tâche de suivi (liaison virtuelle en translation). La Vue 1 et 2 correspondent respectivement à la vue caméra initiale et finale du motif lors de l'exécution de cette tâche. Le motif étant sur le point de disparaître, l'utilisateur interrompt cette première tâche pour effectuer une recherche heuristique d'un autre motif en faisant pivoter la caméra. Il choisit le segment vertical de la Vue 3 pour faire tourner la caméra autour de celui-ci. En fonction du choix du motif visuel sur lequel le robot doit s'asservir, le logiciel calcule la tâche qui est associée au motif. Pour la tâche de rotation, les calculs sont donnés en annexe B.1. La rotation autour du segment est effectuée jusqu'à la Vue 5. La dernière tâche enchaînée est le suivi du mur opposé en utilisant ses deux arêtes horizontales.

## 3.3 Les enchaînements de tâches référencées vision

Le formalisme des liaisons présenté précédemment permet d'exécuter des tâches robotiques élémentaires par asservissement visuel. Toutefois, la réalisation de tâches plus complexes nécessitent de savoir enchaîner plusieurs tâches élémentaires. Se pose donc le problème de la gestion des phases transitoires [1]. En fonction des circonstances, ce problème peut prendre plusieurs aspects :

(1) Au lancement d'une tâche d'asservissement visuel, on désire contrôler la phase de convergence vers un motif de consigne. Cela résulte du fait que la configuration initiale du motif visuel se trouve très éloigné de la configuration désirée. Il s'agit donc de gérer le passage de $e(q_0, 0) = e_0 \neq 0$ à $e(q_0, \tau) = 0$

(2) Au cours d'une tâche d'asservissement, il peut s'avérer utile de changer de motif visuel sans pour autant interrompre la tâche. Ce changement

[1] Nous remercions Bernard Espiau pour son aide lors de l'établissement du cadre formel sur l'enchaînement des tâches à l'intention des utilisateurs

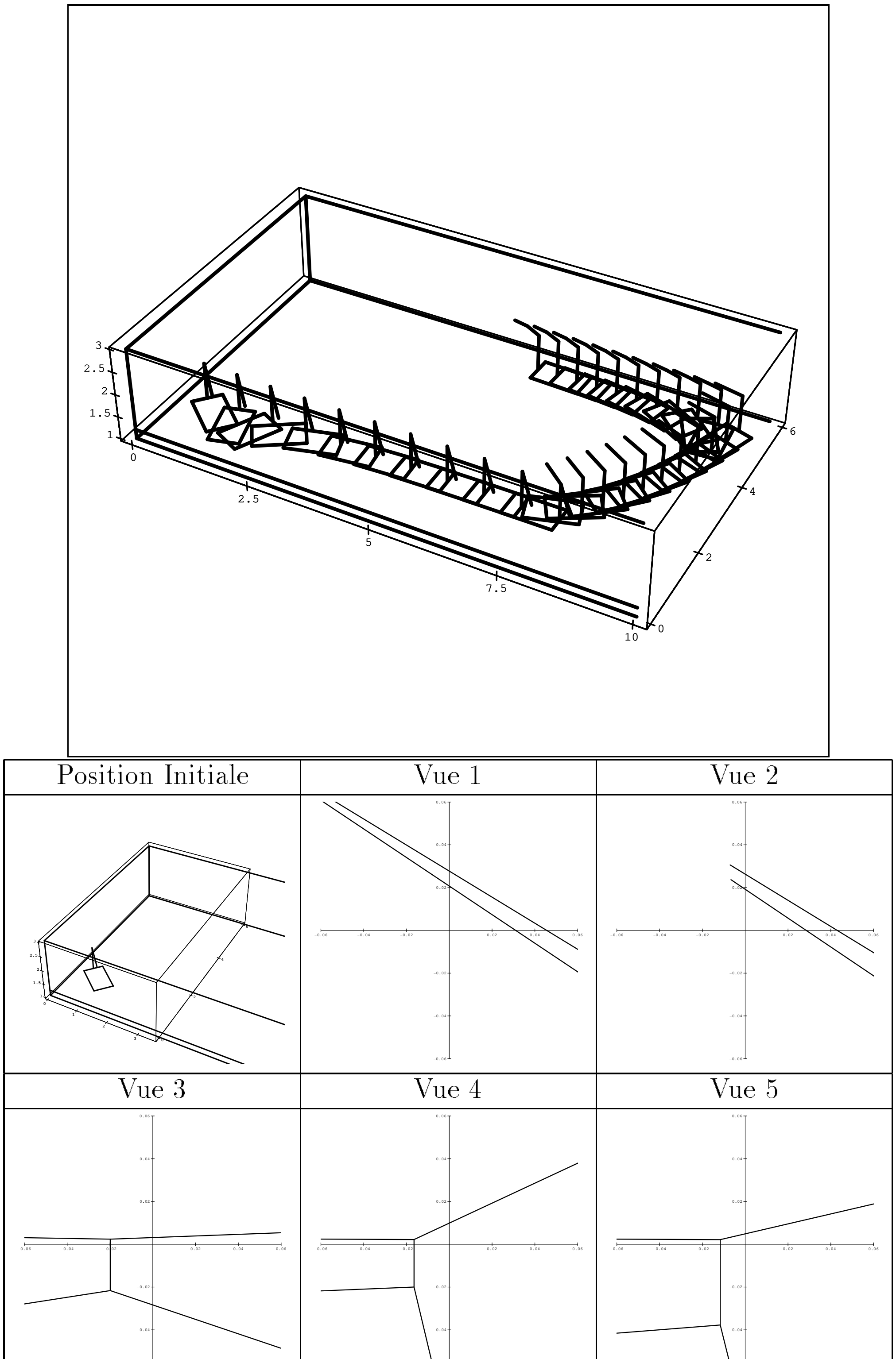



Figure 3.2 : Simulation du demi-tour

peut intervenir par exemple si les bruits de mesures sur le motif courant sont trop importants. Le transitoire à gérer est le passage de $e_1(q_1, t)$ à $e_2(q_2, t + \tau)$ avec $e_1$ et $e_2$ réalisant la même liaison virtuelle.

(3) Enfin, il sera nécessaire de contrôler la transition correspondante à l'enchaînement de deux tâches $e_1$ et $e_2$ successives réalisant des liaisons virtuelles différentes.

Nous décrirons et illustrerons ce type de gestion des transitions pour les cas les plus simples. Nous évoquerons simplement les enchaînements plus délicats des tâches de nature différente.

### 3.3.1 Gestion des transitoires

Le cas le plus simple à considérer est la régulation à zéro d'une fonction de tâche dont les conditions initiales sont non nulles pour une position $q_0$ telles que $e(q_0, 0) = e_0 \neq 0$. Une façon de procéder est de créer une nouvelle tâche $e'$ se confondant avec $e$ au bout d'un certain temps et telle que $e'(q_0, 0) = 0$.

On choisit une fonction $\lambda(t)$ scalaire $C^2$ décroissante sur $[0, \tau]$ telle que $\lambda(0) = 1$ et $\lambda(\tau) = 0$. On peut choisir aussi $\lambda$ tel que $\lim_{t\to\infty} \lambda(t) = 0$. La nouvelle tâche $e'$ peut alors s'écrire :

$$e'(q, t) = \lambda(t)(q - q_0) + (1 - \lambda(t))e(q, t) \tag{84}$$

Cette expression crée un mélange entre les deux tâches. Dans le cas de signaux images, ce principe de mélange peut s'appliquer à tout changement de consignes visuelles. Il suffit de faire évoluer la consigne telle que :

$$s^r(t) = \begin{cases} s^* & t \leq t_0 - \frac{\tau}{2} \\ \lambda(t - t_0 + \frac{\tau}{2})s^* + (1 - \lambda(t - t_0 + \frac{\tau}{2}))s'^* & t \in [t_0 - \frac{\tau}{2}, t_0 + \frac{\tau}{2}[ \\ s'^* & t \geq t_0 + \frac{\tau}{2} \end{cases} \tag{85}$$

Le changement de consigne modifie la trajectoire idéale, il faut théoriquement mettre à jour l'estimée de la matrice d'interaction $\hat{L} = L_{|s=s^r(t)}$ qui était auparavant constante.

Nous illustrons l'intérêt de la gestion du transitoire d'une tâche référencée vision par un exemple de simulation.

La tâche robot à effectuer est un suivi de mur à une distance de un mètre. Au départ, le robot mobile est placé à trois mètres du mur. Si on lance l'asservissement directement sur la consigne visuelle à atteindre, le robot ne peut assurer le mouvement de la caméra. Dans le cas illustré sur la première ligne de la figure (3.3) les vitesses à atteindre sont trop élevées (dans le cas d'une commande en saturation, les limites mécaniques sont atteintes).

En revanche, en gérant la transition de la consigne visuelle de sa valeur initiale à sa valeur finale désirée, la trajectoire du robot devient satisfaisante. Cette trajectoire est présentée sur la deuxième ligne de la figure (3.3).

### 3.3.2 Gestion des transitions entre tâches

Le problème que l'on se pose est le passage, autour d'un certain temps $t_0$, d'une tâche $e_1$ référencée vision à une autre tâche $e_2$ de façon "douce". Ce passage est plus ou moins complexe suivant les situations que l'on considère.

Les notations utilisées, pour la tâche initiale ($i = 1$) et la tâche finale ($i = 2$), sont les suivantes :

- $s_i$ les signaux capteurs avec $p_i = dim(s_i)$
- $S(L_i)$ le sous espace engendré par $L_i$ avec $m_i = rang(L_i)$

A. $\boxed{S(L_1) = S(L_2)\,, m_1 = m_2 = m}$

On désire effectuer un changement de paramétrage ou de combinaison de signaux pour deux tâches de même liaison virtuelle. Dans le cas où les signaux visuels ont la dimension du sous espace engendré , on peut utiliser une pondération des deux vecteurs de signaux pour en construire un global. Si la dimension est différente, le mélange est un peu moins simple.

1. $p_1 = p_2 = m$

Dans ce cas, la dimension de chaque vecteur de signaux visuels est égale à celle du sous espace engendré. Il n'y a donc pas de signaux redondants, par exemple. Pour obtenir une fonction de tâche régulière, on choisit une fonction de transition de signaux telle que :

$$\begin{array}{l} \tilde{s} = \lambda(u)(s_1 - s_1^r(t)) + (1 - \lambda(u))(s_2 - s_2^r(t)) \\ \text{avec } u = t - t_0 + \frac{\tau}{2} \end{array} \tag{86}$$

$\lambda$ est la fonction scalaire définie précédemment. Le changement de paramétrage en $u$ a simplement pour but de rendre la transition symétrique, autour de $t_0$, en supposant que $\lambda(t - \frac{\tau}{2})$ est impaire.

On calcule la nouvelle matrice d'interaction notée $\tilde{L}^T$: :

$$\tilde{L}^T = \lambda(u)L_1 + (1 - \lambda(u))L_2 \tag{87}$$

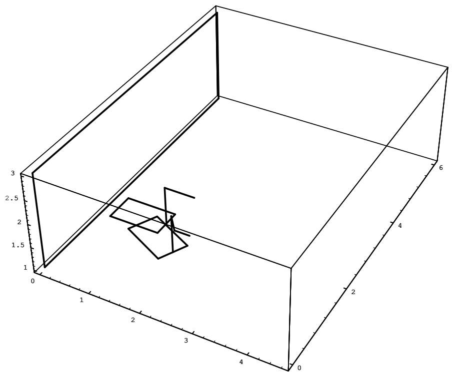
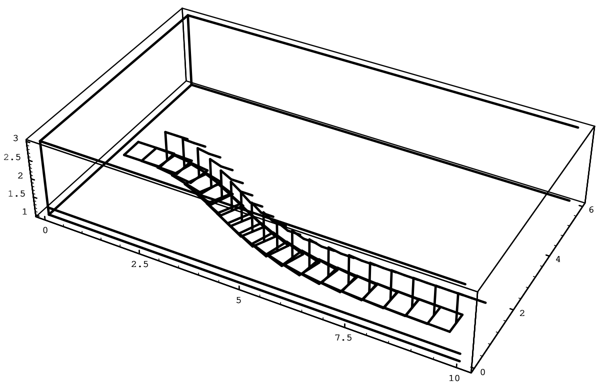
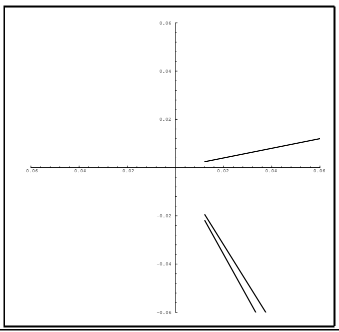
| Simulation interrompue | Diagnostic |
|---|---|
|  | Je me suis arrete a l'échantillon 5(9)<br><br>Limites cinématiques atteintes ! |
| Simulation réussie | Consigne visuelle |
|  |  |

Figure 3.3 : Simulations du suivi du mur

Or, puisque $S(L_1) = S(L_2)$, on peut montrer (cf annexe C.1) qu'il existe une matrice $C$ $p_1 \times p_2$ de rang $m$ telle que $L_2 = L_1 C$. Dans notre cas où $p_1 = p_2 = m$, C est régulière de dimension $m \times m$. On peut alors écrire :

$$\tilde{L}^T = L_1(\lambda(u) I_m + (1 - \lambda(u)))C \tag{88}$$

Une condition nécessaire à la régularité de $\lambda(u) I_m + (1 - \lambda(u))$ est que C soit strictement positive.

La fonction de tâche principale peut s'écrire :

$$\tilde{e}_1 = D\tilde{s} \tag{89}$$

avec la matrice de combinaison $D = W\tilde{L}$ où $W$ est une matrice de vecteurs de base de $S(L_1) = S(L_2) - S(\tilde{L})$.

2. $p_1 \neq p_2$

Dans le cas où la dimension des deux vecteurs de signaux visuels est différente, on ne peut plus faire de somme pondéré. Par contre il est toujours possible de mélanger les signaux en utilisant les matrices de combinaisons. Le détail de ce mélange est donné en annexe C.2.

B. $\boxed{S(L_1) \neq S(L_2)}$

1. $m_1 = m_2 = m$

Dans le cas où les deux sous espaces générés par $L_1$ et $L_2$ ont la même dimension, le mélange des signaux capteurs reste possible en choisissant :

$$\tilde{s} = \lambda(u) D_1(s_1 - s_1^r(t)) + (1 - \lambda(u)) D_2(s_2 - s_2^r(t)) \tag{90}$$

où $D_i$ est une matrice $m \times p_i$ de rang $m$, $i = 1, 2$.

2. $m_1 \neq m_2$

Dans le cas le plus général, où les dimensions des 2 sous espaces $S(L_1)$ et $S(L_2)$ sont différents, une possibilité consiste à passer par une phase d'arrêt à la position atteinte au moment du changement de tâches. Les deux tâches visuelles ne se recouvrent pas et on peut définir un passage en " douceur". Le détail de ces calculs sont également donnés en annexe C.3.

## 3.4 Conclusion et Perspectives

En utilisant les formalismes des liaisons virtuelles et des enchaînements de tâches présentés dans ce chapitre, il est possible de définir et d'enchaîner des lois de commande d'asservissement visuel pour un robot mobile. Cependant, assurer une certaine continuité des lois d'asservissement n'est pas suffisant pour réaliser des tâches robotiques complexes ; il est nécessaire d'y adjoindre un comportement logique capable de séquencer le passage d'une tâche d'asservissement à une autre. Ce comportement local doit notamment prendre en compte les événements pouvant survenir dans l'image. La formalisation de cet aspect discret est difficile et nous n'évoquons ici que quelques idées préliminaires sur la manière dont on pourrait l'aborder.

La prise de décision d'arrêter un asservissement visuel pour en changer doit être faite sur des événements que l'on détecte dans l'image. A une tâche référencée vision, on associe des "fenêtres actives" qui permettent soit de donner des informations visuelles pour la commande (fenêtre de commande), soit de générer des événements à prendre en compte (fenêtre d'observation). Ces fenêtres dépendent du motif visuel de l'asservissement et doivent en assurer la poursuite dans l'image.

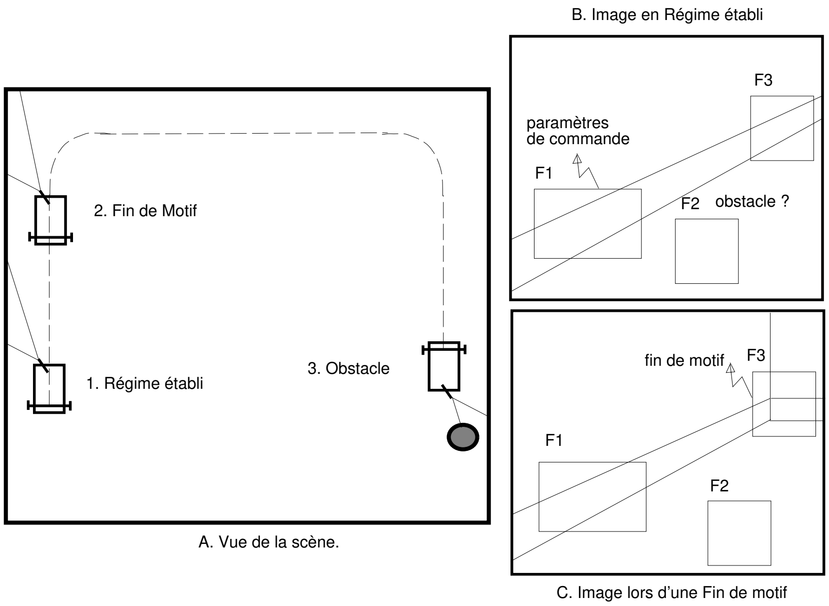



Figure 3.4 : Faire le tour de la pièce

On peut traiter ce type de méthodologie sur l'exemple simple de l'application "Faire le tour d'une pièce jusqu'à trouver un obstacle" (cf. figure (3.4.A)). On utilise une tâche référencée vision qui consiste à "suivre un mur" et une autre pour "passer un coin". On doit associer aux spécifications sur l'aspect continu (matrice d'interaction, motif à suivre, tâche secondaire...) des fenêtres actives (cf. figure (3.4.B)). qui peuvent être par exemple :

- une fenêtre de commande (F1) où l'on extrait les paramètres du motif $s(t)$ pour la commande,
- une fenêtre d'observation (F2) pour observer s'il existe un obstacle sur le parcours du robot (présence de primitives visuelles non prévues),
- une fenêtre d'observation (F3) pour anticiper les événements pouvant produire une modification de la tâche d'asservissement courante. Cette fenêtre est donc chargée de détecter une fin de motif. L'événement "Fin de Motif" qu'elle génère servira, dans un premier temps, à arrêter l'asservissement. Nous espérons pouvoir modéliser les connaissances sur la tâche et l'environnement permettant d'enchaîner une autre tâche sans avoir forcément à arrêter le robot.

On peut simplement définir l'automate (cf. figure (3.5)) qui illustre les différents états pour l'application spécifiée. La tâche référencée vision de suivi de mur assure le suivi en "régime établi" tandis que la tâche "passe un coin" assure un passage lisse entre deux suivis (au niveau loi de commande et fenêtres). L'enchaînement de tâches au niveau discret est une étape essentielle pour pouvoir définir des tâches robotiques constituées par une succession de tâches élémentaires d'asservissements visuels.

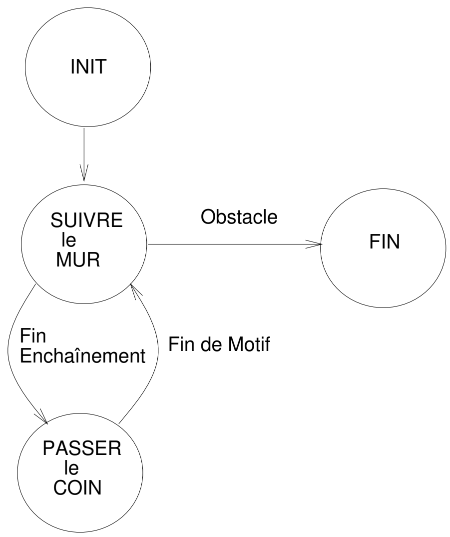



Figure 3.5 : Automate d'une application

Sur ce simple exemple, nous avons illustré qu'il était difficile d'envisager de spécifier une application en la considérant comme un simple enchaînement de lois continus. La prise en compte des événements permettront le séquencement de ces lois. Ces spécifications très simplifiées doivent bien sûr être approfondies.

D'importants problèmes d'implémentation doivent être pris en compte (environnement de programmation, réalisation de système robotique adapté...) pour associer l'aspect continu et discret. Dans la partie expérimentale qui suit, nous proposons une validation qui tienne compte de ces problèmes.

# Deuxième Partie

# Etude Expérimentale

# Chapitre 4

# Outils de conception pour les contrôleurs en robotique

## 4.1 Introduction

La spécification et la simulation des tâches robotiques d'asservissement visuel dans le cadre de la robotique mobile, présentées dans les chapitres 2 et 3, ne sont pas suffisantes pour valider complètement notre approche. Une intégration complète des tâches de commande référencée vision sur un robot mobile réel est indispensable. Cette dernière étape, trop souvent négligée, permet d'obtenir des renseignements très utiles pour affiner voire pour modifier les spécifications initiales de la commande du robot.

Ce passage, du stade des spécifications et de la simulation à celui de l'implémentation, fait appel à un large spectre de connaissance :

- dans les domaines de la mécanique pour définir le robot physique à réaliser,
- dans les domaines de l'automatique pour spécifier les lois de commande,
- dans les domaines de l'informatique temps réel distribuée pour produire un logiciel efficace et performant,
- dans les domaines de l'informatique matérielle c'est à dire : les cartes CPU, les cartes spécialisées, les bus, les réseaux locaux l'interfaçage avec les robots...

La plupart des contrôleurs robotiques commercialisés sont des systèmes fermés qui utilisent leur propre langage de programmation peu adapté à la commande référencée capteurs.

Pour ces raisons, il est difficile de programmer des commandes élaborées sur des systèmes robotiques complexes sans sacrifier aux exigences de robustesse et de réutilisabilité. Pour répondre à ces exigences, nous avons développé un outil d'aide à la spécification de tâches robotiques basées sur l'asservissement visuel (MINISIM) ; puis, nous avons utilisé et adapté un ensemble d'outils d'aide à la conception de tâches robotiques (le système ORCCAD) permettant le passage de la spécification à l'implémentation.

## 4.2 Un outil de spécification: MINISIM

La spécification de tâches d'asservissement visuel réalisables par un robot mobile dans un environnement structuré nécessite l'utilisation d'outils logiciels de calculs et de simulation.

Dans notre approche, le premier travail pour le roboticien est de décrire la tâche robotique désirée en termes de contraintes sur l'attitude du robot vis à vis de son environnement local en utilisant le formalisme des liaisons virtuelles. Cette interaction entre le robot et son environnement doit être ensuite traduite sous la forme d'un motif cible à atteindre dans l'image. Ce travail d'association entre une tâche robotique et un motif visuel à atteindre dans l'image induit de fastidieux calculs comme on a pu le voir dans le chapitre précédent (matrice d'interaction, le noyau de celle-ci...).

Durant cette phase de spécification, il est intéressant de comprendre le rôle que joue certains paramètres du motif dans la tâche robotique visuelle (exemple : la hauteur d'une plinthe, la largeur d'une porte ...). La simulation d'une tâche robotique et sa visualisation sur écran graphique sont très utiles pour obtenir une première validation de cette tâche. Pour une plus grande souplesse d'utilisation, les phases de spécification formelle et de simulation peuvent être couplées.

Pour répondre à ces souhaits, nous avons développé sous MATHEMATICA [1] ([Wolfram 91]), le logiciel MINISIM d'aide à la conception d'asservissement visuel pour un robot mobile. Nous avons choisi MATHEMATICA car les fonctions de calcul formel qu'il fournit sont complétées par de nombreuses fonctions de visualisation graphique et d'un langage de programmation...

MINISIM permet à l'utilisateur de placer un robot dans un environnement structuré et de définir un motif visuel à suivre par la caméra. Une fois le motif défini, le logiciel calcule de façon formelle la matrice d'interaction, son noyau et le type de mouvement de la caméra qui laisse le motif visuel invariant. Après cette phase de calcul, l'utilisateur dispose d'aides pour définir la commande associée

[1] calculateur formel de Wolfram research

au motif. Cette commande en boucle fermée peut être alors simulée par le logiciel. Une description plus complète du logiciel (fonctionnalités, environnement, architecture) est fournie en annexe D.2. Tout les résultats de simulation, calculs et graphiques présentés dans les chapitres et annexes précédents sont issus de MINISIM.

La spécification d'une tâche référencée vision, à l'aide de MINISIM, étant achevée, il reste ensuite à implémenter réellement cette tâche sur le robot mobile.

## 4.3 Un outil de conception : Le système ORCCAD

L'approche ORCCAD propose un ensemble de concepts pour définir une application robotique ; ces concepts ont été à la base de la réalisation d'un ensemble d'outils : le système ORCCAD. Une description détaillée de cette approche est donnée dans [Simon 92], [Coste-Manière 92], [Simon 91], [Borrelly 90].

**Remarque :** Nous conseillons au lecteur de consulter le glossaire en annexe D.1 pour lever les ambiguïtés de vocabulaire très fréquentes en robotique.

### 4.3.1 L'approche ORCCAD

#### 4.3.1.1 Les concepts

L'approche ORCCAD est d'abord un ensemble de concepts où une application est considéré comme étant une séquence d'actions réalisées par un système robotique dans l'objectif d'atteindre un but fixé par l'utilisateur. ORCCAD propose une approche cohérente allant de la spécification à l'implémentation de cette application qui peut être décomposée en un ensemble de commandes en boucle fermée appelées Tâche-robot [2]. Au niveau de l'implémentation, une Tâche-robot est un programme multi-tâches qui réalise des actions robotiques ; Il est construit à partir de tâches élémentaires temps-réel : les Tâches-module [3]. Une Tâche-robot sera donc constituée par un ensemble de Tâches-module interconnectées.

La Tâche-robot est définie par deux composantes :

- une loi d'asservissement élémentaire c'est à dire un schéma de contrôle structurellement invariant durant l'exécution de la tâche,

[2]notée également TR

[3]notées également TM

- un comportement logique associé à un ensemble de signaux susceptibles d'arriver durant la tâche.

A chaque TR correspond l'implémentation d'une loi de commande associée à une fonction de tâche. Au niveau informatique, elle est obtenue par la connexion de plusieurs Tâches-module et peut être répartie sur plusieurs cartes CPU. Le comportement logique est contrôlé par un automate à états finis dont les événements sont générés par des TM spécialisées : les observateurs. Des outils de vérification d'efficacité et de validité sont associés à la TR.

La Tâche-robot est obtenue en activant et en interconnectant l'ensemble de ses Tâches-module. La communication entre TM est définie par des ports virtuels (réutilisabilité) typés (plusieurs types de synchronisation).

#### 4.3.1.2 L'architecture du contrôleur robotique

Une application robotique doit être *facilement* accessible à des utilisateurs de différents niveaux de compétences. Par exemple, un utilisateur qui spécifie une application, telle que *Faire le tour d'une pièce* à un robot mobile, ne doit pas se soucier des problèmes d'allocation de ressources pour les tâches où les réglages de gains d'une commande. Le contrôleur de robot conçu à l'aide d'ORCCAD est accessible à trois niveaux de compétences différentes :

- **Niveau Application** destiné à l'utilisateur du système robotisé. Des outils de programmation de haut niveau d'abstraction lui permettent de spécifier une application, de l'exécuter et de la superviser.
- **Niveau Commande** accessible à l'automaticien chargé de générer les lois de commande et de réaliser les Tâches-robot. Ce niveau fournit les outils pour concevoir, programmer et tester une action robotique en choisissant la commande adaptée.
- **Niveau Système** s'adresse aux informaticiens (temps réel ou matériel) pour la conception de modules et de Tâches-module. Ce niveau assure l'interface avec le matériel, la répartition du code sur les ressources de calcul, la gestion des tâches (système temps réel, la communication...).

Il est a noté qu'une semblable décomposition d'une application de robotique sous-marine a été utilisée, en parallèle et sans concertation, au Naval Postgratued School à Monterey ([Byrnes 93]). Cette architecture dénommée "Rational Behavior Model" utilise trois niveaux voisins (Strategic level, Tactical level, Execution level). Une comparaison, plus en profondeur, de ces deux architectures est un travail qui reste à faire.

#### 4.3.1.3 Le système ORCCAD

Le système ORCCAD est un ensemble d'outils d'aide à la conception d'un contrôleur de robot générique associant la réactivité et la commande robotique. Ces outils permettent de concevoir, tester et implémenter des Tâches-robot et des Tâches-module. Il suit les phases de conception, présentées sur la figure 4.1, et le contrôleur obtenu est accessible (contrôleur ouvert) sur ses trois niveaux d'architecture aux utilisateurs concernés.

Dans le cadre de notre travail, nous avons utilisé les deux niveaux les plus près du système robotique (niveau commande et système). Le but était de constituer une bibliothèque de Tâches-robot référencée vision pour un robot mobile. Le niveau application qui consiste à faire coopérer plusieurs Tâches-robot référencée vision pour réaliser une application est un travail qui reste à faire.

Le processus de conception proposé pour la création d'une Tâche-robot est décomposé en plusieurs phases :

- la définition formelle d'une tâche de commande (i.e une fonction de tâche),
- le choix d'une loi de commande adaptée,
- la décomposition de la loi en un ensemble de Tâches-module,
- la définition des contraintes temporelles des Tâches-module,
- la définition du comportement de la TR,
- la simulation de la TR,
- l'implémentation de la TR.

La conception d'une Tâche-robot se fait en enchaînant les phases décrites précédemment qui permettent de passer progressivement de la spécification à l'implémentation. Lors de cette conception, l'utilisateur est guidé par une interface homme-machine ([Simon 92]). Nous allons maintenant détailler cette construction illustrée en figure 4.1.

#### 4.3.1.4 Spécification de la loi de commande en temps continu

L'utilisateur a spécifié en langage naturel : la tâche à réaliser, le système robotique utilisé et les événements à prendre en compte.

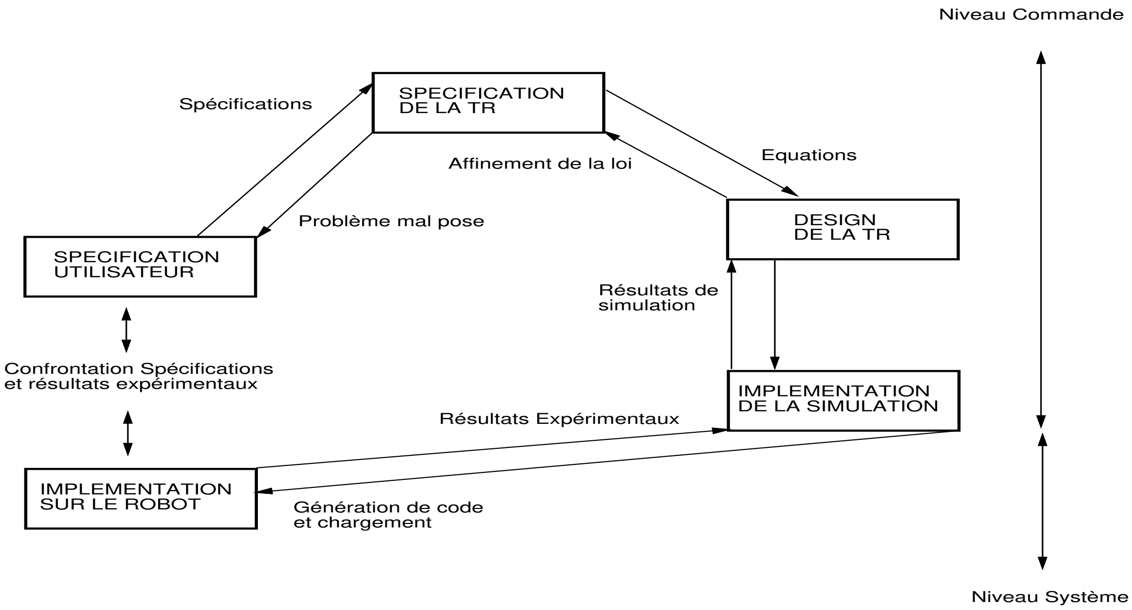



Figure 4.1 : Conception d'une tâche robot

A partir des ces spécifications, l'automaticien-roboticien doit construire une loi de commande qui satisfasse la tâche que le robot doit réaliser. Cette construction peut se faire en terme de blocs diagrammes et/ou d'équations. Cette loi doit être ensuite décomposée en objets fonctionnels ; ORCCAD guide alors l'utilisateur dans cette décomposition fonctionnelle. Dans l'implémentation actuelle, l'utilisateur dispose d'objets pré-définis pour le cas de la commande des bras manipulateurs. Il est envisagé d'utiliser la même approche pour les robots mobiles. Ainsi l'utilisateur pourrait disposer d'objets tels qu'un estimateur de position cartésienne, un générateur de trajectoires, des commandes dans l'espace cartésien, etc...

Le résultat de cette étape est l'obtention d'un graphe fonctionnel de la tâche robot où les noeuds du graphe représentent les objets fonctionnels et les arêtes les transferts de données entre les objets.

#### 4.3.1.5 Prise en compte des contraintes temporelles

Le schéma de spécification fonctionnel est ensuite enrichi en définissant les contraintes temporelles associées qui sont : la période de discrétisation, le temps de calcul estimé, le type de communication et de synchronisation, le code de calcul... L'élément de base de cette construction est la Tâche-module qui est l'objet fonctionnel à qui l'on a associé toutes les contraintes temporelles. Pour une période de $T$ ms, la structure de la Tâche-module est illustrée en figure (4.2) :

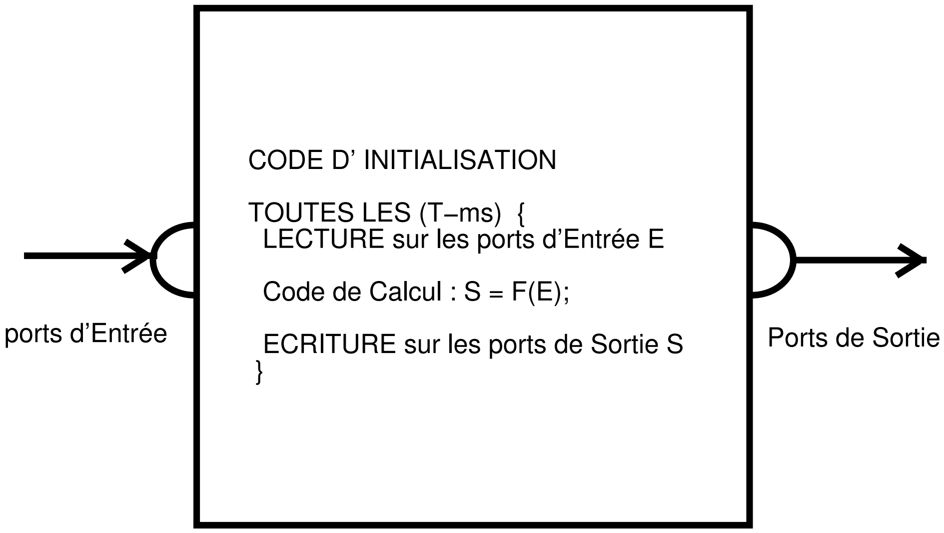



Figure 4.2 : Structure de la Tâche-module

La complexité des algorithmes de commande et le souci de réutilisabilité ont nécessité une approche orienté objet pour spécifier les Tâches-module. L'approche ORCCAD est destinée, en premier lieu, aux robots manipulateurs pour laquelle on dispose du formalisme des fonctions de tâches. Il existe plusieurs types de Tâches-module paramétrables :

- les ressources physiques : modélisation et interface avec un bras manipulateur,
- la commande : fonction de tâche, génération de trajectoire, modèle, commande,
- les observateurs de la TR
- l'automate de la TR
- de type général.

Le résultat de cette étape est un graphe de spécification de contraintes temporelles de la Tâche-robot.

#### 4.3.1.6 Comportement de la Tâche-robot

Le comportement logique de la Tâche-robot est spécifié par un automate qui reçoit et émet des signaux. Ces signaux sont émis par des Tâches-module spécialisées, appelées observateurs, et ils sont associés à un type de traitement prédéfini. Il existent trois types différents de signaux :

- les signaux de pré-conditions : leur présence est obligatoire au démarrage de la Tâche-robot.
- les signaux d'exceptions : ces signaux sont générés durant l'exécution de la tâche robot et indique qu'une anomalie a été détectée. Ils sont associés à trois types différents de traitement suivant l'importance du problème :

  - type 1 : cette exception nécessite une modification d'au moins un paramètre d'une Tâche-module et n'affecte pas la structure de la Tâche-robot.

  - type 2 : elle provoque l'arrêt de la Tâche-robot pour en activer une autre.

  - type 3 : ou "erreur fatale" qui conduit à l'arrêt total de l'application ou la mise en sécurité,
- signaux de post-conditions : leur apparition provoque l'arrêt normal de la Tâche-robot.

Le comportement de la Tâche-robot est spécifié en définissant les différents événements à prendre en compte. Le code du comportement est généré automatiquement en Estérel qui est un langage de programmation synchrone qui peut produire en sortie un automate. L'utilisateur dispose d'outils de visualisation (AUTOGRAPH) et de vérification (AUTO) permettant de valider le comportement de la Tâche-robot. L'automate de comportement de la TR est géré par une Tâche-module spécialisée de type Automate. La description du langage Estérel et des outils associés se trouve dans [Berry 92], [Roy 90].

#### 4.3.1.7 Simulation de la Tâche-robot

A partir de la description des Tâches-robot, il est possible de générer du code de simulation.

ORCCAD utilise SIMPARC ([Astraudo 92]) pour simuler le fonctionnement d'une Tâche-robot. SIMPARC est un simulateur hybride constitué par :

- un simulateur à événements discrets contenant un modèle de l'architecture supportant l'application et les programmes. Il permet de vérifier le bon fonctionnement du logiciel sur la machine cible (périodes d'échantillonnages, deadlock...).

- un simulateur en temps continu contenant les modèles du robot décrit sous forme d'équations différentielles permettant d'évaluer les lois de commande en tenant compte des contraintes d'implémentation.

Son architecture est illustrée sur la figure (4.3). Dans ce simulateur, chaque élément du système robotisé (processeur, bus, capteur, robot ...) est associé à un objet C++. On attribut des propriétés à ces objets pour instancier ces classes, comme par exemple un programme associé à l'objet processeur dont on simulera le temps d'exécution. Le moteur de simulation est dirigé par les événements pouvant provenir de la partie simulation en temps continu.

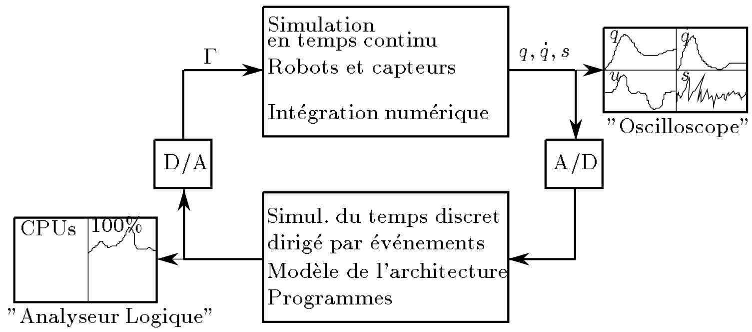



Figure 4.3 : Architecture Simparc

#### 4.3.1.8 Implémentation de la Tâche-robot

L'étape de simulation, ayant permis de régler et valider la loi de commande, il est possible de générer, toujours à partir des spécifications, le code temps-réel de la Tâche-robot qui sera exécuté sur le système robotique.

### 4.3.2 Adaptation d'ORCCAD pour la robotique mobile

La conception générique d'ORCCAD nous laissait à penser que nous pouvions l'employer dans le cadre de la robotique mobile autonome. Dans notre cas, l'implémentation des TR d'asservissement pour un robot mobile a malgré tout nécessité l'adaptation des deux niveaux les plus bas du système et de l'interface homme-machine.

Les modifications de l'interface ont consisté à regrouper tous les outils ORCCAD (éditeur graphique, simulateur, générateurs de code) et à définir une structure informatique commune [4]. Durant la phase de conception, l'utilisateur doit

[4]Je remercie Konstantinos Kapellos pour sa disponibilité et les nombreuses modifications qu'il a consenti à faire sur son interface

pouvoir facilement passer d'une étape à une autre dans le cycle décrit dans la figure (4.1). La mise au point expérimentale, délicate pour un robot mobile, en est ainsi facilitée.

Bien évidemment, l'approche objet pour la modélisation des Tâches-module dédiées aux bras manipulateurs n'a pu être utilisée. Seules les TM qui sont indépendantes de cette modélisation ont été employées. A partir de ces Tâches-module générales, nous avons construit une bibliothèque de TM dédiées aux robots mobiles qui assurent les fonctions telles que :

- l'interface ou la simulation d'un robot mobile "type char",
- la génération de trajectoire dans l'espace cartésien,
- le calcul d'erreur,
- la reconstruction cartésienne,
- la commande stationnaire, instationnaire dans l'espace cartésien,
- la commande-référencée-vision
- les observations du système robotique (arrêt d'urgence, dépassement de registres de comptage des positions, limites mécaniques...)

A partir de ces Tâches-module générales et en procédant à une étude plus exhaustive du monde de la robotique mobile, il serait possible de pré-définir des Tâches-module paramétrables en utilisant une approche objet. Les travaux récents en modélisation et commande pour les robots mobiles, que nous avons présenté au chapitre 2, permettent l'obtention de classes génériques. La figure (4.4) illustre un exemple possible de hiérarchie de la classe commande.

L'architecture du système robotique (que nous décrirons au chapitre 5), sur laquelle la TR doit être exécutée, a induit des modifications au niveau système. Il a fallu implémenter des nouveaux types de communication entre ports via un réseau. De plus l'environnement local dans lequel se déplace le robot mobile nécessite de prendre en compte un plus grand nombre d'événements que pour les bras manipulateurs. Pour des raisons d'efficacité, la prise en compte des événements par la tâche module automate a été modifiée et améliorée [5].

[5] Je tiens à remercier Eve Coste-Manière pour son aide précieuse lors de ces modifications

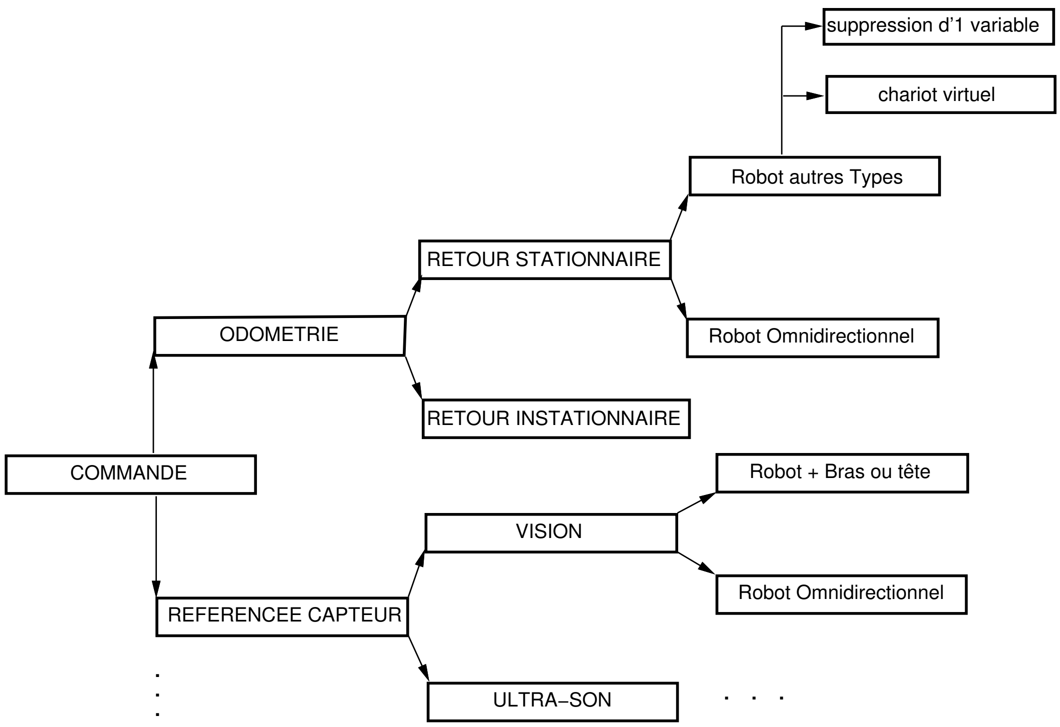



Figure 4.4 : Classe Commande

## 4.4 Perspectives

Nous avons vu au chapitre précédent, qu'une application robotique basée sur un enchaînement d'asservissements visuels nécessitait de combiner l'aspect continu (loi de commande) et l'aspect événementiel ("fenêtre active"). Le système ORCCAD, bien qu'il soit dédié à l'origine aux bras manipulateurs, apparaissait capable d'extension à d'autres classes de systèmes robotiques. L'adaptation, que nous avons proposé et réalisé, a validé cette première intuition. La poursuite de ce travail consistera à utiliser les récentes avancées en modélisation et commande des robots mobiles pour spécifier une modélisation objet de la décomposition d'un contrôleur de robots mobiles. Cette modélisation sera ensuite intégrée dans le système ORCCAD. Cette même approche pourrait être appliquée à la robotique sous-marine par exemple, en utilisant des modélisations objets existantes ([Rigaud 93]).

Au vue de l'utilisation de la première maquette rassemblant les outils ORCCAD, on peut proposer plusieurs améliorations qui seront intégrées dans les versions futures :

- au niveau de la simulation SIMPARC, il sera nécessaire d'intégrer les aspects liés à l'utilisation de capteurs extéroceptifs, ainsi que ceux liés à l'OS temps-réel,

- le générateur de code VXWORKS devra être amélioré,
- les outils d'analyse de performances (visualisation de données, timing...) devront être complétés.

L'étape suivante sera de passer des concepts à l'implémentation en ce qui concerne l'enchaînement des Tâches-robot. Un axe de recherche prometteur découle des travaux de [Rutten 92] visant à mettre à la disposition de l'utilisateur un ensemble de plans qui permettrait la définition d'un langage métier pour chaque type de robots. Mais avant de disposer d'un tel niveau de programmation, il est indispensable de valider l'implémentation d'un asservissement visuel sur un système robotique en utilisant l'ensemble de l'environnement de programmation décrit. C'est ce que nous allons présenter dans le dernier chapitre de cet ouvrage.

# Chapitre 5

# Validation Expérimentale

Nous avons désiré disposer d'un outil capable de valider et d'étudier des problèmes liés à l'utilisation de capteurs dans la boucle de commande telle que l'asservissement visuel. La validation de notre approche réactive basée sur la vision a donc nécessité la réalisation d'un robot mobile expérimental spécifique. Nous avons participé à toutes les phases : de sa conception, à sa réalisation et aux essais.

## 5.1 Le système robotique Anis

### 5.1.1 Spécifications du robot mobile

Les applications typiques que le robot devait être à même de réaliser, concernaient l'exploration automatique d'environnement d'intérieur (scène de *type bureau*). Ainsi les dimensions de la base mobile devaient être suffisamment réduites pour permettre au robot d'effectuer des manœuvres en milieu encombré, de franchir des portes, de passer éventuellement sous des tables, etc...

Nous avons voulu que ce robot mobile expérimental soit ouvert c'est à dire que l'on puisse faire évoluer sa mécanique, son électronique, ses capteurs embarqués... En outre, il était souhaitable de disposer d'un robot autonome, sans ombilical ni liaison radio, pour accomplir des missions de courte durée d'une quinzaine de minutes environ. En revanche pendant la phase de développement, celui-ci devait pouvoir être muni d'un ombilical assurant l'alimentation et la communication avec le système de développement.

Il est à noter que la réalisation d'un premier robot expérimental Youpi par l'isia-Ecole des Mines de Paris et l'essi-Université de Nice (cf. [Pissard 90]) a permis de tirer des enseignements très utiles aux spécifications du robot mobile ANIS.

Les spécifications élaborées en commun par la société ECA et les partenaires publics sont décrites dans le document [ECA 92]. A partir de celle-ci, la société ECA a étudié ([ECA 93a]) et réalisé le robot mobile. Ce robot s'insère dans notre système robotique ouvert dédié à l'expérimentation. Nous décrivons ce système dans la partie qui suit.

## 5.1.2 Description du système robotique

### 5.1.2.1 L'architecture

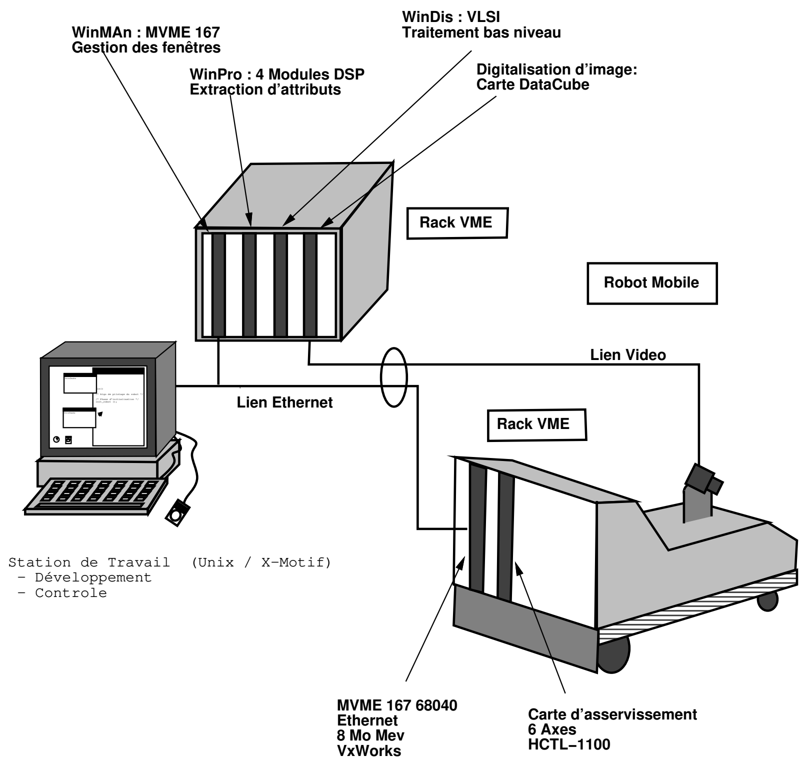



Figure 5.1 : Le système robotique

Le système robotique expérimental destiné à valider les approches réactives basées sur la vision, est composé de plusieurs sous-systèmes illustrés en figure (5.1) :

- un robot mobile équipé d'une tourelle munie d'une caméra,

- un système de vision dédié à l'asservissement visuel,
- un système de développement et de téléchargement du logiciel embarqué.

En phase de développement et d'essais, un réseau ETHERNET permet la communication entre les différents composants du système robotique. Dans le cas d'une mission demandant une autonomie complète du robot, le système de vision serait embarqué.

Avant de présenter les résultats expérimentaux obtenus avec notre système, nous allons décrire les principales caractéristiques des différents composants le constituant.

#### 5.1.2.2 La base mobile

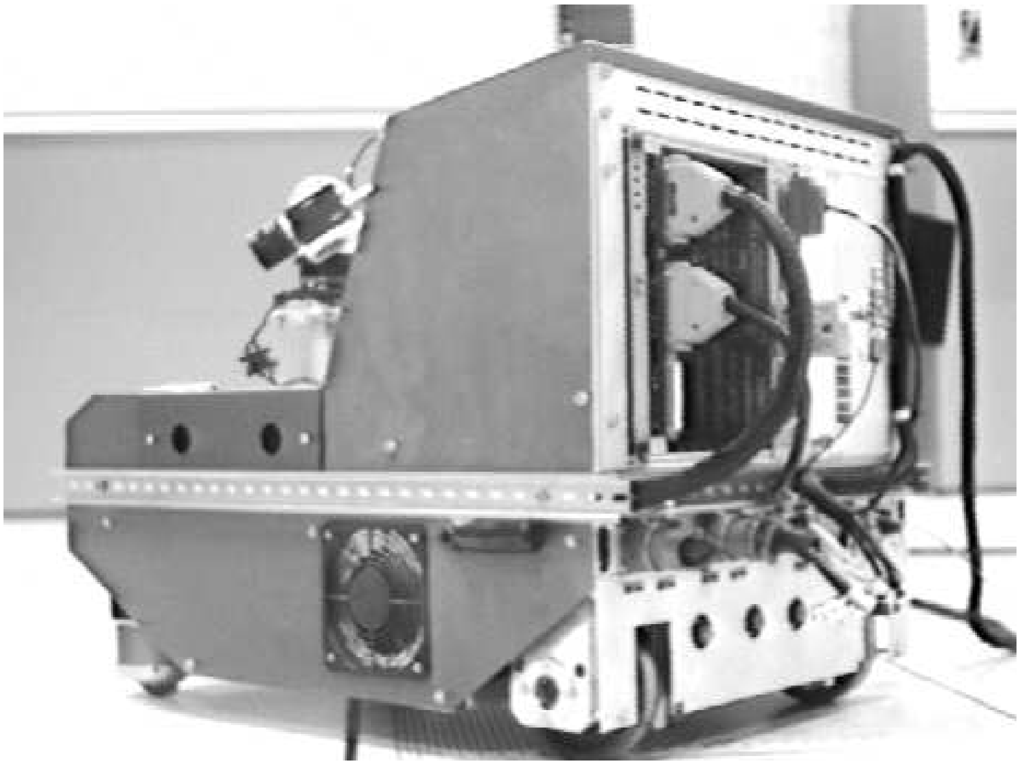


Figure 5.2 : Le robot mobile

L'élément clé de notre système robotique, réalisé à partir des spécifications énoncées auparavant, est un robot mobile de "type Char" : il dispose de deux roues motrices à l'arrière, la direction est assurée par la commande en différentiel de ses deux roues. Les deux autres roues stabilisatrices et libres sont sphériques

| | | | |
|---|---|---|---|
| Longueur : | 700 mm | Masse : | 75 Kg |
| Largeur : | 500 mm | Vmax : | 1 $ms^{-1}$ |
| Hauteur : | 600 mm | Acc : | 1 $ms^{-2}$ |

Figure 5.3 : Caractéristiques du Robot Mobile

pour minimiser les problèmes de contacts qui entraînent des écarts parasites de trajectoire lors des démarrages et dans les petites manœuvres. La mécanique de la base mobile est configurable également en version tricycle (une roue directrice et motrice) par démontage (cette version est prévue mais non réalisée).

Les dimensions et les performances du robot, données dans le tableau (5.3), ont été choisies pour permettre d'évoluer dans un environnement d'intérieur : Le contact permanent des quatre roues au sol est assuré par un système à balancier placé sur l'axe des roues stabilisatrices et la garde au sol du robot est de 30 mm. L'utilisation de différents capteurs embarqués est bien sûr prévu sur le robot :

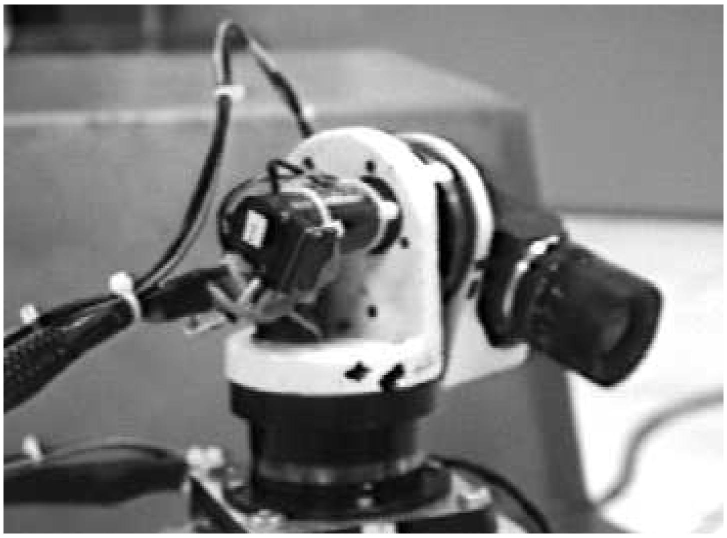


Figure 5.4 : La “tête” du robot mobile

- les codeurs optiques mesurent la position angulaire des roues motrices,
- un mécanisme à deux degrés de liberté (angles en lacet et tangage) est placé sur l'avant du robot pour diriger la caméra (figure (5.4)),
- un rail extérieur ceinturant le robot permet la pose d'autres capteurs (proximètres...).

#### 5.1.2.3 Les calculs embarqués et le système de développement

Les moyens de calculs embarqués sont construits autour d'un rack VME d'une capacité de douze cartes. Pour nos expérimentations, nous utilisons une carte CPU MOTOROLA MVME 167 ( 68040, 33 Mhz, 8 Mo mémoire) avec le système temps-réel VxWorks et une carte d'asservissement six axes ASS6 que nous avons développée.

Entièrement programmable (période, gains, modes de commande...), elle assure l'asservissement numérique des quatre axes du robot ( les deux roues motrices et les deux axes de la tête). L'élément de base de cette carte est le composant dédié d'Hewlett-Packard HCTL-1100 qui autorise plusieurs modes de contrôle (position, vitesse, trapézoïdal) pour différents types de moteurs (courant continu à aimant permanent, brushless, pas à pas).

Lors du développement d'une Tâche-robot, la spécification à l'aide d'ORCCAD est faite sur une station de travail Sun-Sparc, puis validée en simulation ; ensuite le code VxWorks généré est téléchargé sur la carte CPU du robot via ETHERNET. Au cours des phases d'essais, l'exécution de la Tâche-robot est contrôlée depuis la station de travail.

#### 5.1.2.4 Le système de vision dédié

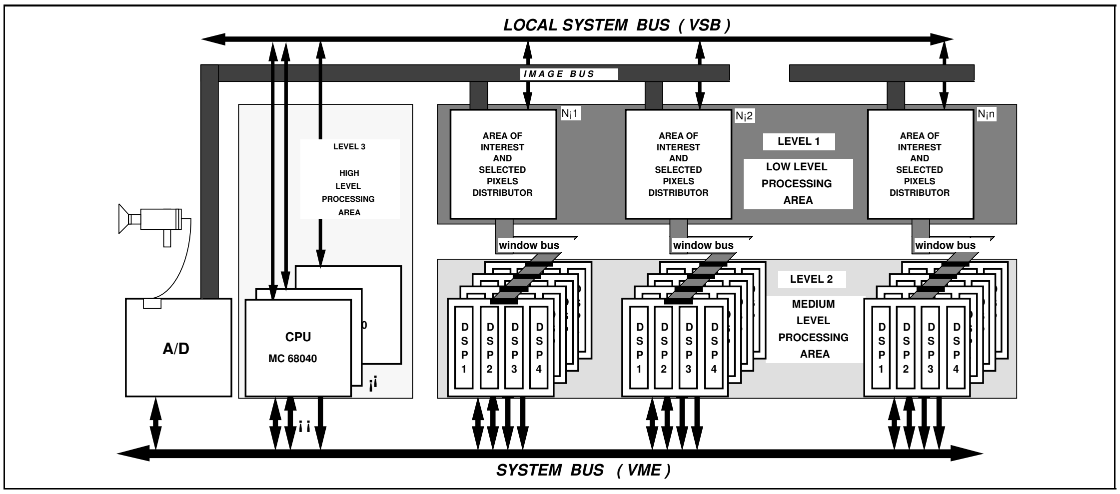



Figure 5.5 : Le Système de Vision Dédié

L'implémentation efficace d'une commande de robot basée sur la vision nécessite de résoudre plusieurs problèmes au niveau du traitement d'image. Les principales difficultés sont induites par les contraintes temps-réel et le traitement d'un flot important de données. Pour répondre à ces difficultés, un système de vision original a été développé ([Rives 93c], [Martinet 91]) caractérisé par sa modularité et ses capacités temps-réel. Son architecture, illustrée en figure (5.5), est basée sur des composants VLSI pour le traitement d'image bas niveau (filtres de convolution, seuillage pour l'extraction de contours) et des processeurs DSP pour les traitements de plus haut-niveau (extraction d'attributs...). Le système de vision utilise le concept de fenêtre active : une fenêtre active est attachée à une région d'intérêt particulière dans l'image et assure l'extraction de paramètres d'un motif désiré dans cette région. A chaque fenêtre active est associé un filtre temporel capable de suivre la primitive visuelle sur la séquence d'image. Plusieurs fenêtres actives peuvent être définies dans l'image et plusieurs types de traitement peuvent être affectés dans chaque fenêtre.

Pour des facilités de développement, le système de vision est placé dans un rack VME extérieur au robot mais il pourra être embarqué dans le cas d'un fonctionnement sans ombilical.

## 5.2 Résultats Expérimentaux

### 5.2.1 La tâche robot d'asservissement cartésien

Il était raisonnable de tester le robot mobile et son environnement de programmation sur une TR simple utilisant moins de ressources que la TR référencée vision. La TR d'asservissement cartésien est un exemple parfait qui nous a également permis de valider expérimentalement des algorithmes de commande très élaborés ([Samson 90b], [Samson 91b]).

Au niveau de la commande nous avons testé deux lois différentes : la stabilisation par retour d'état stationnaire sur un chariot virtuel et la stabilisation par retour d'état continu non stationnaire sur une configuration terminale. Nous allons présenter brièvement ces deux lois de commande avant de détailler la structure de la Tâche-robot destinée à les implémenter.

**La stabilisation par retour d'état stationnaire**

Cette stabilisation consiste à poursuivre une trajectoire de référence définie dans le repère $(R_O)$ par $\begin{pmatrix} x_r(t) & y_r(t) & \theta_r(t) \end{pmatrix}^T$ d'un "chariot virtuel" qui a pour vitesse en translation/rotation à l'entre-axe de ces roues (point de référence choisi dans notre implémentation) $\begin{pmatrix} v_r(t) & \dot{\theta}_r(t) \end{pmatrix}^T$. L'erreur $\begin{pmatrix} x(t) & y(t) & \tilde{\theta}(t) \end{pmatrix}^T$ entre la trajectoire du chariot réel et celle de référence est exprimé dans le repère

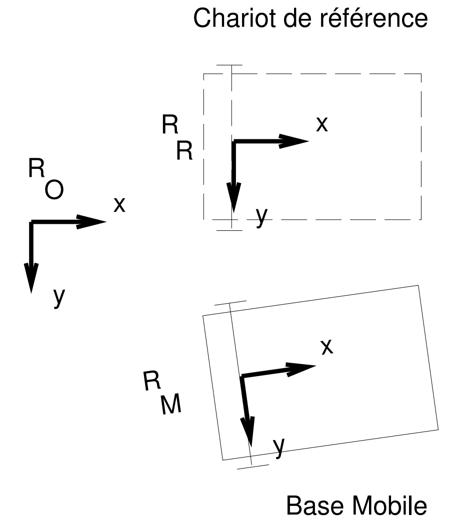



Figure 5.6 : Les repères utilisés

du chariot ($R_M$). Pour de petites erreurs, la commande linéarisée en $\begin{pmatrix} v & \dot{\theta} \end{pmatrix}^T$ est de la forme :

$$\begin{cases} v = g_1 x \\ \dot{\theta} = g_2 v_r y - g_3 \tilde{\theta} \end{cases} \tag{91}$$

Tout les détails sur l'établissement de ce type de lois de commande se trouve dans ([Ait-Abderrahim 93] ou [Samson 90b]).

**La stabilisation par d'état continu instationnaire**

Bien qu'il ne soit pas possible de stabiliser une base mobile sur une configuration terminale par un retour d'état stationnaire, il a été démontré qu'il existait des retours d'état continu instationnaire pour le faire ([Samson 91b]). Nous avons choisi une commande en $\begin{pmatrix} v & \dot{\theta} \end{pmatrix}^T$ qui dépend du temps :

$$\begin{cases} v = y^{\frac{1}{3}}\ sin(\omega t) + g_1\ x\ cos(\tilde{\theta}) \\ \dot{\theta} = \frac{2}{3}\ g_2\ y^{\frac{1}{3}}\ v \frac{sin(\tilde{\theta})}{\tilde{\theta}} + g_3\ \tilde{\theta} \end{cases} \tag{92}$$

où l'erreur entre la configuration terminale et le robot mobile $\begin{pmatrix} x(t) & y(t) & \tilde{\theta}(t) \end{pmatrix}^T$ est exprimée dans le repère fixe ($R_0$) . Il est montré dans ([Samson 91b]) que ce type de commande stabilise le chariot sur la configuration terminale.

**La Tâche-robot**

Un fois que la loi de commande est établie, il faut procéder à sa décomposition en schéma-bloc. La décomposition de la TR en TM est présentée en figure (5.7). La paramétrisation des TM se fait, pour l'instant, par fichier et permet à l'utilisateur de choisir le type de commande, les conditions initiales et la trajectoire de consignes.

La TM TF calcule une erreur dans l'espace cartésien $\begin{pmatrix} x & y & \tilde{\theta} \end{pmatrix}^T$ en comparant les consignes générées par la TM TG de génération de trajectoire et l'état reconstruit par la TM CARTESIAN. Cette erreur est utilisée pour le calcul de la commande (TM CONTROL) qui envoie à la TM MOBILE les consignes vitesses $(\dot{q}_d, \dot{q}_g)$. Cette TM assure l'interface avec la base mobile : l'envoi des commandes et la lecture des positions des axes des roues. La reconstruction cartésienne, assurée par la TM CARTESIAN est uniquement réalisée en utilisant les capteurs odométriques.

Toutes les TM de la Tâche-robot, exceptées la TM MOBILE et la TM AUTOMATE, sont périodiques. L'utilisateur peut fixer les périodes d'activation; dans cet exemple la période utilisée est de 10 ms. La Tâche-module MOBILE d'interface avec le robot envoie une consigne en vitesse à une carte d'asservissement qui calcule numériquement les couples moteurs avec une période d'échantillonnage de 1 ms.

Le type de synchronisation pour les communications entre les tâches périodiques est de type asynchrone-asynchrone. En revanche, l'envoi des données vers les TM non périodiques est fait de manière asynchrone-synchrone pour les synchroniser. Pour avoir plus de détails sur les mécanismes de synchronisations, on conseille de se reporter à [Simon 92].

Le comportement de la tâche réagit à quatre événements qui sont :

- un signal de pré-condition pour lancer la tâche robot (touche appuyée au clavier),
- trois signaux qui provoquent l'arrêt de la tâche

  - durée d'exécution dépassée,

  - dépassement des registres hardware de comptage des capteurs odométriques

  - arrêt d'urgence actionné par l'utilisateur

**Les Résultats**

Une fois que le schéma de la Tâche-robot est établi, l'utilisateur peut pré-régler les lois de commande en simulation et finalement les faire exécuter sur le robot.

Pour la stabilisation à retour d'état stationnaire, nous avons choisi comme trajectoire de référence une fonction sinusoïdale telle que :

$$\left( \; v_r(t) = 0.5 \; sint \quad \dot{\theta}_r(t) = 0 \; \right)^T$$

Les gains optimaux choisis sont $g_1 = 1.0$, $g_2 = 1.0$ et $g_3 = 1.41$. Les résultats expérimentaux (la trajectoire reconstruite du robot, les consignes en vitesse sur les moteurs, la position et l'orientation du robot dans le repère fixe $(R_O)$) sont présentés sur la figure (5.8).

Pour la stabilisation par retour d'état instationnaire, le point terminal à atteindre est l'origine du repère fixe $(R_O)$ avec une orientation nulle. Les gains et la pulsation ont été réglés expérimentalement à $g_1 = 1.0$, $g_2 = 2.0$, $g_3 = 1.0$ et $\omega = 1.0$ et les résultats sont donnés en figure (5.9).

Durant la mise au point, il est apparu évident que la reconstruction de la position du robot sans capteurs extéroceptifs est très sensible aux bruits, aux patinages des roues dues aux accélérations. Pour valider la commande nous n'avons donc pas utilisé des gains très élevés. Les performances de la Tâche-robot peuvent donc être améliorées par la mise en place de filtres, de gains variables suivant la vitesse du robot, d'utilisation de capteurs extéroceptifs... Ces améliorations dépassaient le but de cette expérimentation qui était bien de valider l'environnement de programmation qui permet de passer facilement d'une loi de commande "théorique" à son implémentation.

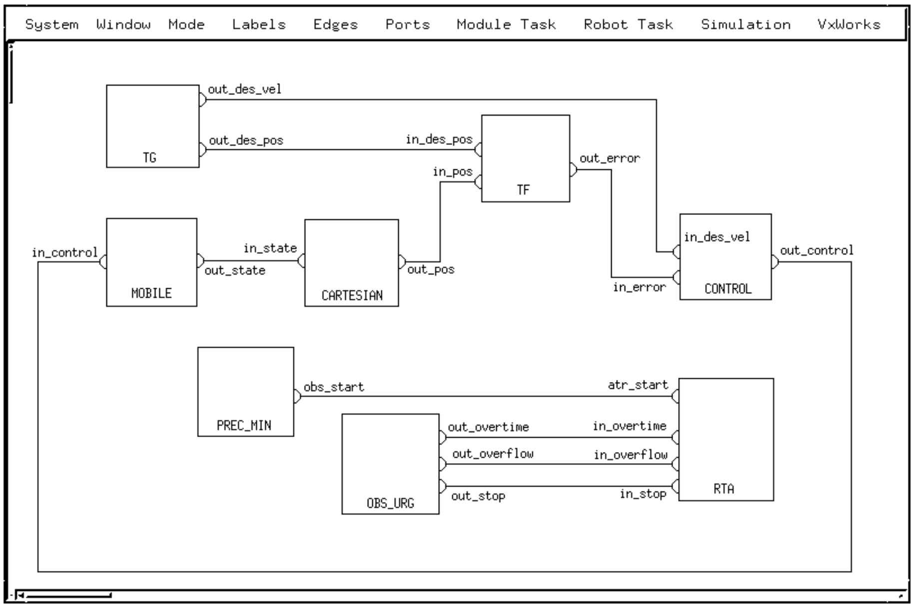



Figure 5.7 : Asservissement Cartésien

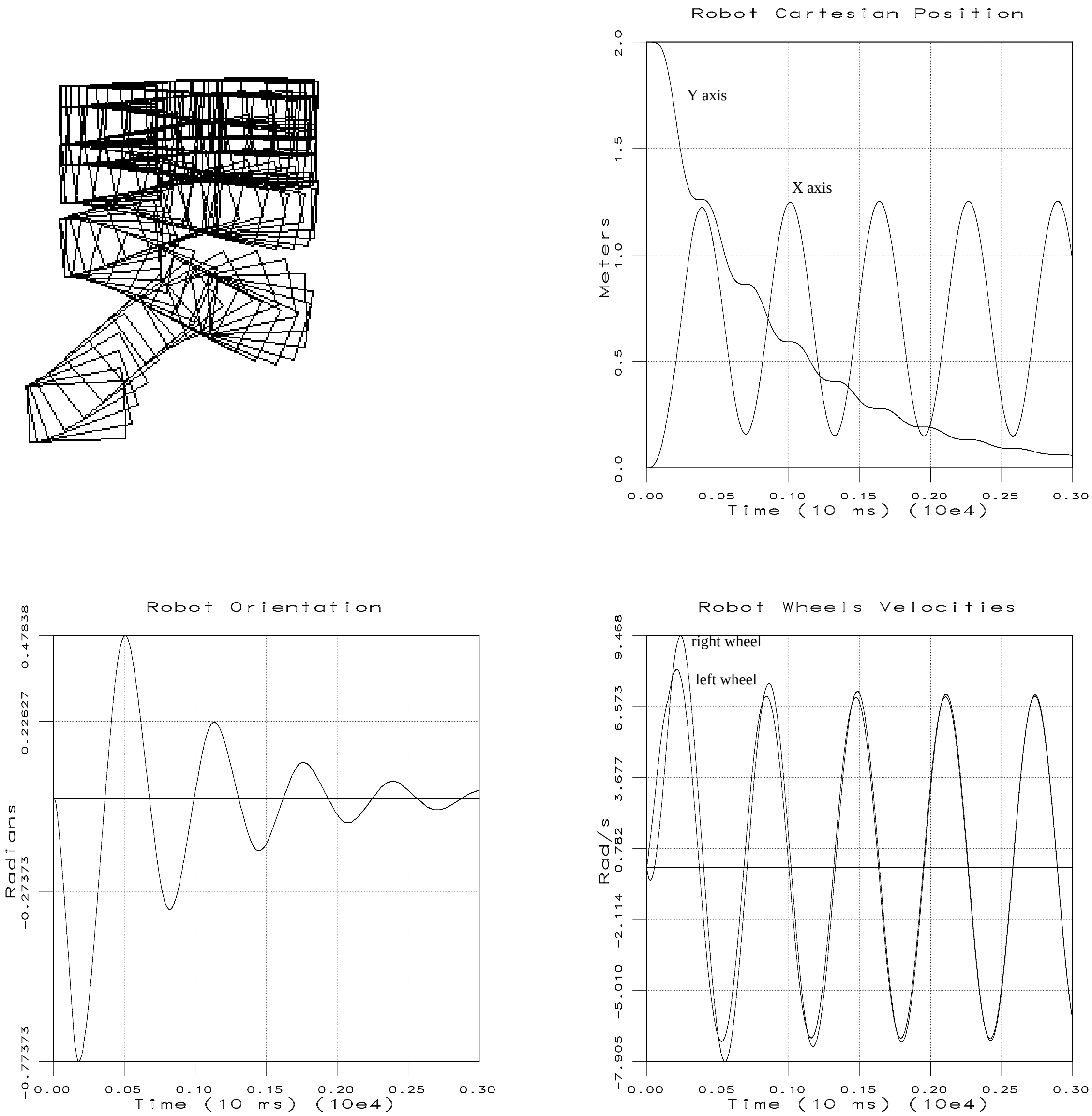



Figure 5.8 : Stabilisation sur un chariot de référence

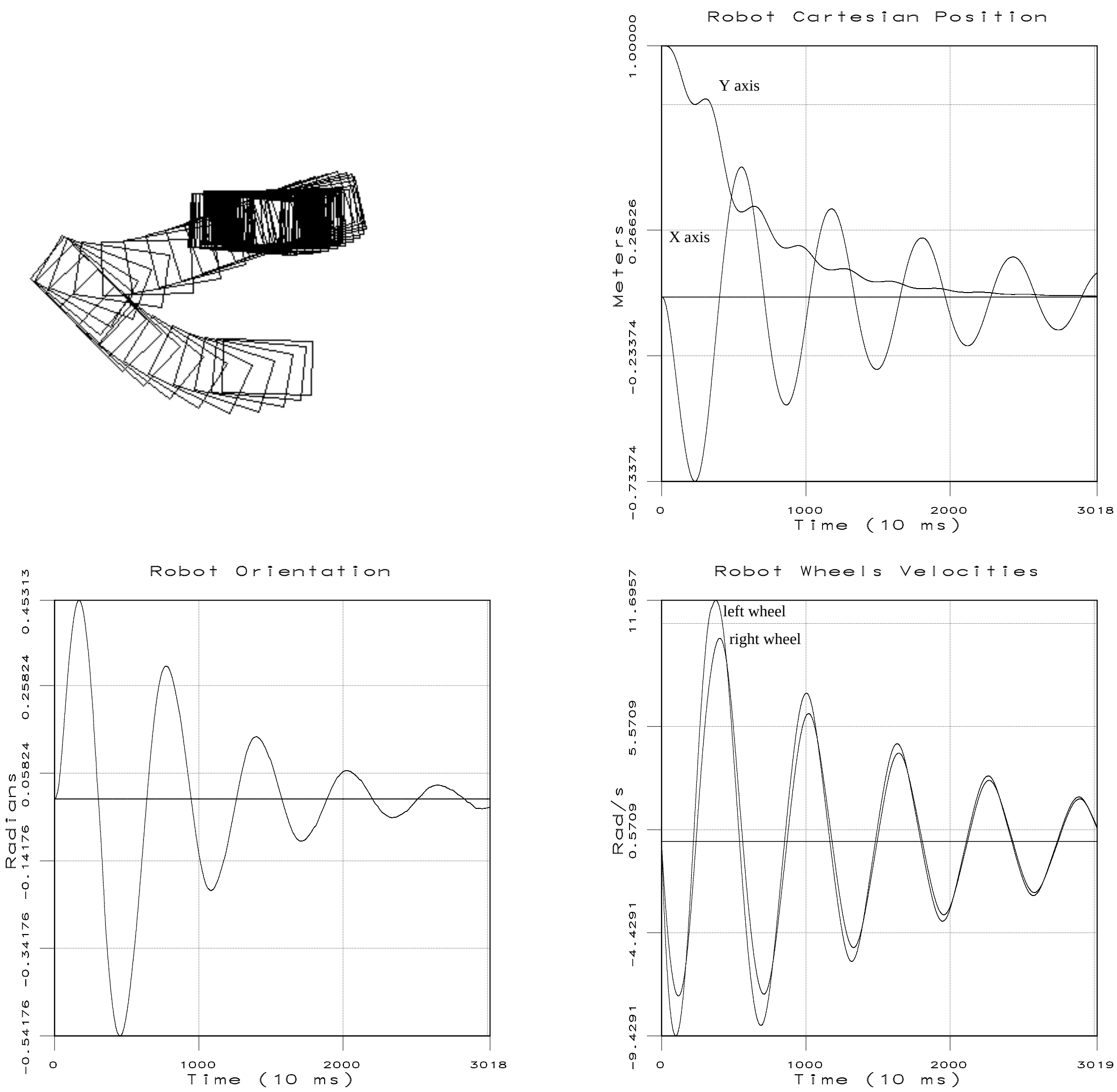



Figure 5.9 : Stabilisation instationnaire sur une configuration terminale

### 5.2.2 La tâche robot d'asservissement visuel

La loi de commande-référencée-vision a été découpée en 7 TM périodiques qui communiquent entre elles :

- la TM GT_VS génère une trajectoire de consigne dans l'espace image $\underline{s}^*(t)$,
- la TM WINFEAT assure l'interface avec les cartes spécialisées pour l'asservissement visuel; elle fournit en sortie les paramètres du motif image courant $\underline{s}(t)$ ,
- la TM ERROR_VS calcule la fonction d'erreur entre le motif de consigne et le motif courant $\underline{\epsilon} = \underline{s}(t) - \underline{s}^*(t)$,
- la TM TF_VS calcule le torseur cinématique caméra en fonction de l'erreur sur le motif visuel $\underline{T_C} = -\lambda.L^T_{s=s^*}.\epsilon$,
- la TM INVJAC calcule la jacobienne $J^{-1}$ inverse du robot (ensemble mécanique constitué par le robot et le bras manipulateur) ,
- la TM CIN utilise la jacobienne inverse et le torseur cinématique caméra pour fournir les vitesses articulaires soit $\underline{\dot{q}} = J^{-1}.\underline{T_C}$,
- la TM ANIS assure l'interface avec le robot (envoi des consignes vitesses au robot et lecture des positions articulaires)

La période d'activation (10 $ms$) pour les TM de calcul de la jacobienne du robot (INVJAC, CIN, ANIS par synchronisation) est plus faible que celle (100 $ms$) associée aux TM de calcul visuel (WINFEAT, ERROR_VS, TF_VS, GT_VS) dont la limite est fixée par la période vidéo. Une fois la mise au point des cartes de vision terminée, nous atteindrons cette limite (40 $ms$).

Les événements qui définissent le comportement du robot (TM RTA_VS) peuvent provenir de multiples sources :

- de l'utilisateur qui désire lancer ou arrêter la Tâche-robot (action au clavier),
- de l'environnement visuel local (fenêtres d'observation spécifiées dans l'image) qui signale la présence d'un obstacle, une fin ou une perte du motif choisi dans l'image,
- du robot dont les limites mécaniques sont atteintes ou les registres de comptage odométrique sont saturés,
- d'une surveillance de la durée de la commande.

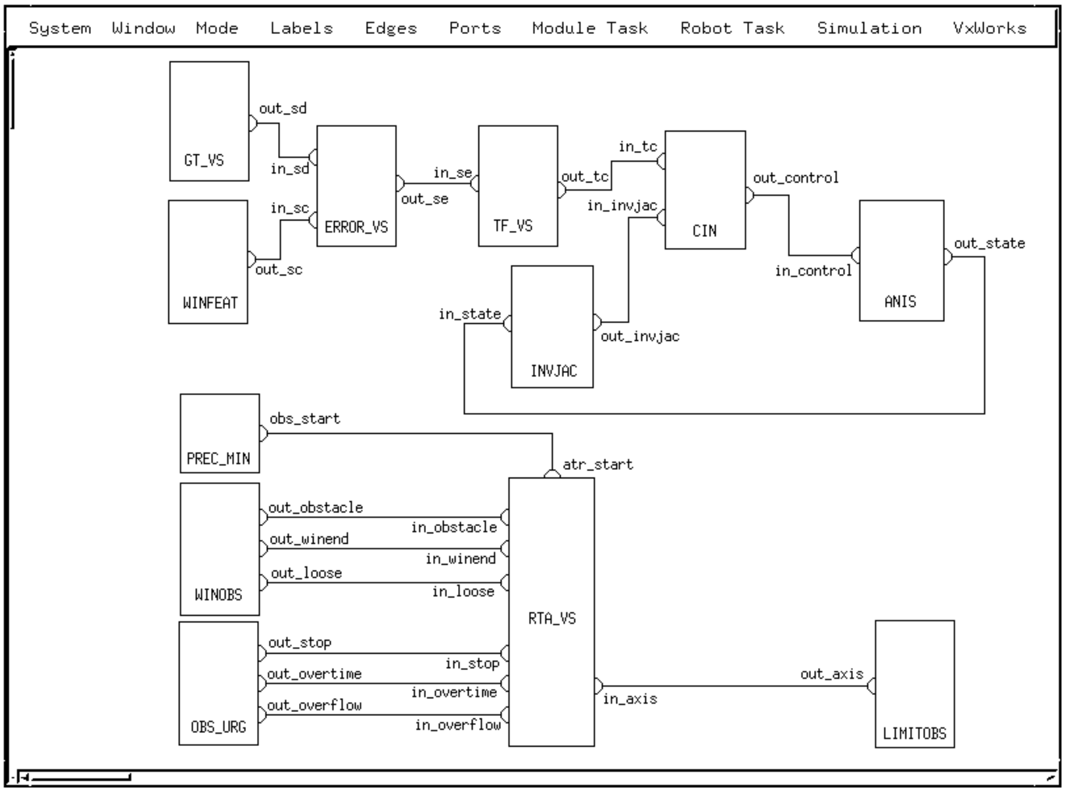



Figure 5.10 : Asservissement Visuel

Après la description de la Tâche-robot, il reste à la configurer en choisissant le type de motif sur lequel on désire réaliser l'asservissement visuel. Nous proposons un exemple de configuration que nous avons conduit jusqu'à l'expérimentation.

#### 5.2.2.1 Le positionnement vis à vis d'un cône

Nous désirons expérimenter une Tâche-robot d'asservissement visuel en positionnement vis à vis d'un motif en contrôlant trois degrés de liberté de notre robot (la base mobile + le lacet de la tourelle). Cette Tâche devrait permettre de valider et d'illustrer l'approche de coopération entre la base mobile et la tourelle supportant la caméra que nous avons décrite dans l'étude théorique. Le choix du motif 3D à suivre a été guidé par des raisons pratiques. Nous avons choisi un cône comme motif visuel pour positionner la caméra dans le but de réutiliser les algorithmes de fenêtrages et d'extraction déjà implémentées sur le système de vision dans le cadre de la commande d'un bras manipulateur ([Rives 93a]).

Nous allons brièvement décrire le choix du paramétrage de la Tâche-robot d'asservissement visuel pour permettre d'implémenter la tâche de positionnement vis à vis d'un cône (cf. figure (5.11)).

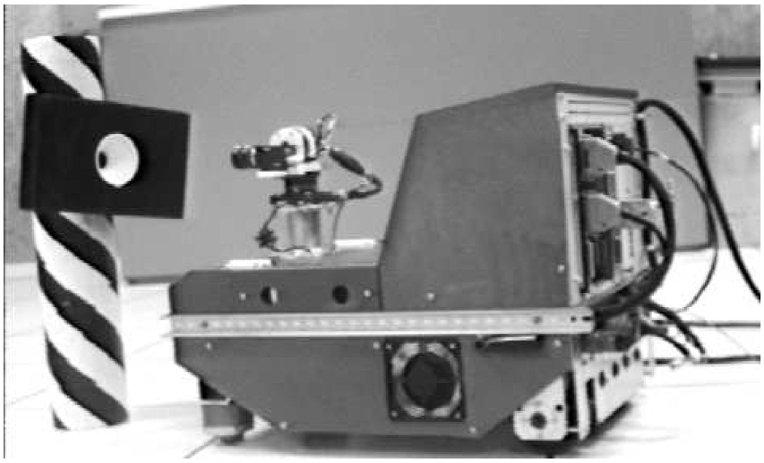


Figure 5.11 : Positionnement vis à vis du cône

D'une façon générale, le cercle inférieur et le cercle supérieur constituant les bords du cône se projettent dans l'image sous la forme de deux ellipses (cf. figure (5.12)). Ces ellipses constituent le motif visuel courant extrait de l'image caméra.

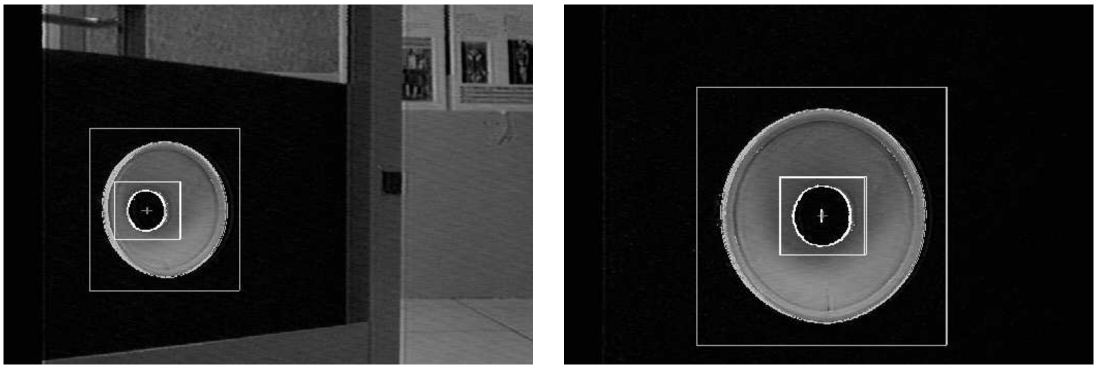


Figure 5.12 : Motif en début et en fin de Tâche

Si l'axe optique de la caméra est aligné avec l'axe de révolution du cône (cf. figure (5.13)), le motif projeté est formé par deux cercles concentriques. Ce motif constitue l'image cible à atteindre traduisant le fait que "la caméra est en face du cône".

Le paramétrage d'une ellipse et le calcul de sa matrice d'interaction sont détaillés dans ([Chaumette 90] et [Rives 93a]).

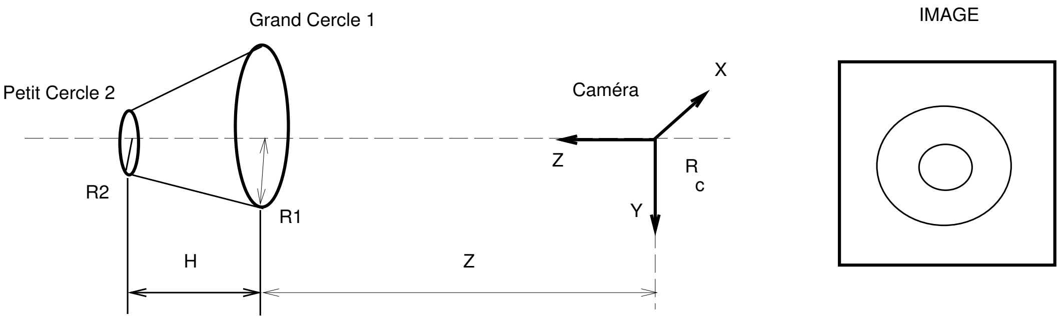



Figure 5.13 : Position Nominale

La représentation des ellipses retenues pour des raisons de simplicité au niveau du traitement de l'image, est la représentation par moments réduits. $(x_{ci}, y_{ci}, \mu_i^{1,1}, \mu_i^{2,0}, \mu_i^{0,2})$ avec $i = [1,2]$ pour les deux ellipses dans l'image. A partir de ces paramètres, on constitue le vecteur des paramètres du motif à atteindre. Du fait que la dimension de la tâche d'asservissement visuel à réaliser est réduite à trois, nous utiliserons seulement une partie des dix paramètres caractérisant les deux ellipses. On choisit :

$$\underline{s} = \begin{pmatrix} x_{c1} \\ x_{c2} \\ \zeta \end{pmatrix}$$

avec $\zeta = \frac{R_1}{\sqrt{\mu_1^{2,0} + \mu_1^{0,2}}}$

En configuration nominale (cf. figure (5.13)), l'axe optique de la caméra est aligné avec l'axe de révolution du cône à une distance $Z$ du centre du grand cercle de l'ellipse. Les paramètres ont alors comme valeur :$x_{ci} = y_{ci} = 0$ et $\mu_i^{2,0} = \mu_i^{0,2}) = R_i^2$ pour $i = [1,2]$. Le vecteur $\underline{s}^*$ nominal a ainsi pour expression :

$$\underline{s}^* = \begin{pmatrix} 0 \\ 0 \\ \frac{1}{\sqrt{2}} \end{pmatrix}$$

A ce point d'équilibre, la matrice d'interaction a pour expression :

$$L^T_{s=s^*} = \begin{pmatrix} -\frac{1}{Z} & 0 & 0 & 0 & -1-\left(\frac{R_1}{Z}\right)^2 & 0 \\ -\frac{1}{Z+H} & 0 & 0 & 0 & -1-\left(\frac{R_1}{Z+H}\right)^2 & 0 \\ 0 & 0 & \frac{\lambda}{R_1} & 0 & 0 & 0 \end{pmatrix} \tag{93}$$

On voit immédiatement que cette matrice d'interaction permet le contrôle de trois degrés de liberté de la caméra : en translation suivant son axe optique et l'axe $\vec{x}$ et en rotation d'axe $\vec{y}$. Ces degrés de liberté correspondent bien à

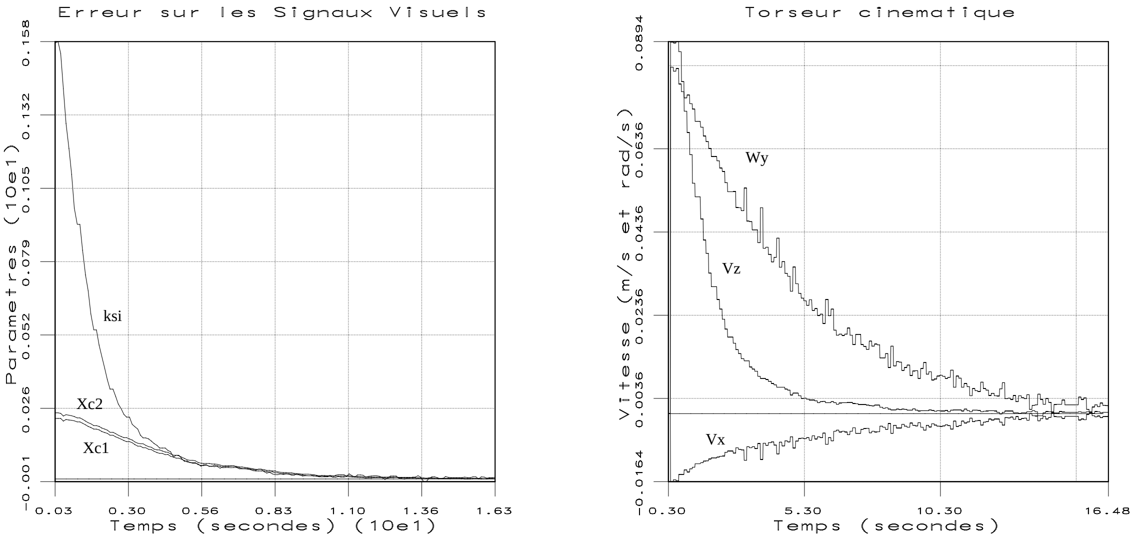



Figure 5.14 : Relevés pour un positionnement

ceux que l'on désirait contrôler au niveau de la base mécanique. Il y a donc bien cohérence entre le motif visuel de l'asservissement et le type de la tâche que l'on désirait réaliser.

Les résultats expérimentaux de cette Tâche-robot de positionnement visuel sont donnés en figure (5.14). On peut observer l'évolution du signal d'erreur $\epsilon$ dans l'espace image et celle du torseur cinématique de consigne pour la caméra durant la phase de positionnement vis à vis du cône. L'image de départ et l'image d'arrivée sont sur la figure (5.12). On y remarque bien que le motif de départ se positionne sur le motif de consigne (deux cercles concentriques).

Nous avons testé cette même commande pour le suivi du cône mobile. Le fait de ne pas estimer la vitesse de la cible pour en tenir compte dans la commande est bien sûr pénalisant au point de vue des performances. Pour des vitesses de la cible peu élevés, la caméra (donc le robot) suit le cône, malgré l'erreur de traînage. Une trajectoire reconstruite du robot durant un suivi du cône déplacé par l'utilisateur est visualisée en figure (5.15).

## 5.3 conclusion

Nous aurions désiré implémenter et tester un plus grand nombre de tâches référencées vision (suivi de ligne, rotation autour d'un segment vertical,...) mais nous avons préféré dans le temps qui nous était imparti traiter un exemple en proposant une chaîne complète d'implémentation d'asservissement visuel. A par-

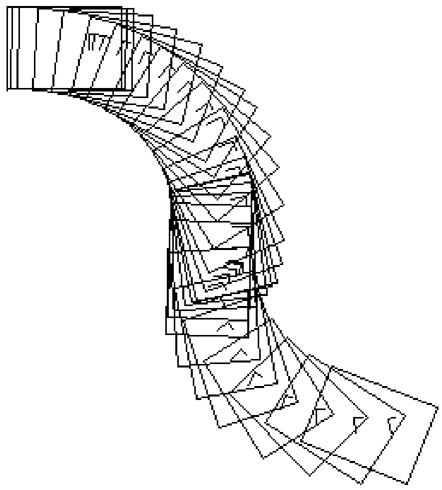


Figure 5.15 : Trajectoire du robot durant un suivi de cible

tir de ce travail, la poursuite de l'expérimentation consistera à élaborer d'autres Tâches-robot d'asservissement visuel et à valider leur enchaînement sur une application simple telle que le tour d'une pièce vide.

# Conclusion Générale

Le travail présenté dans ce document portait sur la mise en œuvre de techniques d'asservissement visuel en robotique mobile. Il devait répondre à plusieurs questions :

- Etait-il possible de commander par asservissement visuel des robots mobiles dont la structure mécanique comporte des liaisons non-holonomes ?
- Placé dans un environnement d'intérieur structuré, quelles types de tâches référencées vision un robot mobile peut-il effectuer ?
- Comment enchaîner ces tâches pour réaliser une application ?
- Les réponses apportées devaient être validées expérimentalement, ce qui a entraîné la question : comment concevoir et gérer un système robotique expérimental destiné à tester notre approche ?

Nous avons proposé une modélisation cinématique "générale" permettant de calculer une jacobienne d'un robot mobile muni d'un bras manipulateur qui porte une caméra. Cette modélisation de l'ensemble permet la coopération des mouvements entre la base mobile et son bras. Le fait de commander le déplacement de la caméra par l'ensemble mécanique contourne le problème de la commande des robots non-holonomes.

Nous avons ensuite catalogué et illustré les différentes tâches référencées vision qu'un robot mobile pourrait exécuter en utilisant des motifs visuels issus d'une scène d'intérieur. L'utilisation de ces tâches devrait permettre au robot d'explorer son environnement. Pour ce type d'application, il est alors nécessaire d'enchaîner plusieurs tâches référencées vision en utilisant différents motifs dans l'image. Nous avons proposé un cadre formel pour combiner les tâches pour avoir un enchaînement "le plus continu possible" entre deux motifs.

Nous avons considéré que l'expérimentation est une étape indispensable pour valider notre approche. Nous avons conçu et réalisé un robot mobile en ayant comme soucis de maîtriser totalement la chaîne de commande qui va de la consigne numérique aux courants dans les moteurs (la plupart des robots mobiles sont plutôt considérés comme des boites noires où l'utilisateur accède par des "goto x,y,$\theta$"). La réalisation et l'utilisation d'un système complexe nécessite l'accès à tous ses sous-ensembles. La programmation du système a conduit à développer un nouveau outil de spécifications (MINISIM) et à l'adaptation du système ORCCAD à la robotique mobile.

Nous avons mené notre travail des spécifications à l'expérimentation dans le cas d'une tâche de positionnement par asservissement visuel d'un robot mobile.

Ce travail couvrant un large spectre de différents domaines a soulevé et laissé en suspens de nombreuses questions. Les problèmes "théoriques" à aborder pour une meilleure utilisation de la commande référencée capteur en robotique mobile sont nombreux :

- Il serait souhaitable de **contrôler effectivement l'orientation de la base durant une tâche référencée vision**. Deux approches peuvent être explorées soit en utilisant un bras manipulateur redondant permettant le contrôle de l'orientation dans une tâche secondaire, soit en combinant la commande référencée vision et la commande à retour d'état instationnaire.

- Il faut proposer une méthodologie pour **prendre en compte des événements durant un enchaînement de tâche**. Le cadre formel pour l'enchaînement de tâches ne tient compte que de l'aspect continu de la loi de commande. Au niveau de la Tâche-robot, il est indispensable de prendre en compte les événements que l'on associera à des fenêtres dans l'image pour prendre la décision de changer de tâche.

- **Une approche objet de modélisation pour un contrôleur de robots mobiles** doit être étudiée. Il serait intéressant d'approfondir l'état de l'art sur les récentes techniques de modélisation et de commande des robots mobiles pour l'intégrer dans une approche objet permettant de modéliser une décomposition pour ORCCAD.

- **la fusion de différentes informations capteurs au niveau directement de la matrice d'interaction** est un problème qui reste ouvert. Il serait profitable d'utiliser les différents capteurs extéroceptifs dont les robots mobiles sont souvent dotés (vision, ultra-son, laser).

- Il serait intéressant de **transposer notre approche à d'autres systèmes robotiques** tel que les robots sous-marins.

- Au regard de leur complémentarité, il serait intéressant **de faire coopérer la reconstruction visuelle 3D de l'environnement et les techniques**

**d'asservissement visuel**. On pourrait imaginer un système doté de moyens de reconstruction 3D (stéréoscopie, vision dynamique...) dont la fréquence de calcul est peu élevée et qui utiliserait pour le déplacement du robot des techniques d'asservissement visuel sur des primitives 3D reconstruites...

Parallèlement à l'étude de ces problèmes, il est indispensable de poursuivre l'amélioration du très jeune (un an) système robotique ANIS. Gérer la complexité d'un tel système nécessite essentiellement d'améliorer l'environnement logiciel original (simulation, génération de code, visualisations de données, outils de vérifications, modélisation objet pour les robots mobiles, générateurs de code...).

Ainsi nous avons proposé une approche pour appliquer les techniques d'asservissement visuel à la robotique mobile que nous avons validé expérimentalement. En outre, nous espérons avoir convaincu le lecteur qu'en robotique comme d'en tant d'autres domaines, la théorie et l'expérimentation sont indissociables.

# Annexe A

# Modélisation et commande des robots mobiles

## A.1 Les notations

| | **Les distances et longueur (en mètres)** |
|---|---|
| $r$ | rayon des roues motrices du robot |
| $d$ | distance axe des roues - base de la tourelle |
| $R$ | distance point milieu de l'axe - roue |
| $l_1$ | longueur du premier segment du bras |
| $l_2$ | longueur du deuxième segment du bras |
| $l_3$ | longueur du troisième segment du bras |
| $l_{1a}$ | décalage entre l'axe vertical et l'axe suivant du bras |

| | **Les points définis** |
|---|---|
| $O$ | point fixe de la scène |
| $M$ | point milieu de l'axe des roues |
| $N$ | point attaché à la base du bras |
| $E$ | point attaché à l'effecteur |
| $C$ | centre optique de la caméra |

| | **Les repères** |
|---|---|
| $R_O$ | repère de la Scène |
| $R_M$ | repère attaché au point M |
| $R_N$ | repère base du bras |
| $R_E$ | repère effecteur |
| $R_C$ | repère caméra |

| | **vitesses et torseur** |
|---|---|
| $\underline{T}_c^{(R_n)}$ | torseur cinématique au point C exprimé dans le repère $R_n$ |
| $\vec{V}_M$ | vitesse linéaire du point M exprimée dans $R_m$ |
| $\Omega_{R_m/R_o}$ | vitesse de rotation du repère $R_m$ par rapport au repère $R_o$ |

| | **Vitesse angulaire sur les axes (en $rad.s^{-1}$)** |
|---|---|
| $\begin{pmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{pmatrix} = \underline{\dot{q}}_{bras}$ | vitesse des 3 axes du bras |
| $\begin{pmatrix} \dot{q}_d \\ \dot{q}_g \end{pmatrix}$ | vitesse des roues droite et gauche |
| $\begin{pmatrix} \dot{q}_d \\ \dot{q}_g \\ \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{pmatrix} = \underline{\dot{q}}$ | vitesse sur les 5 axes du robot |

## A.2 Changement de repère et paramétrage pour le torseur cinématique

**changement de repère** La commande-référencée-vision fournit le torseur cinématique au point $C$ exprimé dans le repère caméra $R_c$. Pour exprimer ce torseur dans le repère $R_n$, on utilise une matrice $M_{cr}$ de changement de repères telle que :

$$T_C^{(R_N)} = M_{cr}.T_C^{(R_C)} \tag{94}$$

La matrice est composée de la matrice $R$ de rotation du modèle géométrique du bras afin que :

$$M_{cr} = \begin{pmatrix} R & 0 \\ 0 & R \end{pmatrix} \tag{95}$$

Pour notre application on obtient alors :

$$M_{cr} = \begin{pmatrix} -s_1 & -c_1s_{23} & c_1c_{23} & 0 & 0 & 0 \\ c_1 & -s_1s_{23} & s_1c_{23} & 0 & 0 & 0 \\ 0 & c_{23} & s_{23} & 0 & 0 & 0 \\ 0 & 0 & 0 & -s_1 & -c_1s_{23} & c_1c_{23} \\ 0 & 0 & 0 & c_1 & -s_1s_{23} & s_1c_{23} \\ 0 & 0 & 0 & 0 & c_{23} & s_{23} \end{pmatrix} \tag{96}$$

**Changement de paramétrage** Pour passer de $\underline{\dot{X}}_{rtl}$ à $T_C^{(R_N)}$, on dérive suivant les axes de roulis, tangage, lacet :

$$\underline{\dot{\theta}} = \begin{pmatrix} \dot{\theta}_x \\ \dot{\theta}_y \\ \dot{\theta}_z \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \dot{\Psi} + \begin{pmatrix} -sin\Psi \\ cos\Psi \\ 0 \end{pmatrix} \dot{\Theta} + \begin{pmatrix} cos\Psi cos\Theta \\ sin\Psi cos\Theta \\ -sin\Theta \end{pmatrix} \dot{\Phi} \tag{97}$$

Sous forme matricielle, l'équation s'écrit :

$$\begin{pmatrix} \dot{\theta}_x \\ \dot{\theta}_y \\ \dot{\theta}_z \end{pmatrix} = \begin{pmatrix} 0 & -sin\Psi & cos\Psi cos\Theta \\ 0 & cos\Psi & sin\Psi cos\Theta \\ 1 & 0 & -sin\Theta \end{pmatrix} \begin{pmatrix} \dot{\Psi} \\ \dot{\Theta} \\ \dot{\Phi} \end{pmatrix} \tag{98}$$

On rappelle le système d'équation (30) de changements de paramétrage :

$$\begin{cases} \Psi = q_1 + \frac{\Pi}{2} \\ \Theta = 0 \\ \Phi = \frac{\Pi}{2} - (q_2 + q_3) \end{cases} \tag{99}$$

En utilisant (30), le système inverse de (98) s'écrit alors :

$$\begin{pmatrix} \dot{\Psi} \\ \dot{\Theta} \\ \dot{\Phi} \end{pmatrix} == \begin{pmatrix} 0 & 0 & 1 \\ -c_1 & -s_1 & 0 \\ -s_1 & c_1 & 0 \end{pmatrix} \begin{pmatrix} \dot{\theta}_x \\ \dot{\theta}_y \\ \dot{\theta}_z \end{pmatrix} \tag{100}$$

En utilisant la contrainte mécanique $\Theta = 0$, on peut alors construire la matrice inverse de changement de paramétrage $M_{cp}^{-1}$ de dimension $5 \times 6$ pour avoir :

$$\underline{\dot{X}}_{rtl} = M_{cp}^{-1} T_C^{(R_N)} \tag{101}$$

avec

$$M_{cp}^{-1} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -s_1 & c_1 & 0 \end{pmatrix} \tag{102}$$

On obtient alors la matrice finale de passage telle que :

$$M_p = M_{cp}^{-1} M_{cr} = \begin{pmatrix} -s_1 & -c_1 s_{23} & c_1 c_{23} & 0 & 0 & 0 \\ c_1 & -s_1 s_{23} & s_1 c_{23} & 0 & 0 & 0 \\ 0 & c_{23} & s_{23} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & c_{23} & s_{23} \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \tag{103}$$

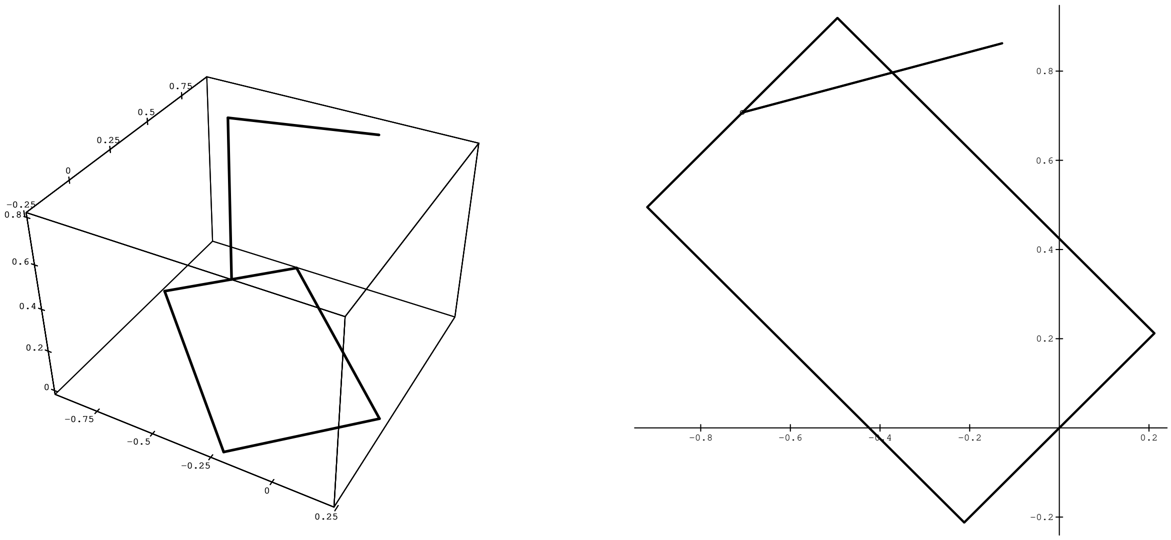
On peut alors en déduire le vecteur de commande à partir du torseur cinématique de la caméra exprimée dans son repère :

$$\underline{\dot{q}} = J_c^{-1} M_p T_C^{(R_C)} \tag{104}$$

## A.3 Résultats de Simulation

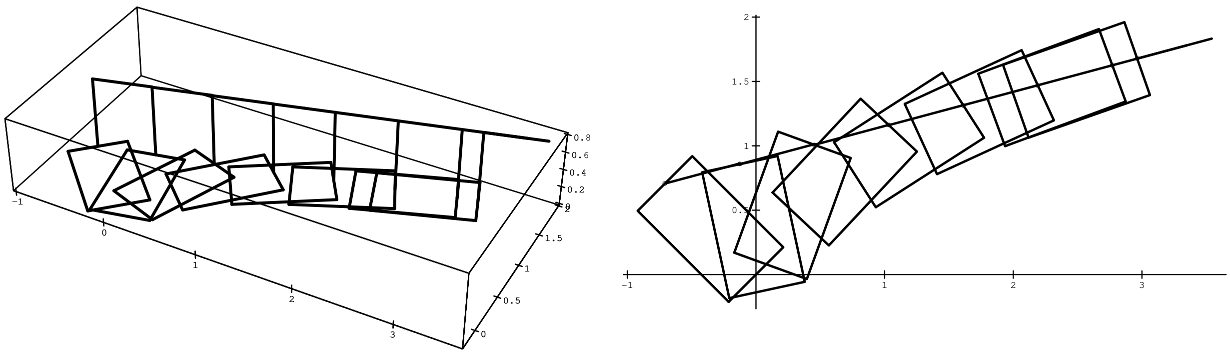
Les torseurs cinématiques de la caméra sont exprimés dans le repère caméra où $z$ est l'axe optique et $(x, y)$ le repère image. Le temps écoulé entre deux dessins est 600 ms .

**Schéma de la position et de l'orientation initiale du robot pour cette séquence de simulation.**

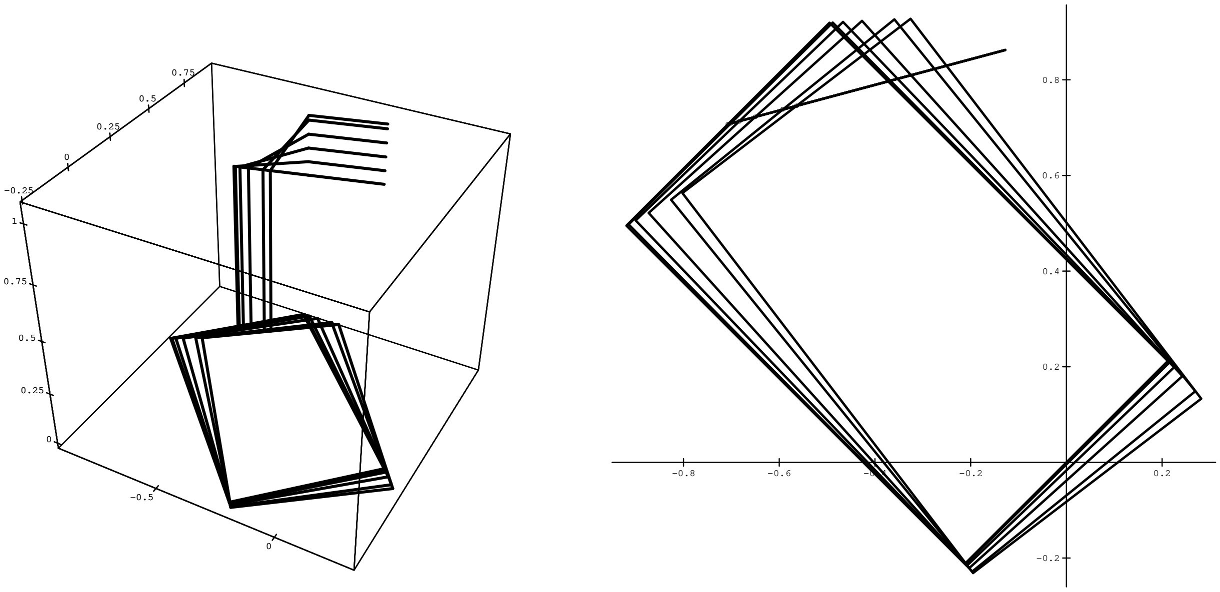


**Translation suivant l'axe optique de la caméra :**

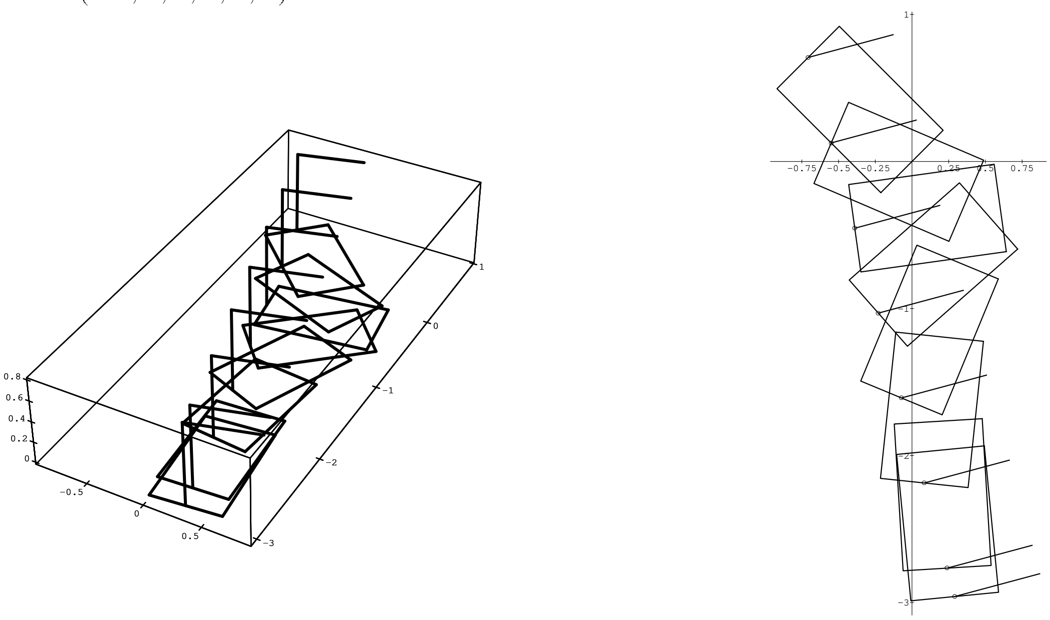
$Tc^{(R_c)} = (0, 0, 1, 0, 0, 0)^T$



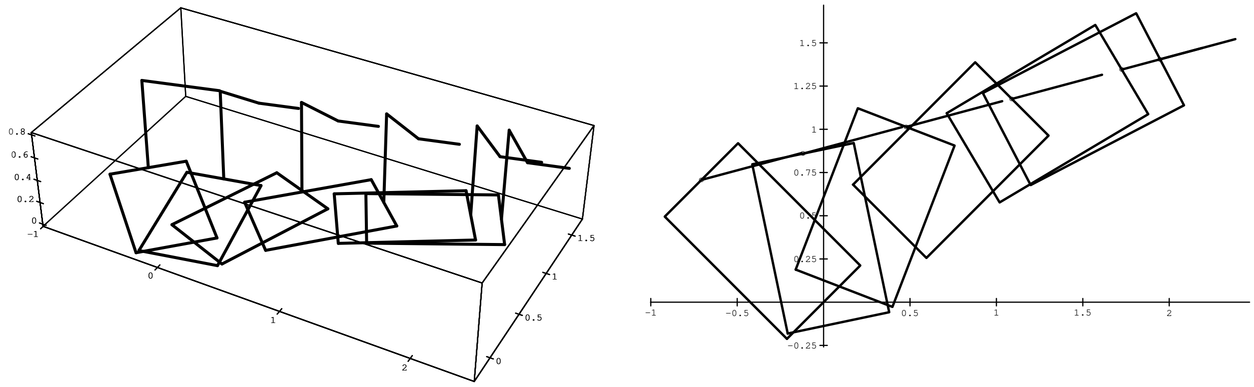
**Translation de la caméra le long d'un axe vertical :**
$Tc^{(R_c)} = (0, 0.1, 0, 0, 0, 0)^T$,



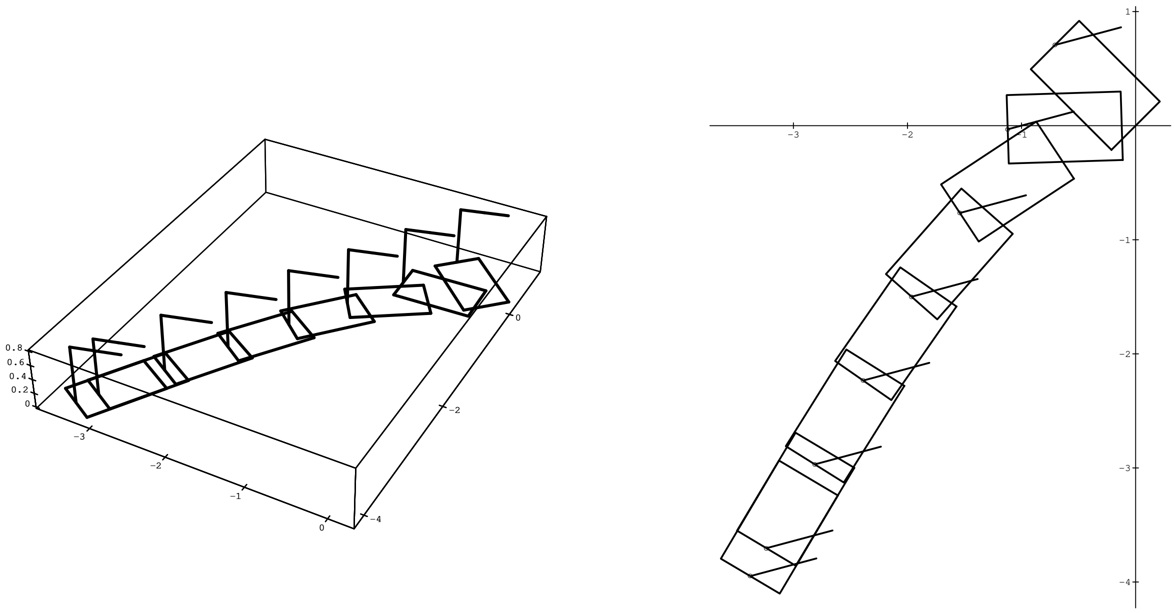
**Translation de la caméra le long d'un axe orthogonal à son axe optique. :**
$Tc^{(R_c)} = (-1, 0, 0, 0, 0, 0)^T$



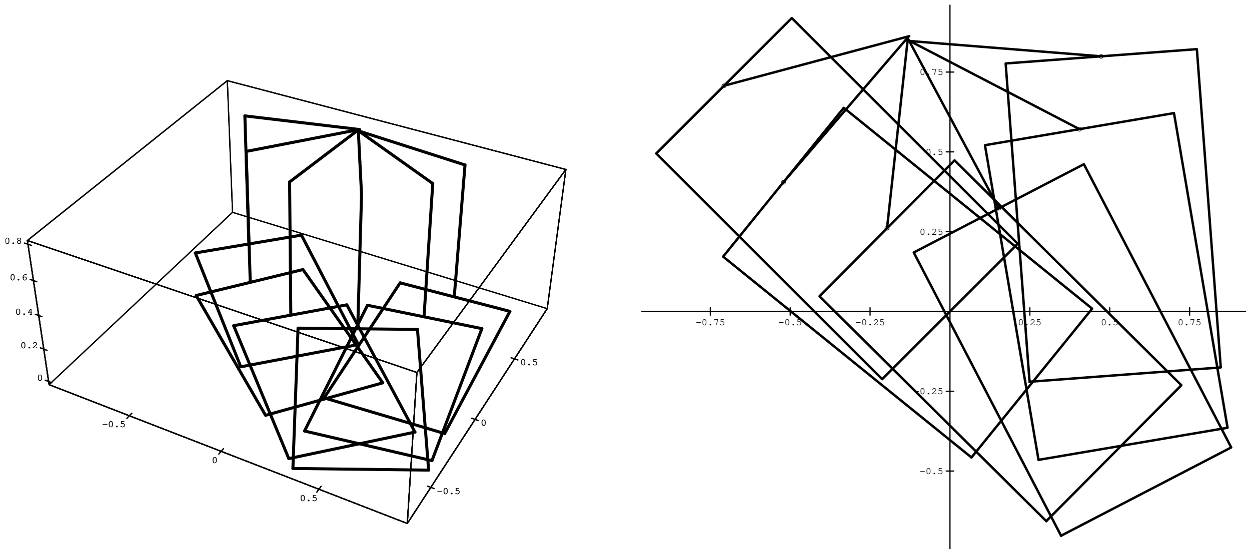
**Translation de la caméra dans le plan $(z, y)$ qui combine une translation suivant l'axe optique et une descente verticale :**
$Tc^{(R_c)} = (0, -0.1, 1, 0, 0, 0)^T$



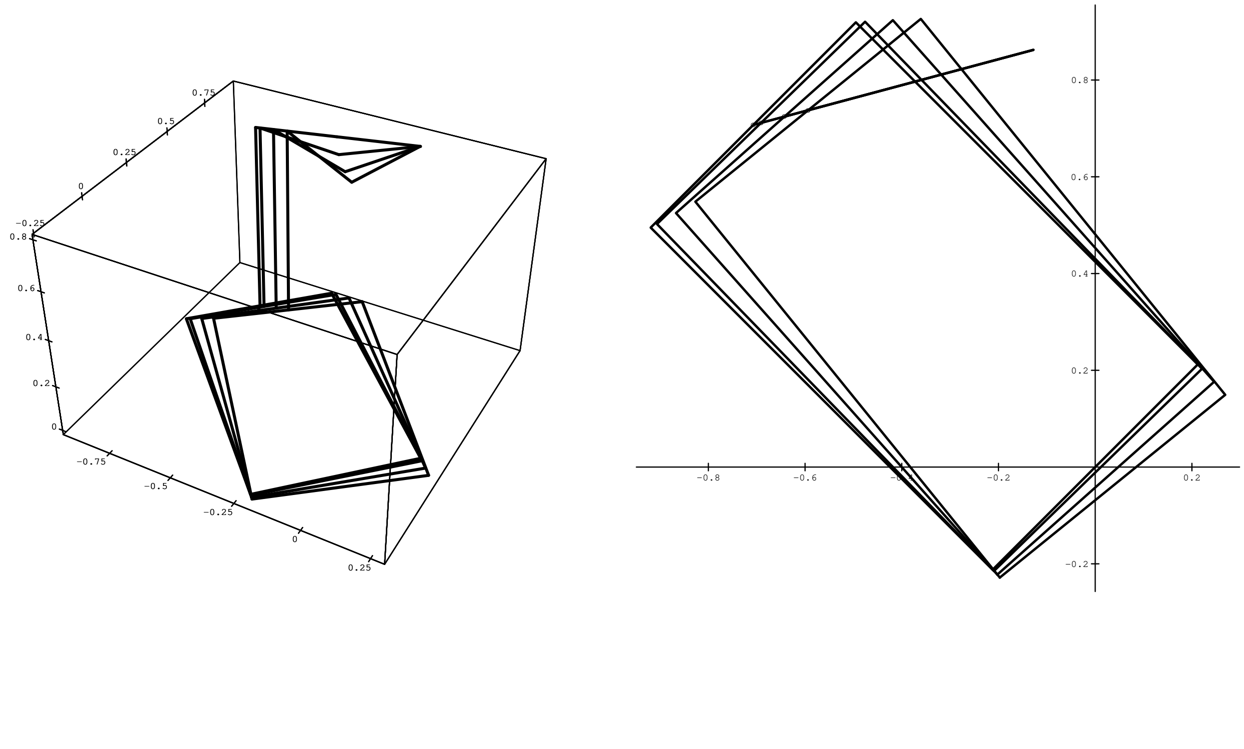
**Translation de la caméra dans le plan (x,z) :**
$Tc^{(R_c)} = (-1, 0, -1, 0, 0, 0)^T$



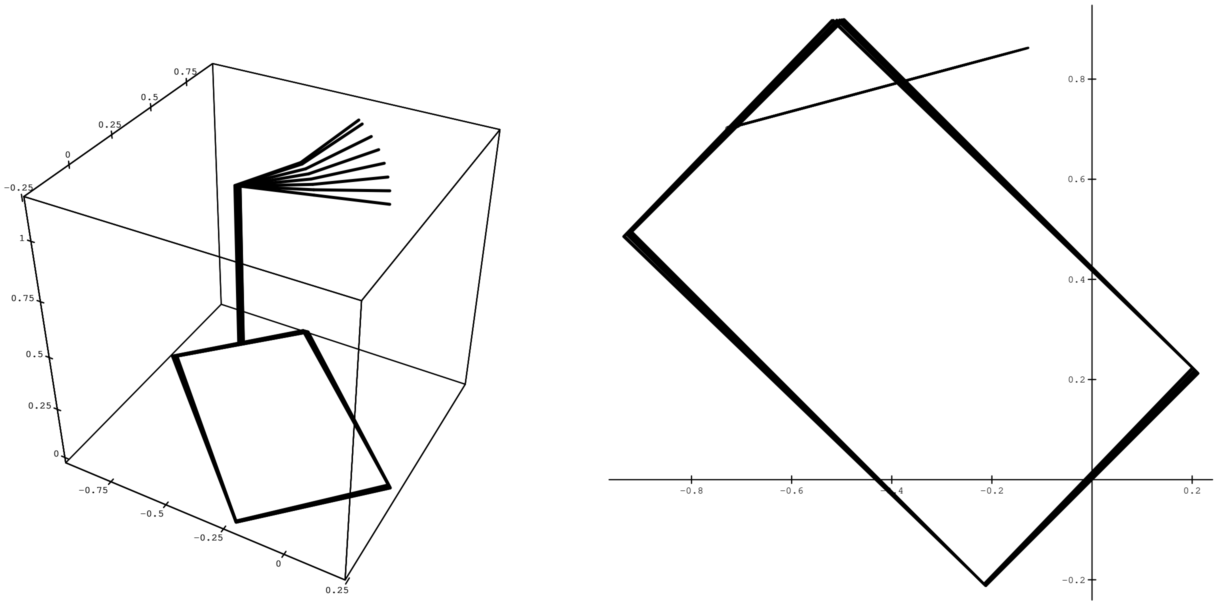
**Rotation de la caméra autour de son axe y :**
$Tc^{(R_c)} = (0, 0, 0, 0, 1, 0)^T$



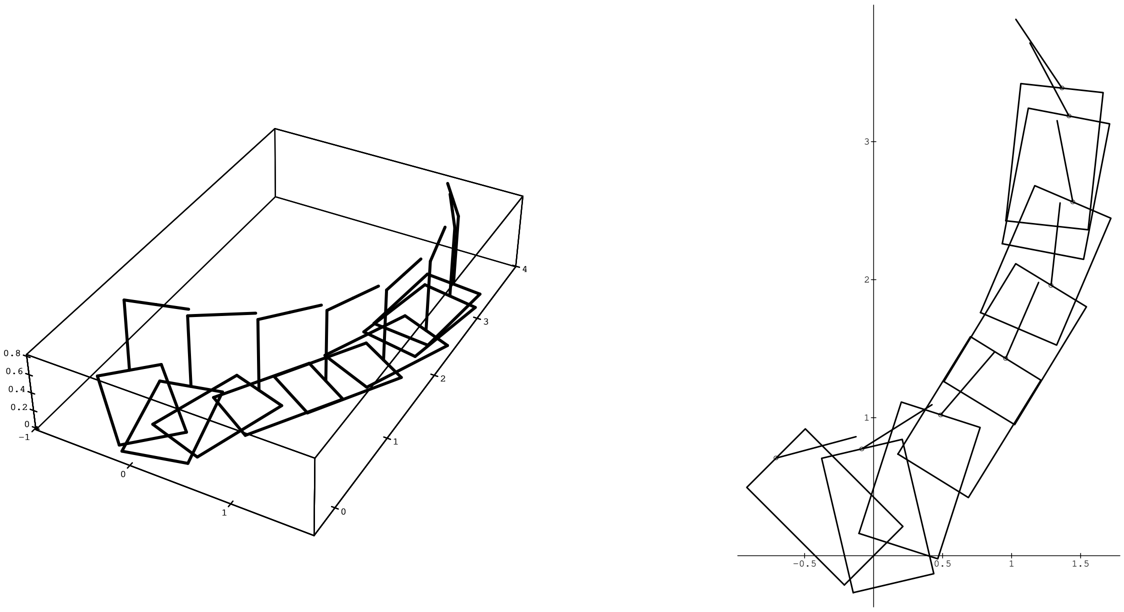

**Rotation de la caméra autour de son axe x :**
$Tc^{(R_c)} = (0, 0, 0, -1, 0, 0)^T$



**La trajectoire de la caméra est un cercle dans le plan (y,z) :**
$Tc^{(R_c)} = (0, 0.1, 0, -0.1, 0, 0)^T$



**La trajectoire de la caméra est un cercle dans le plan (y,z) avec l'axe optique dans la direction des tangentes au cercle.**
$Tc^{(R_c)} = (0, 0.1, 0, -0.1, 0, 0)^T$



# Annexe B

# Simulation de Tâches référencée vision

## B.1 Calculs pour une tâche référencée vision

**Calcul pour la rotation autour du segment**

**Motif à suivre dans le repère image :**

Segment de coefficients : $\{0.201134, 0\}$

**Expression du torseur vitesse Caméra :** Camera a 5 ddl sur bras

**Matrice d'interaction :**

$$\begin{pmatrix} -0.121674 & 0 & 0.00244727 & 0.0000489455 & -1.0004 & 0.00243348 \\ 0 & -0.121674 & 0.000296091 & 1.00001 & -0.0000489455 & -0.0201134 \\ -0.121674 & 0 & 0.00244727 & -0.000440509 & -1.0004 & -0.0219013 \\ 0 & -0.121674 & -0.00266481 & 1.00048 & 0.000440509 & -0.0201134 \end{pmatrix}$$

**Un vecteur appartenant au noyau de la matrice**

$$\begin{pmatrix} -0.992481 \\ 0 \\ 0.0199622 \\ 0 \\ 0.120759 \\ 0 \end{pmatrix} \in Ker(L^T)$$

**Conclusion:**
La tâche est de classe 1

Les mouvements laissés libres sont : Trajectoire circulaire:
- axe de rotation = $(0\ 1\ 0)^T$
- centre du cercle dans le plan $(x, z) = (-0.165306\ \ -8.21869)^T$
- rayon du cercle = 8.22035

On choisit comme torseur = $\{-1., 0., 0.0201134, 0., 0.121674, 0.\}$

## B.2 Illustrations de simulation

Les résultats graphiques qui suivent sont issus de l'utilisation du simulateur MINISIM. C'est l'utilisateur du logiciel qui choisit le type de tâche élémentaire à effectuer en sélectionnant le motif image à suivre et la tâche secondaire. L'arrêt de la tâche élémentaire est effectué sur une durée limite ou la disparition du motif dans l'image. L'utilisateur peut ensuite sélectionner un autre motif image à suivre. Nous proposons trois exemples de simulation.

#### B.2.0.2 Le déplacement dans un couloir (cf. figure B.1)

Ce déplacement est effectué par une succession de suivi de murs constituants le couloir. Le premier suivi de mur utilise la plinthe du mur (visualisé en Vue1) comme motif de référence. A la disparition de ce motif (cf. Vue2), la tâche est arrêtée. Le motif de référence sélectionné par l'utilisateur est alors la plinthe du mur suivant (cf. Vue3). A la fin de la tâche, la plinthe suivante n'étant pas visible (cf. Vue4), l'utilisateur choisit l'arête verticale du mur (cf. Vue4) pour faire effectuer une rotation de la caméra autour du motif. Une fois que la plinthe suivante apparaît dans l'image, le dernier suivi de mur est effectué.

#### B.2.0.3 Le déplacement en utilisant le carrelage (cf. figure B.2)

A partir des vues du sol (cf. "Vue du carrelage") de l'environnement 3D défini sur le figure B.2, l'utilisateur extrait deux droites de l'image pour guider visuellement le robot en translation suivant l'axe optique $\vec{z}$ de la caméra (première et troisième tâches élémentaires) ou suivant son axe $\vec{x}$ (deuxième tâche élémentaire). Pour la simulation, l'arrêt d'une tâche est effectué en fixant une distance limite de déplacement en translation.

#### B.2.0.4 Le déplacement en se dirigeant vers des coins (cf. figure B.3)

En utilisant un motif constitué par un coin extrait de l'image, il est possible de diriger le robot en direction de ce motif. Pour le premier déplacement, la caméra se dirige vers un coin d'une table (cf. Vue1 et Vue2). Un coin formé par l'ouverture de la porte de l'environnement 3D est ensuite utilisé (cf. Vue3 et Vue4) pour assurer le déplacement du robot.

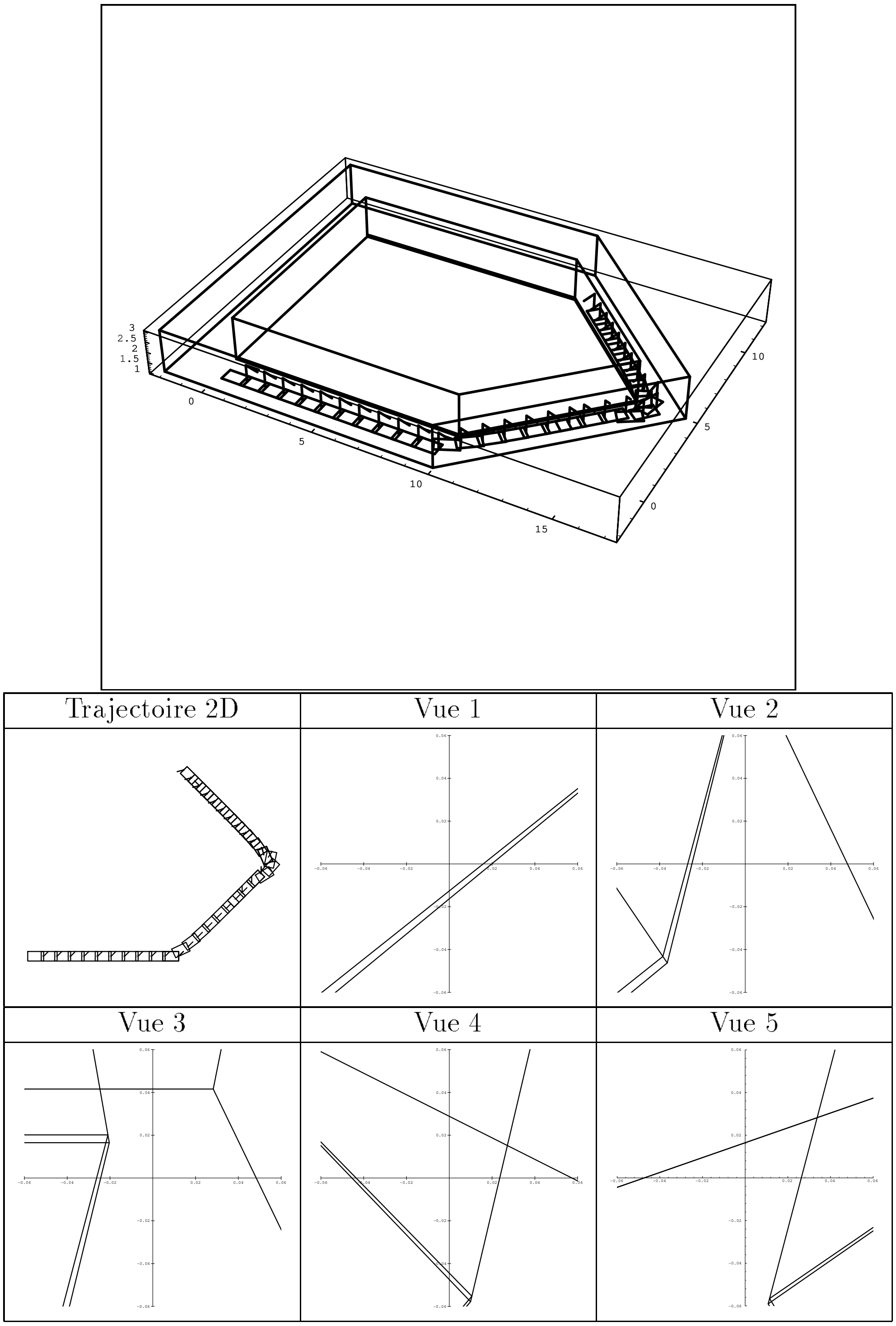



Figure B.1 : Déplacement dans un couloir

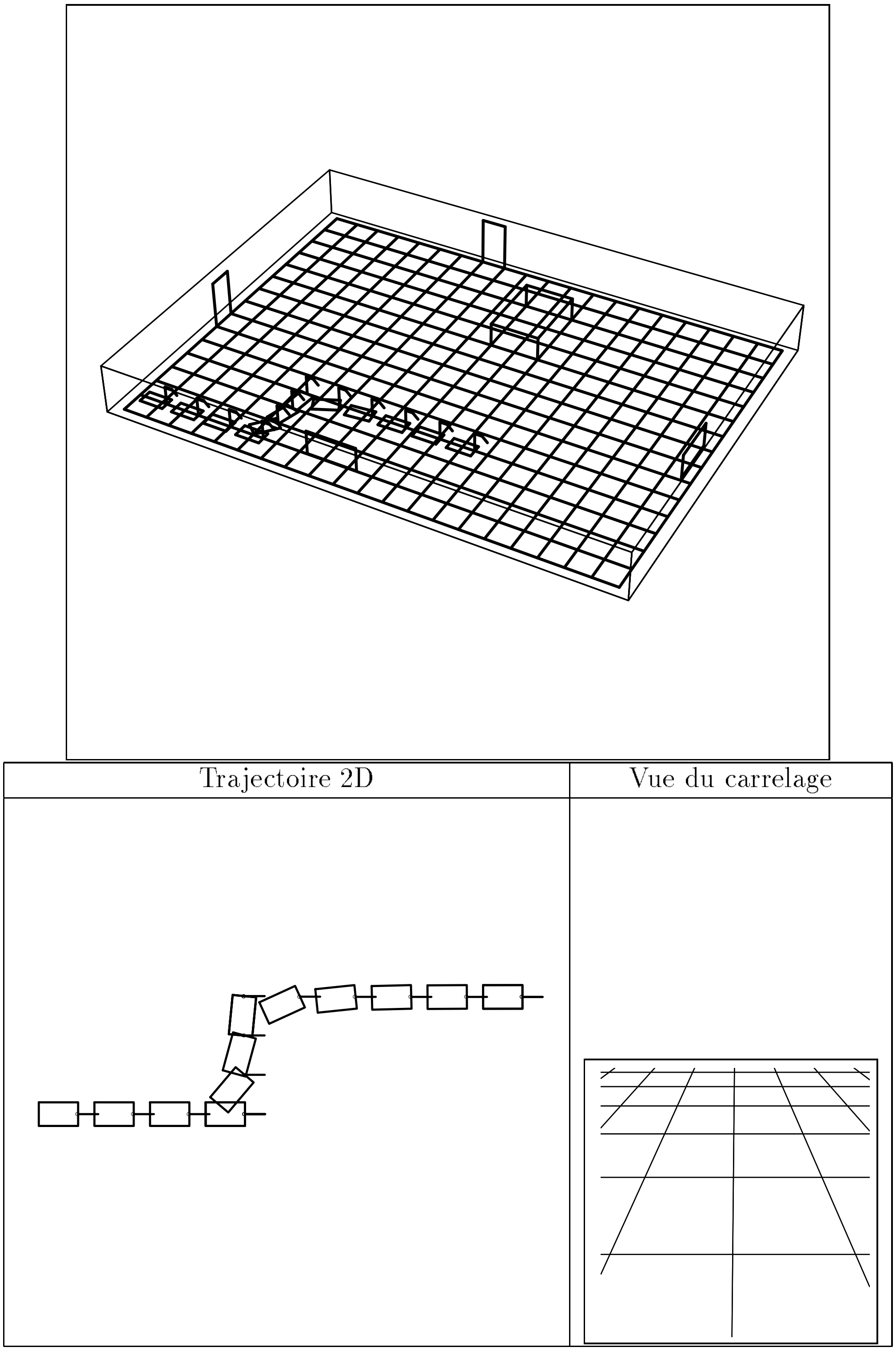



Figure B.2 : Déplacement en utilisant le carrelage

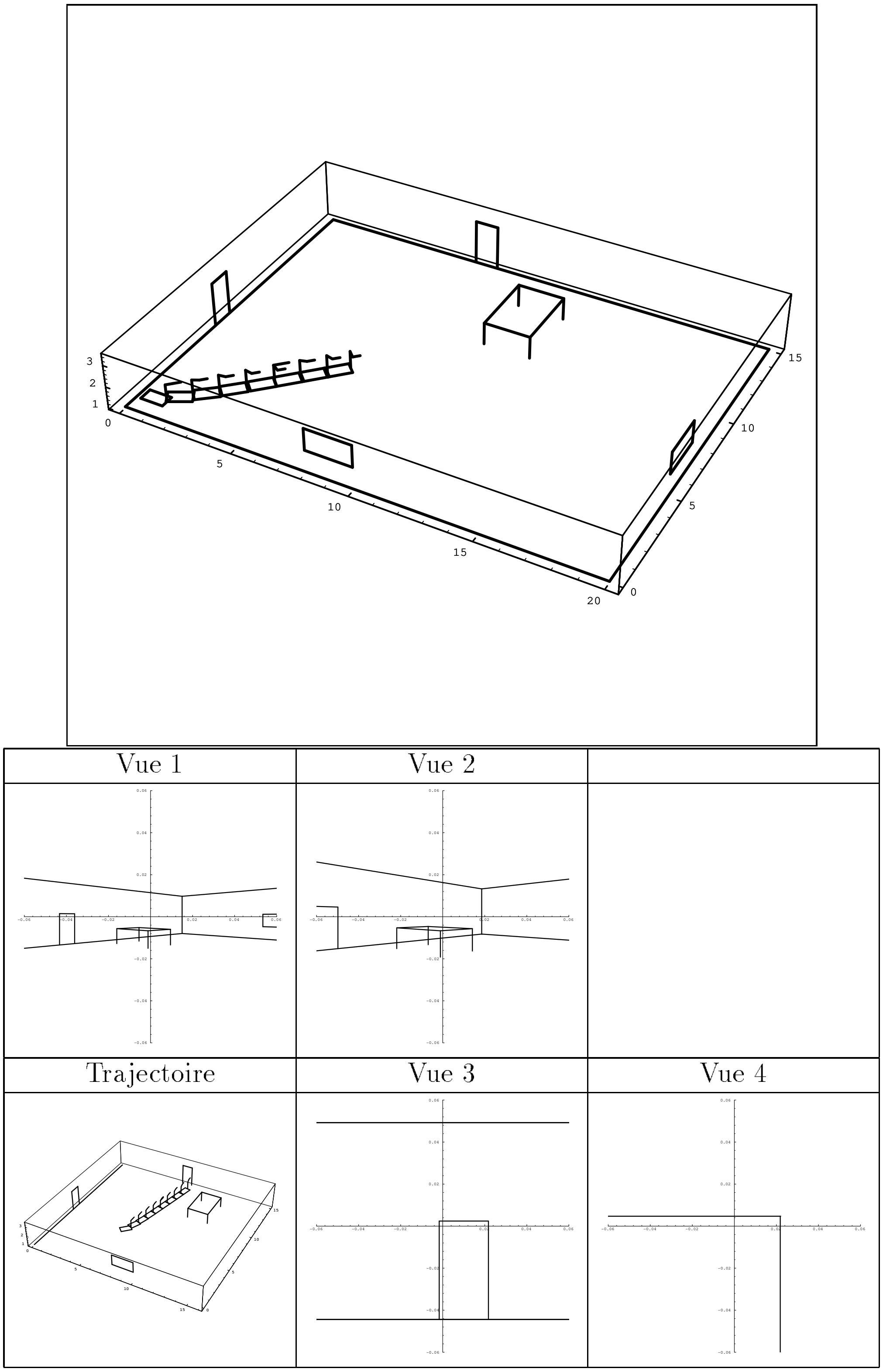



Figure B.3 : Déplacement en se dirigeant vers des “coins”

# Annexe C

# Enchaînements de Tâches référencées vision

## C.1 $L_2 = L_1 C$

Soit $\{e_1 \ldots e_m\}$ une base de $S$. Notons $L_1^k$ et $L_2^k$ les $k$ièmes colonnes de, respectivement, $L_1$ et $L_2$. Nous pouvons écrire:

$$L_1^k = \sum_{j=1}^{k} \lambda_{kj}^1 e_j \quad k = 1...p_1 \tag{105}$$

$$L_2^k = \sum_{j=1}^{k} \lambda_{kj}^2 e_j \quad k = 1...p_2 \tag{106}$$

Soit $E$ la matrice $6 \times m$: $\{e_1 \ldots e_m\}$, et $\Lambda_i$ les matrices $m \times p_i$ de rang $m$:

$$\Lambda_i = \begin{pmatrix} \lambda_{11}^i & \lambda_{1p_i}^i \\ \lambda_{m1}^i & \lambda_{mp_i}^i \end{pmatrix} \tag{107}$$

avec $i = 1, 2$. D'où: $L_i = E\Lambda_i$.

On cherche s'il existe une matrice $C$ telle que $L_2 = L_1 C$, donc telle que $E\Lambda_2 = E\Lambda_1 C$.

Cela est en particulier vrai si $\Lambda_1 C = \Lambda_2$. Soit donc:

$$C = \Lambda_1^{\dagger} \Lambda_2 = \Lambda_1^T (\Lambda_1 \Lambda_1^T)^{-1} \Lambda_2 \tag{108}$$

Comme $\dim(\Lambda_i) = m \times p_i$, et $\mathrm{rang}(\Lambda_i) = m$, on a bien trouvé une matrice $p_1 \times p_2$ de rang $m$ qui résoud le problème. Notons qu'on montrerait bien sûr de façon identique $L_1 = L_2 C'$.

## C.2 $S(L_1) = S(L_2)$, $p1 \neq p2$

On ne peut plus faire ici la somme pondérée de $s_1$ et $s_2$, leurs dimensions étant différentes. Une possibilité de mélange est donnée par:

$$\tilde{e}^1 = \lambda(u)(D_1 s_1 - \sigma_1^r(t)) + (1 - \lambda(u))(D_2 s_2 - \sigma_2^r(t)) \tag{109}$$

avec $D_i = W L_i$ et $\sigma_i^r(t) = D_i s_i^r(t)$ avec $(i = (1, 2))$, $W$ base de $S(L_1) = S(L_2)$. D'où:

$$\tilde{e}^1 = W\left(\lambda(u) L_1 s_1 + (1 - \lambda(u)) L_2 s_2\right) + \lambda(u)\sigma_1^r(t) - (1 - \lambda(u))\sigma_2^r(t) \tag{110}$$

## C.3 $S(L_1) \neq S(L_2)$

### C.3.1 $m_1 = m_2$

Un mélange de signaux reste possible sous la forme:

$$\tilde{s} = \lambda(u)D_1(s_1 - s_1^r(t)) + (1 - \lambda(u))D_2(s_2 - s_2^r(t)) \tag{111}$$

où $D_i$ est une matrice $m \times p_i$ de rang $m$, $i = 1, 2$. A noter, puisque $S(L_1)$ et $S(L_2)$ sont différents, qu'il ne faut pas s'attendre à ce que les deux tâches puissent être satisfaites simultanément, même en s'arrangeant pour que $\tilde{s}(t_0 - \frac{\tau}{2}) = 0$, c'est-à-dire en choisissant $s_2^r(t_0 - \frac{\tau}{2}) = s_2(t_0 - \frac{\tau}{2})$.

On peut prendre comme d'habitude $D_i = WL_i$. Le jacobien

$$\tilde{J}_1 = \frac{\partial \tilde{s}}{\partial \bar{r}} = \lambda(u)WL_1L_1^T + (1 - \lambda(u))WL_2L_2^T \tag{112}$$

n'exhibe pas à première vue de propriétés particulières.

### C.3.2 $m_1 \neq m_2$

Une dernière possibilité, d'ailleurs d'usage plus général, consiste à passer par une phase intermédiaire d'arrêt à la position atteinte au moment du désir de changement de tâche. Les deux tâches référencées capteur ne se recouvrent pas et le passage peut être un peu plus régulier:

**Phase 1**

Soit:

$$\tilde{e} = \lambda(u)e^1 + (1 - \lambda(u))\epsilon^* \tag{113}$$

où (par exemple) $u = t - t_0 + \tau$, et:

$$\epsilon^* = \begin{pmatrix} x - x^* \\ O_e(r, r^*) \end{pmatrix}, \tag{114}$$

$x^*, O(., r^*)$ étant une paramétrisation de la situation atteinte à $t = t_0 - \tau$, correspondant à $q^* = q(t_0 - \tau)$, mesurée. D'où:

$$\frac{\partial \tilde{e}}{\partial q} = \frac{\partial \tilde{e}}{\partial \bar{r}}\frac{\partial \bar{r}}{\partial q} = (\lambda \frac{\partial e^1}{\partial \bar{r}} + (1 - \lambda)\left(\begin{array}{c|c} I_3 & 0 \\ \hline 0 & W \end{array}\right))J \tag{115}$$

où $J$ est le jacobien du robot. Pour les raisons exposées plus haut, le terme $\frac{\partial e^1}{\partial \bar{r}}$ peut souvent être rendu positif, mais seulement le long de sa trajectoire idéale. Toutefois, $q^* = q(t_0 - \tau)$ se trouve en principe sur celle-ci si l'erreur initiale $e(t_0 - \tau)$ est faible. Le terme $\left(\begin{array}{c|c} I_3 & 0 \\ \hline 0 & W \end{array}\right)$ est quant à lui positif si la paramétrisation de l'attitude est bien choisie. La situation est donc meilleure que dans le cas précédent.

**Remarque**

Lorsque la tâche secondaire est un suivi de trajectoire, une autre façon de procéder consiste à conserver $\tilde{e} = e^1$, mais en remplaçant de façon douce la trajectoire à poursuivre par $q^*$ ou $(x^*, r^*)$.

**Phase 2**

Une fois l'équilibre atteint, on active la deuxième fonction de tâche:

$$\tilde{e}' = \lambda(t - t_0)e^* + (1 - \lambda(t - t_0))e^2, \tag{116}$$

et l'on utilise une gestion des conditions initiales de $e^2$ du type de celle exposée en section (3.3.1), car il n'y a bien sûr aucune raison pour que $e^2(t_0) = 0$.

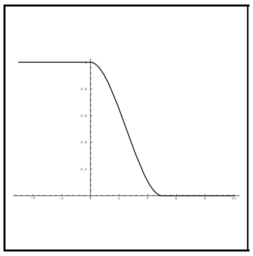


Figure C.1 : Fonction $\lambda$ pour $\tau = 5s$

## C.4 Choix d'une fonction $\lambda$

Pour générer un trajectoire de consignes ou pour mixer deux tâches, nous devons choisir une fonction scalaire $C^2$ $\lambda(t)$ décroissante sur $[0, \tau]$ telle que $\lambda(t0) = 1$ et $\lambda(\tau) = 1$. Nous avons choisi une fonction polynomiale peu coûteuse à implémenter :

$$\lambda(t) = \begin{cases} 1 & t < 0 \\ a_3t^3 + a_2t^2 + a_1t + a0 & 0 \leq \tau \\ 0 & t > \tau \end{cases}$$

Le calcul des coefficients du polynôme $p(t) = a_3t^3 + a_2t^2 + a_1t + a0$ utilise les contraintes suivantes qui assurent la continuité :

$$\begin{cases} p(0) = 1 \\ p(\tau) = 0 \\ p'(0) = 0 \\ p'(\tau) = 0 \end{cases}$$

Le polynôme dérivé a pour expression: $p'(t) = 3a_3t^2 + 2a_2t + a_1$. En résolvant le système on obtient trivialement :

$$\begin{cases} a_0 = 1 \\ a_1 = 0 \\ a_2 = -\frac{3}{\tau} \\ a_3 = \frac{2}{\tau^3} \end{cases}$$

Un tracé de la fonction est illustré pour un $\tau = 5s$. C'est ce paramètre qui permet de contrôler la durée de la transition.

# Annexe D

# Outils de conception pour les contrôleurs en robotique

## D.1 Glossaire

Afin de lever des ambiguïtés de vocabulaire, très fréquentes en architecture robotique, nous allons rappeler un ensemble de définitions.

- *système robotisé :* ensemble de ressources (robots, capteurs ...) coopérants pour la réalisation d'une application.
- *application :* séquence d'opérations exécutées par un système robotisé en vue de réaliser un objectif donné.
- *contrôleur :* ensemble de ressources informatiques (matérielles et logicielles) permettant le contrôle en ligne du système robotisé.
- *module :* ensemble de cartes de calcul et d'interfaces communicant par un bus.
- *Tâche-module :* tâche informatique, au sens classique temps réel, résidente dans un module. Ce terme sera noté par la suite TM.
- *Tâche-robot :* Ensemble de Tâches-modules interconnectées qui réalise un asservissement élémentaire auquel on associe un comportement logique pour prendre en compte un ensemble de signaux susceptibles d'arriver durant la tâche.
- *générateur de trajectoire :* émission <u>en ligne</u> de consignes vers la commande d'un robot.

## D.2 Description Technique de MINISIM

### D.2.1 Les fonctionnalités du logiciel

Les fonctions fournies à l'utilisateur par ce simulateur vont de la spécification d'une tâche d'asservissement visuel à sa simulation et son illustration graphique. Les fonctions que comportent ce logiciel sont nombreuses :

- la description d'un environnement par fichier,
- le choix du motif visuel dans un environnement,
- le calcul formel ou numérique de la tâche ($L^T$, $Ker(L^T)$),

- la simulation cinématique du robot mobile et de son bras dans l'environnement,
- la simulation de la commande référencée vision,
  - cette commande peut être simple, enchaînée ou mixée,
  - elle est décrite dans un fichier,
  - l'utilisateur dispose d'aide au calcul de ce fichier,
- la simulation de la caméra,
- la visualisation des résultats :
  - calculs dans un fichier latex,
  - graphique statique (scène 3D, scène "vue de dessus", vue caméra),
  - graphique animé sous X (vue caméra et scène "vue de dessus").

Pour utiliser l'ensemble de ces fonctions de manière correcte, le logiciel fournit plusieurs types d'interfaces : un utilisateur averti peut faire appel directement aux fonctions *Mathematica* (interface niveau 0) tandis que le lancement de l'exécution d'un script de simulation peut être lancé par un "novice". Ainsi, il existe quatre niveaux d'interfaces qui sont :

- Niveau 0 : appel aux fonctions *Mathematica*, écriture de nouvelles fonctions.
- Niveau 1 : utilisation du "menu texte" sous *Mathematica* (cf. figure (D.3)),
- Niveau 2 : utilisation du "menu graphique" OSF-Motif, génération de scripts (cf. figure (D.4)),
- Niveau 3 : modification de scripts (paramètres) à l'aide d'un éditeur de texte et simulation (cf. figure (D.2)),
- Niveau 4 : simulation de scripts sans modification, visualisation d'animation.

Le logiciel a été développé en utilisant le prototypage. Ainsi MINISIM comporte tout les avantages d'une maquette :

- souplesse d'utilisation dûe au langage *Mathematica* interprété,
- implémentation modulaire (cf. figure (D.1) sur l'architecture du logiciel) permettant l'ajout de fonctions facilement,
- développement aisé.

Par contre il en assume tout les inconvénients tels que :

- efficacité et lisibilité du code réduite,
- logiciel peu transportable (*lié à Mathematica*).

Cette maquette a été développée et utilisée en parallèle de l'étude sur la modélisation du robot mobile muni d'un bras et sur la spécification de tâches d'asservissement visuel. Cet outil a été très efficace pour s'assurer de la validité des calculs et de les illustrer graphiquement. Tout les exemples illustrés en annexe A.3 et B sont issus de l'utilisation de cet outil.

### D.2.2 Environnement du logiciel

La maquette MINISIM est écrite en *Mathematica v 1.2* qui est un logiciel de calcul formel. Le répertoire *minisim* contient les différents modules du logiciel (**.m*) qui doivent être chargés et interprétés par *Mathematica*. D'autres sous-répertoires contiennent les informations utiles au déroulement du logiciel :

- *data* contient les fichiers :

  *torseur.dat* séquence des torseurs caméra tout les Te secondes,

  *com.dat* séquence de commandes $(v_d, v_g, v_1, v_2, v_3)$ à appliquer au robot toutes les Tm secondes,

  *etat.dat* séquence de l'état du robot tout les Te secondes $(x, y, \theta, q_1, q_2, q_3)$,

  *editeur.sh* script shell pour permettre a l'utilisateur de modifier la génération de la séquence torseur dans *buildtorseur.m*,

- *tex* contient les différents fichiers LaTeXcréés pour visualiser les calculs du logiciel (**.tex*) et des scripts shells pour compiler, imprimer (**.sh*),

- *commande* contient les fichiers de paramétrisation d'un asservissement visuel,

- *vue* pour les différentes vues simulées par MINISIM :

  *des3M.ps* et *des2M.ps* pour vue globale ("3D") et vue caméra du motif,

  *des3S.ps* et *des2S.ps* idem pour la scène,

  *des02.ps* et *des12.ps* vue de dessus de l'évolution du robot sur un schéma sans ou avec axes,

  *des03.ps* et *des13.ps* idem pour une vue "3D",

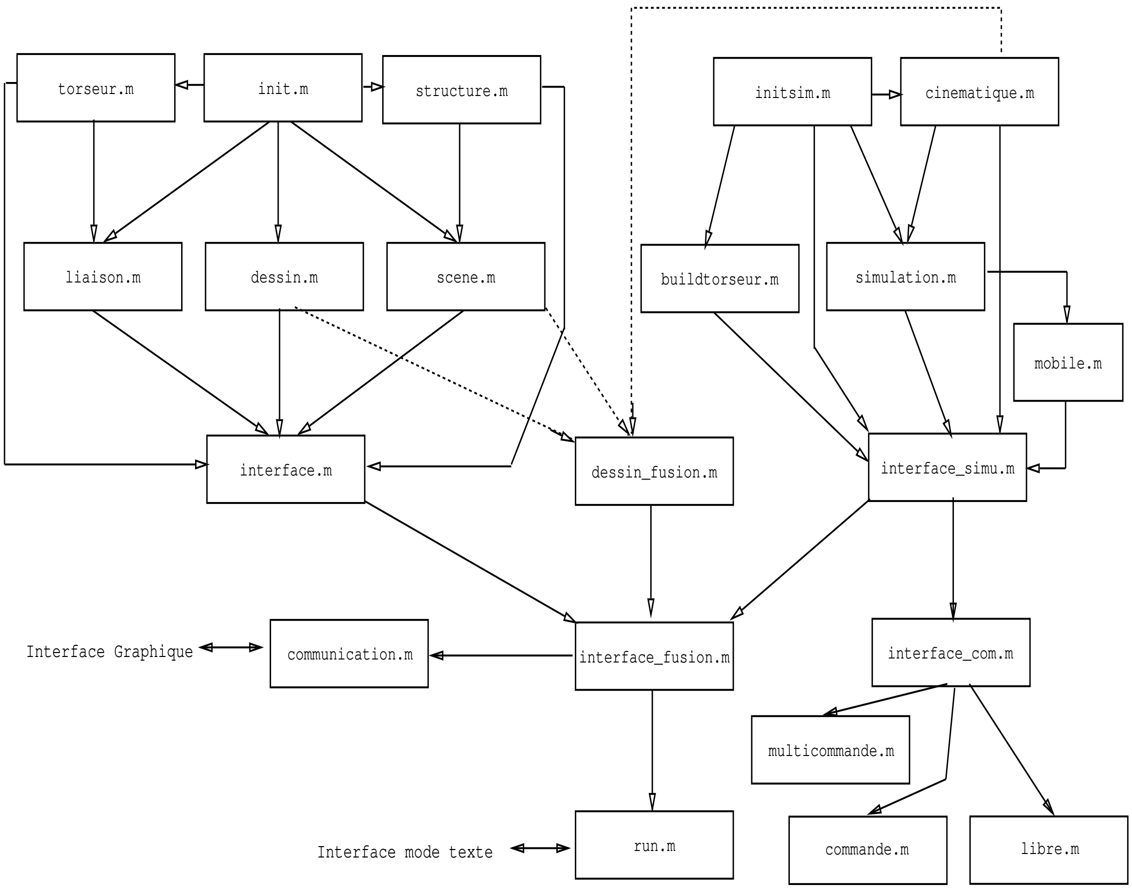
- *image* sauvegarde des sessions de MINISIM
- *scenario* sauvegarde des résultats de simulation (calculs latex, graphique postcript),
- *histo* sauvegarde des scripts de simulations,
- *doc* documentation de MINISIM.

### D.2.3 Architecture du logiciel

Le prototypage de MINISIM a été faite de manière modulaire afin d'assurer une souplesse d'écriture et d'utilisation. Le logiciel est donc composé de plusieurs modules dont les rôles sont donnés dans le tableau 1. L'architecture du logiciel est illustrée en figure (D.1).

| |
|---|
| init_simu.m<br>Initialise les variables globales concernant la simulation cinématique du robot |
| cinematique.m<br>Regroupe les fonctions de calcul cinématique pour le robot |
| simulateur.m<br>Calcule à partir d'une séquence de torseurs caméra l'état du robot |
| mobile.m<br>Interfaçage entre le robot mobile et le simulateur |
| buildtorseur.m<br>Fonction de création d'un fichier de séquence du torseur camera |
| interface_simu.m<br>Interface entre le simulateur du mobile et l'utilisateur |
| structure.m<br>Outils pour la construction de Scènes et Motifs |
| dessin.m<br>Fonctions de dessin de la Scène et du Motif en vue globale et vue caméra |
| torseur.m<br>Calcul de la matrice d'interaction, de son noyau, de la classe de la tâche |
| liaison.m<br>Analyse les liaisons virtuelles entre la caméra et l'environnement |
| interface.m<br>Interface du calculateur de liaisons virtuelles avec l'utilisateur |
| scene.m<br>Exemple de construction d'une scène |
| dessin_fusion.m<br>Dessin de déplacement du robot dans la scène |
| interface_fusion.m<br>Interface utilisateur regroupant les modules d'interface |
| run.m<br>Chargement et lancement de l'interface |

**Tableau 1: Rôles des modules** *Mathematica*



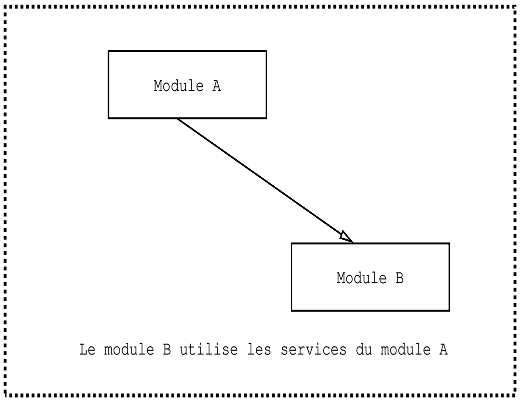




Figure D.1 : Les Modules de MINISIM

```
AllInit;AllInit;
Interactif=0;
Charge["pentagone.num"];
VisuFen=0;
FreNbVue=0;
FicIm=1;
DeplaceRobot[2,{Pi/4,-Pi/4,0}];
Extrait[8,0];
Extrait[16,0];
AddLI[1,0,0,0];
AddLI[2,0,0,0];
Dim=5;
Sig=0;
KernelScrewI[1];
Fre=1;
NbEch=42;
GoI;
(*Sauve["penta1"];*)
InitMotifI;
Extrait[11,0];
Extrait[17,0];
AddLI[1,0,0,0];
AddLI[2,0,0,0];
Dim=5;
KernelScrewI[1];
NbEch=18;
GoI;
...
```

Figure D.2 : Script pour le déplacement dans le pentagone

## D.3 Interfaces de Minisim

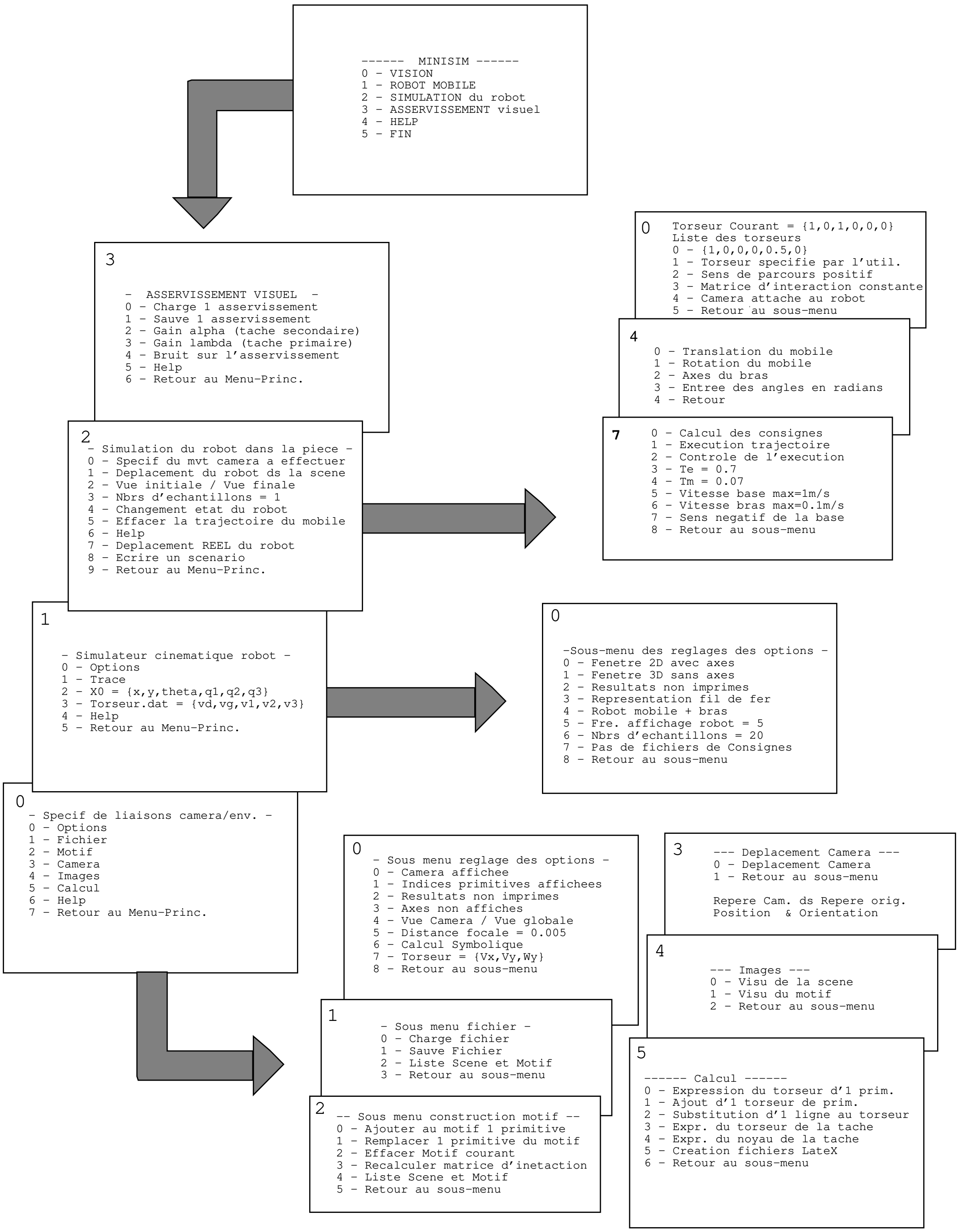



Figure D.3 : Interface Texte

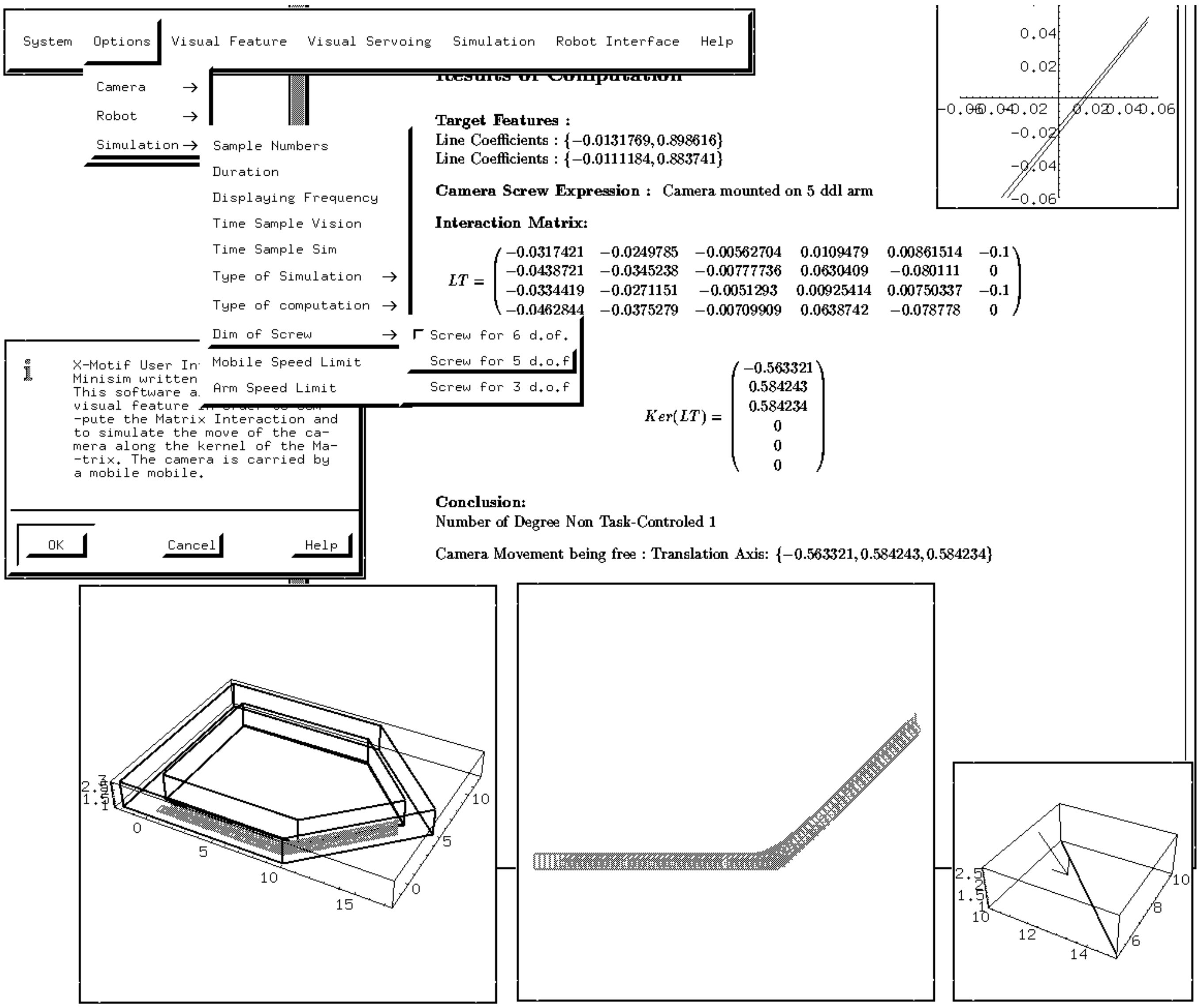



Figure D.4 : Interface Graphique

# Références bibliographiques


[Ait-Abderrahim 93] K. Ait-Abderrahim : *Commande des robots mobiles*, Thèse de l'école des Mines de Paris, Janvier 1993.

[Alami 93] R. Alami, R.Chatila, B. Espiau: *Designing an Intelligent Control Architecture for Autonomous Robots*, International Conference on Advanced Robotics, Tokyo, pp 435-440, November 1-2, 1993.

[Alami 90] R. Alami, R. Chatila, P. Freedman : *Task level teleprogramming for intervention robots*, IARP 90 "Mobile robots for subsea environments", Monterey california, 23-26 October 90.

[Albus 90] J. Albus, R.Quintero : *Toward a Reference Model Architecture for Real-Time Intelligent Control Systems (ARTICS)*, ASME press series, Jamshidi and Saif Editors, vol 3, 1990.

[Atiya 91] S. Atiya, G. Hager : *Real-time vision-based robot localisation*, Proceedings IEEE Int. Conf. on robotics and automation, Sacramento, pp 639-644, April 91.

[Andréa-Novel 91] B. d'Andréa-Novel, G. Bastin, G. Campion : *Modelling and control of non holonomic wheeled mobile robots*, Proceedings IEEE Int. Conf. on robotics and automation, Sacramento, pp 1130-1135, April 91.

[Agin 77] G. Agin : *Servoing with visual feedback*, Symp. on Industrial Robotics, pp 551-560, Tokyo, October 1977.

[Agin 79] G. Agin : *Real time control of a robot with a mobile camera*, Technical Note 179, SRI International, February 1979.

[Allen 91] P. Allen, B. Yoshimi, A. Timcenko : *Real-time visual servoing*, Proceedings IEEE Int. Conf. on robotics and automation, Sacramento, pp 851-856, April 91.

[Allen 92] P. Allen, A. Timcenko, B. Yoshimi, P. Michelman : em Trajectory filtering and predicting for automated tracking and grasping of a moving object, Proceedings IEEE Int. Conf. on robotics and automation, Nice, vol 3, pp 1850-1856, May 92.

[Archibald 91] C. Archibald, M. Van de Panne : *Tracking and grasping moving objects using reflex behaviour*, ICAR Fifth International Conference of Advanced Robotics, Pisa, pp 643-648, June 91.

[Ayache 88] N. Ayache : *Construction et fusion de représentations visuelle 3D - Application à la robotique mobile*, Thèse de Docteur es Sciences, Université de Paris Sud, Centre d'Orsay, Mai 1988.

[Astraudo 92] C. Astraudo, J.J. Borrelly : *Simulation of Multiprocessor Robot Controllers*, Proceedings IEEE Int. Conf. on robotics and automation, Nice, May 91.

[Bamberger 81] Y. Bamberger : *Mécanique de l'ingénieur 1 : Systèmes des corps rigides*, Hermann, 1981.

[Badreddin 91] E. Badreddin : *Tailoring the behaviour of a mobile robot - a recursive approach* -, Proceedings IEEE Int. Conf. on robotics and automation, Sacramento, pp 2556-2561, April 91.

[Barth 91] M. Barth, H. Ishiguro, S. Tsuji : *Computationally inexpensive egomotion determination for a mobile robot using an active camera*, Proceedings IEEE Int. Conf. on robotics and automation, Sacramento, pp 2792-2797, April 91.

[Benveniste 91] A. Benveniste, G. Berry : *The synchronous approach to Reactive and Real-Time Systems*, Proc. of the IEEE, vol 79, n 9, September 1991.

[Berry 92] G. Berry, G. Gonthier : *The synchronous Esterel Programming Language : Design, Semantics, Implementation*, Science of Computer Programming, Vol 19, no 2, pp 87-152, Nov. 1992.

[Borenstein 87] J. Borenstein, Y. Koren : *Motion control analysis of a mobile robot*, Journal of dynamic systems, vol 109, n2, pp 73-79, 1987.

[Borrelly 90] J. J. Borrelly, D. Simon : *Propositions d'architecture de contrôleur ouvert pour la robotique*, Rapport de recherche INRIA, n 1304, Octobre 1990.

[Boukir 93] S. Boukir, F. Chaumette, P. Bouthemy, D. Juvin : *An active robot vision system for real-time 3-D structure recovery* , First IFAC, Pergamon Press, University of Southampton Hampshire, pp 139-144 , 18-21 April 1993.

[Bouthemy 88] P. Bouthemy : *Modèles et méthodes pour l'analyse du mouvement dans une séquence d'images*, Technique et Sciences Informatiques, Vol 7, n 6, pp 527-546, 1988.

[Brooks 86] R. Brooks : *A robust layered control system for a mobile robot*, IEEE Journal of robotics and Automation, vol 2, n1, pp 14-33, 1986.

[Brooks 90] R. Brooks : *Elephants don't play chess*, Robotics and autonomous system, Vol 6, n1 , pp 3-15, June 90.

[Brooks 91] R. Brooks : *Intelligence without reason*, IJCAI 91, April 91.

[Buffa 93] M. Buffa : *Navigation d'un robot mobile à l'aide de la Stéréovision et de la triangulation de Delaunay*, Thèse de l'université de Nice, Décembre 1993.

[Burke 93] T. Burke, H. Durrant-Whyte : *Modular mobile robot design*, First IFAC, Pergamon Press, University of Southampton Hampshire, pp 295-300, 18-21 April 1993.

[Byrnes 93] R. Byrnes : *The Rational Behavior Model: a multi paradigm tri-level software architecture for the control of autonomous vehicles*, Phd dissertation, Naval Posgradate School, March 1993.

[Cameron 93] J. Cameron, D. Mac Kenzie, K. Ward, R. Arkin, W. Book : *Reactive control for mobile manipulator*, Proceedings IEEE Int. Conf. on robotics and automation, vol 3 ,pp 228-234, Atlanta, 2-6 May 1993.

[Campion 93] G. Campion, G. Bastin, B. d'Andrea-Novel : *Structural properties and classification of kinematic and dynamic models*, Proceedings IEEE Int. Conf. on robotics and automation, vol 1, pp 463-469, Atlanta, 2-6 May 1993.

[Canudas 91] C. Canudas de Wit, R. Roskam : *Path following of a 2-DOF wheeled mobile robot under path and input torque constraints*, Proceedings IEEE Int. Conf. on robotics and automation, Sacramento, pp 1142-1147, April 91.

[Chatila89] A. Chatila, F. Noreils : *Control of mobile robot actions*, IEEE International Conference on robotics and automation, vol2, pp 701-707, 1989.

[Chaumette 90] F. Chaumette : *La commande référencée vision: une approche aux problèmes d'asservissements visuels en robotique*, Thèse de l'université de Rennes, Juillet 1990.

[Chaumette 91a] F. Chaumette, P. Rives, B. Espiau : *The task function approach applied to vision-based control*, ICAR Fifth International Conference of Advanced Robotics, Pisa, pp 1392-1397, June 91.

[Chaumette 91b] F. Chaumette, P. Rives, B. Espiau : *positioning of a robot with respect to an object, tracking it and estimating its velocity by visual servoing*, Proceedings IEEE Int. Conf. on robotics and automation, Sacramento, pp 2248-2253, April 91.

[Connel 92] J. Connel : *SSS: a hybrid architecture applied to robot navigation*, Int. Conf. IEEE on Robotics and Automation, Vol 3, pp 2719-2724, Nice, May 1992.

[Corke 92] P. Corke, M. Good : *Dynamic effects in high performance visual servoing*, Int. Conf. IEEE on Robotics and Automation, Vol 3, pp 1838-1843, Nice, May 1992.

[Coste-Manière 93] E. Coste-Manière, P. Couvignou, P. Khosla : *Visual Servoing in the task-function framework: a contour following task*, to be published in "Journal of Intelligent and Robotics Systems".

[Coste-Manière 92] E. Coste-Manière, B. Espiau, D. Simon : *Reactive objects in a task level open controller*, Int. Conf. IEEE on Robotics and Automation, Vol 3, pp 2732-2737, Nice, May 1992.

[Coste-Manière 91] E. Coste-Manière : *Synchronisme et asynchronisme dans la programmation des systèmes robotiques: apport du langage* ESTEREL *et de concepts objets*, Thèse de l'Ecole des Mines de Paris, Juillet 1991.

[Couvignou 92] P. Couvignou, N. Papanikolopoulos, P. Khosla : *Hand-eye robotic visual servoing around moving objects using active deformable models*, International Conference on Intelligent Robot and Systems (IROS'92), Raleigh, July 7-10, 1992.

[Cox 91] I. Cox : *Blanche-An experiment in guidance and navigation of an autonomous robot vehicle*, IEEE transactions on robotics and automation, vol 7, n2, pp 193-203, April 91.

[Crowley 88] J. L. Crowley : *navigation for an intelligent mobile robot*, IEEE journal of robotics and automation, vol 1, n1, pp 31-41, 1988.

[Crowley 89] J. Crowley : *Asyncronous control of orientation and displacement in a robot vehicule*, IEEE International conference on robotics and automation, vol2, pp 1277-1282, 1989.

[Crowley 91] J. Crowley, P. bobet, K. Sarachik, S. Mely, M. Kurek : *Mobile robot perception using vertical line stereo*, Robotics and Autonomous Systems, Vol 7, n 2-3, pp 125-138, August 1991.

[Crowley 92] J. Crowley, P. Bobet, M. Mesrabi : *Gaze control for a binocular camera head*, ECCV-2, Genova, Italy, May 1992.

[Davis 91] L. Davis : *Visual navigation at the university of Maryland*, Robotics and Autonomous Systems, Vol 7, n 2-3, pp 85-98, August 1991.

[Dickmanns 88] E. D. Dickmanns, V. Graefe : *Applications of dynamic monocular machine vision*, Machine vision and Applications, Verlong, New-York Drc, 1988.

[Dickmanns 91] E. Dickmanns, Th. Christians : *Relative 3D-state estimation for autonomous visual guidance of road vehicles*, Robotics and Autonomous Systems, Vol 7, n 2-3, pp 113-123, August 1991.

[Dombre 88] E. Dombre, W. Khalil : *Modélisation et commande des robots*, traité des nouvelles technologies, série robotique, Hermes, 1988.

[Dunlay 88] R. T. Dunlay : *Obstacle avoidance perception processing for the autonomous land vehicle*, IEEE Int conf on robotics and automation, pp 912-917, 1988.

[Durrant-Whyte 88] H. F. Durrant-Whyte : *Sensor models and multisensor integration*, Int. Journal of Robotics Research, Vol. 7, n. 6, pp 97-113, December 1988.

[ECA 92] R. Eteki : *Spécifications techniques de la base mobile du robot version 2*, Rapport interne ECA, Document no 857.2.11.001, Mai 1992.

[ECA 93a] P. Gargani : *Dossier de fabrication du robot mobile RMV*, Rapport interne ECA, Document no 857.2.11.000, 1993.

[ECA 93b] R. Pissard-Gibollet : *Spécifications de tests pour le robot mobile RMV*, Document no 000.5.00.110, Janvier 1993.

[Ekhlund 93] J.L. Ekhlund, K. Pahlavan, T Uhlin : *Dynamic Fixation*, 4th Int.Conf. on Computer Vision, pp 412-419, 1993.

[Espiau 87] B. Espiau, P. Rives : *Estimation de primitives 3D au moyen d'une caméra mobile*, TSI, vol 4, n 3, pp 259-270, 1987.

[Espiau 90] B. Espiau, F. Chaumette, P. Rives : *Une nouvelle approche de la relation vision-commande en robotique*, Rapport de recherche INRIA, n1172, Avril 1990.

[Espiau 92] B. Espiau, F. Chaumette, P. Rives : *A new approach to visual servoing in robotics*, IEEE Transactions on robotics and automation, vol 8, n 3, pp 313-326, Juin 1992.

[Espiau 93] B. Espiau : *Effect of Camera Calibration Errors on Visual Servoing in Robotics*, Third International Symposium on Experimental Robotics, Kyoto, Japan, pp 187-193, Oct. 28-30, 1993.

[Faugeras 88] O. Faugeras : *Quelques pas vers la vision artificielle*, TSI, 1988.

[Feddema 89] J. Feddema, O. Mitchell : *Vision-guided servoing with feature-based trajectory generation*, IEEE transactions on robotics and automation, vol 5, n5, pp 691-700, oct 1989.

[Feddema 91] J. Feddema, G. Lee : *Weighted selection of image features for resolved rate visual feedback control*, IEEE transactions on robotics and automation, vol 7, n1, pp 31-47, Feb 1989.

[Feng 89] D. Feng, H. Friedman, B. Krogh : *The servo-control system for an omnidirectionnal mobile robot*, IEEE International conference on robotics and automation, vol3, pp 1566-1571, 1989.

[Ferrari 91] F. Ferrari, M. Fossa, E. Grosso, M. Grassi, G. Sandini : *A practical implementation of a multilevel architecture for vision-based Navigation*, ICAR Fifth International Conference of Advanced Robotics, Pisa, pp 1092-1097, June 91.

[Flynn 88] A. Flynn, R. Brooks : *MIT mobile robots: what's next ?*, IEEE International conference on robotics and automation, Philadelphie, pp 611-617, 1988.

[Fraichard 91] T. Fraichard, C. Laugier : *On line reactive planning for a non holonomic mobile in a dynamic world*, Proceedings IEEE Int. Conf. on robotics and automation, Sacramento, pp 432-437, April 91.

[Frohn 89] H. Frohn et W. Seelen : *VISOCAR: an autonomus industrial transport vehicule guided by visual navigation*,IEEE International conference on robotics and automation, vol 3, pp 1155-1159, 1989.

[Gandolfo 91] F. Gandolfo, G. Sandini, M. Tistarelli : *Towards vision guided manipulation*, ICAR Fifth International Conference of Advanced Robotics, Pisa, pp 661-667 June 91.

[Garcia 89] P. Garcia, C. Doncarli : *Localisation of a two-dimensional object for tracking in a series images*, rapport ENSM, 1989.

[Gelbgras 89] D. Gelbgras, M. Bogaert, G. Clermont : *Robot control algorithms for visual edge tracking*, Research center of the belgian metalworking industry, 1989.

[Giai-checa 93] B. Giai-Checa, P. Bouthemy, T. Viéville : *Détection d'objets en mouvement*, Rapport de Recherche INRIA, n 1906, Mai 1993.

[Giralt 90] G. Giralt, R. Chatila, M. Vaisset : *An integrated navigation and motion control system for autonomous multisensory mobile robots*, Autonomous Robot Vehicles, I. G. Cox, G. T. Wilfong Editors, pp 420-443, Springer Verlag, 1990.

[Grosso 92] E. Grosso, D. Ballard : *Head-Centered Orientation Strategies in Animate Vision*, Technical Report, n 442, University of Rochester, New-York, October 1992.

[Gros 92] P. Gros, L. Quan : *Présentation de la théorie des invariants sous une forme utilisable en vision par ordinateur*, Rapport technique LIFIA, RT 69 IMAG-7 LIFIA, Juillet 91.

[Hashimoto 91] K. Hashimoto, T. Kimoto, T. Ebine, H. Kimura : *Manipulator control with image-based visual servo*, Proceedings IEEE Int. Conf. on robotics and automation, Sacramento, pp 2267-2273, April 91.

[Hartley 91] R. Hartley, F. Pipitone : *Experiments with the subsumption architecture*, Proceedings IEEE Int. Conf. on robotics and automation, Sacramento, pp 1652-1658, April 91.

[Heel 90] J. Heel : *Dynamic motion vision*, Robotics and autonomous system, vol6, n3, pp 297-313, July 1990.

[Hong 91] J. Hong, X. Tan, B. Pinette, R. Weiss, E. Riseman : *Image-based homing*, Proceedings IEEE Int. Conf. on robotics and automation, Sacramento, pp 620-625, April 91.

[Hock 91] C. Hock : *A landmark navigation with Athene*, ICAR Fifth International Conference of Advanced Robotics, Pisa, pp 1099-1104, June 91.

[Hootsmans 91] N. Hootsmans, S. Dubowsky : *Large motion control of mobile manipulators including vehicle suspension characteristics*, Proceedings IEEE Int. Conf. on robotics and automation, Sacramento, pp 2336-2341, April 91.

[Hutchinson 91] S. Hutchinson : *Exploting visual constraints in robot motion planning*, Proceedings IEEE Int. Conf. on robotics and automation, Sacramento, pp 1722-1727, April 91.

[Isik 88] C. Isik, A. Meystel : *Pilot level of hierarchical controler for an unmanned mobile robot*, IEEE Journal of robotics and Automation, vol4, n3, pp 241-255, 1988.

[Jang 91] W. Jang, Z. Bien : *Feature-based visual servoing of an eye-in-hand robot with improved tracking performance*, Proceedings IEEE Int. Conf. on robotics and automation, Sacramento, pp 2254-2260, April 91.

[Jurie 93] F. Jurie, P. Rives, J. Gallice, J.L. Brame : *High speed vehicle guidance based on vision*, First IFAC, Pergamon Press, University of Southampton Hampshire, pp 205-210, 18-21 April 1993.

[Kabuka 87] M. Kabuka, A. Arenas : *Position verification of a mobile robot using standard pattern*, IEEE Journal of robotics and Automation, vol3, n6, pp 505-516, 1987.

[Kabuka 88] M. Kabuka, P. Shironoshita : *An adaptative approach to video tracking*, IEEE Journal of robotics and Automation, vol4, n2, pp 228-236, 1988.

[Kanayama 89] Y. Kanayama, Noguchi : *Locomotion functions for a mobile robot language*, IEEE RSJ International workshop on intelligent robots and systems, Tsukuba Japon, 4-6sept 1989.

[Kanayama 91] Y. Kanayama, M. Onishi : *Locomotion functions in the mobile robot language, MML*, Proceedings IEEE Int. Conf. on robotics and automation, Sacramento, pp 1110-1115, April 91.

[Kanade 91] T. Kanade, M. Okutomi : *A stereo matching algorithm with an adaptive window: Theory and experiment*, Proceedings IEEE Int. Conf. on robotics and automation, Sacramento, pp 1088-1096, April 91.

[Kay 91] Y. Kay, S. Lee : *A robust 3-D motion estimation with stereo cameras on a robot manipulator*, Proceedings IEEE Int. Conf. on robotics and automation, Sacramento, pp 1102-1107, April 91.

[Kuipers 91] B. Kuipers, Y.T. Byun : *A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations*, Robotics and Autonomous systems, Vol 8, n 1-2, pp 47-63, November 1991.

[Lebègue 92a] X. Lebègue, J. Aggarwal : *Extraction and interpretation of semantically significant line segments for a mobile robot*, Int. Conf. on Robotics and Automation, Nice, France, pp 1778-1785, May 1992.

[Lebègue 92b] X. Lebègue, J. Aggarwal : *Detecting 3-D parallel lines for perceptual organization*, European Conference on Computer Vision (ECCV'92), Santa Margherita Ligure, Italy, pp 720-724, May 1992.

[Le Moigne 88] J. Le Moigne : *Domain-dependant reasoning for visual navigation of roadways*, IEEE journal of robotics and automation, vol 1, n2, pp 40-46, August 88.

[Laurgeau 83] C. Laurgeau, M. Parent : *Les Robots: Langages et méthodes de programmation*, Edition Hermes, 1983.

[Laumond 91] J. P. Laumond : *Controllability of a multibody mobile robot*, ICAR Fifth International Conference of Advanced Robotics, Pisa, pp 1033-1038 June 91.

[Maitre 85] H. Maitre : *Un panorama de la transformée de Hough*, traitement du signal, vol2, n4, pp 305-317, 1985.

[Makela 93] H. Makela, K. Koskinen, K. Rintanen : *Navigation of a land vehicle by fusing dead reckoning and camera measurements*, First IFAC, Pergamon Press, University of Southampton Hampshire, pp 101-106, 18-21 April 1993.

[Marce 84] L. Marce : *Contribution à l'autonomie des robots mobiles*, Thèse d'état INSA de Rennes, 1984.

[Marsh 86] P. Marsh : *Le temps des robots*, encyclopédie visuelle, Bordas, 1986.

[Martinet 91] P. Martinet, P. Rives, P. Fickinger, J.J. Borrelly : *Parallel architecture for visual servoing applications*, Workshop on Computer Architecture for Machine Perception, Paris, Décembre 1991.

[Mataric 90] M. Mataric : Environment learning using a distributed representation, Proc. IEEE Int. Conf. on Robotics and Automation, Vol 1, pp 402-406, Cincinnati, May 1990.

[Matthieu 89] H. Matthieu : Le robot mobile de l'INRIA, Science Techniques et Technologie, vol 10, 1989.

[Maybeck 90] P. Maybeck : *The kalman filter: An introduction to concepts*, Autonomous Robot Vehicles, I. G. Cox, G. T. Wilfong Editors, pp 194-204, Springer Verlag, 1990.

[Michel 93] H. Michel, P. Rives : *Singularities in the determination of the situation of a robot effector from the perspective view of 3 points*, Rapport de Recherche INRIA, n 1850, Février 1993.

[Mishck 92] W. Mishck, D. Schroeder : *Performance-functionnal based controler design for a mobile manipulator*, Int. Conf. IEEE on Robotics and Automation, Vol 1, pp 227-232, Nice, May 1992.

[Moravec 77] H. Moravec : *Towards automatic visual obstacle avoidance*, IJCAI, pp 584, August 1977.

[Moravec 90] H. Moravec : *The Stanford cart and the CMU rover*, Autonomous Robot Vehicles, I. G. Cox, G. T. Wilfong Editors, pp 407-419 , Springer Verlag, 1990.

[Nagel 86] H. Nagel : *From phenomenology towards a theoritical foundation*, IEEE Journal of robotics and automation , pp 1176-1185, 1986.

[Nasr 88] H. Nasr, B. Bhanu : *Landmark recognition for autonomous mobile robots*, IEEE Int conf on robotics and automation, pp1218-1223, 1988.

[Nelson 93] B. Nelson, P. K. Khosla : *Increasing the tracking region of an eye in hand system*, Proceedings IEEE Int. Conf. on robotics and automation, vol 3, pp 418-423, Atlanta, 2-6 May 1993.

[Nelson 88] W. L. Nelson, I. J. Cox : *Local path control for an autonomous vehicle*, Proc of the IEEE Int. Conf. on robotics and automation, philadelphia, pp 1504-1510, 1988.

[Noreils 91] F. Noreils, R. Prajoux : *From plannning to execution monitoring control for indoor mobile robots*, Proceedings IEEE Int. Conf. on robotics and automation, Sacramento, pp 1510-1517, April 91.

[Occello 93] M. Occello : *"Blackboards distribués et parallèles: Application au contrôle de systèmes dynamiques en robotique et en informatique musicale"*, Thèse de l'université de Nice, Janvier 1993.

[Papanikolopoulos 93] N. Papanikolopoulos, P. Khosla, T. kanade : *Visual tracking of a moving target by a camera mounted on a robot : a combination of control and vision*, IEEE Transactions on robotics and automation, vol 9, n 1, February 1993.

[Papanikolopoulos 91] N. Papanikolopoulos, P. Khosla, T. Kanade : *Vision and control techniques for robotic visual tracking*, Proceedings IEEE Int. Conf. on robotics and automation, Sacramento, pp 857-864, April 91.

[Pissard 90] R. Pissard-Gibollet : *Pilotage d'un robot mobile*, rapport de stage ESSI3-DEA Automatique et Traitement du signal LASSY, Sophia-Antipolis, 1990.

[Pissard 91] R. Pissard-Gibollet, P. Rives : *Asservissement visuel appliqué à un robot mobile : Etat de l'art et modélisation cinématique*, Rapport de Recherche INRIA, n 1577, Décembre 1991.

[Rehg 91] J. Rehg, A. Witkin : *Visual tracking with deformation models*, Proceedings IEEE Int. Conf. on robotics and automation, Sacramento, pp 844-850, April 91.

[Rigaud 93] V. Rigaud, E. Coste-Manière, M. Perrier, A. Peuch, D. Simon : VORTEX*: Versatile and Open Subsea Robot for Technical Experiment*, Proceedings of OCEANS'93, Victoria, Canada, pp 198-203, 18-21 October 1993.

[Rives 87] P. Rives : *Dynamic vision : theoritical capabilities and practical problems*, Nato Workshop on kinematic and dynamic issues in sensor based control, Italy, Oct 1987.

[Rives 89] P. Rives, F. Chaumette, B. Espiau : *Visual servoing based on a task function approach*, First International Symposium on Experimental Robotics, Montreal, June 1989.

[Rives 92] P. Rives, R. Pissard-Gibollet, *Reactive Mobile Robots Based on a Visual Servoing Approach*, Artificial Intelligence Planning and Simulation (AIS'92), 8-10 Juillet, Perth, Australie, 1992.

[Rives 93a] P. Rives, H. Michel : *Visual Servoing Based on Ellipse Features*, Proc. of the Spie Int. conference on "Intelligent Robots and Computer Vision: Active Vision and 3D methods", pp 356-367, Boston, 8-9 September 1993.

[Rives 93b] P. Rives, R. Pissard-Gibollet, K. Kapellos : *Development of a Reactive Mobile Robot Using Real Time Vision*, Third International Symposium on Experimental Robotics, Kyoto, Japan, pp 91-109, Oct. 28-30, 1993.

[Rives 93c] P. Rives, J.J. Borrelly, J. Gallice, P. Martinet : *A Versatile Parallel Architecture for Vision Based Applications*, Workshop on Computer Architecture for Machine Perception, New Orleans, Dec 15-17, 1993

[Roy 90] V. Roy, R. de Simone : *Auto and autograph*, In R. Kurshan, editor, *proceedings of Workshop on Computer Aided Verification*, New Brunswick, June 1990.

[Rutten 92] E. Rutten, E. Coste-Manière, B. Espiau : *Task-level robot programming language and its reactive execution*, Int. Conf. IEEE on Robotics and Automation, Vol 3, pp 2751-2756, Nice, May 1992.

[Samson 90a] C. Samson, B. Espiau, M. Le Borgne : *Robot control: the task function appproach*, Oxford University Press, 1990.

[Samson 90b] C. Samson, K. Ait-Abderrahim, : *Feedback control of a nonholonomic wheeled cart in cartesian space*, Rapport de recherche INRIA n 1288, Juin 1990.

[Samson 91a] C. Samson : *Preliminary results on time-varying feedback stabilization of a nonholonomic car-like mobile robot*, Rapport de recherche INRIA n 1512, Septembre 91.

[Samson 91b] C. Samson : *Time-varying feedback stabilization of nonholonomic car-like mobile robots*, Rapport de recherche INRIA n 1515, Septembre 91.

[Samson 91c] C. Samson, K. Ait-Abderrahim : *Feedback control of a nonholonomic wheeled cart in cartesian space*, Proceedings IEEE Int. Conf. on robotics and automation, Sacramento, pp 1136-1141, April 91.

[Samson 93] C. Samson : *Time varying feedback stabilisation of a car like wheeled mobile robot*, The International Journal of Robotics Research, vol 12, n 1, pp 55-64, February 93.

[Simon 91] D. Simon, A. Joubert : *ORCCAD: Towards an Open Robot Controller Computer Aided Design system*, Rapport de recherche INRIA, n 1396, Février 1991.

[Simon 92] D. Simon, B. Espiau, E. Castillo, K. Kapellos : *Computer-aided design of generic robot controller handling reactivity and real-time control issues*, rapport de recherche INRIA, n 1801, Décembre 1992.

[Simon 93] D. Simon, B. Espiau, E. Castillo, K. Kapellos : *Computer-aided design of generic robot controller handling reactivity and real-time control issues*, IEEE Trans. on Control Systems and Technology, to appear, fall 1993.

[Singh 91] S. Singh, P. Keller : *Obstacle detection for high speed autonomous navigation*, Proceedings IEEE Int. Conf. on robotics and automation, Sacramento, pp 2798-2805, April 91.

[Skofteland 91] G. Skofteland : *Computing position and orientation of a freeflying polyhedron from 3D*, Proceedings IEEE Int. Conf. on robotics and automation, Sacramento, pp 150-156, April 91.

[Steele 86] J. P. Steele, N. D. Ebrahimi : *Control of mobile robots*, International journal of robotics and automation, vol 4, n4, pp 419-426, August 88.

[Suzuki 91] S. Suzuki, M. Habib, J. Iijima, S. Yuta : *How to describe the mobile robot's sensor-based behavior*, Robotics and Autonomous systems, Vol 8, n 1-2, pp 47-63, November 1991.

[Swain 91] M. Swain, M. Stricker editors of : *Promising directions in active vision*, Technical Report of University of Chicago, CS 91-97, November 91.

[Taalebinezhaad 91] A. Taalebinezhaad : *Direct robot motion vision by fixation*, Proceedings IEEE Int. Conf. on robotics and automation, Sacramento, pp 626-631, April 91.

[Takeno 91] J. Takeno, N. Mizuguchi, K. Sorimati : *Realization of a 3D vision mobile robot that can avoid collision with moving obstacles*, Proceedings IEEE Int. Conf. on robotics and automation, Sacramento, pp 2010-2023, April 91.

[Thomas 89] M. C. Thomas : *Apport du génie logiciel et de la conception par objets à la programmation des tâches robotisées*, Thèse de Doctorat d'Etat, Université de Nice, Février 1989.

[Thorpe 91] C. Thorpe : *Outdoor visual navigation for autonomous robots*, Robotics and Autonomous Systems, Vol 7, n 2-3, pp 85-98, August 1991.

[Tigli 93] J.Y. Tigli, M. Occello, M.C. Thomas : *Toward a new intelligent reactive controller for autonomous mobile robots*, Proceedings IEEE Int. Conf. on robotics and automation, vol 3, pp 249-254, Atlanta, 2-6 May 1993.

[Toscani 88] G. Toscani, R. Deriche, O. Faugeras : *3D motion estimation using a token traker*, Proceeding IAPR workshop on computer vision, Tokyo, oct 1988.

[Tournassoud 88] P. Tournassoud et O. Jehl : *Motion planning for a mobile robot with kinematic constraint*, IEEE International conference on robotics and automation, vol2, pp 1785-1790, 1988.

[Vernon 90] D. Vernon, M. Tistarelli : *Using camera motion to estimate range for robotic parts manipulation*, IEEE transactions on robotics and automation, vol6, n5, pp 509-521, October 1990.

[Viéville 92] T. Viéville : *3D vision of an active camera system*, British Machine Vision Association meeting on Active Vision, London, 1992.

[Viéville 93] T. Viéville and al. : *Using Visual Feedback to Drive a Mobile Robot on its Trajectory*, Workshop on computer vision for space applications, Antibes, September 1993.

[Walker 93] R. Walker, C. Harris : *A multi-sensor fusion system for a laboratory based autonomous vehicle*, First IFAC, Pergamon Press, University of Southampton Hampshire, pp 107-112, 18-21 April 1993.

[Wang 93] C. Wang, V. Kumar : *Velocity control of mobile manipulators*, Proceedings IEEE Int. Conf. on robotics and automation, vol 2, pp 713-718, Atlanta, 2-6 May 1993.

[Wavering 93] A. Wavering, J. Fiola, K. Roberts, R. Lumia : *Triclops: A high performance trinocular Active vision system*, Proceedings IEEE Int. Conf. on robotics and automation, vol 3, pp 410-417, Atlanta, 2-6 May 1993.

[Weiss 87] L. E. Weiss, A. C. Sanderson, C. P. Neuman : *Dynamic sensor-baed control of robots with visual feedback*, IEEE Journal of robotics and automation, vol RA-3, n5, pp 404-417, October 87.

[Westmore 91] D. Westmore, W. Wilson : *Direct dynamic control of a robot using an end-point mounted camera and kalman filter position estimation*, Proceedings IEEE Int. Conf. on robotics and automation, Sacramento, pp 2376-2384, April 91.

[Wijesoma 93] S. Wijesoma, D. Wolfe, R. Richards : *Eye to hand coordination for vision guided robot control applicaton*, The International Journal of Robotics Research, vol 12, n 1, pp 65-78, February 93.

[Wolfram 91] S. Wolfram : *Mathematica: a system for doing mathematics by computer*, Addison-Wesley, second edition, 1991.

[Yamamoto 93] Y. Yamamoto, X. Yun : *Control of mobile manipulators following a moving surfacing*, Proceedings IEEE Int. Conf. on robotics and automation, vol 3, pp 1-6, Atlanta, 2-6 May 1993.

[Zhang 93] Z. Zhang, R. Weiss, A.Harson : *Automatic calibration and visual servoing for a robot navigation system*, Proceedings IEEE Int. Conf. on robotics and automation, vol 1, pp 14-19, Atlanta, 2-6 May 1993.

[Yu Zheng 91b] J. yu Zheng, M. Barth, S. Tsuji : *Autonomous landmark selection for route recognition by a mobile robot*, Proceedings IEEE Int. Conf. on robotics and automation, Sacramento, pp 2004-2009, April 91.

[Yu Zheng 91a] J. Yu Zheng, Q. Chen, S. Tsuji : *Active camera guided manipulation*, Proceedings IEEE Int. Conf. on robotics and automation, Sacramento, pp 632-638, April 91.