No d'Ordre : 528

THESE

Présentée devant

L'Université de Rennes I

Institut de Formation Supérieure en Informatique et Communication

Pour obtenir

Le Titre de Docteur de l'Université de Rennes I
Mention INFORMATIQUE

par
François CHAUMETTE

Titre de la thèse

La relation vision-commande : théorie et application à des tâches robotiques.

Soutenue le 19 juillet 1990 devant la commission d'examen

MM.	Jean-Pierre	Banatre	Président
	James L.	Crowley	Rapporteur
	Bernard	Espiau	Rapporteur
	Olivier	Faugeras	Examinateur
	Jean	Gallice	Examinateur
	Claude	Labit	Examinateur
	Patrick	Rives	Examinateur
	Giulio	Sandini	Examinateur

Je tiens à remercier les membres de ce jury d'avoir accepté de juger ce travail :

Jean-Pierre Banatre, Professeur à l'INSA de Rennes et Directeur de l'IRISA, qui me fait l'honneur de présider ce jury,

Jim Crowley, Professeur à l'INPG de Grenoble, pour avoir accepté la lourde tâche d'être rapporteur,

Olivier Faugeras, Directeur de recherche à l'INRIA Sophia-Antipolis, et Giulio Sandini, Professeur à l'Université de Gênes, pour l'intérêt qu'ils ont bien voulu porter à mes travaux,

Jean Gallice, Professeur à l'Université Blaise-Pascal de Clermont-Ferrand, pour sa collaboration et pour avoir notamment accepté de nous prêter un prototype de leur carte de traitements d'images,

Claude Labit, Directeur de recherche à l'IRISA, responsable de l'équipe TEMIS pour m'avoir accueilli pendant ces trois ans et pour m'avoir apporté sa confiance.

Il m'est difficile de dissocier dans ces remerciements Bernard Espiau, Directeur de recherche à l'INRIA, détaché actuellement à l'Ecole des Mines de Paris en tant que Directeur de l'ISIA, et Patrick Rives, Chargé de recherche à l'INRIA. Ils m'ont proposé un sujet de thèse passionnant et n'ont pas épargné leurs efforts pour m'aider tout au long de cette thèse. Je leur dois beaucoup et même davantage.

Je voudrais également remercier :

Jean-Pierre Merlet et Claude Samson pour les trop rares mais très enrichissantes discussions que nous avons eues,

Jean-Luc Corre pour m'avoir aidé à utiliser son logiciel de visualisation et qui m'a permis d'obtenir des résultats de simulation non rébarbatifs. Je lui souhaite sincèrement, ainsi qu'à ses collègues de TIMEAT, une brillante carrière dans le monde industriel.

Jean-Jacques Borrelly pour son aide inestimable lors de la mise au point du banc expérimental et pour m'avoir fourni, clé en main, une commande numérique particulièrement performante. Je le remercie également pour sa patience dont j'ai souvent abusé. Je me souviendrai longtemps d'une mise en route de l'AID effectuée par ses soins à plus de 1000 km de l'AID !

mes différents colocataires de bureau, les membres de l'atelier MICRO, des équipes SIAMES, Téléopération et, bien entendu, TEMIS, les deux Patrick notamment, qui ont fait régner une ambiance excellente tout au long de ces trois ans.

Je voudrais enfin remercier ceux qui m'ont soutenu lors de ma candidature à l'INRIA de Rennes et qui m'ont permis d'intégrer l'équipe TEMIS en tant que Chargé de recherche.

Je n'oublie pas, non plus, Sophie et les membres de l'IAF qui m'ont fait oublier, de temps à autre, la commande référencée vision.

Trugarez bras deoc'h.

Table des Matières

Introduction

O combien était triste la vie des premiers robots : aller à la position A, prendre l'objet O, aller à la position B, déposer l'objet O, retourner en A, prendre un nouvel objet O, et ainsi de suite, jusqu'à ce qu'on lui apprenne un beau jour de nouvelles positions A' et B'...

Cependant, comme l'avait deviné Maxime Le Forestier :

> Petit robot qui va venir,
> Il est joli ton avenir.

En effet, depuis quelques années, les développements technologiques des capteurs (miniaturisation, robustesse) ont permis leur implantation directement sur l'effecteur des robots mobiles ou manufacturiers. L'intérêt principal de cette intégration est de fournir une information sensorielle directe entre le robot et son environnement et d'ouvrir ainsi de nombreuses perspectives nouvelles en robotique.

Les tâches, telles qu'évitement d'obstacles en utilisant des capteurs proximétriques, positionnement et suivi d'objets mobiles en utilisant la vision, ou suivi de surface avec des capteurs d'effort, paraissent maintenant tout à fait envisageables.

Ainsi, le champ d'application des robots est passé d'un univers fixe et entièrement connu à un univers parfois totalement inconnu ou variable.

Bien entendu, ces nouvelles perspectives ont amené de nombreux problèmes théoriques, concernant notamment l'aspect commande. Ces nouveaux problèmes

ont donné naissance à la commande référencée capteurs : celle-ci consiste, non plus à utiliser les informations fournies par les capteurs dans des procédures de haut niveau, mais à les intégrer dans des lois de commande en Boucle Fermée sur ces informations.

L'utilisation du capteur de vision, à laquelle nous nous sommes attachés, est particulièrement intéressante en raison de la grande richesse des informations qu'une caméra peut fournir et en raison de la grande variété des tâches qu'elle permet de réaliser.

Cette grande richesse nécessite cependant de disposer d'algorithmes de traitement d'images particulièrement performants afin d'extraire, à une cadence proche de la cadence vidéo, les informations qui seront utilisées en commande. Ces algorithmes de traitements d'images ont déjà fait, et font encore, l'objet de nombreux travaux. Aussi nous sommes-nous principalement intéressés à l'aspect commande.

Cet ouvrage est structuré en cinq chapitres qui portent respectivement sur les points suivants :

(1) Après avoir présenté les multiples intérêts des capteurs embarqués et les diverses utilisations de la vision en robotique, nous décrivons les différentes approches que l'on peut trouver dans la littérature concernant l'emploi d'informations visuelles au sein d'une boucle de commande. L'approche *asservissement visuel*, que nous développerons exclusivement par la suite, y est particulièrement détaillée.

(2) Le chapitre 2 est consacré à la modélisation des informations visuelles. Celles-ci sont décomposées en un ensemble de signaux élémentaires monodimensionnels et mesurables dans l'image. Chaque signal est relié aux primitives géométriques 2D provenant de la projection dans l'image des primitives 3D de la scène.

On définit ensuite une notion absolument fondamentale : le *torseur d'interaction* qui relie les variations d'un signal en fonction des différents mouvements possibles de la caméra. Une méthode générale de calcul du torseur d'interaction est proposée pour l'ensemble des signaux élémentaires que l'on peut définir à partir des primitives géométriques paramétrables. Les résultats explicites du torseur d'interaction des signaux élémentaires les plus représentatifs sont donnés.

(3) Dans le chapitre 3, on rappelle tout d'abord la classification des tâches robotiques qui a été effectuée à partir de la notion de *liaison virtuelle* entre le capteur et son environnement et qui est tout à fait applicable à l'approche asservissement visuel. On présente ensuite, pour chaque type de tâches, différents *motifs* capables de les représenter correctement dans l'espace image. Ces motifs sont constitués d'une combinaison adéquate de signaux élémentaires et sont obtenus grâce à la connaissance du *torseur invariant*, torseur réciproque du torseur d'interaction.

(4) Après un bref rappel des résultats obtenus par Claude Samson et Bernard Espiau [Samson 90b] en commande des robots manipulateurs rigides et en commande référencée capteurs, leur approche, basée sur la notion de *fonction de tâche*, est appliquée au cas spécifique du capteur visuel. Cette approche permet notamment d'élaborer, toujours à partir du torseur d'interaction, des lois de commande particulièrement robustes et de considérer, en outre, les aspects de *redondance*, aussi bien au niveau tâche qu'au niveau informations visuelles.

(5) Nous présentons enfin les résultats qui ont été obtenus, soit en simulation, soit sur site expérimental, et qui portent sur la réalisation des tâches qui nous ont paru les plus significatives.

Quelques perspectives de recherche sont tracées en guise de conclusion.

Chapitre Premier

Cadre de l'étude

Après le rappel de quelques définitions qui nous seront utiles par la suite, nous présentons dans ce chapitre l'intérêt d'utiliser en robotique des capteurs, appelés extéroceptifs car capables de fournir des informations sur l'environnement d'un robot. Nous nous focalisons ensuite sur une fonction sensorielle particulière : la vision. Après avoir cité les multiples utilisations de la vision en robotique, nous décrivons les différentes approches concernant la commande à partir des informations visuelles.

1.1 Rappel de quelques définitions usuelles en robotique

Nous ne faisons dans ce court paragraphe qu'une présentation très simplifiée des notions de base en robotique. Elles seront reprises par la suite, et ce de manière beaucoup plus détaillée, dans le cadre de la commande référencée capteurs.

Un robot est un système mécanique capable de se mouvoir (pour un robot mobile) ou de déplacer son effecteur (pour un robot manipulateur) afin de réaliser une tâche qui lui a été assignée.

La *situation* de l'effecteur (dans notre cas un capteur embarqué), notée $\bar{r}$, décrit la position et l'orientation de celui-ci par rapport à un repère choisi. On peut signaler que six variables, appelées *variables opérationnelles* et notées $\underline{r}$, sont nécessaires et suffisantes pour représenter une situation : trois pour la position (coordonnées cartésiennes, cylindriques ou sphériques) et trois pour l'orientation

(angles d'Euler par exemple). Ainsi, un robot évolue dans un univers à trois dimensions et à six degrés de liberté.

Le nombre de *degrés de liberté* de l'effecteur d'un robot est défini comme étant le nombre de paramètres indépendants capables de représenter l'ensemble des situations que peut atteindre son effecteur. Ce nombre de degrés de liberté varie de un à six en fonction du nombre d'axes du robot et de leur configuration.

Les mouvements d'un robot sont engendrés par l'application de forces ou de couples, notés $\underline{\Gamma}$, qui sont transmis aux différents axes du robot par l'intermédiaire d'actionneurs (moteurs électriques par exemple). Les capteurs proprioceptifs permettent de mesurer les effets de ces actionneurs sur l'état interne du robot. Les variables représentant cet état sont appelées les *variables articulaires* et sont notées $\underline{q}$. Elles fournissent les positions respectives de chaque axe par rapport à une position de référence.

Le *modèle géométrique* d'un robot manipulateur permet de calculer la situation $\bar{r}$ de l'effecteur correspondant à une position quelconque $\underline{q}$. Le *modèle géométrique inverse* fournit, lui, une des solutions $\underline{q}^*$, si elle existe, correspondant à une situation voulue $\bar{r}^*$ représentée par $\underline{r}^*$ ($\underline{q}^* = MGINV(\underline{r}^*)$). Le schéma de commande d'un classique asservissement en position dans l'espace opérationnel est représenté Figure 1.1.

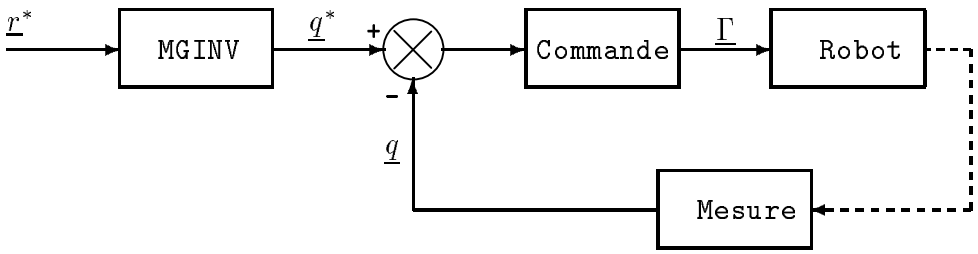


Figure 1.1 : Asservissement en position

On peut également définir le *jacobien* d'un robot manipulateur. Celui-ci relie la vitesse $\dot{\underline{q}}$ dans l'espace articulaire au torseur cinématique $\dot{\bar{r}}$ ($\dot{\bar{r}} = J(\underline{q})\ \dot{\underline{q}}$). Pour réaliser un asservissement en vitesse dans l'espace opérationnel (ce qui peut être le cas si l'on souhaite contrôler la situation de l'effecteur au cours d'un déplacement ou si la consigne est directement exprimée sous forme de vitesse dans l'espace opérationnel), on utilise, quand cela est possible, le *jacobien inverse*

$J^{-1}_{(\underline{q})}$ (voir Figure 1.2 où la commande capable de réaliser cet asservissement en vitesse n'est pas détaillée). Le jacobien n'est évidemment pas inversible lorsque le nombre de degrés de liberté du robot n'est pas égal à son nombre d'axes. Il ne l'est pas davantage lorsque le robot traverse une éventuelle *singularité*, position $\underline{q}$ particulière telle que le déterminant de $J(\underline{q})$ soit nul.

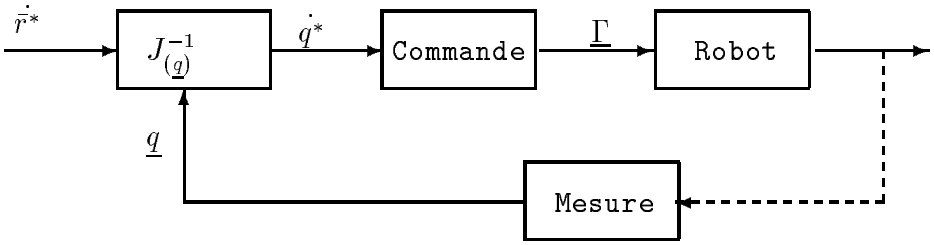


Figure 1.2 : Asservissement en vitesse

Enfin, pour terminer ces brefs rappels, signalons que les tâches à effectuer peuvent généralement s'exprimer [Samson 90b] sous la forme d'une régulation (à zéro) d'une *fonction de tâche*, notée $\underline{e}(\underline{q}, t)$. Quelques exemples de telles fonctions sont :

- $\underline{e}(\underline{q}, t) = \underline{q} - \underline{q}^*(t)$ où $\underline{q}^*(t)$ est une trajectoire désirée dans l'espace articulaire,
- $\underline{e}(\underline{q}, t) = \underline{r}(\underline{q}) - \underline{r}^*(t)$ où $\underline{r}^*(t)$ est une trajectoire désirée dans l'espace opérationnel de l'effecteur d'un robot à six degrés de liberté.

Nous verrons par la suite qu'avec l'aide des capteurs extéroceptifs de nombreuses autres tâches peuvent être réalisées et qu'il est notamment possible de les exprimer directement dans l'espace du capteur utilisé.

1.2 Intérêts des capteurs extéroceptifs

L'utilisation exclusive des capteurs proprioceptifs paraît extrêmement limitative : en effet, les tâches ne peuvent alors s'exprimer que dans les espaces articulaire ou opérationnel. Ainsi, seules les tâches de suivi de trajectoires prédéfinies, dont on a donné les fonctions de tâche précédemment, ou de positionnement

(que l'on peut d'ailleurs considérer comme un suivi de trajectoire particulière) sont réalisables.

Il faut cependant souligner les problèmes de précision que l'on rencontre en pratique pour passer de l'espace articulaire, dans lequel sont exprimées les informations fournies par les capteurs proprioceptifs, à l'espace opérationnel. En effet, cette étape nécessite l'emploi du modèle géométrique du robot. Ce modèle est généralement entâché d'erreurs en raison des problèmes complexes liés à son identification. Aussi la situation de l'effecteur fournie par le modèle géométrique est-elle une situation estimée, et même souvent biaisée. Les tâches exprimées dans l'espace opérationnel peuvent ainsi être correctement réalisées du point de vue "commande" (c'est-à-dire que la situation mesurée est égale à la situation souhaitée) sans être réalisées effectivement : aucun réel retour sur la situation exacte de l'effecteur n'est possible en utilisant les capteurs proprioceptifs.

De plus, les problèmes de frottement et de roulement avec ou sans glissement interdisent à un robot mobile d'effectuer précisément une trajectoire prédéfinie, et même d'effectuer plusieurs fois la même trajectoire. Ces problèmes de répétabilité sont beaucoup moins sensibles pour les robots manipulateurs. Ils subsistent cependant car directement liés au bon fonctionnement et à la précision plus ou moins grande des capteurs proprioceptifs.

Ainsi, quelle que soit la performance de la loi de commande utilisée, il est impossible de connaître exactement la position d'un robot mobile ou la situation de l'effecteur d'un robot manipulateur. Même dans le cas où un robot évolue dans un univers parfaitement connu, les capteurs proprioceptifs ne permettent pas de situer précisément le robot dans cet univers. Ceux-ci fournissent des informations sur l'état interne du système robotique, et seulement sur cet état interne.

Il paraît alors important d'utiliser des capteurs extéroceptifs capables de fournir des informations soit sur l'environnement du robot, soit sur la situation du robot dans cet environnement. Le champ d'application des robots s'en trouve d'ailleurs grandement élargi : en utilisant des capteurs adéquats et performants, on peut envisager l'emploi des robots dans des univers complexes, peu ou mal connus et comprenant des objets en mouvement (c'est d'ailleurs le domaine d'application essentiel de la robotique mobile).

La première utilisation des capteurs extéroceptifs qui vient à l'esprit consiste à contrôler le bon déroulement d'une tâche. Les informations (du style "Le robot n'est pas au bon endroit" ou mieux "Le robot doit aller à tel endroit") sont

alors utilisées dans des procédures de haut niveau : la loi de commande reste inchangée pour réaliser un positionnement ou un suivi de trajectoire. Par contre, les informations fournies par les capteurs permettent de modifier éventuellement la trajectoire ou la position désirées, toujours exprimées dans l'espace articulaire ou opérationnel, de manière à réaliser correctement la tâche qui a été assignée au robot.

Mais, il paraît beaucoup plus intéressant d'intégrer les informations fournies par les capteurs extéroceptifs dans des lois de commande en boucle fermée sur ces informations. En effet, les fonctions de tâches peuvent alors s'exprimer directement dans l'espace du capteur et non plus dans les espaces articulaire ou opérationnel. Elles sont spécifiées sous la forme d'une relation entre le robot et son environnement et la régulation de ces fonctions de tâche permet d'obtenir la situation souhaitée du robot dans cet environnement. Les imprécisions du modèle du robot, qui empêchaient tout positionnement précis dans l'espace opérationnel, n'interdisent plus un tel positionnement dans l'environnement du robot. En effet, les perturbations des commandes envoyées au robot, entraînées par ces imprécisions de modèle, sont compensées par la loi de commande en boucle fermée sur les informations fournies par les capteurs extéroceptifs.

De plus, la définition des fonctions de tâches dans l'espace du capteur permet de considérer des tâches beaucoup plus variées que celles de positionnement ou de suivi de trajectoire. De fait, toutes les tâches qui peuvent s'exprimer sous la forme d'une relation entre le robot et son environnement sont réalisables en utilisant les capteurs adéquats. Par exemple, si l'on dispose d'un capteur capable de fournir la distance d entre l'effecteur du robot et un objet, une tâche de positionnement à une distance d^* de cet objet peut simplement s'écrire $e(\underline{q}, t) = d(\underline{q}, t) - d^*$ si l'objet est mobile et $e(\underline{q}) = d(\underline{q}) - d^*$ si l'objet est fixe. On voit sur cet exemple que cette tâche ne spécifie, par sa définition même, que la distance entre le robot et l'objet. Une infinité de situations du robot (si celui-ci a plus d'un degré de liberté et si la satisfaction de $e = 0$ est possible) permet de réaliser cette tâche. Aussi son expression sous la forme d'un suivi de trajectoire aurait-elle été impossible. Nous verrons par la suite de nombreux autres exemples de tâches, appelées naturellement *tâches référencées capteurs*, capables de contrôler de un à six degrés de liberté du robot.

Une classe d'utilisation des capteurs extéroceptifs semble particulièrement intéressante : celle où les capteurs sont embarqués sur l'effecteur du robot considéré. En effet, on accède ainsi à des informations sensorielles directes sur l'interaction qui relie le robot à son environnement local. La phase

d'interprétation des informations fournies est alors réduite à sa plus simple expression puisqu'il n'est plus nécessaire de situer le robot et l'environnement par rapport au capteur, puis le robot par rapport à son environnement. Au contraire, la liaison entre le capteur et l'effecteur du robot permet d'utiliser directement dans les boucles de commande les informations issues des capteurs extéroceptifs.

Pour revenir à l'exemple précédent de positionnement à une distance d^* d'un objet, la fonction de tâche peut maintenant s'écrire sous la forme $e(\underline{q}, t) = d'(\underline{q}, t) - d'^*$ où d' et d'^* représentent respectivement les distances mesurée et voulue entre le capteur embarqué et l'objet, la valeur de d'^* étant calculée à partir de d^* et de la liaison entre le capteur et l'effecteur du robot.

L'utilisation de capteurs extéroceptifs, si elle a permis d'envisager de nombreuses perspectives en robotique, a bien évidemment amené de nombreux problèmes théoriques concernant notamment les points suivants :

- la modélisation des capteurs (et l'identification de la liaison entre le capteur et l'effecteur du robot dans le cas des capteurs embarqués) afin d'utiliser à bon escient les informations fournies,
- le choix des capteurs et des informations à prendre en compte pour réaliser une tâche donnée,
- la fusion des données fournies par des capteurs, ceux-ci pouvant être de nature différente,
- l'élaboration et l'étude de lois de commande en boucle fermée intégrant ces informations, ce point ayant donné naissance à la *commande référencée capteurs*.

Nous ne nous sommes pas intéressés à l'aspect fusion des données capteurs (voir par exemple sur ce sujet [Durrant-Whyte 88]), puisque nous n'avons utilisé qu'un seul capteur : la vision. Par contre, nous reviendrons très souvent par la suite à l'étude générale de la commande référencée capteurs, sous l'approche fonction de tâche, proposée dans [Samson 90a]. Nous avons en effet appliqué cette approche à la commande référencée vision.

Généralement, les travaux en commande référencée capteurs ont surtout porté sur des capteurs choisis pour la simplicité et la rapidité d'obtention des informations fournies (de manière à pouvoir élaborer des lois de commande à haute fréquence d'échantillonnage et ayant donc de bonnes propriétés de stabilité). On peut citer, concernant les capteurs proximétriques et des tâches telles

qu'évitement d'obstacles, centrage ou positionnement par rapport à un plan, les travaux décrits dans [Wampler 84], [Balek 85], [Espiau 85], [Cheung 89]. Dans [Espiau 87b], une approche générale concernant la commande référencée capteurs en utilisant des capteurs proximétriques est présentée, ayant notamment aboutie à une classification des différentes tâches en fonction de la liaison souhaitée entre le capteur et son environnement.

Concernant les télémètres acoustiques, on peut citer les travaux de [André 85] et [Flynn 88]. Les capteurs d'effort n'ont encore été que peu utilisés. Les applications industrielles, concernant notamment le soudage et le suivi de surface, devraient entraîner dans les années qui viennent de nombreux travaux sur ce domaine.

Le capteur de vision peut bien évidemment être utilisé lui-aussi pour réaliser des tâches robotiques (c'est d'ailleurs le sujet de cet ouvrage !). Il est particulièrement intéressant par la grande richesse des informations qu'il peut fournir sur l'environnement du robot. Il ne faut pas non plus oublier l'intérêt non négligeable de tenter de "donner la vue à un robot".

Après la description succinte des différentes possibilités d'utilisation de la vision en robotique, nous présenterons les informations susceptibles d'être considérées en commande, caractérisées par leur rapidité d'extraction de l'image. Nous détaillerons ensuite les différentes approches concernant l'aspect commande.

1.3 Utilisation de la vision en robotique

Le capteur de vision est assurément l'un des capteurs extéroceptifs les plus riches : il fournit, en effet, non pas une seule information, comme c'est par exemple le cas des capteurs proximétriques (distance du capteur à la surface la plus proche dans la direction du capteur), mais un signal vidéo que l'on peut numériser dans une image, tableau bidimensionnel dont chaque élément, appelé *pixel*, contient une information sur l'intensité lumineuse d'un point visible de la scène.

Les informations que l'on peut extraire d'une image peuvent donc être extrêmement variées et de plus ou moins haut niveau selon les différents traitements que l'on effectue sur cette image : de l'intensité lumineuse d'un pixel à la reconnaissance et à la localisation dans l'image d'un objet. Bien entendu, plus les informations sont de haut niveau, plus les algorithmes de traitement d'image nécessaires à leur extraction sont complexes.

Afin d'utiliser à bon escient ces différentes informations, une phase préalable de modélisation est, comme pour tout capteur, indispensable. Cette phase consiste à identifier les paramètres du modèle de la caméra (distance focale, dimensions d'un pixel,...). Ce modèle, constitué principalement par une projection perspective (voir Figure 1.3), permet d'établir la correspondance entre l'univers 3D et sa projection 2D dans l'image. De nombreux travaux ont été effectués sur ce domaine, ayant donné naissance à plusieurs méthodes de calibration. On peut notamment citer [Faugeras 87], [Tsai 87a], [Chaumette 89b] et [Abi-Ayad 89].

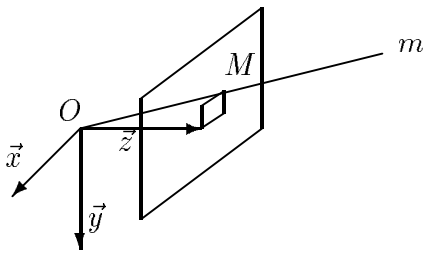


Figure 1.3 : Projection perspective

On trouvera en Annexe, à la fin de cet ouvrage, la méthode de calibration qui a été mise en œuvre afin d'identifier un système robot manipulateur-caméra embarquée. Elle repose notamment sur l'utilisation des informations fournies par la caméra pour identifier les paramètres du modèle du robot et sur l'utilisation d'une méthode non linéaire pour identifier les paramètres du modèle de la caméra.

Une information visuelle ne permet pas à elle seule de remonter à l'information tridimensionnelle de l'environnement : sur un pixel de l'image peut se projeter l'ensemble des points de la droite (O, M) situés devant le plan image et il n'est pas possible, sans connaissance a priori sur la scène, de déterminer lequel de ces points s'est projeté sur M.

Par contre, connaissant les dimensions d'un objet, il est possible de retrouver la localisation de cet objet dans l'espace tridimensionnel [Rives 81], [Horaud 87], [Dhome 89], [Faugeras 90].

Lorsque l'on ne peut formuler aucune hypothèse, plusieurs images sont nécessaires pour reconstruire la scène 3D. Celles-ci peuvent être obtenues :

- soit à l'aide de systèmes stéréoscopiques à deux, ou mieux, trois caméras [Faugeras 86] [Ayache 88] [Faugeras 88],
- soit à l'aide d'une caméra mobile dont les mouvements sont connus entre

chaque acquisition : c'est là le domaine de la vision dynamique [Rives 87], [Xie 89], [Crowley 90] et même de la vision active quand les mouvements de la caméra sont générés de manière à améliorer la perception tridimensionnelle de l'environnement [Alomoinos 87], [Espiau 87a], [Sandini 90].

A partir de cette information tridimensionnelle sur l'environnement du robot, il est alors possible de calculer une trajectoire dans cet environnement afin d'amener le robot à une situation voulue et en tenant compte des éventuels obstacles de la scène [Faverjon 84], [Boissonnat 88], [Blisset 88]. Les tâches de positionnement et de suivi de trajectoires que le robot peut réaliser doivent alors s'exprimer dans l'espace opérationnel. Pour pouvoir considérer des scènes dynamiques (c'est-à-dire comportant des objets mobiles) et prendre en compte les erreurs de modèle du robot, une boucle fermée sur cette reconstruction 3D est alors indispensable.

Cependant, cette phase de reconstruction 3D n'est pas toujours nécessaire pour réaliser une tâche robotique. Certaines tâches ne nécessitent même aucune information sur la localisation des objets dans la scène. Considérons par exemple une tâche de positionnement d'un objet au centre de l'image : la seule information qui est indispensable pour réaliser cette tâche est de pouvoir extraire de l'image les coordonnées du centre de gravité de la projection de cet objet. Cette seule information permet de calculer la direction de déplacement de la caméra afin de ramener l'image de cet objet au centre de l'image et ainsi de réaliser la tâche voulue.

De plus, comme les algorithmes mis en œuvre pour effectuer une reconstruction 3D sont souvent coûteux au niveau temps de calcul et viennent, de surcroît, s'ajouter aux algorithmes d'extraction des informations de l'image, il paraît intéressant d'éviter d'avoir à reconstruire l'environnement du robot pendant l'exécution d'une tâche. C'est ce que permet l'approche asservissement visuel, qui sera détaillée dans le paragraphe suivant.

Généralement, les informations visuelles qui ont été utilisées en commande ont été choisies en fonction de l'existence préalable d'algorithmes permettant de les extraire de l'image en un temps compatible avec la boucle de commande du robot (c'est-à-dire proche de la cadence vidéo). De ce fait, il n'est pas étonnant que la majorité des travaux porte sur l'utilisation de primitives *contour* : points et segments de droite [Rives 87], [Espiau 87a] pour des applications d'évitement d'obstacles, ou de primitives *région* : surface, centre de gravité, axes principaux d'inerties [Gilbert 80] pour la poursuite de cibles et [Weiss 84], [Urban 90] notamment, pour des applications de positionnements. Différentes primitives,

obtenues sans segmentation dans l'image (différence de phase entre les transformées de Fourier de deux images, surface à l'intérieur d'une fenêtre), sont présentées dans [Kabuka 88]. On peut également citer une approche originale [Sanderson 80] où un objet polyédrique est décrit en termes de graphe relationnel : les nœuds de ce graphe représentent la surface des faces de l'objet et les arcs, la longueur de ses arêtes. Cette approche permet de prendre en compte les différents éléments de ce graphe selon leur présence éventuelle dans l'image.

De nombreux travaux ont également été réalisés sur la détection de droites à la cadence vidéo. Les applications envisagées concernaient bien évidemment la conduite automatique de véhicules sur route [Dickmanns 85], [Kuan 87] ou dans un environnement approprié [Reid 87], en l'occurence un champ parsemé de sillons...

Dans le paragraphe suivant sont présentées les différentes approches qui ont été utilisées en commande à partir de ces informations.

1.4 Types de commande utilisant la vision

On peut distinguer deux approches concernant l'utilisation d'informations visuelles en commande :

- la première est basée sur un asservissement en situation de la caméra par rapport à son environnement,
- la seconde, plus récente et se rapprochant davantage de l'approche commande référencée capteurs, est basée sur une régulation dans l'image.

Les premiers travaux ayant abouti à ce formalisme sont dûs à Weiss et Sanderson [Sanderson 83].

1.4.1 Asservissement en situation

Sous cette approche, les tâches à réaliser, en général de positionnement par rapport à un objet, sont exprimées sous la forme d'une situation $\bar{r}^*$ à atteindre entre la caméra et cet objet. Une estimation $\hat{\bar{r}}$ de la situation courante est alors indispensable afin d'atteindre la situation voulue (voir Figure 1.4).

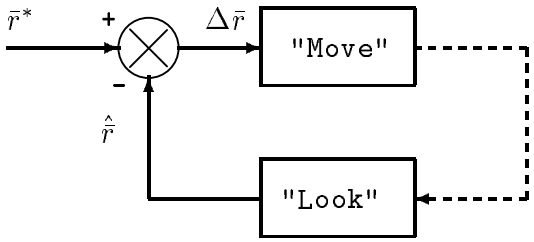


Figure 1.4 : Asservissement en situation

Une étape d'interprétation des informations extraites de l'image courante permet quand elles sont bien choisies, de fournir cette estimation mais nécessite une des hypothèse suivantes :

- soit les dimensions de l'objet sont connues,
- soit la distance entre la caméra et l'objet est connue.

En effet, comme nous l'avons déjà évoqué précédemment, les informations contenues dans une image ne permettent pas de calculer de manière unique la situation d'un objet par rapport à la caméra. Une méthode qui permet d'outrepasser ces hypothèses consiste à utiliser une seconde caméra, soit fixe, soit également embarquée de manière à remonter à l'information tridimensionelle de l'objet par triangulation.

1.4.1.1 "Static Look and Move"

Le contrôle du robot est assuré par un asservissement classique en position dans l'espace cartésien. Le mode de fonctionnement consiste en un enchaînement séquentiel des deux étapes suivantes :

(1) la caméra acquiert une image, un algorithme en extrait les informations choisies et en déduit une estimation de la situation courante entre la caméra et l'environnement,

(2) la loi de commande calcule un déplacement de la caméra en fonction de l'erreur entre la situation actuelle et la situation à atteindre, traduit ce déplacement dans l'espace cartésien sous la forme de consignes dans l'espace articulaire à l'aide du modèle géométrique inverse ; le robot exécute ce déplacement et attend l'arrêt de la caméra avant l'acquisition d'une nouvelle image.

Théoriquement, une seule itération est nécessaire pour asservir la caméra à la situation voulue. Malheureusement, les différentes erreurs qui peuvent s'introduire dans ce processus nécessitent une répétition de la séquence "Look and Move" jusqu'à l'obtention d'une erreur entre la situation courante et la situation souhaitée inférieure à celle spécifiée lors de la définition de la tâche.

Trois types d'erreurs peuvent intervenir dans ce processus :

- les erreurs d'extraction des informations visuelles, inhérentes à tout traitement dans l'image (problèmes de seuillage par exemple),
- les erreurs d'estimation de la situation entre la caméra et son environnement, dûes aux inévitables erreurs de calibration de la caméra et aux erreurs d'interprétation,
- les erreurs de modèle du robot, dûes aux erreurs de construction ou à une modélisation imprécise, qui impliquent que la situation réelle de la caméra ne correspond pas exactement avec celle qui est mesurée.

Cependant, la simplicité de ce processus a permis son utilisation dès la fin des années 1970 dans les premières applications robotiques utilisant la vision et portant sur la localisation et la prise d'objets simples. On peut citer :

- [Agin 77], [Tani 77], [Birk 81] : la pièce à saisir et la caméra sont dans deux plans parallèles à une distance constante et connue. Les trois degrés de liberté qui sont asservis, à savoir les deux translations et la rotation dans ce plan sont ainsi obtenues directement à partir des informations visuelles.
- [Kashioka 77] : la situation entre la caméra embarquée et l'objet est obtenue par triangulation à l'aide d'une seconde caméra fixe.

Le fait d'attendre la fin du mouvement du robot avant de pouvoir acquérir une nouvelle image entraîne de faibles performances au niveau temps de réponse et interdit l'utilisation du "Look and Move" dans les cas de scènes non statiques. Cette structure, du fait de son aspect statique, est cependant la seule à contrôler uniquement la situation finale de la caméra, et non pas sa vitesse (ou trajectoire). Dans les cas de reconfiguration nécessaire ou de présence d'obstacles entre la situation courante et la situation à atteindre, la structure "Look and Move" permet de résoudre simplement ces problèmes par les différents choix possibles de trajectoires entre deux situations.

1.4.1.2 "Dynamic Look and Move"

La structure du "Dynamic Look and Move" est identique à celle du "Static Look and Move" hormis le fait que les deux étapes "Look" et "Move" décrites précédemment peuvent s'effectuer simultanément, bien que, la plupart du temps, à des cadences différentes. Cette structure permet de considérer des objets en mouvement puisque les informations visuelles sont prises en compte dès qu'elles sont opérationnelles. Cependant, les temps de calcul des phases d'extraction d'informations visuelles et d'interprétation doivent rester compatibles avec la vitesse de l'objet de manière à ce que celui-ci reste dans le champ de vue de la caméra entre deux acquisitions successives.

De plus, une loi de commande doit gérer les effets provoqués par la différence de cadence entre les deux étapes, de manière à assurer la stabilité du système. L'utilisation du jacobien, au lieu du modèle géométrique inverse, pour effectuer les déplacements du robot permet de prendre en compte l'aspect dynamique du système en assurant que l'objet reste continuellement dans le champ de vue de la caméra pendant les déplacements du robot.

La première implantation d'une telle structure est dûe à [Agin 79] pour une application de suivi de pièces sur un tapis roulant. Dans [Ward 79] et [Albus 81], des systèmes équivalents sont proposés mais sans présenter de résultats. Par contre, dans [Coulon 83], les caractéristiques dynamiques d'un système à un degré de liberté couplé à une caméra sont étudiées et une commande avec gains proportionnel et intégral est proposée.

1.4.1.3 "Position Based Visual Servoing"

Cette structure, proposée dans [Sanderson 83], supprime l'asservissement en boucle fermée sur la position du robot et permet de s'affranchir des calculs du modèle géométrique inverse ou du jacobien. Une loi de commande doit alors calculer directement les couples à envoyer aux moteurs à partir de l'erreur entre les situations estimée et voulue. Cette structure n'a encore jamais été utilisée en pratique en raison de la difficulté d'élaborer des lois de commande pour contrôler un robot en boucle ouverte à partir de consignes dans l'espace cartésien et à cadence très faible (temps d'acquisition d'une image, d'extraction des informations utiles et d'interprétation souvent incompatibles avec la dynamique d'un robot). Dans certains cas particuliers (contrôle d'un seul degré de liberté, robot à structure cartésienne par exemple), il est certainement possible d'élaborer de telles lois de commande mais la structure "Position Based Visual Servoing" est alors quasiment identique à celle du "Dynamic Look and Move" et ne présente alors que très peu d'intérêt.

1.4.2 Asservissement visuel

L'emploi des techniques basées sur un asservissement sur la situation de la caméra présente l'inconvénient majeur de nécessiter une phase d'interprétation des informations visuelles, de manière à reconstruire la situation courante entre la caméra et son environnement. Les problèmes inhérents à cette interprétation sont de deux sortes :

- Les algorithmes qu'elle nécessite peuvent être extrêmement lourds dans le cas de scènes complexes, réduisant grandement la dynamique du système.
- Les erreurs de précision de cette phase d'interprétation entraînent inévitablement des erreurs dans l'estimation de la situation courante. Ainsi, la convergence de l'algorithme peut être obtenue (c'est-à-dire que la situation estimée est égale à la situation voulue) sans que la tâche soit effectivement réalisée (situation réelle différente de la situation voulue).

L'approche asservissement visuel permet, elle, de s'affranchir totalement de cette phase d'interprétation et semble être optimale au niveau temps de calcul puisque les informations extraites de l'image sont directement utilisées dans la boucle de commande. En effet, les consignes ne sont plus exprimées sous la forme de situation entre la caméra et la scène, mais sous la forme d'un motif à atteindre dans l'image : les informations visuelles $\underline{s}$ choisies pour constituer ce motif doivent atteindre les valeurs $\underline{s}^*$ qui correspondent à une bonne réalisation de la tâche (voir Figure 1.5).

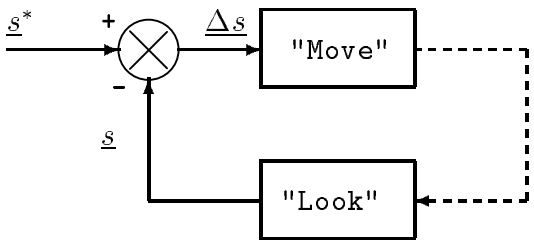


Figure 1.5 : Asservissement visuel

Les tâches réalisables par l'approche asservissement en situation sont également réalisables par l'approche asservissement visuel. Une phase d'interprétation inverse, consistant très simplement à projeter dans l'image les données de la scène correspondant à la situation voulue permet de définir les valeurs que doivent atteindre les informations visuelles. Il est cependant nécessaire de s'assurer que

les informations choisies permettent de réaliser effectivement la tâche, comme il était également nécessaire, dans l'approche asservissement en situation, de s'assurer que les informations visuelles choisies permettaient, lors de la phase d'interprétation, de remonter à la situation courante.

De plus, l'approche asservissement visuel est tout à fait adaptée pour réaliser un éventail beaucoup plus large de tâches s'exprimant directement dans l'image (telles qu'asservir la caméra de sorte qu'un objet soit centré dans l'image par exemple).

Certaines applications traitées par cette approche concernant notamment la poursuite de cibles [Gilbert 80], [Dzialo 86], la convergence d'un système stéréoscopique sur un point donné [Bajcsy 88], [Clark 89] ou encore le guidage d'un véhicule sur route [Dickmanns 85], ont pu être réalisées sans étude approfondie du point de vue commande. En effet, grâce à la simplicité des tâches considérées et au faible nombre de degrés de liberté contrôlés (deux, sauf dans [Gilbert 80] où quatre degrés de liberté sont utilisés), les informations visuelles, choisies de manière intuitive mais efficace, permettent d'estimer directement les valeurs que doivent atteindre les axes en question de manière à réaliser la tâche.

On peut également signaler les résultats originaux décrits dans [Corke 89] où la fonction de transfert entre les coordonnées dans l'image du centre de gravité d'un objet et un robot cartésien à deux degrés de liberté est donnée, prenant en compte les différents retards dûs aux temps de calcul d'extraction des informations visuelles et au temps de réponse d'une consigne de déplacement du robot.

Les premiers travaux spécifiques en commande sont dûs à Sanderson [Sanderson 80] mais pour le cas très simple d'un seul degré de liberté : une loi proportionnelle, avec gain adaptatif, basée sur la mesure dans l'image de la surface d'un objet parfaitement connu est choisie pour commander cet axe. Des résultats beaucoup plus significatifs sont proposés dans [Weiss 84] et [Weiss 87], toujours fondés sur la commande adaptative. Le système proposé (Single Input Single Output Model Reference Adaptative Controller) supprime, comme dans la structure "Position Based Visual Servoing", la boucle fermée sur l'asservissement du robot et ne nécessite donc pas le calcul du jacobien inverse du robot pour contrôler celui-ci. L'intérêt d'une telle approche est, selon eux, de se libérer des erreurs de commande impliquées par les imprécisions inévitables du modèle d'un robot. Une telle structure paraît séduisante du point de vue théorique. Cependant, des approximations et restrictions importantes sont nécessaires pour pouvoir appliquer leur loi de commande :

- Chaque degré de liberté du robot ne peut être contrôlé que par une seule information visuelle alors qu'il serait souhaitable d'en considérer plusieurs

afin de compenser les erreurs de mesure dans l'image.

- Le choix des informations visuelles ne peut s'effectuer qu'après une lourde phase d'apprentissage : à partir de la situation souhaitée entre la caméra et la scène (ce qui signifie que l'on a déjà pu réaliser la tâche auparavant !), les variations des informations visuelles en fonction d'un déplacement incrémental de chaque axe du robot sont estimées. L'information visuelle choisie pour contrôler un axe est telle que sa variation soit la plus forte possible selon cet axe et la plus faible possible pour l'ensemble des autres axes.

- La structure du robot ne doit pas présenter de couplages importants entre différents axes de manière à pouvoir sélectionner les informations visuelles. De plus, le nombre de degrés de liberté que l'on peut contrôler par cette méthode semble limité (des résultats sont donnés dans [Weiss 84] et [Weiss 87] pour des robots ayant au maximum trois axes), toujours en raison de la difficulté de sélectionner les informations visuelles.

- La convergence de la commande n'est assurée que lorsque la prépondérance de la variation de l'information visuelle sur l'axe choisi par apprentissage est respectée, ce qui n'est le cas que dans un voisinage de la situation souhaitée. Aucun résultat sur l'étendue d'un tel voisinage n'a encore été donné.

- La phase d'apprentissage doit être effectuée pour chaque tâche que doit réaliser le robot et les résultats qu'elle fournit ne sont absolument pas généraux puisque spécifiques à chaque structure de robot et à chaque scène.

On peut alors se demander quel est l'intérêt d'élaborer des lois de commande liant directement le robot aux informations visuelles, d'autant plus que les erreurs introduites par les imprécisions du modèle du robot sont compensées par l'aspect boucle fermée de la loi commande.

Le travaux de Weiss et Sanderson ont cependant permis de mettre en évidence un point absolument essentiel pour utiliser proprement l'approche asservissement visuel : le lien entre les variations dans l'image des informations visuelles et les mouvements de la caméra, (ce que nous appelerons par la suite torseur d'interaction) afin, d'une part, de choisir les informations visuelles capables de représenter une tâche et, d'autre part, d'élaborer des lois de commande à partir de celles-ci.

Très peu de travaux ont, jusqu'à présent, été menés sur cet aspect de modélisation. Le calcul explicite du torseur d'interaction n'a été obtenu que pour des informations visuelles très simples :

- les coordonnées d'un point ([Feddema 89a]),
- les paramètres représentant un segment (longueur, orientation et coordonnées du milieu du segment) dans le cas particulier où celui-ci est parallèle au plan image ([Feddema 89b]), c'est-à-dire que les rotations autour des axes du plan image ne sont pas considérées.

Un effort important de modélisation nous a paru indispensable de manière à obtenir une approche générale, d'une part, pour obtenir un éventail plus étendu d'informations visuelles utilisables en commande, et, d'autre part, pour pouvoir considérer l'ensemble des mouvements possibles de la caméra. Cette étude dont on peut trouver des résultats préliminaires dans [Rives 89] et [Espiau 90] est l'objet du chapitre suivant. On y trouvera une méthode pour calculer de manière explicite le torseur d'interaction de toute information visuelle que l'on peut définir à partir des primitives géométriques paramétrables.

Cette phase de modélisation nous a permis de traiter l'ensemble des problèmes d'asservissement visuel sous forme globale. Dans le Chapitre 3 sont classées les différentes tâches réalisables par asservissement visuel. La classification des tâches repose sur la liaison qui doit exister entre la caméra et la scène correspondant à une bonne réalisation de la tâche (suivi de droite ou de plan, par exemple). Grâce à la modélisation des informations visuelles qui permet justement de connaître cette liaison, différentes combinaisons des informations visuelles sont proposées, de manière à offrir un éventail de motifs capables de représenter une classe de tâches donnée.

Enfin, la modélisation des informations visuelles permet d'intégrer les problèmes d'asservissement visuel dans le domaine de la commande référencée capteurs, notamment sous l'approche fonction de tâche. Les résultats généraux, décrits dans [Espiau 87b], [Samson 90a] et [Samson 90b] notamment, s'appliquent parfaitement au capteur vision apportant des résultats entièrement nouveaux sur la robustesse et la stabilité des lois de commande, la possibilité d'employer des informations visuelles redondantes ou de considérer le concept des tâches hybrides (par exemple, tâche de positionnement par la commande référencée vision tout en effectuant un suivi de trajectoire prédéfinie). Ces différents résultats concernant l'aspect commande sont détaillés dans le Chapitre 4.

Chapitre 2

Modélisation

Dans ce chapitre, on définit les informations visuelles que l'on peut utiliser en commande à travers l'approche proposée. On définit également le torseur d'interaction, notion fondamentale, qui relie les variations des informations visuelles aux mouvements de la caméra (ou de la scène). Une méthode de calcul du torseur d'interaction est ensuite proposée pour l'ensemble des primitives géométriques paramétrables. Les résultats explicites sont donnés pour les primitives les plus usuelles : point, segment, droite, cercle, sphère et enfin cylindre.

2.1 Définitions

2.1.1 Notations

Nous précisons dans ce court paragraphe les différentes notations relatives aux espaces dans lesquels nous travaillerons par la suite. Nous rappelons également la définition et les propriétés fondamentales des torseurs.

Soit e l'espace euclidien à 3 dimensions, l'espace vectoriel associé étant $\mathbb{R}^3$. L'espace de configuration des corps rigides et des repères est le groupe de Lie des déplacements de e, appelé traditionnellement SE_3 (Special Euclidian Group), isomorphe à $\mathbb{R}^3 \times SO_3$ où SO_3 est le groupe des rotations. Comme nous l'avons vu dans le chapitre 1 (paragraphe 1.1), c'est une variété différentielle de dimension 6 dont un élément est une situation $\bar{r}$.

L'espace tangent à SE_3 en l'identité, noté se_3, est une algèbre de Lie isomorphe à l'algèbre de Lie des champs équiprojectifs de e dans $\mathbb{R}^3$, ce qui signifie que tout élément (champ) de se_3 n'est autre que le traditionnel torseur cinématique.

Classiquement, un torseur H est aussi défini par son vecteur ω et la valeur

de son champ en un point O de e. Exprimé dans un repère, un torseur est un vecteur de $\mathbb{R}^6$ et on notera donc $H = (H(O), \omega)$. Il est bien entendu possible d'exprimer la valeur d'un torseur en tout point M de e et on a l'équation bien connue :

$$H(M) = H(O) + \omega \times OM \tag{2.1}$$

Le produit de deux torseurs est l'application bilinéaire associée à

$$\begin{bmatrix} 0 & \mathbb{I}_3 \\ \mathbb{I}_3 & 0 \end{bmatrix}$$

où $\mathbb{I}_3$ est l'identité de $\mathbb{R}^3$ dans $\mathbb{R}^3$. Il s'écrit, quel que soit le point O considéré :

$$H_1 \bullet H_2 = \omega_1 \,.\, H_2(O) + \omega_2 \,.\, H_1(O) \tag{2.2}$$

où . est le produit scalaire usuel entre deux vecteurs de $\mathbb{R}^3$. Le produit de torseurs induit un isomorphisme entre un espace de torseurs et son dual (ou espace cotangent). Ainsi, le dual de se_3, noté se_3^*, est lui-même un espace de torseurs. Le torseur d'interaction, qui sera défini par la suite, appartient à cet espace se_3^*.

2.1.2 Les informations visuelles

On définit une information visuelle s, également appelée signal capteur par analogie avec la commande référencée capteur, par la donnée d'une application différentiable de SE_3 dans $\mathbb{R}$.

Cette définition implique que la valeur d'une information visuelle ne dépend que de la situation entre la caméra et son environnement. Ainsi on considère que seules des modifications de nature géométrique (c'est-à-dire des déplacements de la caméra ou d'objets constituant l'environnement) sont susceptibles de faire varier la valeur d'une information visuelle.

Signalons que nous ne considèrerons dans la suite de cet ouvrage que des informations visuelles de type géométrique. Ainsi, nous n'étudierons pas de signaux capteurs basés sur des mesures photométriques dans l'image. En effet, leur haute sensibilité aux conditions d'éclairement fait que ces mesures peuvent parfois varier en dehors de tout déplacement. Elles ne rentrent alors plus dans le cadre de la définition ci-dessus. De plus, les hypothèses simplificatrices usuelles (surface des objets parfaitement Lambertienne, éclairage diffus) sont généralement irréalistes en robotique et interdisent toute modélisation précise.

Revenons à présent à la caractérisation des informations visuelles : on sait que la caméra est embarquée sur l'effecteur d'un robot. Les déplacements de la caméra sont donc réalisés à l'aide des différents axes constituant le robot et la situation de la caméra ne dépend que de la valeur des coordonnées articulaires $\underline{q}$. De plus, si les objets perçus sont eux-mêmes mobiles, alors s peut s'écrire :

$$s = s(\underline{q}, t) \tag{2.3}$$

où le paramètre temps t représente cette contribution du mouvement des objets.

On peut ainsi intégrer parfaitement le domaine de la commande sous l'approche fonction de tâche qui a été évoquée précédemment. En effet, les tâches qui peuvent être réalisées par asservissement visuel peuvent s'exprimer comme la régulation de la fonction de tâche suivante :

$$\underline{e}(\underline{q}, t) = C\ (\underline{s}(\underline{q}, t) - \underline{s}^*) \tag{2.4}$$

où

- $\underline{s}$ est le vecteur représentant l'ensemble des informations visuelles choisies pour réaliser la tâche,
- $\underline{s}^*$ est la valeur que $\underline{s}$ doit atteindre pour que la tâche soit effectivement réalisée,
- C est une matrice, dite de combinaison, qui permet notamment de prendre en compte dans la commande un nombre d'informations visuelles supérieur au nombre de degrés de liberté contraints par $\underline{e}$.

Nous verrons dans le chapitre suivant une classification des différentes tâches et les différentes combinaisons possibles d'informations visuelles s capables de représenter une tâche donnée par le vecteur $\underline{s}$. La partie commande, portant notamment sur la construction de la matrice C et sur la régulation de la fonction $\underline{e}$ sera, elle, traitée dans le chapitre 4. On peut cependant observer dès à présent l'importance majeure de $\underline{s}$ dans la constitution de la fonction de tâche et l'on va maintenant caractériser sa différentielle de manière à pouvoir calculer, par analogie avec le jacobien d'un robot, le jacobien d'une tâche.

2.1.3 Le torseur d'interaction

La différentielle de $\underline{s}$ permet de relier les variations des informations visuelles dans l'image aux mouvements de la caméra et de la scène. A partir de

l'équation (2.3), on obtient facilement :

$$\dot{\underline{s}} = \frac{\partial \underline{s}}{\partial \underline{q}} \, \dot{\underline{q}} + \frac{\partial \underline{s}}{\partial t} \tag{2.5}$$

Le terme $\frac{\partial \underline{s}}{\partial \underline{q}}$ peut en fait se décomposer sous la forme :

$$\frac{\partial \underline{s}}{\partial \underline{q}} = \frac{\partial \underline{s}}{\partial \bar{r}} \, \frac{\partial \bar{r}}{\partial \underline{q}} \tag{2.6}$$

où l'on reconnaît le classique jacobien du robot $\frac{\partial \bar{r}}{\partial \underline{q}}$ si l'on choisit pour $\bar{r}$ la situation de la caméra. Cette décomposition permet donc d'exhiber :

- un terme ne dépendant que de la configuration du robot et en aucun cas de la tâche : le jacobien du robot,
- un terme ne dépendant que de la tâche choisie, représentée par $\underline{s}$, et en aucun cas de la configuration du robot porteur de la caméra : le terme $\frac{\partial \underline{s}}{\partial \bar{r}}$ que l'on peut appeler jacobien de la tâche.

Il est donc essentiel de caractériser le jacobien de la tâche pour conserver une approche générale applicable à tout porteur potentiel de caméra et, notamment, quel que soit son nombre de degrés de liberté.

Pour ce faire, examinons une composante s_j de $\underline{s}$. En vertu des préliminaires mathématiques énoncés plus haut, et de la définition de s_j, nous savons que sa différentielle au point $\bar{r}$, $ds_{j|_{\bar{r}}}$, est une application linéaire de $se_{3|_{\bar{r}}}$ dans $\mathbb{R}$. Par ailleurs, on sait que la différentielle de toute fonction analytique d'une variété M dans $\mathbb{R}$ peut être identifiée avec un élément de l'espace cotangent. Dans notre cas, ceci implique que la différentielle de s_j en $\bar{r}$ n'est autre qu'un élément de se_3^*, c'est-à-dire un torseur. Rappelant qu'un élément de se_3 est aussi un torseur cinématique T, nous pouvons finalement écrire au point $\bar{r}$ de SE_3 la relation fondamentale :

$$\dot{s}_j = H_j \bullet T \tag{2.7}$$

où :

- T est le torseur cinématique représentant la vitesse relative de la scène par rapport à la caméra,
- $\bullet$ est le produit de torseurs défini plus haut,

- H_j est un torseur dont l'expression dépend à la fois des caractéristiques de l'environnement et du capteur lui-même. Il caractérise complètement les interactions entre capteur et environnement et nous l'appellerons donc *torseur d'interaction*. Nous verrons par la suite qu'il joue un rôle essentiel aussi bien dans les aspects de modélisation que dans la commande elle-même.

Avec un abus de notation évident, la relation (2.7) peut aussi s'écrire :

$$\dot{s}_j = L_j^T \; T \quad \text{où} \quad L_j^T = H_j \begin{pmatrix} 0 & \mathbb{I}_3 \\ \mathbb{I}_3 & 0 \end{pmatrix} \tag{2.8}$$

L_j^T est la représentation matricielle du torseur d'interaction H_j exprimé dans un repère et en un point donnés. De même, la représentation matricielle de l'ensemble $\{H_1 \cdots H_k\}$ associé à $\underline{s} = (s_1 \cdots s_k)^T$ est appelée *matrice d'interaction* et notée L^T. Exprimée dans le repère de la caméra, cette matrice correspond exactement au jacobien de la tâche représentée par $\underline{s}$ lorsque :

$$T = \frac{\partial \bar{r}}{\partial \underline{q}} \, \dot{\underline{q}} \tag{2.9}$$

c'est-à-dire lorsque l'environnement perçu par la caméra est immobile.

Nous allons maintenant, dans le reste de ce chapitre, nous attacher à caractériser plus précisément les informations visuelles utilisables en commande référencée vision.

2.2 Quelles informations visuelles peut-on utiliser ?

Deux conditions sont essentielles pour pouvoir utiliser une information visuelle en commande référencée vision : tout d'abord, une condition évidemment nécessaire est qu'il existe des algorithmes de traitement d'images capables de mesurer cette information, et ce à une cadence la plus proche possible de la cadence d'échantillonnage de la commande du robot. La seconde condition est qu'il soit possible de calculer de manière explicite le torseur d'interaction de cette information, afin de connaître le jacobien de la tâche qui utilisera cette information. Avant de proposer une méthode générale de calcul du torseur d'interaction pour toute information définissable à partir des primitives géométriques paramétrables, nous allons considérer les deux informations visuelles les plus simples : les coordonnées d'un point dans l'image.

2.2.1 Un cas très simple : le point

Signalons tout d'abord que le repère de la caméra $(O, \vec{x}, \vec{y}, \vec{z})$ donné Figure 2.1 sera par défaut celui dans lequel seront exprimées par la suite toutes les grandeurs nécessaires, par exemple, le torseur cinématique, les coordonnées des points, les torseurs d'interaction, etc.

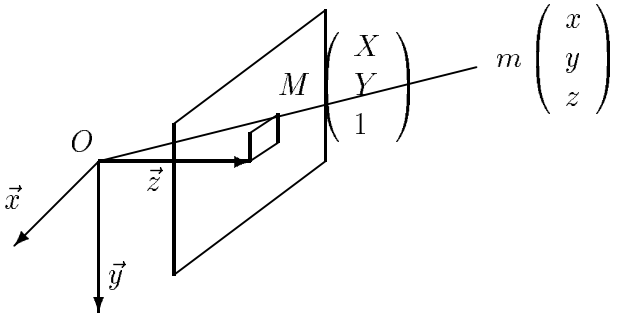


Figure 2.1 : Modèle de la caméra

Sans la moindre perte de généralité, on peut supposer la distance focale de la caméra égale à 1, de sorte que tout point m de e ayant pour coordonnées $\underline{x} = (x\, y\, z)^T$ se projette par projection perspective sur le plan image en M de coordonnées $\underline{X} = (X\, Y\, 1)^T$ avec :

$$\underline{X} = \frac{1}{z}\, \underline{x} \tag{2.10}$$

En différentiant cette équation, on obtient les variations dans l'image des coordonnées X et Y de M par rapport à la vitesse $V(m)$ du point m :

$$\left\{ \begin{array}{rcl} \dot{X} & = & \left(\begin{array}{ccc} 1/z & 0 & -x/z^2 \end{array} \right) . \; V(m) \\ \dot{Y} & = & \left(\begin{array}{ccc} 0 & 1/z & -y/z^2 \end{array} \right) . \; V(m) \end{array} \right. \tag{2.11}$$

Supposons par exemple que le point m soit immobile et que la caméra soit animée d'une vitesse de translation $V(O)$ et d'une vitesse de rotation Ω. Le torseur cinématique prend pour valeur $T = (-V(O), -\Omega)$ et la vitesse du point m s'écrit :

$$V(m) = -V(O) - \Omega \times \underline{x} \tag{2.12}$$

L'équation (2.11) peut alors aisément se simplifier, en utilisant l'équation (2.10), sous la forme :

$$\left(\begin{array}{c} \dot{X} \\ \dot{Y} \end{array} \right) = L_{of}^T(\underline{X}, z)\; T \tag{2.13}$$

où

$$L_{of}^T(\underline{X}, z) = \begin{pmatrix} -1/z & 0 & X/z & XY & -(1+X^2) & Y \\ 0 & -1/z & Y/z & 1+Y^2 & -XY & -X \end{pmatrix} \quad (2.14)$$

Nous retrouvons là les équations bien connues de l'"Optic Flow" reliant les variations des coordonnées d'un point dans l'image au torseur cinématique. Pour nous, la matrice L_{of}^T est la représentation matricielle des torseurs d'interaction des deux informations visuelles constituées des coordonnées d'un point dans l'image.

Nous allons maintenant généraliser cette approche aux informations visuelles construites à partir de primitives géométriques quelconques.

2.2.2 Caractérisation des informations visuelles

Considérons, dans e, un solide rigide auquel peuvent être liées des primitives 3D (points, lignes, sphères, etc,...). Nous appellerons *motif dans la scène* un tel ensemble de primitives. Une configuration de ce motif est un élément p de l'ensemble $\mathcal{P}_s$ de toutes ses configurations possibles. Lorsque ce motif caractérise complètement la situation du corps auquel il est lié, la dimension n' de $\mathcal{P}_s$ est égale à 6. Sinon, elle est inférieure.

Soit f l'application "projection perspective" telle que $f(p) = P$. P est un *motif dans l'image* et appartient à l'ensemble $\mathcal{P}_i$ de dimension $m < n'$. Nous restreindrons notre étude au cas où p appartient à un ouvert U inclus dans $\mathcal{P}_s$ tel que $P = f(p)$ ne soit pas un élément "dégénéré" de $\mathcal{P}_i$ (cas par exemple où une droite se projette dans l'image sous la forme d'un point, un cercle sous la forme d'un segment, un cylindre sous la forme d'une ellipse,...). Cette restriction implique en particulier qu'il existe un paramétrage complet $\underline{P}$ de P sur tout l'ouvert $V = f(U) \subseteq \mathcal{P}_i$. On supposera en outre que ce paramétrage est différentiable et minimal (de dimension m).

Soit $\underline{p}$ un paramétrage de p de dimension $n \geq n'$, complet et unique sur U au sens où un seul paramétrage est nécessaire pour représenter n'importe quelle configuration du motif p dans U. Si l'on note $\underline{p} = \phi(p)$ et $\underline{P} = \psi(P)$, on a alors (voir Figure 2.2) :

$$\underline{P} = \psi \circ f \circ \phi^{-1}(\underline{p}) \quad (2.15)$$

$$\begin{array}{ccccccc} W \subseteq SE3 & \longrightarrow & U \subseteq \mathcal{P}_s & \longrightarrow & V \subseteq \mathcal{P}_i & & \\ (\bar{r}) & \delta & (p) & f & (P) & & \\ & & \Big\downarrow \phi & & \Big\downarrow \psi & & \\ & & \mathbb{R}^n & \longrightarrow & \mathbb{R}^m & \longrightarrow & \mathbb{R} \\ & & (\underline{p}) & \psi \circ f \circ \phi^{-1} & (\underline{P}) & \sigma & (s) \end{array}$$

Figure 2.2 : Passage du motif scène au motif image

Par ailleurs, le groupe des déplacements SE_3 opère sur U par l'application δ de telle sorte que, si l'on restreint les déplacements à un ouvert W de SE_3 tel que $\delta(W) \subseteq U$ de manière à ne jamais obtenir de motif image P dégénéré, alors :

$$\underline{p} = \phi \circ \delta(\bar{r}) \tag{2.16}$$

d'où :

$$\underline{P} = \psi \circ f \circ \delta(\bar{r}) \tag{2.17}$$

On peut alors choisir comme information visuelle une fonction $s = \sigma(\underline{P})$ telle que σ soit différentiable, le cas le plus fréquent étant de choisir pour s une composante de $\underline{P}$.

Le calcul de L_s^T, représentation matricielle du torseur d'interaction de s, se ramène au calcul de l'expression :

$$L_s^T = \frac{\partial s}{\partial \underline{P}} \frac{\partial \underline{P}}{\partial P} \frac{\partial P}{\partial p} \frac{\partial p}{\partial \bar{r}} \tag{2.18}$$

Toutefois, cette forme n'est assurément pas la forme la plus appropriée pour un calcul analytique : comment, par exemple, obtenir explicitement le terme $\frac{\partial P}{\partial p}$?

Dans le cas où le paramétrage $\underline{p}$ de p est différentiable, comme c'est par exemple le cas des coordonnées pluckeriennes représentant les droites, l'équation précédente peut alors s'écrire :

$$L_s^T = \frac{\partial s}{\partial \underline{P}} \frac{\partial \underline{P}}{\partial \underline{p}} \frac{\partial \underline{p}}{\partial \bar{r}} \tag{2.19}$$

où chaque terme peut être évalué séparément.

Cependant, il est souvent extrêmement contraignant, quand ce n'est pas impossible, d'utiliser des paramétrages complets, uniques, minimaux et différentiables pour tout motif de la scène ! Aussi propose-t-on dans le paragraphe suivant une méthode de calcul du torseur d'interaction pour toute information visuelle construite à partir de primitives géométriques paramétrables.

2.2.3 Méthode de calcul du torseur d'interaction

Soit $\{p_1, \cdots, p_j\}$ l'ensemble des j primitives constituant le motif p de la scène. De même, soit $\{P_1, \cdots, P_j\}$ l'ensemble des j primitives constituant le motif P de l'image, résultant de la projection de p. Si l'on note $\underline{P_i}$ le paramétrage de P_i alors s peut s'écrire sous la forme générale :

$$s = \sigma(\underline{P_1}, \cdots, \underline{P_j}) \tag{2.20}$$

Ainsi, des informations visuelles reliant plusieurs primitives de l'image entre elles (par exemple orientation entre deux droites, distance d'un point à une droite) peuvent être considérées. A partir de cette équation (2.20), on peut écrire L_s^T sous la forme :

$$L_s^T = \sum_{i=1}^{j} \frac{\partial s}{\partial \underline{P_i}} \frac{\partial \underline{P_i}}{\partial \bar{r}} \tag{2.21}$$

Le calcul du terme $\frac{\partial s}{\partial \underline{P_i}}$ est souvent trivial, aussi allons-nous maintenant nous intéresser au calcul de $\frac{\partial \underline{P_i}}{\partial \bar{r}}$ qui correspond à la matrice d'interaction L^T associée au paramétrage $\underline{P_i}$. Pour simplifier les notations ultérieures, p_i, $\underline{p_i}$, P_i et $\underline{P_i}$ seront respectivement notées p, $\underline{p}$, P et $\underline{P}$.

Les primitives du motif dans la scène peuvent généralement être spécifiées par une équation de la forme :

$$h(\underline{x}, \underline{p}) = 0 \tag{2.22}$$

où h définit la nature de la primitive et où la valeur de $\underline{p}$ correspond à une de ses configurations.

En utilisant l'équation du modèle de la caméra (2.10), l'équation (2.22) s'écrit :

$$h(z\underline{X}, \underline{p}) = 0 \tag{2.23}$$

soit

$$h'(\underline{X}, z, \underline{p}) = 0 \tag{2.24}$$

Le théorème des fonctions implicites permet alors de dire autour d'une solution $\underline{x_0}$ de (2.22), sous la condition $\frac{\partial h'}{\partial z} \neq 0$ toujours vérifiée exceptés les cas dégénérés décrits précédemment, qu'il existe une fonction unique μ telle que :

$$z = \mu(\underline{X}, \underline{p}) \tag{2.25}$$

Considérons tour à tour les différents types de primitives que l'on peut rencontrer : droites, primitives planes et tridimensionnelles, le cas trivial du point

ayant déjà été traité. Nous allons nous attacher à obtenir de manière explicite la fonction μ et la projection dans l'image de $h(\underline{x}, \underline{p}) = 0$ que nous écrirons sous la forme :

$$g(\underline{X}, \underline{P}) = 0 \tag{2.26}$$

- Cas de la droite et des primitives planes : h est alors de dimension 2 ($h = (h_1\ h_2)^T$). On peut supposer par exemple que h_2 représente le plan auquel appartient la primitive (dans le cas de la droite, l'écriture de h_2 n'est pas unique). μ est alors construite à partir de $h'_2(\underline{X}, z, \underline{p}) = 0$. En utilisant (2.25), $h'_1(\underline{X}, z, \underline{p}) = 0$ donne :

$$\tilde{h}(\underline{X}, \underline{p}) = 0 \ \text{ avec } \dim \tilde{h} = 1 \tag{2.27}$$

  qui s'écrit, après reparamétrisation $g(\underline{X}, \underline{P}) = 0$.

- Cas des primitives tridimensionnelles (sphères, cylindres, tores, ..., ou même intersection de plusieurs primitives tridimensionnelles) : pour ce cas également, il est possible d'exhiber les fonctions μ et g données par les équations (2.25) et (2.26). En effet, la projection dans l'image d'une telle primitive est alors entièrement caractérisée par le contour de cette projection. La fonction $g(\underline{X}, \underline{P})$ qui représente ce contour, est alors donnée par l'équation des limbes et la particularité des points 3D correspondant aux points contours peut s'exprimer sous la forme $h'_0(\underline{X}, z, \underline{p}) = 0$, avec $\dim h'_0 = 1$, qui fournit la fonction $z = \mu(\underline{X}, \underline{p})$.

Le calcul de la matrice d'interaction L^T peut à présent s'effectuer en remarquant que l'hypothèse naturelle de rigidité des primitives géométriques implique :

$$\dot{g}(\underline{X}, \underline{P}) = 0\ ,\ \forall \underline{X} \in P \tag{2.28}$$

D'où, en dérivant :

$$\frac{\partial g}{\partial \underline{P}}(\underline{X}, \underline{P})\ \dot{\underline{P}} = -\frac{\partial g}{\partial \underline{X}}(\underline{X}, \underline{P})\ \dot{\underline{X}}\ ,\ \forall \underline{X} \in P \tag{2.29}$$

Or, on a vu qu'il était possible de relier $\dot{\underline{X}}$ au torseur cinématique par les équations d'Optic Flow. On rappelle que :

$$\dot{\underline{X}} = L^T_{of}(\underline{X}, z)\ T \tag{2.30}$$

En utilisant (2.25) dans (2.30), il vient :

$$L^T_{of}(\underline{X}, z) = L'^T_{of}(\underline{X}, \underline{p}) \tag{2.31}$$

Finalement, (2.29), (2.30) et (2.31) donnent :

$$\frac{\partial g}{\partial \underline{P}}(\underline{X},\underline{P})\ \dot{\underline{P}} = -\frac{\partial g}{\partial \underline{X}}(\underline{X},\underline{P})\ L_{of}'^{T}(\underline{X},\underline{p})\ T\ ,\ \forall \underline{X} \in P \tag{2.32}$$

que l'on peut résoudre, soit, après utilisation explicite de (2.26) dans (2.32), par identification terme à terme (voir par exemple dans le paragraphe suivant le calcul de la matrice d'interaction dans le cas des droites et des cercles), soit de la façon suivante :

La matrice L^T recherchée est de dimension $m \times 6$. On choisit alors m points distincts M_i de coordonnées $\underline{X}_i$ appartenant à P, caractérisant complètement cette primitive, et l'on obtient :

$$\left\{ \begin{array}{rcll} \dfrac{\partial g}{\partial \underline{P}}(\underline{X}_i,\underline{P}) & = & \alpha_i^T(\underline{P}) & ,i = 1 \text{ à } m \\ -\dfrac{\partial g}{\partial \underline{X}}(\underline{X}_i,\underline{P}) L_{of}'^{T}(\underline{X}_i,\underline{p}) & = & \beta_i^T(\underline{p},\underline{P}) & ,i = 1 \text{ à } m \end{array} \right. \tag{2.33}$$

La matrice d'interaction s'écrit alors :

$$L^T(\underline{p},\underline{P}) = \begin{pmatrix} \alpha_1^T \\ \vdots \\ \alpha_m^T \end{pmatrix}^{-1} (\underline{P}) \begin{pmatrix} \beta_1^T \\ \vdots \\ \beta_m^T \end{pmatrix} (\underline{p},\underline{P}) \tag{2.34}$$

- **Remarque :** Il apparaît dans cette dernière équation (2.34) que le calcul de L^T n'est possible que si la matrice $(\alpha_1 \cdots \alpha_m)$ est inversible, c'est-à-dire, entre autres, si la représentation de g par les paramètres $\underline{P}$ est minimale.

Enfin, signalons que si l'on souhaite calculer la matrice d'interaction correspondant à la même primitive, mais associée à un autre paramétrage $\underline{P}'$, également différentiable et minimal, il n'est pas nécessaire d'utiliser les méthodes précédentes. En effet, il suffit alors de connaître le difféomorphisme ν tel que $\underline{P}' = \nu(\underline{P})$ et son réciproque ν^{-1} pour pouvoir calculer simplement la matrice d'interaction associée à $\underline{P}'$ à partir de celle associée à $\underline{P}$:

$$L^T(\underline{p},\underline{P}') = \frac{\partial \underline{P}'}{\partial \underline{P}}\ L^T(\underline{p},\underline{P}) \tag{2.35}$$

où $\underline{P}$, présent dans le second membre de cette équation, est remplacé par $\nu^{-1}(\underline{P}')$.

Nous pouvons maintenant appliquer ces différentes méthodes de calcul du torseur d'interaction à l'ensemble des primitives géométriques. Nous décrivons dans le paragraphe suivant celles qui nous ont paru les plus significatives : segment, droite, cercle, sphère et cylindre.

2.3 Etude des primitives les plus usuelles

Pour chaque primitive qui seront étudiées dans ce paragraphe, nous calculerons tout d'abord l'équation de la projection dans l'image de cette primitive. Nous proposerons ensuite les différentes paramétrisations possibles et calculerons les matrices d'interaction correspondantes.

2.3.1 Les segments

Considérons un segment 3D d'extrémités les points m_1 et m_2. Ce segment peut être simplement représenté dans l'image par les coordonnées X_1, Y_1, X_2 et Y_2 de ses extrémités M_1 et M_2. La matrice d'interaction est alors simplement obtenue à partir des équations d'Optic Flow (2.13) et on a :

$$\begin{array}{lcl}
L_{X_1}^T & = [& -1/z_1 \quad 0 \quad X_1/z_1 \quad X_1Y_1 \quad -(1+X_1^2) \quad Y_1 \ ] \\
L_{Y_1}^T & = [& 0 \quad -1/z_1 \quad Y_1/z_1 \quad 1+Y_1^2 \quad -X_1Y_1 \quad -X_1 \ ] \\
L_{X_2}^T & = [& -1/z_2 \quad 0 \quad X_2/z_2 \quad X_2Y_2 \quad -(1+X_2^2) \quad Y_2 \ ] \\
L_{Y_2}^T & = [& 0 \quad -1/z_2 \quad Y_2/z_2 \quad 1+Y_2^2 \quad -X_2Y_2 \quad -X_2 \ ]
\end{array} \tag{2.36}$$

Une autre représentation, notée $\underline{P'}$, des segments est souvent utilisée : la longueur l, l'orientation α et les coordonnés X_c et Y_c du milieu du segment. Le passage de cette représentation à la précédente, notée $\underline{P}$, est immédiat :

$$\left\{ \begin{array}{l}
X_c = (X_1 + X_2)/2 \\
Y_c = (Y_1 + Y_2)/2 \\
l = \sqrt{(X_1 - X_2)^2 + (Y_1 - Y_2)^2} \\
\alpha = \arctan(Y_1 - Y_2)/(X_1 - X_2)
\end{array} \right. \tag{2.37}$$

A l'aide de ces relations, on en déduit :

$$\begin{pmatrix} L_{X_c}^T \\ L_{Y_c}^T \\ L_l^T \\ L_\alpha^T \end{pmatrix} = \begin{pmatrix} 1/2 & 0 & 1/2 & 0 \\ 0 & 1/2 & 0 & 1/2 \\ \Delta X/l & \Delta Y/l & -\Delta X/l & -\Delta Y/l \\ -\Delta Y/l^2 & \Delta X/l^2 & \Delta Y/l^2 & -\Delta X/l^2 \end{pmatrix} \begin{pmatrix} L_{X_1}^T \\ L_{Y_1}^T \\ L_{X_2}^T \\ L_{Y_2}^T \end{pmatrix} \tag{2.38}$$

où $\Delta X = X_1 - X_2$ et $\Delta Y = Y_1 - Y_2$.

On obtient ainsi les torseurs d'interaction associés à $\underline{P'}$ exprimés en fonction de $\underline{P}$. En utilisant les relations suivantes :

$$\left\{ \begin{array}{ll}
X_1 = X_c + l\cos\alpha/2, & Y_1 = Y_c + l\sin\alpha/2 \\
X_2 = X_c - l\cos\alpha/2, & Y_2 = Y_c - l\sin\alpha/2
\end{array} \right. \tag{2.39}$$

on obtient les torseurs d'interaction associés à $\underline{P'}$, et, cette fois-ci, exprimés en fonction de $\underline{P'}$:

$$L_{X_c}^T = \begin{bmatrix} -\lambda_2 & 0 \\ \lambda_2 X_c - \lambda_1 l\cos\alpha/4 & X_c Y_c + l^2\cos\alpha\sin\alpha/4 \\ -(1+X_c^2+l^2\cos^2\alpha/4) & Y_c \end{bmatrix}$$

$$L_{Y_c}^T = \begin{bmatrix} 0 & -\lambda_2 \\ \lambda_2 Y_c - \lambda_1 l\sin\alpha/4 & 1+Y_c^2+l^2\sin^2\alpha/4 \\ -X_c Y_c - l^2\cos\alpha\sin\alpha/4 & -X_c \end{bmatrix}$$

$$L_l^T = \begin{bmatrix} \lambda_1\cos\alpha & \lambda_1\sin\alpha \\ \lambda_2 l - \lambda_1(X_c\cos\alpha + Y_c\sin\alpha) & l[X_c\cos\alpha\sin\alpha + Y_c(1+\sin^2\alpha)] \\ -l[X_c(1+\cos^2\alpha) + Y_c\cos\alpha\sin\alpha] & 0 \end{bmatrix}$$

$$L_\alpha^T = \begin{bmatrix} -\lambda_1\sin\alpha/l & \lambda_1\cos\alpha/l \\ \lambda_1(X_c\sin\alpha - Y_c\cos\alpha)/l & -X_c\sin^2\alpha + Y_c\cos\alpha\sin\alpha \\ X_c\cos\alpha\sin\alpha - Y_c\cos^2\alpha & -1 \end{bmatrix} \quad (2.40)$$

avec $\lambda_1 = (z_1 - z_2)/z_1 z_2$ et $\lambda_2 = (z_1 + z_2)/2z_1 z_2$.

De la même manière, si l'on considère un triangle, on peut le représenter :

- soit par les coordonnées dans l'image de ses trois sommets,
- soit par les coordonnées de son centre de gravité, la longueur de ses arêtes et l'orientation d'une de ses arêtes ou d'un de ses axes d'inertie.

Le calcul du torseur d'interaction de la seconde paramétrisation s'effectue exactement par la même méthode que dans le cas du segment. Enfin, rappelons qu'il est possible de calculer le torseur d'interaction de toute autre information visuelle s basée sur les points (surface, centre de gravité d'un polygone, etc...) du moment que cette information puisse s'exprimer sous la forme $s = \sigma(X_1, \cdots, X_n, Y_1, \cdots, Y_n)$.

2.3.2 Les droites

2.3.2.1 Représentation par les coordonnées pluckeriennes

Une droite de l'espace euclidien e est souvent représentée par les deux vecteurs $\underline{u} = (u_1, u_2, u_3)$, de norme 1, et $\underline{l} = (l_1, l_2, l_3)$ tels que, quel que soit un point m,

de coordonnées $\underline{x}$, appartenant à cette droite (voir Figure 2.3) :

$$\underline{x} \times \underline{u} = \underline{l} \tag{2.41}$$

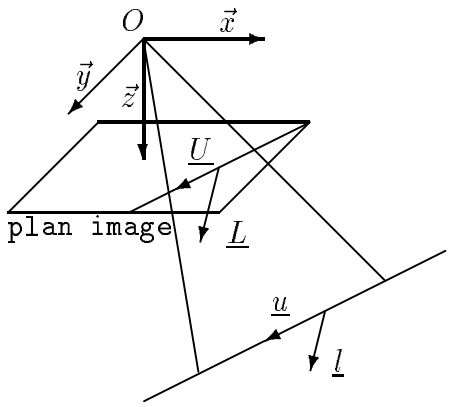


Figure 2.3 : Représentation des droites par les coordonnées pluckeriennes

Cette représentation par les coordonnées pluckeriennes $\underline{p} = (\underline{u}, \underline{l})$ est complète mais n'est pas minimale. Cependant, les vecteurs $\underline{u}$ et $\underline{l}$ sont liés par les contraintes $\|\underline{u}\| = 1$ et $\underline{u}.\underline{l} = 0$ aussi est-il possible de calculer $\frac{\partial \underline{p}}{\partial \bar{r}}$. On a (voir [Espiau 87a]) :

$$\begin{cases} \dot{\underline{u}} = \Omega \times \underline{u} \\ \dot{\underline{l}} = V \times \underline{u} + \Omega \times \underline{l} \end{cases} \tag{2.42}$$

ce qui nous permet d'écrire :

$$\frac{\partial \underline{p}}{\partial \bar{r}} = \begin{pmatrix} 0 & 0 & 0 & 0 & u_3 & -u_2 \\ 0 & 0 & 0 & -u_3 & 0 & u_1 \\ 0 & 0 & 0 & u_2 & -u_1 & 0 \\ 0 & u_3 & -u_2 & 0 & l_3 & -l_2 \\ -u_3 & 0 & u_1 & -l_3 & 0 & l_1 \\ u_2 & -u_1 & 0 & l_2 & -l_1 & 0 \end{pmatrix} \tag{2.43}$$

Soit $\underline{U}$ et $\underline{L}$ les coordonnées pluckeriennes de la droite 2D, résultant de la projection dans l'image de la droite 3D. On a :

$$\underline{X} \times \underline{U} = \underline{L} \tag{2.44}$$

Les propriétés de la projection perspective, s'écrivant ici $\underline{X} \times \underline{x} = 0$, donnent :

$$\underline{U} = \begin{pmatrix} l_2/\Delta \\ -l_1/\Delta \\ 0 \end{pmatrix} \quad , \quad \underline{L} = \begin{pmatrix} l_1/\Delta \\ l_2/\Delta \\ l_3/\Delta \end{pmatrix} \tag{2.45}$$

avec $\Delta = \sqrt{l_1^2 + l_2^2}$.

La représentation des droites de l'image par les coordonnées pluckeriennes n'est pas minimale. On a en effet $U_1 = L_2$, $U_2 = -L_1$ et $L_1^2 + L_2^2 = 1$. Calculons cependant $\frac{\partial \underline{L}}{\partial \underline{p}}$ (le calcul de $\frac{\partial \underline{U}}{\partial \underline{p}}$ est inutile puisque U est entièrement déterminé par L). On a :

$$\frac{\partial \underline{L}}{\partial \underline{p}} = \begin{pmatrix} 0 & 0 & 0 & L_2^2/\Delta & -L_1L_2/\Delta & 0 \\ 0 & 0 & 0 & L_1L_2/\Delta & L_1^2/\Delta & 0 \\ 0 & 0 & 0 & -L_1L_3/\Delta & -L_2L_3/\Delta & 1/\Delta \end{pmatrix} \tag{2.46}$$

On en déduit :

$$\begin{aligned} L_{L_1}^T &= \begin{bmatrix} L_1L_2u_3/\Delta & L_2^2u_3/\Delta & -(L_1L_2u_1 + L_2^2u_2)/\Delta \\ L_1L_2L_3 & L_2^2L_3 & -L_2 \end{bmatrix} \\ L_{L_2}^T &= \begin{bmatrix} -L_1^2u_3/\Delta & -L_1L_2u_3/\Delta & (L_1^2u_1 + L_1L_2u_2)/\Delta \\ L_1^2L_3 & -L_1L_2L_3 & L_1 \end{bmatrix} \\ L_{L_3}^T &= \begin{bmatrix} (u_2 + L_2L_3u_3)/\Delta & -(u_1 + L_1L_3u_3)/\Delta & (-L_2L_3u_1 + L_1L_3u_2)/\Delta \\ L_2(1 + L_3^2) & -L_1(1 + L_3^2) & 0 \end{bmatrix} \end{aligned} \tag{2.47}$$

On vérifie aisément que les torseurs d'interaction associés à L_1 et L_2 ne sont pas indépendants (puisque $L_1^2 + L_2^2 = 1$).

Dans le paragraphe suivant, nous allons étudier deux paramétrages bien connus des droites 2D, qui ont, eux, l'avantage d'être minimaux.

2.3.2.2 Autres représentations

On représente à présent les droites de e comme l'intersection de deux plans. On a :

$$h(\underline{x}, \underline{p}) = \begin{cases} a_1x + b_1y + c_1z + d_1 = 0 \\ a_2x + b_2y + c_2z + d_2 = 0 \end{cases} \tag{2.48}$$

Cette représentation n'est absolument pas minimale (quatre paramètres sont suffisants pour représenter une droite de e), mais elle a l'avantage d'être complète et ne nécessite donc pas l'emploi de plusieurs cartes (le lecteur intéressé par les différentes représentations des droites pourra, par exemple, se reporter à [Ayache 88]).

Rappelons que l'on ne s'intéresse pas au cas dégénéré où la droite passe par le centre de projection ($d_1 = d_2 = 0$). En utilisant les équations de la projection perspective (2.10), on obtient immédiatement :

- la fonction $\mu(\underline{X}, \underline{p})$, définie par l'équation (2.25), à partir de h_1 ou h_2 :

$$1/z = -(a_i X + b_i Y + c_i)/d_i \tag{2.49}$$

avec $i = 1$ si $d_1 \neq 0$ ou $i = 2$ si $d_2 \neq 0$,

- l'équation de la droite 2D, notée $\mathcal{D}$, résultant de la projection dans l'image de la droite 3D :

$$AX + BY + C = 0 \text{ avec } \begin{cases} A = a_1 d_2 - a_2 d_1 \\ B = b_1 d_2 - b_2 d_1 \\ C = c_1 d_2 - c_2 d_1 \end{cases} \tag{2.50}$$

La paramétrisation (A, B, C) n'est pas minimale, c'est pourquoi il nous faut choisir une autre représentation des droites 2D, afin de pouvoir calculer la matrice d'interaction associée.

2.3.2.2.1 Représentation (a, b) :

Cette représentation est minimale, mais nécessite l'emploi de deux cartes :

- Carte 1 : $Y = aX + b$ $(a = -A/B, b = -C/B)$, qui ne peut pas représenter les droites parallèles à l'axe $\vec{y}$ $(B = 0)$,
- Carte 2 : $X = aY + b$ $(a = -B/A, b = -C/A)$, qui ne peut pas représenter les droites parallèles à l'axe $\vec{x}$ $(A = 0)$.

Il est possible de calculer le torseur d'interaction associé à chaque carte. Intéressons nous tout d'abord à la première carte. On a :

$$g(\underline{X}, \underline{P}) = Y - aX - b = 0 \tag{2.51}$$

L'équation (2.29) s'écrit dans le cas présent sous la forme :

$$\dot{a}X + \dot{b} = \dot{Y} - a\dot{X} \ , \ \forall (X, Y) \in \mathcal{D} \tag{2.52}$$

Pour calculer $\dot{a}$ et $\dot{b}$, utilisons la méthode décrite au paragraphe 2.2.3 nous ramenant à la résolution d'un système linéaire de la forme (2.33). Choisissons donc deux points de $\mathcal{D}$, par exemple les points de coordonnées $\underline{X}_1 = (0, b)$ et $\underline{X}_2 = (1, a + b)$. On a alors :

$$\begin{cases} \dot{b} &= \dot{Y}_1 - a\dot{X}_1 \\ \dot{a} + \dot{b} &= \dot{Y}_2 - a\dot{X}_2 \end{cases} \tag{2.53}$$

où $\dot{X}_1, \dot{Y}_1, \dot{X}_2$ et $\dot{Y}_2$ sont donnés par les équations d'Optic Flow :

$$\begin{array}{rcccccccc}
\dot{X}_1 = [& -1/z_1 & 0 & 0 & 0 & -1 & b &] T \\
\dot{Y}_1 = [& 0 & -1/z_1 & b/z_1 & 1+b^2 & 0 & 0 &] T \\
\dot{X}_2 = [& -1/z_2 & 0 & 1/z_2 & a+b & -2 & a+b &] T \\
\dot{Y}_2 = [& 0 & -1/z_2 & (a+b)/z_2 & 1+(a+b)^2 & -(a+b) & -1 &] T
\end{array} \tag{2.54}$$

et où $1/z_1$ et $1/z_2$ sont obtenus à partir de l'équation (2.49) :

$$\begin{cases} 1/z_1 = -(b_i b + c_i)/d_i \\ 1/z_2 = -(a_i + b_i(a+b) + c_i)/d_i \end{cases} \tag{2.55}$$

On obtient enfin :

$$\begin{array}{rcccccc}
L_b^T = [& \lambda_1 a & -\lambda_1 & \lambda_1 b & 1+b^2 & a & -ab &] \\
L_a^T = [& \lambda_2 a & -\lambda_2 & \lambda_2 b & ab & -b & -1-a^2 &]
\end{array} \tag{2.56}$$

avec $\lambda_1 = 1/z_1 = -(b_i b + c_i)/d_i$ et $\lambda_2 = 1/z_2 - 1/z_1 = -(a_i + b_i a)/d_i$.

On peut appliquer exactement la même méthode pour calculer la matrice d'interaction associée à la seconde carte :

$$g(\underline{X}, \underline{P}) = X - aY - b = 0 \tag{2.57}$$

en prenant, par exemple, les points de coordonnées $\underline{X}_1 = (b, 0)$ et $\underline{X}_2 = (a+b, 1)$. On obtient alors :

$$\begin{array}{rcccccc}
L_b^T = [& -\lambda_1 & \lambda_1 a & \lambda_1 b & -a & -1-b^2 & ab &] \\
L_a^T = [& -\lambda_2 & \lambda_2 a & \lambda_2 b & b & -ab & 1+a^2 &]
\end{array} \tag{2.58}$$

avec $\lambda_1 = 1/z_1 = -(a_i b + c_i)/d_i$ et $\lambda_2 = 1/z_2 - 1/z_1 = -(a_i a + b_i)/d_i$.

Le passage d'une carte à l'autre, parfois nécessaire, peut entraîner des difficultés en commande. En effet, l'erreur $s - s^*$, quelle que soit l'information visuelle définie à partir de la représentation (a, b), n'a aucune signification si s ne peut se calculer qu'à l'aide de la carte 1 (droite horizontale) et s^* à l'aide de la carte 2 (droite verticale). En utilisant cette représentation, il est donc impossible d'amener l'image d'une droite verticale sur une droite horizontale (voir Figure 2.4).

Aussi préfère-t-on utiliser une représentation des droites complète (au sens où un seul paramétrage est nécessaire pour représenter l'ensemble des droites), telle que celle que nous détaillons à présent.

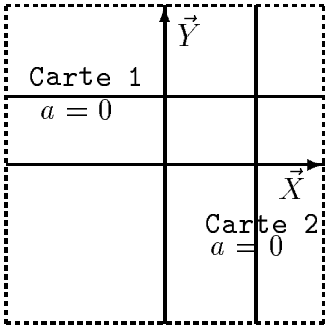


Figure 2.4 : Passage obligé de la carte 1 à la carte 2

2.3.2.2.2 Représentation (ρ, θ) :

L'équation d'une droite $\mathcal{D}$ est donnée, en utilisant cette représentation, par (voir Figure 2.5) :

$$g(\underline{X}, \underline{P}) = X \cos\theta + Y \sin\theta - \rho = 0 \tag{2.59}$$

où $\theta = \arctan(B/A)$ et $\rho = -C/\sqrt{A^2 + B^2}$.

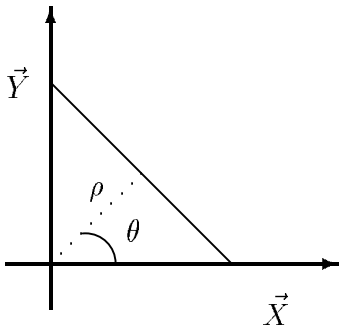
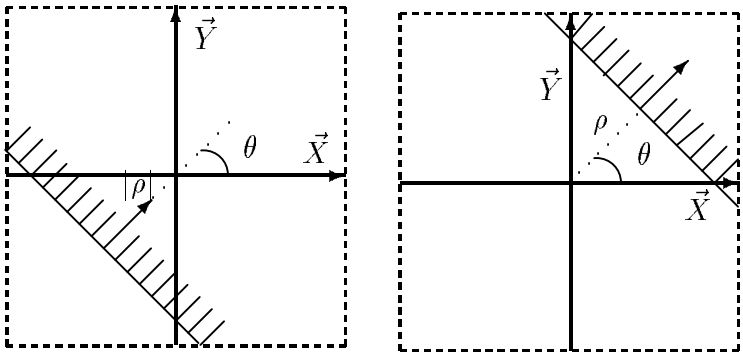


Figure 2.5 : Représentation (ρ, θ) des droites 2D

L'ambiguïté de cette représentation (la même droite peut être paramétrée indifféremment par $(\rho, \theta \pm 2k\pi)$ et $(-\rho, \theta \pm (2k+1)\pi)$) est levée en partie en choisissant pour θ l'orientation entre l'axe $\vec{X}$ et la normale orientée dans le sens de la diminution du gradient des points contours constituant la droite (voir Figure 2.6). Cette convention permet de fixer le signe de ρ et d'éviter ainsi certaines aberrations qui pourraient se produire en commande. De plus, l'ambiguïté dûe aux multiples choix possibles de θ, $(\theta \pm 2k\pi)$ d'une part, n'a aucune influence sur la valeur des torseurs d'interaction associés à ρ et θ et, d'autre part, peut facilement être levée en modulant à 2π l'erreur $\theta - \theta^*$ si θ est choisie comme information visuelle.



$\rho < 0$ $\rho > 0$

Figure 2.6 : Convention pour choisir le signe de ρ

Calculons à présent les torseurs d'interaction correspondant à cette représentation. On pourrait très bien utiliser la méthode précédente, ou le difféomorphisme qui permet de passer de (a, b) à (ρ, θ), mais, on préfère ici employer la méthode suivante :

L'équation (2.29) s'écrit maintenant sous la forme :

$$\dot{\rho} + (X \sin\theta - Y \cos\theta)\, \dot{\theta} = \dot{X} \cos\theta + \dot{Y} \sin\theta \ , \ \forall (X, Y) \in \mathcal{D} \tag{2.60}$$

A partir de l'équation (2.59), on écrit X en fonction de Y si $\cos\theta \neq 0$ (ou Y en fonction de X sinon) et l'équation (2.60) peut alors s'écrire, en utilisant (2.13) et (2.49) :

$$(-\dot{\theta}/\cos\theta)\, Y + (\dot{\rho} + \rho \tan\theta\ \dot{\theta}) = Y\ K_1(\underline{p}, \underline{P})\, T + K_2(\underline{p}, \underline{P})\, T \ , \ \forall Y \in \mathbb{R} \tag{2.61}$$

On en déduit aussitôt :

$$\left\{ \begin{array}{lcl} \dot{\theta} & = & -K_1(\underline{p}, \underline{P})\ \cos\theta\, T \\ \dot{\rho} & = & (K_2(\underline{p}, \underline{P}) + K_1(\underline{p}, \underline{P}) \rho \sin\theta)\ T \end{array} \right. \tag{2.62}$$

d'où :

$$\begin{array}{l} L_\rho^T = [\ \ \lambda_\rho \cos\theta \quad \lambda_\rho \sin\theta \quad -\lambda_\rho \rho \quad (1+\rho^2)\sin\theta \quad -(1+\rho^2)\cos\theta \quad 0 \quad] \\ L_\theta^T = [\ \ \lambda_\theta \cos\theta \quad \lambda_\theta \sin\theta \quad -\lambda_\theta \rho \quad -\rho\cos\theta \quad -\rho\sin\theta \quad -1 \quad] \end{array} \tag{2.63}$$

$$\begin{array}{lllll} \text{avec} & \lambda_\rho & = & (a_i \rho \cos\theta + b_i \rho \sin\theta + c_i)/d_i & \\ & & = & [(c_2 a_1 - c_1 a_2)\cos\theta + (c_2 b_1 - c_1 b_2)\sin\theta]/\sqrt{A^2 + B^2} & \\ \text{et} & \lambda_\theta & = & (a_i \sin\theta - b_i cos\theta)/d_i & \\ & & = & (a_2 b_1 - a_1 b_2)/\sqrt{A^2 + B^2} & \end{array}$$

2.3.3 Les cercles

Considérons maintenant le cas d'un cercle dans l'espace 3D. Il peut être représenté comme l'intersection d'une sphère et d'un plan coupant la sphère en son centre. On a alors :

$$h(\underline{x},\underline{p}) = \begin{cases} h_1 = (x-x_0)^2 + (y-y_0)^2 + (z-z_0)^2 - r^2 = 0 \\ h_2 = \alpha(x-x_0) + \beta(y-y_0) + \gamma(z-z_0) = 0 \end{cases} \tag{2.64}$$

La fonction $\mu(\underline{X},\underline{p})$, nécessaire pour le calcul des torseurs d'interaction, s'obtient facilement à partir de h_2 en utilisant l'équation de la projection perspective (2.10) :

$$1/z = aX + bY + c \quad \text{avec} \begin{cases} a &= \alpha/(\alpha x_0 + \beta y_0 + \gamma z_0) \\ b &= \beta/(\alpha x_0 + \beta y_0 + \gamma z_0) \\ c &= \gamma/(\alpha x_0 + \beta y_0 + \gamma z_0) \\ &= (1 - a x_0 - b y_0)/z_0 \end{cases} \tag{2.65}$$

et, toujours à l'aide de (2.10), on obtient, en injectant l'équation précédente dans h_1, l'équation de la projection dans l'image d'un cercle. Cette équation peut s'écrire , après quelques développements, sous la forme :

$$K_0X^2 + K_1Y^2 + 2K_2XY + 2K_3X + 2K_4Y + K_5 = 0 \tag{2.66}$$

$$\text{avec} \begin{cases} K_0 &= a^2(x_0^2+y_0^2+z_0^2-r^2) + 1 - 2ax_0 \\ K_1 &= b^2(x_0^2+y_0^2+z_0^2-r^2) + 1 - 2by_0 \\ K_2 &= ab(x_0^2+y_0^2+z_0^2-r^2) - bx_0 - ay_0 \\ K_3 &= ac(x_0^2+y_0^2+z_0^2-r^2) - cx_0 - az_0 \\ K_4 &= bc(x_0^2+y_0^2+z_0^2-r^2) - cy_0 - bz_0 \\ K_5 &= c^2(x_0^2+y_0^2+z_0^2-r^2) + 1 - 2cz_0 \end{cases}$$

L'image d'un cercle est donc généralement une ellipse (un segment dans les cas dégénérés que nous n'étudierons pas). On peut remarquer que cette ellipse se transforme en cercle lorsque $K_0 = K_1$ et $K_2 = 0$, soit, quel que soit le point (x_0, y_0, z_0) :

$$\begin{cases} a = b = 0 \\ \text{ ou} \\ a = 2x_0/(x_0^2+y_0^2+z_0^2-r^2)\ , \\ b = 2y_0/(x_0^2+y_0^2+z_0^2-r^2) \end{cases} \tag{2.67}$$

Ainsi, il existe deux orientations possibles (sauf si $x_0 = y_0 = 0$) pour que l'image d'un cercle soit un cercle, la première, bien connue, correspondant au cas où le plan du cercle 3D est parallèle au plan image, la seconde paraissant moins triviale...

L'équation (2.66) peut se factoriser sous la forme :

$$\frac{(X - X_c + E(Y - Y_c))^2}{A^2(1+E^2)} + \frac{(Y - Y_c - E(X - X_c))^2}{B^2(1+E^2)} - 1 = 0 \qquad (2.68)$$

avec

$$\begin{aligned}
X_c &= (K_1K_3 - K_2K_4)/(K_2^2 - K_0K_1) \\
Y_c &= (K_0K_4 - K_2K_3)/(K_2^2 - K_0K_1) \\
E &= (K_1 - K_0 \pm \sqrt{(K_1 - K_0)^2 + 4K_2^2})/2K_2 \qquad (2.69) \\
A^2 &= 2(K_0X_c^2 + 2K_2X_cY_c + K_1Y_c^2 - K_5)/(K_0 + K_1 \pm \sqrt{(K_1 - K_0)^2 + 4K_2^2}) \\
B^2 &= 2(K_0X_c^2 + 2K_2X_cY_c + K_1Y_c^2 - K_5)/(K_0 + K_1 \mp \sqrt{(K_1 - K_0)^2 + 4K_2^2})
\end{aligned}$$

On reconnaît là la paramétrisation naturelle des ellipses, notée $\underline{P}_0$, où X_c, Y_c, A, B et E représentent respectivement les coordonnées du centre, les axes et l'orientation de l'ellipse (voir Figure 2.7). On peut remarquer que la projection du centre du cercle $(x_0/z_0, y_0/z_0)$ ne correspond pas au centre (X_c, Y_c) de l'ellipse sauf dans le cas où les plans du cercle et de l'image sont parallèles ($a = b = 0$).

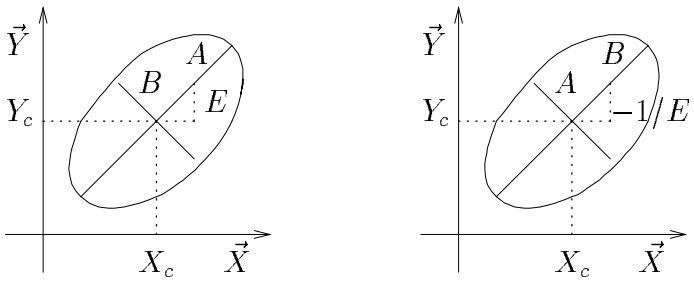


Figure 2.7 : Ambiguïté de la représentation naturelle des ellipses

De plus, cette représentation est toujours ambiguë. En effet, les deux ensembles de paramètres (X_c, Y_c, A, B, E) et (X_c, Y_c, B, A, $-1/E$) représentent toujours la même ellipse (voir Figure 2.7). Aussi ne paraît-il pas judicieux d'utiliser cette paramétrisation en commande en raison des problèmes de choix qu'il faudrait résoudre pour chaque calcul de $s - s^*$.

Enfin, il faut noter que lorsque l'ellipse se transforme en cercle, l'orientation E n'est plus définie et il est alors impossible de calculer le torseur d'interaction correspondant.

Nous sommes donc amenés à chercher d'autres paramétrages des ellipses, également minimaux, mais, qui plus est, complets et non ambigus. On peut choisir :

- les paramètres obtenus à partir de la quadrique (2.66) :
  $\underline{P_1} = (A_1, A_2, A_3, A_4, A_5)$, où $A_i = K_i/K_0$, K_0 étant toujours différent de 0 pour les cas non dégénérés. Le passage de cette représentation à $\underline{P_0}$ est donné par les équations suivantes :

$$\left\{\begin{array}{l} A_1 = (A^2 + B^2E^2)/(A^2E^2 + B^2) \\ A_2 = (B^2 - A^2)E/(A^2E^2 + B^2) \\ A_3 = -(X_c + A_2Y_c) \\ A_4 = -(A_2X_c + A_1Y_c) \\ A_5 = (X_c^2 + 2A_2X_cY_c + A_1Y_c^2) - A^2B^2(1 + E^2)/(A^2E^2 + B^2) \end{array}\right. \tag{2.70}$$

  On peut ainsi vérifier que cette représentation n'est pas ambiguë et qu'elle est complète : dans le cas du cercle, on a $A_1 = 1$, $A_2 = 0$, $A_3 = -X_c$, $A_4 = -Y_c$ et $A_5 = X_c^2 + Y_c^2 - R^2$ si R est le rayon de ce cercle.

- les paramètres définis à partir des moments $m_{ij} = \sum_{X\in\mathcal{E}}\sum_{Y\in\mathcal{E}} X^iY^j$ d'ordre inférieur à 3 ($i + j \leq 2$). On peut en effet calculer :

$$\left\{\begin{array}{l} m_{00} = \pi AB\ ,\ m_{10} = \pi ABX_c\ ,\ m_{01} = \pi ABY_c \\ m_{20} = \pi AB(A^2 + B^2E^2)/4(1 + E^2) + \pi ABX_c^2 \\ m_{11} = \pi ABE(A^2 - B^2)/4(1 + E^2) + \pi ABX_cY_c \\ m_{02} = \pi AB(A^2E^2 + B^2)/4(1 + E^2) + \pi ABY_c^2 \end{array}\right. \tag{2.71}$$

  On choisit donc comme seconde représentation des ellipses les paramètres $\underline{P_2} = (X_c, Y_c, \mu_{20}, \mu_{11}, \mu_{02})$ où :

$$\left\{\begin{array}{l} X_c = m_{10}/m_{00} \\ Y_c = m_{01}/m_{00} \\ \mu_{20} = 4(m_{20} - m_{00}X_c^2)/m_{00} = (A^2 + B^2E^2)/(1 + E^2) \\ \mu_{11} = 4(m_{11} - m_{00}X_cY_c)/m_{00} = E(A^2 - B^2)/(1 + E^2) \\ \mu_{02} = 4(m_{02} - m_{00}Y_c^2)/m_{00} = (A^2E^2 + B^2)/(1 + E^2) \end{array}\right. \tag{2.72}$$

  Cette représentation par les moments, dont les paramètres sont fonction des paramètres $\underline{P_0}$, est bien, comme la précédente, non ambiguë et complète (dans le cas où l'ellipse se transforme en cercle, on a à présent $\mu_{20} = \mu_{02} = R^2$ et $\mu_{11} = 0$). Cette représentation a en plus l'avantage de conserver un aspect "naturel" et s'obtient en outre très facilement à partir d'une image numérisée.

Les différents changements de représentation ainsi que les calculs des matrices d'interaction correspondant à ces deux paramétrages ont été renvoyés en Annexe à la fin de ce chapitre. On en donne ici les résultats finaux :

- Pour le paramétrage $\underline{P}_1$:

$$
\begin{array}{lccc}
L_{A_1}^T = [& 2bA_2 - 2aA_1 & 2A_1(b - aA_2) & 2bA_4 - 2aA_1A_3 \\
 & 2A_4 & 2A_1A_3 & -2A_2(A_1 + 1) \quad] \\
L_{A_2}^T = [& b - aA_2 & bA_2 - a(2A_2^2 - A_1) & a(A_4 - 2A_2A_3) + bA_3 \\
 & A_3 & 2A_2A_3 - A_4 & A_1 - 2A_2^2 - 1 \quad] \\
L_{A_3}^T = [& c - aA_3 & a(A_4 - 2A_2A_3) + cA_2 & cA_3 - a(2A_3^2 - A_5) \\
 & -A_2 & 1 + 2A_3^2 - A_5 & A_4 - 2A_2A_3 \quad] \\
L_{A_4}^T = [& A_3b + A_2c - 2aA_4 & A_4b + A_1c - 2aA_2A_4 & bA_5 + cA_4 - 2aA_3A_4 \\
 & A_5 - A_1 & 2A_3A_4 + A_2 & -2A_2A_4 - A_3 \quad] \\
L_{A_5}^T = [& 2cA_3 - 2aA_5 & 2cA_4 - 2aA_2A_5 & 2cA_5 - 2aA_3A_5 \\
 & -2A_4 & 2A_3A_5 + 2A_3 & -2A_2A_5 \quad]
\end{array}
\tag{2.73}
$$

Il faut noter que la matrice d'interaction L^T associée à ce paramétrage est toujours de rang 5, excepté dans le cas où la projection du cercle 3D est un cercle centré dans l'image. On a alors :

$$
a = b = x_0 = y_0 = 0 \;\Leftrightarrow\; A_1 = 1, A_2 = A_3 = A_4 = 0, A_5 = -R^2 \tag{2.74}
$$

et la matrice L^T s'écrit :

$$
L^T = \begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
1/z_0 & 0 & 0 & 0 & 1+R^2 & 0 \\
0 & 1/z_0 & 0 & -(1+R^2) & 0 & 0 \\
0 & 0 & -2R^2/z_0 & 0 & 0 & 0
\end{pmatrix} \tag{2.75}
$$

Les vecteurs L_{A_1} et L_{A_2} sont donc nuls et la matrice L^T est de rang 3. Il s'agit là d'une singularité (au sens perte de rang) isolée puisque la matrice L^T est de rang 5, même dans les autres cas où le cercle 3D se projette dans l'image sous la forme d'un cercle. Cette perte de rang implique en particulier qu'il est impossible de construire une loi de commande capable d'amener l'image d'un cercle sur le cercle centré, et ce, en utilisant des informations visuelles basées uniquement sur un cercle. Nous y reviendrons dans le chapitre suivant.

• Pour le paramétrage $\underline{P}_2$:

$$L^T_{X_c} = [\begin{array}{ccc} -1/z_c & 0 & X_c/z_c + a\mu_{20} + b\mu_{11} \\ X_cY_c + \mu_{11} & -1 - X_c^2 - \mu_{20} & Y_c \end{array}]$$

$$L^T_{Y_c} = [\begin{array}{ccc} 0 & -1/z_c & Y_c/z_c + a\mu_{11} + b\mu_{02} \\ 1 + Y_c^2 + \mu_{02} & -X_cY_c - \mu_{11} & -X_c \end{array}]$$

$$L^T_{\mu_{20}} = [\begin{array}{ccc} -2(a\mu_{20} + b\mu_{11}) & 0 & 2[(1/z_c + aX_c)\mu_{20} + bX_c\mu_{11}] \\ 2(Y_c\mu_{20} + X_c\mu_{11}) & -4\mu_{20}X_c & 2\mu_{11} \end{array}]$$

$$L^T_{\mu_{11}} = [\begin{array}{ccc} -a\mu_{11} - b\mu_{02} & -a\mu_{20} - b\mu_{11} & aY_c\mu_{20} + (3/z_c - c)\mu_{11} + bX_c\mu_{02} \\ 3Y_c\mu_{11} + X_c\mu_{02} & -Y_c\mu_{20} - 3X_c\mu_{11} & \mu_{02} - \mu_{20} \end{array}]$$

$$L^T_{\mu_{02}} = [\begin{array}{ccc} 0 & -2(a\mu_{11} + b\mu_{02}) & 2[(1/z_c + bY_c)\mu_{02} + aY_c\mu_{11}] \\ 4Y_c\mu_{02} & -2(Y_c\mu_{11} + X_c\mu_{02}) & -2\mu_{11} \end{array}] \qquad (2.76)$$

où $z_c = 1/(aX_c + bY_c + c)$.

Pour cette représentation également, la matrice L^T est toujours de rang 5, excepté dans le cas du cercle centré où l'on observe la même perte de rang. La matrice L^T s'écrit alors en effet :

$$L^T = \begin{pmatrix} -1/z_0 & 0 & 0 & 0 & -1 - R^2 & 0 \\ 0 & -1/z_0 & 0 & 1 + R^2 & 0 & 0 \\ 0 & 0 & 2R^2/z_0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2R^2/z_0 & 0 & 0 & 0 \end{pmatrix} \qquad (2.77)$$

2.3.4 Les sphères

Etudions à présent le cas de la sphère, exemple de primitive volumique. On a :

$$h(\underline{x}, \underline{p}) = (x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2 - r^2 = 0 \qquad (2.78)$$

Calculons tout d'abord les fonctions $g(\underline{X}, \underline{P})$ et $\mu(\underline{X}, \underline{P})$. En utilisant l'équation (2.10) dans (2.78), on obtient :

$$(X - \frac{x_0}{z})^2 + (Y - \frac{y_0}{z})^2 + (1 - \frac{z_0}{z})^2 - \frac{R^2}{z^2} = 0 \qquad (2.79)$$

soit un polynôme du deuxième degré en $1/z$:

$$\frac{1}{z^2}\,(x_0^2 + y_0^2 + z_0^2 - R^2) - \frac{2}{z}\,(x_0X + y_0Y + z_0) + X^2 + Y^2 + 1 = 0 \qquad (2.80)$$

Les points appartenant au contour de l'image d'une sphère sont tels que l'intersection entre leur ligne de vue et la sphère soit unique. Dans le cas présent, cette propriété est équivalente au fait que le discriminant Δ de (2.80) soit nul. On a donc :

$$(x_0X + y_0Y + z_0)^2 - (x_0^2 + y_0^2 + z_0^2 - R^2)(X^2 + Y^2 + 1) = 0 \qquad (2.81)$$

qui s'écrit après reparamétrisation :

$$K_0X^2 + K_1Y^2 + 2K_2XY + 2K_3X + 2K_4Y + K_5 = 0 \qquad (2.82)$$

$$\text{avec} \begin{cases} K_0 &= R^2 - y_0^2 - z_0^2 \\ K_1 &= R^2 - x_0^2 - z_0^2 \\ K_2 &= x_0y_0 \\ K_3 &= x_0z_0 \\ K_4 &= y_0z_0 \\ K_5 &= R^2 - x_0^2 - y_0^2 \end{cases}$$

L'image d'une sphère est donc généralement une ellipse (un cercle dans le cas où $x_0 = y_0 = 0$). De plus, on peut remarquer, comme dans le cas précédent du cercle, que la projection du centre de la sphère ne correspond pas au centre de l'ellipse (excepté si $x_0 = y_0 = 0$).

Par ailleurs, la fonction μ est facilement obtenue à partir de la racine double de (2.80) correspondant à $\Delta = 0$ et on a :

$$1/z = aX + bY + c \text{ avec } \begin{cases} a = x_0/(x_0^2 + y_0^2 + z_0^2 - R^2) \\ b = y_0/(x_0^2 + y_0^2 + z_0^2 - R^2) \\ c = z_0/(x_0^2 + y_0^2 + z_0^2 - R^2) \end{cases} \qquad (2.83)$$

Le calcul des torseurs d'interaction associés à une ellipse a déjà été effectué dans le paragraphe précédent. Aussi les matrices d'interaction correspondant à la projection d'une sphère sont-ils simplement obtenus en remplaçant dans les équations (2.73) et (2.76) a, b et c par les valeurs particulières données en (2.83).

La matrice L^T est toujours de rang 3, quelle que soit la configuration de la sphère et la paramétrisation choisie. Trois paramètres sont donc suffisants pour caractériser l'image d'une sphère. On propose :

- Pour la représentation basée sur les coefficients A_i de la quadrique : les paramètres A_3, A_4 et A_5, car la matrice d'interaction associée est toujours de rang plein 3, alors que, si l'on utilise A_1 ou A_2, on a encore L_{A_1} et L_{A_2} nuls dans le cas où l'image de la sphère est un cercle.
- Pour la représentation basée sur les moments : les paramètres X_c, Y_c et, par exemple, $(\mu_{20} + \mu_{02})/2$.

2.3.5 Les cylindres

On peut définir un cylindre de rayon R comme l'ensemble des cercles de rayon R tels que leur centre appartienne à une droite $\mathcal{D}$ perpendiculaire au plan de ces cercles.

On rappelle qu'un cercle de centre $m_i = (x_i \ y_i \ z_i)^T$ peut s'écrire comme l'intersection d'une sphère avec un plan :

$$\begin{cases} (x-x_i)^2 + (y-y_i)^2 + (z-z_i)^2 - R^2 = 0 \\ a(x-x_i) + b(y-y_i) + c(z-z_i) = 0 \end{cases} \tag{2.84}$$

avec, par exemple, $a^2 + b^2 + c^2 = 1$.

La propriété d'appartenance du centre du cercle à la droite $\mathcal{D}$ s'écrit, si $m_0 = (x_0 \ y_0 \ z_0)^T$ est un point quelconque de $\mathcal{D}$:

$$\forall (x_i \ y_i \ z_i) \in D, \ \exists \lambda \in \mathbb{R} \text{ tel que } \begin{cases} x_i = x_0 + \lambda a \\ y_i = y_0 + \lambda b \\ y_i = y_0 + \lambda c \end{cases} \tag{2.85}$$

En remplaçant ces valeurs x_i, y_i et z_i dans les équations (2.84), on obtient :

$$\begin{cases} (x-x_0)^2 + (y-y_0)^2 + (z-z_0)^2 - R^2 \\ +\lambda^2 - 2\lambda[a(x-x_0) + b(y-y_0) + c(z-z_0)] = 0 \\ \\ \lambda = a(x-x_0) + b(y-y_0) + c(z-z_0) \end{cases} \tag{2.86}$$

On en déduit l'équation paramétrique d'un cylindre quelconque dans e :

$$\begin{aligned} h(\underline{x}, \underline{p}) \ = \ & (x-x_0)^2 + (y-y_0)^2 + (z-z_0)^2 \\ & -[a(x-x_0) + b(y-y_0) + c(z-z_0)]^2 - R^2 = 0 \end{aligned} \tag{2.87}$$

Calculons à présent la projection dans l'image de ce cylindre. Comme pour le cas de la sphère, en appliquant les équations de la projection perspective (2.10) sur l'équation (2.87), on obtient un polynôme de degré 2 en $1/z$:

$$A \ 1/z^2 - 2 \ B \ 1/z + C = 0 \tag{2.88}$$

$$\text{avec } \begin{cases} A = x_0^2 + y_0^2 + z_0^2 - (ax_0 + by_0 + cz_0)^2 - R^2 \\ B = \alpha_1 X + \beta_1 Y + \gamma_1 \\ C = X^2 + Y^2 + 1 - (aX + bY + c)^2 \end{cases}$$

$$\text{où } \begin{cases} \alpha_1 = (1-a^2)x_0 - aby_0 - acz_0 \\ \beta_1 = -abx_0 + (1-b^2)y_0 - bcz_0 \\ \gamma_1 = -acx_0 - bcy_0 + (1-c^2)z_0 \end{cases}$$

On peut remarquer que lorsque $A \leq 0$, le centre de projection de la caméra (de coordonnées $(0\ 0\ 0\ )^T$) appartient à l'enveloppe du cylindre ($A = 0$) ou à son intérieur ($A < 0$). Nous ne nous intéresserons pas à ces cas particuliers et considèrerons uniquement le cas où $A > 0$.

Les points de la projection dans l'image du cylindre qui nous intéressent sont les points appartenant aux contours de cette projection et sont caractérisés par le fait qu'il n'y a qu'un seul point d'intersection entre une ligne de vue et le cylindre. Cela peut s'écrire, comme dans le cas de la sphère, sous la forme :

$$\Delta' = B^2 - A\,C = 0 \tag{2.89}$$

On en déduit aussitôt $1/z = B/A$ qui fournit la fonction $\mu(\underline{X}, \underline{p})$:

$$\mu(\underline{X}, \underline{p}) : 1/z = \alpha_1/A\ X + \beta_1/A\ Y + \gamma_1/A \tag{2.90}$$

Par ailleurs, on peut facilement trouver après quelques développements :

$$\Delta' = R^2 C - (\alpha_2 X + \beta_2 Y + \gamma_2)^2 \text{ avec } \begin{cases} \alpha_2 = cy_0 - bz_0 \\ \beta_2 = az_0 - cx_0 \\ \gamma_2 = bx_0 - ay_0 \end{cases} \tag{2.91}$$

Lorsque $\Delta' = 0$ qui est le cas qui nous concerne, on a bien évidemment $C = B^2/A$, ce qui permet d'obtenir la factorisation :

$$\Delta' = [RB/\sqrt{A} - (\alpha_2 X + \beta_2 Y + \gamma_2)][RB/\sqrt{A} + (\alpha_2 X + \beta_2 Y + \gamma_2)] = 0 \tag{2.92}$$

La projection dans l'image d'un cylindre est donc caractérisée par les deux droites d'équations :

$$\begin{aligned} D_1 \ &: \ (R\alpha_1/\sqrt{A} - \alpha_2)X + (R\beta_1/\sqrt{A} - \beta_2)Y + (R\gamma_1/\sqrt{A} - \gamma_2) = 0 \\ D_2 \ &: \ (R\alpha_1/\sqrt{A} + \alpha_2)X + (R\beta_1/\sqrt{A} + \beta_2)Y + (R\gamma_1/\sqrt{A} + \gamma_2) = 0 \end{aligned}$$

- **Remarque :** Lorsque le rayon du cylindre est nul, on retrouve bien le fait que sa projection est une droite ($D_1 = D_2$).

En utilisant la représentation (ρ, θ) pour chacune de ces droites et les résultats déjà obtenus en (2.63), il est possible de calculer la matrice d'interaction associée à la projection d'un cylindre. On obtient si (ρ_1, θ_1) représente D_1 et si (ρ_2, θ_2) représente D_2 :

$$\begin{aligned} L_{\rho_1}^T &= [\ \lambda_{\rho_1}\cos\theta_1 \quad \lambda_{\rho_1}\sin\theta_1 \quad -\lambda_{\rho_1}\rho_1 \quad (1+\rho_1^2)\sin\theta_1 \quad -(1+\rho_1^2)\cos\theta_1 \quad 0\ ] \\ L_{\theta_1}^T &= [\ \lambda_{\theta_1}\cos\theta_1 \quad \lambda_{\theta_1}\sin\theta_1 \quad -\lambda_{\theta_1}\rho_1 \quad -\rho_1\cos\theta_1 \quad -\rho_1\sin\theta_1 \quad -1\ ] \\ L_{\rho_2}^T &= [\ \lambda_{\rho_2}\cos\theta_2 \quad \lambda_{\rho_2}\sin\theta_2 \quad -\lambda_{\rho_2}\rho_2 \quad (1+\rho_2^2)\sin\theta_2 \quad -(1+\rho_2^2)\cos\theta_2 \quad 0\ ] \\ L_{\theta_2}^T &= [\ \lambda_{\theta_2}\cos\theta_2 \quad \lambda_{\theta_2}\sin\theta_2 \quad -\lambda_{\theta_2}\rho_2 \quad -\rho_2\cos\theta_2 \quad -\rho_2\sin\theta_2 \quad -1\ ] \end{aligned} \tag{2.93}$$

avec $\lambda_{\rho_i} = -(\alpha_1 \rho_i \cos\theta_i + \beta_1 \rho_i sin\theta_i + \gamma_1)/A$
et $\lambda_{\theta_i} = (\beta_1 \cos\theta_i - \alpha_1 \sin\theta_i)/A$

Nous terminons ainsi l'étude des primitives géométriques qui nous ont paru les plus significatives. Les résultats, concernant les différentes représentations possibles des primitives étudiées, sont condensés dans le tableau (2.1) ci-dessous.

Primitives 3D	Primitives 2D	Représentations possibles
point	point	(X, Y)
segment	segment	(X_1, Y_1, X_2, Y_2) (X_c, Y_c, l, α)
droite	droite	$\underline{L}$ non minimale (a, b) incomplète (ρ, θ)
cercle	ellipse	$(A_1, A_2, A_3, A_4, A_5)$ singularité pour $(X_c, Y_c, \mu_{20}, \mu_{11}, \mu_{02})$ le cercle centré
sphère	ellipse	(A_3, A_4, A_5) $(X_c, Y_c, (\mu_{20} + \mu_{02})/2)$
cylindre	2 droites	$(\rho_1, \theta_1, \rho_2, \theta_2)$

Tableau 2.1 : Représentations des différentes primitives étudiées

Un lecteur intéressé pourra, sans trop de difficultés, étudier des primitives plus complexes (tores, intersection de plusieurs sphères, etc) et calculer les torseurs d'interaction associés en utilisant les méthodes décrites précédemment. Nous lui signalons que les calculs les plus pénibles de ce chapitre, notamment ceux qui sont décrits dans l'Annexe suivante, ont été réalisés avec l'aide du logiciel de calcul symbolique MACSYMA [MACSYMA 85].

2.4 Annexe : calcul des torseurs d'interaction pour l'ellipse

Nous allons dans cette annexe calculer les torseurs d'interaction associés à la représentation naturelle des ellipses $\underline{P}_0$, puis donner les changements de paramétrage permettant de calculer les torseurs d'interaction associés aux représentations $\underline{P}_1$ et $\underline{P}_2$ qui ont été choisies précédemment. On rappelle (voir équation (2.68)) qu'une ellipse $\mathcal{E}$ peut s'écrire sous la forme :

$$g(\underline{X}, \underline{P}_0) = \frac{(X - X_c + E(Y - Y_c))^2}{A^2(1 + E^2)} + \frac{(Y - Y_c - E(X - X_c))^2}{B^2(1 + E^2)} - 1 = 0 \quad (2.94)$$

En différentiant cette équation, on obtient :

$$\sum_{i=1}^{5} \frac{\partial g}{\partial P_{0_i}}(\underline{P}_0, X, Y)\dot{P_{0_i}} = -\frac{\partial g}{\partial X}(\underline{P}_0, X, Y)\ \dot{X} - \frac{\partial g}{\partial Y}(\underline{P}_0, X, Y)\ \dot{Y},\ \forall (X, Y) \in \mathcal{E} \quad (2.95)$$

Par ailleurs, $\dot{X}$ et $\dot{Y}$ sont donnés par les équations d'Optic Flow (2.13) où $1/z$ peut s'exprimer en fonction de X, Y et $\underline{p}$. En effet, on a (voir équation (2.65)) :

$$1/z = aX + bY + c\ ,\ \forall (X, Y) \in \mathcal{E} \quad (2.96)$$

L'équation (2.95) s'écrit donc :

$$\sum_{i=1}^{5} \frac{\partial g}{\partial P_{0_i}}(\underline{P}_0, X, Y)\dot{P_{0_i}} = [\lambda_1 \cdots \lambda_6]\ T\ ,\ \forall (X, Y) \in \mathcal{E} \quad (2.97)$$

où $\lambda_j, j = 1$ à 6, est fonction de X, Y, $\underline{p}$ et $\underline{P}_0$. Posons à présent :

$$\sin\theta = \frac{X - X_c + E(Y - Y_c)}{A\sqrt{1 + E^2}} \quad (2.98)$$

On en déduit :

$$\cos\theta = \frac{Y - Y_c - E(X - X_c)}{B\sqrt{1 + E^2}} \quad (2.99)$$

d'où :

$$\begin{cases} X = X_c + (A\sin\theta - BE\cos\theta)/\sqrt{1 + E^2} \\ X = Y_c + (AE\sin\theta + B\cos\theta)/\sqrt{1 + E^2} \end{cases} \quad (2.100)$$

Substituons ces valeurs dans l'équation (2.97), on obtient :

$$\sum_{i=1}^{5} \frac{\partial g}{\partial P_{0_i}}(\underline{P}_0, \cos\theta, \sin\theta)\dot{P_{0_i}} = [\lambda_1 \cdots \lambda_6]\ \ T\ ,\ \forall \theta \in \mathbb{R} \quad (2.101)$$

où les coefficients $\frac{\partial g}{\partial P_{0_i}}$ ont la forme :

$$\frac{\partial g}{\partial P_{0_i}} = \alpha_{1i} \sin^2\theta + \alpha_{2i} \sin\theta\cos\theta + \alpha_{3i} \sin\theta + \alpha_{4i} \cos\theta + \alpha_{5i} \tag{2.102}$$

et où les coefficients λ_j peuvent maintenant s'écrire :

$$\lambda_j = \beta_{1j} \sin^2\theta + \beta_{2j} \sin\theta\cos\theta + \beta_{3j} \sin\theta + \beta_{4j} \cos\theta + \beta_{5j} \tag{2.103}$$

Il suffit donc de résoudre le système linéaire à cinq équations et cinq inconnues :

$$\begin{pmatrix} \alpha_{11} & \cdots & \alpha_{15} \\ \alpha_{21} & \cdots & \alpha_{25} \\ \alpha_{31} & \cdots & \alpha_{35} \\ \alpha_{41} & \cdots & \alpha_{45} \\ \alpha_{51} & \cdots & \alpha_{55} \end{pmatrix} \begin{pmatrix} \dot{X}_c \\ \dot{Y}_c \\ \dot{E} \\ \dot{A} \\ \dot{B} \end{pmatrix} = \begin{pmatrix} \beta_{11} & \cdots & \beta_{16} \\ \beta_{21} & \cdots & \beta_{26} \\ \beta_{31} & \cdots & \beta_{36} \\ \beta_{41} & \cdots & \beta_{46} \\ \beta_{51} & \cdots & \beta_{56} \end{pmatrix} T \tag{2.104}$$

pour en déduire

$$\begin{pmatrix} \dot{X}_c \\ \dot{Y}_c \\ \dot{E} \\ \dot{A} \\ \dot{B} \end{pmatrix} = L^T_{\underline{P}_0}\, T \tag{2.105}$$

Nous ne donnons pas ici la matrice $L^T_{\underline{P}_0}$ sous sa forme littérale. Elle a une forme peu sympathique (!) et nous avons vu qu'il n'était pas souhaitable d'utiliser cette représentation ambiguë et incomplète. On peut toutefois signaler que L^T_E a la forme :

$$L^T_E = \frac{1}{A^2 - B^2} L'^T_E \tag{2.106}$$

et l'on retrouve bien le fait que, lorsque la projection du cercle est un cercle ($A = B$), E est indéfini.

- **Remarque :** on peut bien évidemment obtenir les mêmes résultats en choisissant cinq points indépendants appartenant à l'ellipse et en résolvant le système linéaire obtenu à partir des cinq équations fournies par (2.97). On peut, par exemple, choisir les points ayant pour coordonnées :

$$\begin{aligned}
(X_1, Y_1) &= (X_c + A/\sqrt{1+E^2}, Y_c + AE/\sqrt{1+E^2}) \\
(X_2, Y_2) &= (X_c - A/\sqrt{1+E^2}, Y_c - AE/\sqrt{1+E^2}) \\
(X_3, Y_3) &= (X_c - BE/\sqrt{1+E^2}, Y_c + B/\sqrt{1+E^2}) \\
(X_4, Y_4) &= (X_c + BE/\sqrt{1+E^2}, Y_c - B/\sqrt{1+E^2}) \\
(X_5, Y_5) &= (X_c + (A-BE)\sqrt{2(1+E^2)}, Y_c + (AE+B)\sqrt{2(1+E^2)})
\end{aligned} \tag{2.107}$$

Passons maintenant au calcul des torseurs d'interaction associés aux représentations $\underline{P}_1$ et $\underline{P}_2$. On rappelle que $\underline{P}_1 = (A_1, A_2, A_3, A_4, A_5)$ est obtenue à partir des coefficients de la quadrique (2.66) et qu'elle peut s'écrire en fonction de la représentation $\underline{P}_0$:

$$\left\{\begin{array}{l} A_1 = (A^2 + B^2E^2)/(A^2E^2 + B^2) \\ A_2 = (B^2 - A^2)E/(A^2E^2 + B^2) \\ A_3 = -(X_c + A_2Y_c) \\ A_4 = -(A_2X_c + A_1Y_c) \\ A_5 = (X_c^2 + 2A_2X_cY_c + A_1Y_c^2) - A^2B^2(1 + E^2)/(A^2E^2 + B^2) \end{array}\right. \tag{2.108}$$

Il en est de même de la représentation $\underline{P}_2 = (X_c, Y_c, \mu_{20}, \mu_{11}, \mu_{02})$ calculée à partir des moments d'ordre inférieur ou égal à deux :

$$\left\{\begin{array}{l} X_c = m_{10}/m_{00} \\ Y_c = m_{01}/m_{00} \\ \mu_{20} = 4(m_{20} - m_{00}X_c^2)/m_{00} = (A^2 + B^2E^2)/(1 + E^2) \\ \mu_{11} = 4(m_{11} - m_{00}X_cY_c)/m_{00} = E(A^2 - B^2)/(1 + E^2) \\ \mu_{02} = 4(m_{02} - m_{00}Y_c^2)/m_{00} = (A^2E^2 + B^2)/(1 + E^2) \end{array}\right. \tag{2.109}$$

Les applications réciproques sont données par :

$$\begin{array}{rcl} X_c & = & (A_1A_3 - A_2A_4)/(A_2^2 - A_1) \\ Y_c & = & (A_4 - A_2A_3)/(A_2^2 - A_1) \\ E & = & (A_1 - 1 \pm \sqrt{(A_1 - 1)^2 + 4A_2^2})/2A_2 \\ & = & (\mu_{02} - \mu_{20} \mp \sqrt{(\mu_{20} - \mu_{02})^2 + 4\mu_{11}^2})/2\mu_{11} \\ A^2 & = & 2(X_c^2 + 2A_2X_cY_c + A_1Y_c^2 - A_5)/(1 + A_1 \pm \sqrt{(A_1 - 1)^2 + 4A_2^2}) \\ & = & (\mu_{02} + \mu_{20} \mp \sqrt{(\mu_{20} - \mu_{02})^2 + 4\mu_{11}^2})/2 \\ B^2 & = & 2(X_c^2 + 2A_2X_cY_c + A_1Y_c^2 - A_5)/(1 + A_1 \mp \sqrt{(A_1 - 1)^2 + 4A_2^2}) \\ & = & (\mu_{02} + \mu_{20} \pm \sqrt{(\mu_{20} - \mu_{02})^2 + 4\mu_{11}^2})/2 \end{array} \tag{2.110}$$

Pour obtenir les résultats donnés en (2.73) et (2.76), il suffit alors, comme on l'a signalé précédemment, de calculer pour $k = 1$ ou 2 :

$$L^T(\underline{p}, \underline{P}_k) = \frac{\partial \underline{P}_k}{\partial \underline{P}_0}\, L^T(\underline{p}, \underline{P}_0) \tag{2.111}$$

où $\underline{P}_0$, présent dans le second membre de cette équation, est exprimé en fonction de $\underline{P}_k$ à l'aide des équations (2.110).

Chapitre 3

Classification des tâches référencées vision

Dans ce chapitre, nous examinons l'ensemble des torseurs réciproques du torseur d'interaction. Ceux-ci vont nous permettre d'introduire la notion de liaisons virtuelles, extension immédiate du concept bien connu des mécaniciens de liaisons entre solides.

On peut alors classer les différentes tâches référencées capteurs en fonction de la liaison virtuelle que l'on souhaite réaliser entre les capteurs et leur environnement. On applique cette approche au cas particulier du capteur vision et on propose quelques motifs capables de réaliser ces différentes liaisons.

3.1 Notion de liaison virtuelle

On rappelle que le torseur d'interaction H_j associé à une information visuelle s_j est défini par l'équation fondamentale :

$$\dot{s}_j = H_j \bullet T \tag{3.1}$$

où T est le torseur cinématique. Soit T^* un mouvement virtuel laissant invariante la sortie capteur s_j à une situation $\bar{r}$ donnée. T^* est solution de l'équation :

$$H_j \bullet T^* = 0 \tag{3.2}$$

et est, par définition, un torseur réciproque de H_j.

Passons à présent au capteur complet, caractérisé par sa sortie $\underline{s}$ constituée par les k informations visuelles s_j, $j = 1$ à k. L'ensemble des mouvements T^* laissant $\underline{s}$ invariant est alors le sous-espace S^* réciproque du sous-espace S de

torseurs engendré par l'ensemble $\{H_1 \cdots H_j \cdots H_k\}$. En utilisant la représentation matricielle L^T des torseurs d'interaction associés à $\underline{s}$ ($\dot{\underline{s}} = L^T T$), on peut noter, avec un abus de notation évident, que :

$$S^* = \text{Ker } L^T \tag{3.3}$$

Par exemple, si $\underline{s}$ est constitué des coordonnées (X_0, Y_0) d'un point M_0 de l'image, résultant de la projection d'un point m_0 de l'espace euclidien e, on a (revoir l'équation (2.13)) :

$$L^T = \begin{pmatrix} -1/z_0 & 0 & X_0/z_0 & X_0Y_0 & -(1+X_0^2) & Y_0 \\ 0 & -1/z_0 & Y_0/z_0 & 1+Y_0^2 & -X_0Y_0 & -X_0 \end{pmatrix} \tag{3.4}$$

et on en déduit :

$$S^* = \begin{pmatrix} X_0 & 0 & z_0(1+X_0^2+Y_0^2) & 0 \\ Y_0 & 0 & 0 & z_0(1+X_0^2+Y_0^2) \\ 1 & 0 & 0 & 0 \\ 0 & X_0 & -X_0Y_0 & 1+X_0^2 \\ 0 & Y_0 & -(1+Y_0^2) & X_0Y_0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \tag{3.5}$$

S^* représente ici l'ensemble des mouvements possibles de la caméra tels que la projection de m_0 dans l'image soit toujours le point M_0. On y reconnaît entre autres les mouvements de translation sur l'axe OM_0 et de rotation autour de ce même axe ainsi que l'on pouvait s'y attendre intuitivement.

Plus précisément, nous pouvons dire qu'**un ensemble de contraintes compatibles, de la forme $\underline{s}(\bar{r}) - \underline{s}^* = 0$ constitue une** *liaison virtuelle* **entre les objets de l'environnement et le capteur**.

- **Remarque :** Le sous espace de torseurs S^* va nous permettre de caractériser une liaison virtuelle en utilisant des informations visuelles puisque $\dot{\underline{s}} = 0$ est une conséquence immédiate de $\underline{s}(\bar{r}) - \underline{s}^* = 0$.

A une situation $\bar{r}$ où ces contraintes sont satisfaites, la dimension N de S^* est appelée la *classe* de la liaison virtuelle au point $\bar{r}$. Soit $m = 6 - N$. Lorsque $m = k$, les k contraintes introduites par les signaux élémentaires sont indépendantes. Intuitivement, la dimension de $\underline{s}$ correspond alors au nombre m de degrés de liberté que l'on peut et souhaite contrôler à partir de $\underline{s}$. Le cas $k > m$ présente souvent cependant un intérêt pratique, par exemple par les effets de filtrage qu'il peut impliquer. Nous n'exclurons donc pas ce cas où les informations visuelles

sont redondantes et le prendrons même en compte explicitement dans la partie commande, exposée dans le chapitre suivant.

La notion de liaison virtuelle est clairement une extension immédiate du concept de liaison entre solides tel qu'il est classiquement défini en théorie des mécanismes. Cette notion de liaison virtuelle, qui peut d'ailleurs se transformer en liaison réelle dans le cas de capteurs de contact, nous permet de spécifier les tâches robotiques référencées capteurs que nous souhaitons réaliser d'une façon à la fois simple et cohérente avec la démarche classique connue sous le nom de commande hybride.

Dans le tableau 3.1 page suivante sont répertoriées les liaisons mécaniques sans frottement les plus classiques (extraites de la norme AFNOR E04-E015). La classe de chaque liaison et le nombre de degrés de liberté non contraints (en translation (**T**) ou en rotation (**R**)) qui permettent cette classification y sont indiqués. Signalons que nous nous limiterons par la suite à ne caractériser que les liaisons de classe 0, 1, 2 et 3. Les autres, en effet, sont peu courantes et ne paraissent pas adaptées pour être réalisées à l'aide d'une caméra embarquée.

Nous allons donc maintenant proposer différents motifs capables de représenter ces liaisons. Un travail équivalent a déjà été réalisé dans le cadre des capteurs proximétriques [Espiau 87b]. On rappelle qu'un capteur proximétrique fournit un signal dépendant, entre autres, de la distance, dans la direction du capteur, entre celui-ci et l'objet le plus proche. Il fallait donc choisir le nombre de capteurs et leur emplacement sur l'effecteur de manière à réaliser la liaison souhaitée. Dans le cadre de la vision qui nous intéresse, la caméra embarquée peut fournir, à travers les images qu'elle acquiert, un très grand nombre d'informations visuelles selon la complexité de la scène observée. Le problème consiste alors à choisir, dans cet ensemble d'informations, lesquelles sont capables de représenter la liaison voulue.

De plus, dans cet ensemble, il pourrait être intéressant de disposer d'un critère capable d'extraire les informations visuelles les plus performantes pour réaliser la liaison voulue. Ce critère devrait aussi bien inclure les aspects de traitements d'images (robustesse d'extraction vis à vis des bruits de mesure, ...) que des aspects liés à la commande (découplage, sensibilité, ...). Une étude a récemment été menée en ce sens [Feddema 89a] dans le cas très simple d'une tâche de positionnement par rapport à un ensemble de points. Le critère retenu, basé sur la norme de la matrice d'interaction associée à chaque triplet de points que l'on

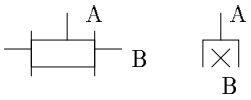
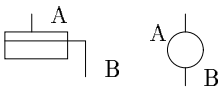
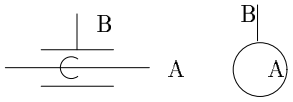
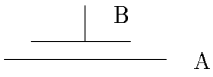
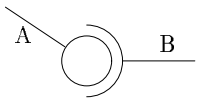
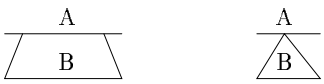
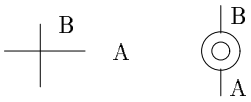
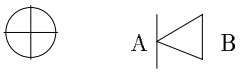
Nom de la liaison	**Classe**	**T**	**R**	**Symbole géométrique**
Rigide	0	0	0	
Prismatique	1	1	0	
Rotoïde	1	0	1	
Pivot glissant	2	1	1	
Appui plan	3	2	1	
Rotule	3	0	3	
Linéaire rectiligne	4	2	2	
Linéaire annulaire	4	1	3	
Ponctuelle	5	2	3	

Tableau 3.1 : Liaisons mécaniques

pouvait considérer, aboutit au résultat logique où les points doivent être le plus espacé possible dans l'image. Pour notre part, nous ne nous sommes pas intéressés à ce problème de choix d'un nombre minimal d'informations visuelles puisque l'approche proposée permet parfaitement de prendre en compte des informations redondantes ($k > m$). Aussi choisit-on pour réaliser une liaison virtuelle donnée l'ensemble des informations visuelles disponibles capables de la représenter. Dans les motifs proposés par la suite, nous donnons donc un nombre d'informations à prendre en compte supérieur ou égal au nombre minimal nécessaire.

3.2 Exemples de motifs réalisant les liaisons virtuelles

Dans les motifs proposés par la suite, nous utilisons naturellement les informations visuelles construites à partir des primitives géométriques étudiées dans le chapitre précédent et dont les torseurs d'interaction ont été calculés. Commençons tout d'abord par la liaison rigide.

3.2.1 Liaison de classe 0

Cette liaison correspond à une tâche de positionnement tel qu'il existe une seule situation entre la caméra et son environnement. En effet, on doit avoir $S^* = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}^T$, ce qui signifie qu'aucun mouvement de la caméra n'est permis à la position d'équilibre $\underline{s}(\bar{r}) = \underline{s}^*$. Il nous faut donc choisir au moins six informations visuelles s_j telles que la matrice d'interaction associée soit de rang plein 6. Un grand nombre de possibilités nous est offert...

Considérons tout d'abord le cas où $\underline{s}$ est constitué des coordonnées de trois points, résultant de la projection dans l'image de trois points m_i distincts. On a alors :

$$L^T = \begin{pmatrix} -1/z_1 & 0 & X_1/z_1 & X_1Y_1 & -(1+X_1^2) & Y_1 \\ 0 & -1/z_1 & Y_1/z_1 & 1+Y_1^2 & -X_1Y_1 & -X_1 \\ -1/z_2 & 0 & X_2/z_2 & X_2Y_2 & -(1+X_2^2) & Y_2 \\ 0 & -1/z_2 & Y_2/z_2 & 1+Y_2^2 & -X_2Y_2 & -X_2 \\ -1/z_3 & 0 & X_3/z_3 & X_3Y_3 & -(1+X_3^2) & Y_3 \\ 0 & -1/z_3 & Y_3/z_3 & 1+Y_3^2 & -X_3Y_3 & -X_3 \end{pmatrix} \tag{3.6}$$

Dans la plupart des cas, cette matrice est bien de rang plein mais certaines configurations des points m_i annulent le déterminant de L^T et entraînent donc une perte de rang à L^T. Il s'ensuit qu'elles ne sont pas utilisables en commande

pour réaliser une liaison rigide entre la caméra et la scène. Etudions très brièvement ces configurations particulières. Le déterminant δ de L^T peut se factoriser sous la forme :

$$\delta = a_1 P_1 + a_2 P_2 + a_3 P_3 \tag{3.7}$$

où P_1, P_2 et P_3 sont des polynomes de degrés 3 en X_i, Y_i et $1/z_i$ et où :

$$\begin{cases} a_1 = (1/z_2 - 1/z_1)(X_3 - X_2) - (1/z_2 - 1/z_3)(X_1 - X_2) \\ a_2 = (1/z_2 - 1/z_1)(Y_3 - Y_2) - (1/z_2 - 1/z_3)(Y_1 - Y_2) \\ a_3 = (Y_1 - Y_2)(X_3 - X_2) - (Y_3 - Y_2)(X_1 - X_2) \end{cases}$$

- Lorsque $a_1 = a_2 = a_3 = 0$, ce qui signifie que les trois points m_i sont alignés dans e, δ est évidemment nul.

- Si $a_1 = a_2 = 0$ et $a_3 \neq 0$, ce qui signifie que les trois points m_i sont parallèles au plan image ($z_1 = z_2 = z_3 = z$), δ peut se factoriser sous la forme :

$$\begin{aligned} \delta &= [(X_3^2 + Y_3^2)(X_1 Y_2 - X_2 Y_1) \\ &+ (X_2^2 + Y_2^2)(X_3 Y_1 - X_1 Y_3) \\ &+ (X_1^2 + Y_1^2)(X_2 Y_3 - X_3 Y_2)]\ a_3/z^3 \end{aligned} \tag{3.8}$$

  Il y a donc une infinité de configurations des points m_i pour que δ soit nul. En particulier, si l'un des points m_i se projette sur le point de l'image de coordonnées $(0,0)$ alors, quelle que soit la projection des autres points dans l'image, la matrice L^T ne sera pas de rang 6. Considérons par exemple le triplet de points de coordonnées $(0,0,z)$, $(z,0,z)$ et $(0,z,z)$. On peut facilement calculer le noyau de la matrice d'interaction L^T et l'on obtient :

$$S^* = \begin{pmatrix} z & z & -z & 1 & -1 & 0 \end{pmatrix}^T \tag{3.9}$$

  Nous sommes là en présence d'une singularité isolée. En effet, si on applique un mouvement à la caméra, même infinitésimal, appartenant à S^*, on sort de la singularité et L^T redevient de rang plein. Ce cas est analogue à la singularité que l'on a rencontrée dans le chapitre précédent concernant le cercle se projetant sous la forme d'un cercle centré.

  De plus, si l'on calcule le noyau de L qui représente, par définition, l'ensemble des mouvements dans l'image impossibles à réaliser, on trouve :

$$\text{Ker } L = \begin{pmatrix} 1 & -1 & 0 & 1 & -1 & 0 \end{pmatrix}^T \tag{3.10}$$

  ce qui signifie qu'il n'existe pas de mouvement de la caméra qui permette de réaliser le déplacement dans l'image symbolisé sur la Figure 3.1, et ce, si les trois points m_i sont dans un plan parallèle au plan image !

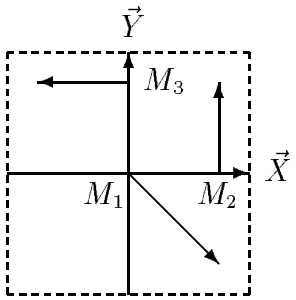


Figure 3.1 : Exemple de mouvement dans l'image physiquement irréalisable

Pour continuer la caractérisation de ces configurations particulières, il faudrait étudier en détail les différents autres cas annulant le déterminant de L^T (par exemple, $a_1 = a_3 = 0$ puis $a_1 = 0$,... qui permettent de simplifier l'écriture de δ). Mais, il faudrait également considérer le cas général où $a_1 \neq 0$, $a_2 \neq 0$ et $a_3 \neq 0$ et où aucune simplification n'apparaît intuitivement. Aussi arrêterons-nous là l'étude de ces configurations entre la caméra et trois points.

Si l'on utilise une autre représentation de ce triangle constitué des sommets m_i (par exemple celle donnée dans le chapitre précédent constituée par les coordonnées du centre de gravité, les longueurs des arêtes et l'orientation d'un axe de ce triangle), on retrouve bien évidemment les mêmes configurations entraînant une perte de rang à L^T, puisque le passage d'une représentation à l'autre est effectué à l'aide d'un difféomorphisme.

De plus, il est bien connu qu'il existe généralement quatre situations distinctes entre la caméra et la scène telles que l'image des trois points soit la même [Rives 81], [Horaud 87], [Dhome 89], [Faugeras 90]. L'application "projection perspective" f donnée dans le schéma 2.2 n'est alors pas injective. Ainsi, dans ce cas, on peut très bien avoir $\underline{s} - \underline{s}^* = 0$ sans que la situation entre la caméra et ces trois points soit celle qui est souhaitée. Aucune loi de commande, basée sur la régulation de $\underline{s} - \underline{s}^*$, ne pourra, à partir d'une situation où $\underline{s} - \underline{s}^* = 0$, faire déplacer la caméra vers une autre situation où ces contraintes sont réalisées.

Le problème est exactement le même si l'on considère le cas de trois droites. Il faut s'assurer, d'une part, qu'une seule situation entre la caméra et ces droites réalise $\underline{s} - \underline{s}^*$ et, d'autre part, que la matrice L^T associée est de rang plein. Si l'on choisit trois droites coplanaires concourantes, on retrouve encore les mêmes

singularités isolées que dans le cas des trois points (on peut considérer un point comme l'intersection de deux droites).

Le problème de trouver une signification physique à ces configurations particulières entraînant une singularité isolée n'est assurément pas un problème simple. Cependant, la géométrie de Grassmann, qui concerne la caractérisation des variétés de droites, pourrait peut-être permettre de résoudre ce problème. En effet, dans le cas des droites, les composantes des torseurs d'interaction des paramètres ρ et θ peuvent être considérés comme des coordonnées pluckeriennes, condition essentielle pour entrer dans le cadre de cette géométrie (voir, pour de plus amples détails, les travaux de Merlet [Merlet 88] qui a utilisé la géométrie de Grassmann pour caractériser les singularités d'un robot parallèle). Une telle étude, pour les singularités qui nous concernent, ne peut malheureusement qu'être évoquée ici, mais le problème, de par son "mystère" et sa complexité, mériterait certainement que l'on s'y attache.

Une solution pour éviter ces problèmes de singularités isolées et de convergence vers une situation non désirée, peut facilement s'obtenir : il suffit de prendre en compte dans $\underline{s}$ des informations visuelles supplémentaires qui ne dépendent pas de celles déjà choisies : les coordonnées d'un quatrième point, les paramètres représentant une quatrième droite, ou toute autre information qui fait que la matrice soit toujours de rang plein et qu'il n'y ait qu'une seule situation possible entre la caméra et la scène lorsque $\underline{s}(\bar{r}) = \underline{s}^*$.

Le choix d'informations visuelles pour les autres liaisons virtuelles comporte beaucoup moins de difficultés, comme nous allons le voir à présent.

3.2.2 Liaisons de classe 1

On va ici étudier deux types de liaison de classe 1 : la liaison prismatique où le degré de liberté non contraint de la caméra est un mouvement de translation et la liaison rotoïde qui permet un mouvement de rotation.

3.2.2.1 Liaison prismatique

Pour pouvoir réaliser une liaison prismatique, le sous-espace S^* correspondant aux informations visuelles choisies doit avoir la forme :

$$S^* = \begin{pmatrix} a & b & c & 0 & 0 & 0 \end{pmatrix}^T \tag{3.11}$$

où la direction du vecteur de composantes (a, b, c) est la direction non contrainte par la liaison.

Pour réaliser cette liaison, nous allons proposer différents motifs.

3.2.2.1.1 Motif basé sur des droites parallèles :

Considérons tout d'abord un faisceau de droites parallèles de direction (a, b, c). Les informations visuelles que l'on choisit sont naturellement les paramètres ρ_i et θ_i représentant ces droites. Trois droites, au moins, doivent être prises en compte de manière à avoir cinq informations visuelles indépendantes.

Considérons donc n droites $\mathcal{D}_i$ de direction (a, b, c) dans e. L'équation les caractérisant peut s'écrire sous la forme (voir équation (2.48)) :

$$h_i(\underline{x}, \underline{p}) = \begin{cases} a_{1i}x + b_{1i}y + c_{1i}z + d_{1i} = 0 \\ a_{2i}x + b_{2i}y + c_{2i}z + d_{2i} = 0 \end{cases} , \; i = 1 \text{ à } n \tag{3.12}$$

$$\text{avec} \begin{cases} a = b_{1i}c_{2i} - b_{2i}c_{1i} \\ b = c_{1i}a_{2i} - c_{2i}a_{1i} \\ c = a_{1i}b_{2i} - a_{2i}b_{1i} \end{cases}$$

La matrice d'interaction associée à la représentation (ρ, θ) peut immédiatement s'obtenir à partir de l'équation (2.63) :

$$L_i^T = \begin{pmatrix} \lambda_{\rho_i}\cos\theta_i & \lambda_{\rho_i}\sin\theta_i & -\lambda_{\rho_i}\rho_i & (1+\rho_i^2)\sin\theta_i & -(1+\rho_i^2)\cos\theta_i & 0 \\ \lambda_{\theta_i}\cos\theta_i & \lambda_{\theta_i}\sin\theta_i & -\lambda_{\theta_i}\rho_i & -\rho_i\cos\theta_i & -\rho_i\sin\theta_i & -1 \end{pmatrix} \tag{3.13}$$

Par ailleurs, on sait (équation (2.50)) que les droites $\mathcal{D}_i$ se projettent dans l'image en des droites d'équations :

$$A_iX + B_iY + C_i = 0 \text{ avec } \begin{cases} A_i = a_{1i}d_{2i} - a_{2i}d_{1i} \\ B_i = b_{1i}d_{2i} - b_{2i}d_{1i} \\ C_i = c_{1i}d_{2i} - c_{2i}d_{1i} \end{cases} \tag{3.14}$$

d'où on peut déduire les valeurs de $\cos\theta_i$, $\sin\theta_i$ et ρ_i à l'aide de l'équation (2.59) :

$$\begin{aligned} \cos\theta_i &= (a_{1i}d_{2i} - a_{2i}d_{1i})/\sqrt{A^2 + B^2} \\ \sin\theta_i &= (b_{1i}d_{2i} - b_{2i}d_{1i})/\sqrt{A^2 + B^2} \\ \rho_i &= -(c_{1i}d_{2i} - c_{2i}d_{1i})/\sqrt{A^2 + B^2} \end{aligned} \tag{3.15}$$

On peut ainsi vérifier que :

$$a\cos\theta_i + b\sin\theta_i - c\rho_i = 0 \; , \; \forall i \tag{3.16}$$

ce qui prouve que $\begin{pmatrix} a & b & c & 0 & 0 & 0 \end{pmatrix}^T$ est inclus dans S^*.

De plus, en prenant en compte au moins trois droites, on peut montrer que la matrice L^T associée est toujours de rang 5. Le motif choisi est donc valide pour réaliser la liaison prismatique.

3.2.2.1.2 Motif basé sur des droites coplanaires concourantes :

Considérons à présent d'autres motifs capables de réaliser une liaison prismatique. Supposons par exemple que l'on dispose dans e d'un faisceau de droites coplanaires concourantes en un point de coordonnées (x^*, y^*, z^*). On peut alors facilement montrer qu'en choisissant comme informations visuelles :

- (1) les coordonnées (X^*, Y^*) du point résultant de la projection du point d'intersection des droites, et
  (2) les paramètres ρ_i et θ_i représentant au moins deux droites de ce faisceau,

ou

- les paramètres ρ_i et θ_i représentant au moins trois droites de ce faisceau,

alors on a :

$$S^* = \begin{pmatrix} X^* & Y^* & 1 & 0 & 0 & 0 \end{pmatrix}^T$$

excepté si $X^* = Y^* = 0$ et si, en outre, le plan contenant ce faisceau de droites est parallèle au plan image.

- **Remarque :** ce motif permet donc de définir toutes les configurations de liaison prismatique sauf celles autorisant des mouvements de la caméra parallèles ou perpendiculaire au plan image.

L'équation de droites coplanaires concourantes s'écrit :

$$h_i(\underline{x}, \underline{p}) = \begin{cases} ax + by + cz + d = 0 \\ a_i x + b_i y + c_i z + d_i = 0 \end{cases} \tag{3.17}$$

$$\text{avec} \begin{cases} ax^* + by^* + cz^* + d = 0 \\ a_i x^* + b_i y^* + c_i z^* + d_i = 0 \end{cases}$$

si (x^*, y^*, z^*) est le point d'intersection de ces droites et si (a, b, c) est la normale du plan contenant ces droites. On a alors :

$$\begin{aligned} \cos\theta_i &= (ad_i - a_i d)/\sqrt{A^2 + B^2} \\ \sin\theta_i &= (bd_i - b_i d)/\sqrt{A^2 + B^2} \\ \rho_i &= -(cd_i - c_i d)/\sqrt{A^2 + B^2} \end{aligned} \tag{3.18}$$

et la matrice d'interaction associée à ρ_i et θ_i s'écrit toujours :

$$L_i^T = \begin{pmatrix} \lambda_{\rho_i}\cos\theta_i & \lambda_{\rho_i}\sin\theta_i & -\lambda_{\rho_i}\rho_i & (1+\rho_i^2)\sin\theta_i & -(1+\rho_i^2)\cos\theta_i & 0 \\ \lambda_{\theta_i}\cos\theta_i & \lambda_{\theta_i}\sin\theta_i & -\lambda_{\theta_i}\rho_i & -\rho_i\cos\theta_i & -\rho_i\sin\theta_i & -1 \end{pmatrix} \tag{3.19}$$

$$\begin{array}{lll} \text{avec} & \lambda_{\rho_i} & = (a\rho_i\cos\theta_i + b\rho_i\sin\theta_i + c)/d \\ \text{et} & \lambda_{\theta_i} & = (a\sin\theta_i - b\cos\theta_i)/d \end{array}$$

On peut vérifier que :

$$x^*\cos\theta_i + y^*\sin\theta_i - z^*\rho_i = 0 \ , \ \forall i \tag{3.20}$$

ce qui prouve que $\begin{pmatrix} X^* & Y^* & 1 & 0 & 0 & 0 \end{pmatrix}^T$ est inclus dans S^*. De plus, si l'on choisit les coordonnées (X^*, Y^*) comme informations visuelles supplémentaires, on a :

$$\begin{array}{l} L_{X^*}^T = [\ -1/z^* \quad 1 \quad X^*/z^* \quad X^*Y^* \quad -(1+X^{*^2}) \quad Y^*\ ] \\ L_{Y^*}^T = [\ 0 \quad -1/z^* \quad Y^*/z^* \quad 1+Y^{*^2} \quad -X^*Y^* \quad -X^*\ ] \end{array} \tag{3.21}$$

et l'on vérifie également que $\begin{pmatrix} X^* & Y^* & 1 & 0 & 0 & 0 \end{pmatrix}^T$ est inclus dans S^*. Comme précédemment, en prenant en compte au moins deux droites et les coordonnées (X^*, Y^*) ou en prenant en compte au moins trois droites, on peut montrer que la matrice L^T associée est toujours de rang 5 si $X^* \neq 0$ ou si $Y^* \neq 0$.

Considérons à présent le cas particulier $X^* = Y^* = 0$ qui correspondrait, si la liaison était réalisée, à permettre un mouvement le long de l'axe optique. A l'aide des équations précédentes, on obtient $\rho_i = 0$ et $\lambda_{\rho_i} = -1/z^*$. On en déduit :

$$\begin{array}{lll} L_{X^*}^T & = [& -1/z^* \quad 1 \quad 0 \quad 0 \quad -1 \quad 0\ ] \\ L_{Y^*}^T & = [& 0 \quad -1/z^* \quad 0 \quad 1 \quad 0 \quad 0\ ] \\ L_{\rho_i}^T & = [& -\cos\theta_i/z^* \quad -\sin\theta_i/z^* \quad 0 \quad \sin\theta_i \quad -\cos\theta_i \quad 0\ ] \\ L_{\theta_i}^T & = [& \lambda_{\theta_i}\cos\theta_i \quad \lambda_{\theta_i}\sin\theta_i \quad 0 \quad 0 \quad 0 \quad -1\ ] \end{array} \tag{3.22}$$

avec $\lambda_{\theta_i} = (a_i b - a b_i)$.

Quel que soit le nombre de droites prises en compte, il apparaît dans cette dernière équation que le rang de la matrice L^T, associée à ces différentes informations visuelles, est toujours de rang 5, excepté le cas où $\lambda_{\theta_i} = 0 \ , \ \forall i$ et qui correspond au cas où $a = b = 0$. En effet, L^T est alors de rang 3 et on a :

$$S^* = \begin{pmatrix} 0 & z^* & 0 \\ 0 & 0 & z^* \\ 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{pmatrix} \tag{3.23}$$

3.2.2.1.3 Motif basé sur les segments :

Il est toutefois possible de réaliser cette liaison en utilisant un motif à peu près équivalent (on peut aussi, bien sûr, utiliser le premier motif proposé, constitué d'un faisceau de droites parallèles orientées selon l'axe optique). Considérons en effet un faisceau de segments parallèles au plan image et concourants, en leur milieu, au point de coordonnées $(0, 0, z^*)$ (voir Figure 3.2).

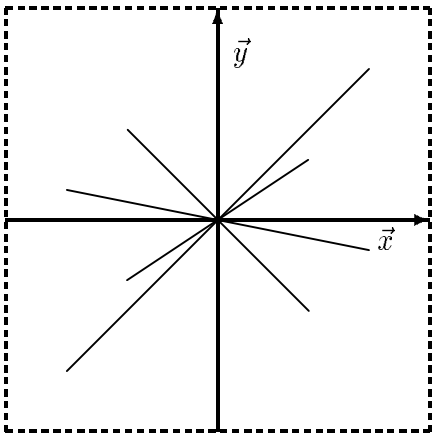


Figure 3.2 : Liaison prismatique à l'aide de segments (plan $z = z^*$)

Ce faisceau se projette sur le plan image sous la forme d'un faisceau de segments ayant les mêmes propriétés et on choisit comme informations visuelles les coordonnées (X_{c_i}, Y_{c_i}) du milieu d'au moins deux de ces segments et l'orientation α_i d'au moins un de ces segments. La matrice d'interaction s'obtient facilement à l'aide de l'étude sur les segments effectuée dans le chapitre précédent (équation (2.40)), et l'on obtient quand $X_{c_i} = Y_{c_i} = 0$:

$$\begin{array}{lcccccccc}
L^T_{X_{c_1}} & = [& -1/z^* & 0 & 0 & l_1^2 \cos\alpha_1 \sin\alpha_1/4 & -(1 + l_1^2 \cos^2\alpha_1/4) & 0 &] \\
L^T_{Y_{c_1}} & = [& 0 & -1/z^* & 0 & 1 + l_1^2 \sin^2\alpha_1/4 & -l_1^2 \cos\alpha_1 \sin\alpha_1/4 & 0 &] \\
L^T_{X_{c_2}} & = [& -1/z^* & 0 & 0 & l_2^2 \cos\alpha_2 \sin\alpha_2/4 & -(1 + l_2^2 \cos^2\alpha_2/4) & 0 &] \\
L^T_{Y_{c_2}} & = [& 0 & -1/z^* & 0 & 1 + l_2^2 \sin^2\alpha_2/4 & -l_2^2 \cos\alpha_2 \sin\alpha_2/4 & 0 &] \\
L^T_{\alpha_i} & = [& 0 & 0 & 0 & 0 & 0 & -1 &]
\end{array} \tag{3.24}$$

où l_1 et l_2 sont la longueur de ces segments. Cette matrice est bien toujours de rang 5, quelles que soient l'orientation et la longueur des segments choisis.

Nous terminons par ce motif l'étude de la liaison prismatique. Passons maintenant à une autre liaison de classe 1 : la liaison rotoïde.

3.2.2.2 Liaison rotoïde

Nous ne nous intéresserons ici qu'à une liaison rotoïde particulière : celle qui permet un mouvement de rotation de la caméra autour de son axe optique. On doit donc trouver des motifs tels que le noyau de la matrice d'interaction associée soit :

$$S^* = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}^T \tag{3.25}$$

Le motif qui vient naturellement à l'esprit est constitué du cercle centré parallèle au plan image. Malheureusement, on a vu dans le chapitre précédent (paragraphe 2.3.3) que ce motif correspondait à une configuration singulière. On rappelle qu'en utilisant par exemple la représentation des ellipses par les moments, la matrice d'interaction associée à cette représentation s'écrit :

$$L^T = \begin{pmatrix} -1/z^* & 0 & 0 & 0 & -1-R^2 & 0 \\ 0 & -1/z^* & 0 & 1+R^2 & 0 & 0 \\ 0 & 0 & 2R^2/z^* & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2R^2/z^* & 0 & 0 & 0 \end{pmatrix} \tag{3.26}$$

où R est le rayon du cercle projeté et où $z - z^*$ est l'équation du plan contenant ce cercle. On a alors :

$$S^* = \begin{pmatrix} 0 & 1+R^2 & 0 \\ 0 & 0 & -1-R^2 \\ 0 & 0 & 0 \\ 0 & -1/z^* & 0 \\ 0 & 0 & -1/z^* \\ 1 & 0 & 0 \end{pmatrix} \tag{3.27}$$

On retrouve bien dans S^* le mouvement attendu mais, pour que S^* représente effectivement la liaison voulue, des informations visuelles supplémentaires doivent être prises en compte. On peut choisir, par exemple (voir Figure 3.3) :

- les informations relatives à un autre cercle parallèle au plan image et également centré (soit dans le même plan et de rayon différent, soit de même rayon et dans un plan différent),
- les coordonnées de la projection d'un point de l'axe $\vec{z}$ (soit de coordonnées $(0, 0, z^*)$ dans e), et non pas les coordonnées du centre du cercle qui sont déjà utilisées.

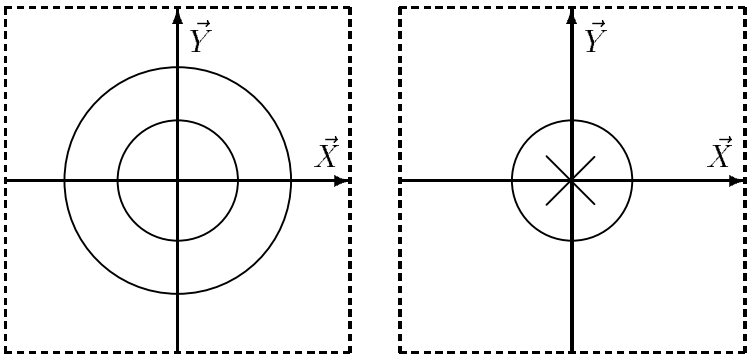


Figure 3.3 : Liaison rotoïde à l'aide de cercles

On peut également réaliser la même liaison en utilisant le motif déjà utilisé constitué d'un faisceau d'au moins deux segments coplanaires et concourants, en leur milieu, au point de coordonnées $(0, 0, z^*)$ (voir Figure 3.2). On utilise, cette fois-ci, comme informations visuelles les coordonnées du milieu des segments et leur longueur. La représentation matricielle des torseurs d'interaction associés s'écrit :

$$\begin{array}{lcccccccc} L^T_{X_{c_i}} & = [& -1/z^* & 0 & 0 & l_i^2 \cos\alpha_i \sin\alpha_1/4 & -(1+l_i^2\cos^2\alpha_i/4) & 0 &] \\ L^T_{Y_{c_i}} & = [& 0 & -1/z^* & 0 & 1+l_i^2\sin^2\alpha_i/4 & -l_i^2\cos\alpha_i\sin\alpha_i/4 & 0 &] \\ L^T_{l_i} & = [& 0 & 0 & l_i/z^* & 0 & 0 & 0 &] \end{array} \tag{3.28}$$

On peut aisément vérifier qu'en utilisant au moins deux segments, la liaison souhaitée est réalisée.

3.2.3 Liaison de classe 2

Etudions à présent la liaison de classe 2 définie dans le tableau 3.1 : le pivot glissant. On doit maintenant trouver des motifs tels que :

$$S^* = \begin{pmatrix} a & 0 \\ b & 0 \\ c & 0 \\ 0 & a \\ 0 & b \\ 0 & c \end{pmatrix} \tag{3.29}$$

où le vecteur de coordonnées (a, b, c) représente l'axe de rotation et la direction de translation des mouvements non contraints de la caméra. Pour réaliser cette

liaison, considérons tout d'abord, un cylindre de rayon R dont la droite génératrice est parallèle au plan image et passe par le point de coordonnées $(0, 0, z^*)$. Ce cylindre a pour équation (voir paragraphe 2.3.5) :

$$h(\underline{x}, \underline{p}) = (bx - ay)^2 + (z - z^*)^2 - R^2 = 0 \tag{3.30}$$

où le vecteur $(a, b, 0)$ représente la direction de la droite génératrice du cylindre (on pose $a^2 + b^2 = 1$). Ce cylindre se projette sur le plan image sous la forme de deux droites parallèles symétriques par rapport à l'origine du plan image (voir Figure 3.4) et d'équations :

$$\begin{cases} D_1 : & bX - aY + R/\sqrt{z^{*2} - R^2} = 0 \\ D_2 : & -bX + aY + R/\sqrt{z^{*2} - R^2} = 0 \end{cases} \tag{3.31}$$

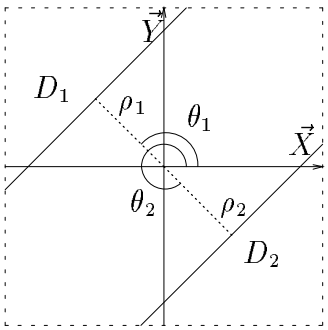


Figure 3.4 : Pivot glissant à l'aide d'un cylindre

En utilisant la représentation (ρ, θ) des droites, on obtient :

$$\begin{cases} \cos\theta_1 & = & -\cos\theta_2 & = & -b \\ \sin\theta_1 & = & -\sin\theta_2 & = & a \\ \rho_1 & = & \rho_2 & = & R/\sqrt{z^{*2} - R^2} \end{cases} \tag{3.32}$$

et on en déduit la valeur des torseurs d'interaction associés à ces paramètres à l'aide des équations (2.93) :

$$\begin{array}{lcccccccc} L_{\rho_1}^T & = & [& \lambda_\rho b & -\lambda_\rho a & -\lambda_\rho \rho & -a(1+\rho^2) & -b(1+\rho^2) & 0 &] \\ L_{\theta_1}^T & = & [& 0 & 0 & 0 & -b\rho & a\rho & -1 &] \\ L_{\rho_2}^T & = & [& -\lambda_\rho b & \lambda_\rho a & -\lambda_\rho \rho & a(1+\rho^2) & b(1+\rho^2) & 0 &] \\ L_{\theta_2}^T & = & [& 0 & 0 & 0 & b\rho & -a\rho & -1 &] \end{array} \tag{3.33}$$

avec $\lambda_\rho = -z^*/(z^{*2} - R^2)$.

On peut facilement calculer l'espace de torseurs invariants S^*. On a :

$$S^* = \begin{pmatrix} a & b(1+\rho^2) \\ b & -a(1+\rho^2) \\ 0 & 0 \\ 0 & \lambda_\rho a \\ 0 & \lambda_\rho b \\ 0 & 0 \end{pmatrix} \tag{3.34}$$

Exprimons cet espace de torseur en un point de l'axe de cette liaison, par exemple au point de coordonnées $(0, 0, z^*)$. On rappelle que, pour tout torseur $T = (V, \omega)$ et tout point m de e, on a $\omega(m) = \omega(O)$ et $V(m) = V(O) + \omega \times Om$. S^* peut alors s'écrire sous la forme :

$$S^* = \begin{pmatrix} a & 0 \\ b & 0 \\ 0 & 0 \\ 0 & a \\ 0 & b \\ 0 & 0 \end{pmatrix} \tag{3.35}$$

Ce motif permet donc de réaliser correctement toutes les liaisons de type pivot glissant dont l'axe est parallèle au plan image ($c = 0$). Celle dont l'axe correspond à l'axe $\vec{z}$ peut facilement s'obtenir à partir des motifs définis lors de l'étude de la liaison rotoïde. On peut en effet vérifier qu'en choisissant :

- pour les motifs basés sur les cercles, les paramètres X_{c_i} et Y_{c_i}, représentant les coordonnées du centre des cercles considérés,
- pour les motifs basés sur les segments, les paramètres X_{c_i} et Y_{c_i}, représentant les coordonnées du milieu des segments considérés,

on obtient :

$$S^* = \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix} \tag{3.36}$$

Nous ne nous intéresserons pas ici au cas général où $a \neq 0, b \neq 0$ et $c \neq 0$ en raison de l'absence de signification physique que représente une telle liaison entre une caméra et une scène.

3.2.4 Liaisons de classe 3

On peut à présent s'intéresser à deux liaisons de classe 3 : l'appui plan et la liaison rotule.

3.2.4.1 Appui plan

Pour pouvoir réaliser cette liaison, il nous faut trouver un motif tel que l'espace des torseurs invariants associés soit :

$$S^* = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{3.37}$$

Comme la liaison appui plan permet des mouvements parallèles au plan image, la scène nécessaire pour réaliser cette liaison doit être constituée d'un maillage parallèle au plan image et régulier de manière à ce que des informations visuelles de même nature soient toujours dans le champ de vue de la caméra. On choisit en outre des maillages tels que tous les segments les constituant aient même longueur (voir Figure 3.5).

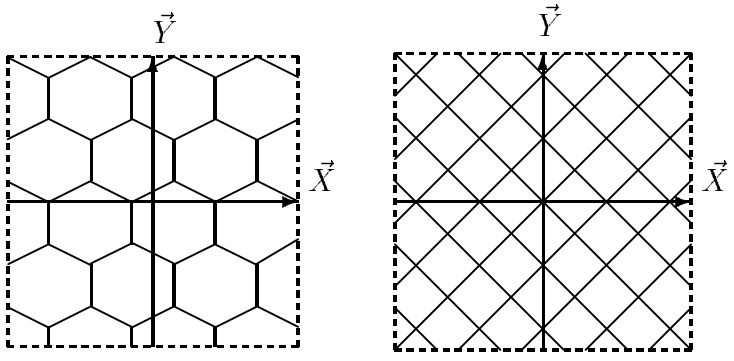


Figure 3.5 : Appui plan à l'aide de maillages

On peut alors choisir comme informations visuelles les longueurs d'au moins trois segments non alignés de ce maillage (par exemple, les trois dans le cas d'un

maillage hexagonal, ou les quatre dans le cas d'un maillage carré, dont l'une des extrémités des segments minimise la distance à l'origine du plan image). On peut en effet vérifier que, pour chaque segment représenté par (X_c, Y_c, l, α), on a :

$$L_l^T = \begin{bmatrix} 0 & 0 \\ l/z^* & l[X_c \cos\alpha \sin\alpha + Y_c(1+\sin^2\alpha)] \\ -l[X_c(1+\cos^2\alpha) + Y_c \cos\alpha \sin\alpha] & 0 \end{bmatrix} \tag{3.38}$$

En prenant en compte au moins trois segments, S^* a la forme voulue pour réaliser l'appui plan (à une distance z^* !) entre la caméra et le maillage.

3.2.4.2 Liaison rotule

Enfin, pour terminer ce descriptif des motifs réalisant les liaisons virtuelles les plus classiques, considérons la liaison rotule. Le motif capable de réaliser cette liaison est constitué des paramètres représentant la projection dans l'image d'une sphère de centre le point de coordonnées $(0, 0, z^*)$. En effet, cette sphère se projette alors sous la forme d'un cercle centré et, en utilisant les calculs développés dans le paragraphe 2.3.4, on peut facilement calculer les torseurs d'interaction associés à la représentation choisie (par exemple les moments X_c, Y_c et $\mu = (\mu_{20} + \mu_{02})/2$). On obtient :

$$\begin{array}{lcccccc} L_{X_c}^T & = [& -1/z^* & 0 & 0 & 0 & -1-R^2 & 0 \ ] \\ L_{Y_c}^T & = [& 0 & -1/z^* & 0 & 1+R^2 & 0 & 0 \ ] \\ L_{\mu}^T & = [& 0 & 0 & 2R^2/z^* & 0 & 0 & 0 \ ] \end{array} \tag{3.39}$$

Exprimée au point de coordonnées $(0, 0, z^*)$, les torseurs d'interaction prennent les valeurs suivantes :

$$\begin{array}{lcccccccc} L_{X_c}^T & = [& -1/z^* & 0 & 0 & 0 & 0 & 0 & \ ] \\ L_{Y_c}^T & = [& 0 & -1/z^* & 0 & 0 & 0 & 0 & \ ] \\ L_{\mu}^T & = [& 0 & 0 & 2R^2/z^* & 0 & 0 & 0 & \ ] \end{array} \tag{3.40}$$

On vérifie ainsi que le motif choisi permet de réaliser la liaison rotule :

$$S^* = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{3.41}$$

3.2.5 Autres liaisons

Nous n'avons jusqu'à présent détaillé que des liaisons virtuelles entre la caméra et son environnement qui correspondent aux liaisons mécaniques classiques décrites dans le tableau 3.1 : liaison prismatique, rotoïde, rotule,... On peut bien entendu généraliser cette étude à des liaisons virtuelles quelconques selon l'application envisagée. Ainsi, des tâches que l'on peut qualifier de typiquement visuelles sont réalisables si l'on peut trouver un motif $\underline{s}^*$ tel que $\underline{s}(\bar{r}) - \underline{s}^* = 0$ représente effectivement la tâche.

Considérons par exemple la poursuite de cible. Cette tâche peut tout simplement s'exprimer sous la forme d'un positionnement de la projection dans l'image d'un objet au centre de l'image. Les deux informations visuelles nécessaires pour réaliser cette tâche sont bien évidemment les coordonnées (X_c, Y_c) du centre de gravité dans l'image de cet objet. La liaison virtuelle obtenue est de classe 4 puisque (voir l'équation (3.5)) :

$$S^* = \begin{pmatrix} 0 & 0 & z_c & 0 \\ 0 & 0 & 0 & z_c \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \tag{3.42}$$

Avant de conclure ce chapitre, il faut signaler que l'on a toujours considéré le cas le plus général où la caméra a six degrés de liberté. On peut bien entendu effectuer la même étude pour les cas plus simples où la caméra a moins de six degrés de liberté. Les motifs proposés précédemment sont évidemment toujours valides pour réaliser les liaisons étudiées et on peut même, si besoin est, les simplifier. En effet, on a vu que le nombre minimal d'informations visuelles à prendre en compte est égal au nombre de degré de liberté de la caméra moins la classe de la liaison virtuelle. Ainsi, si l'on considère par exemple le cas d'une caméra embarquée sur un robot mobile à trois degrés de liberté, les informations visuelles relatives à une seule droite sont suffisantes pour réaliser une liaison prismatique.

De plus, les problèmes de singularités isolées disparaissent dès que les couplages entre les mouvements de translation parallèlement au plan image et ceux de rotation autour des axes du plan image sont physiquement interdits par l'absence de mobilité de la caméra selon telle ou telle direction.

D'autres motifs peuvent également être choisis : la liste proposée dans ce chapitre n'est en aucun cas exhaustive, mais donne un aperçu des choix, souvent

naturels, à effectuer pour réaliser une liaison donnée.

Ainsi s'achève ce chapitre de classification des tâches. Pour chaque type de tâches, caractérisé par la liaison virtuelle entre la caméra et la scène, on a proposé quelques motifs capables de les représenter. On a vu que le torseur d'interaction jouait un rôle essentiel dans la constitution de ces motifs afin de calculer l'espace des torseurs invariants S^*. Le torseur d'interaction joue également un grand rôle, comme nous allons le voir dans le chapitre suivant, dans l'élaboration des lois de commande qui vont permettre de réaliser efficacement une tâche donnée.

Chapitre 4

La commande référencée vision

Dans ce chapitre, nous rappelons brièvement les résultats qui ont été obtenus par Claude Samson et Bernard Espiau en commande des robots manipulateurs et en commande référencée capteurs. Nous ne saurions trop recommander aux lecteurs intéressés par la commande la lecture de leur récent ouvrage [Samson 90b]. Leur approche, basée sur la régulation d'une *fonction de tâche*, intègre parfaitement les aspects de redondance ainsi que les notions de tâches hybrides. Nous appliquons ensuite leurs résultats au cas particulier d'une commande basée sur la vision, la *commande référencée vision*. A la fin de ce chapitre, nous donnons la loi de commande simplifiée qui a été implantée pour réaliser les tâches que nous avons étudiées en simulation et implantées sur site expérimental. L'intégration d'un estimateur de vitesse permettant la réalisation des tâches de suivi d'objets mobiles y est également décrite

4.1 Le concept de fonction de tâche

Il est bien connu que le comportement dynamique d'un robot rigide est décrit par l'équation :

$$\underline{\Gamma} = M(\underline{q})\ \ddot{\underline{q}} + \underline{N}(\underline{q}, \dot{\underline{q}}, t) \tag{4.1}$$

avec dim $\underline{q}$ = dim $\underline{\Gamma}$ = dim $\underline{N}$ = n et où :

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

L'équation (4.1) est l'équation *d'état* du système dont le vecteur d'état, naturellement associé, est $(\underline{q}, \dot{\underline{q}})$. La tâche à effectuer peut alors être spécifiée par l'utilisateur comme une *fonction de sortie* associée à (4.1). Le problème sera alors bien posé si le passage entre l'"espace de commande" et l'"espace de sortie" est régulier en un certain sens.

Plus précisément, on peut montrer [Samson 90b] que l'objectif de l'utilisateur peut généralement être exprimé comme la régulation (à zéro), sur un certain horizon $[0, T]$, d'une certaine fonction de classe C^2 de dimension n, $\underline{e}(\underline{q}, t)$, appelée *fonction de tâche*.

On a déjà cité quelques exemples de telles fonctions dans le chapitre 1 (au paragraphe 1.1) :

- $\underline{e}(\underline{q}, t) = \underline{q} - \underline{q}^*(t)$ où $\underline{q}^*(t)$ est une trajectoire désirée dans l'espace articulaire,
- $\underline{e}(\underline{q}, t) = \underline{r}(\underline{q}) - \underline{r}^*(t)$ où $\underline{r}^*(t)$ représente une trajectoire désirée dans l'espace opérationnel de l'effecteur d'un robot à six degrés de liberté.

Dans le cas particulier qui nous concerne où l'on utilise des capteurs, et même, plus précisément, un ensemble de k informations visuelles s, les fonctions de tâches que l'on cherche à réguler, appelées tâches référencées vision et notées $\underline{e}_1$, de dimension m ($m \leq n$ et $m \leq k$) s'écrivent (voir paragraphe 2.1.2) :

- $\underline{e}_1(\underline{q}, t) = C\ (\underline{s}(\underline{q}, t) - \underline{s}^*)$ où $\underline{s}^*$ est choisi, à l'aide de l'étude effectuée dans le chapitre précédent, pour réaliser une liaison virtuelle, de classe $N = 6 - m$, entre la caméra et son environnement.

Nous reviendrons plus loin sur la construction de la fonction de tâche complète $\underline{e}$ que l'on peut obtenir à partir de $\underline{e}_1$.

Nous reviendrons également sur les conditions régissant les différents choix possibles de la matrice de combinaison C tout au long de ce chapitre, mais on peut déjà s'apercevoir que C, de dimension $m \times k$ doit être de rang plein m afin que, si $\underline{s}$ est correctement choisi, les m composantes de $\underline{e}_1$ soient indépendantes et puissent contrôler effectivement les m degrés de liberté nécessaires à la réalisation de la tâche référencée vision. Ainsi pour pouvoir réaliser, par exemple, une liaison rigide, C doit être de rang 6, pour une liaison rotule ou appui plan, C doit

être de rang 3, etc. Cette matrice de combinaison permet, comme nous l'avons déjà signalé au paragraphe 2.1.2, de prendre en compte dans $\underline{e}_1$ un nombre d'informations visuelles k supérieur au nombre m de degrés de liberté contraints par la liaison. C'est pourquoi, dans le chapitre précédent, nous ne nous sommes pas particulièrement attachés à construire des motifs de dimension minimale m.

Comme cela est précisé en [Samson 87] et [Samson 90b], le problème général de régulation de $\underline{e}$ est bien posé si celle-ci possède certaines propriétés. L'une d'elles, qui signifie que la tâche est réalisable, est l'existence et l'unicité d'une *trajectoire idéale* de classe C^2, notée $\underline{q}_r(t)$, telle que :

$$\begin{cases} \underline{e}(\underline{q}_r(t), t) = 0, \ , \ \forall t \in [0, T] \\ \underline{q}_r(0) = q_0 \end{cases} \tag{4.2}$$

où q_0 est une condition initiale donnée. Une autre condition, extrêmement importante, est la *régularité* du jacobien de la tâche, $\frac{\partial \underline{e}}{\partial \underline{q}}(\underline{q}, t)$, autour de $\underline{q}_r(t)$. Pour notre tâche référencée vision, ce jacobien s'écrit :

$$\frac{\partial \underline{e}_1}{\partial \underline{q}}(\underline{q}, t) = \frac{\partial \underline{e}_1}{\partial \bar{r}} \frac{\partial \bar{r}}{\partial \underline{q}} = C L^T(\bar{r}) J(\underline{q}) \tag{4.3}$$

où l'on retrouve la matrice d'interaction L^T associée à $\underline{s}$ et le jacobien du robot $J(\underline{q})$. Si l'on suppose, comme on le fera par la suite, que le jacobien du robot est parfaitement connu et inversible, la satisfaction de la condition de régularité s'obtient en choisissant la matrice de combinaison C telle que $C L^T(\bar{r})$ soit de rang plein m dans un voisinage autour de $\underline{q}_r(t)$.

Lorsque toutes les conditions requises sont satisfaites, la fonction de tâche est dite *admissible*, ce qui permet alors la synthèse de lois de commande efficaces, dont la robustesse sera liée à la plus ou moins grande *admissibilité* de la tâche.

4.2 Commande et stabilité

Nous donnons ici seulement une idée **intuitive** de l'approche utilisée et du type de résultats obtenus, et renvoyons à [Samson 87] et [Samson 90b] pour une description complète. Considérons d'abord simplement le problème du découplage et de la linéarisation de (4.1) dans le cas idéal. Une commande les réalisant est de la forme :

$$\underline{\Gamma} = M(\underline{q}) \, \underline{u} + \underline{N} \tag{4.4}$$

où $\underline{u}$ est par exemple un retour proportionnel dérivé sur l'erreur $\underline{q} - \underline{q}^*$, si l'on souhaite réaliser un asservissement à la position $\underline{q}^*$:

$$\underline{u} = -\lambda G \left(\mu D \left(\underline{q} - \underline{q}^*\right) + \dot{\underline{q}}\right) \tag{4.5}$$

G et D étant des matrices positives, λ et μ des scalaires positifs, tous devant être fixés par l'utilisateur.

Appliquons la même méthode à présent dans l'espace de sortie. Nous pouvons écrire, puisque $\underline{e}$ est fonction de $\underline{q}$ et t :

$$\dot{\underline{e}}(\underline{q}, t) = \frac{\partial \underline{e}}{\partial \underline{q}}(\underline{q}, t)\ \dot{\underline{q}} + \frac{\partial \underline{e}}{\partial t}(\underline{q}, t) \tag{4.6}$$

et

$$\ddot{\underline{e}}(\underline{q}, t) = \frac{\partial \underline{e}}{\partial \underline{q}}(\underline{q}, t)\ \ddot{\underline{q}} + \underline{f}(\underline{q}, \dot{\underline{q}}, t) \tag{4.7}$$

avec :

$$\underline{f}(\underline{q}, \dot{\underline{q}}, t) = \begin{bmatrix} \vdots \\ \dot{\underline{q}}^T W_i(\underline{q}, t)\ \dot{\underline{q}} \\ \vdots \end{bmatrix} + 2\frac{\partial^2 \underline{e}}{\partial \underline{q} \partial t}(\underline{q}, t)\dot{\underline{q}} + \frac{\partial^2 \underline{e}}{\partial^2 t}(\underline{q}, t) \tag{4.8}$$

où $W_i(\underline{q}, t)(i = 1, \cdots, n)$ est la dérivée partielle de la i-ème colonne de $\left(\frac{\partial \underline{e}}{\partial \underline{q}}\right)^T(\underline{q}, t)$ par rapport à $\underline{q}$.

La fonction $\underline{e}$ étant supposée admissible, on peut combiner les équations (4.1) et (4.7), ce qui donne :

$$\underline{\Gamma} = M \left(\frac{\partial \underline{e}}{\partial \underline{q}}\right)^{-1} \ddot{\underline{e}} + \underline{N} - M \left(\frac{\partial \underline{e}}{\partial \underline{q}}\right)^{-1} \underline{f} \tag{4.9}$$

et, par analogie avec (4.4), une forme de commande linéarisante dans l'espace de la tâche serait :

$$\underline{\Gamma} = M \left(\frac{\partial \underline{e}}{\partial \underline{q}}\right)^{-1} \underline{u}' + \underline{N} - M \left(\frac{\partial \underline{e}}{\partial \underline{q}}\right)^{-1} \underline{f} \tag{4.10}$$

avec un retour proportionnel dérivé de forme générale :

$$\underline{u}' = -\lambda G \left(\mu D \underline{e} + \dot{\underline{e}}\right) \tag{4.11}$$

Naturellement, la commande idéale décrite par les équations (4.10) et (4.11) suppose une connaissance parfaite de tous les termes la constituant ce qui n'est

ni possible, ni même souvent souhaité. On considèrera donc finalement, de façon plus réaliste le schéma de commande suivant, généralisation du précédent :

$$\underline{\Gamma} = -\lambda \hat{M} \left(\widehat{\frac{\partial \underline{e}}{\partial \underline{q}}} \right)^{-1} G \, (\mu D \, \underline{e} + \widehat{\frac{\partial \underline{e}}{\partial \underline{q}}} \, \dot{\underline{q}} + \widehat{\frac{\partial \underline{e}}{\partial t}}) + \hat{\underline{N}} - \hat{M} \left(\widehat{\frac{\partial \underline{e}}{\partial \underline{q}}} \right)^{-1} \hat{\underline{f}} \qquad (4.12)$$

dans lesquels les "chapeaux" indiquent que l'on choisit des modèles (approximations, estimations...) des grandeurs considérées.

Dans cette expression générale, tous les termes, à l'exception de μ, D et G, peuvent être des fonctions de $\underline{q}$ et de t, voire même de $\dot{\underline{q}}$ pour λ, $\hat{\underline{f}}$ et $\hat{\underline{N}}$. La commande (4.12) englobe de nombreux schémas existants : commande dynamique, commande en vitesse, commande adaptative indirecte, etc.

Une analyse de la stabilité du système (4.1) rebouclé par la commande (4.12) a été réalisée par Samson [Samson 90b] dans un cadre non-linéaire, et a permis d'exhiber des conditions suffisantes de stabilité (bornitude de $\|e(t)\|$ et de plus, comportement asymptotique), qui sont principalement de deux classes :

- des conditions portant sur les **gains** de l'asservissement. Il s'agit là de réglages laissant une plus ou moins grande plage de manœuvre à l'utilisateur
- des conditions portant sur les **modèles** utilisés. Parmi celles-ci, certaines concernent les paramètres dynamiques du robot et ne sont pas trop contraignantes (en particulier en raison de la symétrie-définie-positivité de $M(\underline{q})$). Une autre, beaucoup plus critique et liée à la tâche, est :

$$\frac{\partial \underline{e}}{\partial \underline{q}} \left(\widehat{\frac{\partial \underline{e}}{\partial \underline{q}}} \right)^{-1} > 0 \qquad (4.13)$$

(au sens où une matrice A $(n \times n)$ est positive si $x^T A x > 0$, $\forall x \neq 0 \in \mathbb{R}^n$).

En effet, en combinant les équations (4.9) et (4.12) et en supposant pour simplifier l'analyse une connaissance parfaite du modèle dynamique du robot : $\hat{M} = M$, $\hat{\underline{N}} = \underline{N}$ et $\hat{\underline{f}} = \underline{f}$ (le cas général est traité en [Samson 90b]), on obtient :

$$\ddot{\underline{e}} = -\lambda \, \frac{\partial \underline{e}}{\partial \underline{q}} \left(\widehat{\frac{\partial \underline{e}}{\partial \underline{q}}} \right)^{-1} G \, (\mu D \, \underline{e} + \widehat{\frac{\partial \underline{e}}{\partial \underline{q}}} \, \dot{\underline{q}} + \widehat{\frac{\partial \underline{e}}{\partial t}}) \qquad (4.14)$$

Dans cette dernière équation apparaît la condition de positivité nécessaire à la convergence de $\underline{e}(t)$.

Nous reviendrons plus loin sur l'utilisation de cette condition essentielle qui est une façon de caractériser la **robustesse** de la tâche aux incertitudes et aux approximations effectuées.

On peut toutefois, d'ores et déjà, remarquer que, si l'on fait l'hypothèse que $n = 6$ et que le jacobien du robot $J(\underline{q})$ est parfaitement connu et inversible, on a :

$$\ddot{\underline{e}} = -\lambda \frac{\partial \underline{e}}{\partial \bar{r}} \left(\widehat{\frac{\partial \underline{e}}{\partial \bar{r}}} \right)^{-1} G \left(\mu D \, \underline{e} + \widehat{\frac{\partial \underline{e}}{\partial \bar{r}}} \, J(\underline{q}) \, \dot{\underline{q}} + \widehat{\frac{\partial \underline{e}}{\partial t}} \right) \tag{4.15}$$

La condition (4.13) se réduit alors à :

$$\frac{\partial \underline{e}}{\partial \bar{r}} \left(\widehat{\frac{\partial \underline{e}}{\partial \bar{r}}} \right)^{-1} > 0 \tag{4.16}$$

Nous allons à présent, grâce au formalisme de la redondance, construire, à partir de la tâche référencée vision $\underline{e}_1$, la fonction de tâche complète $\underline{e}$.

4.3 Tâches hybrides

Très souvent, la régulation d'une tâche référencée capteurs n'est pas l'unique objectif visé, et cette tâche doit être combinée avec une autre, par exemple un suivi d'une trajectoire. En effet, excepté le cas où la tâche référencée vision consiste à réaliser une liaison rigide entre la caméra et son environnement (on a alors $m = 6$ et on peut écrire $\underline{e} = \underline{e}_1$), les autres tâches réalisent des liaisons de classe non nulle, c'est-à-dire que des degrés de liberté de la caméra ne sont pas contraints par la liaison. Par exemple, un mouvement de translation dans l'axe d'une liaison prismatique conserve cette liaison. Aussi est-il tout à fait possible de combiner la tâche référencée vision avec une autre tâche, par exemple un suivi de trajectoires.

La spécification d'un tel problème conduit en une première étape à définir, comme nous l'avons fait, un vecteur de tâche référencée capteur, $\underline{e}_1(\underline{q}, t)$, constitué de $m \leq n$ composantes indépendantes.

Le second objectif pourrait, a priori, être représenté par un deuxième vecteur $\underline{e}_2(\underline{q}, t)$. Toutefois, il faudrait alors être capable de s'assurer que $\underline{e}_1$ et $\underline{e}_2$ peuvent être regroupés en un seul vecteur $\underline{e}(\underline{q}, t)$ **admissible**. Les deux tâches concernées doivent alors être compatibles et indépendantes, ce qui signifie grossièrement, en termes de liaisons virtuelles que le deuxième objectif peut être atteint grâce à l'ensemble des mouvements réalisables laissés libres par la liaison virtuelle

associée à la tâche référencée capteurs. Pour revenir à l'exemple d'une liaison prismatique, $\underline{e}_2$ ne peut consister qu'en un mouvement de translation dans la direction de la liaison ; tout autre mouvement est incompatible avec la réalisation de cette liaison.

En fait, on peut montrer qu'une façon plus riche de poser le problème consiste à l'immerger dans celui des *tâches redondantes*. Dans cette approche, la tâche $\underline{e}_1$ est considérée comme prioritaire, et $\underline{e}_2$ apparaît alors comme devant représenter la minimisation d'un coût secondaire, sous contrainte de réalisation parfaite de la régulation de $\underline{e}_1$. Rappelons à présent les principaux résultats obtenus en la matière [Samson 88] [Samson 90b].

4.3.1 Le formalisme de la redondance

Soit $\underline{e}_1$ la tâche principale de dimension m et de Jacobien J_1 dans SE_3 ($J_1 = \frac{\partial \underline{e}_1}{\partial \bar{r}}$), et soit h_s le coût secondaire à minimiser, de gradient $\underline{g}_s = \frac{\partial h_s}{\partial \bar{r}}$, représentant la tâche secondaire. On rappelle que le jacobien du robot est supposé inversible et que $n = 6$.

La minimisation de h_s sous la contrainte $\underline{e}_1 = 0$ nécessite la détermination du sous-espace des mouvements laissés libres par cette contrainte. Cela est équivalent à connaître le noyau de J_1, Ker J_1, le long de la trajectoire idéale, c'est-à-dire en d'autres termes, à connaître n'importe quelle matrice $m \times n$ de rang plein, W, telle que :

$$\text{Ker } W = \text{Ker } J_1 \tag{4.17}$$

le long de la trajectoire idéale du robot.

Une fois cette matrice déterminée, on peut alors montrer assez facilement [Samson 88] [Samson 90b] qu'une fonction de tâche réalisant l'objectif de minimisation de h_s sous la contrainte $\underline{e}_1 = 0$ est :

$$\underline{e} = W^+ \underline{e}_1 + \alpha(\mathbb{I}_n - W^+ W)\underline{g}_s^T \tag{4.18}$$

où :

- α est un scalaire positif,
- W^+ est la pseudo-inverse de W
- $(\mathbb{I}_n - W^+ W)$ est un opérateur de projection orthogonale sur le noyau de W, donc sur celui de J_1.

- **Remarque :** Lorsque, comme dans notre cas, $\underline{e}_1$ est construit à partir de signaux capteurs et que h_s exprime un suivi de trajectoire dans SE_3, la tâche représentée par (4.18) porte alors le nom de "tâche hybride".

L'on imagine aisément que le calcul du Jacobien $\frac{\partial \underline{e}}{\partial \bar{r}}$ associé à (4.18), éventuellement requis dans la commande, peut être complexe. La condition de positivité (4.16) s'avère alors de quelque utilité. On peut en effet montrer que si la matrice W, en plus de (4.17), satisfait la propriété :

$$J_1 W^T > 0 \tag{4.19}$$

le long de la trajectoire idéale, alors, dans des "circonstances normales" (voir [Samson 90b] pour de plus amples détails), le Jacobien de $\underline{e}$ dans SE_3 est tel que :

$$\frac{\partial \underline{e}}{\partial \bar{r}} \left(\mathbb{I}_n + \gamma \left(\mathbb{I}_n - W^+ W\right)\right) > 0 \tag{4.20}$$

le long de la trajectoire idéale, $\forall \gamma \geq \gamma_m(\alpha) \geq 0$.

La condition (4.16) est alors satisfaite par le choix :

$$\left(\widehat{\frac{\partial \underline{e}}{\partial \bar{r}}}\right)^{-1} = \mathbb{I}_n + \gamma \left(\mathbb{I}_n - W^+ W\right) \tag{4.21}$$

De plus, si α est "suffisamment petit", alors $\gamma_m = 0$, $\frac{\partial \underline{e}}{\partial \bar{r}}$ est positif, et l'on peut choisir :

$$\widehat{\frac{\partial \underline{e}}{\partial \bar{r}}} = \mathbb{I}_n \tag{4.22}$$

La loi de commande (4.12) s'écrit alors :

$$\underline{\Gamma} = -\lambda \hat{M} J^{-1}(\underline{q})\, G\, (\mu D\, \underline{e} + J(\underline{q})\, \dot{\underline{q}} + \widehat{\frac{\partial \underline{e}}{\partial t}}) + \underline{\hat{N}} - \hat{M} J^{-1}(\underline{q})\, \underline{\hat{f}} \tag{4.23}$$

4.3.2 Le cas spécifique des signaux capteurs

Appliquons à présent cette approche à l'utilisation de signaux capteurs tels que définis au paragraphe 2.1. Rappelons que la tâche principale $\underline{e}_1$ s'écrit :

$$\underline{e}_1 = C\, (\underline{s}(\bar{r}, t) - s^*) \tag{4.24}$$

Le Jacobien de $\underline{e}_1$ dans SE_3 s'écrit alors :

$$J_1 = \frac{\partial \underline{e}_1}{\partial \bar{r}} = C L^T \tag{4.25}$$

et l'on peut aisément montrer que Ker J_1 = Ker L^T puisque C doit être choisie telle que CL^T soit de rang plein. En utilisant (4.18), la tâche à réguler s'écrit finalement [Samson 90a] :

$$\underline{e} = W^+C\ (\underline{s}(\bar{r},t) - \underline{s}^*) + \alpha\ (\mathbb{I}_6 - W^+W)\ \underline{g}_s^T \tag{4.26}$$

W doit satisfaire la propriété (4.17), qui devient alors :

$$\text{Ker } W = \text{ Ker } L^T \tag{4.27}$$

Connaissant L^T, il est possible de calculer le sous-espace de torseurs S engendrés par L^T. On peut ainsi construire W, de rang plein, à partir de S : les lignes de W doivent être formées de vecteurs de base de S.

Quant à la condition de positivité (4.19), elle s'écrit :

$$CL^TW^T > 0 \tag{4.28}$$

qui permet de construire C à partir de W et L.

Le choix idéal pour vérifier cette condition est :

$$C = WL^{T+} \tag{4.29}$$

puisqu'on a $WL^{T+}L^TW^T > 0$. Malheureusement, on a vu dans le chapitre 2 que les matrices d'interaction L^T associées à des informations visuelles sont de la forme $L^T(\underline{p},\underline{P})$ où $\underline{P} = \underline{P}(\bar{r},t)$ est mesurable dans l'image et où $\underline{p} = \underline{p}(\bar{r},t)$ représente l'information tridimensionnelle des primitives considérées. Cette information tridimensionnelle étant a priori inconnue, le choix d'un modèle $\hat{L}$ de L est donc nécessaire. On choisira donc :

$$C = W\hat{L}^{T+} \tag{4.30}$$

où W est obtenue à partir de $\hat{L}$. Plusieurs possibilités nous sont offertes :

- $\hat{L} = L(\hat{\underline{p}},\underline{P})$ si l'on dispose par ailleurs d'un algorithme d'estimation de $\underline{p}$.
- $\hat{L} = L(\underline{p}^*,\underline{P})$ où $\underline{p}^*$ représente la valeur de $\underline{p}$ à la position $\underline{s} = \underline{s}^*$ réalisant $\underline{e}_1 = 0$. Des hypothèses sur la forme et sur les dimensions des primitives considérées sont alors nécessaires. Nous y reviendrons plus loin.
- $\hat{L} = L(\hat{\underline{p}}^*,\underline{P})$ où $\hat{\underline{p}}^*$ est une estimation de $\underline{p}^*$ si l'on ne dispose a priori d'aucune information tridimensionnelle.

Les choix précédents nécessitent un calcul de la matrice C à chaque pas de la boucle de commande. Cela n'est pas forcément souhaitable en raison du temps de calcul des pseudo-inverses prohibitif. Mais surtout, il faut parfois tenir compte d'un éventuel passage dans une singularité isolée (cercle centré (voir paragraphe 2.3.3), configuration particulière d'un triplet de points (voir paragraphe 3.2.1)). Un tel passage à une singularité entraîne une perte de rang de $\hat{L}^T$, une modification de $\hat{W}$, d'où une perte de dimension de la tâche principale $\underline{e}_1$. Le traitement de ces singularités peut s'avérer fort complexe et coûteux, aussi préfère-t-on avoir un modèle $\hat{L}$ constant calculé lors de la définition de la tâche. On peut choisir :

- $\hat{L} = L(\underline{p}^*, \underline{P}^*)$, noté également $L_{|\underline{s}=\underline{s}^*}$, représentant la valeur de la matrice d'interaction pour la position correspondant au motif choisi $\underline{s} = \underline{s}^*$.

La condition de positivité (4.28) n'est alors assurée, quel que soit le choix de C, que dans un voisinage de la position souhaitée $\underline{s} = \underline{s}^*$. Heureusement, cette condition n'est que suffisante et nous verrons dans les résultats proposés dans le chapitre suivant que la convergence de la loi de commande est obtenue, expérimentalement, même à partir de positions initiales très éloignées de la position souhaitée.

Ce choix de $\hat{L}$ nécessite la connaissance de $\underline{p}^*$, ce qui revient à faire des hypothèses sur la forme et sur les dimensions de la scène 3D. Ces hypothèses sont souvent effectuées lors de la définition même de la tâche, et ne paraissent alors pas trop contraignantes : en effet, si la tâche consiste par exemple en un positionnement en face d'une porte, on doit supposer qu'il y a une porte dans la scène et qu'on peut fabriquer des capteurs représentant cette porte (par exemple, ses quatre coins). De plus, si on veut positionner la caméra à une distance fixée de la porte, on doit connaître ses dimensions afin de calculer le motif exact à atteindre dans l'image.

Cependant, dans certains cas, l'hypothèse de forme est la seule condition nécessaire pour calculer le motif (c'est le cas si l'on ne fixe pas la distance à atteindre entre la caméra et l'objet). Alors, si $L^T(\underline{p}^*, \underline{P}^*)$, où la valeur de $\underline{p}^*$ est inconnue, peut s'écrire sous la forme :

$$L^T(\underline{p}^*, \underline{P}^*) = B^T(\underline{P}^*)\, D(\underline{p}^*, \underline{P}^*) \tag{4.31}$$

où D est une matrice régulière $n \times n$ positive et où B est de même rang que L, on peut choisir :

- $\hat{L} = B(\underline{P}^*)$ si la condition de positivité est satisfaite autour de $\underline{s}^*$, c'est notamment le cas si D est diagonale ou si B est de rang plein.

Enfin, si les conditions précédentes ne sont pas remplies ou si l'on n'a aucune connaissance sur la scène (ce qui peut être le cas de tâches consistant à suivre des objets inconnus, le motif dans l'image étant alors construit par une vue initiale de l'objet), on peut choisir :

- $\hat{L} = L(\underline{\hat{p}}^*, \underline{P}^*)$ où $\underline{\hat{p}}^*$ est une estimation de $\underline{p}^*$ pouvant être très grossière, mais permettant cependant la convergence de la loi de commande dans un voisinage satisfaisant autour de la position $\underline{s} = \underline{s}^*$. Il est alors difficile d'assurer la condition de positivité, même pour $\underline{s} = \underline{s}^*$ puisque la valeur de $L_{|\underline{s}=\underline{s}^*}$ est alors inconnue.

Dans les différents résultats du chapitre suivant, les hypothèses de forme et de dimension ont été faites sur les objets constituant l'environnement de la caméra. On a donc $\hat{L} = L_{|\underline{s}=\underline{s}^*}$, W constante, construite à partir de $L_{|\underline{s}=\underline{s}^*}$, et :

$$C = W L^{T+}_{|\underline{s}=\underline{s}^*} \tag{4.32}$$

Avant de présenter ces résultats, nous allons spécifier la commande simplifiée que nous utilisons en pratique et montrer que la condition (4.16) y reste essentielle.

4.4 Une commande cinématique simplifiée

Nous allons chercher à réaliser approximativement un comportement exponentiel découplé pour l'erreur de tâche, à savoir :

$$\dot{\underline{e}} = -\lambda \, \underline{e} \tag{4.33}$$

L'équation (4.6) peut s'écrire :

$$\dot{\underline{e}} = \frac{\partial \underline{e}}{\partial \bar{r}} \, T + \frac{\partial \underline{e}}{\partial t} \tag{4.34}$$

où le torseur cinématique T est relié à $\dot{\underline{q}}$ par l'équation :

$$\dot{\underline{q}} = J^{-1}(\underline{q}) \, T \tag{4.35}$$

où l'on reconnaît le jacobien inverse du robot. Nous allons à présent supposer, d'une part, que ce dernier est parfaitement connu, et, d'autre part, que la consigne accessible à l'utilisateur n'est autre que la vitesse articulaire désirée $\dot{\underline{q}}_c$, comme c'est le cas pour beaucoup de robots industriels. Alors, si l'on est capable d'assurer $\dot{\underline{q}}_c \simeq \dot{\underline{q}}$ (par exemple par l'intermédiaire d'une commande à

grands gains et dans les cas où les accélérations et les perturbations ne sont pas trop grandes), nous pouvons nous contenter de considérer comme grandeur de pseudo-commande, une "consigne de vitesse dans se_3", c'est-à-dire le torseur cinématique désiré T_c.

Idéalement, celui-ci aurait donc la forme :

$$T_c = \left(\frac{\partial \underline{e}}{\partial \bar{r}}\right)^{-1} (-\lambda\, \underline{e} - \frac{\partial \underline{e}}{\partial t}) \tag{4.36}$$

et, si $T_c = T$, l'utilisation de (4.36) dans (4.34) nous redonne bien le comportement souhaité (4.33).

Toutefois, comme nous l'avons déjà vu, seules des approximations peuvent être utilisées dans (4.36), et la forme générale de T_c utilisée en pratique sera donc :

$$T_c = \left(\widehat{\frac{\partial \underline{e}}{\partial \bar{r}}}\right)^{-1} (-\lambda\, \underline{e} - \widehat{\frac{\partial \underline{e}}{\partial t}}) \tag{4.37}$$

Avec l'hypothèse $T_c = T$, l'utilisation de (4.37) dans (4.34) nous donne alors :

$$\dot{\underline{e}} = -\lambda\, \frac{\partial \underline{e}}{\partial \bar{r}} \left(\widehat{\frac{\partial \underline{e}}{\partial \bar{r}}}\right)^{-1} \underline{e} - \frac{\partial \underline{e}}{\partial \bar{r}} \left(\widehat{\frac{\partial \underline{e}}{\partial \bar{r}}}\right)^{-1} \widehat{\frac{\partial \underline{e}}{\partial t}} + \frac{\partial \underline{e}}{\partial t} \tag{4.38}$$

En supposant pour simplifier l'analyse, bien que cela ne soit pas strictement nécessaire [Samson 90b], $\frac{\partial \underline{e}}{\partial t} = \widehat{\frac{\partial \underline{e}}{\partial t}} = 0$, on constate alors, en utilisant (4.38) dans l'expression $\frac{\partial}{\partial t}(\frac{1}{2}\|\underline{e}\|^2) = \dot{\underline{e}}^T \underline{e}$, que la condition de positivité (4.16) est suffisante à assurer la décroissance de $\|\underline{e}\|$.

Revenons à présent à la fonction de tâche donnée par l'équation (4.26). Comme la matrice W a été choisie constante, on a :

$$\frac{\partial \underline{e}}{\partial t} = W^+ \frac{\partial \underline{e}_1}{\partial t} + \alpha\, (\mathbb{I}_6 - W^+ W)\, \frac{\partial \underline{g}_s^T}{\partial t} \tag{4.39}$$

Le terme $\frac{\partial \underline{e}_1}{\partial t}$ représente la contribution d'un éventuel mouvement autonome des cibles et est en général inconnu. Un choix $\widehat{\frac{\partial \underline{e}_1}{\partial t}} = 0$ conduit, dans le cas où ce mouvement existe, à une erreur de poursuite d'amplitude décroissant en fonction de λ. Aussi, pour supprimer cette erreur de poursuite, introduit-on une estimation de la vitesse de l'objet $\widehat{\frac{\partial \underline{e}_1}{\partial t}}$. Cette estimation, notée $\underline{I}$, est actuellement obtenue à l'aide de l'équation :

$$\dot{\underline{I}} = \mu \underline{e} \tag{4.40}$$

et peut certainement être améliorée par des estimateurs plus complexes mais plus efficaces.

- **Remarque :** A l'itération $k+1$, $\underline{I}_{k+1}$ est donné par :

$$\begin{aligned} \underline{I}_{k+1} &= \underline{I}_k + \mu\ \underline{e}_k \quad \text{avec } \underline{I}_0 = 0 \\ &= \mu \sum_{j=0}^{k} \underline{e}_j \end{aligned} \tag{4.41}$$

  On voit que cet algorithme se comporte en fait comme un intégrateur et est construit pour estimer correctement une vitesse constante ($\underline{I}_{k+1} = \underline{I}_k$ si $\underline{e}_k = 0$).

De plus, si le coût secondaire choisi par l'utilisateur, permet de connaître $\frac{\partial \underline{g}_s^T}{\partial t}$, ce qui est le cas pour tout suivi de trajectoire, on peut choisir :

$$\widehat{\frac{\partial \underline{e}}{\partial t}} = W^+ \widehat{\frac{\partial \underline{e}_1}{\partial t}} + \alpha\ (\mathbb{I}_n - W^+ W)\ \frac{\partial \underline{g}_s^T}{\partial t} \tag{4.42}$$

Enfin, nous supposons dans les applications traitées que les choix des modèles d'interaction effectués permettent de satisfaire la condition (4.19), au moins autour de l'équilibre, et que α est suffisamment "petit" dans les tâches hybrides de forme (4.26) que nous utiliserons. Le choix (4.22) peut alors s'appliquer ($\widehat{\frac{\partial e}{\partial \bar{r}}} = \mathbb{I}_6$), et, finalement, la consigne en vitesse aura la forme simple :

$$T_c = -\lambda\ \underline{e} - W^+ \widehat{\frac{\partial \underline{e}_1}{\partial t}} - \alpha\ (\mathbb{I}_n - W^+ W)\ \frac{\partial \underline{g}_s^T}{\partial t} \tag{4.43}$$

Cette commande simplifiée permet de dissocier complètement le problème de la régulation de $\underline{e}$ et celui de la commande en elle-même du robot porteur de la caméra. Ainsi, on peut appliquer la consigne T_c donnée à tout type de robot capable de réaliser un asservissement en vitesse.

De plus, cette commande est extrêmement simple à implanter puisqu'elle ne nécessite, au maximum, que le réglage de trois gains :

- λ qui fixe l'amplitude du torseur cinématique désiré. Ce gain, qui dépend de la fréquence d'échantillonnage du système, ne doit pas être trop élevé pour conserver une bonne stabilité [Samson 90b],
- α qui permet de choisir la prépondérance entre les tâches principale et secondaire (si $\alpha < 1$, la tâche principale est prioritaire),
- μ qui permet à l'algorithme d'estimation du mouvement décrit plus haut de supprimer les erreurs de poursuite (en cas d'objets statiques, on choisit évidemment $\mu = 0$).

Chapitre 5

Résultats expérimentaux

Dans ce dernier chapitre, nous présentons les résultats qui ont été obtenus, soit en simulation, soit sur site expérimental, et qui portent sur la réalisation des tâches qui nous ont paru les plus significatives.

Dans tous les cas, nous nous plaçons dans le cadre le plus général où la caméra a 6 degrés de liberté. L'objectif consiste à positionner cette caméra d'une façon donnée relativement à une cible.

5.1 Résultats de simulation

Pour l'ensemble des simulations décrites dans ce paragraphe, le torseur cinématique $T_c = (V_c, \Omega_c)$ calculé par la loi de commande est appliqué à la caméra sous forme de déplacement incrémental. En effet, si M_k est la matrice homogène représentant, à l'itération k, la situation $\bar{r}$ de la caméra par rapport à un repère fixe quelconque (la représentation des situations par les matrices homogènes est décrite au paragraphe A.1.1 de l'annexe située à la fin de ce mémoire), on a [Le Borgne 87] :

$$M_{k+1} = M_k \begin{pmatrix} R_c & V_c \\ 0 & 1 \end{pmatrix} \tag{5.1}$$

avec $R_c = \cos\theta\, \mathbb{I}_3 + (1 - \cos\theta)\, \underline{u}.\underline{u}^T + \sin\theta \begin{pmatrix} 0 & -u_3 & u_2 \\ u_3 & 0 & -u_1 \\ -u_2 & u_1 & 0 \end{pmatrix}$

où $\theta = \|\Omega_c\|$ et $\underline{u}^T = (u_1\ u_2\ u_3) = \Omega_c / \|\Omega_c\|$.

Nous allons maintenant présenter les résultats de simulation de la réalisation des tâches représentant les différentes liaisons étudiées dans le chapitre 3 (liaison rigide, prismatique, rotoïde, ...). Commençons tout d'abord par la liaison rigide.

5.1.1 Positionnement

Supposons, par exemple, que l'on veuille positionner la caméra par rapport à un objet plan, repérable par quatre points formant un carré. La situation finale de la caméra par rapport à cet objet est choisie telle que l'image des quatre points forme un carré centré dans l'image. Pour réaliser cette tâche, on choisit tout naturellement comme informations visuelles les coordonnées (X_i, Y_i) de ces quatre points : $\underline{s} = (X_1, \cdots, X_4, Y_1, \cdots, Y_4)$. On en déduit $\underline{s}^* = (-a, a, a, -a, a, a, -a, -a)$ où $a = l/2z^*$, l étant la longueur des côtés du carré et z^* la distance finale souhaitée entre la caméra et l'objet.

La matrice d'interaction pour la valeur $\underline{s} = \underline{s}^*$ s'obtient immédiatement en utilisant une fois encore les équations d'Optic Flow (2.13) :

$$L^T_{|\underline{s}=\underline{s}^*} = \begin{pmatrix} -1/z^* & 0 & -a/z^* & -a^2 & -1-a^2 & a \\ -1/z^* & 0 & a/z^* & a^2 & -1-a^2 & a \\ -1/z^* & 0 & a/z^* & -a^2 & -1-a^2 & -a \\ -1/z^* & 0 & -a/z^* & a^2 & -1-a^2 & -a \\ 0 & -1/z^* & a/z^* & 1+a^2 & a^2 & a \\ 0 & -1/z^* & a/z^* & 1+a^2 & -a^2 & -a \\ 0 & -1/z^* & -a/z^* & 1+a^2 & a^2 & -a \\ 0 & -1/z^* & -a/z^* & 1+a^2 & -a^2 & a \end{pmatrix} \tag{5.2}$$

On peut remarquer que cette matrice est toujours de rang 6. Le motif choisi réalise donc bien la liaison rigide souhaitée. Appliquons à présent les résultats obtenus dans le chapitre précédent et construisons la fonction de tâche $\underline{e}$, donnée par l'équation (4.26) :

$$\underline{e} = W^+ C \, (\underline{s}(\bar{r}) - \underline{s}^*) + \alpha \left(\mathbb{I}_6 - W^+ W \right) \underline{g}_s^T \tag{5.3}$$

Comme $L^T_{|\underline{s}=\underline{s}^*}$ est de rang plein, on peut choisir pour W la matrice identité $\mathbb{I}_6$. On en déduit qu'il est impossible d'introduire une tâche secondaire dans cette tâche de positionnement (ce résultat paraît logique puisque les six degrés de la caméra sont utilisés pour réaliser une liaison rigide).

Par ailleurs, la matrice de combinaison C est choisie égale à la pseudo inverse de $L^T_{|\underline{s}=\underline{s}^*}$. On peut la calculer sans trop de difficultés et on obtient :

$$C = \begin{pmatrix} z^*c_1 & z^*c_1 & z^*c_1 & z^*c_1 & -z^*c_2 & z^*c_2 & -z^*c_2 & z^*c_2 \\ -z^*c_2 & z^*c_2 & -z^*c_2 & z^*c_2 & z^*c_1 & z^*c_1 & z^*c_1 & z^*c_1 \\ -z^*c_3 & z^*c_3 & z^*c_3 & -z^*c_3 & z^*c_3 & z^*c_3 & -z^*c_3 & -z^*c_3 \\ -c_4 & c_4 & -c_4 & c_4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & c_4 & -c_4 & c_4 & -c_4 \\ c_3 & c_3 & -c_3 & -c_3 & c_3 & -c_3 & -c_3 & c_3 \end{pmatrix} \tag{5.4}$$

$$\text{avec} \begin{cases} c_1 = -1/4 \ , \ c_2 = (1+a^2)/4a^2 \\ c_3 = 1/8a \ , \ c_4 = 1/4a^2 \end{cases}$$

La fonction de tâche $\underline{e}$ s'écrit donc :

$$\underline{e} = C \ (\underline{s}(\bar{r}) - \underline{s}^*) \tag{5.5}$$

- **Remarque :** Si les dimensions du carré sont inconnues, il est impossible de connaître la distance finale z^* entre la caméra et l'objet. Un positionnement, à une distance inconnue de l'objet, est cependant réalisable : il suffit de se fixer la longueur souhaitée $2a$ du carré dans l'image et de prendre pour C la matrice :

$$C = \begin{pmatrix} c_1 & c_1 & c_1 & c_1 & -c_2 & c_2 & -c_2 & c_2 \\ -c_2 & c_2 & -c_2 & c_2 & c_1 & c_1 & c_1 & c_1 \\ -c_3 & c_3 & c_3 & -c_3 & c_3 & c_3 & -c_3 & -c_3 \\ -c_4 & c_4 & -c_4 & c_4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & c_4 & -c_4 & c_4 & -c_4 \\ c_3 & c_3 & -c_3 & -c_3 & c_3 & -c_3 & -c_3 & c_3 \end{pmatrix} \tag{5.6}$$

En effet, la condition de positivité (4.28) est alors vérifiée pour $\underline{s} = \underline{s}^*$:

$$(CL^TW^T)_{|\underline{s}=\underline{s}^*} = \begin{pmatrix} \mathbb{I}_3/z^* & 0 \\ 0 & \mathbb{I}_3 \end{pmatrix} > 0 \tag{5.7}$$

Sur la Figure 5.1 sont regroupés les résultats de simulation de cette tâche. Les fenêtres du haut représentent la position relative de la caméra (symbolisée par une pyramide) par rapport à l'objet. Les fenêtres du milieu représentent la scène vue par la caméra. Les fenêtres de gauche correspondent à la position initiale et celles de droite à la position finale. Enfin, les fenêtres du bas représentent le comportement, en fonction du temps, de chaque composante $(s_i - s_i^*)$ (à gauche), $\|\underline{s}(\bar{r}) - \underline{s}^*\|$ (au milieu) et les six composantes de T_c (à droite). Notons que, pour cette simulation, le gain λ de la loi de commande qui régule $\underline{e}$ ($T_c = -\lambda \underline{e}$) a été fixé égal à 0.1.

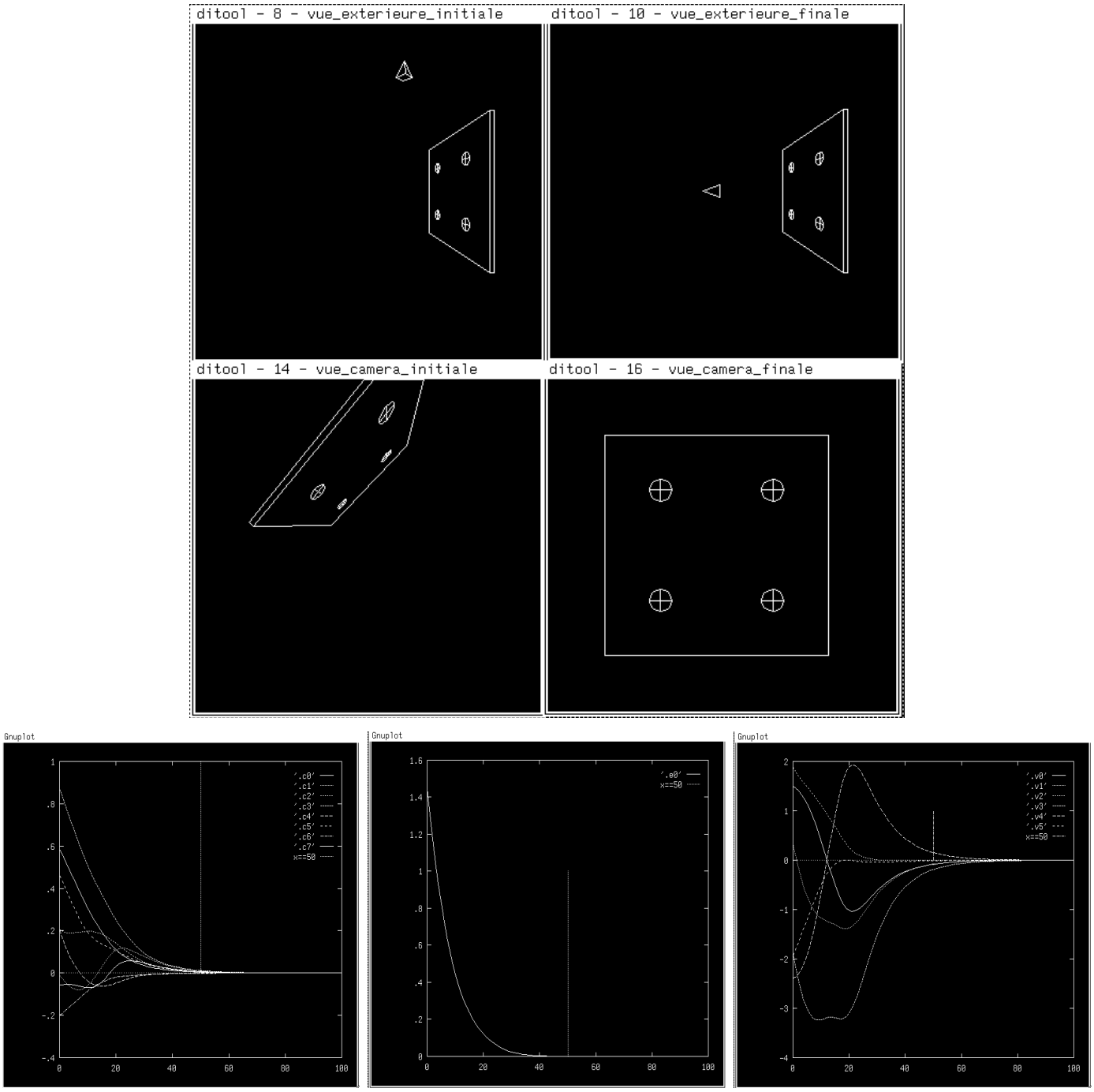


Figure 5.1 : Positionnement par rapport à un carré

La barre verticale indiquée sur les courbes à l'itération $t = 50$ représente l'instant à partir duquel la condition de positivité (4.28), s'écrivant ici $L^{T+}_{|\underline{s}=\underline{s}^*} L^T(\underline{s}) > 0$, est atteinte. On peut s'étonner, à juste titre, que cette condition ne soit pas respectée auparavant ! Cependant, une autre condition, moins forte et également suffisante, est qu'il existe une matrice A positive telle que :

$$AC L^T W^T > 0 \tag{5.8}$$

La difficulté est évidemment de trouver cette matrice et nous ne pouvons pas, malheureusement, donner ici de résultat théorique concluant concernant la convergence de la régulation des fonctions de tâches élaborées.

Sur la Figure 5.2 sont données les courbes, correspondant à la même tâche de positionnement et à la même position initiale de la caméra, mais où la matrice de combinaison C, qui entre dans la constitution de la fonction de tâche $\underline{e}$, est calculée à chaque itération de la loi de commande et a été choisie égale à $L^{T+}(\underline{s})$. La condition de positivité (4.28) est alors assurée dès la première itération. La convergence est plus rapide et les composantes de T_c ne comportent quasiment aucune oscillation. Cependant, ce choix pour C, idéal, est impossible à effectuer en pratique étant donné le fait que la matrice d'interaction dépend, en partie, de termes a priori inconnus ou au mieux estimés.

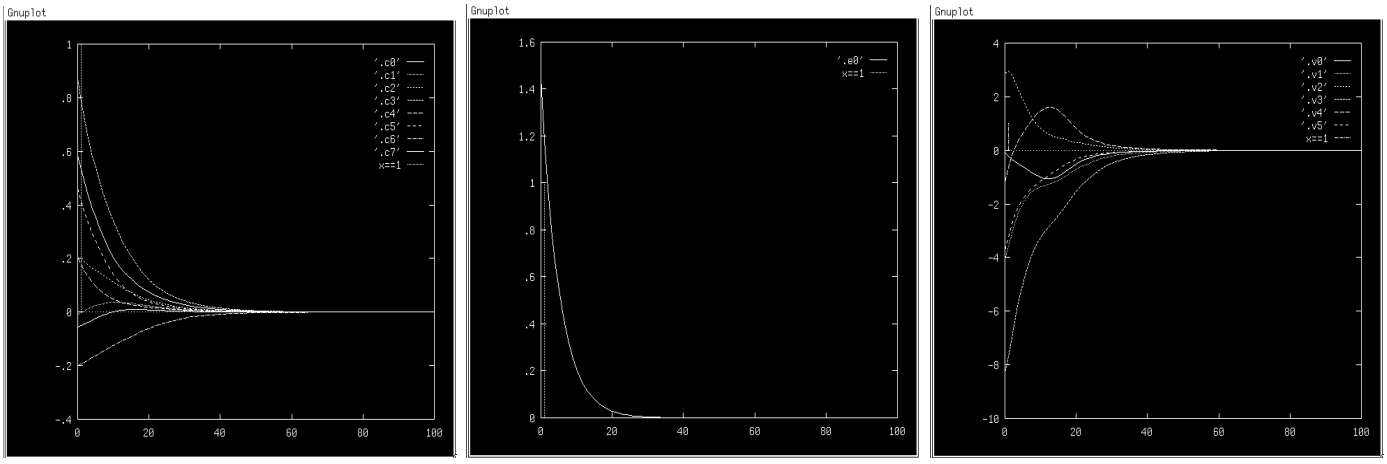


Figure 5.2 : Loi de commande idéale

On se satisfait donc du choix $C = L^{T+}_{|\underline{s}=\underline{s}^*}$ qui sera toujours choisi par la suite. Signalons que nous n'avons jamais observé, quelle que soit la tâche, quel que soit le motif choisi et quelle que soit la position initiale de la caméra par rapport à

la scène, de cas où la régulation de la fonction de tâche n'était pas correctement réalisée.

Si l'on introduit artificiellement du bruit sur les mesures effectuées dans l'image (bruit uniforme d'amplitude 2 pixels sur chaque coordonnée des points) et sur la position de la caméra (bruit uniforme d'amplitude 1 cm sur les axes de translation et de 2 degrés sur les axes de rotation) pour simuler, d'une part, d'éventuelles erreurs de mesure et, d'autre part, une mauvaise calibration ou une mauvaise commande du robot, on obtient des résultats similaires à ceux regroupés Figure 5.1, la convergence et la stabilité n'étant que peu perturbées par l'introduction des différents bruits (voir Figure 5.3).

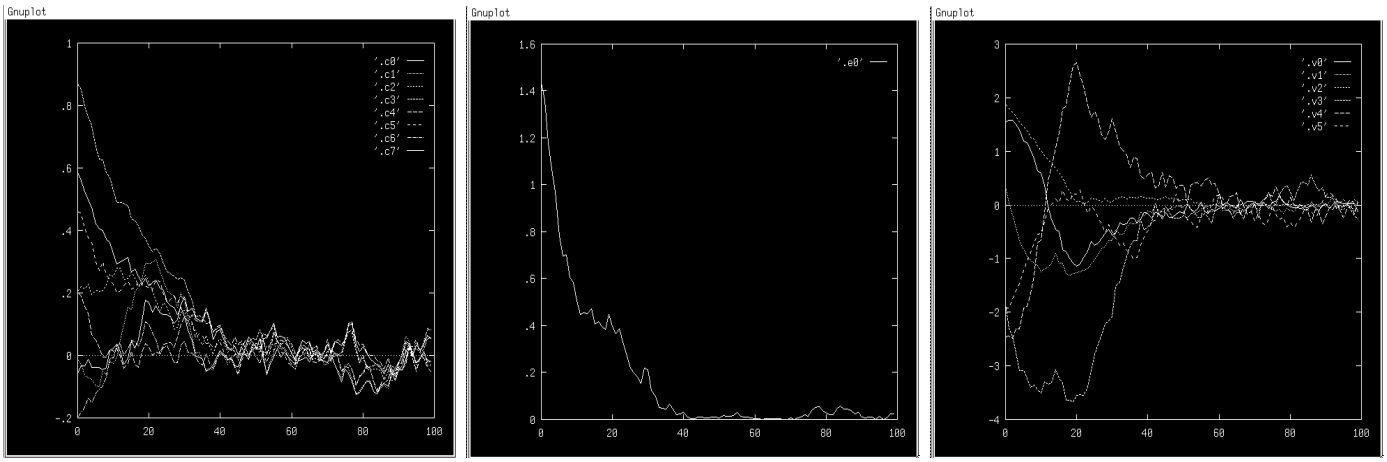


Figure 5.3 : Introduction de bruits sur les mesures

Cette même tâche de positionnement peut être réalisée sur de nombreuses autres scènes. La seule condition nécessaire pour réaliser une liaison virtuelle rigide entre la caméra et une scène étant, comme on l'a vu au paragraphe 3.2.1, de choisir un motif $\underline{s}^*$ tel que la matrice d'interaction associée soit de rang 6 et tel que l'égalité $\underline{s} - \underline{s}^* = 0$ ne soit vérifiée que pour un seul élément $\bar{r}$ de SE_3. Sur la Figure 5.4 sont représentés les résultats d'un positionnement de la caméra par rapport à un cube. Ici encore les informations visuelles choisies pour constituer le motif à atteindre sont les coordonnées de quatre points : les quatre sommets ayant pour coordonnées dans e : $(0,0,0)$, $(l,0,0)$, $(0,l,0)$ et $(0,0,l)$ où l est la longueur des arêtes de ce cube. La situation finale choisie entre la caméra et ce cube fournit $\underline{s}^*$ et la fonction de tâche est obtenue exactement de la même manière que précédemment (calcul de la matrice d'interaction associé à $\underline{s}^*$, puis de sa pseudo inverse pour obtenir la matrice de combinaison C).

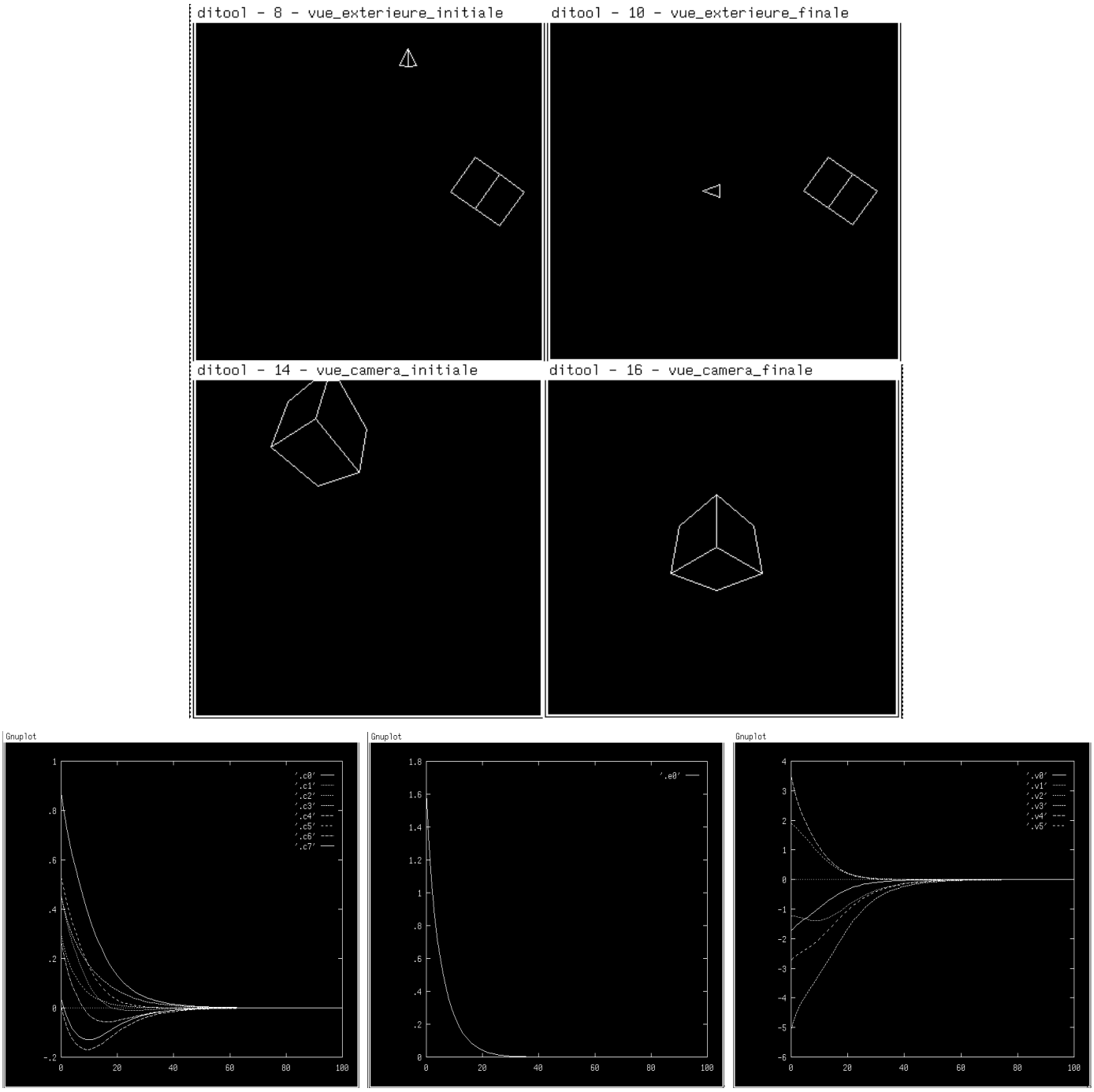


Figure 5.4 : Positionnement par rapport à un cube

5.1.2 Suivi de route

Cet exemple va nous permettre de réaliser une tâche correspondant à une autre liaison virtuelle : la liaison prismatique.

Considérons le cas où l'on souhaite positionner la caméra par rapport à une "route" de largeur l, symbolisée par trois droites parallèles coplanaires représentant les bandes latérales et centrale de cette route. La situation souhaitée de la caméra est choisie de telle sorte que :

- la caméra soit à une hauteur y^* et au milieu de la file droite de la route,
- l'axe $\vec{z}$ de la caméra coïncide avec sa direction et l'axe $\vec{y}$ avec sa verticale.

En utilisant les équations (2.48) et (2.59) obtenues lors de l'étude des droites (voir paragraphe 2.3.2), on obtient aussitôt les fonctions $h(\underline{x}, \underline{p})$ et $g(\underline{X}, \underline{P})$ représentant les équations de ces trois droites dans l'espace euclidien e et dans l'image (on utilise la représentation (ρ, θ) pour les droites 2D) :

$$\begin{array}{lll} D_1 : h_1(\bar{x}, \bar{p}^*) = \left\{ \begin{array}{l} y + y^* = 0 \\ x + l/4 = 0 \end{array} \right. & \Rightarrow & \left\{ \begin{array}{l} \theta_1^* = \arctan(-l/4y^*) \\ \rho_1^* = 0 \end{array} \right. \\ D_2 : h_2(\bar{x}, \bar{p}^*) = \left\{ \begin{array}{l} y + y^* = 0 \\ x - l/4 = 0 \end{array} \right. & \Rightarrow & \left\{ \begin{array}{l} \theta_2^* = \arctan(l/4y^*) \\ \rho_2^* = 0 \end{array} \right. \\ D_3 : h_3(\bar{x}, \bar{p}^*) = \left\{ \begin{array}{l} y + y^* = 0 \\ x - 3l/4 = 0 \end{array} \right. & \Rightarrow & \left\{ \begin{array}{l} \theta_3^* = \arctan(3l/4y^*) \\ \rho_3^* = 0 \end{array} \right. \end{array} \tag{5.9}$$

Les signaux-capteurs choisis pour réaliser cette tâche sont les paramètres représentant les trois droites : $\underline{s} = (\theta_1, \rho_1, \theta_2, \rho_2, \theta_3, \rho_3)$ et on en déduit $\underline{s}^* = (\theta_1^*, 0, \theta_2^*, 0, \theta_3^*, 0)$. On peut facilement calculer la matrice d'interaction associée à $\underline{s}^*$ en utilisant l'équation (2.63) :

$$L^T_{|\underline{s}=\underline{s}^*} = \begin{pmatrix} -\cos^2\theta_1^*/y^* & -\cos\theta_1^*\sin\theta_1^*/y^* & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & \sin\theta_1^* & -\cos\theta_1^* & 0 \\ -\cos^2\theta_2^*/y^* & -\cos\theta_2^*\sin\theta_2^*/y^* & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & \sin\theta_2^* & -\cos\theta_2^* & 0 \\ -\cos^2\theta_3^*/y^* & -\cos\theta_3^*\sin\theta_3^*/y^* & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & \sin\theta_3^* & -\cos\theta_3^* & 0 \end{pmatrix} \tag{5.10}$$

Comme on l'a vu dans le chapitre 3, ce motif réalise une liaison virtuelle prismatique. On vérifie que la matrice $L^T_{|\underline{s}=\underline{s}^*}$ est toujours de rang 5 et que Ker $L^T_{|\underline{s}=\underline{s}^*} = (0\ 0\ 1\ 0\ 0\ 0)^T$.

- **Remarque :** Si la caméra est embarquée sur un robot mobile aux trois degrés de liberté les translations sur les axes $\vec{x}$, $\vec{z}$ et la rotation autour de l'axe $\vec{y}$, cette même tâche peut être réalisée en utilisant les informations visuelles fournies par une seule droite : $\underline{s} = (\theta_1, \rho_1)$. Par contre, avec un robot à six degrés de liberté, $\underline{s}$ doit être de dimension supérieure ou égale à 5 pour réaliser une liaison virtuelle prismatique.

Appliquons à présent la démarche décrite dans le chapitre 3 pour calculer la fonction de tâche $\underline{e}$. On peut prendre pour W la matrice 5×6 :

$$W = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \tag{5.11}$$

La matrice de permutation C est choisie égale à $WL^{T+}_{|\underline{s}=\underline{s}^*}$ et, en utilisant l'équation (4.26), on obtient le vecteur de tâche $\underline{e}$, qui permet en outre de réaliser une tâche secondaire à choisir par l'utilisateur.

$$\underline{e} = W^+ W L^{T+}_{|\underline{s}=\underline{s}^*}(\underline{s}(\bar{r}) - \underline{s}^*) + \alpha\ (I_6 - W^+W)\ \underline{g}^T_s \tag{5.12}$$

- **Remarque :** On rappelle que l'ambiguïté de la représentation (ρ, θ), évoquée dans le paragraphe 2.3.2.2.2 et concernant les différents choix possibles de θ , est levée en modulant à 2π l'erreur $(\theta - \theta^*)$ et n'a donc aucune influence dans le calcul de $(\underline{s}(\bar{r}) - \underline{s}^*)$.

La tâche secondaire peut consister à décrire une trajectoire le long de l'axe $\vec{z}$, par exemple à avancer à vitesse constante V. Cette tâche secondaire correspond à minimiser le coût secondaire h_s :

$$h_s = \frac{1}{2}(z - z_0 - Vt)^2 \tag{5.13}$$

où z_0 est la composante en z initiale de la caméra. On en déduit :

$$\underline{e}_2 = z - z_0 - Vt \tag{5.14}$$

D'où :

$$g_s^T = \begin{pmatrix} 0 \\ 0 \\ z - z_0 - Vt \\ 0 \\ 0 \\ 0 \end{pmatrix} \tag{5.15}$$

Les tâches principale et secondaire sont alors compatibles et indépendantes puisqu'on obtient :

$$\underline{e} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} L^{T+}_{|\underline{s}=\underline{s}^*} \left(\underline{s}(\bar{r}) - \underline{s}^*\right) + \alpha \begin{pmatrix} 0 \\ 0 \\ z - z_0 - Vt \\ 0 \\ 0 \\ 0 \end{pmatrix} \tag{5.16}$$

En remarquant que $\frac{\partial g^T}{\partial t} = (0\ 0\ -V\ 0\ 0\ 0)$, la consigne T_c est obtenue à partir de l'équation (4.43) :

$$T_c = -\lambda e - \alpha \left(I_6 - W^+ W\right) \frac{\partial g_s^T}{\partial t} \tag{5.17}$$

d'où

$$T_c = -\lambda\, W^+ C \left(\underline{s}(\bar{r}) - \underline{s}^*\right) - \lambda\alpha \begin{pmatrix} 0 \\ 0 \\ z - z_0 - Vt \\ 0 \\ 0 \\ 0 \end{pmatrix} + \alpha \begin{pmatrix} 0 \\ 0 \\ V \\ 0 \\ 0 \\ 0 \end{pmatrix} \tag{5.18}$$

Les résultats de simulation correspondant à cette tâche de suivi de route sont présentés sur la Figure 5.5, configurée comme les figures précédentes. Les différents gains ont été fixés, pour cette simulation, aux valeurs suivantes : $\lambda = 0.1$, $\alpha = 1$ et la vitesse V a été choisie égale à 1.25 cm/s.

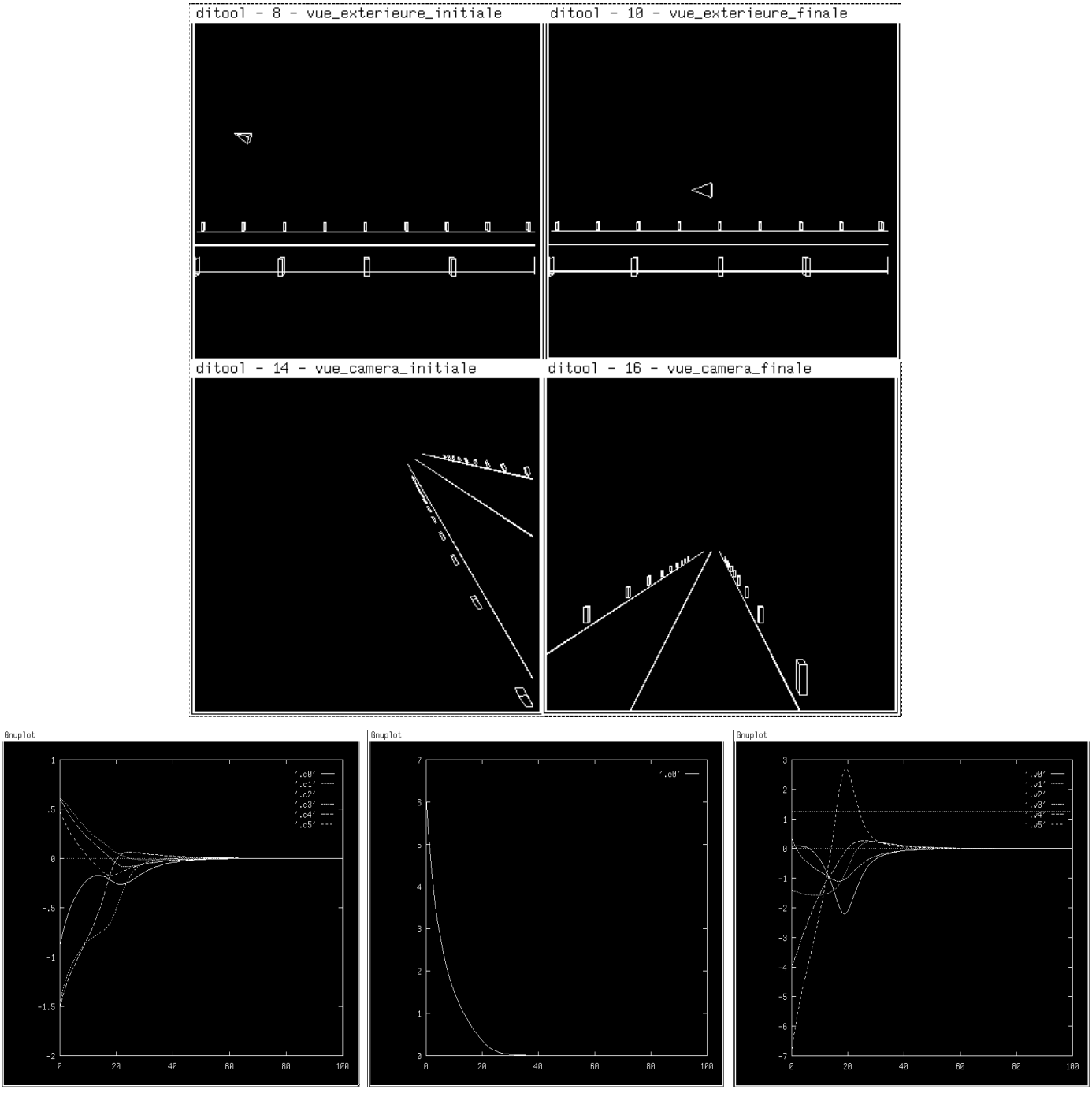


Figure 5.5 : Suivi de route

5.1.3 Positionnement à l'entrée d'un tuyau

Considérons à présent une scène constituée d'un "tuyau" symbolisé par un cylindre de rayon r et de hauteur h. La tâche consiste à positionner la caméra à l'entrée de ce cylindre de manière à obtenir dans l'image deux cercles centrés de rayon respectivement $R_1 = r/z_1$ et $R_2 = r/z_2$ où z_1 est la distance souhaitée entre la caméra et l'entrée du tuyau et où $z_2 = z_1 + h$.

Si l'on choisit comme informations visuelles les paramètres A_i définis à partir de la quadrique définissant une ellipse et résultant de la projection dans l'image d'un cercle (voir le paragraphe 2.3.3), les torseurs d'interaction associés à A_1 et A_2, sont nuls pour une situation de la caméra telle que l'image d'un cercle 3D soit un cercle centré. Aussi ne peut-on pas prendre dans le vecteur $\underline{s}$ les signaux capteurs A_1 et A_2, ceux-ci étant inefficaces pour réaliser la tâche donnée. On choisit alors : $\underline{s} = (A_{3_1}, A_{3_2}, A_{4_1}, A_{4_2}, A_{5_1}, A_{5_2},)$ où A_{i_j} est le paramètre A_i de l'ellipse j considérée. La situation souhaitée de la caméra permet de construire le motif $\underline{s}^* = (0, 0, 0, 0, -R_1^{*^2}, -R_2^{*^2})$. On obtient facilement la matrice d'interaction associée à $\underline{s}^*$ en utilisant l'équation (2.73) :

$$L^T_{|\underline{s}=\underline{s}^*} = \begin{pmatrix} 1/z_1^* & 0 & 0 & 0 & 1+R_1^{*^2} & 0 \\ 1/z_2^* & 0 & 0 & 0 & 1+R_2^{*^2} & 0 \\ 0 & 1/z_1^* & 0 & -1-R_1^{*^2} & 0 & 0 \\ 0 & 1/z_2^* & 0 & -1-R_2^{*^2} & 0 & 0 \\ 0 & 0 & -2R_1^{*^2}/z_1^* & 0 & 0 & 0 \\ 0 & 0 & -2R_2^{*^2}/z_2^* & 0 & 0 & 0 \end{pmatrix} \tag{5.19}$$

Comme on l'avait signalé dans le chapitre 3 au paragraphe 3.2.2.2, la matrice $L^T_{|\underline{s}=\underline{s}^*}$ est de rang 5 et réalise une liaison rotoïde de direction l'axe optique. Aussi choisit-on comme tâche secondaire de faire tourner la caméra autour de son axe $\vec{z}$ à une vitesse constante ω : $\underline{e}_2 = \theta_z - \theta_{z_0} - \omega t$. Les tâches $\underline{e}_1$ et $\underline{e}_2$ sont alors compatibles et indépendantes et on a, en procédant comme précédemment et en choisissant pour W la matrice 5×6 $(I_5\ 0)$.

$$T_c = -\lambda \begin{pmatrix} I_5 \\ 0 \end{pmatrix} C\ (\underline{s}(\bar{r}) - \underline{s}^*) - \lambda\alpha \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \theta_z - \theta_{z_0} - \omega t \end{pmatrix} + \alpha \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \omega \end{pmatrix} \tag{5.20}$$

où la matrice de permutation C est donnée par $C = WL^{T+}_{|\underline{s}=\underline{s}^*}$, soit :

$$C = \begin{pmatrix} c_1 & c_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & c_1 & c_2 & 0 & 0 \\ 0 & 0 & 0 & 0 & R_1^2 z_2 c_3 & R_2^2 z_1 c_3 \\ 0 & 0 & -z_1 c_4 & z_2 c_4 & 0 & 0 \\ z_1 c_4 & -z_2 c_4 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{5.21}$$

$$\text{avec} \begin{cases} c_1 = \dfrac{(1+R_2^2) z_1 z_2}{(1+R_2^2) z_2 - (1+R_1^2) z_1} \ , \ c_2 = \dfrac{-(1+R_1^2) z_1 z_2}{(1+R_2^2) z_2 - (1+R_1^2) z_1} \\ \\ c_3 = \dfrac{-z_1 z_2}{2(R_1^4 z_2^2 + R_2^4 z_1^2)} \ , \ c_4 = \dfrac{-1}{(1+R_2^2) z_2 - (1+R_1^2) z_1} \end{cases}$$

La Figure 5.6 présente les résultats obtenus en choisissant $\alpha = 1$, $\lambda = 0.1$ et $\omega = 4.5$ degrés/s.

Des résultats absolument identiques sont obtenus si l'on choisit l'autre représentation des ellipses, basée sur les moments (voir les paragraphes 2.3.3 et 3.2.2.2). En effet, on peut alors choisir comme informations visuelles $\underline{s} = (X_{c_1}, X_{c_2}, Y_{c_1}, Y_{c_2}, \mu_1, \mu_2)$ où X_{c_j} et Y_{c_j} sont les moments d'ordre 1 de l'ellipse j considérée et où $\mu_j = (\mu_{20_j} + \mu_{02_j})/2$. La matrice d'interaction associée à $\underline{s}^* = (0, 0, 0, 0, 2R_1^2, 2R_2^2)$ s'écrit alors :

$$L^T_{|\underline{s}=\underline{s}^*} = \begin{pmatrix} -1/z_1^* & 0 & 0 & 0 & -1-R_1^2 & 0 \\ -1/z_2^* & 0 & 0 & 0 & -1-R_2^2 & 0 \\ 0 & -1/z_1^* & 0 & 1+R_1^2 & 0 & 0 \\ 0 & -1/z_2^* & 0 & 1+R_2^2 & 0 & 0 \\ 0 & 0 & 2R_1^2/z_1^* & 0 & 0 & 0 \\ 0 & 0 & 2R_2^2/z_2^* & 0 & 0 & 0 \end{pmatrix} \tag{5.22}$$

La fonction de tâche et la loi de commande sont donc exactement les mêmes quelle que soit la représentation choisie des ellipses.

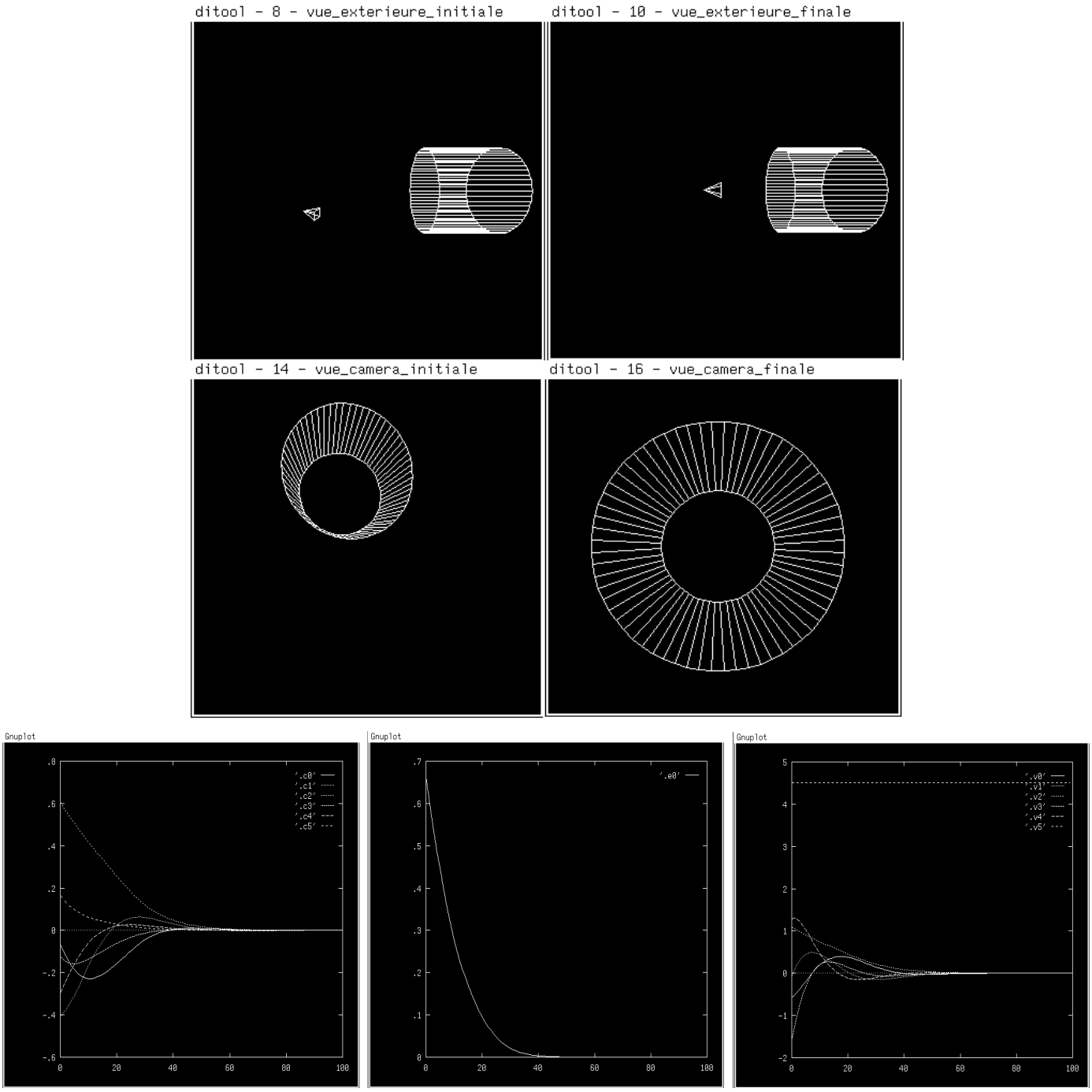


Figure 5.6 : Positionnement à l'entrée d'un tuyau

5.1.4 Positionnement par rapport à une croix

Considérons de nouveau la scène constituée d'un objet plan repérable par quatre points formant un carré. Pour réaliser une liaison virtuelle rotoïde à l'aide de cette scène, choisissons comme informations visuelles, non plus les coordonnées des quatre points, mais :

- les coordonnées (X_c, Y_c) du point d'intersection des deux "segments" d'extrémités les sommets opposés de ce carré,
- les longueurs l_i des quatre segments d'extrémités, d'une part, le point de coordonnées (X_c, Y_c) et, d'autre part, un sommet du carré.

On a donc $\underline{s} = (X_c, Y_c, l_1, l_2, l_3, l_4)$ et on choisit $\underline{s}^* = (0, 0, l, l, l, l)$. A l'aide des résultats obtenus dans le chapitre 2, on peut facilement calculer la matrice d'interaction associée à $\underline{s}^*$ et l'on obtient :

$$L^T_{|\underline{s}=\underline{s}^*} = \begin{pmatrix} -1/z^* & 0 & 0 & 0 & -1 & 0 \\ 0 & -1/z^* & 0 & 1 & 0 & 0 \\ 0 & 0 & l/z^* & l^2 \sin\alpha_1 & -l^2 \cos\alpha_1 & 0 \\ 0 & 0 & l/z^* & l^2 \sin\alpha_2 & -l^2 \cos\alpha_2 & 0 \\ 0 & 0 & l/z^* & l^2 \sin\alpha_3 & -l^2 \cos\alpha_3 & 0 \\ 0 & 0 & l/z^* & l^2 \sin\alpha_4 & -l^2 \cos\alpha_4 & 0 \end{pmatrix} \tag{5.23}$$

La matrice $L^T_{|\underline{s}=\underline{s}^*}$ dépend fortement des orientations α_i des quatre segments. Cette orientation, laissée libre par la liaison, varie si la tâche secondaire consiste, par exemple, à effectuer une rotation autour de l'axe $\vec{z}$. Aussi choisit-on cette fois-ci comme matrice de combinaison, non plus une matrice constante (il faudrait se fixer une valeur arbitraire pour α_i), mais la matrice suivante :

$$C = \begin{pmatrix} -z^* & 0 & z^* \cos\alpha_1/2l^2 & z^* \cos\alpha_2/2l^2 & z^* \cos\alpha_3/2l^2 & z^* \cos\alpha_4/2l^2 \\ 0 & -z^* & z^* \sin\alpha_1/2l^2 & z^* \sin\alpha_2/2l^2 & z^* \sin\alpha_3/2l^2 & z^* \sin\alpha_4/2l^2 \\ 0 & 0 & z^*/4l & z^*/4l & z^*/4l & z^*/4l \\ 0 & 0 & \sin\alpha_1/2l^2 & \sin\alpha_2/2l^2 & \sin\alpha_3/2l^2 & \sin\alpha_4/2l^2 \\ 0 & 0 & -\cos\alpha_1/2l^2 & -\cos\alpha_1/2l^2 & -\cos\alpha_1/2l^2 & -\cos\alpha_1/2l^2 \end{pmatrix} \tag{5.24}$$

qui vérifie $CL^T_{|\underline{s}=\underline{s}^*} = (\mathbb{I}_5\ 0)$ pour la position d'équilibre ($\alpha_2 = \alpha_1 + \pi/2$, $\alpha_3 = \alpha_1 + \pi$, $\alpha_4 = \alpha_1 - \pi/2$, quelle que soit la valeur de α_1). Ainsi, la matrice de combinaison est-elle évaluée à chaque itération.

Sur la Figure 5.7 sont présentés les résultats de simulation de cette tâche. L'écriture de la fonction de tâche s'effectue exactement de la même manière que dans l'exemple précédent de positionnement par rapport à un tuyau et la tâche secondaire a été choisie identique (rotation autour de l'axe $\vec{z}$ à une vitesse constante de 4.5 degrés/s) de même que les valeurs des gains λ et α.

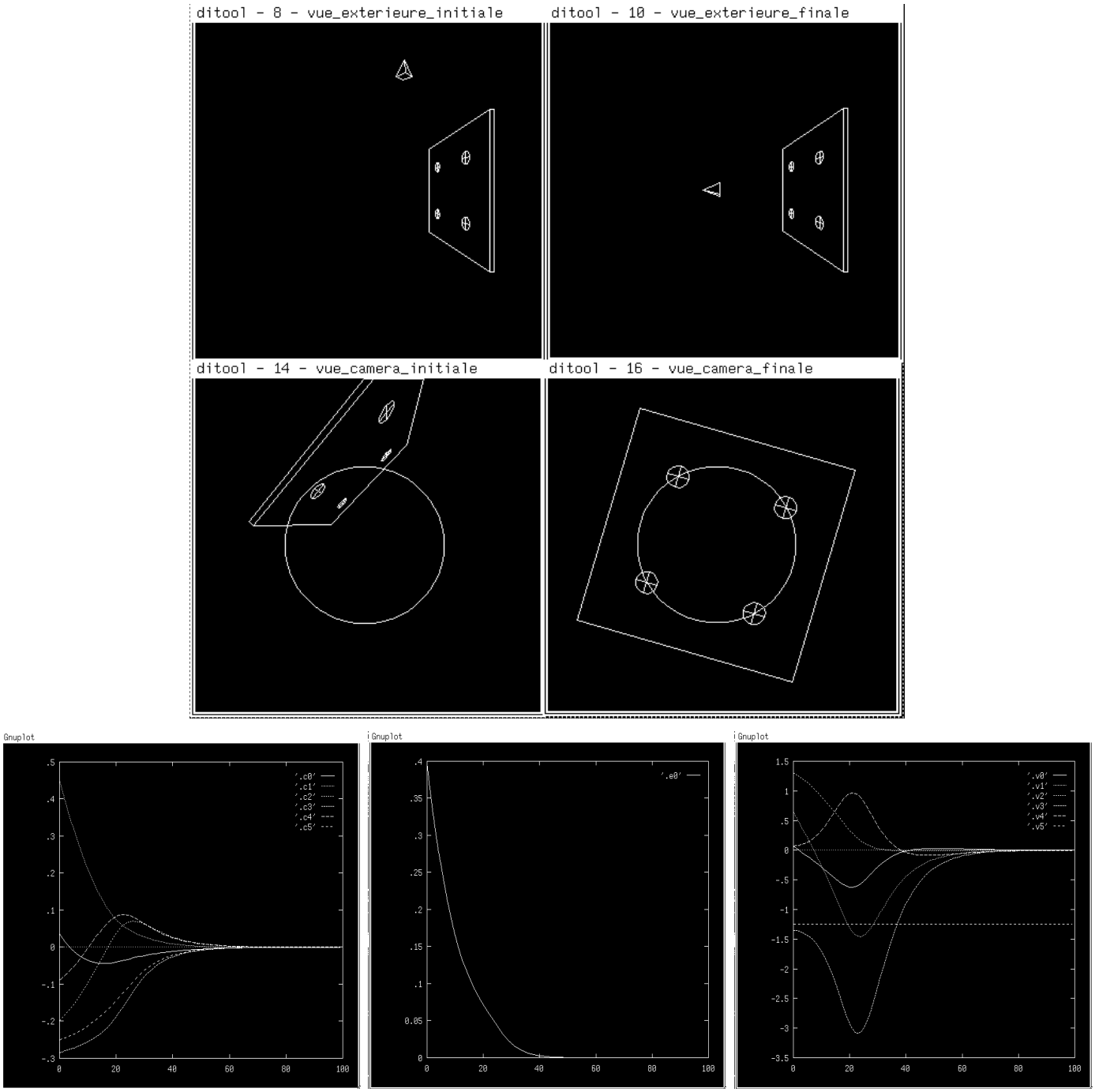


Figure 5.7 : Positionnement par rapport à une croix

5.1.5 Positionnement par rapport à un cylindre

Considérons à présent un cylindre de rayon R. Nous allons réaliser une liaison virtuelle pivot glissant d'axe $\vec{y}$ entre la caméra et celui-ci (voir paragraphe 3.2.3). A une situation désirée, le cylindre doit avoir pour équation :

$$h(\underline{x}, \underline{p}) = x^2 + (z - z^*)^2 - R^2 = 0 \tag{5.25}$$

et doit se projetter dans l'image sous la forme de deux droites d'équations :

$$\begin{cases} D_1 : & X - R/\sqrt{z^{*2} - R^2} = 0 \\ D_2 : & X + R/\sqrt{z^{*2} - R^2} = 0 \end{cases} \tag{5.26}$$

En choisissant pour $\underline{s}$ les paramètres ρ_1, θ_1, ρ_2 et θ_2 qui représentent ces deux droites, on obtient immédiatement $\underline{s}^* = (\rho^*, 0, \rho^*, \pi)$ avec $\rho^* = R/\sqrt{z^{*2} - R^2}$. La matrice d'interaction associée à $\underline{s}^*$ est facilement obtenue à l'aide des équations (3.33) :

$$L^T_{|\underline{s}=\underline{s}^*} = \begin{pmatrix} \lambda_\rho & 0 & -\lambda_\rho\rho^* & 0 & -(1+\rho^{*2}) & 0 \\ 0 & 0 & 0 & -\rho^* & 0 & -1 \\ -\lambda_\rho & 0 & -\lambda_\rho\rho^* & 0 & 1+\rho^{*2} & 0 \\ 0 & 0 & 0 & \rho^* & 0 & -1 \end{pmatrix} \tag{5.27}$$

avec $\lambda_\rho = -z^*/(z^{*2} - R^2)$. Cette matrice est de rang plein 4, aussi peut-on choisir $W = L^T_{|\underline{s}=\underline{s}^*}$ et $C = W L^{T+}_{|\underline{s}=\underline{s}^*} = \mathbb{I}_4$. On en déduit :

$$\underline{e} = W^+(\underline{s}(\bar{r}) - \underline{s}^*) + \alpha\,(I_6 - W^+W)\,\underline{g}_s^T \tag{5.28}$$

où :

$$W^+ = \begin{pmatrix} \lambda_\rho/2l & 0 & -\lambda_\rho/2l & 0 \\ 0 & 0 & 0 & 0 \\ -1/2\lambda_\rho\rho^* & 0 & -1/2\lambda_\rho\rho^* & 0 \\ 0 & -1/2\rho^* & 0 & 1/2\rho^* \\ -(1+\rho^{*2})/2l & 0 & (1+\rho^{*2})/2l & 0 \\ 0 & -1/2 & 0 & -1/2 \end{pmatrix}$$

$$(I_6 - W^+W) = \begin{pmatrix} (1+\rho^{*2})^2/l & 0 & 0 & 0 & \lambda_\rho(1+\rho^{*2})/l & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \lambda_\rho(1+\rho^{*2})/l & 0 & 0 & 0 & \lambda_\rho^2/l & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

avec $l = \lambda_\rho^2 + (1+\rho^{*2})^2$

Sur la Figure 5.8 sont présentés les résultats d'une simulation (avec $\lambda = 0.1$ et $\alpha = 1.0$) où l'on a choisi comme tâche secondaire d'avancer à vitesse constante V_x selon l'axe $\vec{x}$ de la caméra et V_y selon son axe $\vec{y}$ (respectivement à 0.5 cm/s et -0.5 cm/s). Le coût secondaire à minimiser s'écrit alors :

$$h_s = \frac{1}{2}\beta_x(x - x_0 - V_x t)^2 + \frac{1}{2}\beta_y(y - y_0 - V_y t)^2 \tag{5.29}$$

où β_x et β_y sont deux scalaires positifs qui seront fixés par la suite. On a donc :

$$\underline{g}_s^T = \begin{pmatrix} \beta_x(x - x_0 - V_x t) \\ \beta_y(y - y_0 - V_y t) \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} , \frac{\partial \underline{g}_s^T}{\partial t} = \begin{pmatrix} -\beta_x V_x \\ -\beta_y V_y \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \tag{5.30}$$

et on en déduit le torseur cinématique T_c appliqué à la caméra pour réguler la fonction de tâche $\underline{e}$:

$$T_c = -\lambda\, \underline{e} + \alpha \begin{pmatrix} \beta_x V_x (1 + \rho^{*2})^2 / l \\ \beta_y V_y \\ 0 \\ 0 \\ \beta_x V_x (1 + \rho^{*2}) \lambda_\rho / l \\ 0 \end{pmatrix} \tag{5.31}$$

Il suffit de choisir $\beta_x = l/\alpha(1 + \rho^{*2})^2$ et $\beta_y = 1/\alpha$ pour que les vitesses de translation selon les axes $\vec{x}$ et $\vec{y}$ aient les valeurs voulues V_x et V_y lorsque $\underline{e} = 0$. De plus, on peut remarquer que la tâche secondaire choisie entraîne une consigne supplémentaire de rotation autour de l'axe $\vec{y}$ à vitesse constante, fonction de V_x, et telle que la liaison pivot glissant entre la caméra et la scène soit respectée malgré le mouvement de translation le long de l'axe $\vec{x}$. On peut en effet vérifier que :

$$\begin{pmatrix} 1 + \rho^{*2} \\ 1 \\ 0 \\ 0 \\ \lambda_\rho \\ 0 \end{pmatrix} \in \text{Ker } L^T_{|\underline{s}=\underline{s}^*} \tag{5.32}$$

Sur la courbe de la Figure 5.8 représentant les composantes de T_c, on observe que la vitesse de rotation autour de l'axe $\vec{y}$ n'est constante que lorsque la liaison est respectée ($\underline{s} = \underline{s}^*$). En effet, ce degré de liberté est également contraint par la tâche principale.

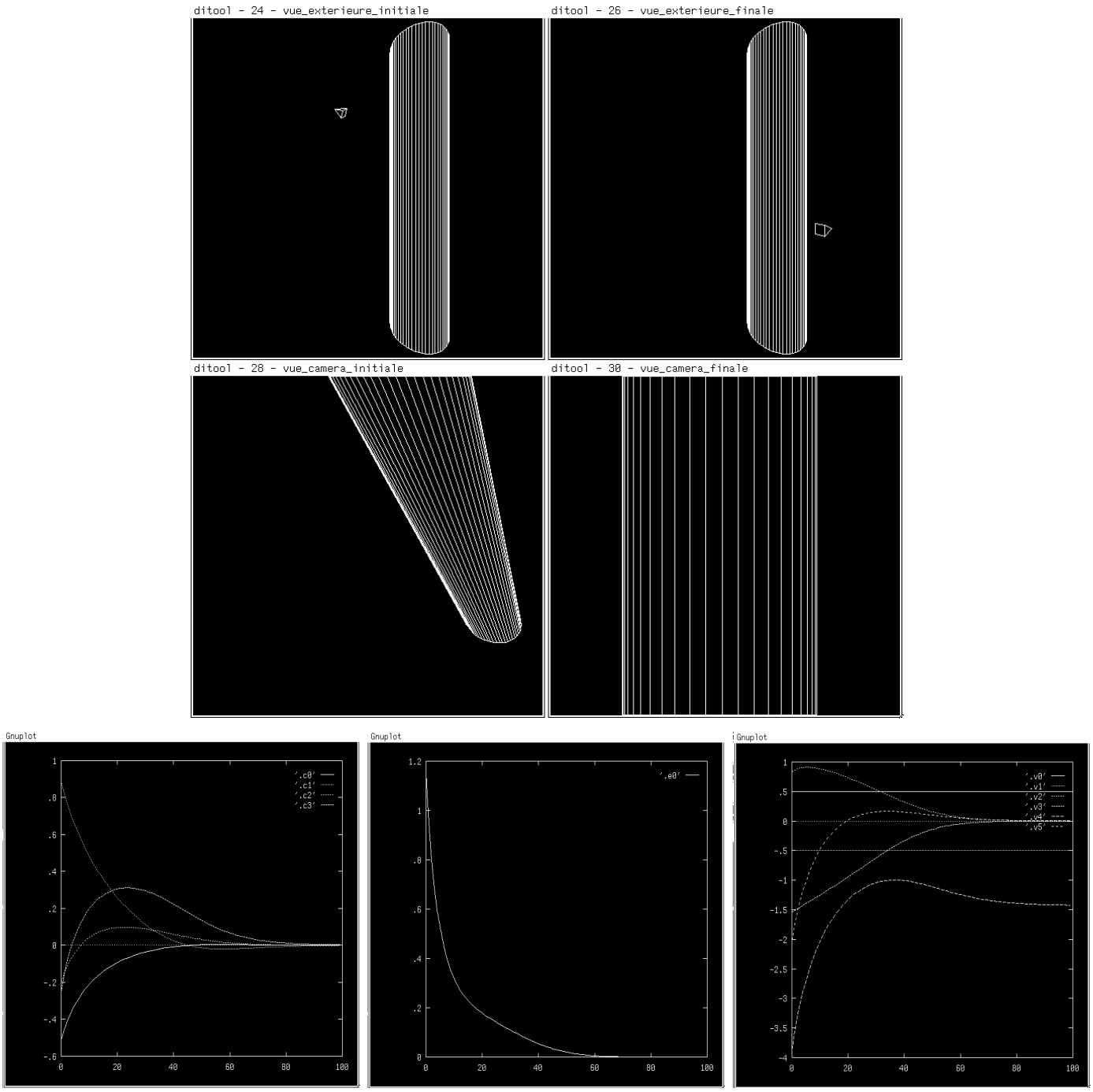


Figure 5.8 : Positionnement par rapport à un cylindre

5.1.6 Positionnement par rapport à un plan

Nous allons à présent réaliser la liaison d'appui plan à l'aide de la scène constituée d'un maillage carré (voir paragraphe 3.2.4.1). Les informations visuelles choisies sont les longueurs l_i des quatre segments les plus proches de l'origine du plan image, d'où $\underline{s}^* = (l^*, l^*, l^*, l^*)$. Lorsque $l_i = l^*$, la représentation matricielle du torseur d'interaction associé à l_i est donnée par l'équation (3.38) :

$$L_{|l_i=l^*} = \begin{pmatrix} 0 \\ 0 \\ l^*/z^* \\ l^*[X_{ci}\cos\alpha_i \sin\alpha_i + Y_{ci}(1+\sin^2\alpha_i)] \\ -l^*[X_{ci}(1+\cos^2\alpha_i) + Y_{ci}\cos\alpha_i \sin\alpha_i] \\ 0 \end{pmatrix} \tag{5.33}$$

où X_{ci} et Y_{ci} sont les coordonnées du segment i considéré et où α_i est son orientation.

Les valeurs de X_{ci}, Y_{ci} et α_i peuvent être variables durant l'exécution de cette tâche aussi la matrice de combinaison C doit-elle être évaluée à chaque itération de la boucle de commande (comme dans le cas précédent du positionnement par rapport à une croix).

Nous donnons sur la Figure 5.9 les résultats de simulation obtenus pour réaliser cette tâche. Des tâches secondaires de suivi de trajectoires ont été introduites successivement (rotation autour de l'axe $\vec{z}$, translation selon l'axe $\vec{x}$ puis $\vec{y}$,...). Sur les courbes représentant les composantes de $\underline{s} - \underline{s}^*$ et de T_c, on observe deux discontinuités (aux itérations 1 et 5). Ces dicontinuiés sont dûes au fait que, durant la régulation de $\underline{e}$, on ne prend pas toujours en compte les mêmes segments, mais ceux les plus proches de l'origine du plan image. Ainsi, un seul même segment est conservé dans $\underline{s}$ au passage d'une maille à la suivante. On s'aperçoit que ces discontinuités n'entravent en rien la bonne convergence de la loi de commande. De plus, lorsque la liaison est respectée, aucune discontinuité n'apparaît puisque tous les segments ont alors la même longueur l^*.

Lorsque la liaison n'est pas respectée, il est impossible de supprimer les discontinuités de $\underline{s} - \underline{s}^*$ (on ne peut considérer en permanence les mêmes segments puisqu'un mouvement parallèle au plan du maillage, permis par la liaison d'appui plan, peut faire sortir ces segments en dehors du champ de vue de la caméra). Cependant, si besoin est, il est possible d'atténuer les discontinuités de $\underline{e}$ et T_c en prenant en compte un nombre supérieur de segments. Par exemple, en considérant les douze segments les plus proches de l'origine du plan image, sept segments sont conservés au pasage d'une maille à l'autre.

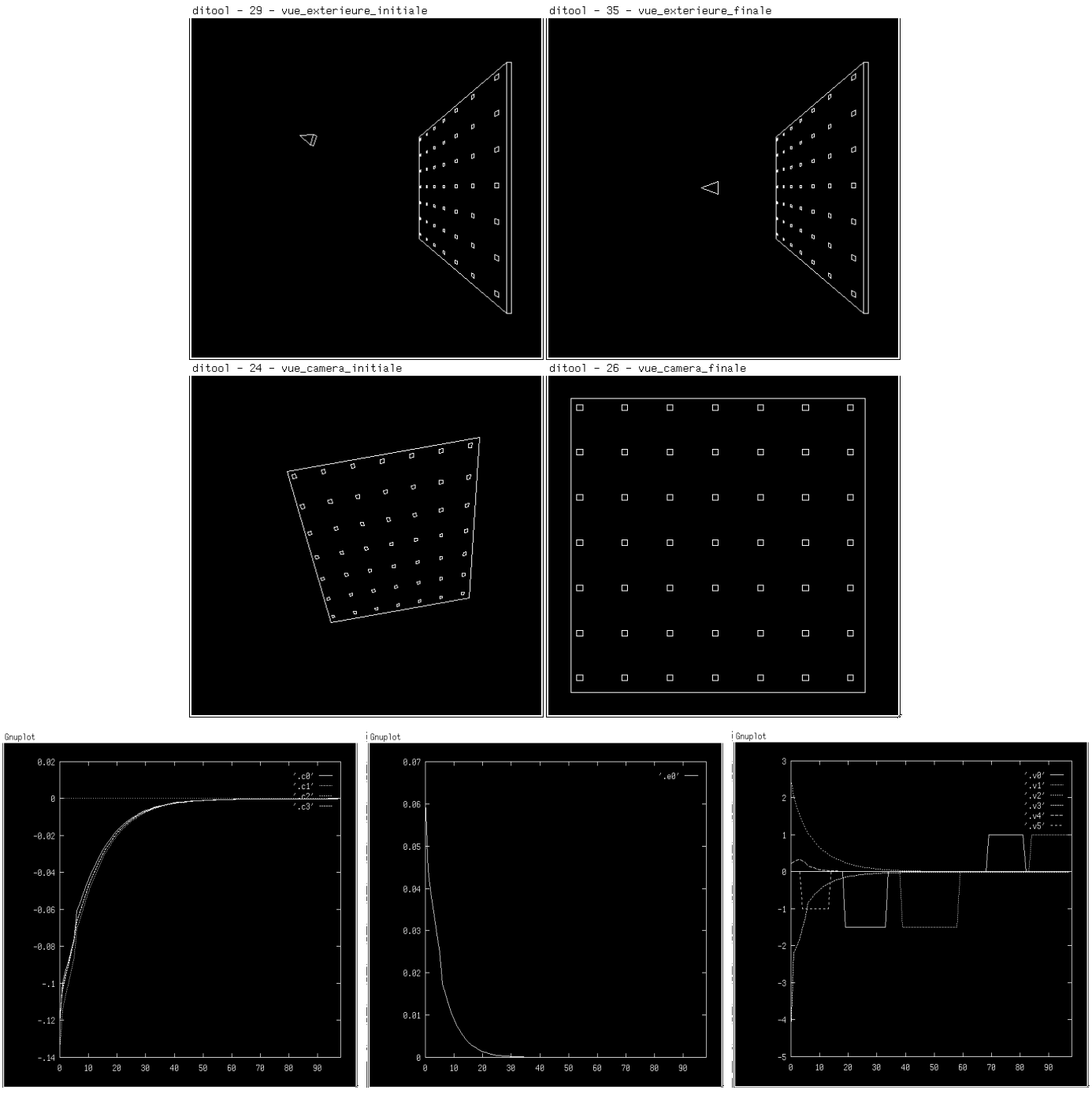


Figure 5.9 : Positionnement par rapport à un plan

5.1.7 Positionnement par rapport à une sphère

Enfin, nous donnons sur la Figure 5.10 les résultats qui ont été obtenus pour réaliser la tâche réalisant la liaison virtuelle rotule entre une sphère et la caméra (voir paragraphe 3.2.4.2) avec une fois encore $\lambda = 0.1$. On a $\underline{s} = (X_c, Y_c, \mu)$, $\underline{s}^* = (0, 0, R^2)$ et

$$L^T_{|\underline{s}=\underline{s}^*} = \begin{pmatrix} -1/z^* & 0 & 0 & 0 & -1-R^2 & 0 \\ 0 & -1/z^* & 0 & 1+R^2 & 0 & 0 \\ 0 & 0 & 2R^2/z^* & 0 & 0 & 0 \end{pmatrix}. \tag{5.34}$$

On choisit $W = L^T_{|\underline{s}=\underline{s}^*}$ et on en déduit

$$C = W L^{T+}_{|\underline{s}=\underline{s}^*} = \mathbb{I}_3 \tag{5.35}$$

On a donc, pour cette tâche réalisant la liaison rotule, $\underline{e}_1 = \underline{s} - \underline{s}^*$ et, comme pour la simulation visualisée Figure 5.10, nous n'avons spécifié aucune tâche secondaire particulière ($\underline{e}_2 = 0$). La fonction de tâche $\underline{e}$ s'écrit :

$$\underline{e} = W^+ \underline{e}_1 = \begin{pmatrix} -z^*/l & 0 & 0 \\ 0 & -z^*/l & 0 \\ 0 & 0 & z^*/2R^2 \\ 0 & (1+R^2)z^{*2}/l & 0 \\ -(1+R^2)z^{*2}/l & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} (\underline{s} - \underline{s}^*) \tag{5.36}$$

avec $l = 1 + z^{*2}(1+R^2)^2$

Nous terminons par cette tâche de positionnement par rapport à une sphère la présentation des résultats que l'on a obtenus en simulation.

Nous tenons à signaler que les figures représentant les différentes scènes et la caméra sous forme de "fil de fer" ont été obtenues grâce à un logiciel de visualisation développé à l'IRISA [Corre 88].

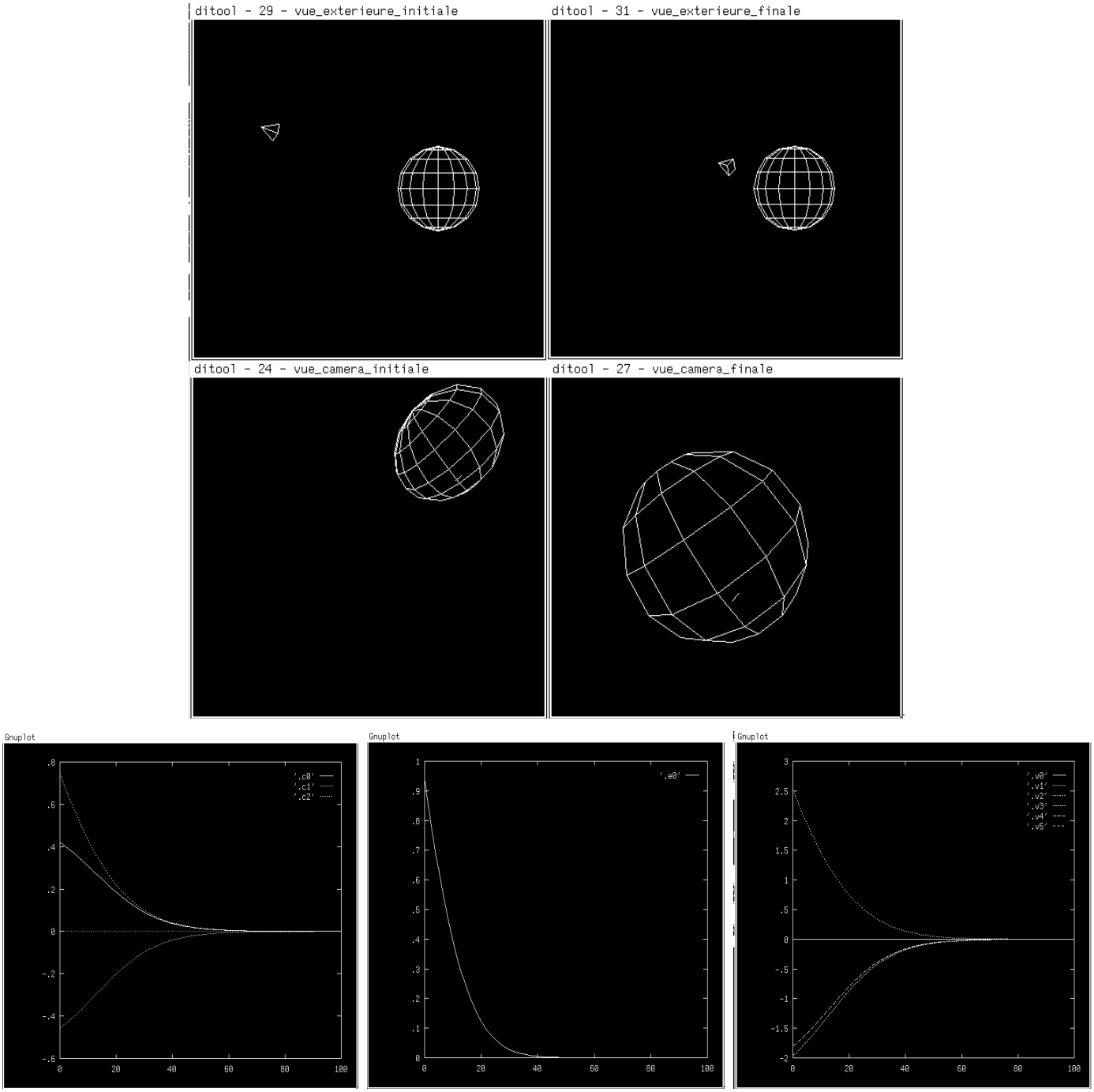


Figure 5.10 : Positionnement par rapport à une sphère

5.2 Résultats sur site expérimental

5.2.1 Description

Nous disposons à l'IRISA d'un système expérimental composé d'une caméra CCD PULLNIX embarquée sur un robot manipulateur AID à 6 degrés de liberté (voir Figure 5.11).

Figure 5.11 : Cellule expérimentale

La calibration de la caméra, qui permet de se ramener au modèle classique de projection perspective ($X = x/z$, $Y = y/z$), et l'identification de la matrice de passage entre le poignet du robot et le repère de la caméra ont été effectuées (voir Annexe A et [Chaumette 89a]). Le torseur cinématique T_c, calculé dans le repère de la caméra, peut donc être traduit sous forme de vitesse désirée dans l'espace articulaire à l'aide du jacobien inverse $J^{-1}(\underline{q})$ du robot :

$$\dot{\underline{q}}_c = J^{-1}(\underline{q})\, T_c \tag{5.37}$$

La commande numérique, qui a été développée à l'INRIA Sophia-Antipolis par Jean-Jacques Borrelly, s'avère extrêmement performante et permet, entre autres, le calcul de $J^{-1}(\underline{q})$ sur une carte à processeur 68020 avec une fréquence d'échantillonnage de 5 ms.

Par ailleurs, nous disposons d'une carte spécialisée de traitements d'images développée à l'Université de Clermont-Ferrand [Errami 86]. Celle-ci permet, entre autres, la mémorisation à la cadence vidéo, non pas de toute l'image, mais uniquement de l'adresse des pixels d'une image ayant la même propriété (dans notre cas, les pixels dont le niveau de gris est supérieur à un seuil donné).

Cette carte et la simplicité des scènes que l'on a considérées permettent de segmenter les images acquises, de calculer les coordonnées du centre de gravité de leurs composantes connexes et la consigne en vitesse T_c, transmise au processeur 68020, en moins de 20 ms.

Cette performance au niveau des temps de calcul nous a permis de réaliser, en temps réel, les tâches dont les informations visuelles sont basées sur des primitives de type point (coordonnées du point) et segments (coordonnées du milieu, longueur et orientation du segment) construits à partir de deux points. Les informations visuelles extraites de toutes les trames paires sont donc utilisées dans la loi de commande à la cadence vidéo.

Nous allons à présent donner les résultats obtenus pour la réalisation de trois tâches : la première, extrêmement simple, de positionnement par rapport à un point, la seconde et la troisième correspondant respectivement aux tâches de positionnement par rapport à une croix et par rapport à un carré, déjà étudiées en simulation.

5.2.2 Positionnement par rapport à un point

Cette tâche consiste à positionner la caméra de telle sorte que le centre de gravité de l'image d'un objet soit au centre de l'image. Les informations visuelles sont donc évidemment les coordonnées (X_c, Y_c) de ce centre de gravité. Cette tâche ne nécessite le contrôle que de deux degrés de liberté de la caméra. On a choisi d'utiliser les rotations autour des axes $\vec{x}$ et $\vec{y}$. On a alors :

$$L^T = \begin{pmatrix} X_c Y_c & -(1+X_c^2) \\ 1+Y_c^2 & -X_c Y_c \end{pmatrix} \tag{5.38}$$

où l'on n'a pris dans L^T que les composantes des torseurs d'interaction associés à X_c et Y_c correspondant aux degrés de liberté qui nous intéressent. Cette matrice L^T est entièrement connue quelle que soit la configuration de l'objet dans l'espace euclidien e puisqu'elle ne dépend que des paramètres X_c et Y_c, mesurés à chaque acquisition d'image. Aussi choisit-on comme matrice de combinaison C l'inverse de L^T, calculée à chaque itération de la boucle de commande. On obtient :

$$C = \begin{pmatrix} -X_c Y_c/(1+X_c^2+Y_c^2) & (1+X_c^2)/(1+X_c^2+Y_c^2) \\ -(1+Y_c^2)/(1+X_c^2+Y_c^2) & X_c Y_c/(1+X_c^2+Y_c^2) \end{pmatrix} \tag{5.39}$$

La fonction de tâche $\underline{e}_1$, de dimension 2, s'écrit donc :

$$\underline{e}_1 = C \begin{pmatrix} X_c - 0 \\ Y_c - 0 \end{pmatrix} \tag{5.40}$$

On en déduit :

$$T_c = \begin{pmatrix} 0 \\ 0 \\ 0 \\ -\lambda\, Y_c/(1 + X_c^2 + Y_c^2) \\ \lambda\, X_c/(1 + X_c^2 + Y_c^2) \\ 0 \end{pmatrix} \tag{5.41}$$

où λ est le gain indiquant l'amplitude voulue de la commande T_c.

Sur la Figure 5.12 sont présentées les images acquises en début, au milieu et à la fin de l'exécution de la tâche. Les courbes du bas représentent, comme précédemment, le comportement en fonction du temps des composantes $s_i - s_i^*$, dans notre cas les coordonnées (X_c, Y_c) en pixels (à gauche), l'erreur $\|\underline{s} - \underline{s}^*\|$ (au milieu) et les composantes de T_c (à droite). Pour cette expérience, λ a été fixé à 0.1. Après environ 120 itérations (soit environ 5 secondes), l'erreur sur chaque coordonnée est inférieure à 1/2 pixel.

Il faut noter que, pour un gain nettement plus fort (λ=3), la convergence est obtenue en moins d'une seconde et, pour un gain encore plus élevé, des oscillations apparaissent autour de la position d'équilibre (ce comportement est dû à l'échantillonnage, qui limite supérieurement les valeurs possibles de λ pour conserver la stabilité du système [Samson 90b]).

Nous tenons à souligner le fait que, pour cette tâche, la seule hypothèse nécessaire à son bon déroulement est la bonne extraction dans l'image des coordonnées du centre de gravité de l'image de l'objet (notamment à la position initiale !) . Aucune hypothèse sur la forme, la dimension ou la situation de l'objet par rapport à la caméra n'est faite. La loi de commande ne nécessite en aucun cas d'informations sur la distance de la caméra à l'objet.

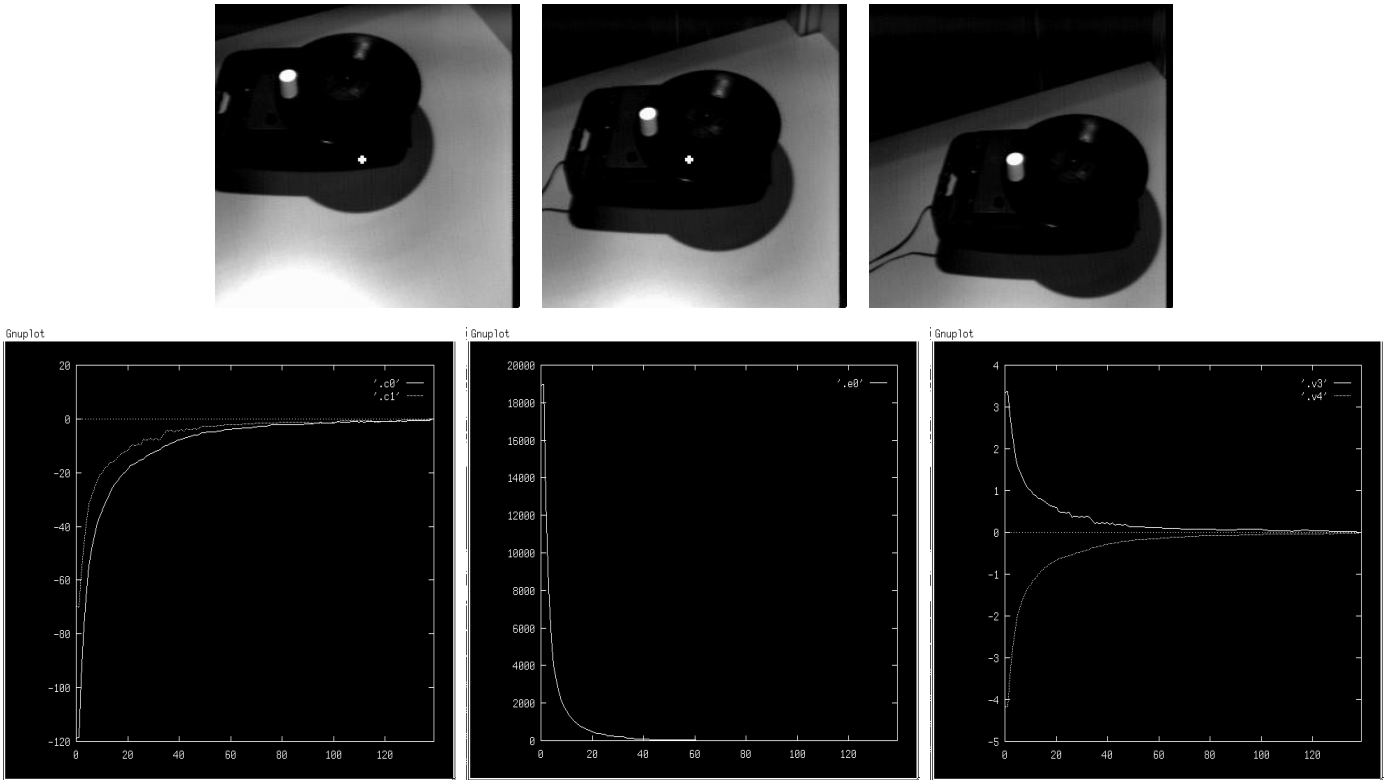


Figure 5.12 : Positionnement par rapport à un point

5.2.3 Positionnement par rapport à une croix

La tâche de positionnement par rapport à une croix, étudiée en simulation dans le paragraphe 5.1.4, a également été implantée sur notre banc expérimental. La Figure 5.13 en présente les résultats. Nous avons là supposé connaître la forme reliant les quatre points (un carré) et ses dimensions, de manière à pouvoir fixer la distance à laquelle la caméra doit se positionner par rapport à l'objet (25 cm). Cela permet également de calculer exactement le rayon du cercle auquel doit appartenir l'image des quatre points à la position finale.

Notons que λ a, une fois de plus, été fixé égal à 0.1 et que nous n'avons pas spécifié de tâche secondaire particulière. De plus, à la position finale, l'erreur sur chaque information visuelle $s_i - s_i^*$ est inférieure à 1/2 pixel.

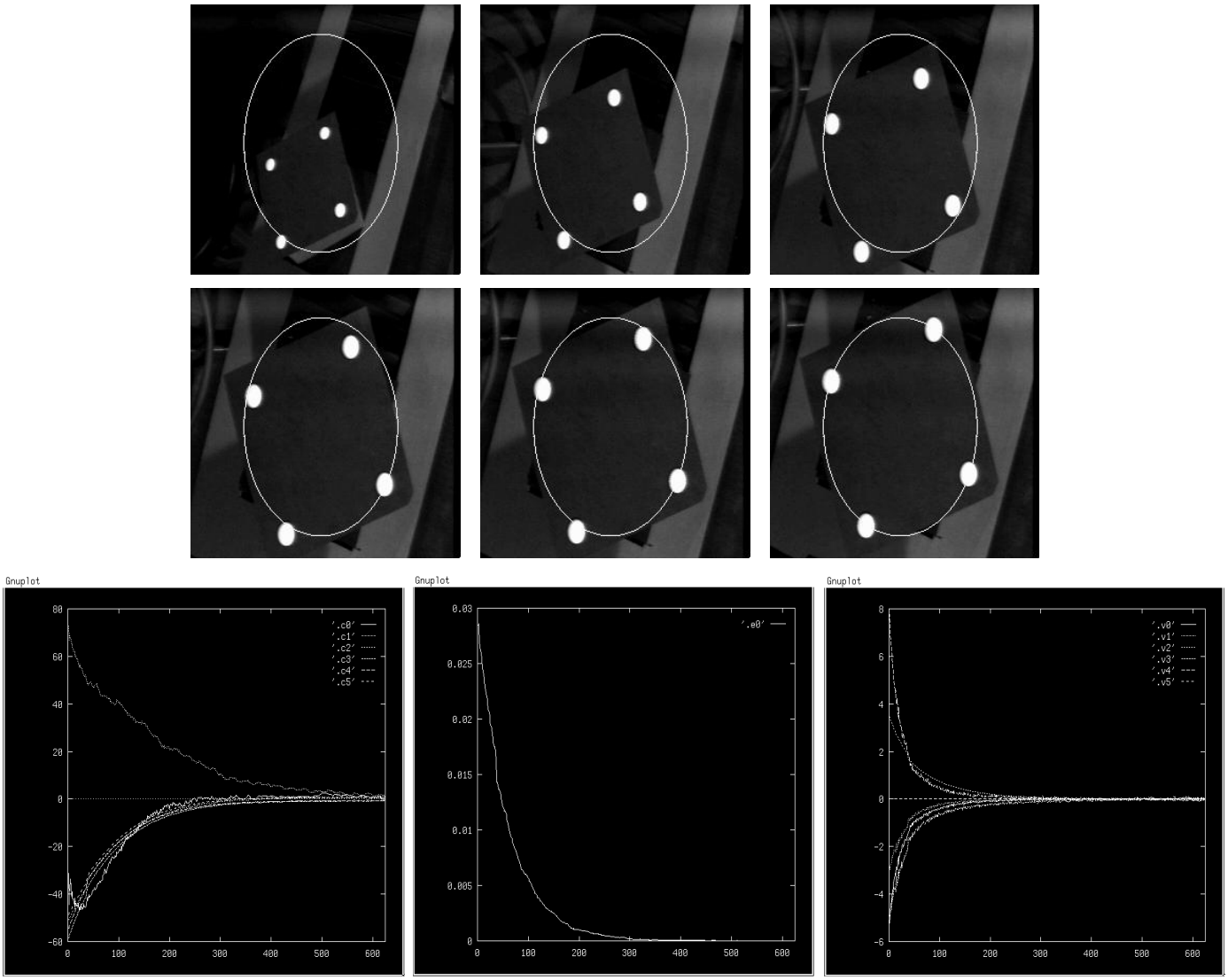


Figure 5.13 : Positionnement par rapport à une croix

5.2.4 Positionnement par rapport à un carré

Nous avons également implanté la tâche de positionnement par rapport à un carré (voir paragraphe 5.1.1) effectuée à partir de la même scène constituée de quatre points. Vu la symétrie d'un carré, quatre situations de la caméra permettent de réaliser le motif choisi. Aussi, pour fixer la situation vers laquelle la caméra doit converger, effectue-t-on une mise en correspondance entre les points observés dans l'image et ceux correspondant au motif à atteindre. Cette mise en correspondance est simplement obtenue en choisissant la solution qui minimise $\|\underline{s} - \underline{s}^*\|$.

Sur la Figure 5.14 sont représentés les résultats de cette tâche à partir d'une situation initiale de la caméra correspondant à la situation finale de la caméra pour la tâche précédente. On peut observer sur la courbe représentant les composantes de T_c le découplage de la loi de commande employée : un simple mouvement de rotation autour de l'axe optique permet d'atteindre le motif souhaité.

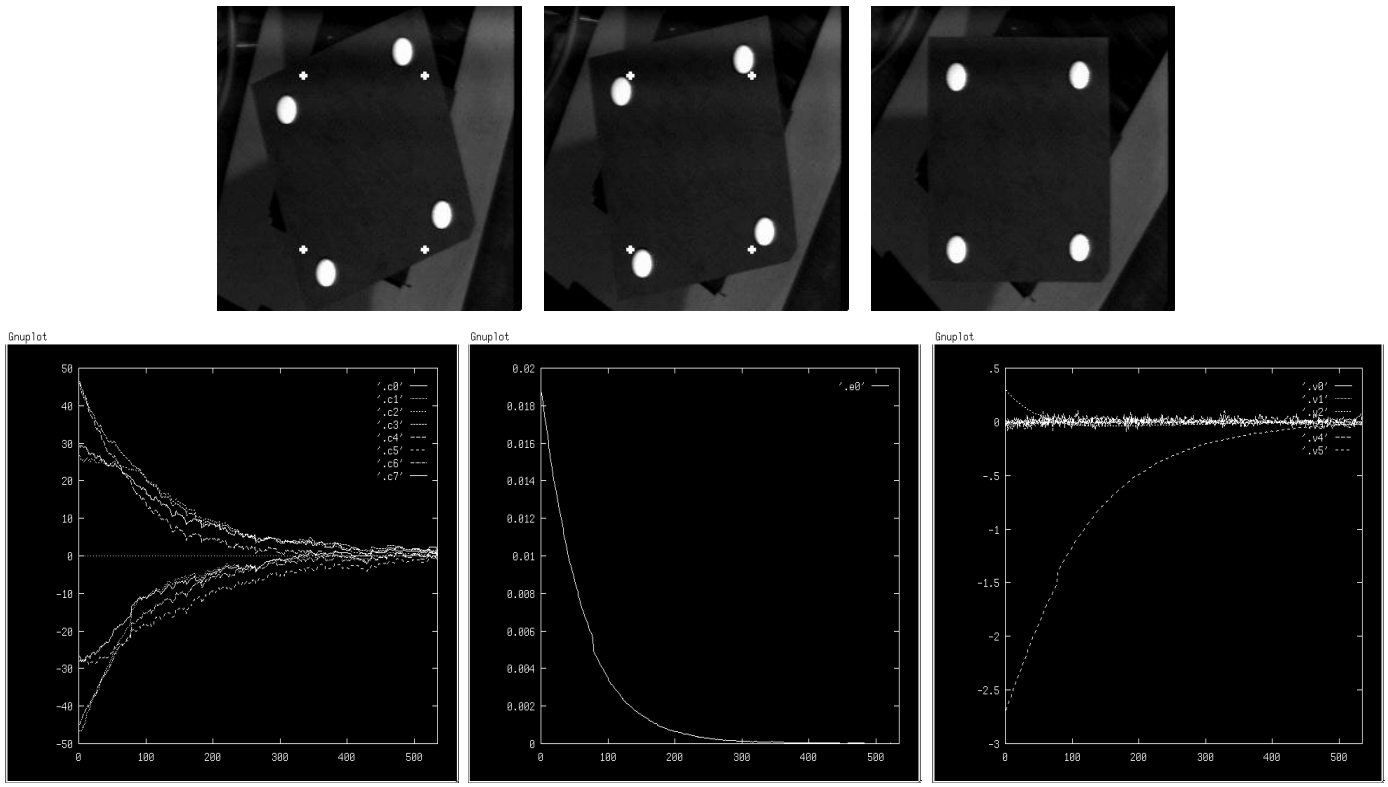


Figure 5.14 : Positionnement par rapport à un carré

La Figure 5.15 présente une séquence d'images acquises lors de l'exécution de la même tâche, mais pour une situation initiale de la caméra fortement distante de la situation souhaitée. Les courbes obtenues sont similaires à celles réalisées en simulation et assez peu bruitées.

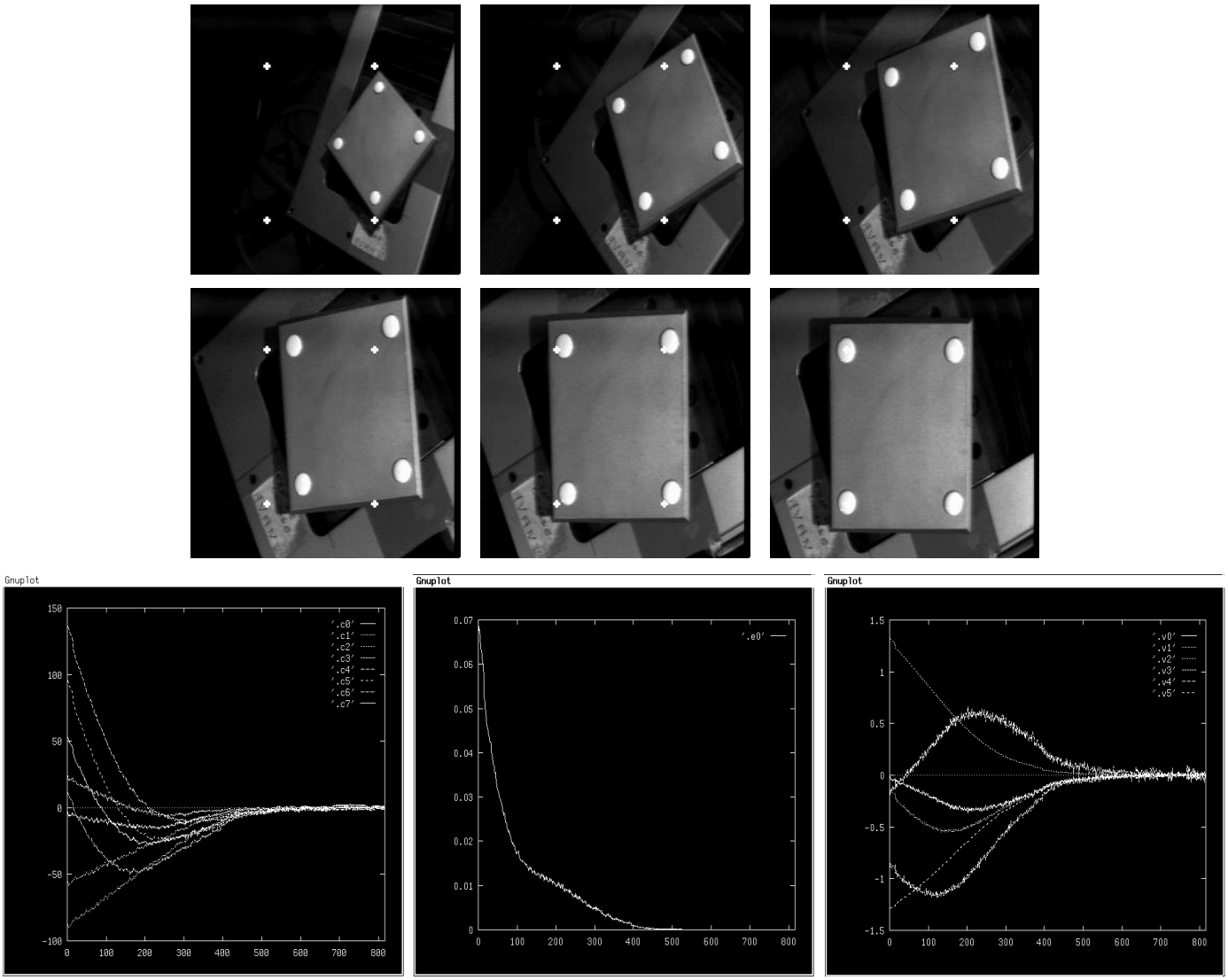


Figure 5.15 : Positionnement par rapport à un carré

On peut bien évidemment spécifier une autre situation à atteindre entre la caméra et l'objet. Comme les dimensions du carré sont supposées connues, il est possible de calculer exactement le motif à atteindre dans l'image. Sur la Figure 5.16 sont visualisés les résultats pour un positionnement de la caméra à 20 cm et avec une orientation de 45 degrés de l'objet à partir de la situation d'équilibre pour le positionnement précédent.

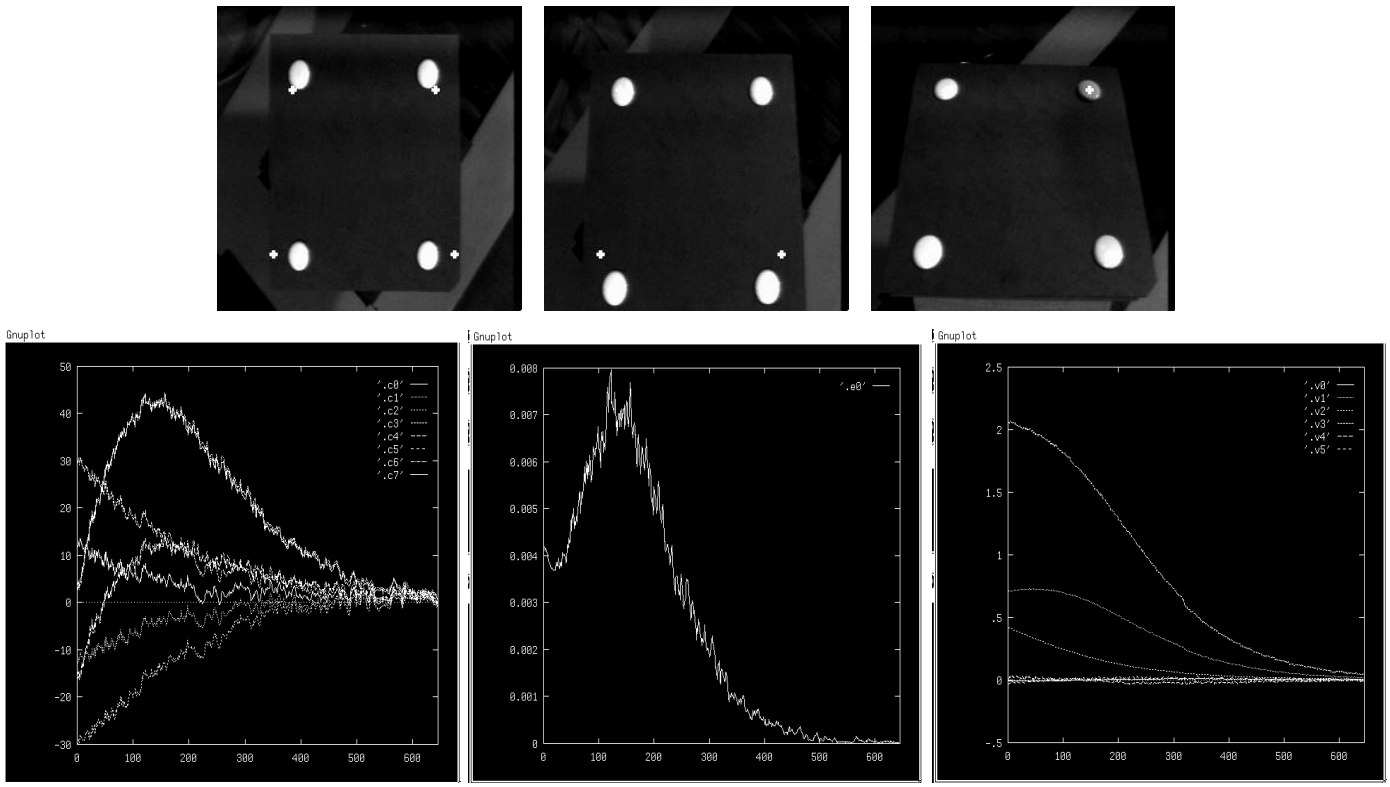


Figure 5.16 : Autre positionnement

Sur cet exemple où la convergence est normalement obtenue, on s'aperçoit cependant que la loi de commande peut parfois entraîner qu'un ou plusieurs des points de l'objet sortent du champ de vue de la caméra (en d'autres termes, la trajectoire des points dans l'image n'est pas forcément en permanence à l'intérieur de l'image), interrompant ainsi le bon déroulement de la tâche. Pour

tenter d'interdire la perte d'informations visuelles durant l'exécution de cette tâche, on peut introduire une tâche complémentaire, prioritaire, qui est, par exemple, que l'intersection des deux segments formés des points opposés du carré soit au centre de l'image. Cette tâche devient, d'après le formalisme de la redondance énoncé dans le chapitre précédent, la tâche principale alors que la tâche de positionnement par rapport au carré devient la tâche secondaire.

On choisit, pour réaliser la tâche principale, d'utiliser exactement la même loi de commande que dans le paragraphe 5.2.1 où la tâche consistait à positionner un objet au centre de l'image à l'aide des deux degrés de liberté de la caméra constitués des rotations autour des axes $\vec{x}$ et $\vec{y}$. La fonction de tâche complète s'écrit alors, d'après l'équation (4.18) :

$$\underline{e} = W^+ \underline{e}_1 + \alpha \, (\mathbb{I}_6 - W^+ W) \underline{g}_s^T \tag{5.42}$$

avec :

$$\underline{e}_1 = \begin{pmatrix} Y_c/(1+X_c^2+Y_c^2) \\ -X_c/(1+X_c^2+Y_c^2) \end{pmatrix}$$

$$W = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

$$W^+ = W^T$$

$$\mathbb{I}_6 - W^+ W = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\underline{g}_s^T = L^{T+}_{|\underline{s}=\underline{s}^*} \, (\underline{s} - \underline{s}^*)$$

Sur la Figure 5.17 sont donnés les résultats de l'exécution de cette tâche. On a fixé $\alpha = 0.1$ afin de rendre $\underline{e}_1$ nettement prioritaire. Le gain λ de la loi de commande $T_c = -\lambda \underline{e}$ a, une fois n'est pas coutume, été choisi égal à 1 (le gain $\lambda\alpha$ qui agit sur la tâche secondaire, est ainsi égal à 0.1). Sur les courbes obtenues, on peut remarquer que l'erreur $\|\underline{s} - \underline{s}^*\|$ est maintenant toujours décroissante mais que les composantes en vitesse des rotations autour des axes $\vec{x}$ et $\vec{y}$ sont fortement bruitées en raison du plus fort gain qui agit sur ces axes ($\lambda = 1$).

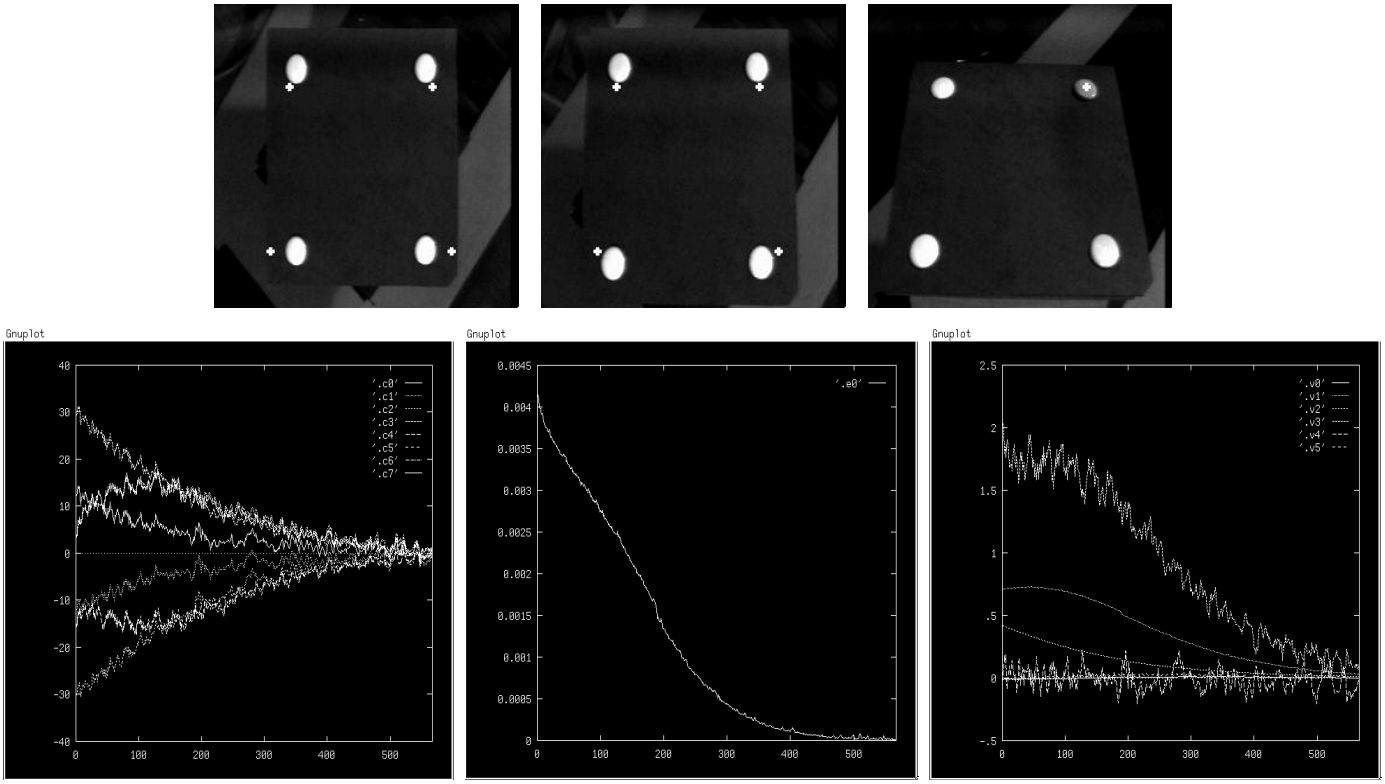


Figure 5.17 : Introduction d'une tâche supplémentaire

5.2.5 Suivi d'objets mobiles

Comme en commande référencée vision, on spécifie les tâches par la liaison virtuelle qui relie la caméra à la scène, les tâches telles que suivi d'objets mobiles sont tout à fait réalisables. Cependant, comme nous l'avons signalé dans le chapitre précédent, une légère adaptation de la loi de commande est nécessaire.

La Figure 5.18 représente les courbes en fonction du temps des composantes de $\underline{s} - \underline{s}^*$ et de T_c (avec $\lambda = 1$) lorsque, à partir d'une situation initiale correcte entre la caméra et l'objet considéré précédemment (itération 0 à 50) :

- celui-ci effectue une translation à vitesse constante (itération 50 à 620),
- s'arrête (itération 620 à 680),
- effectue le déplacement inverse à une vitesse constante plus élevée (itération 680 à 820),
- et enfin s'arrête de nouveau (itération 820 à 900).

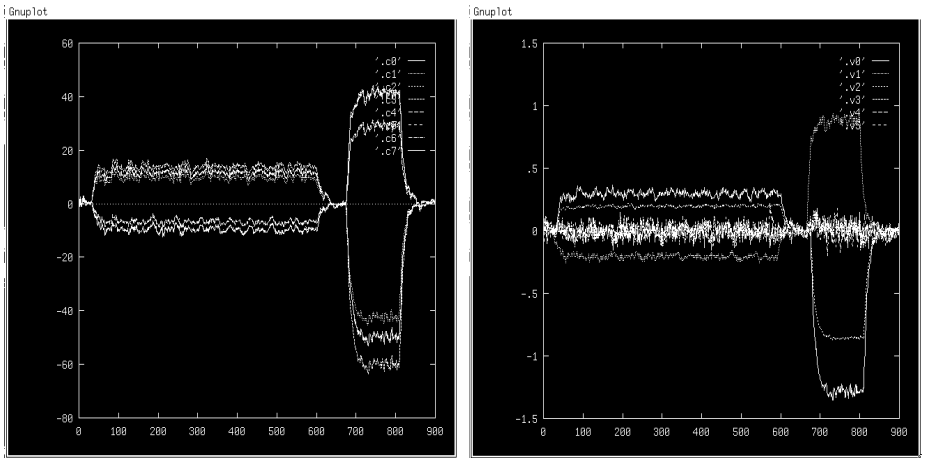


Figure 5.18 : Suivi

Sur la courbe de gauche, représentant les composantes de $\underline{s} - \underline{s}^*$, on constate une erreur de poursuite, constante pour une vitesse constante de l'objet, augmentant en fonction de l'amplitude du mouvement de la cible, et qui disparaît au bout de quelques itérations lorsque l'objet s'immobilise. Cette erreur de poursuite est également décroissante en fonction de λ. Cependant, pour conserver la stabilité du système, il est impossible de choisir une valeur de λ trop élevée (dans le cas présent, la limite de stabilité correspond à $\lambda \simeq 3.0$).

Aussi, pour supprimer cette erreur de poursuite, introduit-on dans la consigne T_c une estimation de la vitesse de l'objet $\frac{\partial \underline{e}}{\partial t}$:

$$T_c = -\lambda \underline{e} - \widehat{\frac{\partial \underline{e}}{\partial t}} \tag{5.43}$$

où $\widehat{\frac{\partial \underline{e}}{\partial t}}$, noté $\underline{I}_k$ à l'itération k, est donné (voir paragraphe 4.4) par l'algorithme d'estimation (4.41) .

$$\underline{I}_{k+1} = \mu \sum_{j=0}^{k} \underline{e}_j \tag{5.44}$$

On a donc :

$$T_{ck} = -\lambda\, \underline{e}_k - \mu \sum_{j=0}^{k} \underline{e}_j \tag{5.45}$$

Sur la Figure 5.19 sont représentées les courbes obtenues lorsque $\lambda = 1$ (comme précédemment) et $\mu = 0.1$. On constate que l'erreur de poursuite est quasiment nulle pour une vitesse de l'objet constante, même de forte amplitude. Par contre, cette erreur dépend fortement des changements de vitesse de l'objet : ce résultat est tout à fait logique puisque l'estimateur de vitesse est construit

pour estimer correctement une vitesse constante de l'objet. Sur la Figure de droite représentant les composantes de la commande T_c, on observe un dépassement aux changements de consignes provoqué par l'"intégrateur" que constitue l'estimateur employé.

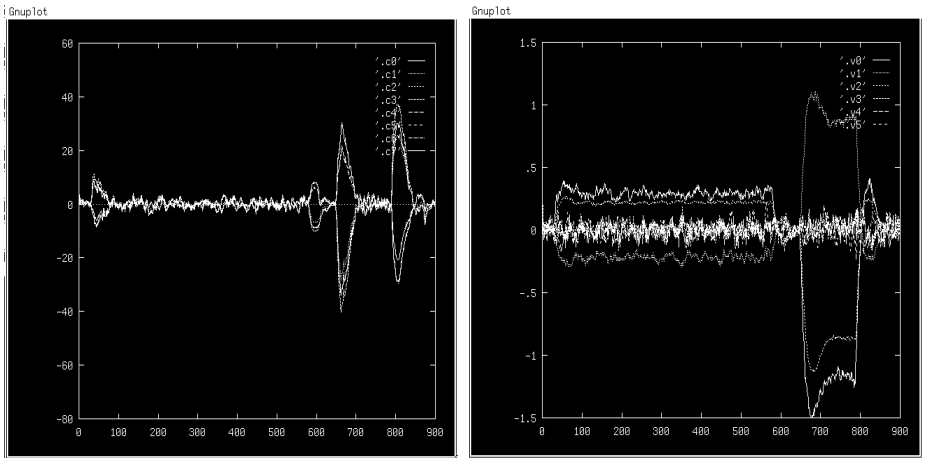


Figure 5.19 : Suivi avec estimation du mouvement

Enfin, sur la Figure 5.20 est donnée l'estimation $\frac{\widehat{\partial e}}{\partial t}$ du mouvement 3D de l'objet. On constate que cette estimation est peu bruitée et que les dépassements qui se produisent à chaque changement de vitesse sont atténués par rapport à ceux observés sur T_c (ceci est encore dû à l'effet de l'intégrateur).

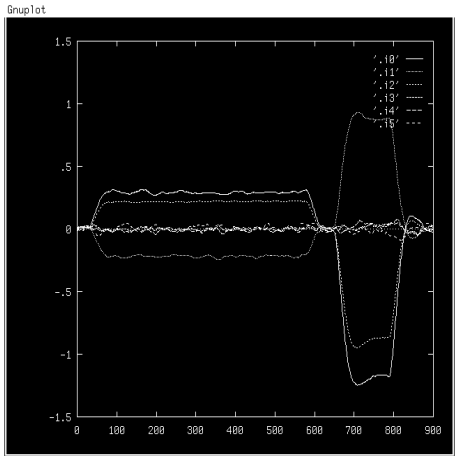


Figure 5.20 : Estimation du mouvement

Nous tenons à rappeler qu'aucune hypothèse sur le type de trajectoire de l'objet n'a été formulée. De plus, ces résultats peuvent très certainement être améliorés, notamment par la construction d'un estimateur plus performant.

Conclusion

Nous avons essayé de montrer dans cet ouvrage que l'évolution des techniques permet d'envisager, aujourd'hui, de contrôler l'interaction entre un robot et son environnement physique par un schéma de commande en boucle fermée directement sur des informations issues de capteurs de vision. L'exploitation d'une telle approche en robotique a nécessité la réponse à deux questions :

- Quels types de tâches robotiques sont concernés par cette approche ? Avec le corollaire suivant, peut-on définir des *classes de tâches génériques* et comment les spécifier ?
- Comment, pour une classe de tâches donnée, peut-on construire une commande capable de la réaliser en garantissant de bonnes propriétés de convergence et de robustesse?

Pour répondre au premier point, rappelons l'hypothèse fondamentale qui soustend notre démarche : seules des modifications géométriques sont susceptibles de faire varier la sortie du capteur. De ce fait, les tâches que nous sommes capables d'appréhender avec ce type d'approche sont celles qui nécessitent un contrôle d'interactions de nature géométrique entre la caméra et l'environnement. Typiquement, cela se traduit par le contrôle de la situation relative entre un repère lié au capteur et un repère lié à l'environnement. Un formalisme, issu de la Mécanique, a déjà été utilisé en robotique d'assemblage avec des capteurs proximétriques. Il consiste à spécifier cette situation en termes de *liaisons* (prismatique, rotoïde, rotule, ...) représentant les contraintes désirées entre les deux repères et caractérisant les degrés de liberté nécessaires à une réalisation correcte de la tâche. Ce formalisme permet de définir les classes de tâches élémentaires (positionnement, suivi, ...) qui sont utilisées pour décrire une application plus complexe. Dans l'approche que nous avons développée, nous avons établi pour

un certain nombre de primitives géométriques dans la scène (points, droites, cercles, sphères et cylindres), la structure et les propriétés du torseur d'interaction permettant ainsi d'associer à un motif de la scène une liaison définie dans le formalisme précédent.

Le deuxième point, concernant l'aspect "mise en œuvre de commandes robustes", a été réalisé en s'appuyant sur les résultats déjà acquis sur la commande des robots manipulateurs et la commande référencée capteurs. Les notions de *redondance* et de *tâches hybrides* y sont parfaitement intégrées.

Quelques questions théoriques restent cependant à examiner :

- Il serait satisfaisant de trouver une signification physique aux configurations entraînant une singularité isolée (notamment le cas mystérieux du triplet de points).
- Il serait intéressant de disposer d'un critère de construction de motifs, choisissant parmi l'ensemble des informations visuelles utilisables pour réaliser une tâche, les plus performantes, l'approche consistant à les employer toutes n'étant pas forcément la meilleure. Ce critère devrait aussi bien inclure les aspects de traitements d'images (robustesse d'extraction vis à vis des bruits de mesure, ...) que des aspects liés à la commande (découplage, sensibilité, ...).
- Il serait très réconfortant de prouver que les lois de commande employées convergent, et non pas de le constater expérimentalement !

Par ailleurs, des études plus poussées sur l'aspect vision restent à accomplir afin de pouvoir confronter la commande référencée vision au cadre de scènes complexes. Détaillons-en brièvement les grandes lignes.

Il ne faut pas oublier qu'une commande en boucle fermée sur des informations visuelles implique une compatibilité entre la fréquence d'acquisition des mesures fournies par la caméra et la fréquence d'échantillonnage du processus à commander. Bien que les informations visuelles utilisées soient de bas niveau (points, droites, ellipses, ...), il sera nécessaire d'utiliser des architectures spécialisées pour les extraire en Temps Réel (c'est à dire à la fréquence vidéo).

De plus, des algorithmes portant sur l'extraction et le suivi d'attributs dans une séquence d'images sont à mettre en œuvre. Ces traitements, spécifiques à chaque type d'attribut, doivent permettre l'extraction des informations utiles à la commande. Ils doivent gérer les éventuelles erreurs d'appariement qui peuvent se produire sur des images complexes de manière à n'extraire effectivement que les informations dont a besoin la commande.

De manière complémentaire, on doit considérer le fait que le nombre et la nature des informations fournies par la caméra peuvent être variables au cours de la réalisation d'une tâche (objets partiellement ou totalement hors du champ de vue de la caméra, occlusion, problèmes d'ombre,...). Deux cas sont à considérer :

- soit les informations sont **redondantes** : les lois de commande qui seront utilisées vont alors différer en fonction des informations disponibles durant l'exécution de la tâche.
- soit les informations sont **incomplètes** : une première démarche peut consister à estimer les informations manquantes puisque le mouvement de la caméra est commandé et donc connu. Une seconde démarche, plus originale, consiste à définir une nouvelle tâche, la plus "proche" possible de la tâche nominale, réalisable avec les informations disponibles et tentant d'accroître les informations de manière à retrouver la tâche nominale. Chaque perte ou ajout d'informations entraîne alors une nouvelle définition de tâche et donc une nouvelle loi de commande.

Les premières applications qui viennent à l'esprit sont les tâches de positionnement dans des scènes tridimensionnelles, telles que certains objets les constituant ne soient pas toujours entièrement visibles par la caméra.

Enfin, l'amélioration des techniques de vision par ordinateur permettra d'envisager l'utilisation de la commande référencée vision dans des univers de plus en plus mal connus. Actuellement, des hypothèses sur la forme et sur les dimensions des objets de la scène sont nécessaires pour réaliser la plupart des tâches. Considérons, par exemple, une tâche de positionnement devant une porte à une distance fixée : les dimensions de cette porte doivent alors être connues ou estimées pour pouvoir construire le motif exact à atteindre dans l'image et contrôler effectivement la distance entre la caméra et la porte.

Or, il devrait devenir possible de se libérer de ces hypothèses. En effet, une estimation des données nécessaires à la réalisation d'une tâche peut s'effectuer parallèlement à l'exécution de celle-ci, notamment par des méthodes de **vision**

dynamique, et modifier interactivement cette tâche jusqu'à obtenir une estimation fiable, soit un motif dans l'image stable.

L'utilisation d'une caméra couplée à un faisceau laser permet de réduire à leur plus simple expression les différents problèmes de traitements d'image. La modélisation des informations visuelles, et notamment le calcul des torseurs d'interaction, est alors à reprendre puisque le modèle du capteur est alors plus complexe que la simple projection perspective. Les résultats pour les points ont été récemment obtenus [Urban 90] et peuvent certainement se généraliser à des primitives plus complexes.

Par contre, une extension de la commande référencée vision à un système stéréoscopique à deux caméras (ou davantage) paraît aisée. L'intérêt d'une telle extension est peut-être d'assurer une meilleure stabilité des liaisons à la position d'équilibre et de disposer de tous les acquis en ce domaine sur la reconstruction tridimensionnelle de l'environnement.

Enfin, pour conclure, citons les deux axes de recherche principaux dans lesquels la commande référencée vision devrait apporter des solutions originales :

- la **vision active**, qui consiste à améliorer la connaissance d'une scène par des mouvements adéquats de la caméra. Une commande utilisant la vision est donc parfaitement indiquée pour gérer ces différents mouvements.
- la **trajectographie 3D**, qui consiste à reconstruire les mouvements des objets mobiles d'une scène. Il paraît en effet judicieux d'utiliser les mouvements de la caméra nécessaires à un suivi d'objet afin de connaître le mouvement de cet objet. Ces deux problèmes d'estimation du mouvement 3D et de suivi sont même intimement liés : un bon suivi entraînera une bonne estimation du mouvement et, de même, une bonne estimation du mouvement aidera à réaliser un suivi correct.

Annexe A

Calibration d'un système expérimental de vision

A.1 Introduction

Dans cette annexe, on propose une méthode pour identifier l'ensemble des paramètres du modèle d'un système expérimental de vision mobile. Les points originaux consistent notamment à :

- utiliser les informations fournies par la caméra embarquée (qui ne sont pas injectées dans une quelconque loi de commande !) pour identifier les paramètres du modèle du robot. Cependant, cette méthode n'est absolument pas générale et ne peut s'appliquer, texto, qu'à des robots de même configuration.
- proposer une technique robuste de calibration de la caméra, basée sur une méthode d'estimation non linéaire et permettant en outre d'identifier une éventuelle distorsion radiale.

Avant de décrire les trois étapes nécessaires à la calibration de notre système expérimental, il nous faut introduire quelques notations concernant les matrices homogènes et les représentations des rotations.

A.1.1 Notations

Soient R_i et R_j deux repères orthonormés $(O_i, \vec{x_i}, \vec{y_i}, \vec{z_i})$ et $(O_j, \vec{x_j}, \vec{y_j}, \vec{z_j})$, on représente la situation de R_j par rapport à R_i par la matrice homogène M_i^j :

$$M_i^j = \begin{pmatrix} R_i^j & T_i^j \\ 0 & 1 \end{pmatrix} \tag{A.1}$$

T_i^j est le vecteur de translation correspondant aux coordonnées de O_j dans R_i et R_i^j la matrice de rotation entre $(\vec{x_i}, \vec{y_i}, \vec{z_i})$ et $(\vec{x_j}, \vec{y_j}, \vec{z_j})$ (voir Figure A.1).

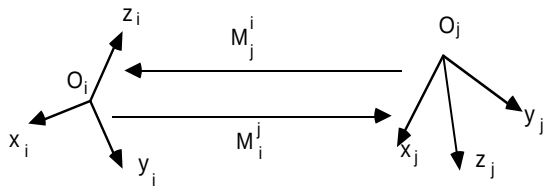


Figure A.1 : Changement de repère

On rappelle que :

$$M_i^j . M_j^k = M_i^k \tag{A.2}$$

et

$$M_j^i = M_i^{j^{-1}} = \begin{pmatrix} R_i^{j^T} & -R_i^{j^T} . T_i^j \\ 0 & 1 \end{pmatrix} \tag{A.3}$$

Dans la suite de cette annexe, les matrices homogènes seront écrites sous la forme réduite :

$$M_i^j = \begin{pmatrix} R_i^j & T_i^j \end{pmatrix} \tag{A.4}$$

De plus, les coefficients d'une matrice de rotation R seront notés $R_{[ij]}$ et une matrice M dépendant du temps sera notée $M_{(t)}$ à l'instant t.

Enfin, nous aurons besoin de la représentation des rotations sous leur forme matricielle, avec les propriétés bien connues d'orthonormalité des lignes et colonnes d'une matrice de rotation, mais également sous leur forme (U, θ) où U est l'axe normé de la rotation et θ l'angle de rotation autour de cet axe. Le passage d'une représentation à l'autre est immédiat :

- Soit $(U = (U_x, U_y, U_z)^T, \theta)$ la représentation d'une rotation et R la matrice de rotation correspondante, on a :

$$R = \cos\theta . I_3 + (1 - \cos\theta) . UU^T + \sin\theta . As(U) \tag{A.5}$$

avec I_3 la matrice identité et As(U) la matrice antisymétrique :

$$As(U) = \begin{pmatrix} 0 & -U_z & U_y \\ U_z & 0 & -U_x \\ -U_y & U_x & 0 \end{pmatrix}$$

- Réciproquement, on a :

$$\cos\theta = \frac{1}{2}(R_{[11]} + R_{[22]} + R_{[33]} - 1) \qquad (A.6)$$

et

$$\sin\theta . As(U) = \frac{1}{2}(R - R^T) \qquad (A.7)$$

En se fixant θ positif, U est déterminé de manière unique si $\sin\theta \neq 0$.

A.1.2 Méthodologie

Les différents repères mis en jeu sont illustrés sur la Figure A.2. Pour pouvoir

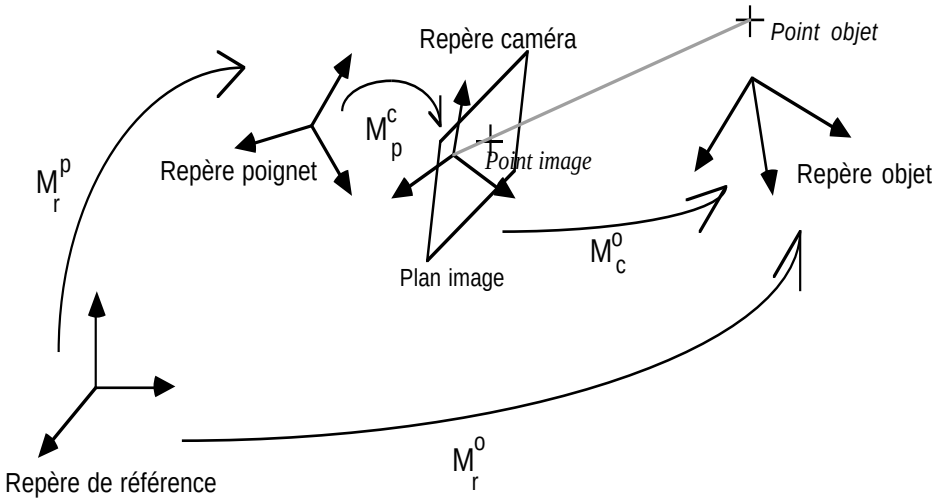


Figure A.2 : Les différents repères utilisés

contrôler les mouvements de la caméra, il est nécessaire de connaître à chaque instant la matrice M_r^c représentant la position et l'orientation de la caméra par rapport à un repère de référence associé au robot. Cette matrice résulte de la composition de deux matrices : M_r^p, correspondant à la transformation repère de référence-repère de l'organe terminal, et M_p^c représentant la transformation repère de l'organe terminal-repère caméra.

- Le calcul de M_r^p constitue l'identification du modèle géométrique du robot, c'est-à-dire la relation entre les coordonnées articulaires et la matrice M_r^p.
- Le calcul de M_p^c se fait par le biais de l'introduction d'un repère supplémentaire fixe : le repère objet dans lequel la scène vue par la caméra est référencée. On a alors :

$$M_r^o = M_r^p.M_p^c.M_c^o = cte \tag{A.8}$$

Le calcul de M_c^o sous-entend le choix d'un modèle de la caméra. Une identification préalable des paramètres du modèle choisi est donc nécessaire. Ce problème classique, la calibration, permet de lier un point (3D), visible par la caméra, à sa position dans l'image (2D).

Enfin, comme il est impossible de connaître précisément M_r^o (aucune hypothèse sur la position de la scène par rapport au repère de référence n'est formulée), on utilisera des déplacements du robot et des calibrations à chaque position correspondante de la caméra pour calculer M_p^c :

$$M_{r_{(1)}}^p.M_p^c.M_{c_{(1)}}^o = M_{r_{(2)}}^p.M_p^c.M_{c_{(2)}}^o = ... \tag{A.9}$$

Il faut remarquer que l'identification du robot, des paramètres du modèle de la caméra et de la matrice M_p^c n'est à effectuer qu'une seule fois. Aussi les différentes méthodes employées ont-elles été choisies pour leur précision et leur robustesse.

A.2 Réglage des zéros angulaires du robot à l'aide de la caméra embarquée

A.2.1 Description

Le robot manipulateur dont nous disposons est un A.I.D. V5 à six degrés de liberté, à articulations rotoïdes et concourantes. Les différents repères de la chaîne articulée et leur orientation de référence sont représentés sur la Figure A.3 (les origines des repères R_4, R_5 et R_6 sont confondues, formant une liaison rotule). Par convention, l'axe z de chaque repère coïncide avec l'axe de rotation de l'articulation correspondante.

A la configuration initiale représentée sur la Figure A.3 correspondent les valeurs des six coordonnées articulaires q_{0i}. Il est primordial de repérer exactement ces valeurs q_{0i} par rapport à une position connue du robot (par exemple les butées mécaniques) afin de pouvoir utiliser le modèle géométrique direct donnant

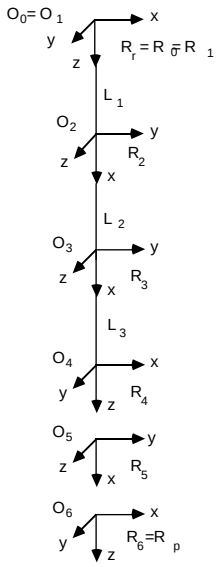


Figure A.3 : Configuration de référence

la situation du poignet (R_p) par rapport à un repère fixe de référence (R_r). En effet, on a, pour une position quelconque q_i :

$$M_r^p = M_0^6 = M_0^1.M_1^2.M_2^3.M_3^4.M_4^5.M_5^6 \tag{A.10}$$

avec :

$$M_0^1 = \begin{pmatrix} \cos(q_1 - q_{01}) & -\sin(q_1 - q_{01}) & 0 & 0 \\ \sin(q_1 - q_{01}) & \cos(q_1 - q_{01}) & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

$$M_1^2 = \begin{pmatrix} -\sin(q_2 - q_{02}) & \cos(q_2 - q_{02}) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \cos(q_2 - q_{02}) & \sin(q_2 - q_{02}) & 0 & L_1 \end{pmatrix}$$

$$M_2^3 = \begin{pmatrix} \cos(q_3 - q_{03}) & -\sin(q_3 - q_{03}) & 0 & L_2 \\ \sin(q_3 - q_{03}) & \cos(q_3 - q_{03}) & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

$$M_3^4 = \begin{pmatrix} 0 & 0 & 1 & L_3 \\ \cos(q_4 - q_{04}) & -\sin(q_4 - q_{04}) & 0 & 0 \\ \sin(q_4 - q_{04}) & \cos(q_4 - q_{04}) & 0 & 0 \end{pmatrix}$$

$$M_4^5 = \begin{pmatrix} \sin(q_5 - q_{05}) & \cos(q_5 - q_{05}) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \cos(q_5 - q_{05}) & -\sin(q_5 - q_{05}) & 0 & 0 \end{pmatrix}$$

$$M_5^6 = \begin{pmatrix} 0 & 0 & 1 & 0 \\ \cos(q_6 - q_{06}) & -\sin(q_6 - q_{06}) & 0 & 0 \\ \sin(q_6 - q_{06}) & \cos(q_6 - q_{06}) & 0 & 0 \end{pmatrix}$$

Les longueurs L_i des trois segments du manipulateur ayant déjà été mesurées avec une précision de l'ordre du dixième de millimètre lors d'expérimentations précédentes, seule l'identification des valeurs initiales q_{0i} des coordonnées articulaires sera détaillée ici. On remarque, de plus, que la valeur q_{01} dépend uniquement du choix du repère de référence et que la valeur q_{06} peut être choisie quelconque : en effet, la matrice de passage entre le repère R_6 et le repère de la caméra qui, seul, nous intéresse, sera identifiée ultérieurement et un éventuel décalage de q_{06} sera donc compensé lors de cette identification.

Le problème se ramène donc à l'identification précise des valeurs q_{02} à q_{05} correspondant aux articulations à l'intérieur de la chaîne cinématique du manipulateur.

La méthode de réglage utilise les propriétés des singularités du manipulateur et la possibilité d'accéder à des mesures d'erreurs très faibles par le biais de la caméra. En effet, à une singularité correspond un ensemble de combinaisons des coordonnées articulaires amenant une position et une orientation constantes du poignet. Les images acquises par la caméra sont donc identiques lorsque le vecteur de coordonnées articulaires appartient à cet ensemble de combinaisons.

A.2.2 Réglage de q_{05}

- Pour une position initiale $q_1, q_2, q_3, q_4, q_5, q_6$ donnant $M^6_{0_{(1)}}$ et pour une position finale $q_1, q_2, q_3, q_4+\delta, q_5, q_6-\delta$ donnant $M^6_{0_{(2)}}$, avec $q_1, q_2, q_3, q_4, q_5, q_6$ et δ $(\neq 0)$ quelconques, on peut facilement montrer que :

$$M^6_{0_{(1)}} = M^6_{0_{(2)}} \text{ si et seulement si } q_5 = q_{05} \text{ ou } q_5 = q_{05} + \pi \qquad \text{(A.11)}$$

En comparant les images aux positions initiale et finale, on peut régler q_5 jusqu'à obtenir deux images identiques. On a alors $q_5 = q_{05}$. En pratique, il suffit de comparer les deux images sur seulement quatre points non-coplanaires dans la scène pour savoir si les deux images sont égales (la scène est évidemment fixe). Par exemple, les points considérés peuvent être obtenus en calculant le centre de gravité de la projection sur l'image de quatre disques.

- Remarque : on ne peut confondre les valeurs q_{05} et $q_{05} + \pi$ car R_5 doit conserver son orientation définie par la position de référence.

A.2.3 Réglage de q_{02} et q_{03}

Ayant identifié q_{05} , une démarche analogue permet de régler les zéros des articulations 2 et 3 en amenant ces articulations dans la singularité correspondant à la configuration où le bras est entièrement étendu.

Comme précédemment, on peut démontrer :

- Pour une position initiale $q_1, q_2, q_3, q_4, q_{05}, q_6$ donnant $M^6_{0_{(1)}}$ et pour une position finale $q_1+\delta, q_2, q_3, q_4, q_{05}, q_6-\delta$ donnant $M^6_{0_{(2)}}$, avec q_1, q_2, q_3, q_4, q_6 et δ $(\neq 0)$ quelconques, on a :

$$M^6_{0_{(1)}} = M^6_{0_{(2)}} \text{ si et seulement si } \begin{cases} q_2 = q_{02} \text{ ou } q_2 = q_{02} + \pi \\ q_3 = q_{03} \text{ ou } q_3 = q_{03} + \pi \end{cases} \quad \text{(A.12)}$$

Il faut remarquer qu'il est impossible de dissocier le réglage des articulations 2 et 3. La méthodologie adoptée consiste à régler q_3 au mieux, ce qui fournit $\tilde{q_3}$, puis q_2 en prenant $q_3 = \tilde{q_3}$ pour obtenir $\tilde{q_2}$ et à réitérer le processus jusqu'à obtenir deux images identiques aux positions initiale et finale.

A.2.4 Réglage de q_{04}

Connaissant maintenant q_{02}, q_{03} et q_{05}, il est possible de positionner le robot tel que :

$$\begin{cases} O_6 \in (O_0, \vec{z_0}) \\ \vec{z_0} = \vec{z_6} \\ O_0O_6 = h < L_1 + L_2 + L_3 \end{cases} \quad \text{(A.13)}$$

Il faut (voir Figure A.4):

$$\begin{cases} L_2 \cos q'_2 + L_3 \cos q'_2 + q'_3 = h \\ L_2 \sin q'_2 + L_3 \sin q'_2 + q'_3 = 0 \\ q'_5 = -(q'_2 + q'_3) \end{cases} \quad \text{(A.14)}$$

On démontre :

- Pour une position initiale $q_1, q_2, q_3, q_4, q_5, q_6$ donnant $M^6_{0_{(1)}}$ et pour une position finale $q_1 + \delta, q_2, q_3, q_4, q_5, q_6 - \delta$ donnant $M^6_{0_{(2)}}$, avec q_2, q_3, q_4 et q_5 vérifiant les conditions ci-dessus et q_1, q_2, δ $(\neq 0)$ quelconques, on a :

$$M^6_{0_{(1)}} = M^6_{0_{(2)}} \text{ si et seulement si } q_4 = q_{04} \text{ ou } q_4 = q_{04} + \pi \quad \text{(A.15)}$$

On peut ainsi régler q_4 de la même manière que q_5 pour obtenir q_{04}.

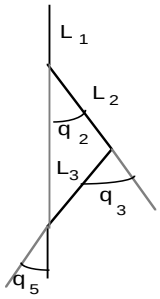


Figure A.4 : Poignet vertical

A.2.5 Résultats expérimentaux

Le réglage de q_{05} a donné des résultats excellents : pas de différence entre les deux images. Par contre, il a été impossible d'obtenir parfaitement q_{02} et q_{03} : des erreurs d'environ un pixel séparent les deux positions de chaque point considéré dans l'image. Le réglage de l'articulation 4 a donné des résultats aussi précis, ce qui laisse supposer une bonne mesure des longueurs des segments du robot et une bonne valeur de q_{04}. Les erreurs constatées sont certainement dues au fait que les articulations du robot ne sont pas parfaitement concourrantes, par défaut de construction.

Des vérifications sur la précision des valeurs obtenues peuvent être effectuées en utilisant le modèle géométrique inverse qui donne l'ensemble des solutions des positions articulaires à partir d'une matrice M_0^6 voulue. En comparant les images correspondant à chaque configuration, on constate des erreurs de 3 pixels au plus séparant les positions d'un même point dans les deux images acquises. La faible valeur de ces erreurs (1 pixel $\sim 20 \times 20\ \mu m^2$) correspondant aux cas de figure les plus défavorables atteste la bonne précision des résultats obtenus.

- Remarque : Lors de l'exécution d'un déplacement, on s'efforcera, pour garantir une excellente précision, d'éviter ces reconfigurations : en effet, les coordonnées articulaires varient alors de manière discontinue d'une position à une autre.

Une caméra, de par sa grande précision, est un excellent moyen de calibration d'un robot et pourrait être utilisée par les constructeurs eux-mêmes pour évaluer la précision de leurs produits. Les résultats obtenus dans notre cas particulier montrent la validité de l'approche. Toutefois, une réflexion plus approfondie doit être menée, visant à proposer une méthodologie et une métrologie adaptables à une large classe de manipulateurs de géométries différentes [Guyot 89].

A.3 Modélisation et calibration de la caméra

La caméra est modélisée de façon très classique par une projection perspective. L'identification des paramètres de ce modèle permet d'établir la correspondance entre la position d'un point dans l'image numérisée, exprimée en pixel, et sa position réelle dans la scène.

Enfin, pour améliorer la précision de ce modèle, nous introduisons un coefficient de distorsion radiale permettant de prendre en compte les défauts éventuels de l'objectif.

Pour plus de détails sur les principales méthodes de calibration, le lecteur pourra se reporter à l'article de Tsai [Tsai 87a].

A.3.1 Mise en équations

L'origine du repère de la caméra est située au centre de projection C (voir Figure A.5). L'axe $(C, \vec{x_c})$ est orienté comme les abscisses du plan image et l'axe $(C, \vec{y_c})$ comme les ordonnées. La distance minimale du centre de projection au plan image, appelée distance focale, est notée f. L'intersection de l'axe $(C, \vec{z_c})$ avec le plan image n'est pas supposée être au centre de l'image, mais a pour coordonnées dans l'image numérisée (X_c, Y_c).

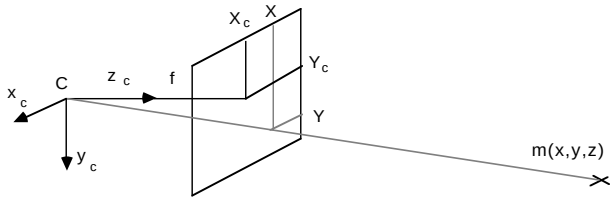


Figure A.5 : Projection perspective

- Remarque : la seule hypothèse faite sur l'image est sa planéité et l'orthogonalité de ses axes.

Dans le repère choisi, la projection perspective d'un point $m(x, y, z)$ sur le plan image est $M(X, Y)$ (voir Figure A.6) avec :

$$\begin{cases} X = f \,.\, x \,/\, z \\ \\ Y = f \,.\, y \,/\, z \end{cases} \tag{A.16}$$

Soit K_{1d} le coefficient de distorsion radiale [Brown 71], alors la position du point M réellement observé dans l'image est $M_d(X_d, Y_d)$ avec (voir Figure A.6) :

$$\begin{cases} X_d = X + K_{1d}.X.(X^2 + Y^2) \\ \\ Y_d = Y + K_{1d}.Y.(X^2 + Y^2) \end{cases} \tag{A.17}$$

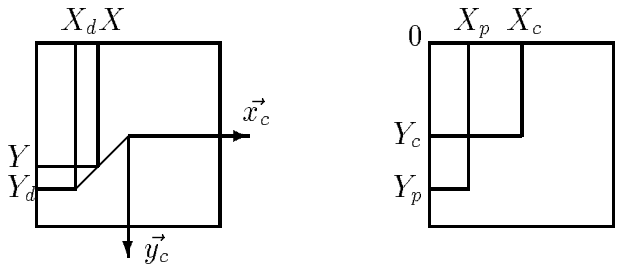


Figure A.6 : Position d'un point dans l'image

Exprimons maintenant M par sa position $M_p(X_p, Y_p)$ dans l'image numérisée (en pixel) :

$$\begin{cases} X_p = X_c + X_d \,/\, l_x \\ \\ Y_p = Y_c + Y_d \,/\, l_y \end{cases} \tag{A.18}$$

où l_x et l_y sont des coefficients proportionnels à la taille d'un pixel élémentaire.

A partir de ces équations, on peut écrire le modèle complet de la caméra :

$$\begin{cases} X_p = X_c + F_x.\frac{x}{z} + K_d.F_x.\frac{x}{z}.(\frac{x^2}{z^2} + \frac{y^2}{z^2}) \\ \\ Y_p = Y_c + F_y.\frac{y}{z} + K_d.F_y.\frac{y}{z}.(\frac{x^2}{z^2} + \frac{y^2}{z^2}) \end{cases} \tag{A.19}$$

où $F_x = f/l_x$, $F_y = f/l_y$ et $K_d = K_{1d}.f^2$.

Les paramètres à identifier, appelés paramètres intrinsèques de la caméra, sont alors :X_c, Y_c, F_x, F_y et K_d .

Dans la majorité des applications, il est inutile de dissocier f,l_x et l_y pour utiliser le modèle de la caméra. On peut, par exemple, poser $f = 1$. Si l'on a tout de même besoin explicitement de ces valeurs, fournies en général par le constructeur, il suffit d'en supposer une exacte et d'en déduire les autres.

Les équations (A.19) ne sont pas directement exploitables car l'on ne connaît pas en pratique les coordonnées (x, y, z) de m dans le repère de la caméra. Par contre, si l'on se fixe un repère objet dans lequel on connaît les coordonnées (x_o, y_o, z_o) de m, il est possible d'identifier les paramètres intrinsèques de la caméra, de même que l'orientation et la position du repère objet par rapport au repère de la caméra, soit la matrice M_c^o. Les paramètres représentant cette matrice sont appelés paramètres extrinsèques.

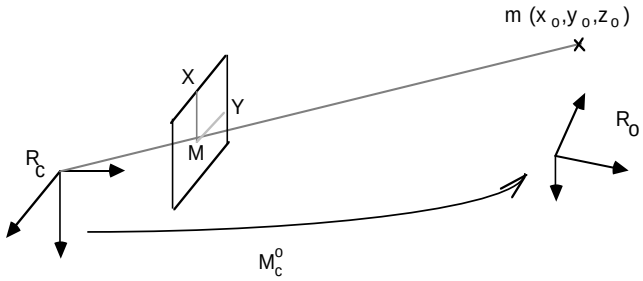


Figure A.7 : Passage du repère caméra au repère objet

Soit m un point de coordonnées (x, y, z) dans R_c et (x_o, y_o, z_o) dans R_o, on a bien entendu (voir Figure A.7) :

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = M_c^o . \begin{pmatrix} x_o \\ y_o \\ z_o \\ 1 \end{pmatrix} \text{ avec } M_c^o = \begin{pmatrix} R_c^o & T_c^o \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} & X_o \\ a_{21} & a_{22} & a_{23} & Y_o \\ a_{31} & a_{22} & a_{33} & Z_o \end{pmatrix} \tag{A.20}$$

En combinant les équations (A.19) et (A.20), on obtient deux équations reliant les coordonnées d'un point dans le repère objet et les coordonnées de son image dans l'image numérisée.

A.3.2 Résolution par un système linéaire – Simulation

Dans un premier temps, nous supposons le coefficient de distorsion K_d nul. Cela va nous permettre de formuler le problème en termes d'un système linéaire. Le modèle de la caméra peut alors s'écrire sous la forme d'une matrice homogène P et on obtient :

$$X_p = \frac{X}{\omega} \ , \ Y_p = \frac{Y}{\omega} \ , \ f = \frac{Z}{\omega} \tag{A.21}$$

avec

$$\begin{pmatrix} X \\ Y \\ Z \\ \omega \end{pmatrix} = P. \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 1/l_x & 0 & X_c/f \\ 0 & 1/l_y & Y_c/f \\ 0 & 0 & 1 \\ 0 & 0 & 1/f \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

Les équations (A.19) se réécrivent :

$$\begin{cases} X_p.z - X_c.z - F_x.x = 0 \\ \\ Y_p.z - Y_c.z - F_y.y = 0 \end{cases} \tag{A.22}$$

et on en déduit, en utilisant (A.20) :

$$\begin{aligned} a_{31}X_p x_o + a_{32}X_p y_o + a_{33}X_p z_o + X_p Z_o \\ -a_{31}X_c x_o - a_{32}X_c y_o - a_{33}X_c z_o - X_c Z_o \\ -F_x a_{11} x_o - F_x a_{12} y_o - F_x a_{13} z_o - F_x X_o &= 0 \\ \\ a_{31}Y_p x_o + a_{32}Y_p y_o + a_{33}Y_p z_o + Y_p Z_o \\ -a_{31}Y_c x_o - a_{32}Y_c y_o - a_{33}Y_c z_o - Y_c Z_o \\ -F_y a_{21} x_o - F_y a_{22} y_o - F_y a_{23} z_o - F_y Y_o &= 0 \end{aligned} \tag{A.23}$$

On obtient le système linéaire à douze inconnues $AI = 0$ où $I = (I_1 \cdots I_{12})^T$ et où A est donné par :

$$\begin{pmatrix} X_p x_o & X_p y_o & X_p z_o & -x_o & -y_o & -z_o & 0 & 0 & 0 & -1 & 0 & X_p \\ Y_p x_o & Y_p y_o & Y_p z_o & 0 & 0 & 0 & -x_o & -y_o & -z_o & 0 & -1 & Y_p \end{pmatrix}$$

avec :

$$\begin{array}{lll} I_1 = a_{31} & I_2 = a_{32} & I_3 = a_{33} \\ I_4 = a_{31}X_c + a_{11}F_x & I_5 = a_{32}X_c + a_{12}F_x & I_6 = a_{33}X_c + a_{13}F_x \\ I_7 = a_{31}Y_c + a_{21}F_y & I_8 = a_{32}Y_c + a_{22}F_y & I_9 = a_{33}Y_c + a_{23}F_y \\ I_{10} = X_c Z_o + F_x X_o & I_{11} = Y_c Z_o + F_y Y_o & I_{12} = Z_o \end{array} \tag{A.24}$$

Six points non-coplanaires sont suffisants pour résoudre ce système, mais, pour améliorer la précision des résultats, il est nécessaire d'en utiliser davantage : on a alors un système surdimensionné à (2xNpoints) équations si N > 6.

Une résolution directe est impossible (solution triviale nulle), aussi utilise-t-on une contrainte liée aux propriétés d'orthonormalité des matrices de rotation, à savoir :

$$a_{31}^2 + a_{32}^2 + a_{33}^2 = 1 \tag{A.25}$$

La méthode de Lagrange permet alors de résoudre ce système en termes de minimisation sous contrainte [Faugeras 87].

En effet, le système peut se réécrire sous la forme $A.X_1+B.X_2 = 0$ en posant $X_1 = (a_{31}, a_{32}, a_{33})^T$. On doit alors minimiser le critère :

$$C = \|A.X_1 + B.X_2\|^2 + \lambda\,(1 - \|X_1\|^2) \tag{A.26}$$

où λ est un multiplicateur de Lagrange. En annulant les dérivées partielles de C, on obtient :

$$\begin{cases} \frac{1}{2}\frac{\partial C}{\partial X_1} = A^T A.X_1 + A^T B.X_2 - \lambda X_1 = 0 \\ \\ \frac{1}{2}\frac{\partial C}{\partial X_2} = B^T A.X_1 + B^T B.X_2 = 0 \end{cases} \tag{A.27}$$

d'où

$$\begin{cases} X_2 = -(B^T B)^{-1} B^T A.X_1 \\ \\ E.X_1 = \lambda X_1 \quad \text{avec} \quad E = A^T A - A^T B (B^T B)^{-1} B^T A \end{cases} \tag{A.28}$$

De plus, en utilisant (A.27), on voit que le critère C a pour valeur λ si X_1 est un vecteur propre unitaire de E (matrice symétrique et positive) correspondant à la valeur propre λ.

X_1 est donc le vecteur propre unitaire correspondant à la plus petite valeur propre de E. On obtient ainsi X_1 et X_2 au signe près, et, pour remonter à I, il suffit de remarquer que I_{12} est positif (le repère objet est choisi arbitrairement devant la caméra).

Grâce aux propriétés des matrices de rotation et aux relations (A.24), on en déduit les paramètres intrinsèques et extrinsèques :

$$\begin{cases} X_c = I_1.I_4 + I_2.I_5 + I_3.I_6 & Y_c = I_1.I_7 + I_2.I_8 + I_3.I_9 \\ F_x = (I_4^2 + I_5^2 + I_6^2 - X_c^2)^{1/2} & F_y = (I_7^2 + I_8^2 + I_9^2 - Y_c^2)^{1/2} \\ a_{31} = I_1 & a_{32} = I_2 \\ a_{33} = I_3 & a_{11} = (I_4 - a_{31}X_c)/F_x \\ a_{12} = (I_5 - a_{32}X_c)/F_x & a_{13} = (I_6 - a_{33}X_c)/F_x \\ a_{21} = (I_7 - a_{31}Y_c)/F_y & a_{22} = (I_8 - a_{32}Y_c)/F_y \\ a_{23} = (I_9 - a_{33}Y_c)/F_y & X_0 = (I_{10} - X_c I_{12})/F_x \\ Y_0 = (I_{11} - Y_c I_{12})/F_y & Z_0 = I_{12} \end{cases} \tag{A.29}$$

Un programme de simulation a été développé pour tester cette méthode. Il consiste à construire les points images à partir d'un modèle donné et de points objets choisis, à bruiter les points images obtenus et, enfin, à retrouver ce modèle.

Les résultats sont excellents pour des simulations exactes (bruit nul). Par contre, en présence d'un bruit blanc uniforme, même peu important, les paramètres retrouvés sont biaisés. Dans le tableau A.1 sont regroupés les résultats (moyenne $\bar{m}$, écart-type σ, minimum et maximum) de 100 simulations pour des bruits blancs d'amplitude maximale $\varepsilon = 0.5$, 1, 1.5 et 2 pixels. La rotation entre le repère caméra et le repère scène est représentée par le vecteur $T = (T_x, T_y, T_z)^T$ qui s'obtient par $T = \theta.U$ à partir de la représentation (U, θ) de cette rotation. L'intérêt d'utiliser cette représentation sera vu ultérieurement.

	$\varepsilon = 0.5$ pixel				$\varepsilon = 1.0$ pixel			
	$\bar{m}$	σ	min	max	$\bar{m}$	σ	min	max
X_c	120.075	0.50	118.95	121.59	120.080	0.95	117.85	122.67
Y_c	139.938	0.82	138.08	141.88	140.267	1.75	136.53	143.74
F_x	399.302	0.64	397.67	400.94	397.812	1.48	393.83	401.72
F_y	499.156	0.80	497.22	501.44	497.317	1.80	492.75	502.35
X_o	-100.115	0.63	-102.16	-98.55	-100.151	1.17	-103.14	-97.31
Y_o	-199.933	0.81	-201.80	-198.03	-200.257	1.73	-203.72	-196.77
Z_o	498.983	1.02	496.37	501.68	496.691	2.30	491.04	502.64
T_x	11.617	0.10	11.40	11.85	11.656	0.20	11.16	12.08
T_y	8.133	0.07	7.94	8.28	8.143	0.13	7.69	8.50
T_z	20.771	0.01	20.74	20.79	20.772	0.03	20.68	20.84
	$\varepsilon = 1.5$ pixel				$\varepsilon = 2.0$ pixel			
	$\bar{m}$	σ	min	max	$\bar{m}$	σ	min	max
X_c	120.675	1.80	115.90	124.35	120.854	2.13	115.54	126.85
Y_c	139.691	2.39	134.29	145.75	140.589	3.06	132.98	147.48
F_x	395.134	1.99	389.87	399.79	391.764	2.72	384.24	399.30
F_y	494.021	2.44	488.36	499.33	489.580	3.34	481.03	498.48
X_o	-100.952	2.24	-105.53	-94.88	-101.260	2.61	-108.06	-94.57
Y_o	-199.632	2.36	-205.55	-194.24	-200.457	2.99	-206.98	-193.08
Z_o	492.566	3.03	485.20	499.28	487.072	4.09	477.30	497.06
T_x	11.598	0.29	10.97	12.26	11.730	0.37	10.81	12.70
T_y	8.099	0.22	7.64	8.67	8.083	0.28	7.35	8.84
T_z	20.773	0.04	20.68	20.89	20.779	0.06	20.64	20.91

Tableau A.1 : Simulation de la méthode de Lagrange

Pour les différentes simulations, les paramètres intrinsèques et extrinsèques du modèle à identifier sont :

$$\begin{cases} X_c = 120 \text{ pixels} & Y_c = 140 \text{ pixels} \\ F_x = 400 & F_y = 500 \\ K_d = 0 \\ X_o = -100 \text{ mm} & Y_o = -200 \text{ mm} & Z_o = 500 \text{ mm} \\ T_x = 11.619 & T_y = 8.136 & T_z = 20.771 \end{cases}$$

En comparant les valeurs de ce modèle avec les paramètres calculés par la méthode décrite ci-dessus, on constate un biais important sur les valeurs F_x, F_y et Z_o, bien que le coefficient de distorsion soit choisi nul dans le modèle de référence.

Dans le cas de l'identification d'une caméra réelle, les imperfections des optiques et du positionnement de l'élément photosensible dans la caméra entraînent inévitablement des erreurs sur la mesure des coordonnées image. Aussi ne peut-on se satisfaire de cette méthode donnant des résultats biaisés en présence de bruit. Toutefois, ces résultats peuvent servir d'estimées initiales à des méthodes plus performantes.

A.3.3 Résolution par une méthode non linéaire – Simulation

Parmi les méthodes d'estimations classiquement utilisées, certains auteurs ont fait appel, pour résoudre ce problème, à une méthode de filtrage étendu de Kalman. L'inconvénient de cette technique réside dans le fait qu'elle nécessite la linéarisation du modèle et qu'elle ne permet pas d'estimer le coefficient de distorsion radiale. Nous avons choisi, plus naturellement, une méthode de minisation non linéaire [Sobel 74]. Le critère à minimiser est :

$$C_2 = \sum_{i=1}^{N} [(X_{pi} - X_{mi})^2 + (Y_{pi} - Y_{mi})^2] \tag{A.30}$$

où (X_{mi}, Y_{mi}) est la mesure du point i dans l'image numérisée et où (X_{pi}, Y_{pi}) représente son estimée par le modèle calculé à partir des équations (A.19) et (A.20), fonction des paramètres intrinsèques et extrinsèques de la caméra.

Cette méthode permet en outre d'identifier le coefficient de distorsion radiale K_d dont l'estimée initiale est évidemment choisie nulle.

La rotation est représentée par le vecteur $T = \theta.U$ pour avoir une représentation minimale des rotations, condition nécessaire pour employer une méthode de minimisation non linéaire sans contrainte classique (dans notre cas, une méthode de type Gauss-Newton adaptée aux fonctions composées de sommes de carrés [Gill 78], [NAG 84]).

Cette méthode a été testée en simulation et nous avons constaté qu'elle donnait des résultats tout à fait satisfaisants et sans biais (voir tableau A.2, résultats correspondant au même modèle que précédemment).

	$\varepsilon = 0.5$ pixel				$\varepsilon = 1.0$ pixel			
	$\bar{m}$	σ	min	max	$\bar{m}$	σ	min	max
X_c	120.037	0.49	118.76	121.43	119.854	0.93	117.52	122.16
Y_c	139.896	0.77	138.03	141.56	140.221	1.66	136.20	143.91
F_x	399.856	0.66	398.07	401.33	400.017	1.49	396.21	404.27
F_y	499.842	0.82	497.81	501.88	500.077	1.80	495.81	505.68
X_o	-100.049	0.61	-101.90	-98.34	-99.817	1.17	-102.62	-97.11
Y_o	-199.896	0.76	-201.49	-198.01	-200.235	1.63	-203.86	-196.42
Z_o	499.830	1.02	497.14	502.20	500.082	2.30	494.77	506.80
T_x	11.610	0.09	11.41	11.81	11.641	0.19	11.11	12.08
T_y	8.134	0.07	7.94	8.27	8.154	0.13	7.77	8.51
T_z	20.771	0.01	20.74	20.79	20.770	0.03	20.72	20.84
	$\varepsilon = 1.5$ pixel				$\varepsilon = 2.0$ pixel			
	$\bar{m}$	σ	min	max	$\bar{m}$	σ	min	max
X_c	120.157	1.77	115.18	123.98	120.049	2.14	115.03	124.91
Y_c	139.604	2.23	134.55	145.12	140.334	2.96	132.36	147.88
F_x	399.972	2.09	394.62	404.73	400.429	2.75	392.79	406.03
F_y	500.090	2.57	494.00	506.67	500.443	3.37	491.92	507.39
X_o	-100.193	2.23	-105.04	-93.73	-100.037	2.67	-106.30	-93.43
Y_o	-199.603	2.20	-204.93	-194.52	-200.302	2.89	-207.55	-192.54
Z_o	500.019	3.16	492.69	506.50	500.401	4.18	490.31	510.78
T_x	11.568	0.27	10.99	12.15	11.661	0.35	10.72	12.62
T_y	8.125	0.22	7.67	8.72	8.118	0.27	7.48	8.93
T_z	20.770	0.04	20.70	20.87	20.775	0.05	20.65	20.89

Tableau A.2 : Simulation de la méthode non linéaire

Si on introduit dans ce modèle un coefficient de distorsion non nul ($K_d = -0.3$, correspondant à une distorsion de 10 pixels pour les points distants de 100 pixels de (X_c, Y_c)), les résultats sont tout aussi satisfaisants (voir tableau A.3). La méthode de Lagrange donne des estimées médiocres (erreurs de plus de 20 pixels sur X_c et Y_c notamment), mais suffisantes pour permettre la convergence de la méthode non linéaire.

	$\varepsilon = 0.5$ pixel				$\varepsilon = 1.0$ pixel			
	$\bar{m}$	σ	min	max	$\bar{m}$	σ	min	max
X_c	120.054	1.71	116.38	124.79	120.473	3.73	108.76	129.79
Y_c	139.991	1.78	135.10	144.23	140.383	3.41	130.82	153.73
F_x	399.975	0.76	397.82	401.95	399.868	1.48	396.42	404.49
F_y	499.965	0.96	497.34	502.38	499.859	1.82	495.73	505.88
K_d	-0.300	0.00	-0.31	-0.29	-0.300	0.01	-0.32	-0.28
X_o	-100.072	2.18	-106.08	-95.41	-100.594	4.75	-112.35	-85.60
Y_o	-199.986	1.80	-204.27	-195.03	-200.403	3.42	-213.71	-190.83
Z_o	499.962	1.40	495.97	502.57	499.502	2.76	492.21	508.25
T_x	11.620	0.20	11.08	12.11	11.644	0.38	10.67	13.05
T_y	8.129	0.25	7.48	8.62	8.067	0.53	6.71	9.64
T_z	20.772	0.03	20.71	20.83	20.781	0.05	20.65	20.94
	$\varepsilon = 1.5$ pixel				$\varepsilon = 2.0$ pixel			
	$\bar{m}$	σ	min	max	$\bar{m}$	σ	min	max
X_c	118.825	6.09	103.90	135.62	119.983	7.45	102.33	141.36
Y_c	140.367	5.07	129.75	157.54	139.133	7.42	117.76	158.44
F_x	400.268	2.17	395.53	406.42	400.192	2.80	393.00	407.09
F_y	500.288	2.80	494.22	508.16	500.189	3.48	489.63	508.63
K_d	-0.298	0.01	-0.33	-0.26	-0.299	0.02	-0.35	-0.25
X_o	-98.474	7.74	-119.85	-79.22	-99.921	9.47	-127.39	-77.18
Y_o	-200.371	5.09	-217.68	-189.80	-199.092	7.42	-218.08	-177.81
Z_o	500.603	4.16	487.78	511.52	500.485	5.68	485.68	518.00
T_x	11.689	0.55	10.26	13.65	11.529	0.84	8.88	13.59
T_y	8.291	0.88	5.66	10.44	8.155	1.08	5.49	10.60
T_z	20.763	0.09	20.57	21.01	20.770	0.11	20.52	21.02

Tableau A.3 : Simulation de la méthode non linéaire pour une distorsion non nulle

A.3.4 Résultats expérimentaux

Pour l'expérimentation en site réel, nous avons choisi les mêmes points objets qu'en simulation, à savoir une grille rectangulaire de 6×8 disques que l'on peut déplacer très précisément (au centième de millimètre) par rapport à l'axe $\vec{z_o}$ du repère objet (voir Figure A.8). Cinq positions équidistantes ont été choisies sur un intervalle de 20 cm, de manière à avoir un ensemble de 240 points de mesure lorsque tous les points sont visibles par la caméra.

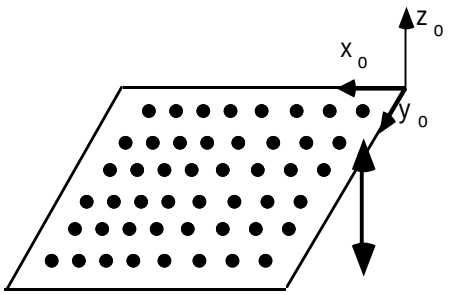


Figure A.8 : Mire utilisée

Les points objets sont situés au centre de ces disques et la position des points images correspondants est obtenue en calculant le centre de gravité de leur projection sur le plan image. L'erreur introduite par cette approximation (la projection du centre d'un cercle n'est pas le centre de la projection de ce cercle sauf s'il est parallèle au plan image) est tout à fait négligeable, d'autant plus que la grille et le plan image sont quasiment parallèles lors de nos expérimentations.

Nous avons calibré deux caméras dont l'une d'entre elles est fortement distordue (voir Figure. A.9, images de la mire acquises par les deux caméras).

On obtient les résultats suivants :

Caméra 1	Caméra 2
$X_c = 120 \pm 5$ pixels	$X_c = 151 \pm 5$ pixels
$Y_c = 153 \pm 5$ pixels	$Y_c = 146 \pm 5$ pixels
$F_x = 387 \pm 2$	$F_x = 208 \pm 2$
$F_y = 578 \pm 2$	$F_y = 296 \pm 2$
$K_d = -0.05 \pm 0.02$	$K_d = -0.21 \pm 0.02$

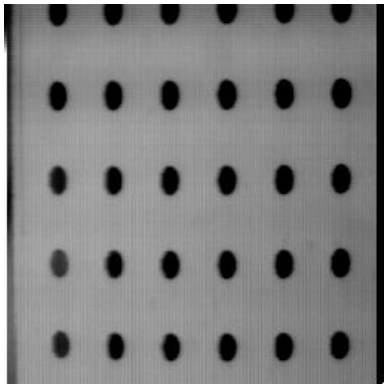
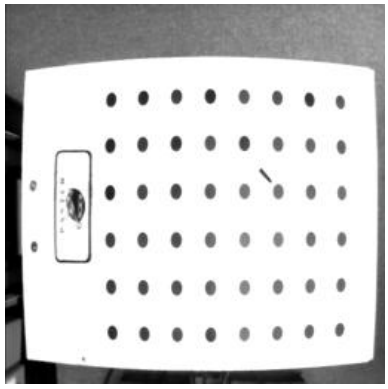



Figure A.9 : Caméra 1 – Caméra 2

Remarques :

1. Vu la taille des images (256×256 pixels) et les valeurs de X_c et Y_c obtenues, on constate un décalage important entre le centre de l'image et la projection (X_c, Y_c) du centre optique sur le plan image. L'hypothèse de décentrage est donc essentielle pour obtenir un modèle correct.

2. La valeur du critère C_2 après minimisation correspond, pour les deux caméras, à une erreur moyenne de l'ordre du quart de pixel entre chaque point mesuré et sa reconstruction par le modèle, ce qui confirme la validité du choix du modèle.

3. D'après les équations (A.17), la distorsion de la caméra 1 est d'environ 1 pixel pour les points situés à une distance de 150 pixels de (X_c, Y_c), c'est-à-dire qu'elle est quasiment négligeable, comme on pouvait l'espérer en observant les images de la mire (voir Figure. A.9) non ou infiniment peu déformées. Comme prévu, le coefficient de distorsion radiale est beaucoup plus important pour la caméra 2.

4. D'après les données fournies par le constructeur de la caméra 1, la matrice de la caméra est constituée de 208 lignes et 288 colonnes, la taille d'un pixel élémentaire est de 28×15 μm^2 et la distance focale est de 9 millimètres. On en déduit $F_x \sim \frac{9}{28.10^{-3}}.\frac{256}{208} = 396$ et $F_y \sim \frac{9}{15.10^{-3}} = 600$. Ces valeurs sont bien du même ordre de grandeur que celles obtenues par identification du modèle de la caméra. La différence importante entre ces valeurs et celles obtenues par identification confirme la faible fiabilité des données fournies par un constructeur.

A.3.5 Simplifications

Une fois les paramètres intrinsèques déterminés, on peut effectuer la même démarche pour identifier uniquement la matrice M_c^o. Les simplifications donnent le nouveau système :

$$A.X_1 + B.X_2 = 0 \tag{A.31}$$

avec

$$X_1 = (a_{31}, a_{32}, a_{33})^T$$

$$X_2 = (a_{11}, a_{12}, a_{13}, a_{21}, a_{22}, a_{23}, X_o, Y_o, Z_o)^T$$

$$A = \begin{pmatrix} (X_p - X_c)x_o & (X_p - X_c)y_o & (X_p - X_c)z_o \\ (Y_p - Y_c)x_o & (Y_p - Y_c)y_o & (Y_p - Y_c)z_o \end{pmatrix}$$

$$B = \begin{pmatrix} -F_x x_o & -F_x y_o & -F_x z_o & 0 & 0 & 0 & -F_x & 0 & X_p - X_c \\ 0 & 0 & 0 & -F_y x_o & -F_y y_o & -F_y z_o & 0 & -F_y & Y_p - Y_c \end{pmatrix}$$

La résolution de ce système par la méthode de Lagrange, en utilisant la même contrainte sur X_1, donne également des résultats biaisés. Aussi emploie-t-on la même méthode de minimisation non linéaire avec seulement les six paramètres extrinsèques X_o, Y_o, Z_o, T_x, T_y et T_z à identifier (voir tableau A.4).

On peut vérifier lors des expérimentations réelles la précision des résultats obtenus en comparant les positions des points dans l'image numérisée et les positions des points reconstruits par le modèle de la caméra. On constate que quelques points ont parfois une erreur supérieure à 1 pixel : celà est dû à des erreurs de mesure mais surtout aux défauts inévitables de l'objectif.

En moyenne, l'erreur est de l'ordre du quart de pixel, ce qui permet d'affirmer que les paramètres du modèle de la caméra ont été correctement identifiés.

	méthode de Lagrange				méthode non linéaire			
	$\varepsilon = 0.5$ pixel							
	$\bar{m}$	σ	min	max	$\bar{m}$	σ	min	max
X_o	-99.858	0.17	-100.44	-99.35	-99.997	0.07	-100.15	-99.81
Y_o	-119.848	0.18	-120.49	-119.44	-119.989	0.04	-120.09	-119.89
Z_o	499.122	0.97	496.98	502.63	499.980	0.13	499.61	500.44
T_x	14.926	0.05	14.79	15.04	14.915	0.01	14.85	14.95
T_y	16.921	0.04	16.80	17.02	16.915	0.01	16.86	16.94
T_z	31.362	0.01	31.33	31.39	31.364	0.01	31.33	31.38
	$\varepsilon = 1.0$ pixel							
	$\bar{m}$	σ	min	max	$\bar{m}$	σ	min	max
X_o	-99.514	0.34	-100.70	-98.81	-99.993	0.11	-100.21	-99.66
Y_o	-119.467	0.37	-120.68	-118.52	-119.998	0.09	-120.23	-119.71
Z_o	496.798	1.91	492.25	503.03	500.002	0.23	499.24	500.69
T_x	14.952	0.09	14.76	15.16	14.911	0.03	14.80	14.99
T_y	16.942	0.09	16.68	17.17	16.918	0.04	16.80	16.99
T_z	31.355	0.02	31.30	31.41	31.363	0.01	31.31	31.41
	$\varepsilon = 1.5$ pixel							
	$\bar{m}$	σ	min	max	$\bar{m}$	σ	min	max
X_o	-98.874	0.57	-100.80	-97.25	-99.983	0.22	-100.63	-99.40
Y_o	-118.815	0.56	-120.50	-117.21	-120.010	0.14	-120.39	-119.71
Z_o	492.869	3.35	481.44	501.86	499.965	0.39	499.08	500.85
T_x	15.016	0.15	14.64	15.36	14.910	0.04	14.78	15.06
T_y	16.981	0.12	16.68	17.31	16.915	0.06	16.72	17.07
T_z	31.346	0.04	31.25	31.42	31.364	0.04	31.27	31.45
	$\varepsilon = 2.0$ pixel							
	$\bar{m}$	σ	min	max	$\bar{m}$	σ	min	max
X_o	-97.957	0.67	-99.71	-96.37	-99.962	0.26	-100.61	-99.23
Y_o	-117.813	0.69	-119.37	-116.19	-120.018	0.18	-120.50	-119.47
Z_o	486.788	3.80	478.17	497.09	500.025	0.48	498.92	501.39
T_x	15.087	0.21	14.22	15.63	14.905	0.07	14.68	15.08
T_y	17.012	0.13	16.76	17.43	16.908	0.08	16.68	17.12
T_z	31.328	0.05	31.16	31.45	31.366	0.04	31.24	31.50

Tableau A.4 : Simulation des deux méthodes de calibration

A.4 Identification de la matrice de passage entre le poignet du robot et la caméra

A.4.1 Mise en équations

Dans la première partie de cette annexe, nous avons montré qu'il est possible de régler précisément les zéros angulaires du robot afin de connaître la position et l'orientation du poignet par rapport à un repère de référence, fixe.

Nous allons maintenant identifier la matrice M_p^c, dernier maillon de la chaîne cinématique du système robot-caméra, représentant la situation de la caméra (et plus précisément de son centre optique) par rapport au poignet.

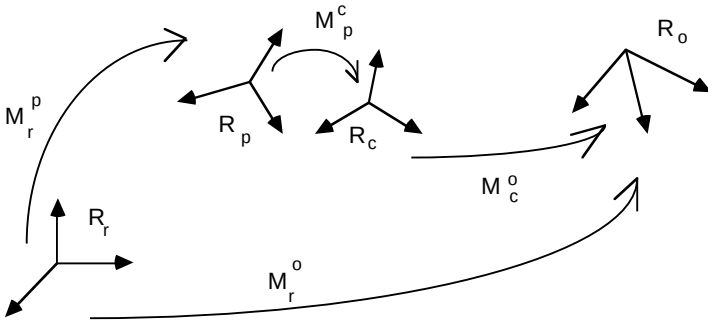


Figure A.10 :

On a (voir figure A.10)

$$M_r^o = M_r^p . M_p^c . M_c^o \tag{A.32}$$

ce qui donne :

$$M_p^c = M_r^{p^{-1}} . M_r^o . M_c^{o^{-1}} \tag{A.33}$$

Si M_r^p et M_c^o peuvent être connues (M_r^p en utilisant le transformateur de coordonnées, M_c^o par calibration), il est, par contre, impossible de connaître précisément la matrice M_r^o bien que celle-ci soit choisie constante. Aussi allons-nous utiliser des mouvements du poignet, et donc de la caméra, pour déterminer M_p^c.

Pour des positions distinctes i et j, on a :

$$M_r^o = M_{r_{(i)}}^p . M_p^c . M_{c_{(i)}}^o = M_{r_{(j)}}^p . M_p^c . M_{c_{(j)}}^o \tag{A.34}$$

donc

$$M_{r_{(j)}}^{p^{-1}}.M_{r_{(i)}}^{p}.M_{p}^{c} = M_{p}^{c}.M_{c_{(j)}}^{o}.M_{c_{(i)}}^{o^{-1}} \tag{A.35}$$

En décomposant ces matrices homogènes en leur partie rotation et translation, on obtient :

$$R_{r_{(j)}}^{p^{-1}}.R_{r_{(i)}}^{p}.R_{p}^{c} = R_{p}^{c}.R_{c_{(j)}}^{o}.R_{c_{(i)}}^{o^{-1}} \tag{A.36}$$

$$\begin{aligned}(R_{r_{(j)}}^{p^{-1}}.R_{r_{(i)}}^{p} - I_3).T_{p}^{c} &= R_{p}^{c}.(T_{c_{(j)}}^{o} - R_{c_{(j)}}^{o}.R_{c_{(i)}}^{o^{-1}}.T_{c_{(i)}}^{o}) \\ &\quad + R_{r_{(j)}}^{p^{-1}}.(T_{r_{(j)}}^{p} - T_{r_{(i)}}^{p})\end{aligned} \tag{A.37}$$

Notons :

$$\begin{array}{ll} R_{r_{(ij)}}^{p} = R_{r_{(j)}}^{p^{-1}}.R_{r_{(i)}}^{p} & T_{r_{(ij)}}^{p} = R_{r_{(j)}}^{p^{-1}}.(T_{r_{(j)}}^{p} - T_{r_{(i)}}^{p}) \\ R_{c_{(ij)}}^{o} = R_{c_{(j)}}^{o}.R_{c_{(i)}}^{o^{-1}} & T_{c_{(ij)}}^{o} = T_{c_{(j)}}^{o} - R_{c_{(j)}}^{o}.R_{c_{(i)}}^{o^{-1}}.T_{c_{(i)}}^{o} \end{array}$$

on a alors :

$$R_{r_{(ij)}}^{p}.R_{p}^{c} = R_{p}^{c}.R_{c_{(ij)}}^{o} \tag{A.38}$$

et

$$(R_{r_{(ij)}}^{p} - I_3).T_{p}^{c} = R_{p}^{c}.T_{c_{(ij)}}^{o} + T_{r_{(ij)}}^{p} \tag{A.39}$$

D'après (A.39), le calcul de T_p^c ne peut évidemment s'effectuer qu'après celui de R_p^c.

Par ailleurs, deux positions du robot sont insuffisantes pour résoudre les systèmes (A.38) et (A.39) (pour (A.39) par exemple, on sait que $R - I_3$ est de rang 2 quel que soit $R \neq I_3$). Une troisième position du poignet, au moins, est donc nécessaire pour pouvoir résoudre ces systèmes.

A.4.2 Résolution – Discussion

La résolution du système (A.38) est loin d'être triviale [Gantmatcher 60] mais les propriétés des matrices de rotation permettent d'aboutir [Shiu 87].

- Soit $P = 2\sin(\frac{\theta}{2}).U$ et $P' = 2\tan\frac{\theta}{2}$ deux représentations des rotations obtenues à partir de (U, θ), alors, la résolution du système (A.38) est équivalente à celle du système linéaire suivant [Tsai 87b]:

$$As(P_{r_{(ij)}}^{p} + P_{c_{(ij)}}^{o}).P_{p}^{'c} = P_{c_{(ij)}}^{o} - P_{r_{(ij)}}^{p} \tag{A.40}$$

Comme As(U) est de rang 2, quelque soit U, il faut au moins une troisième orientation k du poignet (telle que deux des axes $P^p_{r_{(ij)}}, P^p_{r_{(ik)}}$ et $P^p_{r_{(kj)}}$ au moins aient des directions différentes) pour assurer un système de rang plein.

De plus, pour augmenter la précision dans la résolution du système (A.39) donnant T^c_p, nous nous fixons $T^p_{r_{(ij)}} = 0$, c'est-à-dire une position du poignet constante en translation : seules des rotations du poignet seront effectuées.

Enfin, pour atténuer les effets des erreurs de mesure, plus de trois orientations du poignet seront prises en compte pour résoudre, par la méthode des moindres carrés, les systèmes (A.38), puis (A.39).

On obtient les résultats suivants :

$$P'^c_p = \begin{pmatrix} -0.0622 \\ -0.0283 \\ 0.0011 \end{pmatrix} (\Rightarrow \theta = 1.98°) \Rightarrow R^c_p = \begin{pmatrix} 0.9999 & -0.0003 & -0.0143 \\ 0.0008 & 0.9995 & 0.0314 \\ 0.0143 & -0.0314 & 0.9994 \end{pmatrix}$$

$$T^c_p = \begin{pmatrix} 0 \text{ mm} \\ -8\text{mm} \\ 189 \text{ mm} \end{pmatrix}$$

Plusieurs expérimentations ont été effectuées, donnant les mêmes résultats au millimètre près pour la translation et au dixième de degré près pour les angles d'Euler de la rotation.

- Remarque : il existe une autre méthode, théoriquement plus simple, pour calculer la rotation R^c_p entre le poignet et la caméra : supposons l'orientation du poignet constante, on a alors $R^p_{r_{(ij)}} = R^o_{c_{(ij)}} = I_3$ et l'équation (A.39) se réécrit :

$$R^c_p.(T^o_{c_{(j)}} - T^o_{c_{(i)}}) = T^p_{r_{(j)}} - T^p_{r_{(i)}}$$

soit

$$R^c_p.\Delta T^o_c = \Delta T^p_r$$

Ainsi deux translations perpendiculaires du poignet permettent-elles de déterminer la matrice R^c_p : par exemple, la première colonne est donnée par ΔT^o_c correspondant à $\Delta T^p_r = (1\ \ 0\ \ 0)^T$, la seconde colonne par ΔT^o_c correspondant à $\Delta T^p_r = (0\ \ 1\ \ 0)^T$ et la troisième par produit vectoriel des deux premières.

En pratique, cette méthode s'avère beaucoup moins précise que la précédente : elle ne permet aucune correction des erreurs de mesure, d'autant plus que les déplacements du robot sont moins précis en translation qu'en rotation. En effet, les positions de référence des axes 2 et 3 (qui donnent la translation du poignet) n'ont pu être obtenues parfaitement, comme on l'a vu dans la première partie de cette annexe.

A.5 Validation des résultats

La transformation repère de l'organe terminal- repère de la caméra a été rajoutée dans le transformateur de coordonnées afin de connaître la position et l'orientation du centre optique de la caméra par rapport à notre repère de référence en fonction des coordonnées articulaires du robot : ($M_r^c = M_r^p.M_p^c$).

Pour vérifier les résultats obtenus, nous avons considéré plusieurs positions de la caméra et effectué à chaque position correspondante une calibration donnant M_c^o, position et orientation du repère caméra au repère objet; on doit vérifier :

$$M_r^o = M_{r_{(i)}}^c.M_{c_{(i)}}^o = M_{r_{(j)}}^c.M_{c_{(j)}}^o = ... \qquad \text{(A.41)}$$

Nous obtenons (les translations sont données en millimètres):

$$M_{r_{(1)}}^c = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1000 \end{pmatrix}, M_{c_{(1)}}^o = \begin{pmatrix} 0.9992 & -0.0405 & -0.0043 & -156 \\ 0.0404 & 0.9986 & -0.0335 & -166 \\ 0.0057 & 0.0333 & 0.9994 & 609 \end{pmatrix}$$

$$\Rightarrow M_r^o = \begin{pmatrix} 0.9992 & -0.0404 & -0.0044 & -156 \\ 0.0403 & 0.9986 & -0.0335 & -166 \\ 0.0057 & 0.0333 & 0.9994 & 1609 \end{pmatrix}$$

$$M_{r_{(2)}}^c = \begin{pmatrix} 1 & 0 & 0 & 50 \\ 0 & 1 & 0 & 50 \\ 0 & 0 & 1 & 1050 \end{pmatrix}, M_{c_{(2)}}^o = \begin{pmatrix} 0.9990 & -0.0443 & -0.0023 & -204 \\ 0.0442 & 0.9985 & -0.0323 & -214 \\ 0.0037 & 0.0321 & 0.9995 & 559 \end{pmatrix}$$

$$\Rightarrow M_r^o = \begin{pmatrix} 0.9990 & -0.0443 & -0.0022 & -155 \\ 0.0442 & 0.9985 & -0.0322 & -165 \\ 0.0036 & 0.0320 & 0.9995 & 1609 \end{pmatrix}$$

$$M^{c}_{r_{(3)}} = \begin{pmatrix} 0.9924 & 0.0869 & -0.0871 & 50 \\ -0.0944 & 0.9917 & -0.0867 & 50 \\ 0.0789 & 0.0943 & 0.9924 & 1050 \end{pmatrix},$$

$$M^{o}_{c_{(3)}} = \begin{pmatrix} 0.9881 & -0.1330 & 0.0775 & -140 \\ 0.1282 & 0.9897 & 0.0640 & -177 \\ -0.852 & -0.0533 & 0.9949 & 592 \end{pmatrix}$$

$$\Rightarrow M^{o}_{r} = \begin{pmatrix} 0.9991 & -0.0414 & -0.0043 & -155 \\ 0.0412 & 0.9987 & -0.0302 & -163 \\ 0.0055 & 0.0300 & 0.9995 & 1609 \end{pmatrix}$$

La quasi-constance de la matrice M^{o}_{r} confirme la validité des méthodes employées, aussi bien l'identification du robot que la modélisation de la caméra et l'identification de la liaison entre la caméra et le poignet du robot.

Bibliographie

[Abi-Ayad 89] A. Abi-Ayad, B. Thiesse, C. Ragi : *Les multiples facettes du problème de calibrage de caméras*, 7^e congrès AFCET-RFIA, Tome 1, pp. 507-526, Paris, Décembre 1989.

[Agin 77] G. J. Agin : *Servoing with visual feedback*, Proc. of 7^{th} Int. Symp. on Industrial Robotics, pp. 551-560, Tokyo, Japan, October 1977.

[Agin 79] G. J. Agin : *Real Time Control of a Robot with a Mobile Camera*, Technical Note 179, SRI International, February 1979.

[Albus 81] J. S. Albus, A. J. Barbera, M. L. Fitzgerald : *Hierarchical Control for Sensory Interactive Robots*, Proc. of 11^{th} Int. Symp. on Industrial Robotics, pp. 497-505, Tokyo, Japan, October 1981.

[Alomoinos 87] J. Y. Alomoinos, I. Weiss, A. Bandyopadhyay : *Active Vision*, Proc. of 1^{st} IEEE Conf. on Computer Vision, pp. 35-54, London, March 1987.

[André 85] G.André, R. Fournier : *Generalized End Effector Control in a Computer Aided Teleoperation System with Application to Motion Coordination of a Manipulator Arm on a Oscillating Carrier*, International Conference on Advanced Robotics, Tokyo, Japan, September 1985.

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

[Bajcsy 88] R. Bajcsy : *Active Perception*, Research Report MS-CIS-88-24, University of Pennsylvania, Philadelphia, March 1988.

[Balek 85] D.J. Balek, R.B. Kelley : *Using Gripper Mounted Infrared Proximity Sensors for Robot Feedback Control*, IEEE Int. Conf. on Robotics and Automation, Saint Louis, Missouri, USA, March 1985.

[Birk 81] J. Birk, R. Kelley, H. Martins : *An orienting robot for finding workpieces stored in bins*, IEEE Trans. on Systems, Man and Cybernetics, Vol. SMC 11, n. 2, pp 151-160, February 1981.

[Blisset 88] R. J. Blisset, D. Charnley, C.G. Harris : *Towards robot mobility through passive monocular vision*, Proc. of Int. Symposium on Teleoperation and Control, pp. 123-132, July 1988.

[Boissonnat 88] J. D. Boissonnat, O. D. Faugeras, E. Le Bras-Mehlan : *Representing stereo data with the Delaunay triangulation*, Rapport de Recherche INRIA, n. 788, Février 1988.

[Brown 71] D.C. Brown : *Close Range Camera Calibration*, Photogrammetric Engineering, Vol. 37, n. 8, pp. 855-866, 1971.

[Chaumette 89a] F. Chaumette, P. Rives : *Réalisation et calibration d'un système expérimental de vision composé d'une caméra embarquée sur un robot manipulateur*, Rapport de Recherche INRIA, n. 994, Mars 1989.

[Chaumette 89b] F. Chaumette, P. Rives : *Modélisation et calibration d'une caméra*, 7^e congrès AFCET-RFIA, Tome 1, p. 527 - 536, Paris, Décembre 1989.

[Cheung 89] E. Cheung, V. Lumelsky : *Proximity Sensing in Robot Manipulator Motion Planning: System and Implementation Issues*, IEEE Transactions on Robotics and Automation, Vol.5, n. 6, December 1989.

[Clark 89] J. J. Clark, N. J. Ferrier : *Control of Visual Attention in Mobile Robots*, Conf. IEEE Robotics and Automation, pp. 826-831, Scottsdale, Arizona, USA, May 1989.

[Corke 89] P. I. Corke, R. P. Paul : *Video-Rate Visual Servoing for Robots*, First International Symposium on Experimental Robotics, Montreal, Canada, June 1989.

[Corre 88] J. L. Corre : *Système graphique interactif d'aide à l'intervention sous-marine*, Rapport convention IRISA/IFREMER, contrat n. 86/2.350.522, Octobre 1988.

[Coulon 83] P. Y. Coulon, M. Nougaret : *Use of a TV camera system in closed-loop position control mechanisms*, Int. Trends in Manufacturing Technology : Robot Vision, A. Pugh Ed., Bedford, UK:IFS Pub. Ltd., pp. 117-127, 1983.

[Crowley 90] J.L. Crowley, P. Stelmaszyk : *Measurement and Integration of 3-D Structures by Tracking Edge Lines*, Proceedings of First European Conference on Computer Vision, pp. 269 - 280, Antibes, France, April 1990.

[Dhome 89] M. Dhome, M. Richetin, J. T. Lapreste, G. Rives : *Determination of the attitude of 3D objects from a single perspective view*, IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 11, n. 12, pp. 1265-1278, December 1989.

[Dickmanns 85] E. D. Dickmanns, A. Zapp : *Guiding Land Vehicles along Roadways by Computer Vision*, Congrès "Automatique 1985 - the tools for tomorrow", AFCET, pp. 233-244, Toulouse, France, Octobre 1985.

[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.

[Dzialo 86] K. A. Dzialo, R. J. Schalkoff : *Control Implications in Tracking Moving Objects Using Time-Varying Perspective-Projective Imagery*, IEEE Trans. on Industrial Electronics, Vol. 33, no. 3, pp. 247-253, August 1986.

[Errami 86] M. Errami : *Vision temps réel pour le pilotage de robots industriels*, Rapport interne du Laboratoire d'Electronique, Université Blaise Pascal, Clermont-Ferrand II, 1986.

[Espiau 85] B. Espiau, R. Boulic : *Collision Avoidance for Redundant Robots with Proximity Sensors*, Third International Symposium on Robotics Research, Gouvieux, France, October 1985.

[Espiau 87a] B. Espiau, P. Rives : *Closed-Loop Recursive Estimation of 3D Features for a Mobile Vision System*, IEEE Int. Conf. on Robotics and Automation, Raleigh, North Carolina, USA, April 1987.

[Espiau 87b] B. Espiau : *Sensory-based Control: Robustness Issues and Modelling Techniques; Application to Proximity Sensing*, NATO

Advanced Research Workshop on Kinematic and Dynamic Issues in Sensor-based Control, pp. 3-44, Italy, October 1987.

[Espiau 90] B. Espiau, F. Chaumette, P. Rives : *Une nouvelle approche de la liaison vision-commande en robotique*, Rapport de Recherche INRIA, n. 1172, Avril 1990.

[Faugeras 86] O.D. Faugeras, N. Ayache, B. Faverjon : *Building visual maps by combining noisy stereo measurements*, Conf. IEEE Robotics and Automation, pp. 1433-1469, San Fransisco, USA, April 1986.

[Faugeras 87] O.D. Faugeras, G. Toscani : *Camera Calibration for 3D Computer Vision* Proceedings of International Workshop on Machine Vision and Machine Intelligence, Tokyo, February 1987.

[Faugeras 88] O.D. Faugeras : *A few step toward artificial 3D vision*, Rapport de Recherche INRIA, n. 790, Février 1988.

[Faugeras 90] O.D. Faugeras, S. Maybank : *Motion from point matches : multiplicity of solutions*, Rapport de Recherche INRIA, n. 1157, Février 1990.

[Faverjon 84] B. Faverjon : *Obstacle avoidance using an octree in the configuration space of a manipulator*, Proc. of the IEEE Int. Conf. on Robotics and Automation, pp. 504-512, Atlanta, USA, March 1984.

[Feddema 89a] J. T. Feddema, C. S. G. Lee and O. R. Mitchell : *Automatic selection of image features for visual servoing of a robot manipulator*, Conf. IEEE Robotics and Automation, pp. 832-837, Scottsdale, Arizona, USA, May 1989.

[Feddema 89b] J. T. Feddema, O. R. Mitchell : *Vision-Guided Servoing with Feature-Based Trajectory Generation*, IEEE Trans. on Robotics and Automation, Vol. 5, n. 5, pp. 691-700, October 1989.

[Flynn 88] A. Flynn : *Combining Sonar and Infrared Sensors for Mobile Robot Navigation*, International Journal of Robotics Research, Vol 7, n. 6, December 1988.

[Gantmatcher 60] F.R. Gantmatcher : Matrix Theory, *Chelsea Publishing Company*, Vol. 1, New York, 1960.

[Gilbert 80] A. L. Gilbert, K.G. Giles, G. M. Flachs, R. B. Rogers, Y. H. U : *A Real-Time Video Tracking System*, IEEE Transaction on PAMI, Vol. PAMI-2, n.1, pp. 47-56, January 1980.

[Gill 78] P.E. Gill, W. Murray : *Algorithms for the solution of non-linear least squares problem*, SIAM Journal on Numerical Analysis, Vol. 15, n. 5, pp. 977-992, October 1978.

[Guyot 89] G. Guyot : *Calibration simulee d'un robot manipulateur a l'aide de capteur de vision*, Rapport de DEA Traitement du signal, Universite de Nice, Juin 89

[Horaud 87] R. Horaud : *New methods for matching 3D objects with single perspective view*, IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 9, n. 3, pp. 401-412, May 1987.

[Kabuka 88] M. Kabuka, E. McVey, P. Shironoshita : *An Adaptive Approach to Video Tracking*, IEEE Journal of Robotics and Automation, Vol. 4, n. 2, pp. 228-236, April 1988.

[Kashioka 77] S. Kashioka, S. Takeda, Y. Shima, T. Uno, T. Hamada : *An approach to the integrated intelligent robot with multiple sensory feedback : visual recognition techniques*, Proc. of 7^{th} Int. Symp. on Industrial Robotics, pp. 531-538, Tokyo, Japan, October 1977.

[Kuan 87] D. Kuan, G. Phipps, A. C Hsuch : *Autonomous Land Vehicle Road Following*, Proc. of 1^{st} IEEE Conf. on Computer Vision, pp. 557-566, London, March 1987.

[Le Borgne 87] M. Le Borgne : *Quaternion et contrôle sur l'espace des rotations*, Rapport de Recherche INRIA, n. 751, Novembre 1987.

[MACSYMA 85] *VAX UNIX MACSYMA Reference Manual*, Symbolics Inc., Eleven Cambridge Center, Cambridge, November 1985.

[Merlet 88] J.P. Merlet : *Parallel Manipulators, Part 2 : Theory - Singular configurations and Grassmann Geometry*, Rapport de Recherche INRIA, n. 791, Février 1988.

[NAG 84] *Numerical Algorithms Group : Fortran Library Manual*, Mayfield House, Vol. 3, Oxford, 1984.

[Reid 87] J. F. Reid, S. W. Searcy : *Vision-Based Guidance of an Agricultural Tractor*, IEEE Control Systems Magazine, pp. 39-43, April 1987.

[Rives 81] P. Rives, P. Bouthemy, B. Prasada, E. Dubois : *Recovering the orientation and the position of a rigid body in space from a single view*, Technical Report, INRS Télécommunications, Quebec, Canada, 1981.

[Rives 87] P. Rives : *Dynamic vision: theoretical capabilities and practical problems*, NATO Advanced Research Workshop on Kinematic and Dynamic Issues in Sensor Based Control, pp. 251-280, Italy, October 1987.

[Rives 89] P. Rives, F. Chaumette, B. Espiau : *Visual Servoing Based on a Task Function Approach*, First International Symposium on Experimental Robotics, Montréal, Canada, June 1989.

[Samson 87] C. Samson : *Une approche pour la synthèse et l'analyse de la commande des robots manipulateurs*, Rapport de recherche INRIA, n. 669, Mai 1987.

[Samson 88] C. Samson, B. Espiau, M. Le Borgne : *Robot Redundancy : an Automatic Control Approach*, NATO Advanced Research Workshop on Robots with Redundancy, Salo, Italia, Juin 1988.

[Samson 90a] C. Samson, B. Espiau : *Application of the Task Function Approach to Sensor-Based-Control of Robot Manipulators*, IFAC, Tallin, USSR, July 1990.

[Samson 90b] C. Samson, B. Espiau, M. Le Borgne : *Robot Control : the Task Function Approach*, Oxford University Press, 1990.

[Sandini 90] G. Sandini, M. Tistarelli : *Active tracking strategy for monocular depth inference over multiple frames*, IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 12, n. 1, pp. 13-27, January 1990.

[Sanderson 80] A. C. Sanderson, L. E. Weiss : *Image Based Visual Servo Control Using Relational Graph Error Signal*, Proc. of the Int. Conf. on Cybernetics and Society, Cambridge, MA, IEEE SMC, pp. 1074-1077, October 1980.

[Sanderson 83] A. C. Sanderson, L. E. Weiss : *Adaptive Visual Servo Control of Robots*, Reprinted in *Robot Vision*, A. Pugh Ed., Bedford, UK:IFS Pub. Ltd., pp. 107-116, 1983.

[Shiu 87] Y.C. Shiu, S. Ahmad : *Finding the Mounting Position of a Sensor by Solving a Homogeneous Transform Equation AX=XB*, Proceedings of IEEE International Conference on Robotics and Automation, Raleigh, North California, USA, pp. 1666-1671, April 1987.

[Sobel 74] I. Sobel : *On Calibrating Computer Controlled Cameras for Perceiving 3D Scenes*, Artificial Intelligence, Vol. 5, pp. 185-198, 1974.

[Tani 77] K. Tani, M. abe, K. Tanie, T. Ohno : *High Precision Manipulators with Visual Sense*, Proc. of 7^{th} Int. Symp. on Industrial Robotics, pp. 561-568, Tokyo, Japan, October 1977.

[Tsai 87a] R.Y. Tsai : *A Versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology using Off-the-Shelf TV Cameras and Lenses*, IEEE Journal of Robotics and Automation, Vol. RA-3, n. 4, pp. 323-344, August 1987.

[Tsai 87b] R.Y. Tsai, R. Lenz : *A New Technique for Autonomous and Efficient 3D Robotics Hand-Eye Calibration*, Research Report RC13212 IBM T.J. Watson Research Center, Yorktown Heights, NY 10598, October 1987.

[Urban 90] J.P. Urban : *Une approche de type asservissement visuel appliquée a la robotique*, Thèse de l'Universite Blaise Pascal, Clermont-Ferrand II, Juin 1990.

[Wampler 84] C. Wampler: *Multiprocessor Control of a Telemanipulator with Optical Proximity Sensors*, International Journal of Robotics Research, Vol. 3, n. 1, Spring 1984.

[Ward 79] M. R. Ward, L. Rossol, S. W. Holland, R. Dewar : *CONSIGHT : A Practical Vision-Based Robot Guidance System*, Proc. of 9^{th} Int. Symp. on Industrial Robotics, pp. 195-211, Washington, U.S.A, March 1979.

[Weiss 84] L. E. Weiss : *Dynamic Visual Servo Control of Robots. An Adaptive Image based Approach*, Technical Report, CMU-RI-TR-84-16; Carnegie Mellon, April 1984.

[Weiss 87] L. E. Weiss, A. C. Sanderson : *Dynamic Sensor-Based Control of Robots with Visual Feedback*, IEEE Journal of Robotics and Automation, Vol. RA-3, n. 5, pp. 404-417, October 1987.

[Xie 89] M. Xie : *Contribution à la vision dynamique : reconstruction d'objets 3D polyédriques par une caméra mobile*, Thèse de l'Université de Rennes I, Rennes, Juin 89.