Mémoire

Présenté à

L'UNIVERSITE ABOU BEKR BELKAÏD
FACULTE DES SCIENCES DE L'INGENIEUR
DEPARTEMENT D'ELECTRONIQUE



Pour l'obtention du diplôme de :

*Magister*

En **ELECTRONIQUE**

Option : « ***Signaux et Systèmes*** »



Par

Mr HADJ ABDELKADER HICHAM

***Thème***

# ASSERVISSEMENT VISUEL D'UN BRAS MANIPULATEUR A CINQ DEGRES DE LIBERTE

Soutenu en Juillet 2002

*devant la commission d'examen*

**Président** :

Mr N. GHOUALI — Professeur à l'université Abou Bekr Belkaïd, Tlemcen

**Examinateurs** :

Mr F. BEREKSI REGUIG — Professeur à l'université Abou Bekr Belkaïd, Tlemcen
Mr M.F. KHELFI — Maître de Conférence à l'université d'Oran – Es-Sénia

**Rapporteurs**:

Mr B. CHERKI — Maître de Conférence à l'université Abou Bekr Belkaïd, Tlemcen
Mr C. KARA TERKI — Chargé de Cours à l'université Abou Bekr Belkaïd, Tlemcen

*A mes chers parents,*
*à toute ma famille,*
*à tous ceux qui me sont chers.*

# REMERCIEMENTS

Je tiens remercie tout particulièrement, Monsieur B. CHERKI, Maître de conférence à l'université Abou Bekr Belkaïd Tlemcen et Monsieur C. KARA TERKI Chargé de cours à l'université Abou Bekr Belkaïd Tlemcen, pour l'intérêt constant qu'ils ont porté à ce travail. Leurs compétences scientifiques autant que leurs conseils et encouragements m'ont été très précieux pour mener à bien la réalisation de ce travail. Qu'ils trouvent ici le témoignage de ma haute considération.

J'exprime ma reconnaissance à Monsieur N. GHOUALI, Professeur à l'université Abou Bekr Belkaïd Tlemcen, pour avoir bien voulu présider ce jury.

Je remercie également très vivement Monsieur F. BEREKSI REGUIG, Professeur à l'université Abou Bekr Belkaïd Tlemcen, et Monsieur M.F KHELFI Maître de conférence à l'université d'Oran - Es-Sénia , pour l'intérêt qu'ils ont porté à ce travail et pour avoir accepté de l'examiner.

Pour finir, je ne manquerai pas de remercier très sincèrement mes amis et collègues Samad, Zaky et Brahim.

# Table des matières

# Introduction générale

Le contrôle des mouvements d'un robot en utilisant la vision par ordinateur, ou asservissement visuel, intéresse depuis quelques années beaucoup d'industriels. Les applications pratiques considérées sont des tâches de positionnement contrôlant les n degrés de liberté de l'effecteur du robot afin de saisir des outils et les utiliser, positionner des objets et réaliser des tâches d'inspection. La grande généralité des techniques d'asservissement visuel devrait faciliter leur extension à des applications dans la robotique manufacturière, spatiale ou mobile.

Les capteurs visuels apportent aux systèmes robotiques une plus grande connaissance de l'environnement dans lequel ils évoluent. L'utilisation des cameras pourra peut être un jour rendre les robots complètement autonomes dans des milieux inconnus ou difficilement modélisables. En effet, les progrès récents effectués en vision par ordinateur, comme la reconstruction projective et euclidienne d'objets réels, leur reconnaissance et leur suivi, ont ouvert de nouvelles perspectives. Certes, l'utilisation de manière robuste de la plupart de ces techniques dans une boucle de commande à la cadence vidéo n'est pas encore possible. Toutefois, on peut dès maintenant s'interroger sur la manière d'intégrer une partie de ces innovations en vision par ordinateur dans un système robotique et sur les améliorations qu'il est possible d'obtenir en ce qui concerne les techniques d'asservissement visuel.

L'utilisation des techniques d'asservissement visuel sur un site industriel demande souvent à la fois une grande précision, une grande facilité de mise en œuvre, une grande portabilité et surtout une grande fiabilité. Ces quatre aspects pratiques ouvrent des problèmes théoriques très intéressants :

o Afin d'obtenir une grande précision, il est nécessaire dans certaines applications, comme par exemple le positionnement d'objets très encombrants, d'utiliser plusieurs caméras. A cause de la grande taille de l'objet considéré, les caméras observent généralement des parties différentes de la scène. Il s'agit donc de résoudre le problème de la fusion des données provenant de plusieurs capteurs.

o Afin d'obtenir une grande simplicité de mise en œuvre, il est souhaitable d'éliminer, ou simplifier, toutes les procédures de calibration du système, souvent trop lourdes et coûteuses ( surtout lors de l'utilisation de plusieurs caméras).

o Afin d'obtenir une grande portabilité des méthodes d'asservissement, il est souhaitable de recevoir un schéma de commande avec un besoin minimal de connaissances à priori sur les objets

observés par les caméras. Le cas idéal serait de n'utiliser aucune connaissance du modèle de l'objet.

o Afin d'obtenir une grande fiabilité tout en ayant éliminé la phase de calibration, les lois de commande doivent, autant que possible, présenter une grande robustesse aux incertitudes sur les paramètres du systèmes et assurer la convergence à partir de n'importe quelle position initiale.

Bien que les méthodes d'asservissement visuel existantes soient, pour certaines applications, très performantes, beaucoup de problèmes théoriques sont encore ouverts en ce qui concerne les tâches de positionnement. En effet, en l'état actuel, aucune technique d'asservissement visuel ne réunit à la fois toutes les caractéristiques évoquées.

Le premier objectif de notre travail a donc été de poser les bases de la conception d'une part, de techniques d'asservissement visuel qui ne nécessitent ni la connaissance du modèle de l'objet observé, ni une calibration précise du système, et d'autre part, de lois de commande robustes aux incertitudes sur les paramètres.

Dans le cadre de notre étude, nous nous intéresserons au positionnement d'une caméra embarquée sur un robot manipulateur, par rapport à un objet dont on peut extraire des points caractéristiques. Puisque le suivi d'objets réels n'est pas encore possible à cadence élevée, nous utiliserons des objets marqués à partir desquels il est facile d'extraire des informations géométriques à la cadence vidéo. Ainsi, nous supposerons que la mise en correspondance des points observés dans l'image a été déjà résolu, car la mise en correspondance automatique sans aucune connaissance du modèle de l'objet observé reste un problème difficile et loin d'être résolu et fait encore l'objet de nombreux travaux.

Notre travail est organisé de la manière suivante :

- Le premier chapitre traite des différentes approches d 'asservissement visuel nous citerons l'asservissement bidimensionnel et tridimensionnel, direct et indirect .

- Le second chapitre concerne la modélisation de la chaîne d'asservissement visuel, qui est constituée dans notre cas, d'une caméra et d'un robot à cinq degrés de liberté.

- Dans le troisième chapitre, nous étudierons la loi de commande utilisée dans l'asservissement visuel ainsi que des solutions qui améliorent cet asservissement.

- Le quatrième chapitre comprend les résultats de l'application de la commande d'asservissement visuel choisie sur un robot de type IR50p à cinq axes ainsi que les résultats du simulateur d'asservissement visuel élaboré par nos soins.

# Chapitre 1

# Etat de l'art de l'asservissement visuel

Dans ce chapitre, nous présentons l'Etat de l'art en asservissement visuel avec un regard particulier sur ce qui concerne les resultats obtenus par nombreux chercheurs en commande des robots manipulateur, et en commande référencée capteur.

## 1.1 Etat de l'art [11][17] [19][25][28][31][33][34] [35][46]

Il est souvent essentiel d'associer un capteur extéroceptif à un robot pour améliorer ses possibilités et ses performances dans l'exécution de différentes tâches. Parmi les différents capteurs possibles, la vision occupe une place privilégiée, car elle permet d'avoir des informations de grande richesse.

L'asservissement visuel est un domaine de recherche dont le but est d'intégrer en temps réel les informations visuelles dans la boucle d'asservissement des robots manipulateurs ou mobiles. En utilisant une camera embarquée, les techniques d'asservissement visuel permettent de fournir une interaction directe entre la position de l'effecteur du robot et son environnement. Il en résulte une commande plus précise et plus robuste en présence d'incertitudes et d'erreurs de modélisation. La tâche de base en asservissement visuel consiste à contrôler de façon stable et robuste l'attitude de l'effecteur du robot (Position et Orientation), potentiellement variable dans le temps, à l'aide d'informations visuelles. De nombreuses autre tâches peuvent également être réalisées comme par exemple le seul contrôle de l'orientation de la caméra. Dans tous les cas, elles reposent sur la mesure et l'utilisation des informations visuelles extraites en temps réel des images acquises par le capteur.

L'asservissement visuel touche à un certain nombre de domaines de recherche, incluant la cinématique, la dynamique et la commande des robot, le traitement d'image en temps réel, la vision active, le calcul temps réel et l'architecture système. En tant que tel, et comme le note J. Funda , c'est une discipline jeune et pleine de défis avec un grand nombre de résultats préliminaires fort intéressants, aussi bien qu'une multitude de problèmes encore ouverts.

La première contribution importante au domaine de la commande référencée vision est dûe à Weiss et Senderson. Ils ont proposé plusieurs schémas de commande permettant de réaliser des taches de positionnement. D'une part, la stratégie dite « Look and Move » où les aspects de vision et de commande sont considérés comme deux problèmes disjoints, et d'autre part, les techniques d'asservissement visuel, basées sur la régulation d'informations visuelles dans l'image (par exemple, les coordonnées de plusieurs points dans l'image).

Les techniques d'asservissement visuel, véritable boucle fermée sur les informations 2D, consistent à spécifier le problème de l'asservissement en termes de régulation dans l'image, et ne nécessitent pas la reconstruction tridimensionnelle de la scène à chaque itération de la boucle de commande. Les consignes ne sont plus exprimées sous la forme d'une attitude désirée entre la caméra et la scène, mais sous la forme d'un motif à atteindre dans l'image : les informations visuelles $S$ choisies pour constituer ce motif doivent atteindre les valeurs $S^*$ qui correspondent à une bonne réalisation de la tâche (Figure 1.1). Il en résulte par conséquent un gain non négligeable en précision de positionnement et en robustesse par rapport aux erreurs de modélisation et de calibration.

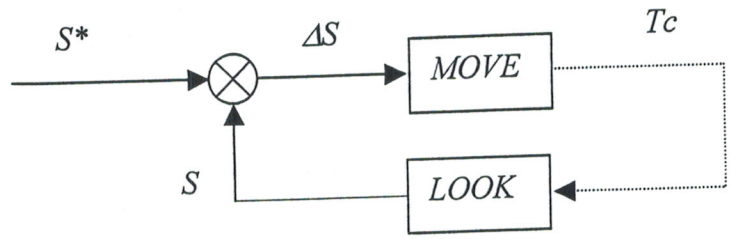



Figure1.1 : Schéma de l'asservissement visuel

Il est montré dans [11] que les informations visuelles utilisables en commande référencée vision peuvent être modélisées comme un ensemble $s$ de signaux visuels élémentaires $s_j$, qui dépendent uniquement de l'attitude entre la caméra et la scène. Par exemple, $s$ peut être constitué des coordonnées de plusieurs points, ou des paramètres décrivant des droites, des ellipses,... etc. En considérant l'image désirée et l'image observée par la caméra; le problème de la commande se réduit à la régulation de $(s - s^*)$ dans l'image.

Pour utiliser les techniques d'asservissement visuel, il s'avère indispensable de modéliser le lien entre les variations dans l'image des informations visuelles et le mouvement relatif entre la caméra et la scène. Ce lien est décrit par une matrice jacobienne, également appelée matrice d'interaction. Connaissant ce lien, il est possible, 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 ces informations. La commande calculée est généralement une consigne de vitesse 3D en translation, et en rotation de la caméra notée $T_c$. Son application fait bien évidemment intervenir le jacobien du Robot ainsi que la relation géométrique entre la caméra et l'extrémité du robot sur laquelle elle est montée. De par son approche en boucle fermée, les tâches d'asservissement visuel sont cependant robustes aux

éventuelles erreurs introduites par ces termes.

Pour les tâches de positionnement par rapport à des objets fixes, Feddema et *al* ont développé une méthode pour sélectionner automatiquement les informations visuelles à prendre en compte dans la commande dans le cas d'objets aux caractéristiques connues. Cette sélection est effectuée en fonction de l'éventuelle facilité d'extraction dans l'image des points 2D caractéristiques de l'objet, et d'un critère sur la commande basé sur le conditionnement de la matrice jacobienne des informations visuelles. Feddema et *al* ont également proposé une méthode permettant de réduire le temps de calcul du traitement d'image (et donc de la boucle d'asservissement) basée sur l'utilisation de la prédiction de la position de l'objet dans l'image, ils emploient un générateur de trajectoire des informations visuelles afin de garder ces derniers dans le champ de vue de la caméra et d'assurer une convergence lisse vers leur position désirée dans l'image.

Nelson et *al.* ont étudié l'utilisation de l'asservissement visuel pour des tâches d'assemblage. Le principal centre d'intérêt de ce travail concerne la modélisation et la commande d'un système caméra-robot, la détermination automatique de l'emplacement optimal de la caméra, et le calcul adaptatif et en ligne de la transformation caméra-objet.

Hashimoto et *al.* proposent une formulation de la commande par asservissement visuel, où le modèle est exprimé comme un système d'équations différentielles non linéaires dans l'espace d'état. Deux stratégies de commande différentes sont étudiées : D'une part, une commande linéaire obtenue par linéarisation autour du point de fonctionnement, et d'autre part, une commande non-linéaire obtenue par compensation et découplage de la dynamique du robot.

Concernant l'utilisation de caméras déportées, qui aboutissent par nature à des systèmes différents des systèmes embarqués (Eye-in-Hand), on peut citer les travaux récents de Dornaika qui nécessitent une calibration fine pour localiser l'objet d'intérêt par rapport à l'effecteur, et Hager qui utilise un système stéréoscopique non calibré.

Concernant les problèmes de suivi d'objet mobile, Papanikolopoulos et *al.* utilisent des techniques classiques de l'automatique théorique (commande optimale LQG, placement de pôle). Cependant, le mouvement de l'objet est considéré dans leur méthode comme une perturbation négligeable, ce qui entraine des erreurs de traînage au cours de suivi.

Nelson et *al.* prennent en compte les singularités et les butées mécaniques du robot afin de les éviter et agrandir l'espace des configurations de la caméra où le suivi reste possible. Ici encore, le mouvement de l'objet n'est pas estimé ou utilisé dans la commande.

Allen et *al.* décrivent un système pour la préhension dynamique d'objets en mouvement. Le système est constitué d'une paire de caméra fixes pour suivre en temps réel le mouvement plan d'un objet et utilise cette information pour commander un bras manipulateur pour saisir dynamiquement

l'objet en mouvement. Cette application intéressante réunit la poursuite visuelle en temps réel et la préhension stable et dynamique d'objets en mouvement. Ils ont développé un algorithme d'estimation du mouvement de l'objet afin de réduire le problème des erreurs de traînage. Cet algorithme, basé sur l'utilisation de filtre à coefficients constants, a le mérite d'être simple à mettre en oeuvre mais manque cependant d'efficacité pour des mouvements complexes.

Par ailleurs, Corke et *al.* se sont intéressées aux facteurs dont les effets limitent les performances des systèmes d'asservissement visuel.

Mis à part la modélisation particulière, le type de commande ou l'approche architecturale du problème d'asservissement visuel, toutes les recherches menées présentent des résultats significatifs dans le sens de l'intégration en temps réel de la vision et de la manipulation en robotique. Finalement, l'ensemble de ces travaux incluent une validation expérimentale des résultats théoriques.

## 1.2 Les differents types d'asservissement visuel [12][14]

Une classification des asservissements par vision a été établie par Sanderson. Bien qu'elle ait été réalisée en 1980 et que la discipline a beaucoup évolué ces dernières années, elle reste encore valable dans ses grandes lignes. Nous la compléterons en tenant compte des développements les plus récents.

Cette classification s'effectue en prenant en compte 3 critères :

**1. Le positionnement de la caméra :**

o *Eye-to-hand,* la caméra n'a pas de liaison mécanique avec le robot asservi par vision. Nous parlerons de configuration où la caméra est extérieure pour désigner ce type d'asservissement.

o *Eye-in-hand,* la caméra est montée sur l'organe terminal du robot.

**2. Les grandeurs asservies :**

o La référence et la mesure sont définies par une attitude. Un modèle géométrique de l'objet d'intérêt est nécessaire pour estimer la mesure. On parle alors d'asservissement <<3D>>.

o La référence, la mesure et la loi de commande de l'asservissement sont définies dans le plan image. Dans ce cas, on parle d'asservissement référencé capteur ou plus prosaïquement d'asservissement <<2D>>.

o La référence est une image mais la loi de commande est établie à partir des grandeurs exprimées dans le plan image et dans le repère caméra. Chaumette et al. ont baptisé ce mode de commande <<2D 1/2>>.

o La référence de l'asservissement est un champ de vitesse dans le plan image. Chaumette et al. ont donné à ce type d'asservissement l'appellation <<d2D/dt>>.

**3. Le type de commande appliquée au robot :**

o Le robot est commandé via le contrôleur de position articulaire fourni par le constructeur du robot. On dénommera ce mode de fonctionnent asservissement visuel indirect.

o La boucle de vision fournit directement les références aux contrôleurs de vitesse ou de couple articulaire des axes du robot. Ce type d'asservissement est défini par asservissement visuel direct.

### 1.2.1 Les asservissements où la caméra est extérieure (Eye-to-hand) [34]

La figure 1.2 décrit la configuration où la caméra est extérieure au robot qui est asservi. Elle est positionnée de manière à ce que l'organe terminal et les objets situés dans l'espace de travail du robot soient dans son champ de vision. Elle peut être fixe ou montée sur un autre système mécanique.

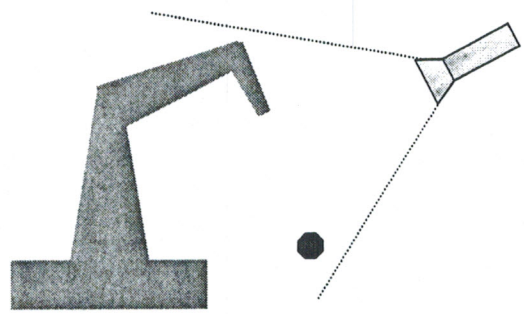


Figure1.2 : Description de la configuration où la caméra est extérieure

Dans cette configuration, l'information visuelle permet non seulement la mesure de l'attitude ou du changement d'attitude d'un objet situé dans l'espace de travail du robot, mais également la mesure de l'attitude ou du changement d'attitude de l'organe terminal.

Hager décrit la théorie et l'implémentation d'un système de commande des robots manipulateurs utilisant les informations visuel de deux caméras (Figure1.3). Le système suit en même temps l'effecteur du Robot et les primitives visuelles utilisées pour définir la position désirée. Par un feedback-vision, il est possible d'atteindre asymptotiquement le positionnement exact du système à la cible malgré l'erreur de calibrage.

Les figures 1.3(B) et 1.3(C) illustrent le principe de base pour un positionnement translationel.

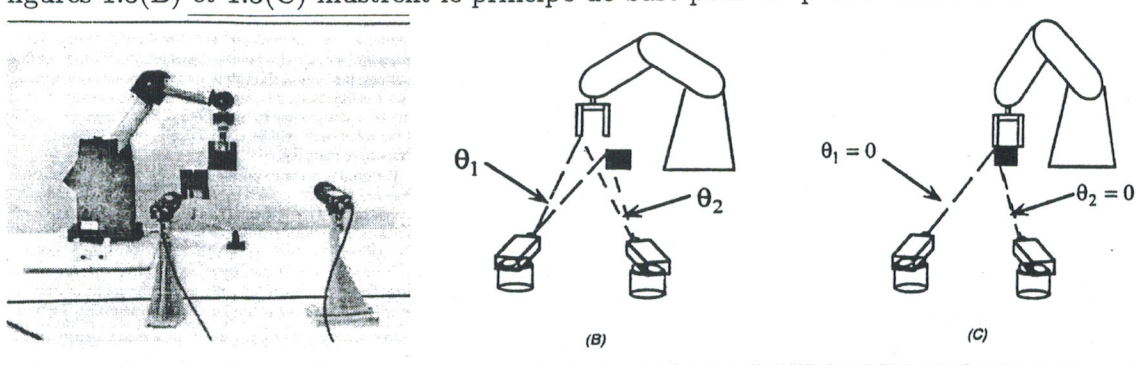



Figure1.3 : Vision stéreo

### 1.2.2 Les asservissements eye-in-hand

C'est la configuration la plus répandue et que nous avons adoptée dans notre travail. La caméra est attachée à l'organe terminal du manipulateur de manière à ce que les objets situés dans l'espace de travail soient dans son champ de vision (Figure 1.4). Dans cette configuration, la caméra est utilisée pour mesurer l'attitude ou le changement d'attitude d'un objet par rapport à l'organe terminal du robot.

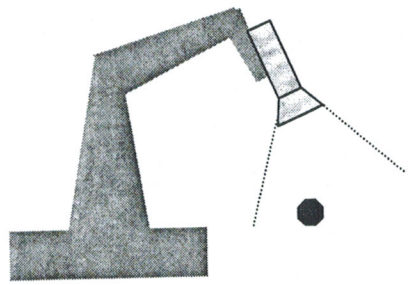


Figure1.4 : Description de la configuration eye in handh

### 1.2.3 Les asservissements 3D [51][52]

Dans un asservissement 3D, la référence est exprimée sous la forme d'une attitude. L'attitude d'un repère par rapport à un autre est définie par une translation et une rotation. Dans la configuration eye in hand, il s'agit de l'attitude $p^*$ d'un repère lié à un objet vu par la caméra par rapport à un repère lié à l'organe terminal. La mesure utilisée dans l'asservissement est une estimation $\hat{p}$ de l'attitude courante $p$ entre l'organe terminal et l'objet.

Cette mesure s'obtient grâce aux primitives extraites de l'image et à un modèle géométrique de la cible. Mais cette mesure est très sensible aux erreurs de calibration de la caméra. Grâce à l'asservissement, $\hat{p} = p^*$ en régime établi. Mais, comme $p \neq \hat{p}$ à cause des erreurs de modélisation, la position réelle du robot par rapport à la cible est biaisée. Ce problème est néanmoins contournable en réalisant un apprentissage de la référence $p^*$. Dans ce cas, le robot est placé par un opérateur de manière à ce que $p = p^*$. Une estimation $\hat{p}^*$ de la consigne est alors déterminée dans cette configuration. Si la consigne de l'asservissement est $\hat{p}^*$, on a donc en régime établi $\hat{p} = \hat{p}^*$. Comme, généralement, la relation qui lie l'attitude réelle $p$ à son estimée mesurée $\hat{p}$ est bijective, on en déduit que si $\hat{p} = \hat{p}^*$, alors $p = p^*$.

De nombreux articles traitent des techniques d'estimation de l'attitude entre une ou plusieurs caméras et un objet dont la géométrie est connue. La complexité des calculs nécessaires à cette estimation croît avec la complexité de l'objet. C'est pourquoi les objets sont souvent simplifiés au maximum dans les asservissements visuels. Par exemple , la cible est constituée d'un tétraèdre dont les sommets sont matérialisés par des points lumineux. De nombreuses manipulations mettent en

oeuvre une cible constituée uniquement de plusieurs disques coplanaires sur un fond uniforme. Le traitement d'image se limite alors à une détection des contours des disques et à un calcul du centre de gravité des points de contour afin de déterminer le centre de chaque disque (Figure 1.5).

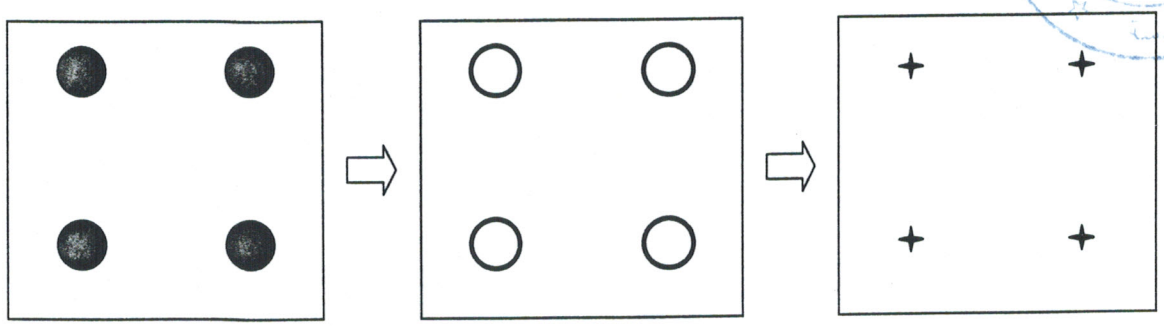


Figure1.5 : Traitement d'image d'une cible constituée de disques coplanaires

Le modèle de la cible, à savoir la position relative de tous les centres des disques, est connu par le système de vision qui peut donc calculer une estimation de l'attitude de l'objet. Il a été montré qu'il est nécessaire de connaître la projection d'au moins 4 primitives de type points pour pouvoir obtenir cette attitude de manière univoque (voir [52] par exemple). Si seulement 3 projections sont disponibles, l'attitude ne peut être déterminée de manière unique. Néanmoins, la connaissance de l'attitude à un instant d'échantillonnage précédent permet, dans certains cas, d'estimer la nouvelle attitude en choisissant celle qui se rapproche le plus de la précédente. Le problème de la reconstruction de l'attitude à partir de la projection de 3 ou 4 points est abordé par de nombreux auteurs. D'autres auteurs proposent une solution par les moindres carrés.

Un des développements les plus récents en matière d'asservissement 3D est décrit par Wilson. Une cible qui contient 5 trous circulaires non coplanaires est utilisée. Le suivi est réalisé suivant 5 degrés de liberté grâce à un bras manipulateur. Le système de vision utilisé est capable de mesurer la position de la projection des 5 trous dans une image binarisée de la cible à la cadence de 61Hz. Partant du constat que la binarisation d'une image est une opération introduisant beaucoup de bruit dans la détection des contours, un filtrage de Kalman de la mesure de l'attitude est réalisé. Ce dernier comprend un observateur très simple du mouvement de la cible par rapport à la caméra où la vitesse relative de déplacement est approximée par une constante. Le filtre de Kalman est également utilisé pour prédire la meilleure estimation (Au sens du minimum de la variance de l'erreur) de la position future des projections des trous dans l'image. Cette estimation est utilisée par le dispositif de traitement d'image pour affiner la position des zones d'intérêt dans l'image.Il s'agit d'un asservissement visuel indirect car il utilise les asservissements de position des axes du robot.

Le correcteur de la boucle de vision est du type Proportionnel Dérivé (PD) mais son réglage est réalisé expérimentalement, en l'absence de modélisation dynamique du manipulateur.

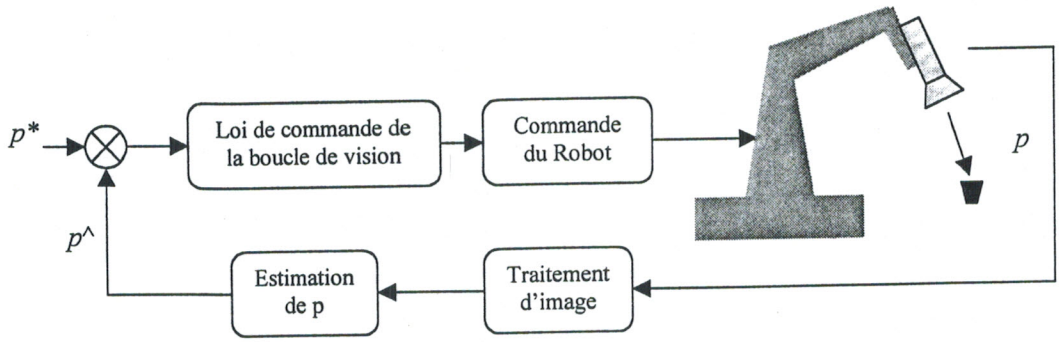



Figure1.6 : Structure d'un asservissement visuel 3D

### 1.2.4 Les asservissements 2D [6][11][26][32]

Dans un asservissement 2D, le signal de référence est exprimé sous la forme de primitives visuelles dans l'image. Une *primitive* est une forme géométrique élémentaire (point, segment de droite, portion d'ellipse,...). Elle sert à modéliser la projection d'un objet dans le plan image.

Le pionnier en matière d'asservissement 2D fut Weiss. Il étudia le lien entre le torseur cinématique du repère de la caméra et les vitesses de déplacement des primitives dans l'image. Cette relation sera baptisée Jacobien image par la suite dans la littérature. Son travail fut développé entre autres par Feddema, notamment du point de vue expérimental.

Espiau et al. proposèrent un cadre plus formel au concept de Jacobien image qu'ils baptisèrent matrice d'interaction Dans cet article, les auteurs étendent le concept de Jacobien image à différents types de primitives visuelles : droites, plans, cercles, sphères. Ils y décrivent également une simulation de suivi dans laquelle la caméra doit se déplacer à vitesse constante au-dessus de la voie de droite d'un modèle de route constitué de 3 droites parallèles.

La plupart des travaux qui traitent de l'asservissement 2D utilisent des primitives constituées de points. Ces points peuvent par exemple être situés à l'intersection de segments dans l'image ou encore être extraits du centre de gravité de la projection de disques. En fait, les cibles les plus couramment rencontrées sont simplement constituées de plusieurs disques coplanaires. De telles cibles ont l'avantage de nécessiter un temps de traitement d'image faible.

Un des principaux avantages de l'asservissement 2D est qu'il n'est pas nécessaire de connaître un modèle de la cible. Dans le cas où les primitives sont des points, si on dispose d'une estimation de la profondeur de la cible (c'est à dire son éloignement par rapport à la caméra suivant l'axe optique), les informations contenues dans l'image suffisent à déterminer le déplacement de la cible par rapport à la caméra. Différentes techniques peuvent être utilisées pour estimer cette profondeur. Lorsqu'il s'agit d'une expérience de suivi de cible, on fait souvent 2 approximations sur la profondeur : on considère que celle-ci ne varie pas avec le temps et on estime que tous les points de la cible ont la même profondeur (la taille de la cible est souvent négligeable par rapport à la distance entre la cible

et la caméra). Il est également possible de recourir à un algorithme adaptatif qui permet d'estimer en ligne la profondeur de la cible.

L'asservissement 2D est également adapté à des cibles complexes ne contenant aucune primitive simple. Une technique de flot optique (optical flow en anglais) est alors utilisée pour déterminer le mouvement de zones d'intérêt dans l'image. Dans ses grandes lignes, celle-ci consiste à minimiser une fonction de coût $\eth(\Delta x, \Delta y)$ :

$$\eth(\Delta x, \Delta y) = \sum_{x=x_0}^{x_0+dx} \sum_{y=y_0}^{y_0+dy} [W_t(x,y) - W_0(x+\Delta x, y+\Delta y)]^2 \tag{1.1}$$

où $W_t(x,y)$ représente la luminance du pixel de coordonnées $(x,y)$ dans une fenêtre rectangulaire $W_t$ de dimension $(dx,dy)$ englobant une région de l'image de la cible acquise à l'instant courant $t$. Le terme $W_0(x+\Delta x, y+\Delta y)$ représente la luminance du pixel de coordonnées $(x+\Delta x, y+\Delta y)$ dans une fenêtre de mêmes dimensions et position contenue dans l'image de référence. Ainsi, le minimum de $\eth(\Delta x, \Delta y)$ est obtenu pour des valeurs de $\Delta x$ et $\Delta y$ représentant le déplacement de la région considérée entre les deux images. On obtient donc une approximation du déplacement moyen des pixels de cette région de l'image.

**Remarque 1.1** *En toute rigueur, le flot optique consiste à calculer un champ dense de vitesses dans l'image ; il est cependant courant d'étendre cette notion à l'estimation du mouvement de zones d'intérêt dans l'image.*

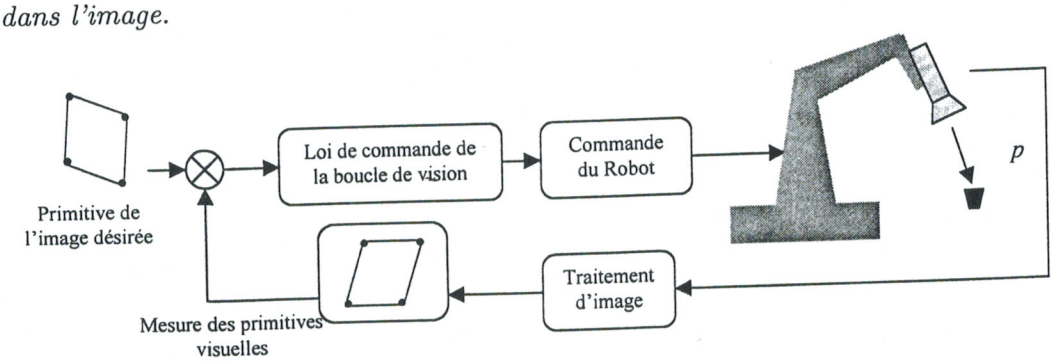



Figure1.7 : Structure d'un asservissement visuel 2D

### 1.2.5 Les asservissements 2D 1/2

Récemment, Chaumette et al. ont proposé une variante de la loi de commande habituellement utilisée, pour un asservissement dans l'image de la position d'une cible.

Les auteurs ont baptisé cet asservissement <<2D 1/2>> car il utilise une combinaison d'informations exprimées pour certaines d'entre elles dans l'image et pour d'autres dans le repère caméra. Avec cette technique, il est possible de contraindre la trajectoire d'un point de la cible à se projeter

suivant une droite dans l'image. L'objectif est un meilleur contrôle de la position de la cible dans l'image lors de régimes transitoires pour des écarts importants entre la position courante de la caméra et sa position de référence. (éviter entre autres qu'un des points ne sorte de l'image).

### 1.2.6 Les asservissements d2D/dt [12]

Dans ce type d'asservissement, la grandeur asservie est une vitesse relative entre la caméra et la cible. La référence est définie dans le plan image par un champ de vitesse des points. La mesure peut être établie par des techniques classiques de flot optique (cf. section précédente).

Une fonction quadratique est souvent choisie pour modéliser le champ de vitesse. La loi de commande est définie par rapport aux paramètres de cette fonction.

Ce type d'asservissement est à un stade émergeant. Actuellement, les traitements d'image qu'il nécessite sont encore très longs et fournissent des informations très bruitées. Des applications pratiques ont néanmoins été réalisées. Par exemple, un asservissement de l'alignement de l'axe optique d'une caméra animée d'un mouvement de translation suivant la direction de cette translation.

### 1.2.7 Les asservissements visuels indirects

Dans un asservissement visuel indirect, le contrôleur du robot fournit par le constructeur est utilisé pour réaliser l'asservissement des positions des coordonnées articulaires.

La plupart des applications d'asservissement par vision réalisées jusqu'à ce jour sont en fait des asservissements visuels indirects. Outre le fait qu'il est parfois matériellement difficile de modifier le contrôleur fournit par le constructeur, son intégration dans la boucle de vision permet de simplifier grandement la conception du système. En effet, ce dernier est souvent pourvu d'une interface de dialogue avec un système extérieur permettant d'avoir accès à toutes sortes de fonctions : contrôle de vitesse Cartésienne dans différents repères, gestions des singularités, gestion des sécurités...

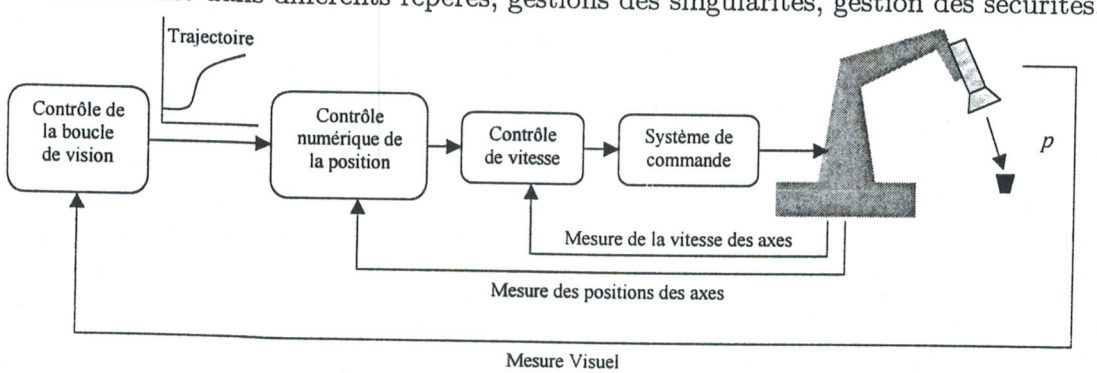



Figure1.8 : Structure d'un asservissement visuel indirect

Néanmoins, une telle architecture n'est absolument pas adaptée à un asservissement par vision visant à atteindre de bonnes performances dynamiques. En effet, lorsque la cadence d'échantillonnage

de l'information visuelle est suffisamment grande par rapport aux constantes de temps mécaniques du robot, des boucles internes de position ne peuvent que ralentir l'asservissement par vision par l'introduction de retards supplémentaires. Son rôle de stabilisation dans la boucle, valable lorsque la période de traitement d'image est grande, n'est plus du tout justifiée dans le cas d'un asservissement par vision rapide. Aussi, ce type d'asservissement correspond à des applications dont le but recherché est autre que la rapidité.

### 1.2.8 Les asservissements visuels directs [19]

Nous ne sommes actuellement qu'au début de l'émergence des asservissements visuels directs. En 1984, Weiss suggérait, au vu de résultats de simulations, que la période d'échantillonnage de la vision devrait être inférieure à 33ms afin de garantir la convergence du contrôleur adaptatif d'un asservissement visuel à 2 degrés de liberté.

Corke fut l'un des pionniers en matière d'asservissement visuel direct. Il proposa différentes stratégies de contrôle pour des systèmes simples à 2 ou 3 degrés de liberté. Il étudia par exemple l'influence d'une commande en couple ou d'une commande en vitesse sur la boucle de vision. En raison de l'utilisation d'une caméra CCIR standard, la fréquence des asservissements visuels qu'il réalisa était limitée à 50Hz. Néanmoins, une telle cadence est tout à fait compatible avec une structure d'asservissement visuel direct.

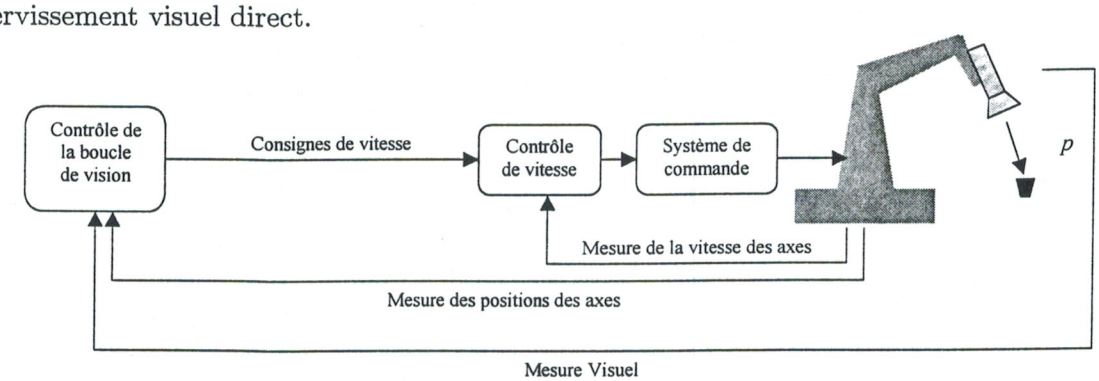



Figure1.9 : Structure d'un asservissement visuel direct

Les asservissements visuels directs devraient se généraliser dans un proche avenir. En effet, la croissance exponentielle de la puissance de calcul conjuguée avec l'apparition de systèmes de vision de plus en plus rapides pousse à adopter une telle structure de commande afin de mieux tirer parti des possibilités dynamiques offertes par le robot. Les auteurs utilisent un nouveau capteur d'image qui intègre dans le même composant un dispositif d'acquisition et de traitement rapide d'image. Ce capteur est ainsi capable de réaliser des traitements comme le calcul du flot optique sur une matrice de 32x32 pixels à la cadence de 1kHz.

Néanmoins, la résolution de ce capteur n'est pas encore compatible avec son utilisation dans

l'asservissement d'un robot suivant 5 ou 6 degrés de liberté.

### Prédiction du mouvement de la cible [6][17]

Dans un contexte de suivi de cible, le modèle dynamique de la cible permet d'établir une prédiction de sa position.

Cette prédiction peut être exploitée pour améliorer la performance du traitement d'image et pour augmenter la rapidité du suivi en utilisant un contrôleur prédictif, par exemple.

Paradoxalement, la nécessité de réaliser une prédiction du mouvement d'un objet qu'on désire suivre est née du manque de puissance pour le traitement d'image. En effet, dans une tâche de suivi, l'image est souvent découpée en zones d'intérêt. Il s'agit de petites imagettes rectangulaires dans l'image contenant les primitives de l'objet qui serviront au calcul de la loi de commande (que ce soit un asservissement 2D ou 3D). Ainsi, le traitement d'image est limité à ces imagettes qui, réunies, ne constituent qu'une fraction très faible de l'image entière. L'avantage est double :

1. Le transfert des données brutes de l'image est limité aux pixels contenus dans les imagettes.
2. Le traitement d'image, très gourmand en puissance de calcul, est limité aux imagettes.

La contrepartie à ce fractionnement de l'image est la nécessité de constamment réactualiser la position des imagettes en fonction du mouvement de l'objet, afin qu'elles soient toujours centrées par rapport aux primitives qu'elles englobent. La technique la plus simple consiste à réactualiser cette position a posteriori, en considérant que la vitesse de déplacement de la primitive dans l'image est suffisamment faible pour qu'elle ne sorte pas de l'imagette à l'acquisition suivante (Figure 1.10). Néanmoins, si la vitesse de la cible est trop importante ou si la fréquence d'acquisition de l'image est trop faible, une telle approche n'est plus valable. La solution consiste alors à réaliser une prédiction de la position future de la primitive dans l'image suivante et de placer l'imagette suivante autour de cette position.

La prédiction de la position future de la cible est souvent réalisée grâce à un filtre de Kalman. Ce dernier permet d'obtenir la meilleure prédiction au sens de la minimisation de la variance de l'erreur. Néanmoins, il requiert un observateur du mouvement de la cible. Pour réaliser cet observateur, la plupart des auteurs font l'approximation d'un mouvement uniforme de la cible. Il s'avère que la réalité du mouvement de la cible est souvent très différente d'un mouvement uniforme.

Ainsi, un changement brusque dans le mouvement de la cible provoque un régime transitoire du filtre qui conduit à une mauvaise estimation transitoire de la position de la cible. C'est pour éviter ce phénomène que Bensalah propose un filtre de Kalman modifié capable de s'adapter très rapidement aux <<ruptures de modèle>>.

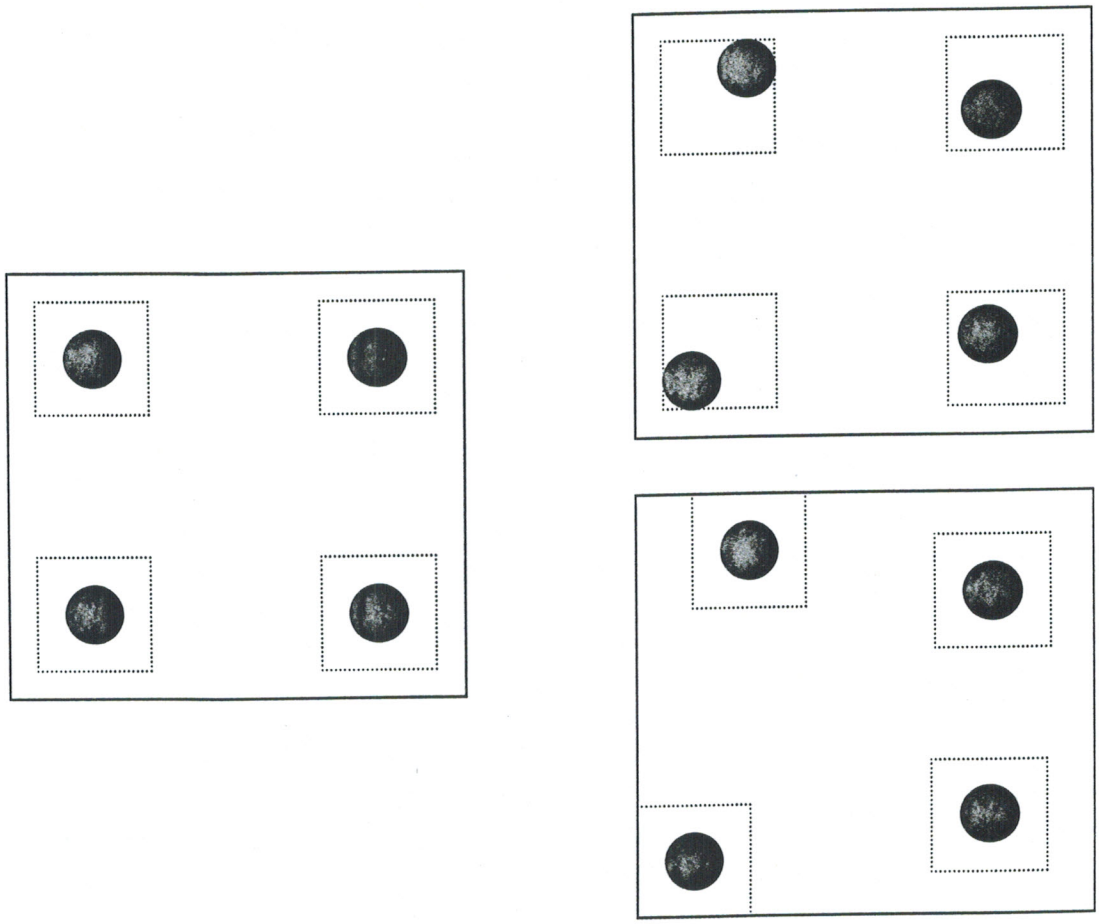


Figure 1.10 : Influence de la prédiction du mouvement de la cible sur la position des imagettes

La prédiction du mouvement futur de la cible peut également être exploitée dans un contexte d'asservissement prédictif. Dans ce cas, les signaux de référence futurs sont transmis à un correcteur prédictif qui réalise le contrôle de la boucle en fonction de l'évolution future de la trajectoire de la cible (voir [17], par exemple).

Dans le cas particulier d'un asservissement 2D, il a été montré par Bensalah que l'estimation du mouvement de la cible et sa prise en compte dans la loi de commande permet d'annuler les erreurs de traînage.

# Chapitre 2

# Modélisation de la caméra et du robot

## 2.1 Notions de base

Quelques notions fondamentales propres aux asservissements par vision sont décrites dans cette section. Elles sont communes à tous les asservissements de l'attitude de l'organe terminal du robot manipulateur à partir de mesures fournies par une caméra.

### 2.1.1 Définition de l'attitude d'un repère [4]

En robotique, il est souvent nécessaire de définir la position relative d'un corps par rapport à un autre : la position des différents axes du robot par rapport à sa base, la position de l'organe terminal par rapport à une pièce, ...etc. Lorsqu'il s'agit d'une position relative suivant 6 degrés de liberté, on parle alors d'attitude. Une attitude est définie mathématiquement par une translation et une rotation qui permettent de passer d'un repère à un autre. On définit l'attitude (pose en anglais) $p_{12}$ du repère direct $R_2$ par rapport au repère direct $R_1$ par une translation $T_{12}$ et une rotation $R_{12}$. Si $O_1$ est l'origine de $R_1$ et $O_2$ est l'origine de $R_2$, alors le vecteur de translation $T_{12}$ est défini par $\overrightarrow{O_1O_2}$ . Soient $x_1, y_1$ et $z_1$ les vecteurs unitaires du repère $R_1$ et $x_2, y_2$ et $z_2$ les vecteurs unitaires du repère $R_2$. La rotation $R_{12}$ est celle qui transforme $x_1$ en $x_2$, $y_1$ en $y_2$ et $z_1$ en $z_2$. La rotation $R_{12}$ peut être définie par une matrice :

$$R_{12} = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix} \tag{2.1}$$

où $\begin{bmatrix} r_{11} & r_{21} & r_{31} \end{bmatrix}^T$ sont les coordonnées de $x_2$ exprimées dans le repère $R_1$, $\begin{bmatrix} r_{12} & r_{22} & r_{32} \end{bmatrix}^T$ sont les coordonnées de $y_2$ exprimées dans le repère $R_1$, $\begin{bmatrix} r_{13} & r_{23} & r_{33} \end{bmatrix}^T$ sont les coordonnées de

$z_2$ exprimées dans le repère $R_1$. Si $P_{R_2} \in R^3$ est un vecteur contenant les coordonnées d'un point $P$ exprimées dans le repère $R_2$, alors, les coordonnées de ce point exprimées dans le repère $R_1$ sont données par :

$$P_{R_1} = R_{12}P_{R_2} + T_{12} \tag{2.2}$$

L'attitude peut aussi être définie sous forme matricielle. On utilise alors le formalisme des matrices homogènes. Celles-ci présentent l'intérêt de pouvoir composer plusieurs attitudes par simple multiplication d'une série de matrices. Ainsi, la matrice homogène de transformation $M_{12} \in R^{4\times4}$ entre le repère R1 et le repère R2 est définie comme suit :

$$M_{12} = \begin{pmatrix} R_{12} & T_{12} \\ 0 & 1 \end{pmatrix} \tag{2.3}$$

Ce formalisme impose l'utilisation de quaternions pour définir les coordonnées d'un point. Ainsi l'équation (2.2) est équivalente à :

$$\begin{pmatrix} P_{R_1} \\ 1 \end{pmatrix} = M_{12} \begin{pmatrix} P_{R_2} \\ 1 \end{pmatrix} \tag{2.4}$$

Soient $P_{R_3}$, les coordonnées de $P$ dans un repère $R_3$ et $M_{23}$, la matrice homogène de transformation entre $R_2$ et $R_3$. Les coordonnées de $P$ dans $R_1$ sont données par :

$$P_{R_1} = M_{12}M_{23}P_{R_3} \tag{2.5}$$

L'attitude peut également être définie de manière unique par un vecteur de 6 coordonnées. On parle alors de coordonnées opérationnelles. Parmi ces 6 coordonnées, 3 définissent la translation et 3 autres définissent la rotation. S'il n'y a pas d'ambiguïté pour la définition de la translation, il n'en est pas de même pour la rotation. Il existe en effet plusieurs conventions permettant de décomposer cette rotation en 3 rotations élémentaires. Celle qui est la plus largement utilisée consiste à décomposer la rotation en 3 rotations $\theta_t, \theta_r$ et $\theta_l$ respectivement autour des axes $x, y$ et $z$. On appelle aussi ces angles tangage, roulis et lacet. Dans l'annexe **A** nous reviendrons plus largement sur la définition de ces angles.

### 2.1.2 Définition des repères

On appelle $R_c$ le repère lié à la caméra. Par convention, $R_c$ est défini comme suit (voir figure 2.1) :

- Son origine est au centre optique de la caméra.
- L'axe $Z_c$ est confondu avec l'axe optique et est dirigé vers la scène.
- L'axe $X_c$ est vertical dans l'image et dirigé vers le haut.
- L'axe $Y_c$ est horizontal et dirigé tel que $R_c$ soit direct.

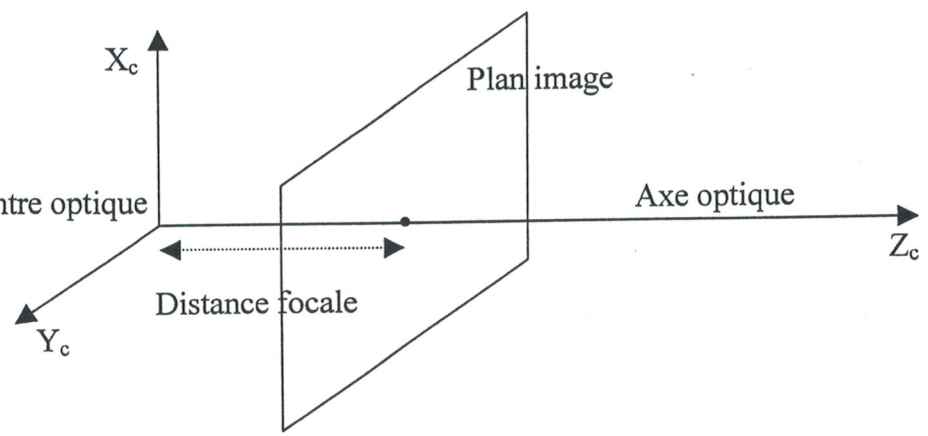



Figure 2.1 : Définition du repère caméra

On désigne par $R_o$ un repère lié à un objet situé dans l'espace de travail du robot. La matrice homogène $M_{co}$ définit l'attitude de $R_o$ par rapport à $R_c$.

Dans un asservissement par vision d'un robot, la grandeur à asservir est souvent l'attitude de $R_o$ (ou d'un autre repère lié à l'objet) par rapport à un repère $R_t$ lié à l'organe terminal. Lorsque la caméra est extérieure (figure 2.2), l'estimation des transformations $M_{ct}$ et $M_{co}$ permet de déterminer la position de l'objet par rapport à l'organe terminal.

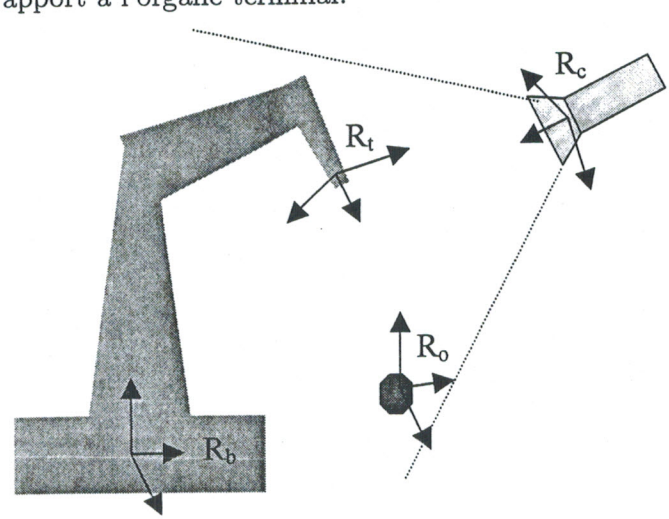



Figure 2.2 : Description de la configuration où la caméra est extérieure

Lorsque la caméra est attachée au robot, elle est généralement solidaire de l'organe terminal.

Dans ce cas, il est judicieux de choisir $R_c = R_t$ pour simplifier les calculs (cf. figure 2.3). La position de l'objet par rapport à l'organe terminal est donc directement obtenue par l'estimation de $M_{co}$.

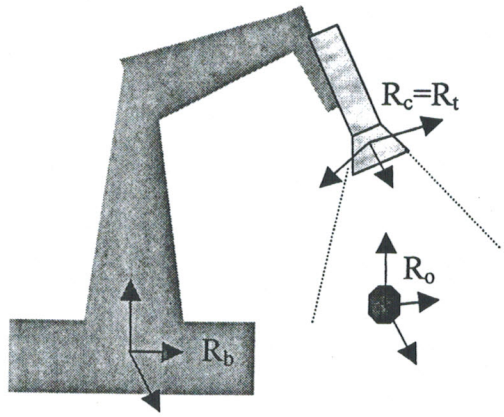



Figure 2.3 : Description de la configuration eye in hand

On a souvent besoin de définir la position de l'organe terminal par rapport à la base du robot. Pour cela, on utilise $R_b$, un repère fixe qui est solidaire de la base du robot.

Le tableau 2.1 récapitule la signification des différents indices utilisés pour les repères.

| Indice | Repère associé |
|---|---|
| c | Repère caméra |
| o | Repère lié à un objet |
| b | Repère lié à la base du robot |
| t | Repère lié à l'organe terminal du robot |

Tableau 2.1 : Convention pour les indices des repères

## 2.2 Modélisation de la caméra

Cette section présente la modélisation d'une caméra. Dans ce cadre, nous introduisons un modèle projectif dans l'image. Nous abordons ensuite la modélisation dynamique d'une caméra. Pour cela, certains aspects de la technologie de ces capteurs sont abordés.

### 2.2.1 Modèle de projection d'un point dans l'image

Nous supposons que l'objectif de la caméra est constitué d'une ou plusieurs lentilles qui peuvent se modéliser par une seule lentille mince convergente comme le décrit la figure 2.4.

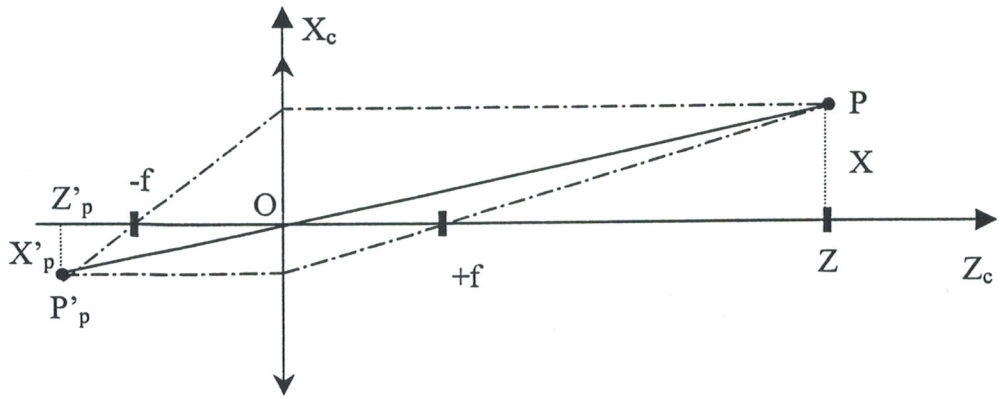



Figure 2.4 : Modèle d'une lentille mince

On définit un repère $R_c\,(x_c, y_c, z_c)$ associé à la lentille mince. Soit $P$ un point lumineux dont l'image à travers la lentille est $P'_p$. La coordonnée de $P$ suivant l'axe $z_c$ est $z$ et la coordonnée de $P'_p$ suivant ce même axe est $z'_p$. L'image de $P$ est réelle si $z > f$ où $f$ est la longueur focale de la lentille. En appliquant la relation fondamentale d'une lentille mince on obtient :

$$\frac{1}{z} - \frac{1}{z'_p} = \frac{1}{f} \tag{2.6}$$

Soient $x$ et $x'_p$ les coordonnées respectives de $P$ et $P'_p$ suivant l'axe $x_c$. Grâce au théorème de Thalès nous obtenons (voir figure 2.4) :

$$\frac{x}{x'_p} = \frac{z}{z'_p} \tag{2.7}$$

Or, d'après (2.6), on a :

$$\frac{z}{z'_p} = 1 - \frac{z}{f} \tag{2.8}$$

D'où :

$$x'_p = \frac{fx}{f - z} \tag{2.9}$$

Comme dans la plupart des cas $z \gg f$, on réalise l'approximation suivante :

$$x'_p \approx -\frac{f}{z}x \tag{2.10}$$

Cette relation définit une projection perspective de centre $O$ (le centre de la lentille) sur un plan perpendiculaire à l'axe optique $z_c$ situé à l'abscisse $-f$. Cette projection réalise une inversion de l'image. Par commodité, on transforme souvent l'image par une symétrie centrale de centre $O$ de manière à la redresser. Dans ce cas, la composition de la projection et de la symétrie centrale conduit

à :

$$x_p = -x'_p \approx \frac{fx}{z} \tag{2.11}$$

La composition de ces 2 transformations équivaut à projeter $P$ par une projection de centre $O$ sur un plan image virtuel situé à l'abscisse $f$ comme l'illustre la figure 2.5.

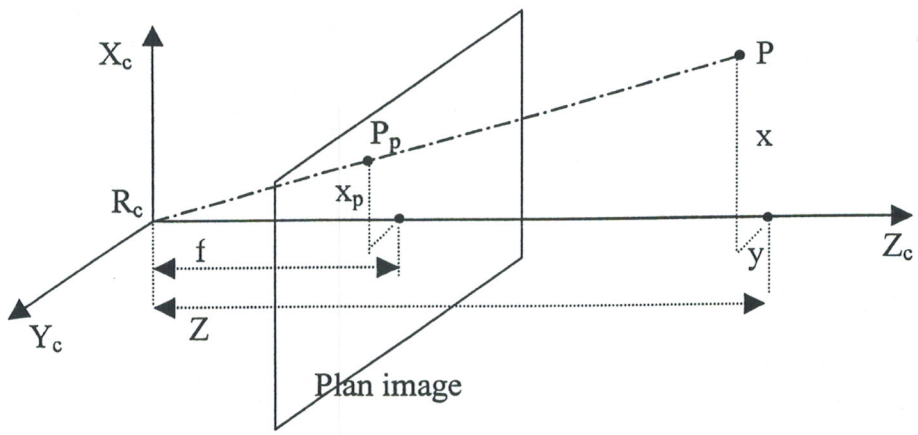



Figure 2.5 : Description de la projection perspective

En raisonnant de la même manière suivant $y_c$, on obtient :

$$y_p = -y'_p \approx \frac{fy}{z} \tag{2.12}$$

### 2.2.2 Le capteur visuel

La plupart des caméras modernes utilisent une matrice de capteurs rectangulaires élémentaires appelés pixels réalisés dans un matériau semi-conducteur. Le capteur le plus répandu est dénommé CCD (*Charge Coupled Device*). Nous ne rentrerons pas dans les détails technologiques. Pour plus d'informations sur le sujet, le lecteur peut se référer au livre de Corke[17].

Chaque pixel se comporte comme un intégrateur de la quantité de photons reçue. Les photons sont transformés en charges électriques et donc à l'issue du temps d'intégration, la charge totale contenue dans un pixel est sensiblement proportionnelle à la quantité de photons reçue. Ce temps d'intégration peut être réglé sur la plupart des caméras CCD. On parle alors d'obturateur électronique. La matrice CCD est placée au niveau du plan image réel de la caméra (situé à l'abscisse $-f$). Néanmoins, nous pouvons la modéliser par commodité comme étant placée à l'abscisse $f$. Par la connaissance de la luminance associée à chaque pixel, on obtient donc un échantillonnage spatial de l'image.

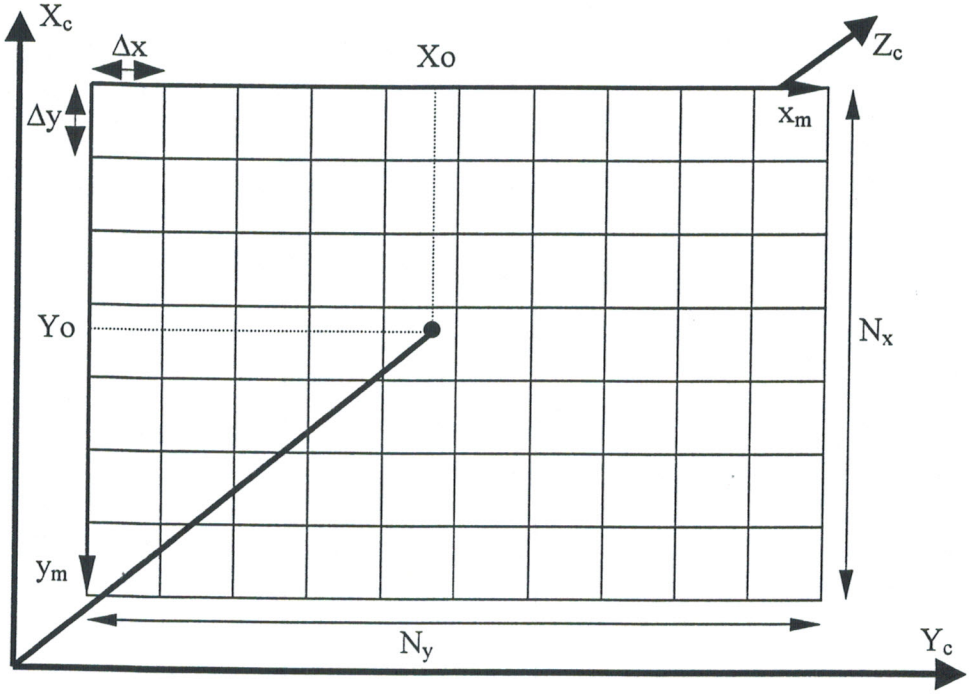



Figure 2.6 : Modélisation d'une matrice CCD

soient $N_x$ et $N_y$ respectivement le nombre de colonnes et de lignes de la matrice CCD (cf. figure 2.6) ;

$(N_x \times N_y)$ est appelé résolution du capteur CCD. Soient $\Delta_x$ et $\Delta_y$ les dimensions d'un pixel. On définit $G_x$ et $G_y$, respectivement les grandissements suivant les axes $x_c$ et $y_c$ du repère $R_c$ par :

$$G_x = \frac{f}{\Delta_x} \tag{2.13}$$

et

$$G_y = \frac{f}{\Delta_y} \tag{2.14}$$

Ainsi, les relations de projection perspective (2.11) et (2.12) peuvent être réécrites de la manière suivante :

$$X_p = G_x \frac{x}{z} \quad \text{et} \quad Y_p = G_y \frac{y}{f} \tag{2.15}$$

où $X_p$ et $Y_p$ sont les coordonnées de $P_p$ dans $R_c$ exprimées en pixels et non en mètres.

**Remarque 2.1** *La matrice CCD réalise un échantillonnage spatial de l'image. Donc, en toute rigueur, $X_p$ et $Y_p$ devraient être des valeurs discrètes. Néanmoins, l'information visuelle est rarement utilisée sous sa forme brute. Grâce à des traitements de l'image, on arrive à en extraire une information sub-pixel, donc en fraction de pixels. Les équations (2.15) se justifient donc dans ce contexte. Dans les relations (2.15), nous considérons que le repère associé aux coordonnées $X_p$ et $Y_p$ est $R_c$*

*(voir figure 2.5). Dans la pratique, l'axe optique $z_c$ coupe le capteur CCD approximativement en son centre. Néanmoins, en fonction des réglages de la lentille, sa position est susceptible de fluctuer légèrement (tout comme les grandissements $G_x$ et $G_y$). Nous appelons $X_0$ et $Y_0$ les coordonnées de ce point d'intersection ($O_p$) dans un repère $R_m$ $(x_m, y_m)$ solidaire du capteur (voir figure 2.6). Les grandeurs $G_x$, $G_y$, $X_0$ et $Y_0$ sont appelées paramètres intrinsèques de la caméra.*

### 2.2.3 Modélisation dynamique de la caméra

Avec une caméra CCD standard, le transfert des niveaux de luminance des pixels se fait séquentiellement et de manière analogique. Il en résulte un signal vidéo analogique continu dont la tension varie en fonction du temps suivant la valeur des niveaux de gris des pixels balayés. Le balayage de l'image s'effectue toujours de la gauche vers la droite et de haut en bas. Lorsque l'image est entrelacée, toutes les lignes paires, puis toutes les lignes impaires sont balayées alternativement. On définit une trame comme étant le signal résultant du balayage de toute l'image. Une image entrelacée est donc constituée d'une trame paire et d'une trame impaire. Une image non entrelacée ne comprend qu'une seule trame.

Le processus d'acquisition d'une image peut se décomposer en 3 étapes :

o Le temps d'intégration Ti au cours duquel les éléments sensibles intègrent les charges produites par les photons qui les atteignent.

o Le transfert des charges : les charges électriques contenues dans les éléments sensibles sont transférées dans un registre à décalage. A partir de cet instant, la mesure est terminée jusqu'à la prochaine trame.

o Les registres à décalage sont vidés séquentiellement afin de générer le signal vidéo.

Comme le montre la figure 2.7, le signal vidéo de la trame $n-1$ est émis pendant que la trame $n$ est acquise. Il y a donc un retard de la durée $Te$ d'une trame entre le moment où l'image commence à être mesurée et le moment où elle commence à être transférée. Comme l'émission du signal vidéo dure approximativement le temps d'une trame, il existe en fait un retard de $2Te$ entre le moment où l'image commence à être mesurée et le moment où son transfert et sa digitalisation au niveau du système de vision sont achevés.

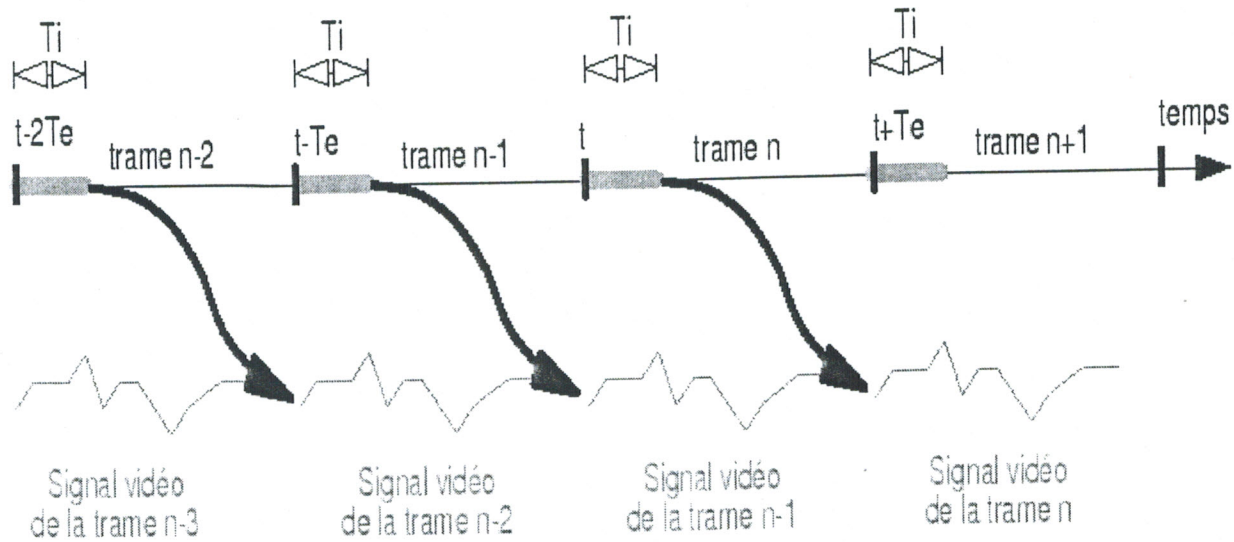



Figure 2.7 : Chronogramme du processus d'acquisition d'images d'une caméra CCD

La durée $Ti$ correspond à la période de l'obturateur électronique. Etant donné que l'image ne peut être mesurée instantanément, on peut considérer que <<l'instant moyen>> de mesure de la trame $n$ est $t + \frac{T_i}{2}$. Ainsi, si $T_i$ tend vers 0, le retard pur $T_r$ induit par le capteur de vision est de $2T_e$. Par contre, si $T_i = T_e$, ce qui est la configuration classique, $T_r = 1.5T_e$.

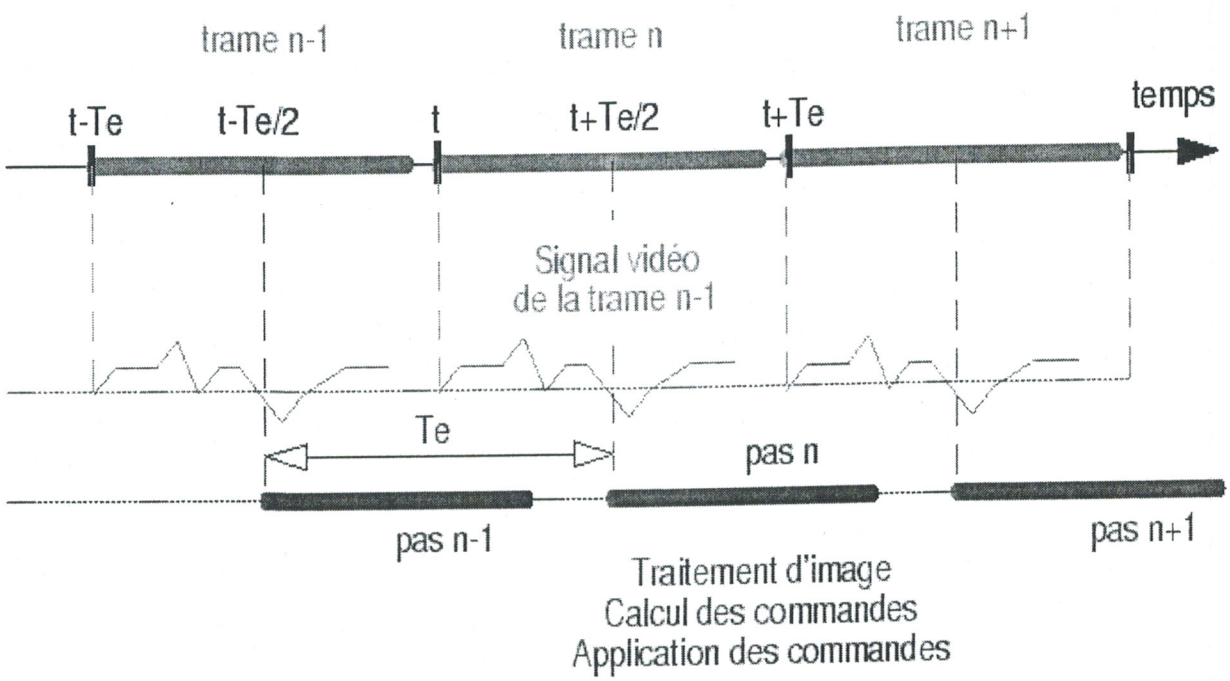



Figure 2.8 : Chronogramme de l'acquisition et du traitement d'images

Les système de vision modernes (i.e., le dispositif qui permet de numériser l'image) permettent de lire l'image en même temps qu'elle est acquise. De plus, il est possible de connaître exactement la ligne de l'image qui est couramment numérisée. Ainsi, dans un contexte d'asservissement par vision, il est possible de synchroniser une boucle de vision sur l'instant $t + \frac{T_e}{2}$ correspondant au milieu d'un trame (voir figure 2.8). Il est donc possible de réaliser un traitement de la moitié de l'image ayant

déjà été acquise tout en continuant, en parallèle, d'acquérir l'autre moitié de l'image. Avec une telle technique, si $T_i = T_e$, on a $T_r = T_e$. On en déduit que la fonction de transfert en $z$ d'un modèle simplifié du processus de vision est $z^{-1}$.

Il est clair que le processus de traitement d'image et de calcul des commandes introduit un second retard pur.

## 2.3 Modélisation d'un robot manipulateur [4] [5][10][20] [29][38]

Dans cette section nous décrivons la modélisation géométrique et cinématique d'un robot manipulateur à 5 degrés de liberté rotoïdes.

### 2.3.1 Modélisation géométrique direct d'un bras manipulateur a 5 degrés de liberté

Le modèle géométrique direct d'un bras manipulateur (MGD) permet d'exprimer la situation de l'effecteur (dans notre cas, une caméra embarquée) en fonction de la configuration du bras. Cette situation est définie par des coordonnées appelées classiquement coordonnées opérationelles, tandis que la configuration du bras est caractérisée par des coordonnées articulaires (généralisées).

Si $X_{camera}$ et $q_{bras} = [q_1 ... q_n]$ (pour un bras à $n$ degrés de liberté) désignant respectivement le vecteur des coordonnées opérationelles et articulaire, le MGD est une fonction (généralement non linéaire) du type $X_{camera} = f(q_{bras})$. Afin de préciser d'avantage cette notion de situation, nous introduisons deux repères orthonormés, l'un lié au bâti du bras manipulateur noté $R_0(O_0, \overrightarrow{X_0}, \overrightarrow{Y_0}, \overrightarrow{Z_0})$, l'autre lié à l'effecteur (ou caméra) noté $R_c(O_c, \overrightarrow{X_c}, \overrightarrow{Y_c}, \overrightarrow{Z_c})$ (cf. figure 2.9). Désormais, par situation de l'effecteur (ou de la caméra), nous entendrons la position et l'orientation de $R_c$ par rapport à $R_0$. En général, six paramètres suffisent pour la caractériser, trois en position et trois en orientation.

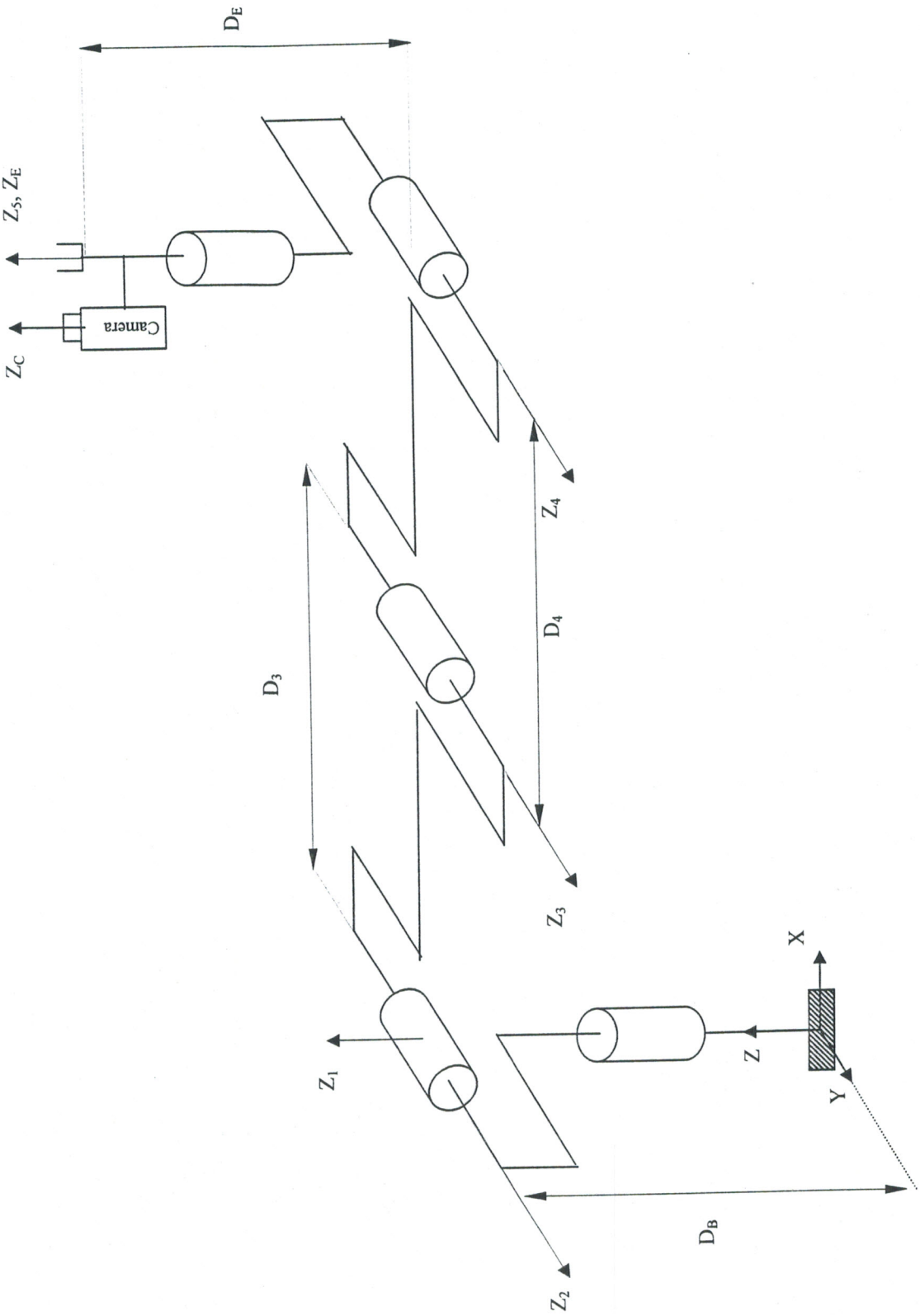



Figure 2.9 : Configuration de référence du bras manipulateur 5degrés de liberté

La détermination du MGD consiste donc à calculer la matrice de passage homogène entre les repères $R_0$ et $R_c$ noté ${}^{R_0}T_{R_c}$. Pour cela, nous appliquons la méthode développée dans [38]. Nous lions à chacun des corps du robot un repère $R_i(O_i, \overrightarrow{X_i}, \overrightarrow{Y_i}, \overrightarrow{Z_i})$ $(i = 1..5)$. La matrice de passage homogène ${}^{R_0}T_{R_c}$ est alors classiquement définie par :

$$ {}^{R_0}T_{R_c} = \prod_{0}^{4} {}^{R_i}T_{R_{i+1}} \; {}^{R_5}T_{R_c} \tag{2.16} $$

La matrice ${}^{R_5}T_{R_c}$ est obtenue par calibration hand-eye et s'exprime comme suit :

$$ {}^{R_5}T_{R_c} = \begin{pmatrix} 1 & 0 & 0 & a \\ 0 & 1 & 0 & b \\ 0 & 0 & 1 & c \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{2.17} $$

où les scalaires $a, b, c$ sont les coordonnées de l'origine du repère $R_c$ dans le repère $R_5$. Les matrices ${}^{R_i}T_{R_{i+1}}$ s'obtiennent quant à elles classiquement par la détermination des paramètres de Denavit-Hartenberg modifiés. Ces paramètres permettent de caractériser la situation du repère $R_i$ par rapport au repère $R_{i-1}$. Il sont définis de la manière suivante :

- $a_i$ : Distance entre $Z_{i-1}$ et $Z_i$ selon l'axe $X_{i-1}$.
- $\alpha_i$ : Angle algébrique entre $Z_{i-1}$ et $Z_i$ mesuré autour de $X_{i-1}$.
- $r_i$ : Distance entre $X_{i-1}$ et $X_i$ selon l'axe $Z_i$.
- $\theta_i = q_i$ : Angle algébrique entre $X_{i-1}$ et $X_i$ mesuré autour de $Z_i$.

La matrice de passage homogène entre $R_{i-1}$ et $R_i$ s'écrit alors :

$$ {}^{R_{i-1}}T_{R_i} = \begin{pmatrix} \cos\theta_i & -\sin\theta_i & 0 & a_i \\ \cos\alpha_i \sin\theta_i & \cos\alpha_i \cos\theta_i & -\sin\alpha_i & -r_i \sin\alpha_i \\ \sin\alpha_i \sin\theta_i & \sin\alpha_i \cos\theta_i & \cos\alpha_i & r_i \cos\alpha_i \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{2.18} $$

Le MGD du bras manipulateur défini par l'équation (2.16) peut alors être établi en calculant les paramètres de Denavit-Hartenberg pour chaque liaison et en déterminant les matrices de passage associées.

### 2.3.2 Modélisation géométrique inverse du bras manipulateur a 5 degrés de liberté

Le problème inverse consiste à calculer les coordonnées articulaires correspondant à une situation donnée de l'organe terminal. Nous nous intéressons dans ce mémoire aux robot à structure ouverte

simple possédant moins de 6 degrés de liberté.

pour une telle structure, le robot ne peut donner à son organe terminal n'importe quelles positions et oriontations. le but de manipuler un bras manipulateur est de ramener le repère terminal $R_E$ dans un repère désiré $R_E^d$, l'insuffisance de degrés de libèrté est compensé par l'imposition de certains éléments de ${}^0T_E^d$, sinon, on est amené à réduire le nombre d'équations en ne considérant que certains éléments géométriques liés aux repères $R_E$ et $R_E^d$, au lieu de réaliser des liaisons du type repère sur repère, on cherchera à satisfaire des liaisons du type point sur point, droite sur droite, ou point-droite sur point-droite. nous etudierons, dans ce qui suit, le dernier type (point-droite sur point-droite).

on définit l'outil par une droite suivant un des vecteurs unitaires du repère effecteur (notons $a_E$) et passant par le point origine du repère effecteur, c'est-à-dire que les éléments de $a_E$ et $P_E$ sont connus.

La tâche consiste à confondre l'origine du repère effecteur et aligné l'axe $a_E$ repectivement avec un point et une droite de l'environnement de coordonnées ${}^0P_E^d$ et ${}^0a_E^d$.

Le système à résoudre est : $\left[ \begin{array}{cc} {}^0a_E^d & {}^0P_E^d \end{array} \right] = \left[ \begin{array}{cc} a_E & P_E \end{array} \right]_{R_0}$

### 2.3.3 Modèle cinématique direct du bras manipulateur

Le modèle cinématique direct (MCD) permet d'établir l'éxpression des vitesses opérationnelles en fonction des vitesses généralisées. Il est de la forme $\dot{X}_{camera} = J(q_{bras}).\ \dot{q}_{bras}$. Il permet donc d'exprimer le torseur cinématique de la caméra par rapport au repère de base $R_0$. Notons que le calcule de ce modèle se ramène à celui de la matrice jacobienne $J$.

Il existe deux méthodes de calcul du MCD : soit on dérive le modèle géométrique direct du robot défini par la relation 2.16, soit on exprime le torseur cinématique de la caméra en fonction de la dérivée des coordonnées articulaires. Nous avons employé cette dernière technique, puisqu'elle s'avère généralement la plus simple dans le cas d'un robot à cinq degrés de libèrté. La méthodologie adoptée étant largement détaillée dans [38], nous nous contenterons de la rappeler trés brièvement ici. Elle consiste à exprimer d'abord le torseur cinématique du poignet caractérisé par le repère caméra $R_c$ en fonction des vitesses articulaires, il s'exprime de la manière suivante dans le cas d'un robot manipulateur comportant $n$ liaisons rotoïdes :

$$J_{poignet} = \begin{pmatrix} z_1 \wedge O_1O_n & \cdots & z_n \wedge O_nO_n \\ z_1 & \cdots & z_n \end{pmatrix} \tag{2.19}$$

Les calculs concernant le bras complet sont détaillés dans [38].

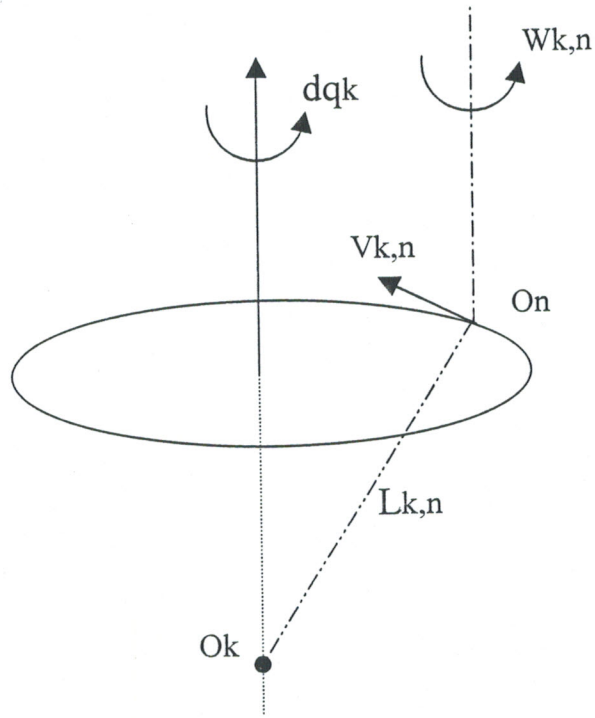



Figure 2.10 : Représentation des differents vecteurs vitesse pour un axe

### 2.3.4 Modélisation du robot ROB3i IR50p 5 degrés de liberté

Nous exposons dans cette section la modélisation géométrique et cinématique du manipulateur que nous avons utilisé lors de nos expériences : le ROB3i IR50p. Pour cela, nous reprenons les principes exposés dans la section précédente que nous appliquons au cas particulier de ce robot.

#### Présentation générale

Le robot ROB3i IR50p est un bras manipulateur à 5 degrés de liberté. Il possède 5 axes en rotation et un préhenseur pour la préhension des objets. La figure 2.11 en donne un aperçu.

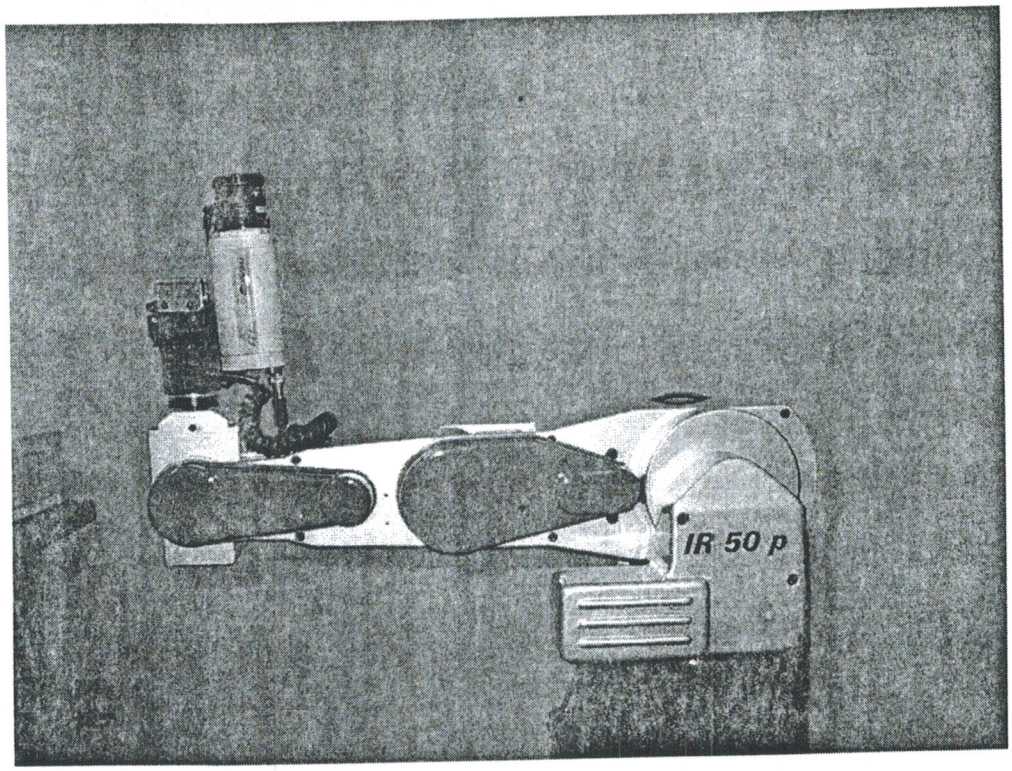



Figure 2.11 : Le robot ROB3i IR50p

Chacun de ces axes est mû par des servo moteur DC. La position absolue de chaque axe est détèrminée par un capteur potentiométrique rotatif de position ce qui veut dire que la position en cours est connue à tout moment. Nous présentons dans l'annexe **B** les caratéristiques du ROB3i IR50p.

### Modélisation géométrique

Le tableau 2.2 donne les valeurs des paramètres de Denavit-Hartenberg modifiés du modèle géométrique du robot ROB3i IR50p suivant la figure 2.9. Les valeurs minimales et maximales de la rotation de chaque axe sont précisées.

| i | $a_i$ | $\alpha_i$ | $r_i$ | $\theta_i$ | $\min(\theta_i)$ | $\max(\theta_i)$ |
|---|---|---|---|---|---|---|
| 1 | 0 | 0 | $D_b$ | $\theta_1$ | $-110°,4$ | $109°,6$ |
| 2 | 0 | $\frac{\pi}{2}$ | 0 | $\theta_2$ | $-8°$ | $188°$ |
| 3 | $D_3$ | 0 | 0 | $\theta_3$ | $-98°$ | $98°$ |
| 4 | $D_4$ | 0 | 0 | $\theta_4$ | $-97°$ | $97°$ |
| 5 | 0 | $-\frac{\pi}{2}$ | 0 | $\theta_5$ | $-194°,2$ | $195°,8$ |

Tableau 2.2 : Convention pour les indices des repères

Les matrices de transformation intère-repère ${}^{R_{i-1}}T_{R_i}$ pour $i = 1 \cdots 5$ seront représentées dans l'annexe **C** suivant le modéle schématisé par la figure 2.9. Ainsi que le modèle géométrique inverse.

### Modélisation cinématique

Nous ne présentons donc, dans l'annexe **C**, que les résultats, Jacobien caméra-base ${}^{0}J_c$, obtenus dans le cas du manipulateur représenté sur la figure 2.9.

### Identification du modèle dynamique des axes asservis en position

Nous faisons l'hypothèse que les mouvements des differents axes sont découplés. Chaque articulation est modélisé par une fonction de transfert en boucle fermée de la forme :

$$F(s) = \frac{w_n^2}{(\alpha s + 1)(s^2 + 2\xi w_n s + w_n^2)} \tag{2.20}$$

pour une entrée échelon $\frac{1}{s}$, la sortie est de la forme :

$$y(t) = 1 + \frac{b}{\alpha} e^{-\frac{1}{\alpha}t} + c\left[e^{-\xi w_n t}\cos\left(w_n\sqrt{1-\xi^2}t\right) - \frac{\xi w_n - \frac{d}{c}}{w_n\sqrt{1-\xi^2}} e^{-\xi w_n t}\sin\left(w_n\sqrt{1-\xi^2}t\right)\right] \tag{2.21}$$

avec

$$\begin{aligned} b &= \frac{\alpha^3 w_n^2}{2\alpha\xi w_n - \alpha^2 w_n^2 - 1} \\ c &= -\frac{b+\alpha}{\alpha} \\ d &= -2\xi w_n - \alpha w_n^2 - b w_n^2 \end{aligned}$$

l'optimisation par la méthode des moindres quarrés (sous Matlab *leastsq* ou *lsqnonlin*) a donnée des résultats satisfaisants, les résultats pour chaque axe avec leurs reponses indicielles sont données au annexe **C**.

# Chapitre 3

# Modélisation de l'asservissement visuel

## 3.1 Introduction [2][7] [10][13][15] [19][24][45] [52]

Dans ce contexte, nous allons généraliser le calcul des matrices d'interaction à des primitives géométriques de type point. Nous décrivons, aussi, un cadre pour la conception et l'éxécution des tâches de positionnement en utilisant la vision, basé sur un formalisme emprunté à la théorie des mécanismes. Nous introduisons le concept de "liaisons virtuelles" comme un moyen de décrire les tâches référencées capteurs. En fait, les tâches sont exprimées comme des ensembles de contraintes sur le mouvement d'un repère attaché à la caméra. D'autres études ont été consacrées à l'utilisation de la commande référencée vision pour les robots mobiles, ainsi qu'à la détection des singularités de la matrice d'intéraction dans le cas de trois points comme primitives. D'autres ont menés des recherches sur la robustesse de l'asservissement aux erreurs de calibration et sur le contrôle du zoom d'une caméra. Parallèlement, F. Chaumette a poursuivi ses recherches dans le domaine de l'asservissement visuel en traitant le cas de la poursuite d'un objet mobile. Le mouvement de l'objet est compensé en intégrant dans la loi de commande, soit un terme intégrateur ( ce qui permet de ne suivre correctement que des objets a vitesse constante), soit une estimation de ce mouvement basée sur un filtrage de Kalman (pour des mouvements avec changements brusques de direction ou d'amplitude par exemple).

Nous avons cité deux approches d'asservissement visuel, 3D basée sur la position d'objet et 2D basée sur des primitives objet dans l'image (coordonnées des points caractérisqtiques). Avec une approche 3D, la position et l'orientation de l'objet relative a la caméra sont extraites, en utilisant la photogrammetrie, stéreo ou la profondeur technique. Elles seront considérées comme une sortie du svstème de traitement d'image, afin de controler le manipulateur. Pour celà, un modèle géométrique de l'objet est requis ainsi qu'une bonne calibration du système Robot-caméra. Leurs precisions sont critiquees aux performances du système. Par contre, l'approche 2D "referencé image" utilise des

primitives extraites de l'image acquise directement de la boucle d'asservissement du système, ainsi, les primitives sont controlées directement dans le plan image avec une commande judicieuse du manipulateur qui reagit sur ces dernières. Par conséquent ; le taux de calcul est réduit, et les erreurs de calibration du modèle géometrique de l'objet et du Robot-caméra peuvent être éléminées.Une etude comparative est faite par Corke et Hutchinson[19].

Dans ce chapitre, les problèmes d'asservissement visuel 2D sont traités.Nous commençons avec un bref aperçu sur l'approche fonction de tache. Aprés une définition de la matrice d'intéraction (Jacobien Image), nous abordons l'approche classique 2D, traitons les problèmes pour ce qui concerne les primitives rodandantes, la singularité du jacobien image, stabilité, et la loi de commande linéarisée. Nous terminons ce chapitre avec une approche pour les systèmes non calibrés avec une estimation à priori.

## 3.2 L'approche fonction de tâche [12] [16][28][33] [41][42]

L'approche fonction de tâche a été développée par Samson pour la commande des robots manipulateurs. Cette approche a servi de base pour développer la commande référencée capteurs, et de nombreux chercheurs l'ont appliquée au cas particulier du capteur visuel.

Samson a montré qu'une tâche robotique peut généralement être exprimée comme la régulation, sur un certain horizon temporel $[0, T]$, d'une fonction $e(q,t)$ (où $q$ est le vecteur des configurations articulaires du robot) de classe $C^2$ et de dimension $m$, appelée *fonction de tâche.* Des fonctions de tâches classiques sont :

- $e(q,t) = q - q^*(t)$ où $q^*(t)$ est une trajectoire désirée dans l'espace articulaire.
- $e(q,t) = r(q) - r^*(t)$ où $r^*(t)$ est une trajectoire désirée dans l'espace cartésien.
- $e(q,t) = \bar{e}(\xi(q), \xi^*(t))$ où $\xi^*(t)$ est une trajectoire désirée dans l'espace du capteur.

On s'intéresse particulièrement au dernier cas où les informations capteur $\xi(q,t)$ utilisées pour appréhender la valeur de la fonction de tâche à l'instant $t$ sont issues d'une caméra.

Le problème de la régulation de la fonction de tâche est bien posé si elle possède certaines propriétés. L'une d'elles, qui signifie que la tâche est réalisable, est l'existance et l'unicité d'une trajectoire idéale de classe $C^1$ (dans le cas où le robot est commandé en couple, la trajectoire doit être de classe $C^2$), notée $q_r(t)$, telle que $e(q_r(t),t) = 0, \forall t \in [0,T]$ et $q_r(0) = q_0$, où $q_0$ est une condition initiale donnée. Une autre condition, extrèmement importante, est la régularité du jacobien de la tâche, $\frac{\partial e}{\partial q}(q,t)$, autour de $q_r(t)$. En effet, ce jacobien lie l'espace différentiel articulaire à l'espace différentiel de la fonction de tâche :

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

$$\ddot{e}(q,t) = \frac{\partial e}{\partial q}(q,t)\ \ddot{q} + h(q,\dot{q},t)$$

avec

$$h(q,\dot{q},t) = \begin{bmatrix} \vdots \\ \dot{q}^T H_i(q,t)\dot{q} \\ \vdots \end{bmatrix} + 2\frac{\partial^2 e}{\partial q \partial t}(q,t)\dot{q} + \frac{\partial^2 e}{\partial^2 t}(q,t)$$

où $H_i(i = 1, ..., n)$ est la i-ème matrice hessienne de $e$ par rapport à $q$.

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

Une autre condition importante à respecter, est que les informations visuelles nécessaires à la réalisation de la tâche soient toujours présentes dans l'image (condition de *visibilité* de la tâche). Soient $\xi = \{\xi_1, \xi_2, \cdots\}$ l'ensemble des informations visuelles élémentaires $\xi_i$ théoriquement disponibles, $\xi_{\min}$ le nombre minimal d'informations visuelles nécessaires à la réalisation de la fonction de tâche et $\xi_{obs} = \{\xi_i : \xi_i \in \xi, \xi_i \in I\}$ l'ensemble des informations visuelles réellement observées ($I$ étant le domaine de définition des informations visuelles, c'est-à-dire l'image). La condition de visibilité peut alors s'écrire comme :

$$\dim(\xi_{obs}) \geq \xi_{\min}, \forall t$$

Cette condition n'est généralement pas prise en compte dans la commande. Ceci implique que, suivant le type de fonction de tâche choisi, on aura une probabilité plus au moins grande que les informations visuelles restent dans le champ de vision de la caméra.

La reconstruction de la fonction de tâche peut être comme suit :

♦ $e(q) = r(\xi(q)) - r(\xi^*(t))$. La position $r$ de l'effecteur du robot est estimée à partir des informations visuelles. Ceci est valable pour l'approche 3D.

♦ $e(q,t) = C(\xi(q) - \xi^*(t))$ où C est une matrice qui permet de tenir compte d'une éventuelle rodondance d'information. La fonction de tâche est construite à partir de la différence entre les informations visuelles courantes et désirées dans l'image. ceci est propre à l'approche 2D que nous détaillons ci-aprés.

Une des difficultés majeures pour la reconstruction de la fonction de tâche est la définition de la trajectoire désirée $\xi^*(t)$. En effet, il est nécessaire de construire la fonction de tâche de telle manière qu'elle soit nulle si les informations visuelles coïncident avec celle désirées ($e(q,t) = 0$ si $\xi(q(t)) = \xi^*(t)$). Une méthode pour définir les informations visuelles désirées est basée sur leur modélisation en utilisant des connaissances parfaites de la géométrie du système de vision pour que $\xi^*(t)$ puisse physiquement exister. Une technique pour éviter cette phase de modélisation consiste à

effectuer un apprentissage expérimental de l'image de réference. Dans ce cas, on a alors $\xi^*(t) = \xi^*$, ce qui permet d'éviter la difficulté de générer une trajectoire $\xi^*(t)$. Considérons, par exemple, la tâche classique du positionnement par rapport à un objet d'une caméra embarquée sur un robot (cette configuration reste utile pour toute notre étude). Aprés avoir appris l'image correspondant à la position désirée de la caméra, et aprés avoir déplacé la cible et/ou le robot, on peut définir la fonction de tâche sur la base d'informations obtenues à partir des deux vues de la cible. Une fonction de tâche nulle implique que la caméra a rejoint sa position désirée avec une précision qui est indépendante des erreurs de calibration du système de vision. Il est clair que les informations visuelles puissent varier dans le temps, $\frac{\partial \xi^*}{\partial t} \neq 0$. Pour simplifier le problème, on supposera par la suite que cette dernière est nulle (position de la cible désirée est fixe).

Si on peut assurer qu'il existe $q^*$ tel que $e(q^*) = C(\xi(q^*) - \xi^*) = 0$, on peut donc écrire le développement de Taylor de la fonction de tâche $e(q) = C(\xi(q) - \xi^*) = \bar{e}(\xi(q))$ de la manière suivante :

$$\begin{aligned} \bar{e}(\xi) &= \bar{e}(\xi^*) + \left.\frac{\partial \bar{e}}{\partial \xi}\right|_{\xi=\xi^*} (\xi - \xi^*) + \frac{1}{2}(\xi - \xi^*)^T \left.\frac{\partial^2 \bar{e}}{\partial \xi^2}\right|_{\xi=\xi^*} (\xi - \xi^*) + \eth^2(\xi)(\xi - \xi^*) \\ &= \left[\left.\frac{\partial \bar{e}}{\partial \xi}\right|_{\xi=\xi^*} + \frac{1}{2}(\xi - \xi^*)^T \left.\frac{\partial^2 \bar{e}}{\partial \xi^2}\right|_{\xi=\xi^*} + \eth^2(\xi)\right](\xi - \xi^*) \\ &= C(\xi, \xi^*, A, g)(\xi - \xi^*) \end{aligned} \tag{3.1}$$

où $C$ est une matrice qui dépend des informations visuelles courantes et désirées, de la matrice $A$ contenant les paramètres intrinsèques de la caméra, et d'un vecteur $g$ contenant les paramètres géométriques relatifs à la scène observée.

Un des grands problèmes pour l'analyse de la stabilité des systèmes d'asservissement visuel est la difficulté de modéliser ou mème d'obtenir une forme analytique de la matrice $C$ en asservissement visuel 3D. Le problème peut être résolut en asservissement visuel 2D.

Une fois que la fonction de tâche définie, une loi de commande doit être conçue afin de faire tendre cette fonction à zéro. Cette loi de commande doit présenter des caractéristiques suffisantes de robustesse afin d'assurer la convergence de l'asservissement visuel en présence d'erreurs de mesure et de calibration de la géométrie du système.

## 3.3 Jacobien Image [1][28][37][43][48]

Le jacobien robot qui fait le lien entre les vitesses articulaires et les vitesses cartésiennes de l'effecteur, joue un rôle important dans la commande des robots manipulateurs dans l'espace cartésien. Il est identique pour les systèmes d'asservissement visuel, où on a besoin d'un jecobien qui trans-

forme les vitesses articulaires à des vitesses des primituves dans l'espace image, dite *jacobien image* (d'autres l'appelent matrice d'interaction).

### 3.3.1 Définition

Soit $\xi$ le vecteur des primitives visuelles (par exemple, les coordonnées d'un point ou les paramètres dans l'image d'une droite), $q$ le vecteur des positions articulaires, $p$ le vecteur des coordonnées généralisées représentant la position de l'objet, et $\theta^*$ le vecteur des vitesses de l'objet de dimension $l$ tel que $\dot{p} = W(p).\theta^*$, $W$ est une matrice fonction de dimension $m_0 \times l$. On a :

$$\dot{\xi} = J.\dot{q} + L_o.\dot{p} \tag{3.2}$$

où

$$J = \frac{\partial \xi}{\partial r}\frac{\partial r}{\partial q} \quad \text{et} \quad L_o = \frac{\partial \xi}{\partial r}\frac{\partial r}{\partial p} \tag{3.3}$$

avec $r = [X\ Y\ Z\ \alpha\ \beta\ \gamma]$ est le vecteur des coordonnées d'objet dans le repère caméra, soit $s_{cam}(6\times 1)$ (le vecteur de position caméra) et $s_{obj}(6 \times 1)$ (vecteur de position objet) dans le repère de base $R_b$, donc on a

$$r = {}^cT_b(s_{obj} - s_{cam}) \tag{3.4}$$

avec ${}^cT_b$ est la matrice de transformation homogène entre le repère de base et le repère caméra.

Les matrice $J(2n \times m)$ ($n$ primitives visuelles et $m$ articulations) et $L(2n \times m_0)$ ($m_0$ degrés de liberté de l'objet) sont appelées *Jacobien Image* et *Jacobien de mouvement d'objet.*

$${}^cJ_{rob} = \frac{\partial r}{\partial q} = {}^cT_w\frac{\partial s_{cam}}{\partial q} \tag{3.5}$$

est le jacobien du robot exprimé dans le repère caméra. Le jacobien, qui exprime les petites variations des primitives dans l'espace image par rapport au mouvement d'effecteur dans le repère caméra, est défini par

$$J_{img} = \frac{\partial \xi}{\partial r} = \begin{bmatrix} J^1_{img} \\ \vdots \\ J^n_{img} \end{bmatrix} \tag{3.6}$$

où

$$J^i_{img} = \begin{bmatrix} -\frac{f}{Z_i} & 0 & \frac{x_i}{Z_i} & \frac{x_iy_i}{f} & -\frac{x_i^2+f^2}{f} & y_i \\ 0 & -\frac{f}{Z_i} & \frac{y_i}{Z_i} & \frac{y_i^2+f^2}{f} & -\frac{x_iy_i}{f} & -x_i \end{bmatrix} \tag{3.7}$$

**Preuve.** Soit un point $P$ de coordonnées $[X\ Y\ Z]^T_{R_c}$ dans le repère caméra se projettant dans le

plan image sous la forme d'un point $P_p$ de coordonnées $[x\ y\ 1]^T$ avec (voir figure 3.1) :

$$P_p = \frac{f}{Z}P \tag{3.8}$$

où $f$ est la distance focale.

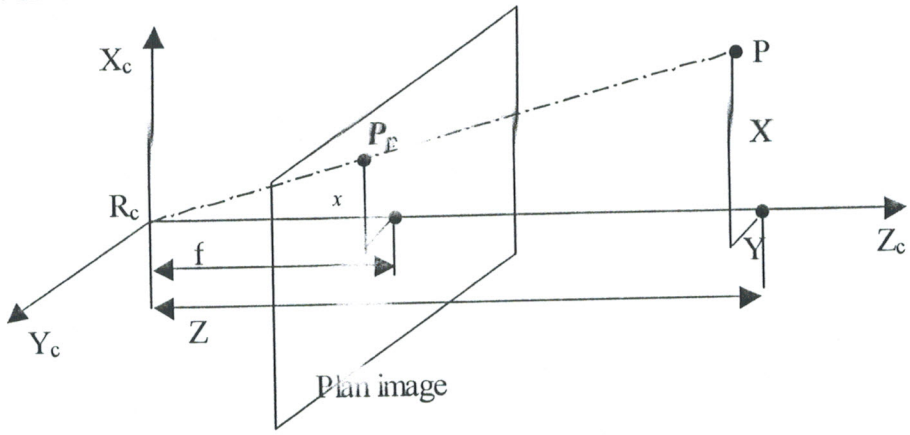



Figure 3.1 : Projection perspective d'un point

La dérivée par rapport au temps de (3.8) donne :

$$\dot{P}_p = -f\frac{\dot{Z}}{Z^2}P + \frac{f}{Z}\dot{P}$$

Sachant que l'équation fondamentale de la cinématique pour un objet immobile et une caméra mobile est :

$$\dot{P} = -V - \Omega\Lambda P$$

avec $V$ et $\Omega$ sont les composantes des vitesses en translation et en rotation de la caméra (effecteur). Nous obtenons :

$$\dot{P} = -\begin{pmatrix} V_x \\ V_y \\ V_z \end{pmatrix} - \begin{pmatrix} \Omega_x \\ \Omega_y \\ \Omega_z \end{pmatrix} \Lambda \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} -V_x + Y.\Omega_z - Z.\Omega_y \\ -V_y + Z.\Omega_x - X.\Omega_z \\ -V_z + X.\Omega_y - Y.\Omega_x \end{pmatrix}$$

Nous en déduisons

$$\dot{P}_p = f\frac{V_z - X.\Omega_y + Y.\Omega_x}{Z^2}\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} + \frac{f}{Z}\begin{pmatrix} -V_x + Y.\Omega_z - Z.\Omega_y \\ -V_y + Z.\Omega_x - X.\Omega_z \\ -V_z + X.\Omega_y - Y.\Omega_x \end{pmatrix}$$

$$= \begin{pmatrix} -f\frac{1}{Z}V_x + f\frac{X}{Z^2}V_z + f\frac{XY}{Z^2}\Omega_x - f\left(1+\frac{X^2}{Z^2}\right)\Omega_y + f\frac{Y}{Z}\Omega_z \\ -f\frac{1}{Z}V_y + f\frac{Y}{Z^2}V_z + f\left(1+\frac{Y^2}{Z^2}\right)\Omega_x - f\frac{XY}{Z^2}\Omega_y - f\frac{X}{Z}\Omega_z \\ 0 \end{pmatrix}$$

Ce qui peut s'écrire, en remplaçant les termes $f\frac{X}{Z}$ et $f\frac{Y}{Z}$ par $x$ et $y$, sous la forme

$$\dot{P}_p = \begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} = \begin{pmatrix} -\frac{f}{Z} & 0 & \frac{x}{Z} & \frac{xy}{f} & -\frac{x^2+f^2}{f} & y \\ 0 & -\frac{f}{Z} & \frac{y}{Z} & \frac{y^2+f^2}{f} & -\frac{xy}{f} & -x \end{pmatrix} \begin{pmatrix} V_x \\ V_y \\ V_z \\ \Omega_x \\ \Omega_y \\ \Omega_z \end{pmatrix}$$

■

### 3.3.2 Primitives dégénérées

Soit une primitive point qui se repose sur l'axe optique du caméra, le mouvement du point sur cet axe ne provoque aucun changement sur l'image, ce qui explique que la position de la caméra suivant l'axe optique est non-Contrôlable par ce point. D'autre part, les rotations de la caméra dans n'importe quelle direction autour de l'objet contrôlé, ne provoque aucun changement dans l'image, donc, cette primitive ne peut controler l'orientation de la caméra. On dit alors que la primitive est dégénérée pour contrôler les six degrés de liberté du caméra.

En général, les primitives qui ne changent pas par un mouvement du robot ou de l'objet lui même, sont dites *primitives dégénérées*. Un simple test pour celà est de vérifier le rang du jacobien. S'il existe des directions ou des mouvements dans l'espace articulaire $\tilde{\theta}$ qui ne génèrent aucun changement des primitives, alors l'équation suivante est vérifiée

$$J\ \tilde{\theta} = 0$$

Le jacobien n'est donc pas de rang complet. S'il l'est, on a alors

$$J\ \delta\theta \neq 0$$

pour tous $\delta\theta \in R^m$, donc les primitives changent pour toutes les directions de mouvement dans l'espace articulaire. Il est similaire pour le movement de l'objet (Jacobien de mouvement d'objet).

Par conséquent, pour éviter les primitives dégénérées, on suppose que pour tout $q \in \vartheta$ et $p \in \psi$, les jacobiens $J$ et $L$ sont de rang plein par les colonnes. $rank\ J = m$ et $rank\ L = m_0$, $\forall q \in \vartheta$ et $p \in \psi$.

Pour satisfaire cette condition, $2n \geqslant m$ est nécessaire mais non suffisante. Un exemple trivial d'un robot à six degrés de libèrté, $(m = 6)$, trois point alignés sont dégénérés pour la rotation du caméra autour la ligne qui contient les trois point.

### 3.3.3 Primitives redondantes

On dit que les primitives sont redondantes si le nombre de primitives est plus grand que celui des articulations. Pour quatre primitives points, le nombre de primitives est huit (les coordonnées $x$ et $y$ dans l'image), alors il y a une redondance pour contrôler les six degrés de libèrté de la caméra. La condition suffisante pour que le jacobien image soit de rang complet est donné par le lemme suivant

**Lemme 3.1** *[33] Supposant qu'il y a quatre points dans un plan et le vecteur des primitives correspondant est admissible. Alors le jacobien image est de rang complèt si n'importe quels trois points primitives sont non colinéaires dans le plan image.*

**Preuve.** Soit le plan, où les quatre points existent définit par $Z = pX + qY + r$. alors les $Z_i$ satisfont $Z_i = pX_i + qY_i + r$ pour $i = 1..4$ .Remplaçons (2.10) par ce dernier, donc

$$\begin{aligned}
\frac{f}{Z_i} &= \frac{f}{pX_i + qY_i + r} \\
&= f\frac{r}{Z_i r} \\
&= f\frac{Z_i - pX_i - qY_i}{Z_i r} \\
&= f\frac{\frac{Z_i - pX_i - qY_i}{Z_i}}{r} \\
&= \frac{f - \frac{f}{Z_i}.pX_i - \frac{f}{Z_i}q.Y_i}{r}
\end{aligned}$$

$$\frac{f}{Z_i} = \frac{f - p.x_i - q.y_i}{r} \tag{3.9}$$

et le remplacement de (3.9) dans (3.7) donne

$$J_{img}^{(i)} = \begin{bmatrix} -\frac{f - px_i - qy_i}{r} & 0 & \frac{x_i}{f}\frac{f - px_i - qy_i}{r} & \frac{x_i y_i}{f} & -\frac{x_i^2 + f^2}{f} & y_i \\ 0 & -\frac{f - px_i - qy_i}{r} & \frac{y_i}{f}\frac{f - px_i - qy_i}{r} & \frac{y_i^2 + f^2}{f} & -\frac{x_i y_i}{f} & -x_i \end{bmatrix}.$$

posons $M_i$ et $N$ comme suit

$$M_i = \begin{bmatrix} f & 0 & x_i & y_i & 0 & 0 & \frac{x_i^2}{f} & \frac{x_i y_i}{f} \\ 0 & f & 0 & 0 & x_i & y_i & \frac{x_i y_i}{f} & \frac{y_i^2}{f} \end{bmatrix}$$

$$N = \frac{1}{r} \begin{bmatrix} -1 & 0 & 0 & 0 & -r & 0 \\ 0 & -1 & 0 & r & 0 & 0 \\ \boldsymbol{p} & \boldsymbol{0} & \boldsymbol{1} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ q & 0 & 0 & 0 & 0 & r \\ 0 & p & 0 & 0 & 0 & -r \\ 0 & q & 1 & 0 & 0 & 0 \\ 0 & 0 & -p & 0 & -r & 0 \\ 0 & 0 & -q & r & 0 & 0 \end{bmatrix}$$

donc on a $J_{img}^{(i)} = M_i N$, alors $J = MN\ {}^{c}J_{rob}$ où $M = \begin{bmatrix} M_1^T & M_2^T & M_3^T & M_4^T \end{bmatrix}$. $J$ est de rang complet si $M$ et ${}^{c}J_{rob}$ sont inversibles et $N$ de rang complet.

Vérifier la singularité de $M$, revient à vérifier le déterminant

$$\det M = \begin{vmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{vmatrix} \cdot \begin{vmatrix} 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \\ 1 & x_4 & y_4 \end{vmatrix} \cdot \begin{vmatrix} 1 & x_3 & y_3 \\ 1 & x_4 & y_4 \\ 1 & x_1 & y_1 \end{vmatrix} \cdot \begin{vmatrix} 1 & x_4 & y_4 \\ 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \end{vmatrix}$$

donc $M$ est inversible car n'importe quels trois points primitives sont non colinéaires dans le plan image. Seul l'éspace non singulier du robot nous intéresse, ce qui implique que ${}^{c}J_{rob}$ est inversible. D'autre part, si $p^2 + q^2 \neq 0$, les six premieres lignes de $N$ son linéairement indépendantes. Donc $J$ est de rang complet.

■

L'observabilité et la sensibilité des primitives image, ainsi que la commandabilité et la stabilité du robot manipulateur doivent être assurées pour une loi de commande référencée image fiable. Pour celà, des mesures amèliorées pour évoluer les performances de l'asservissement visuel sont introduites par l'utilisation des primitives redondantes et les valeurs singulières du jacobien image. Dans[40], les primitives image sont sélectionnées pour que la valeur singulière minimale soit large afin d'améliorer la sensibilité de la commande référencée image.

Nous allons présenter une approche pour une amélioration des performances du contrôle par une description d'une matrice jacobienne image de troix blocs triangulaires basée sur les points fyants (*vanishing points*), une méthode pour réduire le rapport valeur singulière maximale sur valeur singu-

lière minimale (conditionnement), par l'augmentation (diminution) de la valeur singulière minimale (maximale), basée sur l'utilisation des blocs diagonaux de la matrice jacobienne (élimination d'un bloc supérieur).

## 3.4 Amélioration de l'asservissement visuel [35][40][44]

Le conditionnement et les valeurs singulières minimale et maximale sont utilisés pour améliorer les mesures du jacobien image. Le conditionnement $\kappa$ est défini par :

$$\kappa = \frac{\sigma_{\max}}{\sigma_{\min}} \geqslant 1$$

où $\sigma_{\max}$ est la valeur singulière maximale et $\sigma_{\min}$ est la valeur singulière minimale du jacobien image. $\kappa$ est utilisé pour représenter la commandabilité et la sensibilité du système. La commandabilité $c(J)$ d'un jacobien image $J$ s'exprime comme :

$$c(J) = \|J\| \,.\, \left\|J^{-1}\right\|$$

Soit $\sigma_1 \cdots \sigma_p$ les valeurs singulières de $J$. on a donc :

$$\begin{aligned} \|J\| &= \max\{\sigma_1, \cdots, \sigma_p\} \\ \left\|J^{-1}\right\| &= \frac{1}{\min\{\sigma_1, \cdots, \sigma_p\}} \\ \kappa &= \|J\| \,.\, \left\|J^{-1}\right\| = \frac{\sigma_{\max}}{\sigma_{\min}} \end{aligned}$$

$\left\|J^{-1}\right\|$ représente la differentielle du mouvement du robot par rapport à la differentielle du changement des primitives dans l'image. Quand cette norme est grande (c'est-à-dire que la valeur singulière minimale de $J$ est petite), un petit changement des primitives dans l'image provoque un grand mouvement et brusque pour le robot. Contrairement, une differentielle dans les primitives de l'image est trés sensible au bruit dans le cas où la valeur singulière maximale de $\|J\|$ est grande. Ainsi nous pouvons obtenir une loi de commande insensible au bruit si le conditionnement est proche de 1.

Soit un jacobien image à trois blocs triangulaires, la suppression d'un bloc de la matrice jacobienne de manière qu'elle soit diagonale, rendra le système plus performant. Le théorème suivant présente celà.

**Théorème 3.1** *[40] Soit $A$ une matrice triangulaire à tois blocs et $C(A)$ la matrice $A$ réduite à une*

*matrice diagonale à deux blocs diagonaux données comme suit :*

$$A = \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix} \quad et \quad C(A) = \begin{bmatrix} A_{11} & 0 \\ 0 & A_{22} \end{bmatrix}$$

*alors,*

$$\sigma_{\max\ C(A)} \leqslant \sigma_{\max\ A} \quad et \quad \sigma_{\min\ C(A)} \geqslant \sigma_{\min\ A}$$

**Preuve.** La matrice diagonale $C(A)$ peut être écrite comme

$$C(A) = \frac{1}{2}\left(A + UAU^{-1}\right)$$

où

$$U = \begin{bmatrix} I & 0 \\ 0 & -I \end{bmatrix}$$

Par l'inégalité triangulaire, on à

$$\|C(A)\| \leqslant \frac{1}{2}\|A\| + \frac{1}{2}\|UAU^{-1}\| \quad \text{et} \quad \|C(A)\| \leqslant \|A\| \tag{3.10}$$

Comme $A$ et $UAU^{-1}$ sont similaires et leur valeurs propres et valeurs singulières sont égales. Alors

$$\sigma_{\max\ C(A)} \leqslant \sigma_{\max\ A}$$

L'inverse de la matrice $A$ est donnée par

$$A^{-1} = \begin{bmatrix} A_{11}^{-1} & -A_{11}^{-1}A_{12}A_{22}^{-1} \\ 0 & A_{22}^{-1} \end{bmatrix} \quad \text{et} \quad C(A^{-1}) = \begin{bmatrix} A_{11}^{-1} & 0 \\ 0 & A_{22}^{-1} \end{bmatrix}$$

Par (3.10),

$$\frac{1}{\sigma_{\min\ C(A)}} \leqslant \frac{1}{\sigma_{\min\ A}}$$

celà implique que

$$\sigma_{\min\ C(A)} \geqslant \sigma_{\min\ A}$$

En conséquence, la sensibilité de $C(A)$ augmente par rapport à celle de $A$. ■

Nous avons vu précédemment que la matrice jacobienne définie par le mouvement relatif entre l'objet cible et la caméra (effecteur) est non triangulaire. Une redondance des primitives peut nous amener à une formulation d'un jacobien image triangulaire. Pour celà, nous décrivons la méthode

des points fuyants (Vanishing points) pour un objet réctangulaire planaire.

### 3.4.1 Jacobien image basé sur les points de fuite (*Vanishing points*)

La matrice jacobienne image triangulaire à trois blocs est basée sur les points fuyants d'un rectangle planaire présenté sur la figure 3.2 puisque la rotation peut être extraite de ces points indépendamment de la position du réctangle.

Soit

$u, v$ : les coordonnées du centre de gravité de l'objet.

$a$ : la surface de l'objet.

$x_v$ : l'Abscisse du point fuyant verticale.

$x_h, y_h$ : les coordonnées de point de fuite horizontale.

Soit $R$ le repère de référence, $R'$ la translation de $R$ dans la direction $x$, $R"$ est la rotation de $R'$ par l'angle de tangage (*pitch*) $\beta$, angle de lacet (*yaw*) $\alpha$ et angle de roulis (*roll*) $\gamma$, $R_{img}$ le repère image où $f$ est la distance focale. Donc, les coordonnées des points fuyants et de la surface du réctangle sont données par

$$\begin{aligned} x_v &= f\frac{\cos\gamma}{\tan\beta} \\ x_h &= -f\left(\frac{\sin\gamma}{\tan\alpha\ \cos\beta} + \tan\beta.\cos\gamma\right) \\ y_h &= f\left(\frac{-\cos\gamma}{\tan\alpha\ \cos\beta} + \tan\beta.\sin\gamma\right) \\ a &= \frac{cf^2}{z^2}\cos\alpha\cos\beta \end{aligned} \tag{3.11}$$

où $c$ est la surface du réctangle dans le repère image $R_{img}$. La démonstration de (3.11) est presentée dans l'annexe **A**.

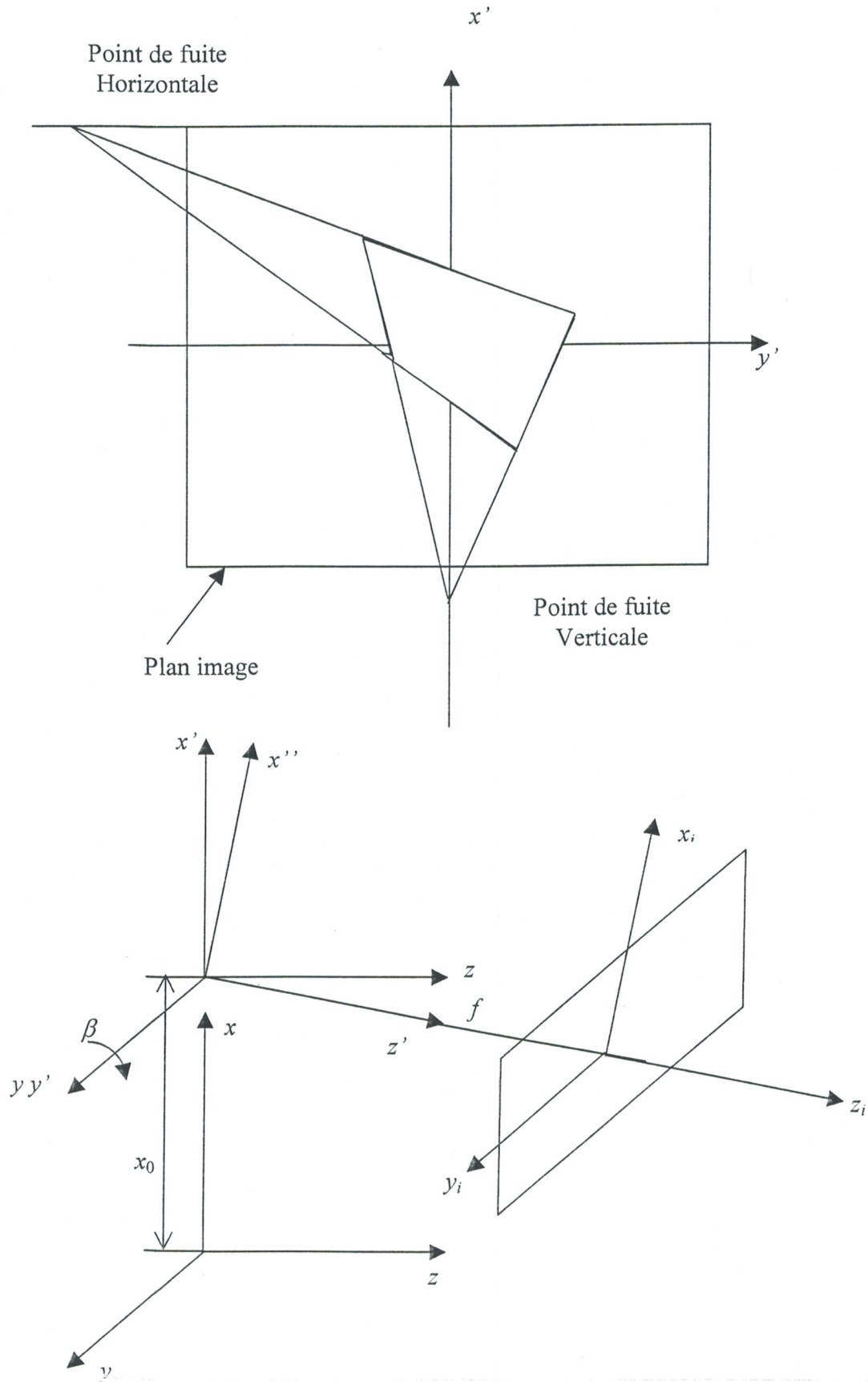



Figure 3.2 : Les points de fuite (Vanishing points)

Le jacobien image triangulaire peut être obtenu par la différentiation des coordonnées et la surface données dans (3.11) et de la combinésation avec le jacobien image donnée dans (3.7), on obtient

$$\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{a} \\ \dot{x}_v \\ \dot{x}_h \\ \dot{y}_h \end{bmatrix} = \begin{bmatrix} J_{11} & J_{12} \\ 0 & J_{22} \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{\alpha} \\ \dot{\beta} \\ \dot{\gamma} \end{bmatrix}$$

où

$$J_{11} = \begin{bmatrix} \frac{f}{z} & 0 & -\frac{u}{z} \\ 0 & \frac{f}{z} & -\frac{v}{z} \\ 0 & 0 & -\frac{2cf^2}{z^3}\cos\alpha\cos\beta \end{bmatrix}$$

$$J_{12} = \begin{bmatrix} -\frac{uv}{f} & \frac{f^2+u^2}{f} & -v \\ -\frac{f^2+u^2}{f} & \frac{uv}{f} & u \\ \frac{cf^2}{z^2}\cos\alpha\cos\beta & \frac{cf^2}{z^2}\cos\alpha\cos\beta & 0 \end{bmatrix}$$

$$J_{22} = \begin{bmatrix} 0 & -\frac{\cos\gamma}{\tan^2\beta} & -\frac{\sin\gamma}{\tan\beta} \\ \frac{\sin\gamma}{\sin^2\alpha\cos\beta} & -\frac{\tan\beta\sin\gamma}{\tan\alpha\cos\beta} - \frac{\cos\gamma}{\cos^2\beta} & -\frac{\cos\gamma}{\tan\alpha\cos\beta} + \tan\beta\sin\gamma \\ \frac{\cos\gamma}{\sin^2\alpha\cos\beta} & -\frac{\tan\beta\cos\gamma}{\tan\alpha\cos\beta} + \frac{\sin\gamma}{\cos^2\beta} & \frac{\sin\gamma}{\tan\alpha\cos\beta} + \tan\beta\cos\gamma \end{bmatrix}$$

Le contrôle de vitesse des différentes composantes du repère caméra dépend de la vitesse des primitives image via l'inverse (ou le pseudo-inverse) de la matrice jacobienne image. Dans les approches classiques, l'orientation de la caméra (ou effecteur) est contrôlé par les coordonnées des primitives visuelles dans l'image, eux même dépendant de la position de la camèra dans l'espace cartésien. Les coordonnées des points de fuite ne dépendent que des angles d'orientation $\alpha$, $\beta$ et $\gamma$ (suivant 3.11), l'inverse du jacobien image est :

$$J_{img}^{-1} = \begin{bmatrix} J_{11}^{-1} & -J_{11}^{-1}J_{12}J_{22}^{-1} \\ 0 & J_{22}^{-1} \end{bmatrix}$$

lorsque le terme $-J_{11}^{-1}J_{12}J_{22}^{-1}$ est négligeable, l'inverse du jacobien image peut être approximé par :

$$J_{img}^{-1} = \begin{bmatrix} J_{11}^{-1} & 0 \\ 0 & J_{22}^{-1} \end{bmatrix}$$

Le passage d'un jacobien triangulaire à un jacobien diagonale nous assure une augmentation (diminution) de la valeur singulière minimale (maximale), et donc un conditionnement plus faible.

## 3.5 Loi de commande en asservissement visuel [10] [12][21][22] [32][33][3[41][43][47]

Le but de la plus part des système d'asservissement visuel est de placer ou suivre l'objet cible pour que les primitives de l'objet restent dans une position désirée dans l'image. Pour celà, le modèle du robot et de la caméra sont nécessaires, et avec une bonne précision, pour introduire une loi de commande robuste et stable.

Il est naturel d'adopter les primitives visuelles comme des variables de commande, et les vitesses articulaires comme des variables d'entrée.

Au minimum, le nombre des primitives doit au moins être égal au nombre de degrés de libèrté ($n = m$). D'autre part, une redondance des primitives provoque la non-commandabilité du système a cause de l'impossibilité du mouvement des primitives dans $R^n$ pour certaines positions désirées choisies arbitrairement. Pour résoudre ce problème, on est ramené à resoudre les contraintes géométriques non linéaire dans les primitives qui représentent la rigidité de l'objet. Une linéarisation des contraintes à la position de référence ou une réduction de la dimentions du vecteur des primitives à la dimention de l'espace articulaire nous permet de générer les variables de commande, appelées *fonction de tâche.*

Soit $r_d$ la position nominale pour une position désirées $\xi_d$ des prémitives. On définit la matrice $C$ (3.1) comme suite

$$C = \begin{cases} J(q^*, p^*) & \text{si } n \geqslant m \\ I & \text{si } n = m \end{cases} \tag{3.12}$$

$J$ est le jacobien image qui depend des position articulaires $q$ et la position de l'objet $p$ (dans le cas d'un objet mobile), $q^*$ et $p^*$ satisfont $s_{cam}(q^*) - s_{obj}(p^*) = r_d$. La variable de commande (ou fonction de tâche) est définie comme suit

$$e = C^T(\xi - \xi_d)$$

### 3.5.1 Commande en couple

Pour une commande en couple d'un bras manipulateur ayant le modèle dynamique suivant

$$\frac{d}{dt}\begin{bmatrix} q \\ \dot{q} \end{bmatrix} = \begin{bmatrix} \dot{q} \\ -M^{-1}h \end{bmatrix} + \begin{bmatrix} 0 \\ M^{-1} \end{bmatrix} \tau$$

où $\tau$ est le vecteur couple, $M$ matrice d'inertie, $h$ est le vecteur qui rassemble les contributions des forces de gravité, centrifuges et de coriolis, la seconde dérivée de $e$ donne

$$\ddot{e} = C^T J M^{-1}(\tau - h) + \varphi(q, p, e, \theta^*)$$

alors la commande en couple est donnée avec la nouvelle entrée $u$ par

$$\begin{aligned} \tau &= M(C^T J)^{-1}(u - \varphi(q, p, e, \theta^*)) + h \\ u &= -K_1 e - K_2 \dot{e} \end{aligned} \tag{3.13}$$

où $K_1$ et $K_2$ sont des matrices de gain définies positives. Avec cette loi de commande, la fonction de tâche et sa dérivée convergent exponentiellement vers zéro.

### 3.5.2 Commande en vitesse

Le jacobien de la fonction de tâche est obtenue en multipliant la matrice $C^T$ par la matrice d'interaction (jacobien image) $J_{img} = \dfrac{\partial \xi}{\partial r}$ associée aux information visuelles

$$L = \frac{\partial e}{\partial \xi}\frac{\partial \xi}{\partial r} = C^T . J_{img}$$

Dans le cas de l'asservissement visuel 3D, la matrice $C$ est imposée par le choix de la fonction de tâche, alors que dans le cas d'asservissement visuel 2D, c'est le choix de $C$ qui détermine la fonction de tâche.

Une autre solution simple largement adoptée est de choisir $C = I_n$. Afin d'avoir une fonction de tâche admissible, il faudrait choisir $n = 6$ (3 primitives point), mais il est possible de rencontrer une singularité de la matrice d'intéraction. Si on prend plus que 3 primitives point ($n > 6$), la dimension du vecteur de sortie (ou de tâche) est supèrieure à celle du vecteur de commande (le vecteur de vitesse de la caméra est de dimention 6), alors on sort du cadre défini par l'admissibilité de la fonction de tâche, En effet, le nombre d'informations visuelles que l'on désire contrôler est supérieur aux degrés de liberté du robot et il existe alors des trajectoires irréalisables pour lesquelles il n'existe pas de mouvement du robot (caméra).

Une solution qui permet de tenir compte de la redondance d'informations est de chosir $C^T = J^+_{img}(q^*, p^*, z^*)$, mais elle nécessite une inversion (pseudo-inverse) de la matrice d'interaction. La solution présentée dans (3.12) (pour le cas redondant $n \geqslant m$) élimine la nécessité de l'inversion de la matrice d'interaction.

Le schéma bloc de l'asservissement visuel 2D est donné en figure 3.3.

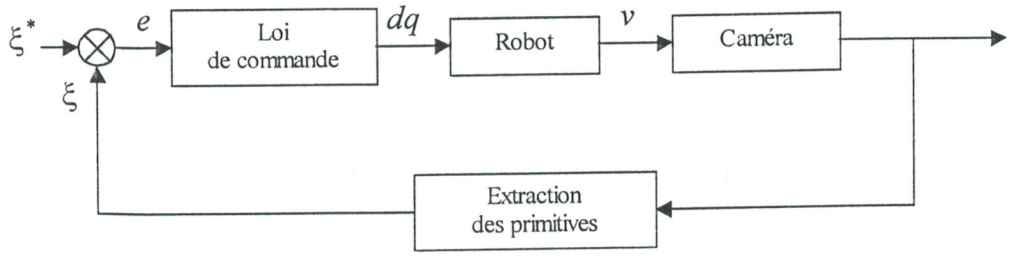



Figure 3.3 : Schéma bloc de l'asservissement visuel 2D

Pour une convergence exponentielle de la fonction de tâche, la loi de commande est donnée par

$$\dot{r} = -\lambda \hat{L}^{-1} e \tag{3.14}$$

où $\dot{r}$ le vecteur des vitesses cartésiennes de la caméra (effecteur), $\lambda$ est le gain fixant la vitesse de décroissance de la fonction de tâche, $L$ le jacobien de la fonction de tâche.

Dans une tâche de positionnement, un contrôleur proportionnel est suffisant car le système à contrôler est déjà de nature intégrale et le contrôleur bas niveau du robot est généralement de type PID. Une intégration supplémentaire est utile seulement dans le cas de poursuite d'une cible mobile à vitesse constante. D'autres lois de commande ont été utilisées, une étude comparative entre une commande proportionnelle intégrale, une commande à placement de pôles et une commande LQG a été effectuée par Papanikolopoulos [46] et une commande optimale de type LQ a été utilisée par Hashimoto [35]. De plus, une commande de type GPC a été récement appliquée à l'asservissement visuel par Gangloff [32].

Le choix de $\hat{L}$ est souvent lié au choix de la matrice $C^T$ car $\hat{L} = C^T \hat{J}_{img}$. Differents choix pour $C^T$ sont donc possible

- le cas où $C = I$ alors

♦ La matrice d'interaction peut être approchée par une équation aux différences finies et estimée en effectuant une série de mouvements de calibration orthogonaux ou par les réseaux de neurones et la logique floue.

♦ La matrice d'intéraction peut être calculée à chaque itération sur la base du modèle donné par l'équations (3.7), donc $\hat{L} = J_{img}(\xi, \hat{z})$, on voit que la matrice d'intéraction dépend de la profondeur $z$ des points de primitives. Une estimation de cette dernière peut être obtenue en utilisant un algorithme de calcul de pose (le modèle géométrique de la cible est connu), ou en utilisant un

algorithme de vision dynamique (si le déplacement de la caméra est connu).

Dans le premier cas (estimation de la matrice d'intéraction), on ne peut pas calculer analytiquement l'équation différentielle qui régit le comportement du système en boucle fermée, d'où la difficulté de démontrer la stabilité du système. Dans le deuxième cas, la matrice $Q$ de la boucle fermée du système est donnée par

$$Q = J_{img}(\xi, z).J^{+}_{img}(\xi, \hat{z})$$

Il faut remarquer que le jacobien image depend des paramètres intrinsèques de la caméra. Mème dans le cas idéal où les paramètres intrinsèques ont une bonne calibration, ainsi qu'une bonne estimation de la profondeur $\hat{z} = z$, pour une redondance d'information visuelle (*nombre de point* $> 3$) alors $Q \neq I$. De plus, ce choix peut conduire le système à atteindre un minimum local (voir une singularité de la matrice d'intéraction).

- Le cas où $C^T = J^{+}_{img}(q^*, \hat{z}^*)$, alors

♦ La matrice d'intéraction peut être choisie constante $J_{img}(q^*, \hat{z}^*)$, donc $\hat{L} = C^T.J_{img} = I$, dans ce cas, la matrice d'intéraction $\hat{L}$ utilisée dans la loi de commande ne peut être singulière si la position désirée a été bien choisie avec une estimation correcte de la profondeur $\hat{z}^*$. L'utilisation d'une matrice constante contraint moins la trajectoire des primitives dans l'image et ceux-ci peuvent éventuellement sortir du champ de vision de la caméra. Toutefois, la matrice d'interaction réelle peut toujours devenir singulière au cours de la tâche de mouvement du robot. En effet, la matrice de transfet en boucle fermée $Q$ dépend toujours de la matrice d'interaction réelle

$$Q = L.\hat{L}^{-1} = J^{+}_{img}(q^*, \hat{z}^*).J_{img}(q, z)$$

Dans un voisinage de la position désirée ($q = q^*$ où $\xi = \xi^*$ et $z = z^*$), et dans le cas idéal où la calibration de la caméra est parfaite, ce type de commande réalise un découplage de la boucle fermée $Q \approx I > 0$, alors le système est stable.

♦ La matrice d'interaction $\hat{J}_{img}$ peut être estimée à chaque itération. toutefois, ce cas est plus coûteux en temps de calcul sans pour autant apporter d'améliorations.

- Le cas où $C^T = J^{T}_{img}(q^*, z^*)$, alors

♦ La matrice d'intéraction peut être choisie telle que $\hat{J}_{img} = J^{T+}_{img}(q^*, \hat{z}^*)$, donc $\hat{L} = I$, La matrice $Q$ de la boucle fermée du système dépend toujours de la matrice d'intéraction réelle et elle est donnée par $Q = J^{T}_{img}(q^*, \hat{z}^*).J_{img}(q, z)$. Par rapport au choix précédent, dans un voisunage de la position désirée ($\xi = \xi^*$ et $z = z^*$), et dans le cas idéal où la calibration de la caméra est parfaite, ce type de commande est stable car $Q \approx J^{T}_{img}.J_{img} > 0$, mais elle ne réalise pas de découplage de la

boucle fermée.

♦ La matrice d'interaction peut être estimée à chaque itération. Là encore, approche coûteuse en temps de calcul.

### 3.5.3 Réglage du gain proportionnel $\lambda$

Dans la loi de commande (3.14), le gain $\lambda$ régit la vitesse de convergence de la fonction de tâche $e$, plus une convergence rapide est souhaitée, plus ce gain doit être fixé à une valeur forte. Cependant, $\lambda$ agit également directement sur la stabilité du système, une valeur trés forte peut entrainer l'instabilité et donc une divergence de la loi de commande. Le réglage de $\lambda$ consiste donc à trouver un compromis satisfaisant entre la stabilité, qui doit absolument être assurée, et une vitesse de convergence la plus grande possible.

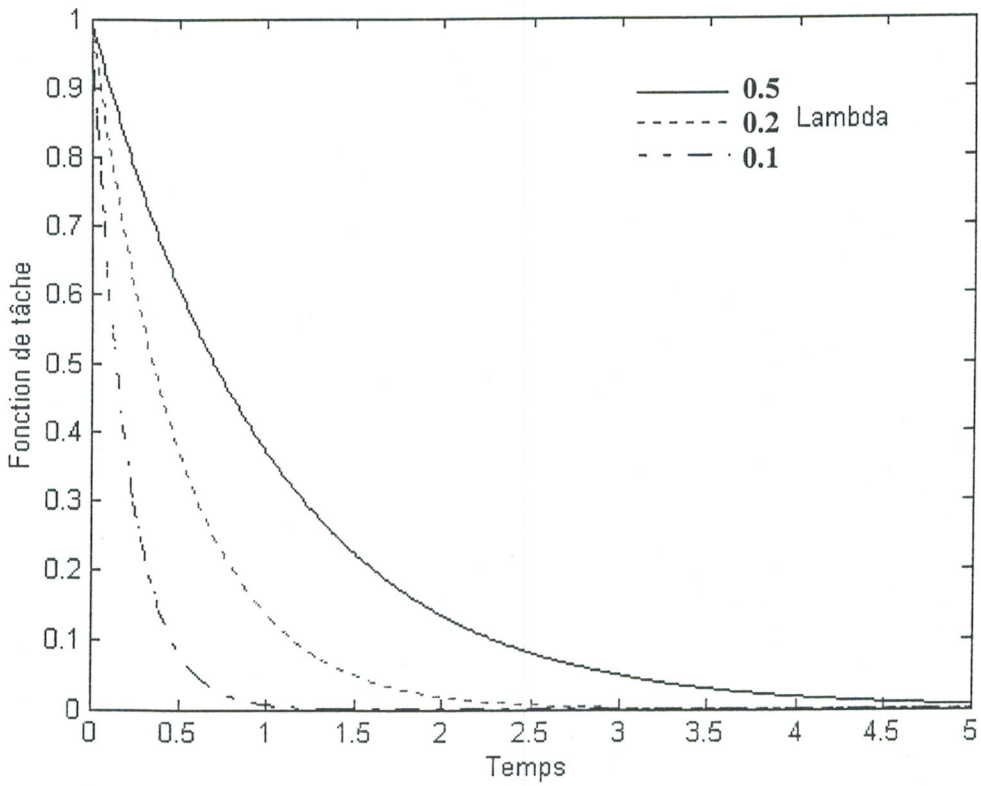



Figure 3.3 : Influence du gain $\lambda$ sur la décroissance de la fonction de tâche

Plus précisèment, $\lambda$ dépend essentiellement de la période d'échantillonnage du système. Plus cette période est faible, plus le gain $\lambda$ peut être choisi élevé. Cette période d'échantillonnage peut être déterminée une fois pour toutes en fonction des capacités *software* et *hardware* du matériel informatique sur lequel sera exécuté le traitement d'image et le calcul de la loi de commande. Le gain $\lambda$ dépend de la valeur de la fonction de tâche $e$ et donc de l'erreur dans l'image $\xi - \xi^*$. En effet, pour une forte valeur de $e$, $\lambda$ devra être choisi faible en raison des problèmes de stabilité,

car la consigne de commande du robot exprimée en vitesse de déplacement dépend du terme $\lambda e$ (equation 3.14) ; par contre, pour de faibles valeurs de $e$, $\lambda$ pourra prendre des valeurs plus fortes. En raison de la loi d'évolution exponentielle choisie pour $e$ et qui a permis d'élaborer la loi de commande (3.14), une valeur de $\lambda$ constante, préservant la stabilité du système, va alors donner le comportement suivant : une forte décroissance aux premières itérations, puis une décroissance extrèmement faible pour arriver à la convergence. Une valeur plus forte de $\lambda$ entrainerait une décroissance plus rapide en fin de convergence, mais ne préseverait alors sans doute plus la stabilité pour de fortes valeurs de $e$.

Une solution pour optimiser la vitesse de convergence tout en préservant la stabilité du système consiste, quand cela est possible, à contrôler la trajectoire dans l'image que doit suivre $\xi$ pour atteindre $\xi^*$. Plus précisèment, à chaque itération $k$ de la boucle de commande est calculée une valeur $\xi_k^*$ de manière à ce que l'erreur $e_k = J_{img}^{T+}.(\xi_k - \xi_k^*)$ soit toujours inférieure à une certaine valeur $e_{\max}$ (ce qui revient à saturer la commande). Cela nous permet d'utiliser un gain $\lambda$ constant, déterminé pour assurer la convergence le plus rapidement possible à partir d'une erreur initiale $e_{\max}$.

Cette tâche a été employée pour une tâche qui consiste à positionner la caméra par asservissement visuel, de telle sorte que le centre de gravité d'un objet dans l'image à la position $\xi = (x_0, y_0)^T$ se trouve à la position désirée $\xi^* = (x_d, y_d)^T$ dans l'image (voire figure 3.4), en contrôlant uniquement deux composantes de vitesse articulaire pour qu'il soit un déplacement dans les deux sens à l'image.

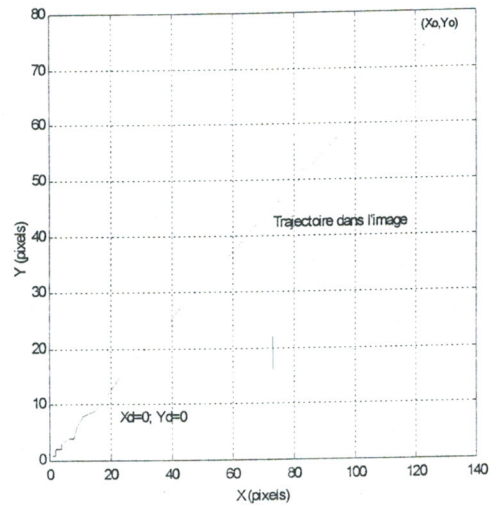
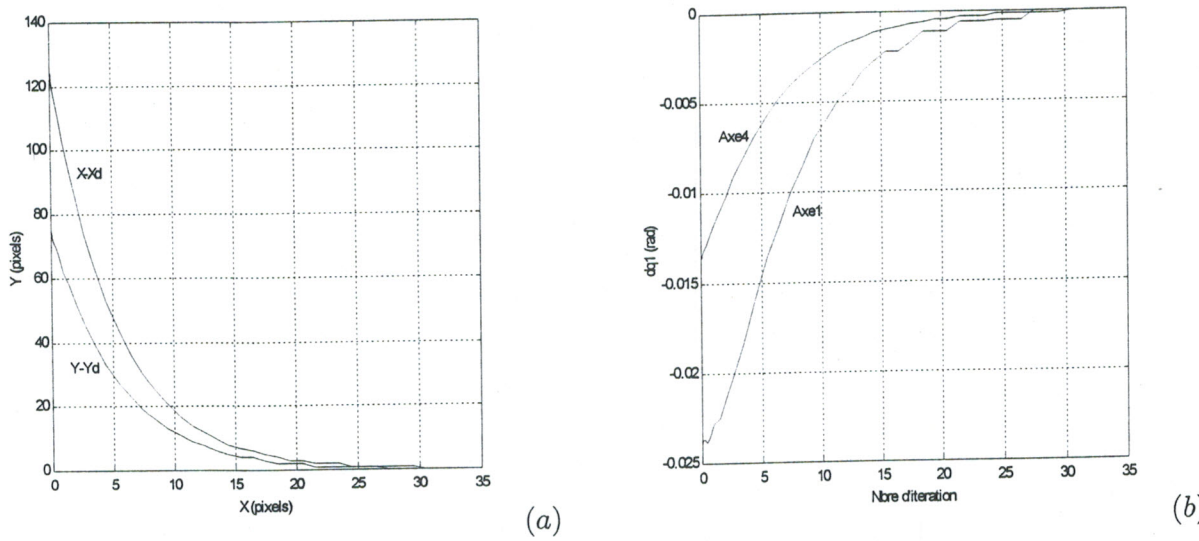



Figure 3.4 : Positionnement d'objet dans l'image




(a) (b)

Figure 3.5 : (a) : Convergence de $\xi - \xi^*$ vers 0 (b) : La commande des axes 1 et 4

Les courbes de la figure 3.5 traduisent le comportement en fonction du temps des composantes $\xi_i - \xi_i^*$ et les deux composantes de $\dot{q}$. le gain $\lambda$ dans cette configuration est constant et l'on peut observer la décroissance exponentielle de $\xi - \xi^*$ et $\dot{q}$.

Pour cette tâche, puisqu'il nous est possible de connaitre la position initiale du centre de gravité et sa position d'arrivée dans l'image, par le simple calcul de la droite passant par ces deux points, il nous est possible de déterminer les coordonnées de points intermédiaires $\xi_i$ situés sur cette image et espacés de façon régulière dans l'image (voir figure 3.6).

Ainsi, pour aller à la position désirée $\xi^*$, la caméra cherchera à amener l'image de l'objet sur les différentes positions intermédiaires successives. Comme la distance entre deux points intermédiaires est plus faible que la distance totale à parcourir, l'écart entre la position relevée à chaque itération et le point intermédiaire suivant sera réduit, de sorte que l'on pourra autoriser une commande plus forte, par l'utilisation d'un gain plus élevé.

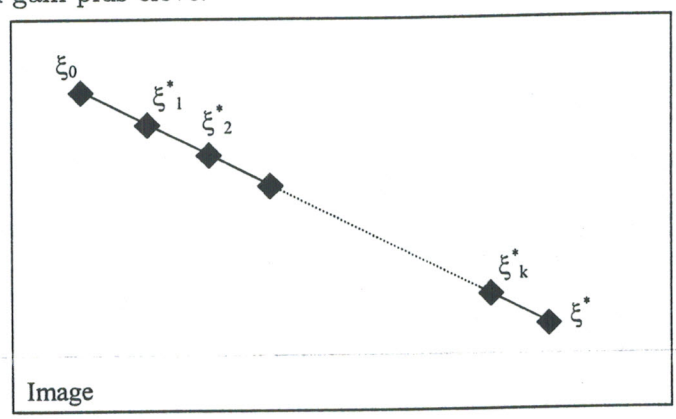



Figure 3.6 : Les points intermédiaires

La validation de ces approches a été effectuée grâce à des simulations présentées dans le chapitre 4.

## 3.6 Approche basée sur l'estimation du modèle visuel-moteur [36]

Dans un contrôle visuel traditionnel, on doit connaitre deux fonctions à priori ou par calibration. La première est celle de la caméra qui transforme les coordonnées des primitives dans l'éspace image à des coordonnées cartesiennes dans l'éspace de travail. La seconde celle du robot (modèle géométrique), qui transforme la position de la caméra (ou effecteur) dans l'éspace de travail à des angles dans l'éspace articulaire.

La connaissance de ces modèles nécessite une précision suffisante pour que le système soit stable, robuste et précise. Celà est limité par les outils de calibration utilisés, même des fois par le materièl de contrôle, par exemple robot non calibré, modèle inconu et commande limitée sur les positions articulaires. La même chose reste valable pour la caméra.

Le modèle globale du système qui transforme les coordonnées primitives $\xi$ dans l'espace image à des angles $q$ dans l'espace articulaire est donnée par $\xi = f(q)$ tel que $f^{-1} = g(h(\cdot))$ est la fonction du modèle globale, $g$ est le modèle du robot et $h$ le modèle de la caméra.

Nous présentons, dans cette partie, une méthode qui permet d'apprendre la fonction inconnue $f$. Le système n'exige aucune connaissance a priori du modèle de système. Elle est estimée chaque itération de la boucle d'asservissement (On line) sous forme d'un modèle linéaire jusqu'à qu'elle atteigne la tâche voulue. Aucun apprentissage ou mouvement ne doit être exécuté au préalable. le modèle est appris avec succès sous des contraintes approximatives.

### 3.6.1 Déduction de la méthode d'estimation

En générale, la fonction qui définit le modèle global du système de vision inconnu, appelé fonction visuel-moteur, $f : R^m \longrightarrow R^n$ peut être écrite sous forme d'un développement de Taylor. Particulièrement, on s'intérèsse pour une approximation linéaire :

$$f(q) = f(q_0) + J(q_0)(q - q_0) + \partial^2(q_0, q)$$

où, pour que $f$ soit suffisament fine, le tèrme $\partial^2$ soit trop faible au voisinage de $q_0$, $J$ est le jacobien visuel-moteur définie comme

$$J_{ji} = \frac{\partial f_j}{\partial q_i}$$

Il exprime les variations élementaires des coordonnées primitives dans l'espace image en fonction des variations élementaires des coordonnées articulaires dans l'éspace articulaire, c'est-à-dire, pour

$\xi = f(q)$, $\xi_0 = f(q_0)$, $\Delta\xi = \xi - \xi_0$ et $\Delta q = q - q_0$ on à

$$\Delta\xi = J.\Delta q$$

ainsi que ce jacobien est local autour de $q_0$.

A l'état initial, la perception des primitives est donnée dans $\xi_0$, le but du contrôle visuel est de ramener les primitives visuelles à la position $\xi^*$, utilisons le modèle linéaire du jacobien pour une prédiction de l'incertitude $\Delta q$, cette dernière est donnée par la solution du système d'équations suivant

$$\xi^* - \xi_0 = J.\Delta q$$

l'exécution de ce mouvement, c'est-à-dire, déplacer le robot à la position $q_1 = q_0 + \Delta q$, agir sur la position de la caméra, qui fait changer les coordonnées primitives dans l'éspace image à $\xi_1$, comme $f$ est un modèle linéaire, typiquement $\xi_1 \neq \xi_0$. Cependant, si $\|\xi^* - \xi_0\|$ est trés petite et $f$ fine, il est vérifié que $\|\xi^* - \xi_1\| < \|\xi^* - \xi_0\|$, le processus est répété jusqu'à ce que les coordonnées primitives atteignent la position désirée $\xi^*$. ceci est une méthode quasi-Newton pour résoudre le système d'équation dans $q$ (chercher $q^*$ qui assure $0 = \xi^* - f(q^*)$). Dans un système continue, la loi de commande est donnée par

$$\xi^* - \xi = kJ\frac{d}{dt}q \tag{3.15}$$

où $k$ est la matrice gain.

### 3.6.2 La méthode de Newton - Formule de Broyden[8]

#### La méthode de Newton

Soit $X$ un espace de Hilbert et $F$ une application $C^1 : X \longrightarrow X$ de dérivée $F'$ localement lipschitzienne. L'equation

$$F(x) = 0$$

Linéarisée en $x$ donne l'équation de Newton

$$F(x) + F'(x).d = 0$$

où $d \in X$ est (si elle existe) la direction de Newton en $x$.

**Algorithme 3.1 (Méthode de Newton)** *Valeures initiales $x_0 \in X$ pour $k = 0$*

*Tant que : $F'(x_k) \neq 0$*

*Calculer $d_k$, direction de Newton en $x_k$*

$$x_{k+1} = x_k + d_k$$

$k = k + 1$

Cet algorithme est bien défini si $F'(x_k)$ est un automorphisme pour tout $k$.

$\tilde{x}$ est dite un zéro régulier de $F$ si $F(\tilde{x}) = 0$ et $F'(\tilde{x})$ est un automorphisme.

### Principe des méthodes de quasi-Newton. Formule de Broyden

Soit $\{x_k\} \longrightarrow \tilde{x}$ une suite dans $X$ Nous allons voir un procédé de calcul d'approximation de $\{F'(x_k)\}$ par une suite $\{M_k\}$, avec une analyse de la convergence de l'algorithme.

Posons

$$\begin{aligned} s_k &= x_{k+1} - x_k \\ y_k &= F(x_{k+1}) - F(x_k) \end{aligned}$$

Alors $F'(x_k).s_k = y_k + o(s_k)$. Ceci motive d'imposer à $M_{k+1}$ la vérification de l'équation de quasi-Newton

$$M_{k+1}.s_k = y_k$$

Cette relation ne détermine pas $M_{k+1}$ de manière unique (si $\dim(X) > 1$). Afin d'obtenir la stabilité numérique de l'algorithme on peut prendre $M_{k+1}$ aussi proche que possible de $M_k$. Avec la norme canonique ceci revient à chosir $M_{k+1}$ solution du problème

$$\min \|M - M_k\| \, ; M.s_k = y_k \tag{3.16}$$

On suppose par la suite que $s_k$ n'est jamais nulle.

**Lemme 3.2** *[8]Une solution du problème (3.16) est donnée par la formule de Broyden*

$$M_{k+1} = M_k + (y_k - M_k.s_k)\frac{s_k}{\|s_k\|^2}$$

Combinant la formule de Broyden avec la méthode de type Newton on obtient la méthode de Broyden.

**Algorithme 3.2 (Méthode de Broyden)** *Donnons $x_0 \in X$, $M_0$ automorphisme de $X$,*

$k = 0$.

*Tant que $F(x_k) \neq 0$ :*

*Calculer $d_k$ solution de*

$$F(x_k) + M_k.d_k = 0$$

$$x_{k+1} = x_k + d_k$$
$$s_k = x_{k+1} - x_k$$
$$y_k = F(x_{k+1}) - F(x_k)$$
$$M_{k+1} = M_k + (y_k - M_k.s_k)\frac{s_k}{\|s_k\|^2}$$
$$k = k + 1$$

**Théorème 3.2 (Convergence linéaire de la méthode de Broyden)** *[8]Soit $\tilde{x}$ un zéro régulier de $F$. Il existe $\varepsilon > 0$ tel que si $x_0 \in B(x, \varepsilon)$ et $M_0 \in B(F'(\tilde{x}), \varepsilon)$ alors la méthode de Broyden est bien définie ($M_k$ inversible, $\forall k$) et $x_k$ tend vers $\tilde{x}$ linéairement.*

### Méthode de Broyden duale

Le calcul d'une approximation du jacobien conduit à résoudre une équation linéaire à chaque itération de la méthode de Broyden. Une alternative consiste à calculer une approximation de l'inverse du jacobien ; si $A_k$ est cette approximation à l'itération $k$ le déplacement $d_k$ sera simplement $-A_k F(x_k)$.

Les concepts et résultats concernant la méthode de Broyden se transposent alors facilement. Ainsi, on peut imposer à $A_k$ de satisfaire l'équation de quasi-Newton duale

$$A_{k+1}.y_k = s_k$$

Si on impose en plus à $A_{k+1}$ d'être le plus proche possible de $A_k$ ceci revient à choisir $A_{k+1}$ solution du problème

$$\min \|A - A_k\| \; ; A.y_k = s_k \tag{3.17}$$

Ce problème est similaire à (3.16) et on passe de l'un à l'autre en effectuant la substitution de variables

$$\begin{pmatrix} M_k \\ s_k \\ y_k \end{pmatrix} \longrightarrow \begin{pmatrix} A_k \\ y_k \\ s_k \end{pmatrix}$$

Ceci et le lemme précédent montrent qu'une solution de (3.17) est donnée par la formule de Broyden duale

$$A_{k+1} = A_k + (s_k - A_k.y_k)\frac{y_k}{\|y_k\|^2}$$

L'algorithme s'écrit alors de la façon suivante

**Algorithme 3.3 (Méthode de Broyden duale)** *Données $x_0 \in X$, $A_0$ automorphisme de $X$,*
$k = 0$

*Tant que* $F(x_k) \neq 0$ *alors*

$$x_{k+1} = x_k - A_k.F(x_k)$$
$$s_k = x_{k+1} - x_k$$
$$y_k = F(x_{k+1}) - F(x_k)$$
$$A_{k+1} = A_k + (s_k - A_k.y_k)\frac{y_k}{\|y_k\|^2}$$
$$k = k + 1$$

**Théorème 3.3 (Convergence linéaire de la méthode de Broyden duale)** *[8]Soit* $\tilde{x}$ *un zéro régulier de* $F$. *Il existe* $\varepsilon > 0$ *tel que si* $x_0 \in B(\tilde{x}, \varepsilon)$ *et* $A_0 \in B(F'(\tilde{x})^{-1}, \varepsilon)$, *la méthode de Broyden duale est bien définie et* $x_k$ *tend vers* $\tilde{x}$ *linéairement.*

### 3.6.3 Estimation du jacobien par la méthode de Broyden dual

L'implémentation de la loi de commande (3.15) nécessite la connaissance de la forme analytique du jacobien. Les approches classiques (vue un peu plus haut) sont basées sur le modèle géométrique du robot et le modèle de la caméra. A l'aide d'une expression analytique du jacobien, on calcule l'expression numérique à chaque itération pour le nouveau $q$, où l'inversion de ce dernier est lourd. Il est plus utile d'utiliser un modèle approximatif pour un positionnement précis. Avec un changement simultané des positions primitives dans l'image en fonction du changement des configurations articulaires du robot, on peut estimer la dérivé de $f$. Aprés un mouvement de $\Delta q$ qui provoque un changement de $\Delta\xi_m$ mesuré dans l'espace image, on a une approximation le long de $\Delta q$. Donc, on doit modifier le jacobien afin d'obtenir le modèle estimé plus précis, suivant les valeurs mesurées, par satisfaction d'une séquence de l'équation suivante

$$\Delta\xi_m = J_{k+1}\Delta q$$

ou

$$J_{k+1}^{+}\Delta\xi_m = \Delta q$$

L'expression du jacobien cherché est donné sous la forme

$$J_{k+1}^{+} = J_k^{+} + \frac{(\Delta q - J_k^{+}.\Delta\xi_m)\Delta\xi_m^T}{\Delta\xi_m^T.\Delta\xi_m} \tag{3.18}$$

### 3.6.4 Le mouvement du robot

Le but de notre système de contrôle visuel est de ramener les primitives de la position initiale $\xi_0$ à la position désirée $\xi^*$, on peut appliquer directement la loi de commande (3.15) aprés avoir estimé une forme optimale du jacobien de la fonction visuel-moteur. Pour des raisons de stabilité et

convergence, on divise le chemin entre $\xi_0$ et $\xi^*$ en $p$ séquences sous positions désirées $\xi_l^*$ appelées points de chemin

$$\xi_l^* = \xi_0 + \frac{l}{p}(\xi^* - \xi_0)$$

Le robot accomplit un mouvement pour atteindre la position désirée $\xi_l^*$, l'algorithme suivant génère le mouvement nécessaire

**Algorithme 3.4** *1. calculer la position désiré $q_k$ par la résolution de (3.15).*

*2. exécution de la commande $q_{k+1} = q_k + \Delta q_k$*

*3. Calculer le nouveau jacobien par l'équation (3.18).*

*4. répéter les étapes ci-dessus pour la sous position désirée.*

Pour atteindre la position désirée $\xi^*$, les étapes 1 à 3 sont répétées jusqu'à la convergence.

# Chapitre 4

# Simulation et application

## 4.1 Simulateur 3D pour un asservissement visuel 2D

La simulation est une étape indispensable à la réalisation de toute réalisation pratique d'asservissement. Elle permet de décrire le comportement du système asservi et de résoudre sans risque de déterioration du dispositif réel, les problèmes relatifs à la commande comme par exemple la stabilité et les performances.

C'est dans cette optique que nous avons élaboré un simulateur nous permettant ainsi de tester différentes approches et permettant à d'autres utilisateurs de cerner le problème d'asservissement visuel à moindre coût et sans avoir à disposer d'un robot.

**Remarque 4.1** *la simulation consiste à amener une cible, se présentant sous forme d'un disque noir, dans une position désirée dans l'image (cible immobile et mobile assurant la présence de la cible dans l'image au cours de l'asservissement). Cela reste pour l'application réel. Ceci concorde avec l'application réelle.*

Le simulateur conçu est la combinaison de trois sous systèmes :

### 4.1.1 Simulation du Robot [18]

La simulation du robot tient compte des dynamiques des axes. Ces dynamiques ont été identifiées préalablement sur le IR50p par la procédure décrite dans le chapitre 2 (avec l'hypothèse de découplage entre les différents axes du Robot). Le modèle géométrique (paramètres de Denavit Hartenberg modifiés) ainsi que le modèle cinématique (jacobien du robot IR50p) sont décrits dans l'annexe. La représentation graphique sous forme d'un squelette est réalisée grâce à la toolbox robotics [18] de Matlab, avec quelques modifications de fonctions pour y inclure la visualisation de l'objet cible ainsi que la prise en compte des paramètres DH modifiés, etc...

### 4.1.2 Simulation de la caméra

La caméra est modélisée par la transformation de la position de l'objet dans l'espace opérationnel en une primitive point qui décrit les coordonnées de l'objet dans l'image.Cela signifie que seules les primitives qui servent à l'asservissement sont simulées. La synthèse et le traitement de l'image ne sont donc pas réalisés en simulation. La modélisation de la caméra est basée sur la théorie de la projection perspective avec une distance focale variable, il en est de même pour le diamètre de la matrice CCD.

### 4.1.3 Simulation de la commande

La commande est un bloc fonction qui calcule les vitesses articulaires désirées à partir de la primitive image actuelle et de la primitive désirée, ce bloc peut être modifié afin de permettre à l'utilisateur de tester différentes approches d'asservissement.

### 4.1.4 Schéma général du simulateur (sur simulink)

Le simulateur que nous avons développé prend en compte différents paramètres ; il est possible de choisir la position désirée des primitives, la trajectoire de mouvement de l'objet cible, le gain de commande, et permet la visualisation des différents signaux du système asservi comprenant ces positions et ces vitesses articulaires, les primitives et la trajectoire de l'objet dans l'image ainsi qu'une animation du squelette du robot et de la cible.

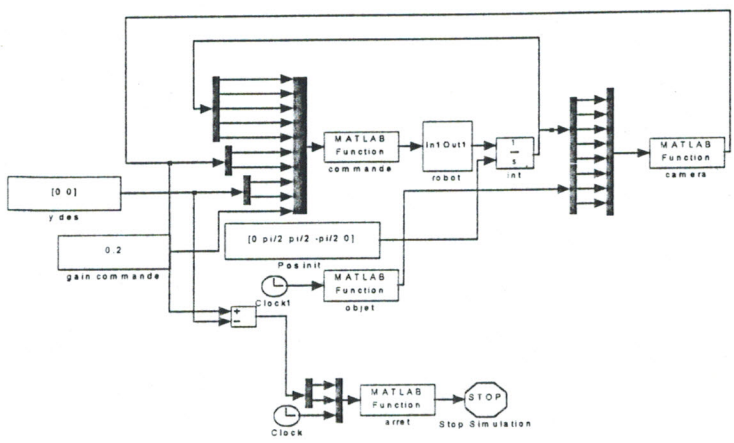



Figure 4.1 : (a) Simulateur de l'asservissement visuel 2D

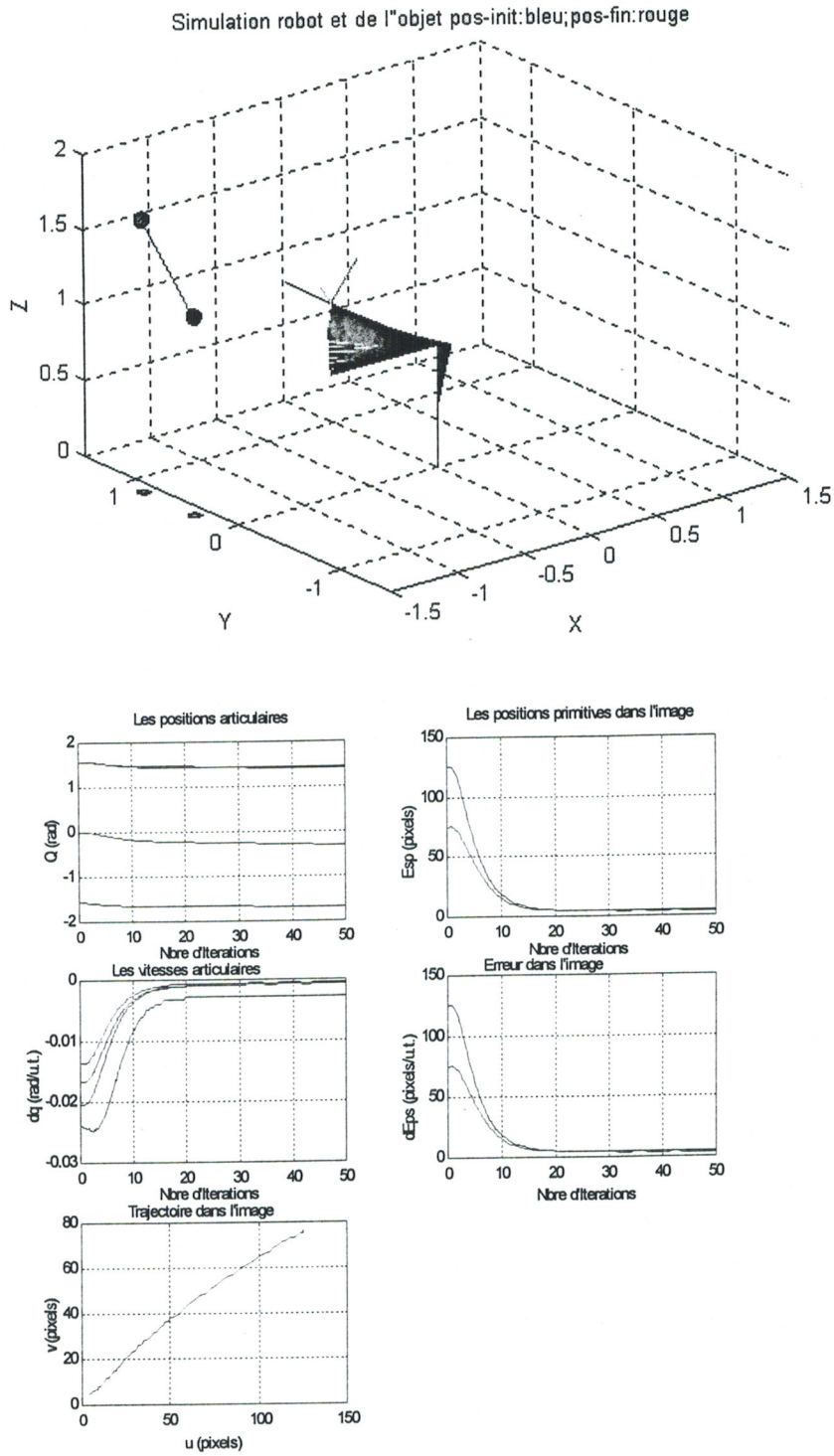



Figure 4.1 : (b) Resultats de simulation

### 4.1.5 Résultats de Simulation des différentes approche

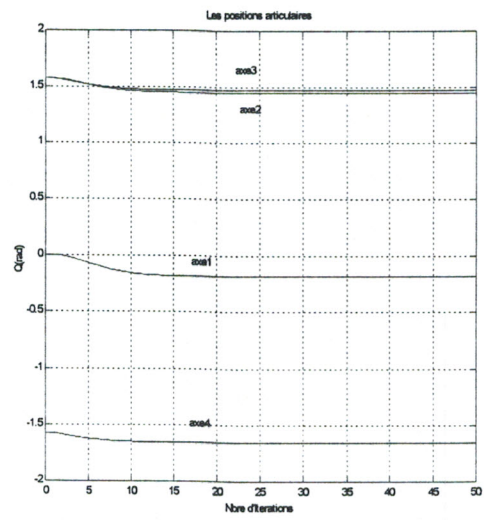
Afin de vérifier la validité de l'approche 2D classique, ainsi que celle de l'estimation du jacobien image (fonction visuel-moteur), nous présentons ici les réponses des vitesses articulaires, l'erreur des primitives dans l'image, les composantes 2D dans l'image et les positions articulaires, ainsi que la trajectoire dans l'image et du squelette robot en 3D, pour un positionnement d'une cible à une position désirée. La première approche est basée sur le calcul itératif du jacobien image, la deuxième pour un jacobien calculé aux primitives désirées dans l'image ainsi que la distance objet-caméra, et la dernière pour l'estimation du jacobien par la méthode de Broyden.

La position initiale des différentes articulations $qi = \begin{bmatrix} 0 & \frac{\pi}{2} & \frac{\pi}{2} & -\frac{\pi}{2} & 0 \end{bmatrix}$, les coordonnées de l'objet vu par la caméra (inclus dans le champ de vision) pour cette configuration initiale du robot est $\xi_0 = \begin{bmatrix} 125 & 75 \end{bmatrix}^T$, la référence à atteindre dans l'image étant $\xi_d$ ou $\xi^* = \begin{bmatrix} 0 & 0 \end{bmatrix}$ (centre de l'image). Les figures ci-dessous présentent, pour differentes valeurs du gain de commande, la commande $dq$, le changement des positions articulaires $q$, le changement des coordonnées des primitives dans l'image et la trajectoire de l'objet dans l'image.

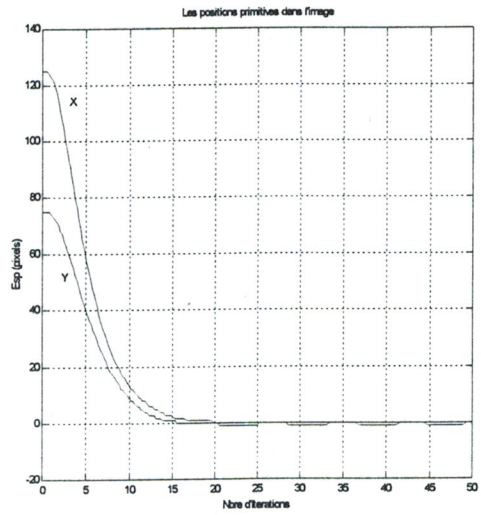
Trois techniques ont été simulées pour tester cette approche 2D classique :

- Calcul du jacobien image aprés chaque itération.
- Calcul du jacobien pour la position désirée.
- Estimation du jacobien avec la méthode de Broyden.

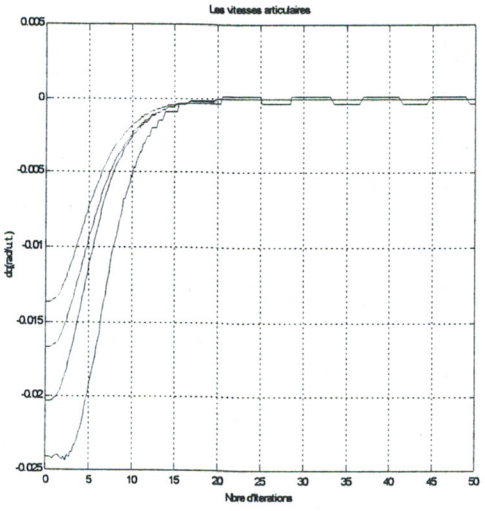
a) Dans cette simulation le jacobien image est calculé à chaque itération pour la position des primitives dans l'image.



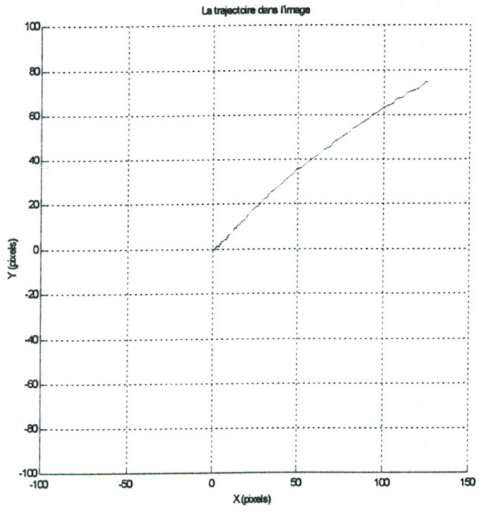

(a)




(b)




(c)




(d)

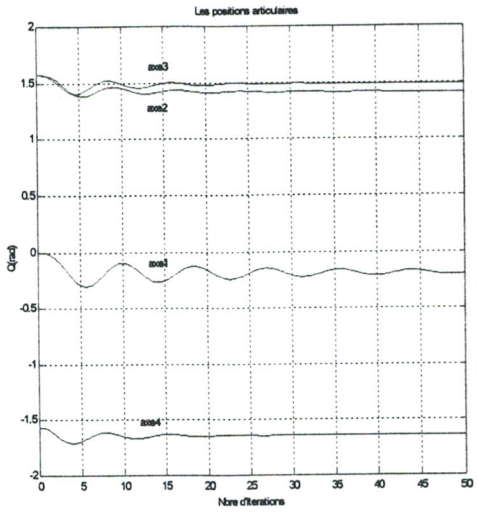
Figure 4.2 : Résultats de simulation avec un gain $\lambda = 0,2$

(a) Les positions articulaires

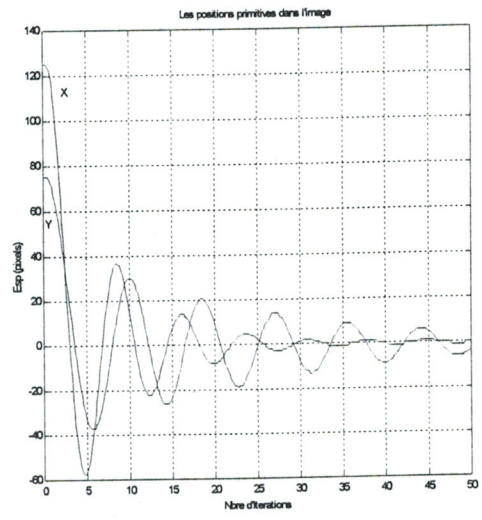
(b) Les positions primitives dans l'image

(c) Les vitesses articulaires

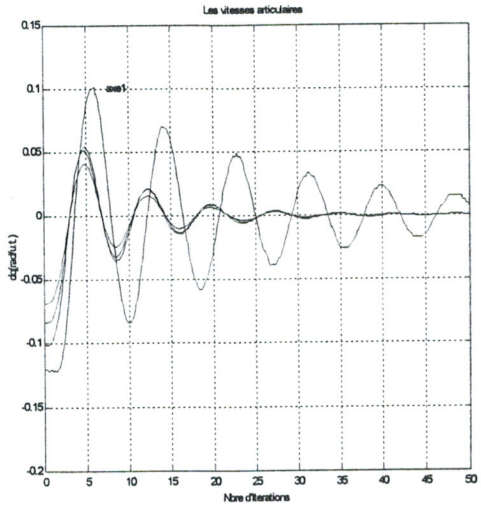
(d) Trajectoire dans l'image



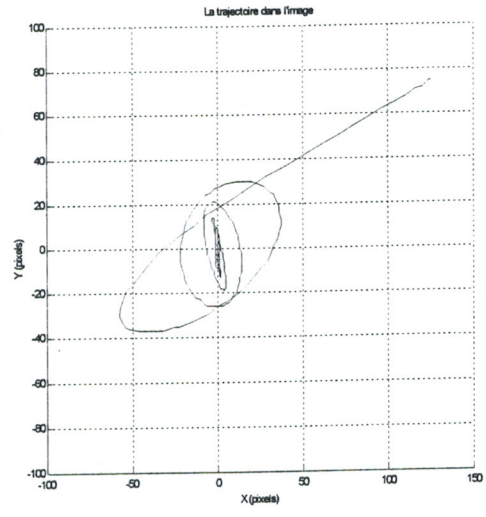

(a)




(b)




(c)




(d)

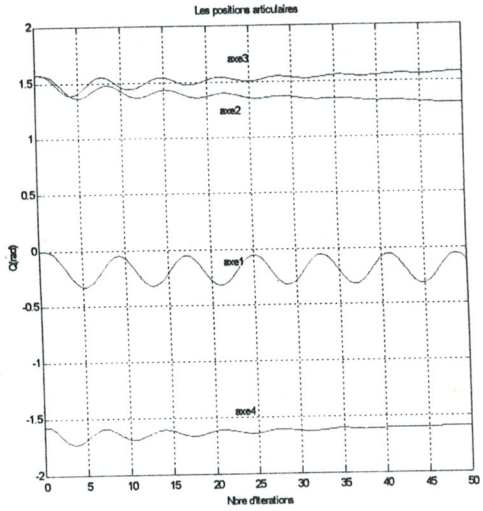
Figure 4.3 : Résultats de simulation avec un gain $\lambda = 1$

(a) Les positions articulaires

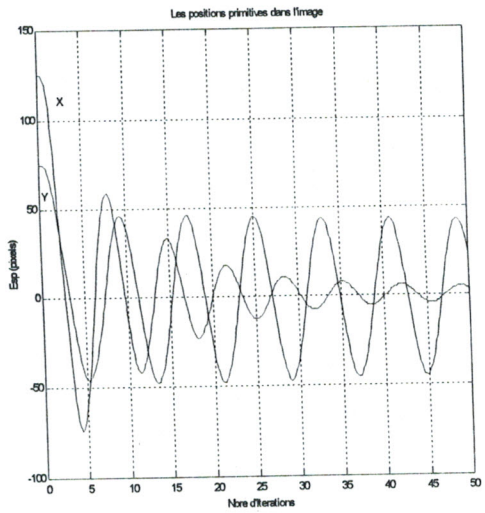
(b) Les positions primitives dans l'image

(c) Les vitesses articulaires

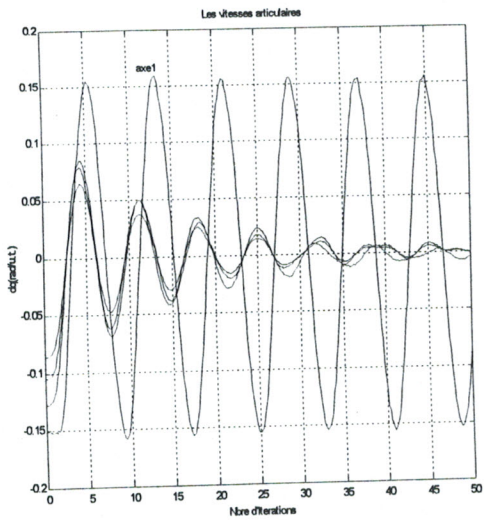
(d) Trajectoire dans l'image



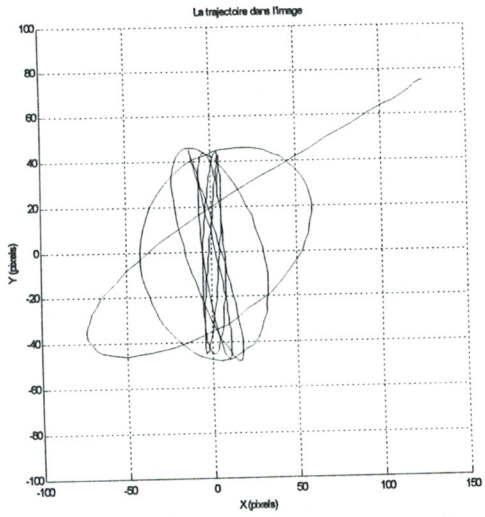

(a)




(b)




(c)




(d)

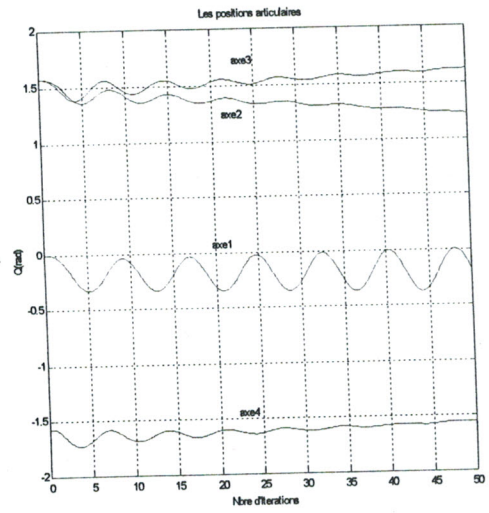
Figure 4.4 : Résultats de simulation avec un gain $\lambda = 1,25$

(a) Les positions articulaires

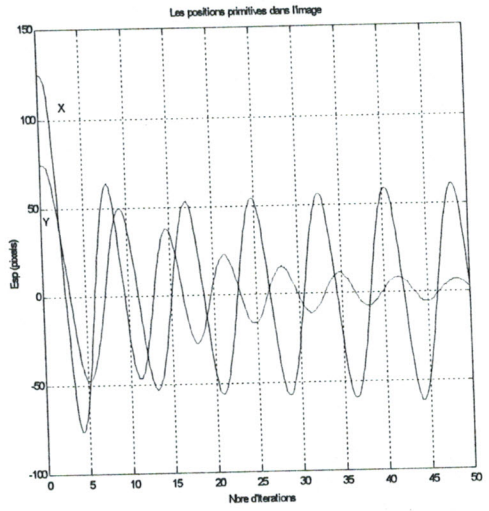
(b) Les positions primitives dans l'image

(c) Les vitesses articulaires

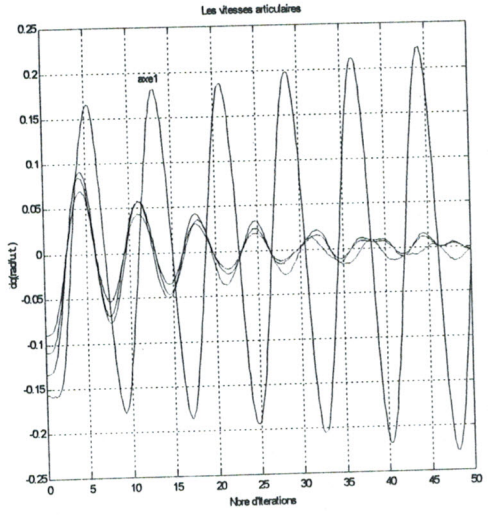
(d) Trajectoire dans l'image



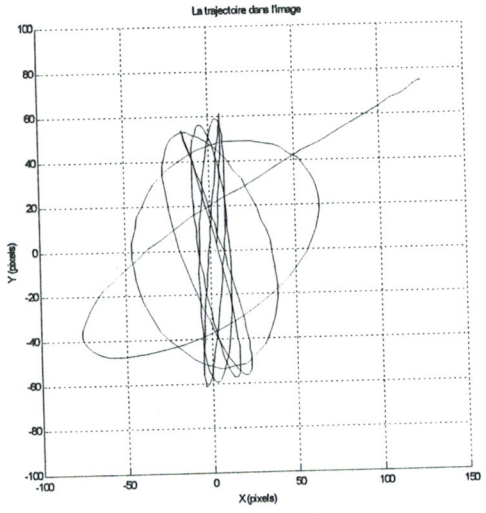

(a)




(b)




(c)




(d)

Figure 4.5 : Résultats de simulation avec un gain $\lambda = 1,3$

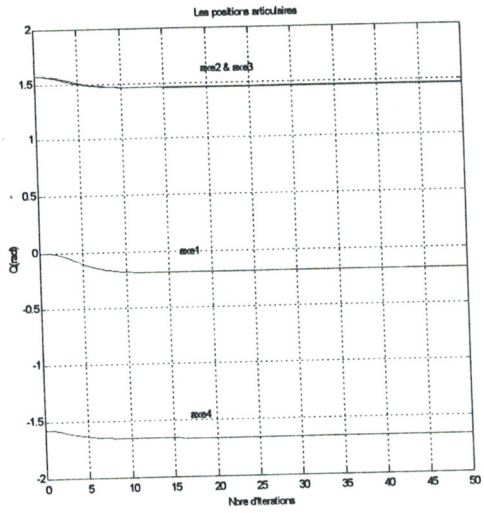
(a) Les positions articulaires

(b) Les positions primitives dans l'image

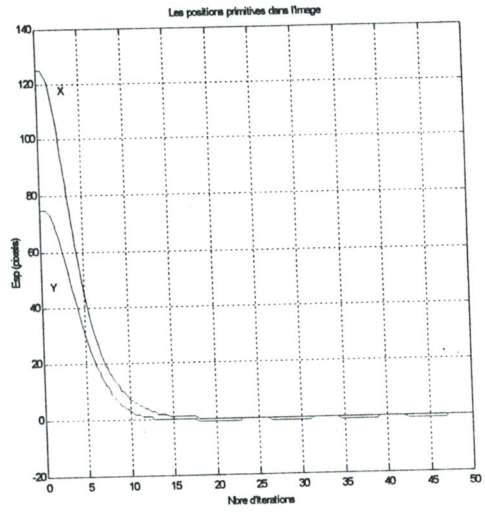
(c) Les vitesses articulaires

(d) Trajectoire dans l'image

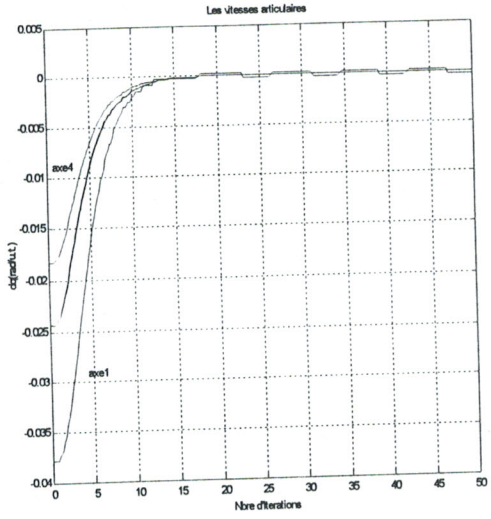
b) Dans cette simulation, le jacobien image est calculé une seule fois à la position désirée dans l'image.



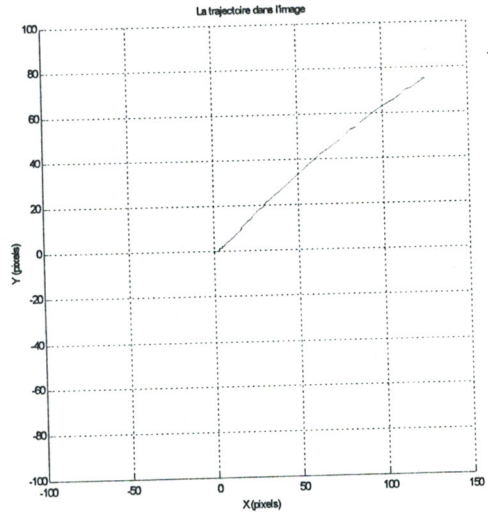

(a)




(b)




(c)




(d)

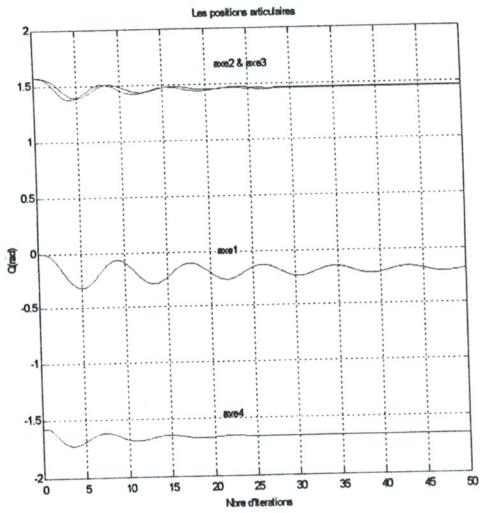
Figure 4.5 : Résultats de simulation avec un gain $\lambda = 0,2$

(a) Les positions articulaires

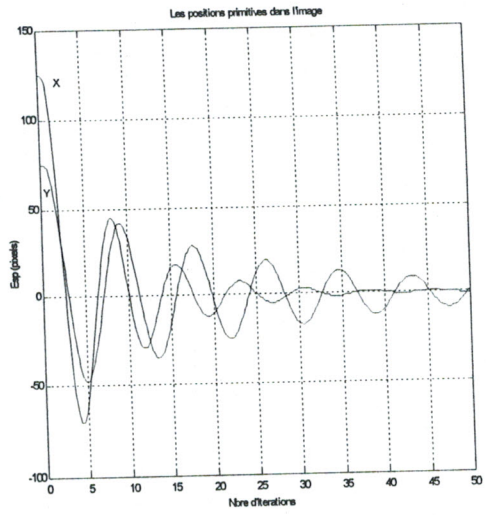
(b) Les positions primitives dans l'image

(c) Les vitesses articulaires

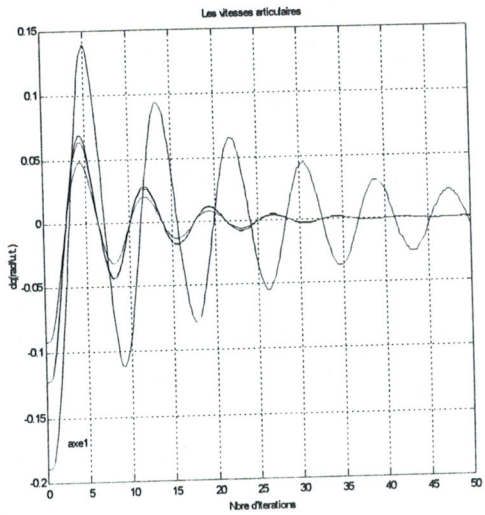
(d) Trajectoire dans l'image



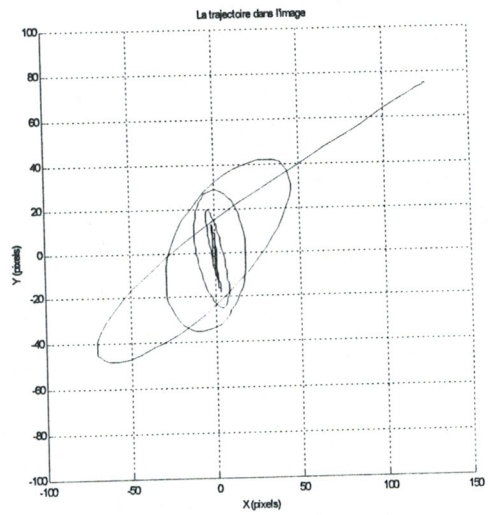

(a)




(b)




(c)




(d)

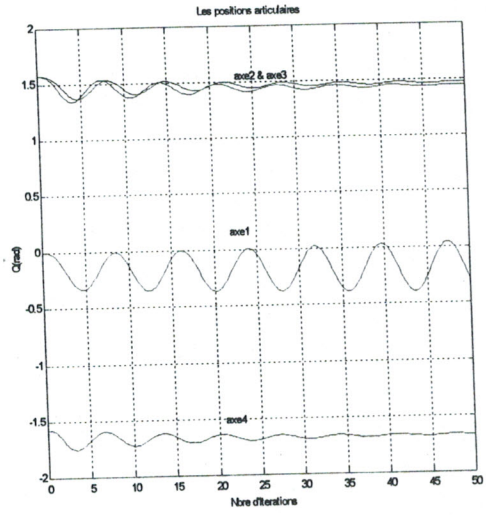
Figure 4.6 : Résultats de simulation avec un gain $\lambda = 1$

(a) Les positions articulaires

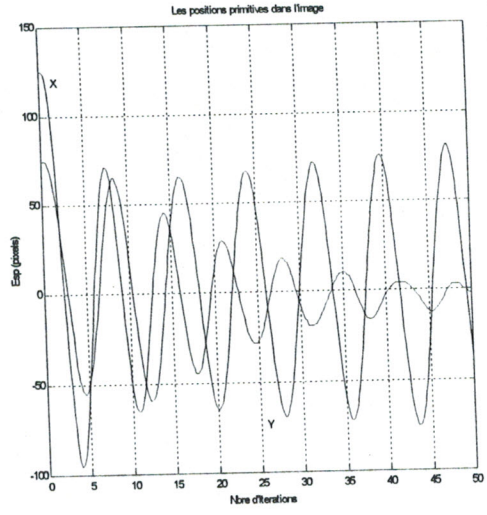
(b) Les positions primitives dans l'image

(c) Les vitesses articulaires

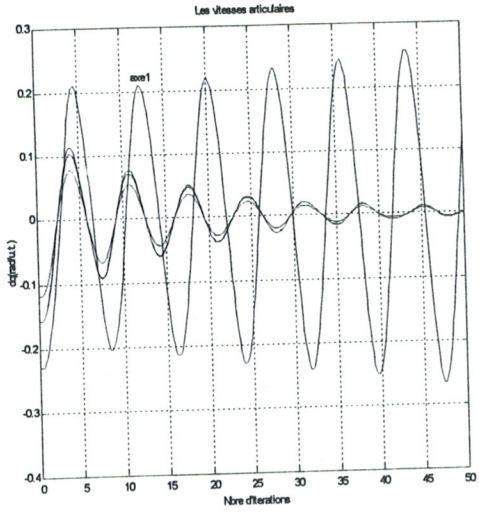
(d) Trajectoire dans l'image



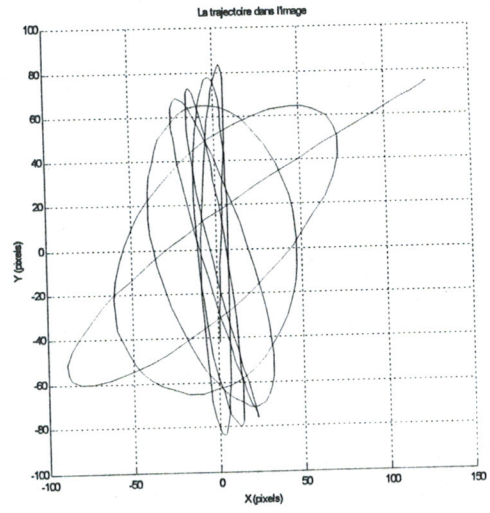

(a)




(b)




(c)




(d)

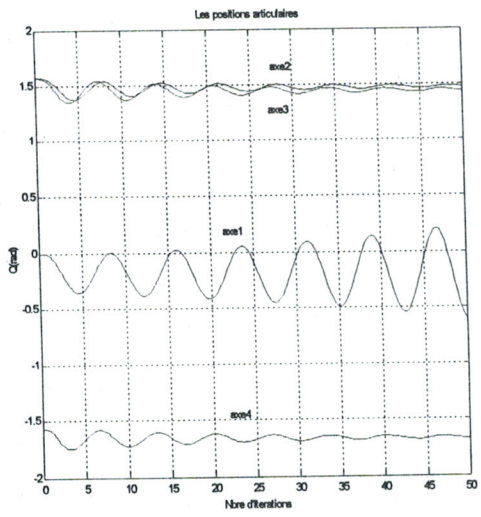
Figure 4.7 : Résultats de simulation avec un gain $\lambda = 1,25$

(a) Les positions articulaires

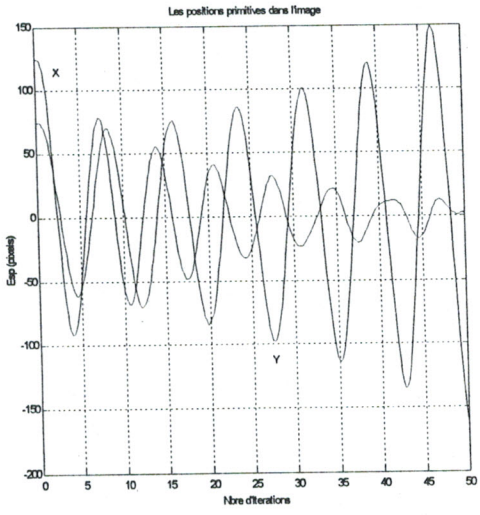
(b) Les positions primitives dans l'image

(c) Les vitesses articulaires

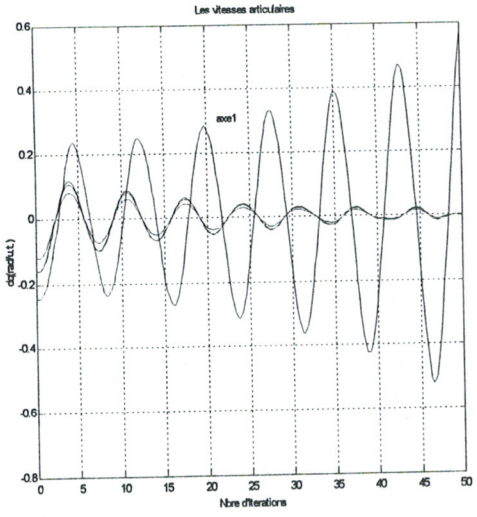
(d) Trajectoire dans l'image



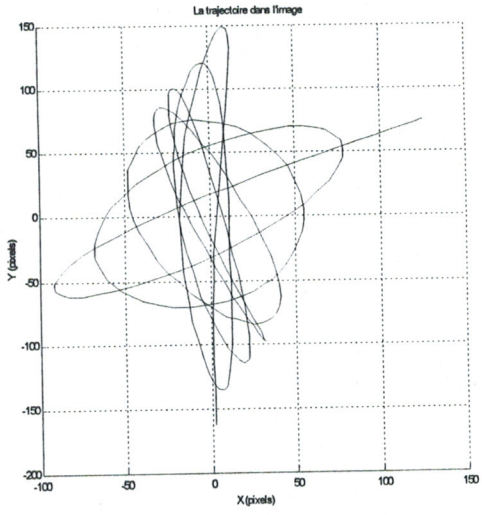

(a)




(b)




(c)




(d)

Figure 4.8 : Résultats de simulation avec un gain $\lambda = 1,3$

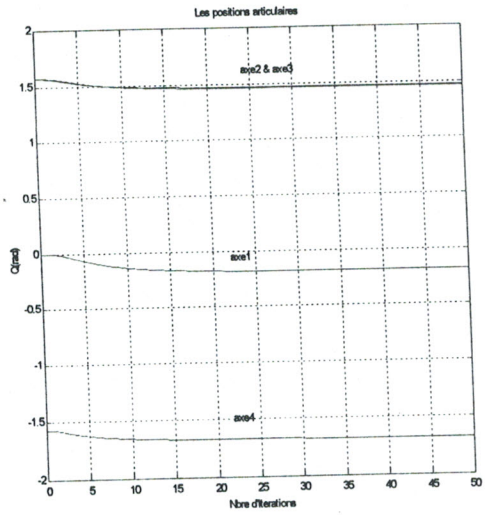
(a) Les positions articulaires

(b) Les positions primitives dans l'image

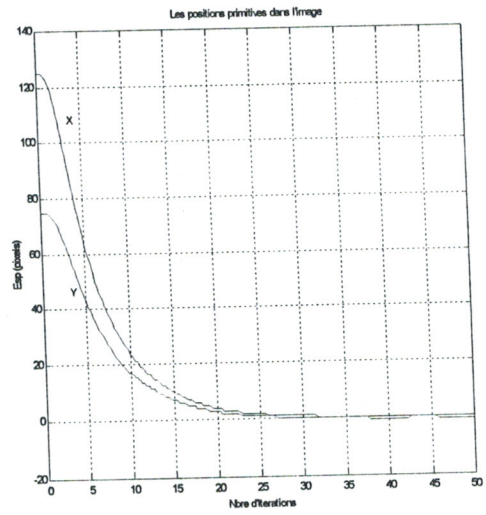
(c) Les vitesses articulaires

(d) Trajectoire dans l'image

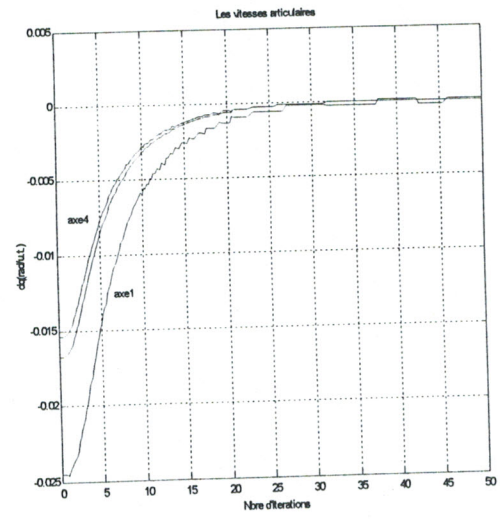
c) Dans cette simulation, le jacobien image est estimé par la méthode de BROYDEN DUALE pour que la fonction visuel-moteur soit optimale.



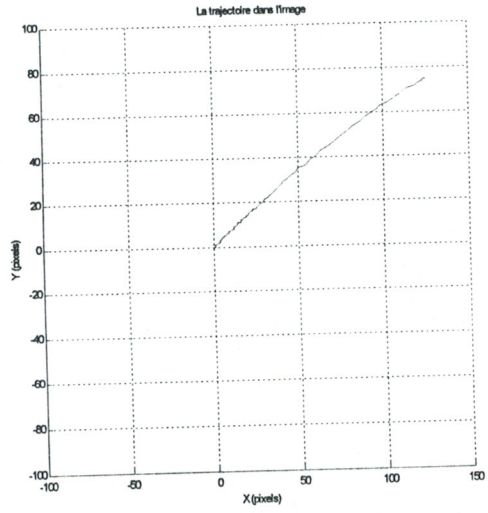

(a)




(b)




(c)




(d)

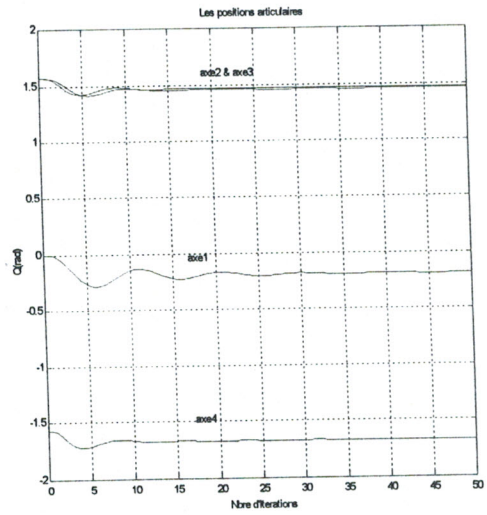
Figure 4.9 : Résultats de simulation avec un gain $\lambda = 0,2$

(a) Les positions articulaires

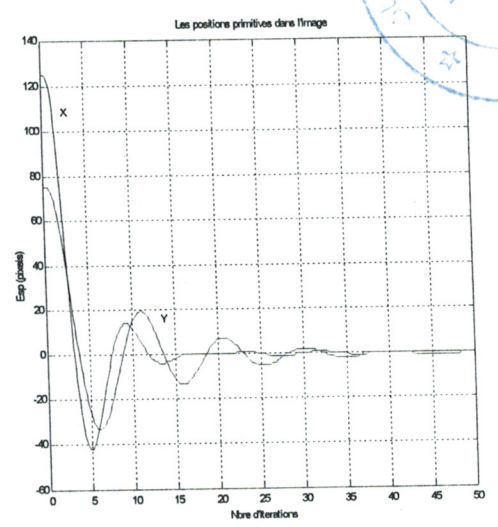
(b) Les positions primitives dans l'image

(c) Les vitesses articulaires

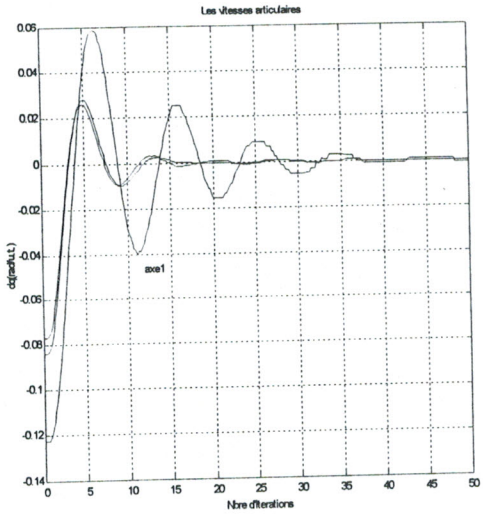
(d) Trajectoire dans l'image



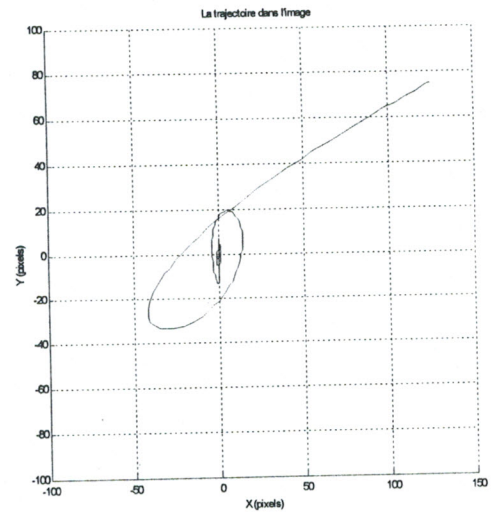

(a)




(b)




(c)




(d)

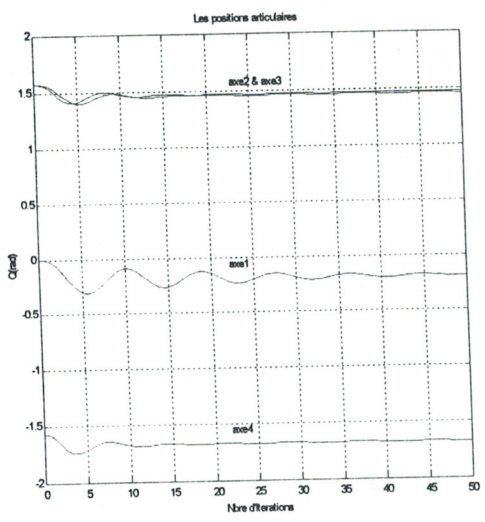
Figure 4.10 : Résultats de simulation avec un gain $\lambda = 1$

(a) Les positions articulaires

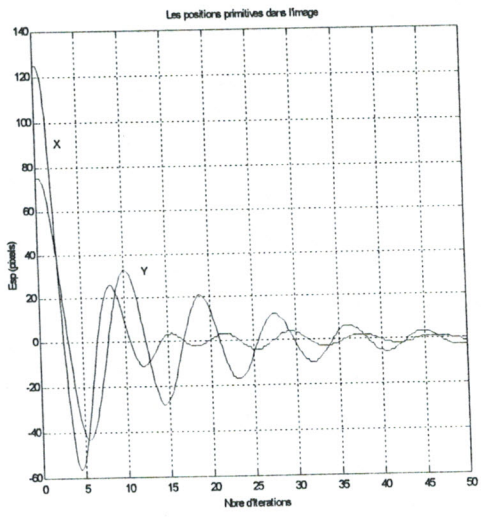
(b) Les positions primitives dans l'image

(c) Les vitesses articulaires

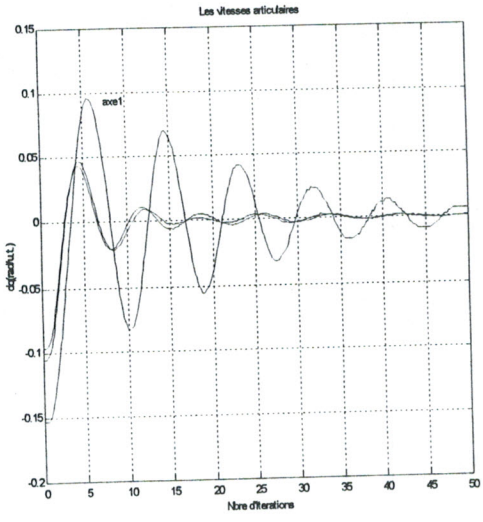
(d) Trajectoire dans l'image



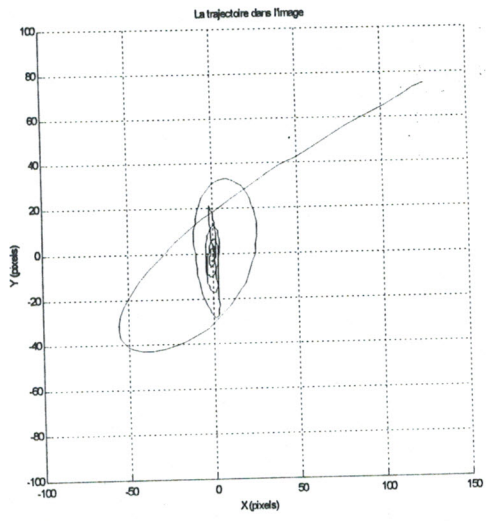

(a)




(b)




(c)




(d)

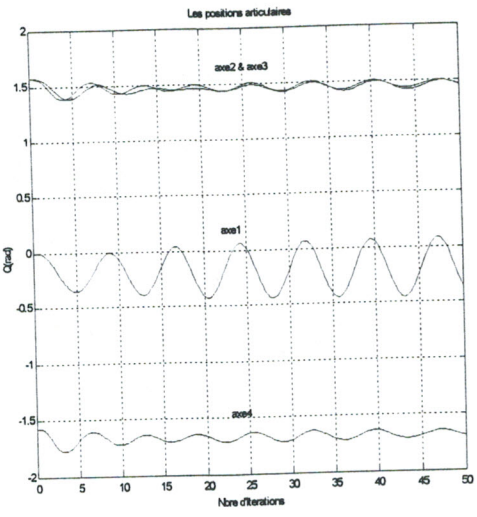
Figure 4.11 : Résultats de simulation avec un gain $\lambda = 1,25$

(a) Les positions articulaires

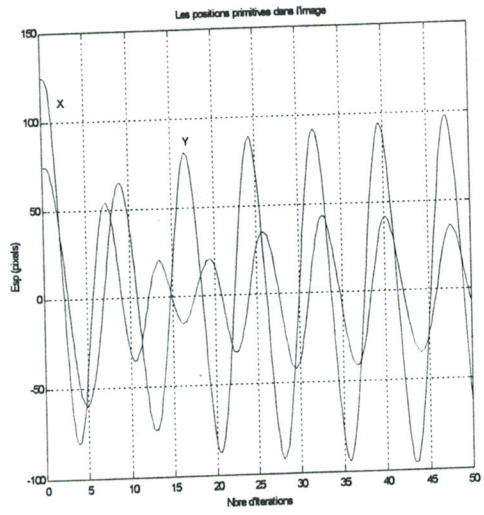
(b) Les positions primitives dans l'image

(c) Les vitesses articulaires

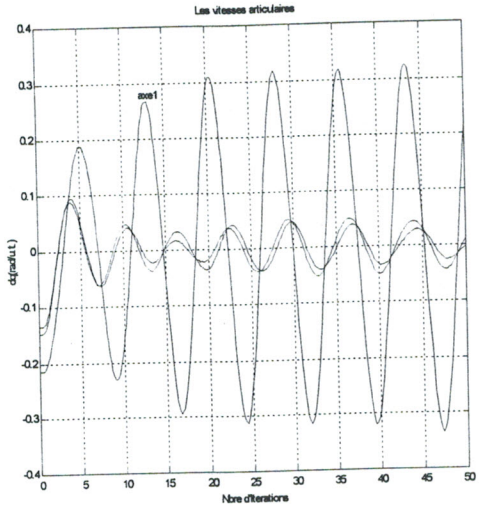
(d) Trajectoire dans l'image



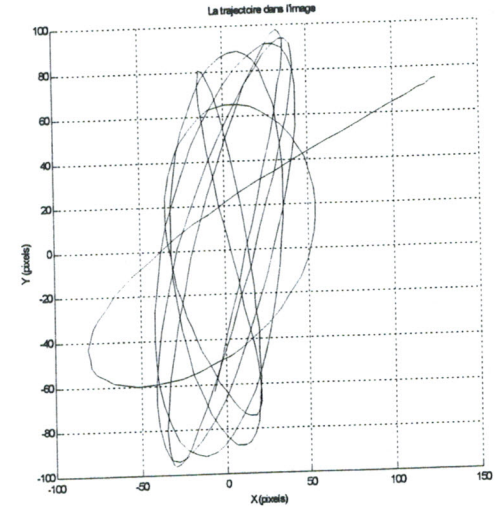

(a)




(b)




(c)




(d)

Figure 4.12 : Résultats de simulation avec un gain $\lambda = 1,57$

(a) Les positions articulaires

(b) Les positions primitives dans l'image

(c) Les vitesses articulaires

(d) Trajectoire dans l'image

Le gain de la commande a un effet direct sur la rapidité du système asservi, plus ce gain est important, plus la convergence est rapide (les figures 4.x.(a) et (b)).

Mais l'augmentation du gain a aussi un effet déstabilisant, et comme nous pouvons le voir sur les résultats des simulations, des gains élevés provoquent des oscillations (les figures 4.x.(c) avec un gain de $1,25$) voire même l'instabilité (les figures 4.x.(c) avec un gain $\geqslant 1,3$ pour les deux premières approches et un gain $\geqslant 1,75$ pour la dernière approche).

Notre but est évidemment d'assurer une bonne performance du système tout en le maintenant

stable et non oscillant, il nous faut donc faire un compromis.

Les approches classiques simulées donnent des résultats très semblables ce qui favorise le choix du jacobien calculé aux primitives désirées. En effet pour la première approche il est nécessaire de calculer le jacobien à chaque itération ce qui se révele très contraignant lors d'une application en temps réel, contrainte qui n'est pas présente lorsque le jacobien est calculé une seule fois à la position désirée pour la deuxième approche.

La dernière approche estime l'inverse du jacobien visuel-moteur à chaque itération (on line) par la méthode de Broyden duale. Dans cette approche le modèle de la caméra et la cinématique du robot ne sont pas pris en compte, donc une plus grande simplicité d'implémentation (ou programmation) sur un système réel que les deux premières approches.

## 4.2 Application sur un bras manipulateur IR50p

Après avoir validé les différentes approches vues dans le chapitre 3 par un simulateur, et avoir vérifié la stabilité et le temps de réponse, nous avons appliqué ces techniques à un système réel (Figure 4.2). composé d'un bras manipulateur IR50p, d'une caméra CCD fixé sur l'éffecteur et l'ensemble piloté par un PC.

Les problèmes que nous avons rencontrés lors de l'application pratique sont dûs essentiellement à la nature didactique du bras manipulateur sur lequel nous avons travaillé. Celui-ci, n'est pas un système ouvert. Nous ne disposons que de peu d'informations sur sa constitution interne des procédures de communication avec un P.C. Les contrôleurs internes (boucle de contrôle bas niveau) ne sont pas accessibles ni même connus. Ce qui limite notre champ d'action au niveau de la commande à des consignes de position articulaire..

De plus, le temps de réponse pour les capteurs de position au niveau des articulations est grand (à l'ordre de 50 ms).

Les résultats de simulation, nous ont conduit à opter pour l'approche du jacobien estimé par la méthode de Broyden duale, vu qu'elle est moins consommatricee de temps de calcul et ne necessite ni le modèle cinématique du robot, ni le modèle de la caméra. Nous l'avons donc appliquée.

### 4.2.1 Schéma du système de contrôle visuel

La figure 4.2 donne un aperçu de la structure du système expérimental.

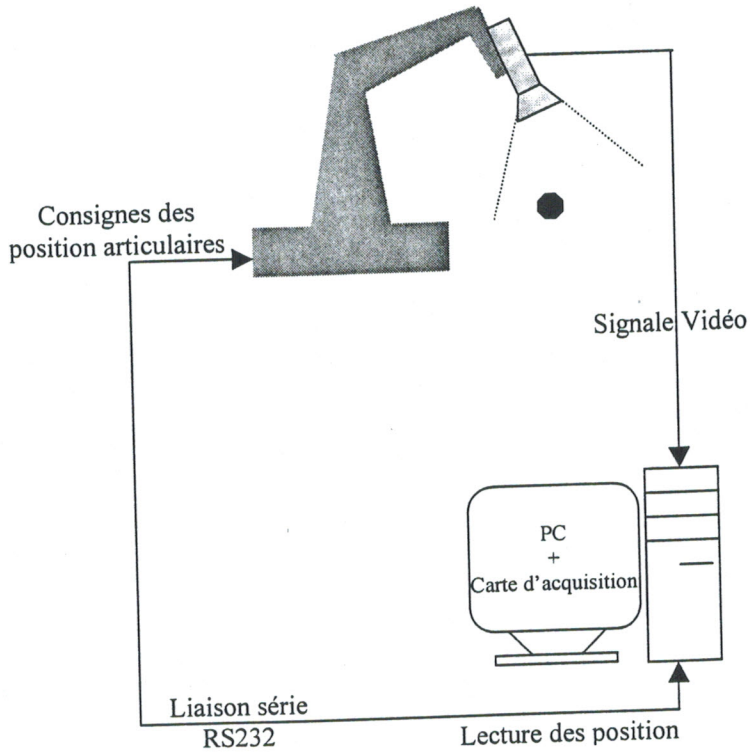



Figure 4.13 : Schéma de la plate-forme expérimental

Nous avons choisi une architecture dans laquelle tous les traitements (traitements d'image et contrôle de la boucle) sont réalisés par la même machine afin de limiter au maximum les délais dus au transfert des données.

### 4.2.2 Le capteur de vision

Le capteur visuel utilisé pour les manipulations est une caméra CCD Couleur CCTV permettant d'acquérir 25 images entrelacées par seconde, c'est à dire 50 trames par seconde. Sa résolution est de 512x582 pixels. L'intégration de la lumière pouvant être effectuée à chaque trame, ce capteur permet de fournir une mesure à la cadence de 50Hz. L'objectif utilisé pour les manipulations a une longueur focale f=8mm.

### 4.2.3 Le système d'acquisition d'images

Le système d'acquisition d'image est constitué d'un ordinateur de type PC (PIII 1GHz) équipé d'une carte d'acquisition TV MIRO PC TV Tuner au format PCI et d'une caméra CCD.

La lecture du buffer de la carte d'acquisition d'image via le driver sous Windows est exposée dans l'annexe **D**.

### 4.2.4 Traitements d'image

La simplicité de l'objet contrôlé ainsi que son envirennement assure un temps de traitement faible avec moins d'erreur, ce qui assure la robustesse du système. A cet effet, 2 types de détection d'objet sont implémentés. Dans le premier, nous avons défini l'objet par un disque plein noir sur un plan blanc. Les primitives extraites de l'image sont les coordonnées du centre de gravité de l'objet. Le centre du disque s'obtient avec une détection des sommets du disque balayant de haut en bas, et de gauche à droite, les coordonnées détectées déterminent le centre de gravité du disque. (Voir figure 4.3). Ce calcul de centre de gravité est peu coûteux en temps et assure un traitement en temps réel pour notre application.

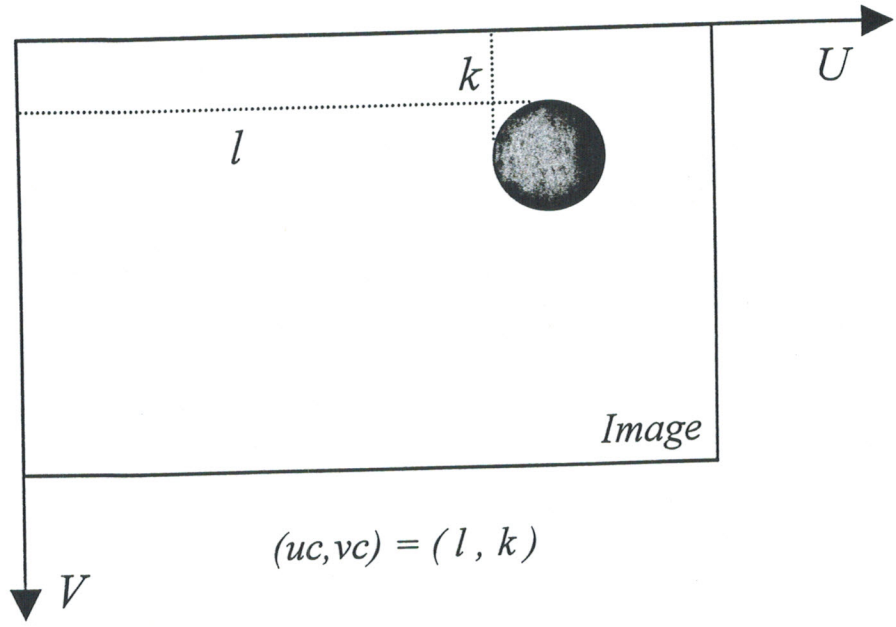



Figure 4.14 : Calcule du centre de gravité d'un disque

L'extraction des primitives visuelles est basée sur des images binarisées (Noir et Blanc), la lecture de l'image acquise est faite directement en niveaux de gris à partir du buffer vidéo. L'image lue est ensuite binarisée.

Pour le deuxième type, nous avons défini l'objet par la détection de la couleur. celle-ci doit être unique dans l'environnement de travail afin d'éviter la ressemblance des niveaux de couleur RVB. L'objet est détecté par un double clique droit par la souris. Pour cela, nous avons implémenté un algorithme qui détecte l'objet partiellement.

Soit un objet à une couleur séparable dans l'environnement de travail, la couleur moyenne d'une surface d'objet centrée par le point de double clique, la taille de cette surface est définie par l'utilisateur au préalable (dépendante de la taille de l'objet), alors que la couleur moyenne est la moyenne des niveaux de couleur de tous les pixels de la surface.

Un filtrage de couleur pour générer une image qui ne contient que la surface définie de l'objet. Les coordonnées du centre de la surface (le point de double clique) définissent les primitives image.

### 4.2.5 Le système multitâche [50]

Le système multitâche de Windows est destiné à réaliser l'interface de bas niveau avec le matériel

Le véritable multitâche se passe au niveau des threads, c'est à dire que plusieurs threads sont exécutés de manière concurrente. Sous Windows 95/98, il faut faire appel à la technique du temps partagé. Dans ce principe, de courts segments de temps sont alloués successivement aux différentes tâches qui sont exécutées cycliquement. Chaque tâche a un temps alloué de 20 ms, avec une priorité soit statique ou dynamique au fil d'ordonnancement.

Sous Windows 95/98, l'API Win32 propose des fonctions pour la création des tâches avec une priorité définie par le programmeur, cela nous a permis de définir l'asservissement visuel sous un thread avec une priorité élevée, afin qu'il obtienne le processeur dès qu'il en a besoin.

### 4.2.6 Organigramme de l'asservissement

L'organigramme de l'asservissement (figure 4.15) représente le fonctionnement du mécanisme de notre contrôleur.

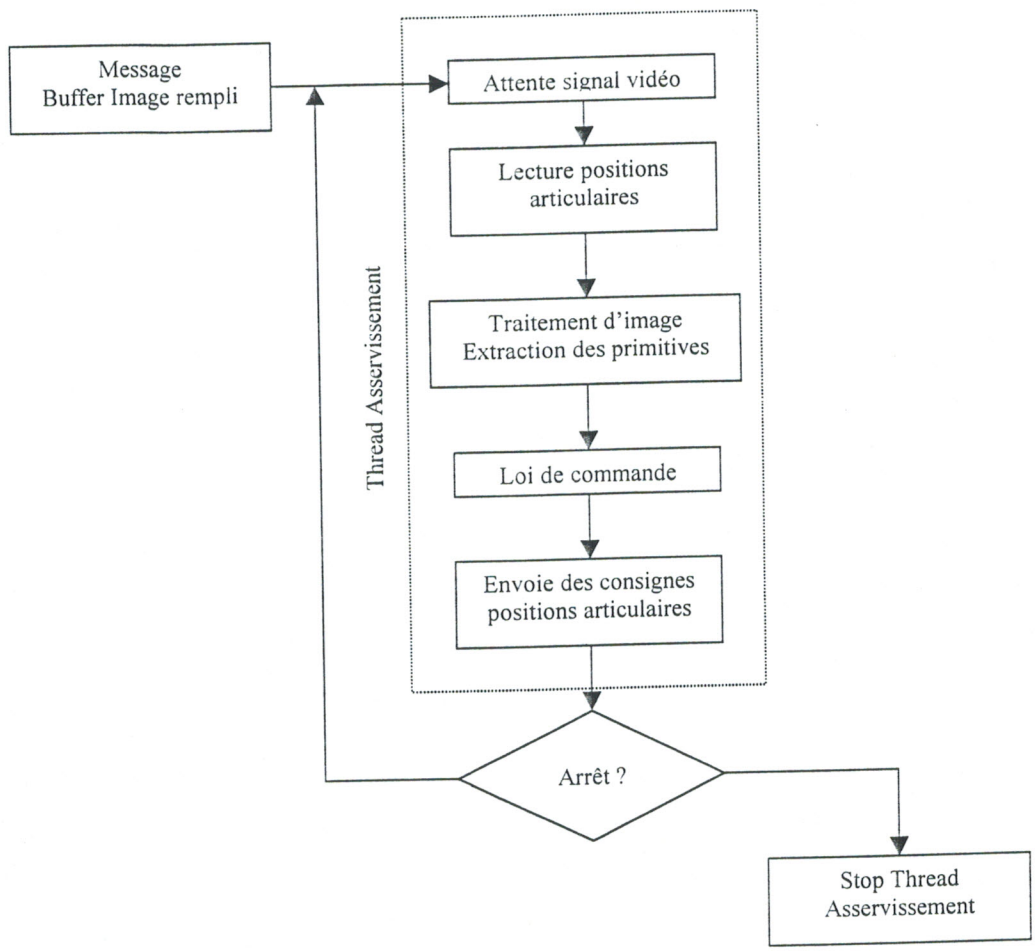



Figure 4.15 : Organigramme de l'asservissement

### 4.2.7 Expérience

L'application de la commande donne des résultats satisfaisants malgré une conaissance limitée du modèle du robot et de son contrôleur bas niveau fourni par le constructeur. Nous avons imposé au système asservi de placer l'objet à une position arbitraire désirée dans l'image.

Nous nous sommes limités à l'utilisation de deux axes pour éviter les problèmes de redondance. Notre choix s'est porté sur les articulations 1 et 4 pour contrôler respectivement les positions X et Y de la cible dans l'image, pour justifiier le choix de ces axes, précisons d'abord que le seul axe permettant le contrôle de la position X de l'objet dans l'image est l'axe 1, et l'axe 4 fournit les meilleurs résultats comparé, à ceux obtenus avec les axes 2 et 3. Enfin précisons que les mouvements de l'axe 5 affectent les positions X et Y de l'objet dans l'image et peut causer des singularités conduisant à des dégénerescences.

Le suivi de la consigne se révele relativement bon malgré la présence d'erreurs statiques sur les

deux axes. Cela est dû essentiellement à la sensibilité numérique du jacobien estimé, qui dépend du pas différentiel de l'axe et de la distance entre la caméra et le plan observé sur lequel se trouve l'objet.

L'autre constatation que nous pouvons faire concerne les oscillations assez importantes apparaissant sur certains axes lors des mouvements notamment, l'axe 4. Celles ci sont dues apparemment à l'inertie des axes du robot et la présence de frottements secs assez importants. Ces oscillations nous conduisent à croire que le robot pourrait même être à la limite de la stabilité.

Aussi, nous avons remarqué quelques fois qu'il est impossible de commander le robot à des cadences rapides. En effet celui-ci répond aux consignes avec un certain retard voire jusqu'à ne plus répondre du tout. Cela est dû surement à un problème de tampon au niveau du controleur interne. Ce tampon est prevu pour stocker les differentes commandes envoyées à travers l'interface. A un certain moment, il doit se saturer.

Deux facteurs influent de manière significative sur les performances du système asservi, il s'agit du gain de la commande en position et la vitesse de mouvement du robot qui ont pour effet d'augmenter sa rapidité lorsqu'ils sont élevés, mais d'un autre côté des valeurs trop élevés de ces deux facteurs provoquent une détérioration.

Les figures 4.16.(a) et 4.16.(b) présentent les résultats d'une application réelle de faire parcourir un objet quelconque d'une couleur unique (rouge dans (a) et bleu dans (b)) choisie dans l'environnement vers une position désirée dans l'image. Le filtrage est par couleur. Pour un objet disque noir sur un fond claire, l'application est présentée dans les figures 4.17.

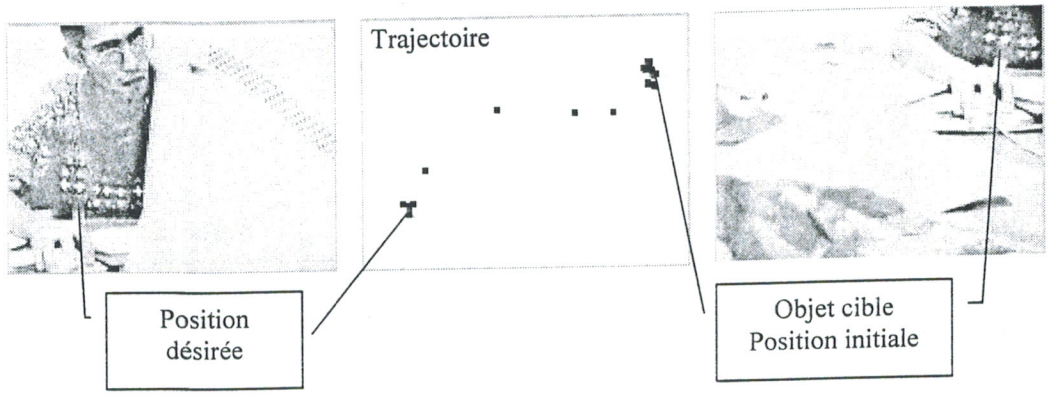



Figure 4.16.(a) : parcourt d'un objet dans une position désirée dans l'image (filtrage par couleur)

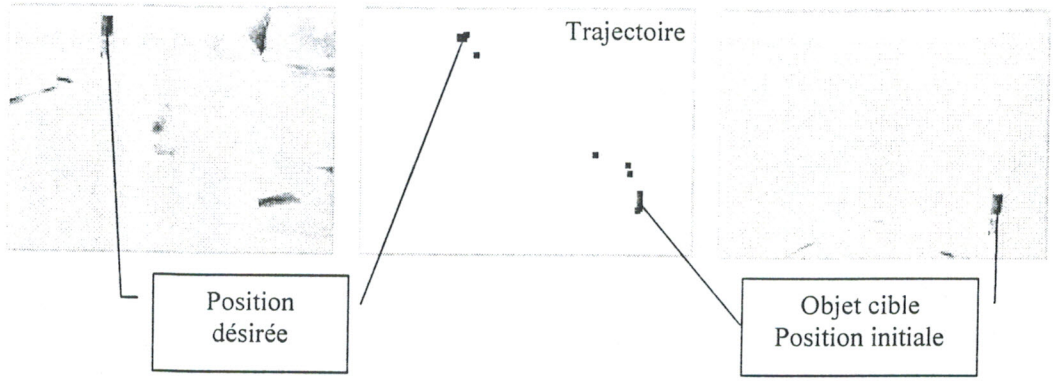



Figure 4.16.(b) : parcourir un objet dans une position désirée dans l'image (filtrage par couleur)

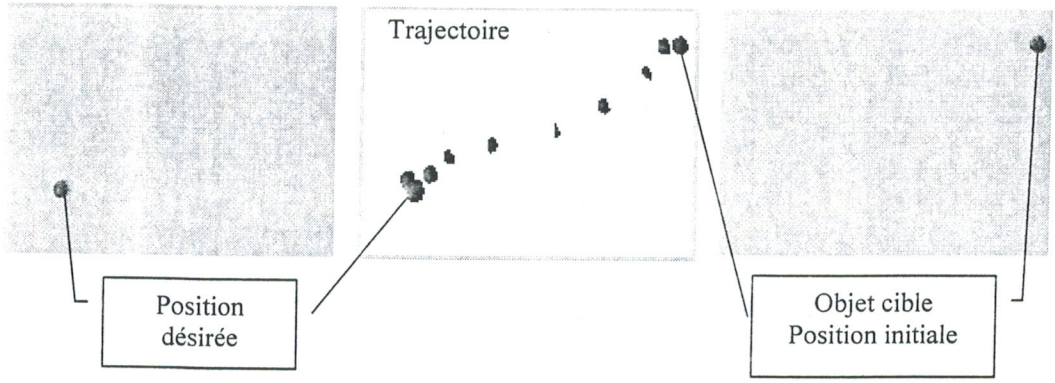



Figure 4.17 : parcourt d'un objet dans une position désirée dans l'image (objet noir sur fond claire)

Cet exemple d'application peut être facilement utilisé, pour l'apprentissage de tâche à effectuer par le robot. En effet, il suffit de lui indiquer face à un pointeur de couleur discriminable, la trajectoire à parcourir et d'enregister simultanément les commandes envoyées aux controleurs. Cet enregistrement pourrait être alors reutilisé pour que la tâche soit répétée.

# Conclusion générale

Ce travail nous a permis de valider les différentes approches théoriques de l'asservissement visuel d'un bras manipulateur. Grâce à un simulateur que nous avons développé, nous avons testé les différentes approches d'asservissement visuel 2D.

- Calcul du jacobien image aprés chaque itération.
- Calcul du jacobien pour la position désirée.
- Estimation du jacobien avec la méthode de Broyden.

Les résultats de simulations nous ont permis de faire un choix sur la meilleure approche à utiliser dans le cas d'un Robot réel disponible dans notre laboratoire, les critères de choix ont été l'insensibilité au modèle pour une meilleure robustesse et le temps de calcul pour une application en temps réel. L'approche jacobien estimé nous a donné, dans ce sens, de très bons résultats.

Le fait que le robot utilisé, n'était pas de type professionnel, a néanmoins fait dégrader les performances (oscillations, imprécision).

Le système Caméra- Robot-PC, que nous avons mis en œuvre pourra être utilisé et développé grâce un logiciel que nous avons développé. Celui ci permet à travers une interface utilisateur, d'assigner des consignes, d'afficher les positions des axes, de lancer la commande et d'afficher les images captées par la caméra en temps réel.

Ce travail possède de nombreuses extensions possibles, la première est bien entendu l'implémentation d'une commande bas niveau qui prend en compte les dynamiques du robot, il sera ainsi aussi possible d'utiliser un modèle plus complet dans le simulateur, celui que nous avons utilisé ne prenant pas compte les forces de couplage des axes et les frottement secs.

Il est aussi envisageable d'utiliser une seconde caméra extérieure qui permettrait de détecter la position absolue de l'objet dans le repère robot et ainsi détecter l'objet lorsqu'il est hors du champ de vision de la caméra embarquée, il sera aussi possible alors d'éviter les singularités du robot.

Une autre perspective est l'implémentation d'un procédé de prédiction de mouvement de l'objet cible.

L'introduction de primitives plus riches permettrait d'effectuer des suivis plus précis et de commander l'orientation de la caméra embarquée pour choisir un angle de vue de l'objet. Il sera ainsi possible d'inclure un suivi de profilé en plus du suivi d'une cible.

# Bibliographie


[1] A. Ait Mohamed, "Commande Dynamique des Robots Redondants dans l'Espace Operationnel", Thèse de Doctorat, Université de Nantes, 1995.

[2] Andreas, "Closing The Loop Between Articulated Motion And Stereo Vision : A Projective Approach", Thèse de Doctorat, INPG, 2001.

[3] N. Andreff, "Asservissement Visuel à Partir de Droites Auto-Etalonnage Pince-Camera", Thèse de Doctorat, Institut National Polytechnique de Grenoble, 1999.

[4] J. Angeles, "Fundamentals of Robotic Mechanical Systems : Theory, Methods and Algorithms", Springer New York, 1997.

[5] J. Baillieul, S. S. Sastry, H. J. Sussmann, "Essays on Mathematical Robotics", Springer-Verlag New York, 1998.

[6] F. Bensalah, "Estimation du mouvement par vision active", Thèse de Doctorat, Université de Rennes I, 1996.

[7] F. Bensalah et F. Chaumette, "Detection de rupture de modele applique a l'asservissement visuel", *IRISA, Publication Interne* No. 886, Novembre 1994.

[8] F. Bonans, "Cours d'optimisation DEA", Universite de Nante, 1991-1992.

[9] J. L. BUESSLER, "Architecture Neuro Mimetique Modulaire Appliquer à L'asservissement Visuel du Systeme Robotique", Thèse de Doctorat, Université de HAUTE-ALSACE, 1999.

[10] V. Cadenat, "Commande Referencee Vision d'un Robot Mobile Muni d'une Camera", Thèse de Doctrat, Université Paul Sabatier, 1999.

[11] F. Chaumette, "La Relation Vision Commande : Théorie et Application à des Tâches Robotique", Thèse de l'Université de Rennes I, Juillet 1990.

[12] F. Chaumette, "Potential problems of stability and convergence in image-based and position-based visual serving", *The confluence of vision and control*, éd. par D. Kreigman, G. Hager et A. Morse, LNCIS 237, pp. 66-78, Springer Verlag, 1998.

[13] F. Chaumette, "De la Perception à l'Action : l'Asservissement Visuel, de l'Action à la Perception : la Vision Active", Habilitation à Diriger des Recherches, Université de Rennes I, 1998.

[14] F. Chaumette et E. Maliz, "2 1/2 D visual servoing, a possible solution to improve image-based and position-based visual servoing", *IEEE Int. Conf. on Robotics and Automation,* Avrile 2000.

[15] C. Collewet, F. Chaumette, L. Wallian et P. Marchal, "Positionnement par rapport a un objet plan de forme inconnue par asservissement visuel 2D", *IRISA, Publication Interne* No 1186, Avril 1998.

[16] C. Collewet, "Contribution à l'Elargissement du Champ Applicatif des Asservissements Visuels 2D", Thèse de Doctorat, Université de Rennes I, 1999.

[17] P. I. Corke, "Visual Servoing of Robots", Research Studies Press Ltd, Paris, Troisième édition, 1992.

[18] P. I. Corke, "Robotics Toolbox : for use with Matlab", Robotics Toolbox User's Guide, juillet 1994.

[19] P.I. Corke et Hutshinson S.A., "Real time vision traking and control", *Proc. of IEEE Int. Conf. on Robotics and Automation*, vol. 2, pp. 1838-1843, 1999.

[20] J. J. Craig, "Introduction To Robotics, Mechanics And Control", Second Edition, Addison-Wesley Publishing Company. 1995.

[21] A. Crétual et F. Chaumette, "Positioning a camera parallel to a plane using dynamic visual servoing", *IEEE International conference on intelligent Robots and systems*, Vol. 1, pp. 43-48, Septembre 1997.

[22] A. Crétual, "Asservissement Visuel à Partir d'Informations de Mouvement dans l'Image", Thèse de Doctorat, Université de Rennes I, 1998.

[23] I. F. Croall et J. P. Mason, "Industriel Applications of Neural Networks", Projet 2092, ANNIE, Vol 1, Springer-Verlag, 1992.

[24] N. Courty et E. Marchand, "Computer animation : a new application for image-based visual servoing", *IEEE Int. Conf. on Robotics and Automation, ICRA* '2001, vol. 1, pp. 223-228, Seoul Korea, Mai 2001.

[25] F. Dornaika, "Contribution à l'Intégration Vision/Robotique : Calibration, Localisation, Asservissement", Thèse de l'INPG, Grenoble, Septembre 1995.

[26] B. Espiau, F. Chaumette et P. Rives, "A new approach to visual servoing in robotics", *IEEE Transactions on Robotics and Automation*, vol. 8, No 3, pp. 313-326, 1992.

[27] J.T. Feddema, Lee (C.S.G.), "Adaptative image feature prediction and control for visual tracking with a hand-eye coordinated camera", *IEEE Trans. on Systems, Man, and Cybernetics*, vol. 20, No 5, pp. 1172-1183, Octobre 1990.

[28] J.T. Feddema, Lee (C.S.G.) et Mitchell (O.R.), "Weighted selection of image features for resolved rate visual feedback control", *IEEE Trans. on Robotics and Automation*, vol. 7, No 1, pp. 31-47, Février 1991.

[29] F. Ferguene, "Mise en Oeuvre de Lois de Commande en Position et en Vitesse d'un Robot Manipulateur : Application a une Tâche de Manipulation d'Objet (Assemblage)", Thèse de Magister, USTHB, 1997.

[30] G. Flandin, F. Chaumette et E. Marchand, "Eye-in-hand/eye-to-hand cooperation for visual servoing", *IEEE International Conference on Robotics and Automation*, Avril 2000.

[31] J. Funda, "Book review : Visual servoing", *IEEE Trans. on Robotics and Automation*, vol. 11, No 3, pp. 144-174, Juin 1995.

[32] J. Gangloff, "Asservissement visuels rapides d'un robot manipulateur à six degrés de liberté", Thèse de Doctorat, Université Louis Pasteur, 1999.

[33] B. K. Ghosh, X. Ning et T. J. Tarn, "Control in Robotics and Automation, Sensor-Based Integration", Academic Press Series in Engineering, 1999.

[34] G. D. Hager, W. C. Chang et A.S. Morse, "Robot hand-eye coordination based on stereo vision", *IEEE Control Systems*, vol. 15, No 1, pp. 30-39, fevrier 1995.

[35] H. Hashimoto et H. Kimura, "LQ Optimal and Nonlinear Approches to Visual Servoing", *Visual Servoing*, éd. par Hashimoto (K.), pp. 165-198, World Scientific Series in Robotics and Automated Systems, vol 7, World Scientific Press, Singapore, 1993.

[36] M. Jägersand, "On-Line Estimation of Visuel Motor Models for Robot Control and Visual Simulation", Thèse de Doctorat, University of Rochester, New York, 1997.

[37] D. Khadraoui, G. Motils, P. martinet, J. Galice et f. Chaumette, "Visual servoing in robotics scheme using a camera/laser-strip sensor", *IRISA, Publication Interne,* No. 898, Janvier 1995.

[38] W. Khalil et E. Dombre, "Modélisation, Identification et Commande des Robots", Collection Robotique, Deuxième édition, HERMES Science Publications, Paris, 1999.

[39] D. KUHN, "Une Approche Neuronale pour l'Asservissement Visuel d'un Robot Manipulateur", Thèse de Doctorat, Université de HAUTE-ALSACE, 1997.

[40] J. S. Lee, H. Suh, B. J. You et S. R. Oh, "A novel visual servoing approach involving disturbance observer", *IEEE International Conference on Robotics and Automation*, Mai 1999.

[41] E. Maliz, "Contributions à la modélisation et à la commande en asservissement visuel", Thèse de Doctorat, Université de Rennes I, 1998.

[42] E. Maliz, F. Chaumette et S. Boudet, "D $\frac{1}{2}$D visuel servoing", *IEEE Transaction on robotics and automation*, vol. 15, No 2, Avrile 1999.

[43] E. Marchand, F. Chaumette, F. Spindler et M. Perrier, "Controlling the manipulator of an underwater ROV using a coarse calibrated Pan/Tilt camera", *IEEE Int. Conference on Robotics and Automation ICRA* '2001, vol. 3, pp. 2773-2778, Seoul Korea, Mai 2001.

[44] E. Marchand et F. Chaumette, "A new formulation for non-linear camera calibration using virtual visual servoing", *IRISA, Publication Interne* No. 1366, Janvier 2001.

[45] Y. Mezouar et F. Chaumette, "Path planning for robust image-based visual servoing", *IRISA, Publication Interne*, No. 4097, Janvier 2001.

[46] N. P. Papanicolopoulos et K. Khosla, "Visual traking of a moving target by a camera mounted on a robot : a combination of control and vision", *IEEE Transaction on robotics and automation*, vol. 9, No. 1, Avril 1993.

[47] N. P. Papanikolopoulos, N. Nelson et P. K. Khosla, "Six dgree-of-freedom hand/eye visual traking with uncertain parameters", *IEEE Transaction on Automatic Control*, Vol. 38, No. 3, 429-445, Mars 1994.

[48] Y. O. Paul et P. K. Allen, "Visual servoing by partitioning degrees of freedom", *IEEE Transactions On Robotics And Automation*, vol. 17, No. 1, Fevrier 2001.

[49] A. Ruf et R. Horaud, "Visual trajectories from uncalibrated stereo", *Computer Graphics Image Processing*, 1989.

[50] M. Tischer et B. Jennrich, "La Bible PC, Programmation Système", Sixième édition, Micro Application, 1997.

[51] W. J. Wilson, C. W. Hulls et G. S. Bell, "Relative end-effector control using cartesian position based visual servoing", *IEEE Transactions on Robotics an Automation*, vol. 12, No 5, pp. 684-696, 1996.

[52] J.S.C. Yuan, "A general photogrammetric method for determining object position and orientation", *IEEE Transaction on Robotics and Automation*, vol. 5, No 2, avril 1989.

[53] A. Zaatri et H. V. Brussel, "Error detection by anticipation for vision-based control", *Science and Technologie*, No. 15, pp. 83-89, Juin 2001.

# Annexe A : Détermination des coordonnées des points fuyants

## Décomposition en tangage, roulis et lacet

La décomposition d'une rotation en tangage, roulis et lacet est décrite par la figure 5.1. Le repère d'origine $R_i$ est défini par les vecteurs $x_i$, $y_i$ et $z_i$. $R_f$ est le repère final définit par $x_f$, $y_f$ et $z_f$. L'angle de tangage $\theta_t$ L'angle de tangage $R_i$ à $R_t$, L'angle de roulis $\theta_r$ transforme $R_t$ en $R_r$.Finalement, l'angle de lacet $\theta_l$transforme $R_r$ en $R_f$.

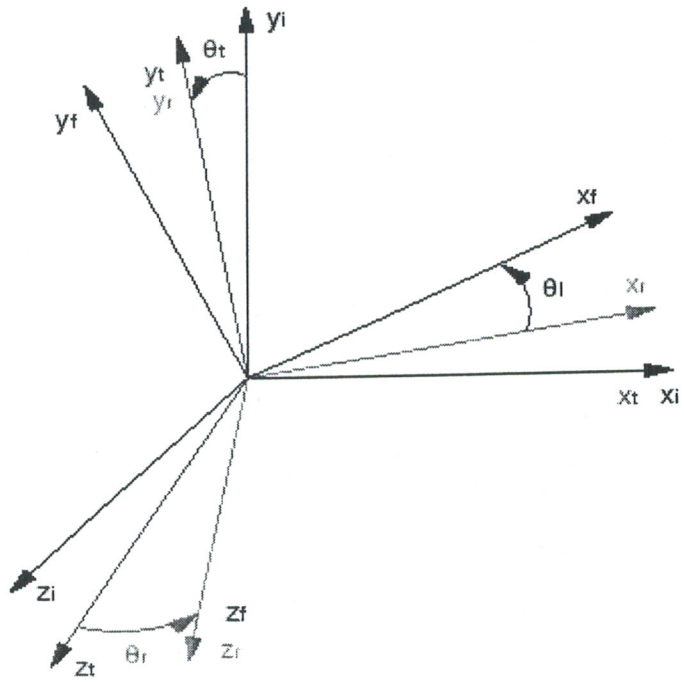



Figure 5.1 : Définition des angles de tangage, roulis et lacet

Détermination des coordonnées des points fuyants exposée dans (3.11)

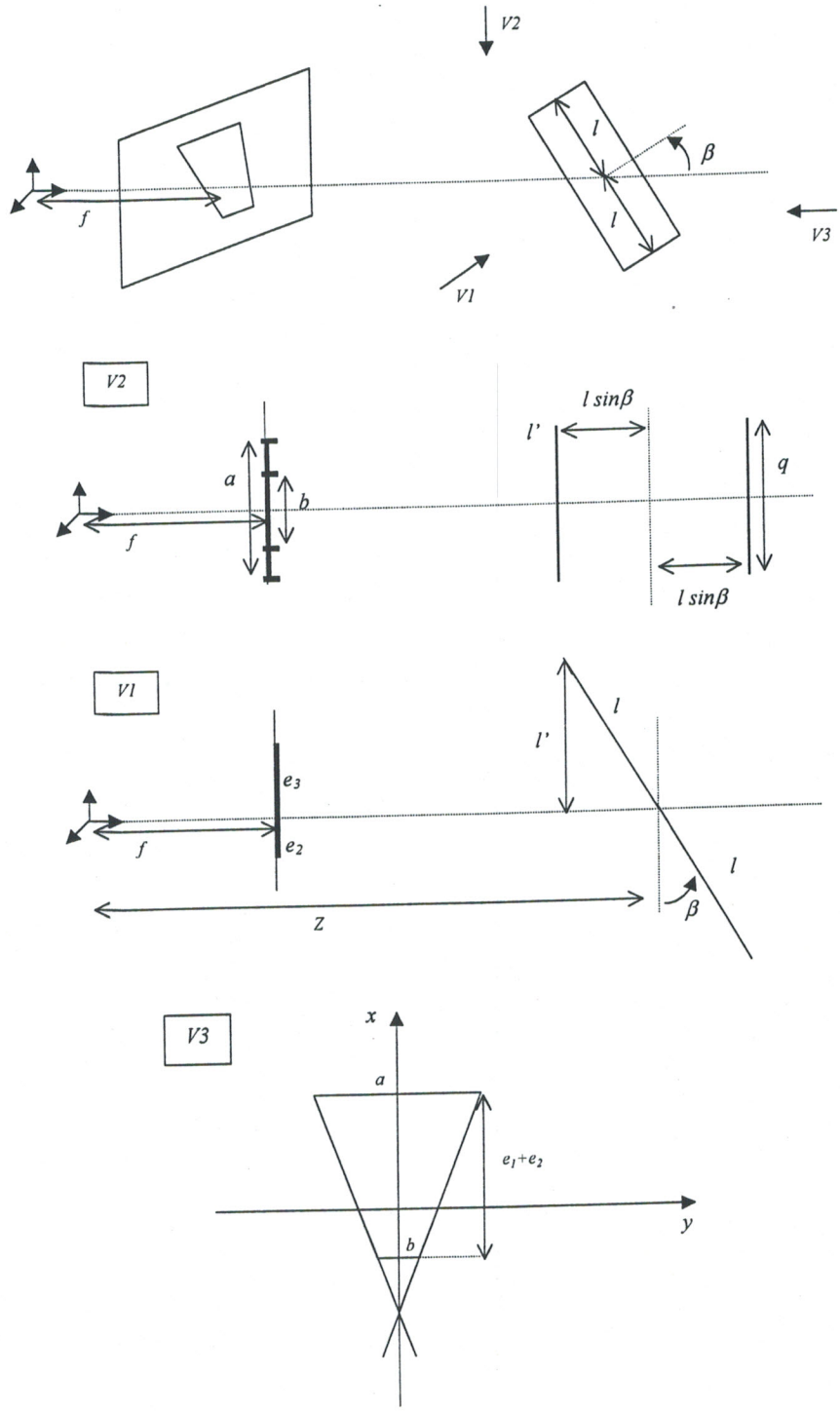



Figure 5.2

En se basant sur la théorie géométrique, suivant la figure 5.2 qui présente un aperçu sur la projection perspective d'un rectangle sur le plan image suivant trois champs de vision ($v_1, v_2$ et $v_3$) (nous avons pris au compte que l'angle $\beta$, l'effet des autres angles est trivial), nous avons :

$$
\begin{aligned}
l' &= l.\cos\beta \\
e_1 &= \frac{f}{Z - l.\sin\beta} l' \quad \text{et} \quad e_2 = \frac{f}{Z + l.\sin\beta} l' \\
e_1 + e_2 &= f\, l' \frac{2Z}{Z^2 - l^2 \sin^2\beta} \\
&= f.l.\cos\beta \frac{2Z}{Z^2 - l^2 \sin^2\beta}
\end{aligned}
$$

$$
\begin{aligned}
a &= \frac{f}{Z - l.\sin\beta} q \\
b &= \frac{f}{Z + l.\sin\beta} q \\
a - b &= f.q \frac{2.l.\sin\beta}{Z^2 - l^2.\sin^2\beta}
\end{aligned}
$$

$$
\begin{aligned}
\frac{\frac{x'_v}{a}}{2} &= \frac{\frac{e_1 + e_2}{a - b}}{2} \\
x'_v &= \frac{2Z.f}{2(Z - l.\sin\beta)} \frac{1}{\tan\beta} \\
x_v &= x'_v - e1 \\
&= \frac{Zf}{Z - l.\sin\beta} \frac{1}{\tan\beta} - \frac{f}{Z - l.\sin\beta} l.\cos\beta \\
&= f \frac{1}{\tan\beta}
\end{aligned}
$$

A l'angle $\gamma$ non nul, $\gamma = f \dfrac{\cos\gamma}{\tan\beta}$

Pour le reste des coordonnées, le même principe peut être suivi et nous amène à donner leur forme générale (3.11) exposées dans le chapitre 3.

| Mécaniques | La description |
|---|---|
| Cinématique | 5 axes pour le bras du robot |
| Moteurs | Servo-moteurs DC |
| Détection de position | Potentiomètre |
| Axe1 (base) | 200° |
| Axe2 (épaule) | 200° |
| Axe3 (coude) | 200° |
| Axe4 (poignet) | 200° |
| Axe5 (outil) | 400° |
| Résolution Axe1 | 512 incréments |
| Résolution Axe2 | 512 incréments |
| Résolution Axe3 | 512 incréments |
| Résolution Axe4 | 512 incréments |
| Résolution Axe5 | 512 incréments |
| Vitesse mximale Axe1 | 50 $°/s$ |
| Vitesse mximale Axe2 | 50 $°/s$ |
| Vitesse mximale Axe3 | 75 $°/s$ |
| Vitesse mximale Axe4 | 120 $°/s$ |
| Vitesse mximale Axe5 | 220 $°/s$ |
| Portée maximale | 590 $mm$ |
| Charge mximale | 500 $g$ |

Tableau 1. Caractéristique mécanique

| Système de contrôle | 9V/1A DC |
|---|---|
| Axes | 24V/8A DC |
| limenttion secteur | 220V/0,5KVA AC |
| Protection de l'appareil | Bouton d'arrêt d'urgence |
| Poids | $\approx 7Kg$ |
| Dimension | $\approx 210mm$ x $300mm$ x $140mm$ |

Tableau 3. Caractéristique de l'alimentation

# Annexe C : Modèle du robot IR50p

## Modèle géométrique

### M.G.D.

Les matrices ${}^{j}T_i$ de transformation homogène

$${}^{\circ}T_1 = \begin{pmatrix} \cos q_1 & -\sin q_1 & 0 & 0 \\ \sin q_1 & \cos q_1 & 0 & 0 \\ 0 & 0 & 1 & D_6 \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad {}^{1}T_2 = \begin{pmatrix} \cos q_2 & -\sin q_2 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ \sin q_2 & \cos q_2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$${}^{2}T_2 = \begin{pmatrix} \cos q_3 & -\sin q_3 & 0 & D_3 \\ \sin q_3 & \cos q_3 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad {}^{3}T_4 = \begin{pmatrix} \cos q_4 & -\sin q_4 & 0 & D_4 \\ \sin q_4 & \cos q_4 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$${}^{4}T_5 = \begin{pmatrix} \cos q_5 & -\sin q_5 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \sin q_5 & -\cos q_5 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$${}^{5}T_E = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & D_E \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad {}^{5}T_c = \begin{pmatrix} 1 & 0 & 0 & a \\ 0 & 1 & 0 & b \\ 0 & 0 & 1 & c \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

avec $D_3 =$, $D_4 =$, $D_E =$, $D_b =$, $a =$, $b =$, $c =$ .

### M.G.I.

$${}^{0}T_5 = \begin{pmatrix} s_x & n_x & b_x & q_x \\ s_y & n_y & b_y & q_y \\ s_z & n_z & b_z & q_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\begin{cases} s_x = \cos_1 \cos_5 \cos_{234} - \sin_1 \sin_5 \\ s_y = \sin_1 \cos_5 \cos_{234} - \cos_1 \sin_5 \\ s_z = \cos_5 \sin_{234} \end{cases} \quad \begin{cases} n_x = -\cos_1 \sin_5 \sin_{234} - \sin_1 \cos_5 \\ n_y = -\sin_1 \sin_5 \cos_{234} - \cos_1 \cos_5 \\ n_z = -\sin_5 \sin_{234} \end{cases}$$

$$\begin{cases} b_x = -\cos_1 \sin_{234} \\ b_y = -\sin_1 \sin_{234} \\ b_z = \cos_{234} \end{cases} \quad \begin{cases} q_x = D_4 \cos_1 \cos_{23} + D_3 \cos_1 \cos_2 \\ q_y = D_4 \sin_1 \cos_{23} + D_3 \sin_1 \cos_2 \\ q_z = D_4 \sin_{23} + D_3 \sin_2 + D_b \end{cases}$$

$$^5T_E = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & D_E \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$U = \begin{pmatrix} p_x \\ p_y \\ p_z \\ 1 \end{pmatrix} = {}^\circ T_5 \begin{pmatrix} 0 \\ 0 \\ D_E \\ 1 \end{pmatrix} = \begin{pmatrix} -D_E \cos_1 \sin_{234} + D_4 \cos_1 \cos_{23} + D_3 \cos_1 \cos_2 \\ -D_E \sin_1 \sin_{234} + D_4 \sin_1 \cos_{23} + D_3 \sin_1 \cos_2 \\ D_E \sin_{234} + D_4 \sin_{23} + D_3 \sin_2 + D_b \\ 1 \end{pmatrix} = V$$

$$A = \begin{pmatrix} a_x \\ a_y \\ a_z \\ 0 \end{pmatrix} = {}^\circ T_5 \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} -\cos_1 \sin_{234} \\ -\sin_1 \sin_{234} \\ \cos_{234} \\ 0 \end{pmatrix} = B$$

$U_1 = {}^1T_0.U \quad \text{et} \quad V_1 = {}^1T_0.V$

$$U_1 = \begin{pmatrix} \cos_1 \ p_x + \sin_1 \ p_y \\ -\sin_1 \ p_x + \cos_1 \ p_y \\ p_z - D_b \\ 1 \end{pmatrix} \qquad V_1 = \begin{pmatrix} -D_E \sin_{234} + D_4 \cos_{23} + D_3 \cos_2 \\ 0 \\ D_E \cos_{234} + D_4 \sin_{23} + D_3 \sin_2 \\ 1 \end{pmatrix}$$

$A_1 = {}^1T_0 A \quad \text{et} \quad B_1 = {}^1T_0 B$

$$A_1 = \begin{pmatrix} \cos_1 \ a_x + \sin_1 \ a_y \\ -\sin_1 \ a_x + \cos_1 \ a_y \\ a_z \\ 1 \end{pmatrix} \qquad B_1 = \begin{pmatrix} \sin_{234} \\ 0 \\ \cos_{234} \\ 0 \end{pmatrix}$$

$$\boxed{\begin{cases} q_1 = \text{atan2}(p_y, p_x) \\ q_1' = q_1 + 180° \end{cases}}$$

on à : $\begin{cases} \sin_{234} = -a_x \cos_1 - a_y \sin_1 \\ \cos_{234} = a_z \end{cases} \Longrightarrow q_{234} = \text{atan2}(-a_x \cos_1 - a_y \sin_1, a_z)$

un système d'équation de type : $\begin{cases} X \cos\theta_i + Y \cos(\theta_i + \theta_j) = Z1 \\ X \sin\theta_i + Y \sin(\theta_i + \theta_j) = Z2 \end{cases}$ a pour solution

$$\begin{cases} \cos\theta_j = \dfrac{Z_1^2 + Z_2^2 - X^2 - Y^2}{2XY} \\ \theta_j = \text{atan2}(\pm\sqrt{1-\cos\theta_j^2}, \cos\theta_j) \\ B_1 = X + Y\cos\theta_j \\ B_2 = Y\sin\theta_j \\ \sin\theta_i = \dfrac{B_1 Z_2 - B_2 Z_1}{B_1^2 + B_2^2} \\ \cos\theta_i = \dfrac{B_1 Z_1 - B_2 Z_2}{B_1^2 + B_2^2} \\ \theta_i = \text{atan2}(\sin\theta_i, \cos\theta_i) \end{cases}$$

Posons :
$$\begin{cases} X = D_3 \quad \text{et} \quad Y = D_4 \\ Z_1 = \cos_1 p_x + \sin_1 p_y + D_E \sin_{234} \\ Z_2 = p_z - D_E \cos_{234} - D_b \\ \cos_3 = \dfrac{Z_1^2 + Z_2^2 - X^2 - Y^2}{2XY} \end{cases}$$

Alors ;
$$\boxed{\begin{cases} q_3 = \text{atan2}(\sqrt{1-\cos^2 3}, \cos 3) \\ q_3' = -q_3 \end{cases}}$$

$B_1 = X + Y\cos_3$

$B_2 = Y\sin_3$

$$\sin_2 = \frac{B_1 Z_2 - B_2 Z_1}{B_1^2 + B_2^2}$$

$$\cos_2 = \frac{B_1 Z_1 + B_2 Z_2}{B_1^2 + B_2^2}$$

Donc :
$$\boxed{\begin{cases} q_2 = \text{atan2}(\sin_2, \cos_2) \\ q_4 = q_{234} - (q_2 + q_3) \\ \forall q_5 \end{cases}}$$

# Modéle cinématique

Je jacobien $^\circ J_5$ présenté ci-desous, transforme les vitesses articulaires en vitesses cartesiennes du repère $R_5$ par rapport au repère de base $R_0$. $^\circ J_5(i,j)$ est l'élément de la $i^{eme}$ ligne et la $j^{eme}$ colonne.

$^\circ J_5(1,1) = -\frac{1}{2}D_4[\sin(q_1+q_2+q_3) + \sin(q_1-q_2-q_3)] + \frac{1}{2}D_3[\sin(q_1+q_2) + \sin(q_1-q_2)]$

$^\circ J_5(1,2) = \frac{1}{2}D_4[\sin(q_1-q_2-q_3) - \sin(q_1+q_2+q_3)] + \frac{1}{2}D_3[\sin(q_1-q_2) - \sin(q_1+q_2)]$

$^\circ J_5(1,3) = \frac{1}{2}D_4[\sin(q_1-q_2-q_3) - \sin(q_1+q_2+q_3)]$

$^\circ J_5(1,4) = 0$

$^\circ J_5(1,5) = 0$

$^\circ J_5(2,1) = \frac{1}{2}D_4[\cos(q_1+q_2+q_3) + \cos(q_1-q_2-q_3)] + \frac{1}{2}D_3[\cos(q_1+q_2) + \sin(q_1-q_2)]$

$^\circ J_5(2,2) = \frac{1}{2}D_4[\cos(q_1+q_2+q_3) - \cos(q_1-q_2-q_3)] + \frac{1}{2}D_3[\cos(q_1+q_2) - \cos(q_1-q_2)]$

$^\circ J_5(2,3) = \frac{1}{2}D_4[\cos(q_1+q_2+q_3) - \cos(q_1-q_2-q_3)]$

$^\circ J_5(2,4) = 0$

$^\circ J_5(2,5) = 0$

$^{\circ}J_5(3,1) = 0$

$^{\circ}J_5(3,2) = D_4\cos(q_2+q_3) + D_3\cos(q_2)$

$^{\circ}J_5(3,3) = D_4\cos(q_2+q_3)$

$^{\circ}J_5(3,4) = 0$

$^{\circ}J_5(3,5) = 0$

$^{\circ}J_5(4,1) = 0$

$^{\circ}J_5(4,2) = \sin(q_1)$

$^{\circ}J_5(4,3) = \sin(q_1)$

$^{\circ}J_5(4,4) = \sin(q_1)$

$^{\circ}J_5(4,5) = \frac{1}{2}\sin(q_1-q_2-q_3-q_4) - \frac{1}{2}\sin(q_1+q_2+q_3+q_4)$

$^{\circ}J_5(5,1) = 0$

$^{\circ}J_5(5,2) = -\cos(q_1)$

$^{\circ}J_5(5,3) = -\cos(q_1)$

$^{\circ}J_5(5,4) = -\cos(q_1)$

$^{\circ}J_5(5,1) = \frac{1}{2}\cos(q_1+q_2+q_3+q_4) - \frac{1}{2}\cos(q_1-q_2-q_3-q_4)$

$^{\circ}J_5(6,1) = 1$

$^{\circ}J_5(6,2) = 0$

$^{\circ}J_5(6,3) = 0$

$^{\circ}J_5(6,4) = 0$

$^{\circ}J_5(6,5) = \cos(q_2+q_3+q_4)$

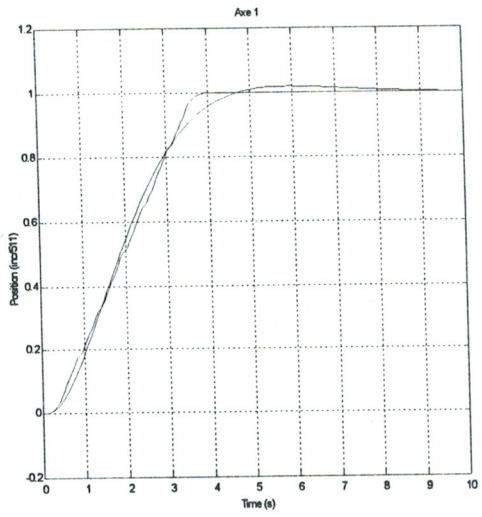
# Modèle dynamique des axes asservis en position

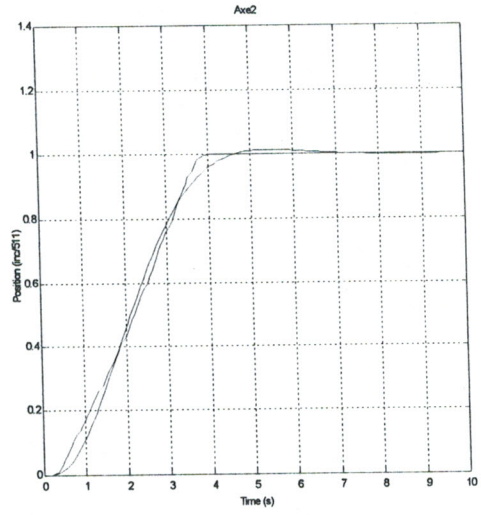
La fonction de transfet est de la forme :

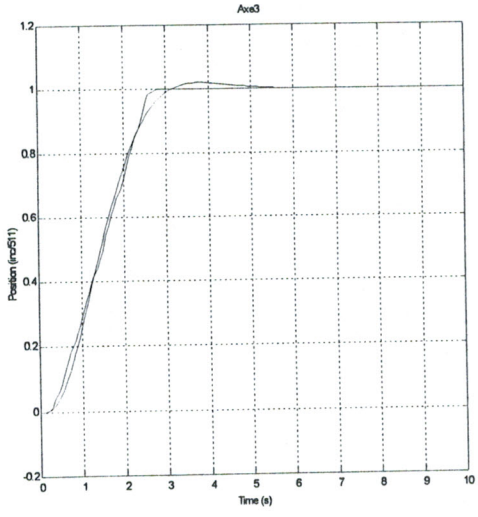
$$F(s) = \frac{w_n^2}{(\alpha s+1)(s^2+2\xi w_n s+w_n^2}$$

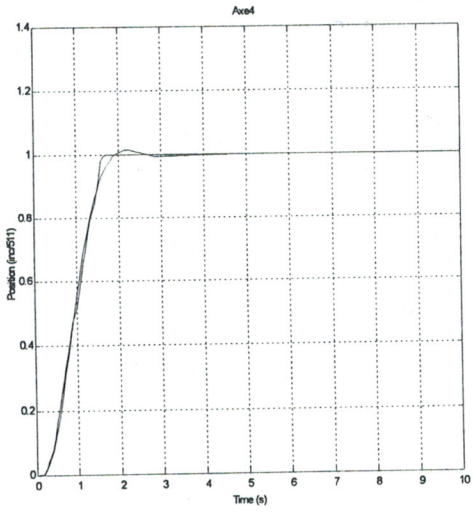
| Axe | $\alpha$ | $\xi$ | $w_n$ |
|---|---|---|---|
| 1 | 0.1515 | 0.7812 | 0.8735 |
| 2 | 1.0576 | 0.6413 | 1.1566 |
| 3 | 0.5668 | 0.7036 | 1.5605 |
| 4 | 0.5531 | 0.5325 | 2.6905 |
| 5 | 0.5638 | 0.5123 | 2.905 |

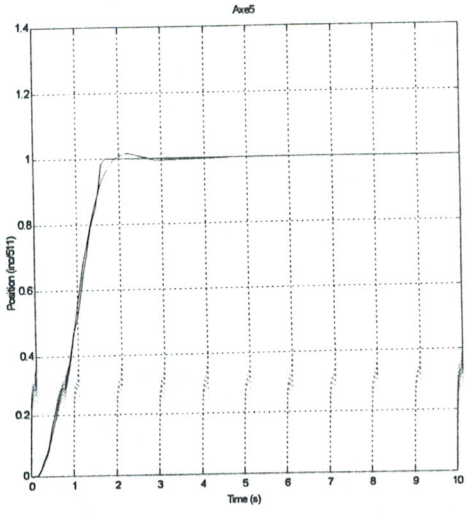
La reponse pour une entrée échelon de chaque axe :


Axe 1
Position (inc/511)
Time (s)


Axe2
Position (inc/511)
Time (s)


Axe3
Position (inc/511)
Time (s)


Axe4
Position (inc/511)
Time (s)


Axe5
Position (inc/511)
Time (s)

# Les fonctions de conversion inc/degré et degré/inc

| Axe | $q_{inc}$ en incrément | $q$ en degré | $\triangle_j$ |
|---|---|---|---|
| 1 | $255 - \frac{q_d}{\triangle_1}$ | $\triangle_1.(255 - q_{inc})$ | 0.43 |
| 2 | $\frac{188 - q_d}{\triangle_2}$ | $188 - q_{inc}.\triangle_2$ | 0.384 |
| 3 | $255 - \frac{q_d}{\triangle_3}$ | $\triangle_3.(255 - q_{inc})$ | 0.384 |
| 4 | $255 - \frac{q_d}{\triangle_4}$ | $\triangle_4.(255 - q_{inc})$ | 0.379 |
| 5 | $255 + \frac{q_d}{\triangle_5}$ | $\triangle_5.(255 - q_{inc})$ | 0.763 |

# Annexe D : Programmation

## Acquisition video sous windows via le driver

Dans cette section nous expliquons toutes les taches nécessaires pour l'initialisation de la capture et la communication avec le driver de capture .

Ces differentes taches en detail sont les suivantes :

♦ - Créer une fenêtre de capture : Les ligne suivantes crée une fenêtre de capture en utilisant la fonction *capCreateCaptureWindow.*

```
hWndC = capCreateCaptureWindow (
(LPSTR) "My Capture Window", // window name if pop-up
WS_CHILD | WS_VISIBLE, // window style
0, 0, 160, 120, // window position and dimensions
(HWND) hwndParent,
(int) nID /* child ID */);
```

♦ Connecter cette fenetre a un driver de cature : On peut connecter la fenêtre de la capture avec le Handle *hWndC* au Driver de MSVIDEO et puis on peut le déconnecter on utilisant le macro *capDriverDisconnect*

```
fOK = SendMessage (hWndC, WM_CAP_DRIVER_CONNECT, 0, 0L); //
// Ou utiliser une macro pour connecter au MSVIDEO driver :
// fOK = capDriverConnect(hWndC, 0); //
// Code de capture ici //
capDriverDisconnect (hWndC); //Pour se deconnecter.
```

♦ Enumerer les drivers de capture installés : Le code suivant utilise la fonction *capGetDriverDescription* pour obtenir les noms et versions des Drivers de capture installés.

```
for (wIndex = 0; wIndex < 10; wIndex++)
{
if (capGetDriverDescription (wIndex, szDeviceName,
sizeof (szDeviceName), szDeviceVersion,
sizeof (szDeviceVersion))
```

```
{
// l'utilisateur doit sélectionner un Driver pour l'utiliser.
}
}
```

♦ Obtenir la situation d'une fenêtre de capture : Le code suivant utilise la fonction *SetWindowPos* pour mettre la dimension de la fenêtre de capture aux dimensions basées sur les informations retournées par le macro du *capGetStatus* dans la structure *CAPSTATUS*.

```
CAPSTATUS      CapStatus;
capGetStatus(hWndC, &CapStatus, sizeof (CAPSTATUS));
SetWindowPos(hWndC, NULL, 0, 0, CapStatus.uiImageWidth,
CapStatus.uiImageHeight, SWP_NOZORDER | SWP_NOMOVE);
```

♦ Afficher la boite de dialogue pour changer les caracteristiques video : Chaque Driver de capture peut fournir jusqu'à trois boîtes du dialogue différentes qui controlent le format et la source de capture. Le code suivant démontre comment afficher ces boîtes de dialogue. Avant d'afficher chaque boîte du dialogue, le macro *capDriverGetCaps* permet, grace à la structure CAPDRIVERSCAPS, de verifier que le driver de capture vidéo est correctement installé.

```
CAPDRIVERCAPS CapDrvCaps;
capDriverGetCaps(hWndC, &CapDrvCaps, sizeof (CAPDRIVERCAPS));
// Video source dialog box.
if (CapDriverCaps.fHasDlgVideoSource)
capDlgVideoSource(hWndC);
// Video format dialog box.
if (CapDriverCaps.fHasDlgVideoFormat)
{
capDlgVideoFormat(hWndC);
// Are there new image dimensions ?
capGetStatus(hWndC, &CapStatus, sizeof (CAPSTATUS));
// If so, notify the parent of a size change.
}
// Video display dialog box.
if (CapDriverCaps.fHasDlgVideoDisplay)
capDlgVideoDisplay(hWndC);
```

♦ Obtenir et choisir le format video : Parce que la structure BITMAPINFO est de longueur variable, le programmeur doit mettre en doute toujours la dimension de la structure et allouer la

mémoire avant de rapporter le format de la vidéo courant. L'exemple suivant utilise le macro du *capGetVideoFormatSize* pour rapporter la dimension du buffer et alors appele le macro *capGetVideoFormat* pour rapporter le format de la vidéo courant.

```
LPBITMAPINFO lpbi;
DWORD dwSize;
dwSize = capGetVideoFormatSize(hWndC);
lpbi = GlobalAllocPtr (GHND, dwSize);
capGetVideoFormat(hWndC, lpbi, dwSize);
// Access the video format and then free the allocated memory.
```

Le programmeur peut utiliser le macro *capSetVideoFormat* (ou le message *WM_CAP_SET_VIDEOFO* pour envoyer une structure BITMAPINFO à la fenêtre de la capture. On doit vérifier la valeur du retour pour déterminer si le format avait été accepté.

♦ L'aperçu video : Le code suivant utilise le macro *capPreviewRate* pour mettre le taux de l'exposition de la frame pour le mode de l'aperçu à 66 millisecondes par Frame et utilise le macro *capPreview* pour placer la fenêtre de la capture dans mode de l'aperçu.

```
capPreviewRate(hWndC, 66); // rate, in milliseconds
capPreview(hWndC, TRUE); // starts preview
// Preview
capPreview(hWnd, FALSE); // disables preview
```

♦ Créer une fonction Fram Callback : L'exemple suivant est une simple fonction Fram callback. Enregistrer ce rappel en utilisant le macro du capSetCallbackOnFrame.

```
LRESULT PASCAL FrameCallbackProc(HWND hWnd, LPVIDEOHDR lpVHdr)
{
if (!ghWndMain)
return FALSE;
wsprintf(gachBuffer, "Preview frame# %ld ", gdwFrameNum++);
SetWindowText(ghWndMain, (LPSTR)gachBuffer);
return (LRESULT) TRUE;
}
```

## Protocole de communication série avec le IR50p

Au debut, le PC hôte doit transmettre l'octet 20hex au robot, en retour le contrôleur répond par l'un des octets suivants :

- 15hex : communication établie, le préhenseur est électrique.
- 16hex : communication établie, le préhenseur est pneumatique.

- F1hex, F2hex, F3hex : en cas d'initialisation multiple du robot.

Toute autre réponse indique que la communication n'est pas établie.

### Commande en position des axes

La commande est constituée par un nombre défini d'octets, le dernier est ETX=03hex, les octets sont dans l'ordre suivant :

**Command** **Data** **ETX**

**Command** : 0000RA.

**A** prend le numéro de l'axe à boujer (entre 0 et 5), ou 7 pour tous les axes.

**R**=1 le robot retourne un message de fin de tâche.

**R**=0 aucun message à retourner.

**Data** : contient deux octets dans le cas d'un seul axes, le premier égale le quotient de division de l'incrément désiré par deux, et le deuxième le reste de cette division. Pour tous les axes, il contient sept octets, les six premiers contiennent le quotient de la division par deux des incréments désirés des axes 1 à 6 (préhenseur inclus), le septième octet contient les restes de cette division.

**ETX** : =03hex.

### Lecture des positions articulaires en cours

La commande à envoyer : **Command** **ETX**

**Command** : 01000A

**A** prend le numéro de l'axe à lire sa position actuelle (entre 0 et 5).

Le message retourné : **Command** **OCT1** **OCT2** **ETX**

Les deux octets **OCT1** et **OCT2** de position actuelle sont codés de la même manière que **Data**.

**A** prend 7 pour tous les axes.

Le message retourné : **Command** **7 Octets** **ETX**

Les sept octets sont codés de la même façon que Précédemment.

### Commande en position des axes avec vitesse

Les mêmes types de commande en position pour ce qui concerne positionnement d'un axes ou de tous les axes,

**Command** **Data** **Time** **ETX**

**Command** : 01110A

Le cas d'un seul axe, **Time** est un octet qui défini la vitesse par 1 $inc/\ (Time * 10ms)$. Pour tous les axes, **Time** est défini par 6 octets, chaque octet défini la vitesse de chaqu'un des 6 axes (préhenseur inclus).

# Programmation du port série RS232 sous windows

Les ports séries sont considérés comme des fichiers, on utilise donc les fonctions : CreateFile, ReadFile, WriteFile et CloseHandle de l'API WIN32.

## Ouverture d'un port série

*HANDLE CreateFile(LPCTSTR szNomFichier,DWORD dwTypeAcces, DWORD dwPartage, LPSECURITY_ATTRIBUTES lpSecurityAttributes, DWORD dwModeCreation, DWORD dwFlagsAndAttributes, HANDLE hTemplateFile);*

Pour ouvrir un port série, il suffit de renseigner le nom de fichier de la façon suivante : "\\\\.\\COM2" pour le port série 2. L'ouverture du port parallèle se fait d'ailleurs de la même façon : "\\.\LPT1".

Le deuxième paramètre : dwTypeAcces permet de définir le type d'accès : GENERIC_READ (pour pouvoir lire) ou GENERIC_WRITE (pour pouvoir écrire). On utilise en général la combinaison de ces deux paramètres : GENERIC_READ | GENERIC_WRITE

Le troisième paramètre : dwPartage indique le mode de partage désiré. Pour un port série, on ne peut pas le partager, on indique donc 0.

Le quatrième paramètre : lpSecurityAttributes est NULL dans notre cas.

Le cinquième paramètre : dwModeCreation nous indique la façon d'ouvrir le fichier. Le port que l'on veut ouvrir doit exister dans notre cas, il faut donc indiquer OPEN_EXISTING.

Le sixième paramètre : dwFlagsAndAttributes permet de définir les attributs du fichier à ouvrir. En général, on utilise le flag FILE_ATTRIBUTE_NORMAL, mais on peut aussi le combiner avec le flag FILE_FLAG_OVERLAPPED. L'utilisation de ce flag sera décrite plus loin.

Le septième et dernier paramètre : hTemplateFile est généralement NULL.

Attention, en cas de problème d'ouverture du fichier, le handle retourné n'est pas NULL, mais INVALID_HANDLE_VALUE.

```
HANDLE hCom = CreateFile(_T("\\\\.\\COM2"),GENERIC_READ | GENERIC_WRITE,0,N
OPEN_EXISTING, FILE_ATTRIBUTE_NORMAL, NULL);
if(hCom == INVALID_HANDLE_VALUE)
{
    //Erreur d'ouverture du port de communication.
}
```

## Paramètrage du port de communication

Pour paramétrer la vitesse du port, le nombre de bits, le nombre de bits de stop etc... on utilise la structure DCB.

On appelle la fonction GetCommState en lui passant le HANDLE du fichier et un pointeur sur une structure de type DCB. Il ne reste plus qu'a mettre à jour les différents éléments de cette

structure afin d'initialiser correctement le port série. Le paramètre EvtChar de cette structure est particulièrement intéressant lors de l'utilisation du mode Overlapped. On appelle ensuite la fonction SetCommState pour mettre à jour la configuration du port série. Si on désire utiliser le paramètre EvtChar de la structure DCB, il faut appeler la fonction SetCommMask.

Si l'on travaille en mode normal, il est fortement conseillé d'utiliser la fonction SetCommTimeouts afin d'éviter d'être bloquer sur une lecture du port série. Dans le pire des cas, la fonction de lecture échouera en time out et le programme ne sera pas gelé.

### La lecture

La lecture d'un port série se fait en utilisant l'instruction ReadFile.

*BOOL ReadFile(HANDLE hCom, LPCVOID pBuffer, DWORD dwNbByteALire, LPDWORD pdwNbByteLu, LPOVERLAPPED pOver);*

Le problème, sur un port série, est de connaître le nombre de caractères à lire, en utilisant la méthode précédente, pas de problème. Si le port de communication a été ouvert en utilisant le flag FILE_FLAG_OVERLAPPED, on est obligé de renseigner le paramètre pOver, sinon on peut indiquer NULL. Il faut évidemment que le buffer de lecture soit suffisamment grand pour pouvoir lire tous les caractères. Une lecture est correcte lorsque le code de retour est vrai et que le nombre de caractères lus est identique au nombre de caractères à lire.

### L'écriture

L'écriture sur un port série se fait en utilisant la fonction ReadFile.

*BOOL WriteFile(HANDLE hCom, LPCVOID pBuffer, DWORD dwNbByteAEcrire, LPDWORD pdwNbByteEcrit, LPOVERLAPPED pOver);*

En général, il n'y a aucun problème pour écrire sur un port série. Si le port de communication a été ouvert en utilisant le flag FILE_FLAG_OVERLAPPED, on est obligé de renseigner le paramètre pOver, sinon on peut indiquer NULL.

### La fermeture

Il ne faut pas oublier de fermer le port de communication. Il sera de toutes façons fermer lors de l'arrêt du programme mais autant travailler proprement. Pour cela, on utilise la fonction CloseHandle.

## Interface logicielle

1. Contrôle du robot : Dans cette zone, on peux choisir les valeurs des positions articulaires en incrément où en degré, même les articulations à commander, un affichage des positions articulaires courantes est disponible, la vitesse de déplacement des articulations est adaptée par l'utilisateur par le contrôleur speed, l'envoi des consignes de commande est validé par le bouton Go !.

2. La vidéo avec et sans traitement : Dans cette section, la séquence vidéo de la scène réele sont dans le cadre gauche, Alors qu'à droite, un aperçu des images après traitement est affiché.

3. Pilote de la carte d'acquisition vidéo : cette zone nous permet d'accéder au pilote (driver) de la carte d'acquisition vidéo. Le bouton V-S pour régler la luminosité et le contraste, ainsi que sélectionné la source du signal vidéo. Le bouton V-F pour choisir le format d'image et la taille.

4. Les boutons radio Filtrage par couleur et Filtrage N/B pour sélectionner le mode de traitement (objet disque noir sur fond clair ou objet à couleur unique dans l'environnement de travail).

Commande visuel

VIDEO CAMERA (160 x 120)

Image réelle

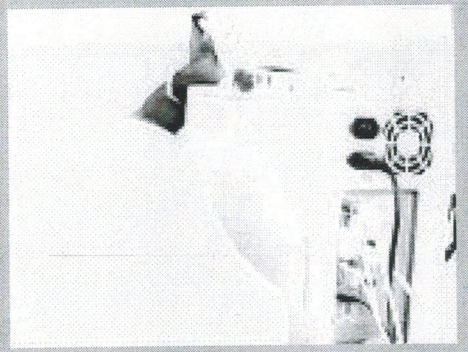


Image binarisée

Choix des Qi (Inc)

| | | |
|---|---|---|
| Q1 | 130 | ☑ |
| Q2 | 255 | ☐ |
| Q3 | 22 | ☐ |
| Q4 | 252 | ☐ |
| Q5 | 6 | ☐ |
| Pince | | ☐ |

Valeurs des Qi (Inc)

| | |
|---|---|
| Q1 | 130 |
| Q2 | 255 |
| Q3 | 22 |
| Q4 | 252 |
| Q5 | 6 |
| Pince | |

Aperçu ☑

Avec trajectoire ☐

Asservissement ☐

Commande du Robot

Speed (inc/10ms) 2

Go !

| | | | | | |
|---|---|---|---|---|---|
| Xo (pixels) | | Xd | 86 | Dx1 | |
| Yo (pixels) | | Yd | 93 | Dx4 | |

○ Filtrage N/B

● Filtrage par Couleur

Seuil 140

| Rouge | Vert | Bleu |
|---|---|---|
| 20 | 20 | 20 |

Video Card Driver

V-S V-F

Lire le Fond

○ Image Non Traitée

Taille de surface 2 X 2

EXIT

**Résumé :**
*Dans ce mémoire, nous décrivons la méthode permettant de réaliser une tâche de positionnement par asservissement visuel 2D où la consigne est une position des primitives dans l'image. Nous traitons dans un premier temps le cas où le système Caméra- Robot est modélisé avec un aperçu sur les performances du système, une solution utilisant les points fuyants. Nous montrons ensuite comment l'approche 2D peut se généraliser indépendamment du modèle de système par la méthode Quasi Newton.*
*La formule de Broyden estime en ligne le jacobien et la loi de commande basée sur la méthode Quasi Newton. Le contrôle est indépendant du modèle du robot et de la caméra, et est capable de suivre les cibles statiques et dynamiques.*
**Mots clés :** Asservissement visuel, Robot, Tâche de positionnement.

**Abstract :**
*In this magister dissertation, we describe a method to achieve positioning by visual servoing image based, where the setpoint is a position of the features in the image. Initially we discuss the case where the system Camera- Robot is known by his model. We analyze the performances of the system using the solution vanishing points.*
*We show then how the 2D approach can be generalized independently of the system by the Quasi Newton method.*
*The formula of Broyden estimates on-line the jacobien and the control based on the Quasi Newton method. The control is independent of the model of the robot-camera, and is able to track the static and dynamic targets.*
**Keywords :** Visual servoing, Robot, Positionning task.