UNIVERSITE DE SAAD DAHLEB DE BLIDA

Faculté des Sciences de l'ingénieur

Département d'électronique

MEMOIRE DE MAGISTER

Spécialité : Electronique

Option : Images et parole.

FUSION MULTISENSORIELLE ET TELEOPERATION : APPLICATION A UN ROBOT MOBILE MANIPULATEUR

Par

RAHICHE Abderrahmane

Devant le jury composé de

K. FERDJANI	Maître de conférence, U. de Blida	Président
K. KARA	Maître de conférence, U. de Blida	Examinateur
N. ACHOUR	Maître de conférence, U.S.T.H.B, Alger	Examinatrice
B. KAZED	Chargé de cours, U. de Blida	Examinateur
B. BOUZOUIA	Directeur de recherche, CDTA, Alger	Rapporteur
M. BERSALI.	Maître assistant, U. de Blida	Invité

Blida, Octobre 2007.

RESUME

Ce travail se situe dans le cadre de la robotique d'intervention. L'utilisation du robot d'intervention provient du besoin de donner à un opérateur qui se trouve éloigné du site d'opération, la possibilité d'agir à distance sur des objets dans un environnement.

Notre objectif est l'élaboration et la mise en oeuvre d'un système de commande de robot mobile manipulateur ROBUTER-ULM à distance. Ce système doit permettre à un opérateur qui se trouve dans un poste de téléopération, et qui n'est pas forcément un expert, d'exercer des commandes sur le robot, tout en envoyant des consignes et en recevant des images vidéo du site et les information de l'état du robot issu des différents capteurs installée sur le robot.

Donc, nous devons récupérer les coordonnés réelles des objets perçus par une seule caméra CCD monochrome, puis nous devons trouver leur position par rapport au robot, ensuite nous calculons les commandes convenables pour que le robot puisse agir sur un objet cible, et nous surveillons l'exécution des missions en exploitant les autres capteurs dont le robot dispose.

ملخص

هذا العمل يندرج ضمن إطار الأبحاث المتعلقة بتطبيقات الروبوت المستعمل في التدخل، المتحكم فيها عن بعد. هذا النوع من التطبيقات الحيوية فرض نفسه باستعمالاته الواسعة و بالنتائج الإيجابية التي حققها، لذا فهو يتطلب الدقة في العمل والفعالية سواء بالنسبة للروبوتات المستعملة أو الأنظمة التي تتحكم فيها.

عملنا هذا يهدف إلى إنجاز وتطبيق نظام تحكم عن بعد في روبوت آلي، يتيح لأي شخص يريد أن يستعمل الروبوت عن بعد، القيام بوظائف معينة دون الحاجة للخبرة والكفاءة. فالنظام يسمح له برؤية مكان الحدث إعتمادا على كاميرا مخصصة لهذا الغرض، وكذا يعطيه معلومات دقيقة عن كل ما يتعلق بالحالة الذاتية للروبوت وكل المعطيات عن محيطه.

بصيغة أخرى، المشكل الذي نود معالجته هو، انه انطلاقا من صورة ثنائية الأبعاد لحائط مثبتة به أشكال مختلفة تأتينا من كاميرا مثبة في طرف ذراع آلي علينا استخراج الأبعاد الحقيقية للأشياء التي نراها في الشاشة، ومن ثم نقوم باصدار الأوامر المناسبة للروبوت، التي تسمح له بأن يلمس أو يصل بطرفه إلى أي شكل موجود في الصورة بدقة وفعالية.

ABSTRACT

This work is within the framework of the robotics of intervention. The usage of the interventional robot comes from the need to give to an operator who is far away from the site of operation, the possibility of acting remotely on objects in an environment.

Our objective is the development and the implementation of a mobile control device of robot remote manipulator ROBUTER-ULM. This system must allow an operator who is in a station of teleoperation, and which is not inevitably an expert, to exert orders on the robot, while sending instructions and by receiving video images of the site and the information of the state of the robot from the various sensors installed on the robot.

The problem that we treated here, it is starting from the monochromic images come from only one camera, we must recover coordinated perceived objects, then we will locate their position compared to the robot, then the suitable orders are calculated so that the robot can act on a target object, and we supervise the execution of the missions by exploiting the other sensors which it lays out the robot.

REMERCIEMENTS

Je tiens à remercier mon dieu, le plus puissant de m'avoir donné le courage, la volonté et la patience pour réaliser ce travail.

Je tiens à remercier mon promoteur, Monsieur BOUZOUIA Brahim, d'avoir accepter de diriger mon mémoire.

Je tiens à remercier également les membres de jury, pour l'intérêt qu'ils ont bien voulus porter à ce travail et d'avoir accepté de faire partie du jury.

Je remercie également toute personne ayant contribuée à la réalisation de ce travail de près ou de loin.

TABLE DES MATIERS

RESUME

REMERCIEMENTS

TABLE DES MATIERES

LISTE DES ILLUSTRATIONS, GRAPHIQUES ET TABLEAUX

INTRODUCTION 13

1. LA TELEOPERATION : ETAT DE L'ART. 17

1.1. Introduction. 17

1.2. Définition. 17

1.3. Historique de la téléopération. 17

1.4. Intérêts et domaines d'application 22

1.4.1. Le domaine nucléaire. 23

1.4.2. Le domaine militaire. 24

1.4.3. Le domaine spatial. 25

1.4.4. Le domaine médical. 26

1.4.5. Le domaine d'exploration sous marine. 29

1.5. Types de téléopération. 29

1.5.1. Classification suivant la nature de liaison. 30

1.5.2. Classification suivant le type d'interaction. 32

1.5.3. Classification suivant la position de la caméra 34

1.6. Conclusion. 36

2. DESCRIPTION DU SYSTEME DE TELEOPERATION 37

2.1. Introduction. 37

2.2. Structure de notre système de téléopération 37

2.3. Description des différents modules du système de téléopération 38

2.3.1. Le PC hôte (client) 39

2.3.2. L'interface Homme/Machine. 39

2.3.3. Communication Réseau (Liaison client /serveur). 40

2.3.4. Le PC embarqué (serveur). 42

2.3.5. Système imageur. 42

2.3.6. Robot mobile manipulateur téléopéré. 45
2.4. Structure de Robuter_ULM. 45
2.5. Description du Robuter. 46
2.6. Le système sensoriel de Robuter. 47
2.6.1. Les Capteurs proprioceptifs du robot mobile. 47
2.6.2 Les Capteurs extéroceptifs du robot mobile. 48
2.7. Description du bras manipulateur ultra léger ULM. 54
2.8. Système sensoriel du bras manipulateur. 54
2.8.1. Capteurs de position des axes. 54
2.8.2. Capteurs de fin de cours. 55
2.8.3. La caméra. 56
2.9. Architecture logicielle du Robuter. 56
2.9.1. L'environnement Linux. 58
2.9.2. Linux Real Time Application Interface (RTAI). 59
2.9.3. SynDEx 59
2.9.4. Le GNUPro Toolchain 60
2.10. Conclusion. 60
3. MODELISATION GEOMETRIQUE DU SYSTEME ROBOTIQUE. 61
3.1. Introduction. 61
3.2. Transformation repère caméra/repère pince. 63
3.3. Transformation repère bras/repère robot mobile 63
3.4. Modélisation du robot mobile (Robuter) 63
3.4.1. Modélisation géométrique du robot mobile 63
3.4.2. Description des roues. 64
3.4.3. Modélisation cinématique du robot mobile. 67
3.5. Modélisation géométrique du bras ULM. 69
3.5.1 Modélisation géométrique directe de ULM. 70
3.5.2. Modèle géométrique inverse. 73
3.6. Calibration de la caméra. 76
3.6.1. Le modèle géométrique de la caméra. 77
3.6.2. Calibration de la caméra. 86
3.7. Conclusion. 87
4. MISE EN OEUVRE EXPERIMENTALE ET RESULTATS 88

4.1. Introduction. 88
4.2. Algorithme globale de l'application 88
4.3. L'application serveur. 90
4.3.1 Déclaration et d'initialisation des variables nécessaires. 91
4.3.2. Sous programme LMS. 91
4.3.3. Sous programme Ultrason (US). 93
4.3.4. Sous programme odométrie. 94
4.3.5. Sous programme lecture des capteurs du bras. 94
4.3.6. Sous programme Communication. 95
4.3.7. Sous programme tâche. 95
4.4. L'application client. 90
4.4.1. Traitement de l'image. 97
4.4.2. Calibration de la caméra. 104
4.4.3. Fusion multi sensorielle. 105
4.5. Stratégie de téléopération utilisée. 107
4.6. Simulation des missions. 108
4.6.1. Présentation de notre simulateur. 109
4.7. Résultats expérimentaux et validation 111
4.7.1. Résultas de la calibration de la caméra CCD. 111
4.7.2. Résultas de la simulation. 114
4.7.3. Résultas de la téléopération (validation réelle). 116
4.8. Conclusion. 122
CONCLUSION. 123
APPENDICE A.
APPENDICE B.
APPENDICE C.
APPENDICE D.
REFERENCES.

LISTE DES ILLUSTRATIONS, GRAPHIQUES ET TABLEAUX

Figure.1.1	Le système maître esclave de Mechanical Pantograph.	19
Figure.1.2	Système de téléopération développé par General Electric en 1960.	19
Figure.1.3	Système de téléopération de la deuxième génération.	20
Figure.1.4	Bras manipulateurs à retour d'effort : A gauche le site maître, à droite le site esclave.	21
Figure.1.5	Un gant de préhension.	22
Figure.1.6	Bras manipulateurs à retour d'effort de CEA.	22
Figure.1.7	Système de démantèlement d'installation nucléaire de CEA.	23
Figure.1.8	Le robot Pioneer à gauche, à droite sa station de contrôle.	24
Figure.1.9	Le système Predator de l'US Air Force à gauche, à droite sa station de contrôle.	25
Figure.1.10	Le robot Sojourner sur Mars.	25
Figure.1.11	Les deux robots de la mission "Mars Exploration Rovers ", à droite le robot SPIRIT, à gauche le robot OPPORTUNITY.	26
Figure.1.12	Le bras installé sur la navette spatiale canadienne.	26
Figure.1.13	Système de Télédiagnostique.	26
Figure.1.14	Illustration d'un système de téléchirurgie.	27
Figure.1.15	Illustration d'une téléchirurgie.	28
Figure.1.16	La première opération à distance fut par le Professeur Marescaux.	28
Figure.1.17	Le robot du projet ARPH.	29
Figure.1.18	Engin sous marin.	29
Figure.1.19	Structure générale d'un système de téléopération	30
Figure.1.20	Structure du système de téléopération non autonome.	32
Figure.1.21	Structure du système de téléopération à autonomie	33

	opérationnelle.	
Figure.1.22	Structure du système de téléopération à autonomie décisionnelle.	34
Figure.1.23	Caméra déportée – *eye to hand* -	35
Figure.1.24	Caméra embarquée- *eye in hand*-.	35
Figure.2.1	Structure du système de télérobotique.	38
Figure.2.2	La station de contrôle	39
Figure.2.3	Synoptique d'une communication client serveur.	41
Figure.2.4	PC embarquée.	42
Figure.2.5	Photo de la caméra CCD utilisée.	44
Figure.2.6	Photos du système de transmission HF.	44
Figure.2.7	Photo réelle du robot ROBUTER-ULM.	46
Figure.2.8	La carte MPC 555.	46
Figure.2.9	Capteur ultrasonore (émetteur récepteur).	49
Figure.2.10	Principe d'un télémètre à ultrason.	50
Figure.2.11	Le SICK LMS200.	51
Figure.2.12	Structure interne du LMS200	51
Figure.2.13	Angle de balayage du LMS200.	52
Figure.2.14	Principe d'un télémètre Laser à mesure de déphasage.	53
Figure.2.15	Mesure de phase entre le signal émis et le signal reçu.	53
Figure.2.16	Calcul de coordonnés d'un point connaissant la distance.	54
Figure.2.17	Codeur incrémentale.	55
Figure.2.18	Capteur de fin de cours.	55
Figure.2.19	Architecture Logiciel du Robuter_ULM.	56
Figure.2.20	Principe général d'une application utilisateur.	57
Figure.2.21	Structure générale d'une application bas niveau.	57
Figure.2.22	Principe général d'une application haute niveau.	58
Figure.2.23	Une fenêtre de développement de SynDEx V.5.	60
Figure.3.1	Les différents repères mis en jeu.	61
Figure.3.2	Les transformations de repères nécessaires.	62
Figure.3.3	Représentation dans le plan du robot mobile.	64
Figure.3.4	Roue conventionnelle orientable centrée.	65
Figure.3.5	Roue conventionnelle orientable décentrée.	65

Figure.3.6	Variables généralisées du bras ULM.	69
Figure.3.7	Paramètres de D&H d'un système articulé.	71
Figure.3.8	Paramètres modifiés de D&H d'un système articulé.	71
Figure.3.9	Paramètres de Denavet-Hartenberg du bras ULM.	72
Figure.3.10	Simulation et validation par MATLAB du MG du bras.	73
Figure.3.11	Les solutions possibles pour le MGI.	76
Figure.3.12	Modèle sténopé d'une caméra CCD.	78
Figure.3.13	Repères et projection perspective du modèle sténopé	78
Figure.3.14	Représentation des différents repères.	79
Figure.3.15	Représentation des paramètres intrinsèques de la caméra.	81
Figure.3.16	Le triangle semblable.	81
Figure.4.1	Diagramme fonctionnel du système de téléopération.	89
Figure.4.2	L'organigramme de l'application de téléopération.	90
Figure.4.3	L'organigramme de l'application serveur.	91
Figure.4.4	L'organigramme de gestion du LMS.	92
Figure.4.5	L'organigramme de gestion de la ceinture à ultrasons.	94
Figure.4.6	L'organigramme de lecture de l'odométrie.	95
Figure.4.7	L'organigramme de lecture des capteurs du bras.	96
Figure.4.8	Présentation de l'interface graphique.	97
Figure.4.9.	Etapes de traitement de l'image.	98
Figure.4.10	Image originale.	100
Figure.4.11	Image filtrée.	100
Figure.4.12	Résultat de Binarisation, à droite image non filtrée, à gauche image filtrée.	101
Figure.4.13	Voisinage d'un pixel.	102
Figure.4.14	Segmentation de l'image	104
Figure.4.15	Résultats de caractérisation.	105
Figure.4.16	Calibration de la caméra.	106
Figure.4.17	Représentation graphique des données des Ultrasons.	106
Figure.4.18	Représentation graphique des données de LMS.	107
Figure.4.19	Affichage des données du bras et de la plate forme mobile	107

Figure.4.20	La cible se trouve dans le champ du travail du robot.	108
Figure.4.21	La cible se trouve hors champ du travail du robot.	109
Figure.4.22	La barre de composant GLSene après son installation.	111
Figure.4.23	L'interface du simulateur SIMROBUTER.	112
Figure.4.24	Comparaison entre les coordonnées image mesurées et calculées.	114
Figure.4.25	Représentation des coordonnées image mesurées et celles calculées.	115
Figure.4.26	Les configurations possibles pour une position donnée.	116
Figure.4.27	Simulation d'une mission.	116
Figure.4.28	Collision entre le bras et la plate forme mobile.	116
Figure.4.29	Opération du Zoom et pivote sur la scène.	117
Figure.4.30	Vérification d'alignement de la caméra.	118
Figure.4.31	L'opérateur nous montre la cible.	118
Figure.4.32	Représentation du système de téléopération.	119
Figure.4.33	Le robot rapproche de la cible.	119
Figure.4.34	Début de mouvement du bras.	119
Figure.4.35	Le robot atteint la cible.	120
Figure.4.36	Le bras reprend sa configuration de départ.	120
Figure.4.37	Position du départ du robot vis-à-vis du panneau	121
Figure.4.38	Traitement des données sur le PC hôte.	121
Figure.4.39	La séquence à atteindre par le robot.	121
Figure.4.40	Déplacement du robot près de la cible.	122
Figure.4.41	Le premier point.	122
Figure.4.42	Position intermédiaire.	122
Figure.4.43	Le deuxième point.	122
Figure.4.44	Position intermédiaire.	122
Figure.4.45	Le troisième point.	123
Figure.4.46	Position intermédiaire.	123
Figure.4.47	Le quatrième point.	123
Figure.4.48	Retour à la position initiale.	123
Figure.B.1	Vue de coté du robot mobile.	128
Figure.B.2	Vue de l'avant du robot mobile.	128

Figure.B.3	Vue de dessus du robot mobile.	129
Figure.B.4	Décomposition du bras manipulateur.	129
Figure.B.5	Dimensions des axes.	130
Figure.B.6	débattements max de l'axe 4.	130
Figure.B.7	débattements max de l'axe 1.	130
Figure.B.8	débattements max de l'axe 6.	130
Figure.B.9	Champ de travail du bras ULM	131
Figure.B.10	Caractéristiques d'émission d'un télémètre ultrason.	131
Figure.B.11	Repère considéré du Robuter	132
Figure C.1	Les couches du modèle TCP/IP	136
Figure C.2	Format des paquets TCP/IP	136
Figure C.3	Schéma d'une matrice de caméra CCD	140
Figure C.4	Etapes d'acquisition d'une image	140
Figure.D.1	Le système de calibration de la caméra CCD.	146
Figure.D.2	Le système réel de calibration de la caméra CCD.	146
Figure.D.3	Exemple des prises de vues avec la mire de calibration.	147
Figure.D.4	Exemple des prises de vues avec la grille de calibration.	148

Tableau.3.1	Paramètres de Denavit-Hartenberg du bras ULM.	73
Tableau.4.1	Représentation Matricielle d'un pixel image et son voisinage.	99
Tableau.4.2	Résultats de la calibration de la camera.	114
Tableau.B.1	Repère lié au Robuter.	132
Tableau.B.2	Les positions des capteurs ultrasons dans le repère de Robuter	132
Tableau.B.3	Caractéristiques du LMS200.	133

INTRODUCTION

Le domaine de la robotique est un domaine de recherche très vaste, et en pleine expansion, il a connu ces dernières années des développements énormes grâce aux progrès dans le domaine informatique et aux avancées technologiques dans tous les secteurs scientifiques. Ces développements technologiques ont permis de passer de la génération des robots manipulateurs fixes à celles des robots mobiles, puis à celle des robots mobiles manipulateurs.

Un Robot Mobile Manipulateur est un robot composé d'une plate forme mobile et d'un bras manipulateur surmontant cette dernière. L'association de ces deux systèmes augmente la capacité de manipulation et le champ de travail de ce robot. Grâce à cette structure un bras manipulateur peut atteindre n'importe quel point se trouvant dans son environnement.

La combinaison entre un bras manipulateur fixe, un robot mobile, de calculateurs embarqués et d'un ensemble de capteurs extéroceptifs et proprioceptifs (tels que : des systèmes de mesure de distance et d'évitement d'obstacles, de systèmes de vision, de capteurs d'efforts, de systèmes de positionnement et de localisation), représente une révolution scientifique réelle. Ces outils permettent au robot d'avoir suffisamment d'informations sur son état interne et sur son environnement. Ils lui permettent de se localiser dans son environnement, d'appréhender ce qui l'entoure, de naviguer et d'éviter des obstacles se trouvant dans son chemin, d'agir sur l'environnement avec la précision et l'efficacité voulues, d'avoir un champ de travail illimité et surtout de prendre des décisions et d'avoir une certaine autonomie, pour s'acquitter de la meilleure façon possible des missions, souvent demandant beaucoup de précision lors de leur exécution.

La nature de ce système robotique le rend éligible pour des tâches exécutées à distance ou dans des milieux hostiles. Cette génération de robots a fait un grand succès, et a trouvé des applications dans plusieurs domaines (industriel, public, recherche, militaire, sanitaire,...). On parle aujourd'hui des robots explorateurs de l'univers, de la planète Mars, de la lune et d'océan, des

robots chirurgiens qui font les opérations les plus difficiles, des robots détectant et désactivant les mines et bombes, et des robots d'intervention qui peuvent être utilisés à la place des agents de la protection civile pour l'extinction d'incendie ou même l'évacuation des blessés.

La robotique d'intervention est un axe de recherche qui vise à faire une sorte d'extension du corps physique des opérateurs, par le biais de robots manipulateurs. Dans ce cadre, l'opérateur humain n'agit qu'indirectement sur le processus à commander. La robotique d'intervention est envisagée soit en *substitution,* où le robot doit se substituer totalement à l'homme dans l'exécution des tâches dangereuses, dans les milieux pollués ou les milieux hostiles, le rôle de l'homme se limite donc à diriger l'exécution de la tâche, ou bien en *coopération,* où le robot aide l'homme à exécuter des tâches impossibles ou dangereuses à mains nues, comme par exemple la maintenance des appareils placés dans des environnements hostiles (radioactifs ou chimiquement contaminés), ou l'exploration et l'étude de lieux inaccessibles (inspection de canalisations souterraines, l'exploration des planètes etc.).

Au CDTA, l'équipe de recherche **S**ystèmes **R**obotisés de **P**roduction du laboratoire de robotique et productique dispose d'un robot mobile manipulateur appelé Robuter_ULM, ce dernier est composé d'une plate forme mobile à quatre roues; deux roues motrices et deux roues foles, surmontée d'un bras manipulateur à six degrés de liberté doté d'une pince en buté. Ce robot est équipé d'un ensemble de capteurs extéroceptifs et proprioceptifs divers, il est doté d'un capteur odométrique pour la plate forme mobile, des capteurs de position pour les liaisons du bras, d'une ceinture à ultrason entourant la plate forme, d'un télémètre Laser placé en avant du robot, d'un capteur d'effort monté au niveau de la pince et d'une caméra CCD monochrome embarquée et installée sur la pince.

L'objet de notre travail est de téléopérer le robot mobile manipulateur en question. En effet, il s'agit de développer un système de contrôle et de commande qui permet de manipuler à distance au moyen du robot mobile manipulateur ROBUTER_ULM des objets se trouvant sur des parois planes situées devant le robot. On peut résumer le travail à accomplir par le cahier des charges suivant :

1. La source des difficultés que rencontre un opérateur placé en condition de téléopération, et qui n'est pas forcément un expert, réside dans l'écart fonctionnel existant entre l'Homme et la Machine. Donc en premier lieu il faut

établir une communication réseau basée sur le concept client serveur reliant le PC embarqué au PC hôte (communication client serveur entre Windows et linux via le réseau local du CDTA), tout on assurant la circulation du flux de données entre les deux machines pour l'échange de données.

2. Développement d'une application serveur du coté PC embarqué garantissant l'exploitation des ressources du robot par la gestion de ses capteurs et actionneurs (acquisition des informations capteurs et exécution des tâches) et l'échange de données avec le client.
3. Développement d'une application utilisateur qui permet à l'opérateur de gérer et de piloter depuis son poste de travail le robot. Cette application prend en charge les tâches élémentaires suivantes:

- La gestion d'une interface graphique de communication homme machine qui prend en charge et interprète les commandes de l'utilisateur.
- La gestion de l'application client pour l'envoi des commandes et la réception des retours sensoriels.
- L'implémentation des algorithmes nécessaires pour le traitement de l'image vidéo acquise par la caméra CCD, et l'extraction des informations utiles (la détection et la localisation des objets dans l'image).
- L'affichage et l'interprétation graphique des informations sensorielles.
- La fusion des informations sensorielles ramenées par les capteurs extéroceptifs et proprioceptifs du robot afin de générer les commandes et les trajectoires appropriées qui permettent l'exécution des missions assignées au robot
- Le développement d'un simulateur graphique virtuel pour les tests sur PC.

4. La Modélisation géométrique du système robotique et du système de vision pour établir les lois de commande adéquates.
5. La Validation sur site réel (environnement intérieur du laboratoire).

Ces points représentent les tâches à prendre en charge dans ce travail, qui seront amplement décrites dans les chapitres de ce manuscrit.

Pour assurer une meilleure présentation du travail effectué et garantir la clarté du mémoire, nous avons divisé ce manuscrit en quatre chapitres, mettant chacun en avant une contribution particulière de ce travail, qui sont présentés suivant l'ordre chronologique donné ci-après:

- Le premier chapitre présente le contexte dans lequel s'inscrivent nos présents travaux. Les ambitions ultimes de la téléopération ainsi que les applications envisagées sont précisées et permettent de situer ce document par rapport aux recherches faites et ceux qui sont en cours de réalisation. Un aperçu des systèmes de téléopération, un état de l'art, des définitions et une classification des types de téléopération sont mis en évidence.
- Le second chapitre décrit en détail la structure et l'architecture proposées pour le système de téléopération développé. Une description matérielle et logicielle détaillée du système robotique Robuter_ULM situé au CDTA, qui fait l'objet de notre étude, sera décrite afin de donner au lecteur une vision globale du travail réalisé.
- Le troisième chapitre aborde le problème de la modélisation mathématique du robot mobile manipulateur et du système de vision. Les changements de repères ainsi que les matrices de passages associés sont démontrés.
- Le quatrième chapitre consacre une partie à la présentation des applications développées et les différents modules les constituant. L'analyse et le traitement des informations issues des capteurs et la méthode de fusion sont suffisamment mis en évidence. Une description du simulateur développé est également présentée, il s'agit d'un démonstrateur pour des applications de téléopération. Le démonstrateur est basé sur l'utilisation d'un prototype virtuel du robot mobile manipulateur Robuter_ULM évoluant dans son environnement. La deuxième partie regroupe les résultats obtenus et les validations expérimentales réalisées.
- Une conclusion résumant le travail fait et discutant les résultats obtenus ainsi que des perspectives d'amélioration de ce travail, est donnée à la fin de ce document.

CHAPITRE 1
LA TELEOPERATION: ETAT DE L'ART

1.1 Introduction

Dans ce chapitre, nous présentons le contexte de notre travail, qui est la téléopération, autrement dit le contrôle et la commande à distance de systèmes robotiques, qui est une forme de télétravail (travail à distance). Nous reportons un bref état de l'art sur la téléopération, où nous présentons une définition, une synthèse et une classification par domaine d'application de la téléopération.

1.2 Définition

Le terme téléopération est composé de deux mots, « télé » de racine grecque signifie à distance, et le mot latin « opération » qui désigne l'exécution d'une certaine tâche. La téléopération désigne donc une mode d'exécution d'une tâche, d'une action, d'un mouvement ou d'un travail à distance [1].

En robotique, la téléopération peut être appelée télérobotique, elle désigne l'ensemble des principes et techniques qui permettent à l'opérateur humain d'accomplir une tâche à distance, au moyen d'un système robotique d'intervention (dispositif esclave), commandé et contrôlé à partir d'une station de contrôle (dispositif maître), à travers un canal de communication en se basant sur des retours visuels.

1.3 Historique de la téléopération

Ce bref historique met en évidence l'origine des systèmes de Téléopération, ainsi que l'évolution technologique, qui a conduit jusqu'aux applications actuelles.

La naissance de cette branche de science est sans doute liée avec les premiers automates conçus et robots inventés, et elle a bien profité des développements des systèmes de communication et de transmission actuels et surtout de l'avènement de l'informatique.

Le concept de travail à distance n'est pas nouveau, les chercheurs et savants travaillaient depuis longtemps dans ce champ, et si on cherche dans l'historique on trouve que les premiers systèmes de téléopération sont assez antiques et datent à des temps bien primaires.

L'homme au cours du temps a développé plusieurs systèmes qui ont la notion de travail de loin: on trouve par exemple la téléphonie et la télégraphie qui désignent tout simplement le dialogue à distance, la télévision qui signifie la vision à distance, la télescopée,... et bien d'autres.

On peut diviser le développement qu'a subit la téléopération au cours du temps en deux parties, avant et après l'évènement de l'informatique.

La première génération de télémanipulateurs est basée sur le principe maître esclave, où les bras manipulateurs sont géométriquement semblables et caractérisés par une réversibilité de fonctionnement. Dans ce procédé, l'opérateur, qui exécute le geste, perçoit le système esclave et l'environnement naturellement. En effet, il s'agit des systèmes mécaniques motorisés, commandés manuellement par des systèmes mécaniques ou électronique ou même hydrauliques (le cas des cases).

Les systèmes de télémanipulation de cette époque ont été développés initialement dans le secteur nucléaire, pour permettre la commande gestuelle des robots manipulateurs avec de bonnes propriétés de dextérité et de transparence. A titre d'exemple, on trouve:

- En 1945, pratiquement le premier système de téléopération maître esclave développé par *Mechanical Pantograph* pour la manipulation des produits radioactifs [2].

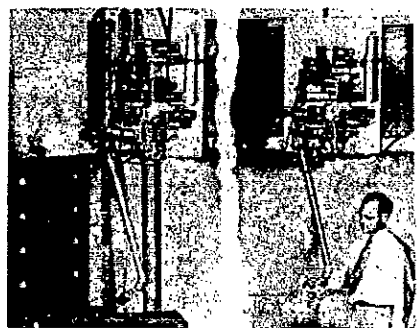


Figure 1.1 : Le système maître esclave de
Mechanical Pantograph [2].

- En 1954, Raymond Goertz et ses collègues du Argonne National Laboratory de Chicago, l'un des pionniers de cette innovation ont développé un système de téléopération électromécanique maître esclave pour les besoins de recherche scientifique [3].
- En 1960, un système de téléprésence composé de deux bras manipulateurs (figure 1.2) a été développé dans General Electric [2].

Figure 1.2 : Système de téléopération développé
par General Electric en 1960 [2].

La deuxième génération de systèmes de téléopération est née avec les calculateurs numériques. Ces calculateurs ont permis d'intégrer dans la chaîne de commande des robots des lois plus performantes qui demandent beaucoup de temps de calcul et de précision, ils ont permis également de créer le concept de

Après l'avènement des ordinateurs, une nouvelle génération de systèmes de téléopération apparaît sous forme de téléopération Assistée par Ordinateur (TAO). Elle se caractérise par l'introduction d'un calculateur dans la commande, permettant l'émergence d'une grande variété de modes semi-automatiques, basés sur un partage de la commande entre l'opérateur et le calculateur.

En général, les systèmes de cette génération sont composés d'une station de contrôle pour surveiller et assurer l'exécution des manipulations, là où se trouve l'opérateur humain, d'un système de communication et d'un système robotique d'exécution qui, dans la plus part des cas, est doté d'un ordinateur embarqué.

Cette génération a été appliquée et a démontré ses compétences avec succès dans plusieurs domaines autres que le domaine nucléaire, comme l'exploration de l'univers et le domaine médical.

La figure1.3 est un très bon exemple illustrant cette structure. Elle représente un opérateur dans une station de commande qui manipule un bras manipulateur distant.

Figure 1.3 : Système de téléopération de la deuxième génération [3].

Les deux dernières décennies ont connu des progrès technologiques importants dont les domaines de robotique, traitement d'image et de graphisme; elles sont aussi caractérisées par l'avènement de l'Internet et ses applications qui a offert beaucoup de perspectives pour la téléopération. Ces progrès ont permis de donner naissance à une nouvelle génération de téléopération, c'est la téléopération assistée par la réalité virtuelle et la réalité augmentée. Ces deux nouveaux concepts (réalité virtuelle et réalité augmentée) ont été introduits

récemment aux vocabulaires scientifiques, ils ne sont à la mode que depuis quelques années.

On désigne par la Réalité Augmentée (RA) la combinaison de la vraie scène visualisée par l'utilisateur et d'une scène virtuelle produite par l'ordinateur [4]. Pour la Réalité virtuelle (RV) il existe dans la littérature autant de définitions qu'il y a de champs d'application de la Réalité Virtuelle. On peut citer parmi ces définitions:

- Définition d'un système de RV [5] : un système de réalité virtuelle est une interface qui implique la simulation en temps réel et des interactions via de multiples canaux sensoriels. Ces canaux sensoriels sont ceux de l'homme, vision, audition, toucher, odorat et goût.
- Définition fonctionnelle de la RV [6] : La réalité virtuelle va permettre de s'extraire de la réalité physique pour changer virtuellement de temps, de lieu et (ou) de type d'interaction: interaction avec un environnement simulant la réalité ou interaction avec un monde imaginaire ou symbolique.
- Définition technique de la RV [6]: La réalité virtuelle est un domaine scientifique et technique exploitant l'informatique et des interfaces comportementales en vue de simuler dans un monde virtuel le comportement.

Les systèmes maîtres de commandes dans les stations de contrôles sont remplacés par des technologies à retour d'effort et des périphériques haptiques: bras manipulateurs à retour d'effort, joystick à retour d'effort, exosquelette à retour d'effort, gant de préhension à retour d'effort. Voici des illustrations montrant des exemples de cette technologie:

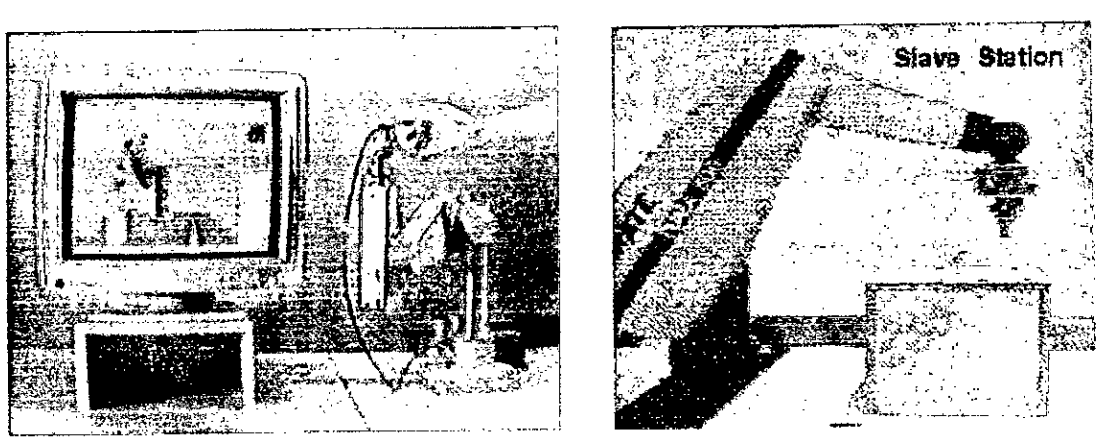


Figure 1.4 : bras manipulateur à retour d'effort: A gauche, le site maître; A droite le site esclave [3].

Exemple de gant haptique sur la photo suivante:

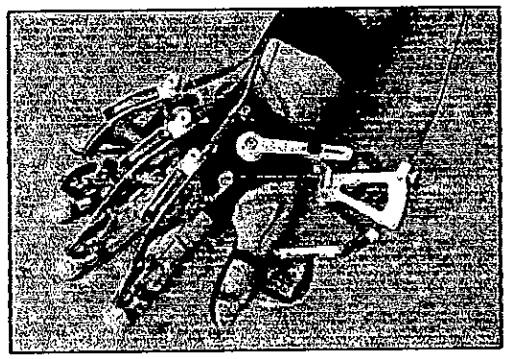


Figure 1.5 : Un gant de préhension [3].

Un autre exemple d'un bras manipulateur à retour d'effort réalisé dans les laboratoires de CEA, sur la photo apparaît ci-après un opérateur qui manipule un bras de commande et le bras distant reproduit les mêmes gestes et mouvements.

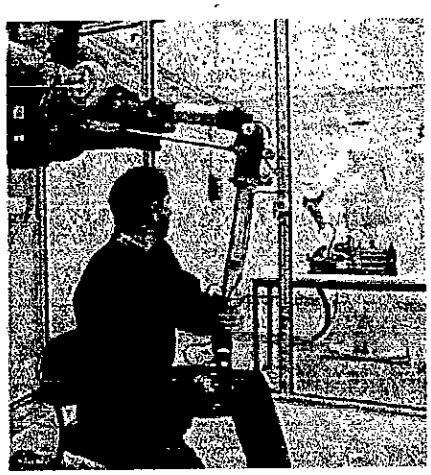


Figure 1.6 : bras manipulateurs à retour d'effort de CEA [7]

1.4 Intérêts et domaines d'application

La téléopération est un axe de recherche actuel, elle suscite beaucoup d'intérêts et trouve son emploi dans diverses disciplines ou activités de la vie moderne. Elle concerne la réalisation de travaux dépassant les capacités physiques de l'homme tels que la manipulation d'objets lourds, ou présentant un danger, cas de robots industriels, ou bien des travaux qui sont réalisés sur des sites inconnus ou hostiles à l'homme (tels que: les robots d'exploration martiens [8], robots industriels, les robots sous marins [9], les robots permettant la détection

et la réparation de fissures dans des conduits radioactifs, les robots chirurgiens, les robots destinés à l'exploration de l'univers, etc.). On peut citer par la suite les grands domaines d'application de la téléopération :

1.4.1 Le domaine nucléaire

Le domaine nucléaire a été le premier domaine à stimuler le développement des systèmes de téléopération. Les premières téléopérations sont issues de la nécessité pour l'homme de se protéger dans la manipulation de produits nucléaires.

Puisque les zones nucléaires sont très dangereuses, il est très commode de remplacer l'homme par des systèmes robotiques dans l'exécution de certaines tâches. L'industrie nucléaire s'intéresse généralement aux applications suivantes :

- La manipulation de produits radioactifs, comme par exemple le système électromécanique maître esclave développé en 1954 par Raymond Goertz et ces collègues dans le laboratoire Argonne National Laboratory de Chicago pour la manipulation des matériaux radioactifs [2].

- Le montage et le démontage des installations nucléaires. A titre d'exemple on peut citer celui représenté dans la figure ci-après, qui illustre un système développé au CEA qui est entrain de démanteler une installation nucléaire.

Figure 1.7 : Système de démantèlement d'installation nucléaire de CEA [4].

- La maintenance des installations, à titre d'exemple le système TAO2000 développé par CEA [4]. Les systèmes robotiques utilisés sont bien adaptés aux tâches à accomplir avec des missions bien définies.

On trouve aussi le robot Pioneer qui a été conçu pour inspecter et diagnostiquer l'état du réacteur nucléaire de TChernobyl [10].

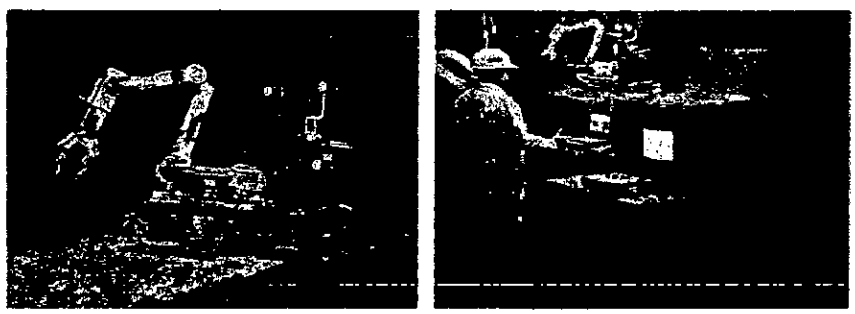


Figure 1.8 : Le robot Pioneer à gauche, à droite sa station de contrôle [10].

- L'intervention en cas d'accident, par exemple pour l'extinction d'un incendie, ou l'évacuation des personnes accidentées.

1.4.2 Le domaine militaire

La téléopération intervient dans ce champ pour :

- La désactivation de mines.
- L'observation de territoires ennemis ainsi que la télécommande d'engins tels que les chars et les avions (les appareils aériens sans pilote).

Les premiers engins aériens téléopérés étaient les drones, aussi appelés *Véhicules Pilotés à Distance* (RPV). Ils étaient utilisés pour l'entraînement des appareils anti-aériens [10].

Durant les années soixante, la NASA a mis en place un programme de développement de *Véhicules de Recherche Pilotés à Distance* (RPRV). A la différence des drones, qui étaient généralement de taille réduite, les RPRV étaient des avions habités, modifiés pour être contrôlés à distance [10].

Aujourd'hui, les *Véhicules Aériens Inhabités* (UAV) sont les engins téléopérés les plus communément utilisés. Les UAV modernes sont pilotés à distance grâce à une liaison radio ou satellitaire et sont utilisés pour des tâches telles que la reconnaissance ou l'identification de cibles pour le guidage des missiles. Egalement, différents UAV ont été utilisés pour le combat, tel que l'US Navy Pioneer et l'US Air Force Predator (figure 1.9) [10].

Figure 1.9 : Le système Predator de l'US Air Force à gauche, à droite sa station de contrôle [10].

1.4.3 Le domaine spatial

Les principales applications de la téléopération dans ce domaine sont l'exploration et la maintenance de satellites et de stations spatiales orbitales. Parmi les travaux réalisés dans ce sens on cite:

- Le projet qui fut développé en 1971 par l'Union Soviétique pour commander à distance un robot mobile sur la Lune [6].
- En 1997, le projet de la mission "*Mars Pathfinder*" avec le robot SOJOURNER sur la planète Mars.

Figure 1.10 : Le robot SOJOURNER sur Mars [6].

- Le projet de la mission " *Exploration Rovers* " avec les deux robots jumeaux SPIRIT et OPPORTUNITY en 2004 sur la planète Mars. En effet, en janvier 2004, deux grands robots d'exploration équipés d'éléments scientifiques perfectionnés se sont posés sur le sol martien et explore la surface de la planète à la recherche de preuves supplémentaires sur la présence d'eau liquide sur Mars dans un

lointain passé. Cette mission est la plus sophistiquée jamais envoyée vers la planète rouge par la NASA.

Figure 1.11 : Les deux robots de la mission "Mars Exploration Rovers", à droite le robot SPIRIT, à gauche le robot OPPORTUNITY [6].

- Le projet de la navette spatiale canadienne, où ils ont installé un bras manipulateur de 17m de longueur et de 7 degré de liberté, pour les opérations d'assemblage et de maintenance en collaboration avec les astronautes.

Figure 1.12 : Le bras installé sur la navette spatiale canadienne [6].

1.4.4 Le domaine médical

Dans le domaine médical, trois axes ont trouvé des applications: la Télédiagnostique, la chirurgie et l'assistance aux personnes handicapées ou âgées, aussi bien à domicile qu'à l'hôpital ou en centre spécialisé.

- Télédiagnostique : ces systèmes sont conçus pour aider les médecins à faire des diagnostiques à distance. Un tel système est schématisé par :

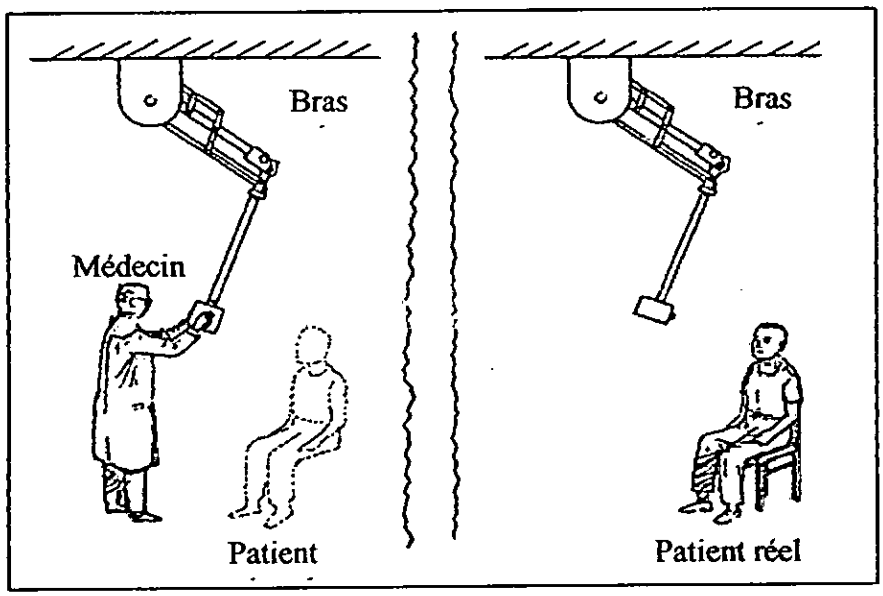


Figure 1.13 : Système de Télédiagnostique [3].

- La Téléchirurgie: pour les besoins de faire des opérations qui demandent une très grande précision et des interventions chirurgicales à distance, les chercheurs développent des systèmes exécutant les missions assignés avec fidélité.

Figure 1.14 : Illustration d'un système de téléchirurgie [3].

Cette figure illustre un médecin qui utilise un robot maître (la photo à gauche) qui est entrain de téléopérer un patient (la photo à droite).

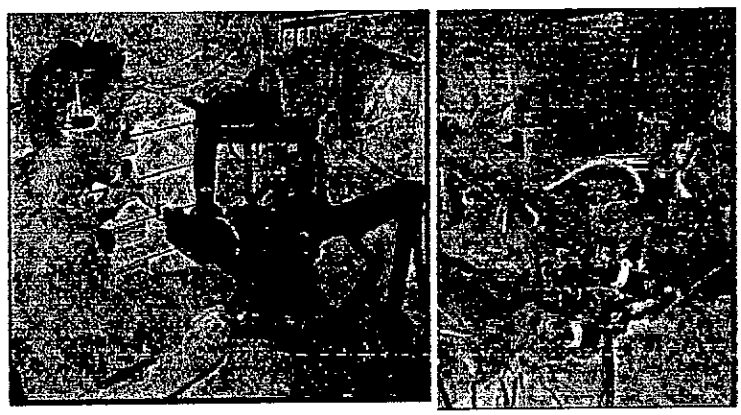


Figure 1.15 : Illustration d'une téléchirurgie [3].

Plusieurs projets sont développés dans cette voie, on cite par exemple:

- L'exemple le plus connu de robots téléopérés est sans doute les robots Da Vinci fournis et développés par la firme Computer Motion. Ce sont les premiers systèmes dans le domaine. En septembre 2001, le professeur Marescaux (IRCAD) a téléopéré ce système depuis New York en réalisant avec succès une cholécystectomie (ablation de la vésicule biliaire) sur une patiente hospitalisée à Strasbourg [3].

Figure 1.16 : La première opération à distance réalisée par le Professeur Marescaux [3].

- Dans la troisième voie, de nombreux projets ont été étudiés et mis en oeuvre afin d'aider des handicapés à mieux vivre. On cite par exemple les projets SPARTACUS et MASTER du CEA qui ont permis l'automatisation de quelques tâches quotidiennes à partir d'une commande utilisant les mobilités disponibles de l'handicapé, le projet ARPH (*Assistance Robotisée aux Personnes Handicapées*)

[11], [12], [13], ainsi que le projet VAHM (Véhicule Autonome pour Handicapés Moteur) [14].

Le robot ARPH est composé d'une plate-forme mobile sur laquelle est installé un bras manipulateur. Son but est d'augmenter l'autonomie des personnes à mobilité réduite.

Figure 1.17 : Le robot du projet ARPH [13].

1.4.5 Le domaine d'exploration sous marine

La téléopération est le système le plus fiable qui peut aider les chercheurs à explorer les profondeurs de l'océan. La figure suivante illustre un exemple d'engin sous marine. Cet engin est composé d'un bras manipulateur et d'un pc embarqués, il a comme mission l'inspection, la construction et la maintenance de conduites, structures, câbles sous-marins voire même des investigations scientifiques (épaves, espèces marines, etc.).

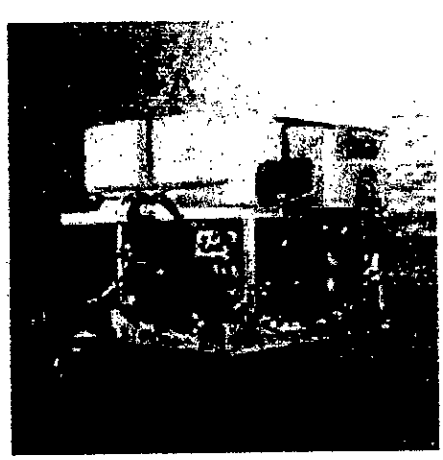


Figure 1.18: Engin sous marin [3].

1.5 Types de téléopération

La téléopération, quant à elle, se caractérise par une action à distance sur un environnement. L'opérateur se situe dans un poste de commande fixe et pilote un système évoluant à distance. Ce dernier renvoie à l'opérateur des informations sur son état et sur celui de l'environnement grâce à différents types de capteurs.

La structure générale d'un système de téléopération (figure 1.19), comprend trois parties essentielles :

1. La station de contrôle où se trouve l'opérateur.
2. Le système téléopéré qui représente l'outil d'exécution.
3. Le système de communication qui va établir la communication entre les deux modules.

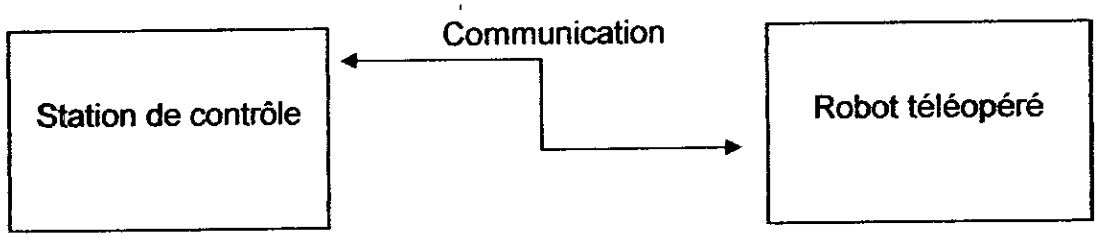


Figure 1.19: Structure générale d'un système de téléopération

La structure développée de cette architecture peut prendre plusieurs formes suivant le matériel utilisé et les besoins imposés.

Il existe Plusieurs types de téléopération. Ces téléopérations sont classifiables suivant plusieurs critères:

1.5.1 Classification suivant la nature de liaison

La fiabilité d'un télétravail dépend fortement de la nature du canal de transmission. Trois types de liaison peuvent être utilisés pour établir une communication entre le robot d'intervention et l'opérateur [15].

1.5.1.1 Liaison mécanique

Cette première méthode est sans doute la plus ancienne, elle consiste à transmettre les ordres de l'opérateur au robot au moyen d'une liaison mécanique permettant de reproduire le mouvement souhaité. Ce type est restreint à des missions de courte distance.

1.5.1.2 Liaison électrique filaire

Dans ce type de transmission, la liaison mécanique est remplacée par des câbles électriques. Les câbles téléphoniques sont les plus utilisés et les plus répondus parce qu'ils permettent d'utiliser une communication de type réseau comme l'Internet, et nul ne peut ignorer la puissance et l'efficacité de cette technologie, puisque elle peut s'étendre à plusieurs Kilomètres sans perte d'information.

On cite comme exemple le système ARITI (Augmented Reality Interface for Telerobotic applications via Internet) [16], qui est le premier système en France de téléopération de robot via Internet. Il est accessible sur le site Web du laboratoire depuis 1998 au http://lsc.cemif.univ-evry.fr:8080/Projets/ARITI, et sur le site de la NASA sur le lien http://ranier.oact.hq.nasa.gov/telerobotics_page/realrobots.html depuis janvier 2000.

Le système ARITI est réalisé dans le but de permettre à un opérateur humain de contrôler et de commander un robot à distance. L'interface Homme Machine (IHM) réalisé est basée sur le concept de la Réalité Mixée, regroupant ainsi la Réalité Virtuelle (RV) et la Réalité Augmentée (RA) [11].

1.5.1.3 Liaison sans fils

Pour ce type, et afin d'assurer la transmission de données entre l'opérateur et le robot plusieurs techniques sont utilisées: la radio fréquence, l'infrarouge, les ondes acoustiques. Cette technique présente l'avantage d'être moins coûteuse par rapport aux autres techniques.

1.5.1.4 Liaison mixte

Ce type de télétravail utilise pour la communication des liaisons de type filaire et d'autre sans fil. De cette façon, il profite des avantages des deux techniques avec un coût réduit et de bonnes performances.

1.5.2 Classification suivant le type d'interaction [15]

Une deuxième classification de la téléopération est possible en considérant le mode utilisé. Ce mode correspond au type d'interaction qui existe entre l'opérateur et le robot. Selon le mode d'interaction, le rôle de l'opérateur au sein de la téléopération et le niveau d'autonomie du robot d'intervention sont différents.

1.5.2.1 Robot d'intervention non autonome

Il s'agit réellement d'une relation maître esclave, ou le robot téléopéré exécute les ordres qui proviennent de l'opérateur maître. Le robot est alors défini comme non autonome puisqu'il est entièrement dépendant de la communication qui le relié avec l'opérateur. Cette structure est bien illustrée ci-après :

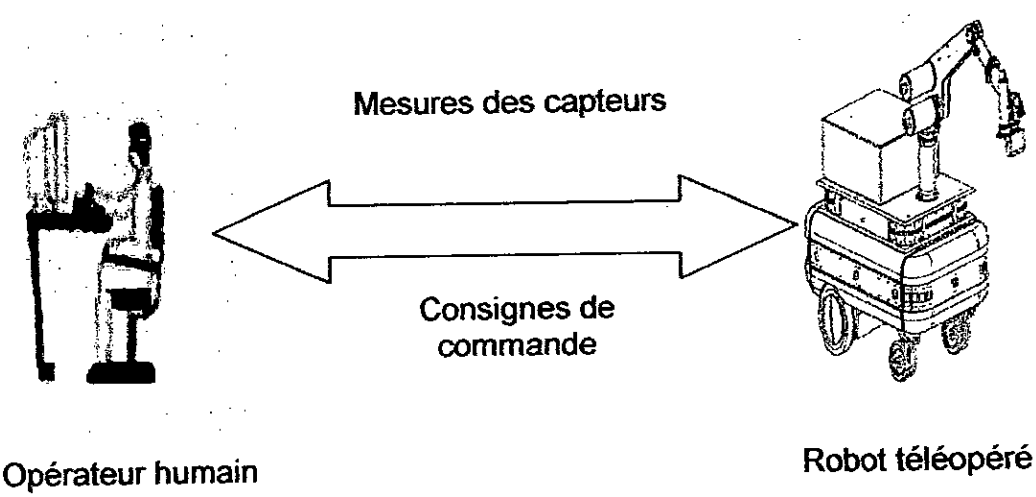


Figure 1.20 : Structure du système de téléopération non autonome [15].

1.5.2.2 Robots d'intervention à autonomie opérationnelle

Ce mode rend le robot semi autonome, c'est-à-dire le robot ne prend pas de décisions concernant la mission et la stratégie utilisée mais il réalise l'asservissement de la tâche, la génération de trajectoire et aussi l'évitement d'obstacle. L'opérateur de son coté lui assigne la mission que doit faire le robot, comment exécuter cette mission et quelle est la stratégie utilisée.

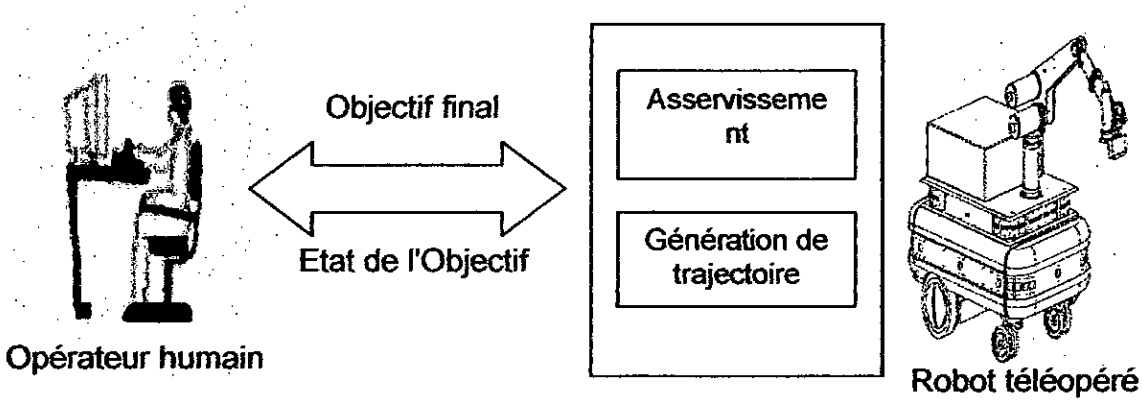


Figure 1.21 : Structure du système de téléopération à autonomie opérationnelle [15].

1.5.2.3 Robots d'intervention à autonomie décisionnelle

Dans ce mode, l'opérateur est vu comme un superviseur. L'intervention de l'opérateur se limite dans ce cas à la désignation d'objectifs qui seront réalisés par le robot. Par exemple, l'opérateur peut désigner sur écran un objet à atteindre par le robot. Un autre exemple consiste à désigner dans un haut niveau une tâche que doit réaliser le robot (saisir objet, déposer objet, etc.). Le robot est équipé par des ordinateurs embarqués et il est doté d'un système de raisonnement qui lui permet d'exécuter des tâches sans le guide de l'opérateur. Ce concept vise une exploitation optimale du côté du robot, mais la présence d'un opérateur reste obligatoire pour limiter les risques de défaillances fatales.

La figure suivante illustre la structure de ce mode de téléopération.

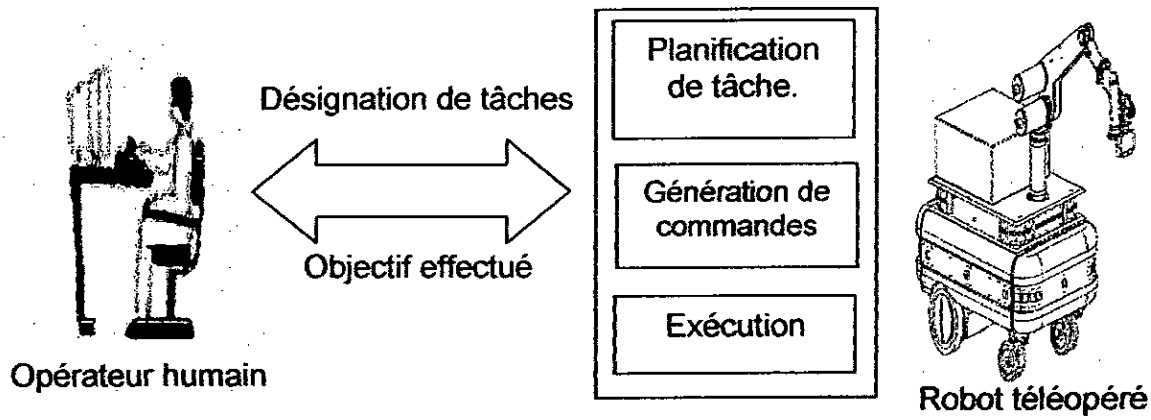


Figure 1.22 : Structure du système de téléopération à autonomie décisionnelle [15].

1.5.3 Classification suivant la position de la caméra

L'emplacement de la caméra par rapport à l'organe terminal du robot et à l'objet cible, dépend fortement de l'application assignée au robot. Ainsi, les aspects de la cinématique du système et des lois de commande associées, vont dépendre de cet emplacement.

On distingue deux types de configurations pour l'intégration d'une caméra dans un système de commande référencié vision [17] [18] : la caméra peut être déportée (eye to hand, Figure.1.23), c'est à dire montée sur un support fixe ou mobile et regardant à la fois le robot et l'objet d'intérêt, ou embarquée (eye in hand, Figure.1.24), c'est à dire montée sur l'organe terminal du robot et regardant l'objet d'intérêt. Le lecteur pourra se référer aux travaux de Brad Nelson [18] sur les stratégies de placement des capteurs visuels.

1.5.3.1 Caméra déportée "eye to hand "[19]

Dans cette configuration, la caméra 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 fixée sur un support fixe ou mobile. L'information visuelle acquise 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 du robot lui même.

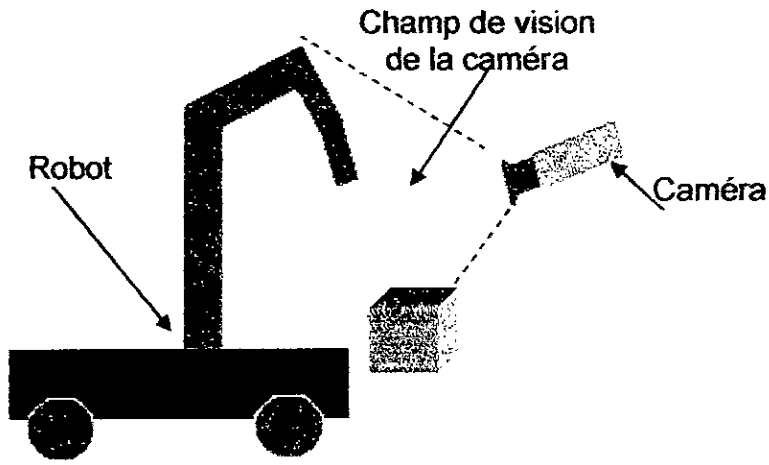


Figure 1.23 : Caméra déportée – *eye to hand* - [18].

1.5.3.2 Caméra embarquée "eye in hand"[19]

Dans cette structure la caméra se trouve attachée à l'organe terminal du manipulateur (sur le poignet) de manière à ce que les objets situés dans l'espace de travail soient dans son champ de vision (Figure.1.24), donc il nous permet d'obtenir une vue de la scène se trouvant devant la pince.

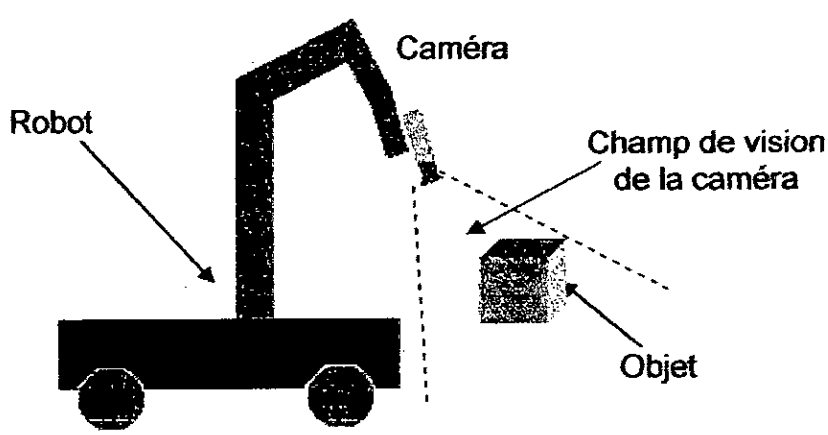


Figure 1.24 : Caméra embarquée- *eye in hand*- [18].

Le choix de la configuration de la caméra dépend essentiellement de la tâche à exécuter. Par exemple, pour un suivi de cible, il est judicieux de placer la caméra sur le robot de manière à ce que l'image vue soit sensiblement toujours la

même quelques soient les mouvements du robot et de la cible. Il peut être intéressant de déporter la caméra pour observer à la fois l'outil et l'objet d'intérêt pour une tâche de positionnement.

1.6 Conclusion

Dans ce chapitre nous avons présenté un bref état de l'art afin de donner au lecteur une vue globale sur le contexte de notre travail. Nous avons donné les définitions, les termes et les concepts nécessaires pour la compréhension de la thématique abordée. Des exemples de travaux réalisés dans ce domaine ont été présentés afin de positionner notre travail par rapport à l'état de l'art international. Dans le chapitre suivant nous allons voir une représentation et description de notre système de téléopération.

CHAPITRE 2.
DESCRIPTION DU SYSTEME DE TELEOPERATION

2.1 Introduction

Dans le chapitre précédent, nous avons parlé des systèmes de téléopération et nous avons donné quelques définitions de base et exposé des architectures de téléopération. Dans ce chapitre nous verrons la structure de notre système de téléopération, représentée par les différents modules qui la constituent tout en détaillant l'architecture, le rôle et l'importance de chaque partie.

2.2 Structure de notre système de téléopération

La structure adoptée pour notre cas doit répondre aux besoins suivants:

- Le système supporte l'échange d'informations entre le site maître et le site esclave.
- L'échange d'informations se fait par voie réseau entre un PC sous Linux et un autre sous Windows.
- Le traitement et l'analyse d'image se fait sous un PC hôte, et la transmission de l'image par un module sans fil HF.
- Le système doit posséder une IHM (Interface Homme Machine) conviviale, qui peut être utilisée facilement par des utilisateurs pas forcément spécialistes du domaine de la robotique ou de l'informatique.
- L'interface doit fournir à l'opérateur toutes les informations nécessaires concernant l'état du robot et la scène ainsi que tous les moyens nécessaires pour agir sur le robot.
- Le système doit être flexible et modulaire, dans la mesure où il doit être ouvert à d'éventuelles modifications.

Pour répondre à tous ces besoins, et par rapport aux problématiques posées et aux matériels disponibles, nous avons adopté la structure suivante:

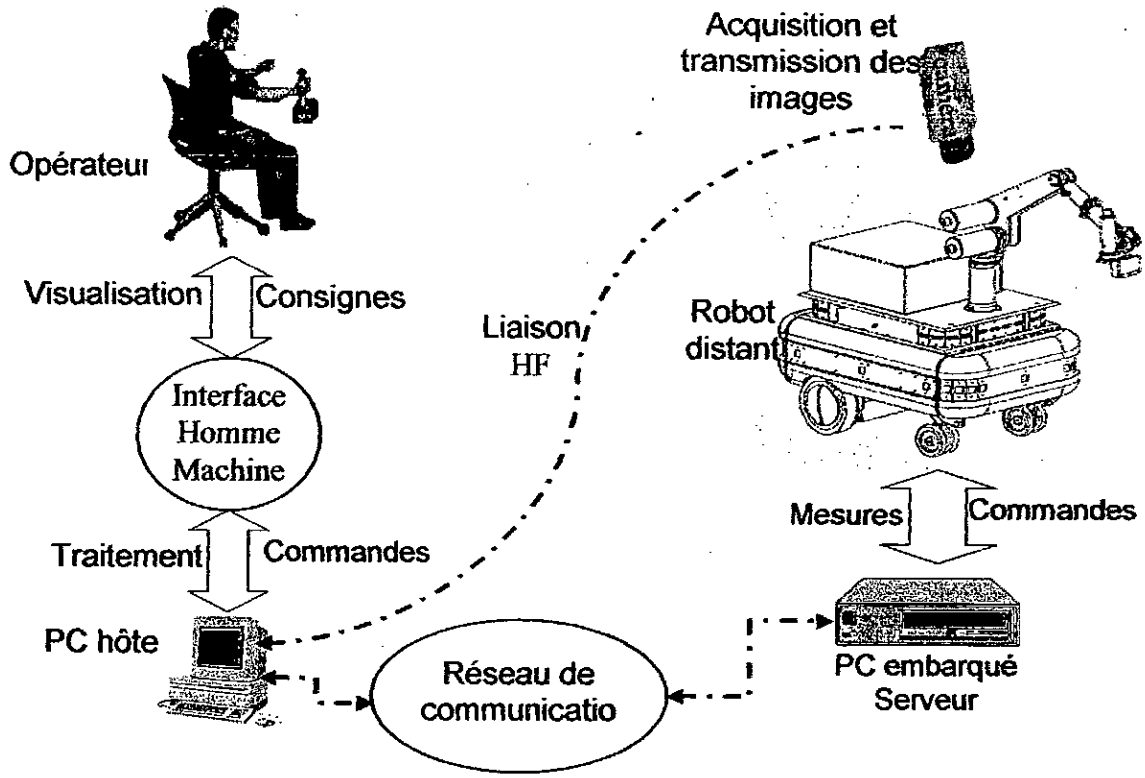


Figure 2.1 : Structure du système de télérobotique.

2.3 Description des différents modules du système de téléopération

L'utilisateur entre en communication indirecte avec le robot, par l'intermédiaire du PC hôte qui représente le Client (vis-à-vis de la communication client serveur) et grâce à une interface de communication Homme machine, il agit sur le robot et décide comment ce dernier doit se déplacer et quelle tâche doit-il effectuer, tout en ayant la possibilité de recevoir des informations sur son état afin de contrôler son mouvement grâce à un système imageur qui a pour rôle l'acquisition des images concernant le robot et son environnement, Le PC embarqué, qui joue le rôle d'un serveur reçoit les informations qui sont envoyées par l'opérateur via un réseau de communication sans fil de type LAN, élabore les lois de commande convenables pour le robot, assure leur exécution, et reçoit les informations de retour venant des capteurs.

Les différents modules constituant le système de téléopération présenté à la figure ci-dessus sont exposés ici avec un peu de détail en décrivant le rôle et les spécificités technologiques de chaque module.

2.3.1 Le PC hôte (client)

La station de commande est un PC de bureau ordinaire, avec un processeur Intel PintiumIV 2.8GHZ et 512Mo de RAM sous lequel est installé le système d'exploitation Windows XP. Le PC hôte qui représente le *Client,* est la liaison entre l'utilisateur et le robot distant, il permet à l'utilisateur de se connecter à distance au robot et le commander via une interface graphique. Il assure les tâches suivantes :

- La gestion de la communication depuis et vers le robot, par l'envoi des consignes de l'utilisateur et la réception des informations captées.
- La réception, l'affichage et le traitement des images vidéo issues du système imageur.
- La réception et l'affichage des différentes informations des capteurs du robot.
- La fusion multisensorielle des différentes informations reçues (images vidéo, informations des capteurs).
- La prise en charge et l'interprétation des commandes données par l'utilisateur.
- La gestion d'une interface de communication homme machine de type graphique (GUI pour *Graphical User Interface*).

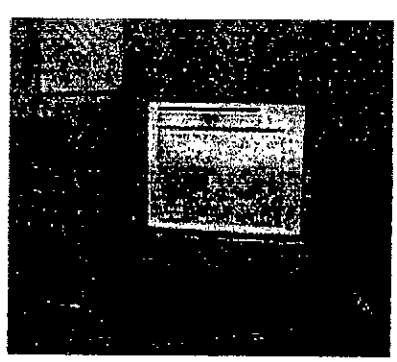


Figure 2.2 : La station de contrôle.

2.3.2 L'interface Homme/Machine

En téléopération, l'Interface Homme Machine (IHM) est un élément essentiel. Il offre la possibilité de communiquer avec le robot distant. Il permet de

traduire les volontés de l'opérateur en consignes pour le robot. Cette interface assure les fonctionnalités suivantes:

- L'affichage des images vidéo du robot et son environnement qui offre la possibilité de voir l'exécution des commandes en temps réel.
- L'affichage des différentes caractéristiques dynamiques du robot (vitesse, orientation, position).
- La représentation 3D virtuelle du robot, le plus proche de la réalité pour pouvoir faire de la programmation avec un robot virtuel.
- La prise en charge des consignes provenant de l'utilisateur.

2.3.3 Communication Réseau (Liaison client /serveur)

Le principal problème de l'humain en situation de téléopération provient de l'appauvrissement sensoriel résultant de la séparation entre l'entité qui commande l'action (l'humain) et celle qui l'exécute (le robot).

La communication doit exploiter les cartes réseau installées sur les deux PC et les modules de transmission WIFI (BrizeNet). Donc notre application de communication sera basée sur une architecture client/serveur qui utilise le mécanisme des Sockets du protocole TCP/IP.

2.3.3.1 Communication par Socket

Les Sockets sont le mécanisme fondamental de communications sous TCP/IP. Ils permettent des communications au sein du même système comme vers l'extérieur. La création d'une Socket est réalisée par l'obtention d'un descripteur sur lequel il est possible de lire et écrire comme pour un fichier.

2.3.3.2 Format d'un Serveur TCP/IP

Les opérations à réaliser de la part du serveur sont:

Création et ouverture d'une Socket en mode flux (TCP/IP): socket(AF_INET, SOCK_STREAM, 0)

Configurer l'adresse et le port: bind() Configurer le nombre d'écoutes: listen()

Dans une boucle

Accepter une connexion: accept()

Recevoir des données: recv() Emettre des données: send()

Déconnecter : shutdown()

Fin

Fermeture de la Socket : closesocket()

2.3.3.3 Format d'un Client TCP/IP

Les opérations à réaliser de la part du client sont:

Initialisation

Ouverture d'une Socket en mode flux (TCP/IP): socket(AF_INET, SOCK_STREAM, 0)

Connexion: connect()

Emettre des données: send()

Recevoir des données: recv()

Fin

Déconnecter : shutdown(). Fermeture de la Socket : closesocket()

On peut résumer tout ça par le schéma synoptique suivant:

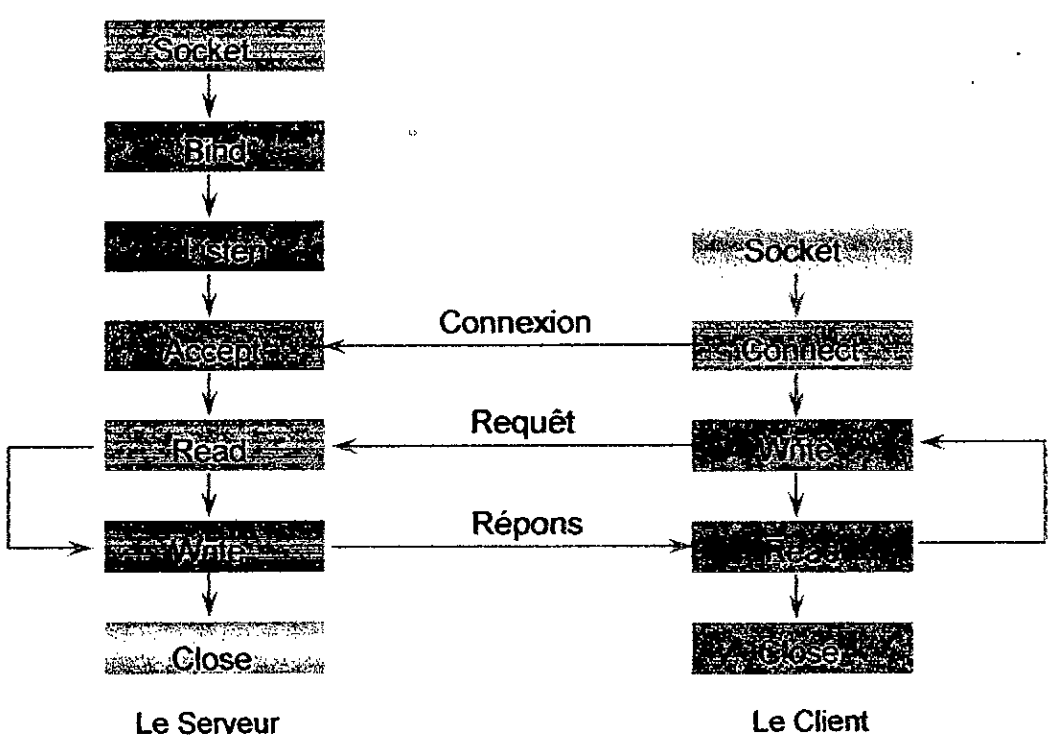


Figure 2.3 : Synoptique d'une communication client serveur

Le lecteur est invité à consulter l'annexe B, pour plus d'informations et détails concernant les réseaux et leurs architectures.

2.3.4 Le PC embarqué (serveur)

Le robot est doté d'un PC embarqué nommé fonctionnellement "Serveur". Ce module accomplit les tâches suivantes:

- Recevoir les commandes envoyées par le programme *PC_Client* et les interpréter pour que le robot puisse exécuter les opérations désirées par l'utilisateur (mouvement du robot et de la caméra).
- Recevoir les informations sur l'environnement du robot, les interpréter et les envoyer au programme *PC_Client* (donc l'utilisateur).
- Générer des lois de commande pour le robot et assurer l'exécution des tâches.

Le protocole réseau utilisé est celui de TCP/IP en mode connecté. Ce mode nécessite l'établissement d'une connexion entre deux machines avant de pouvoir échanger les données.

Figure.2.4 : PC embarquée.

2.3.5 Système imageur

La vision occupe une place privilégiée chez les êtres vivants qui ont des yeux. Il n'est pas étonnant que les scientifiques essaient de le reproduire artificiellement à l'aide de caméras, d'ordinateurs et d'algorithmes, c'est pour cela que l'on appelle la vision artificielle par ordinateur.

La vision artificielle par ordinateur est la science qui développe les bases algorithmiques et théoriques par lesquelles l'information utile relative à l'environnement peut être automatiquement extraite et analysée à partir d'une image, d'un ensemble d'images ou d'une séquence d'images [20].

En robotique la vision artificielle par ordinateur consiste à créer un "œil" pour le robot afin qu'il puisse :

- se localiser dans son environnement.
- localiser un objet dans une scène par rapport à un référentiel donné (reconnaître la position et l'orientation d'un objet observé).
- identification d'objets (reconnaissance d'un objet, ou la description tridimensionnelle d'un objet inconnu).
- vérifier les spécifications d'un objet (mesure de toute propriété spatiale d'un objet telle que la distance entre deux de ses points distincts).

Dans notre projet, nous utilisons un système imageur pour la perception de l'environnement et l'extraction des informations utiles pour la commande du robot mobile manipulateur. Ce système comporte une caméra et un système d'acquisition et de transmission, il s'occupe des trois tâches suivantes : l'acquisition d'image et la perception de l'environnement en permanence, la transmission de l'image vidéo acquise vers le PC distant et enfin le traitement informatique de l'image et l'extraction en temps réel des informations utiles.

2.3.5.1 Acquisition de l'image vidéo

L'acquisition d'image s'effectue au niveau du robot mobile manipulateur, et elle est assurée par une caméra de type CCD (*Charge Coupled Device* en anglais ou DTC pour *Dispositif à Transfert de Charge* en français) installée sur la pince du bras du robot, l'image vidéo acquise est monochrome (en niveau de gris). Ce type de caméra présente les avantages résultant de leur structure intégrée : miniaturisation, robustesse, fiabilité, tension d'alimentation faible, puissance consommée réduite.

La caméra CCD que nous avons utilisée lors la réalisation de ce travail est de marque COSMICAR/PENTAX (figure 2.5), elle a les propriétés suivantes: l'image fournit est monochrome, elle est de taille 320x240 pixels, et chaque pixel a une taille de 6x6 µm. Elle a comme dimensions 44(L) x 29(H) x 57,5(P) mm et 110g de poids.

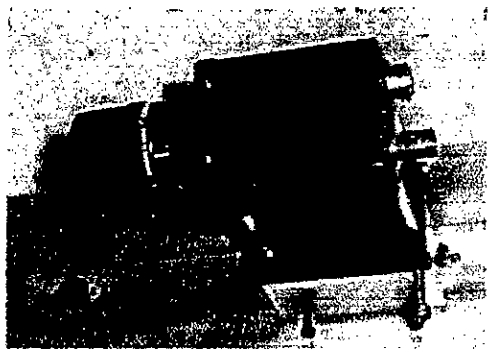


Figure 2.5 : Photo de la caméra CCD utilisée

2.3.5.2 Transmission de l'image vidéo

Vu que la puissance du PC embarqué ne permet pas d'effectuer le traitement nécessaire de l'image acquise à son niveau à cause des capacités moyennes de ce dernier et les performances que demande le traitement d'image, nous avons opté pour une solution permettant de transmettre l'image directement via un système de transmission HF vers le PC distant, pour y subir le traitement nécessaire.

Le système de transmission HF est composé de deux modules: module émetteur et autre récepteur, il utilise la voie hertzienne comme canal de transmission et les ondes électromagnétiques comme support d'information.

Module émetteur

Module récepteur

Figure 2.6 : Photos du système de transmission HF.

Ce dispositif est de marque Falcon PLUS™, sa portée de transmission peut aller jusqu'à 1 mille c'est-à-dire 1600 mètres (diamètre de la zone couverte), ce qui est largement suffisant pour notre application.

Le module émetteur reçoit l'image vidéo de la caméra sous format analogique, et envoie cette dernière vers le récepteur. Le module récepteur récupère l'image vidéo envoyée par l'émetteur et l'introduit ensuite dans une carte d'acquisition vidéo appelée Pinnacle 500-PCI, qui sert à numériser l'image pour que cette dernière soit manipulable sur PC, à ce point, l'image est prête à être traitée et interprétée par l'opérateur.

2.3.5.3 Traitement de l'image vidéo

L'image fournie par la caméra vidéo n'est pas directement exploitable après numérisation, et l'interprétation de cette image ne pourra commencer qu'après avoir transformé et préparé cette dernière afin qu'elle devienne facilement utilisable, c'est-à-dire pouvoir aisément isoler les objets composant la scène du reste (fond ou bruit), donc elle doit subir plusieurs opérations pour être exploitable. Ce traitement sera décrit dans le chapitre suivant.

2.3.6 Robot mobile manipulateur téléopéré

Le robot mobile manipulateur Robuter-ULM est le module d'exécution, il est capable de:

- se déplacer sur un plan horizontal selon les commandes reçues.
- manipuler des objets.

Dans la suite nous donnons une description détaillée de l'architecture matérielle et logicielle du robot téléopéré.

2.4 Structure de Robuter_ULM

Le robot Robuter-ULM est constitué comme le montre la photo ci-après (figure.2.7), d'un robot mobile à quatre roues, sur lequel est installé un bras manipulateur ultra léger à six degrés de liberté, le tout est piloté par un PC embarqué et des cartes à microcontrôleurs. Les détails concernant les dimensions et les dessins techniques sont données en annexe A.

L'architecture et la structure réelle du robot sont présentées par cette photo:

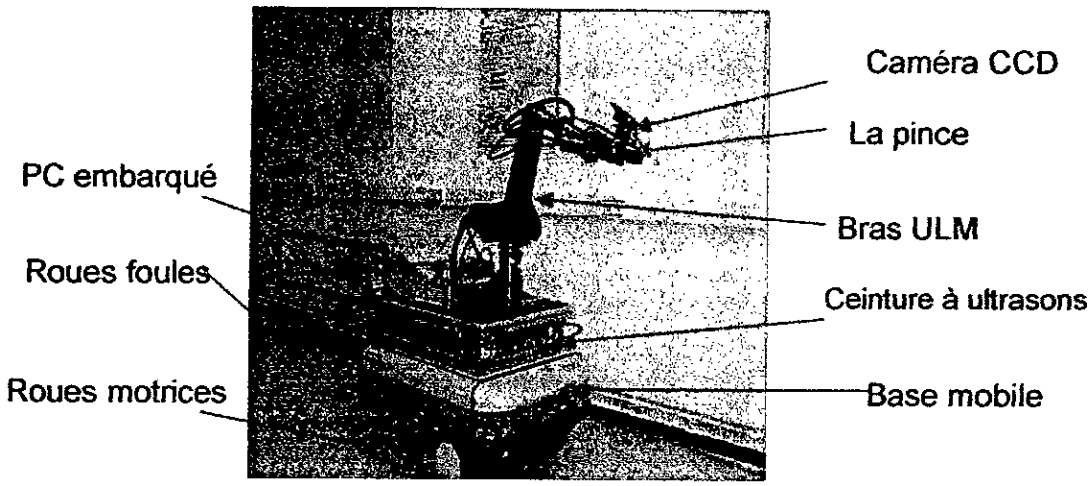


Figure 2.7 : Photo du robot Robuter-ULM.

L'ensemble de la plate forme mobile et du bras manipulateur est contrôlé par un PC embarqué (Pentium MMX 233Mhz sous Linux) et trois cartes à microcontrôleurs MPC555.

Les cartes contrôlent l'ensemble des actionneurs et des capteurs, dont une est attachée à la plate forme mobile et deux autres sont désignées pour le contrôle du bras. Un bus CAN permet la communication entre le PC embarqué et ces cartes. Pour plus de détails concernant l'architecture interne du système de commande, il est recommandé de consulter la référence [21].

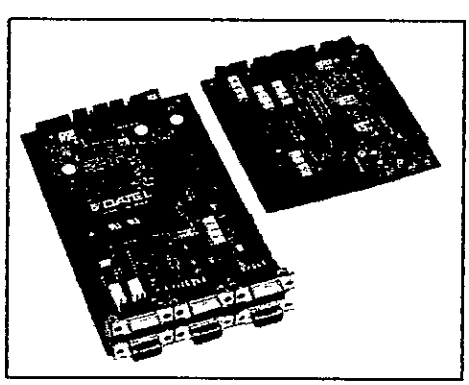


Figure.2.8 : Les cartes MPC 555 [21].

2.5 Description du robot mobile Robuter

Le robot mobile Robuter est une plate forme mobile qui dispose de quatre roues, deux roues motrices et deux roues foules assurant la stabilité et l'équilibre de l'ensemble. Les roues motrices sont actionnées séparément par des moteurs

électriques à courant continu. Ces roues sont de types différentielles, c'est-à-dire qu'ils assurent les déplacements de la plate forme par la différence de vitesse entre eux selon les principes suivants:

- Si les deux vitesses (la vitesse de la roue gauche et la vitesse de la roue droite) sont égales en valeur algébrique et elles ont le même signe, alors le robot se déplace en ligne droite soit vers l'avant ou bien vers l'arrière selon le signe de vitesses.
- Si les deux vitesses sont égales en valeur algébrique et elles ont des signes différents, alors le robot tourne sur lui même vers la droite ou vers la gauche.
- Si les deux roues ont des vitesses différentes en valeur algébrique, le robot effectue un mouvement selon une courbe de rayon à définir.

La plate forme est équipée d'un système sensoriel composé d'un ensemble de capteurs proprioceptifs et extéroceptifs pour mieux gérer et exploiter le robot mobile.

2.6 Le système sensoriel de Robuter [21]

La plate forme mobile est munie d'un système sensoriel riche qui contient plusieurs types de capteurs récoltant des informations variées sur son état interne, et externe par rapport à l'environnement où il se trouve. On peut classer ces capteurs en deux catégories selon qu'ils mesurent l'état interne du robot lui-même, on parle donc de capteurs proprioceptifs, ou l'état de son environnement, il s'agit des capteurs extéroceptifs.

2.6.1 Les Capteurs proprioceptifs du robot mobile

La plate forme mobile contient pour cette catégorie les capteurs suivants:

2.6.1.1 Le capteur de charge de batteries

Il s'agit d'un indicateur du niveau de batterie, il permet de nous renseigner par une barre lumineuse composée de LED sur l'état de charge de la batterie donc le niveau d'énergie restant. Au dessous des LED, ce trouve un afficheur digital qui

nous indique le temps global de fonctionnement de la plate-forme depuis la première mise en route.

2.6.1.2 Les capteurs odométriques

Ces capteurs permettent d'estimer le déplacement du robot à partir de la mesure de rotation des roues. Chaque roue motrice du robot mobile est associée à un moteur à courant continu. Celui-ci est équipé d'un réducteur et d'un dispositif de mesure de rotation disposé sur l'axe du moteur lui-même.

Ce dispositif est appelé codeur incrémental, il nous donne la vitesse de rotation du moteur en nombre d'incréments par unité de temps. Connaissant le temps et le nombre d'incréments on peut facilement calculer la distance parcourue par la roue.

Pour calculer cette distance en utilisant l'information issue du capteur odométrique, on procède comme suit :

- R : Représente le rayon de la roue.
- NP : La résolution du codeur est donnée par un nombre d'impulsions par tour.

Le périmètre de roue représente la distance parcourue par cette dernière si elle fait un tour sur elle même. Un tour fait 2.Π.R de distance, ainsi pour chaque impulsion la roue parcourt 2. Π.R/NP.

Si après un déplacement on obtient un nombre d'incréments égal à N, on peut facilement calculer la distance D parcourue par la roue.

$$D = \left(\frac{2.\pi.R}{NP}\right).N \qquad (2.1).$$

2.6.2 Les Capteurs extéroceptifs du robot mobile

Pour cette catégorie, et pour les besoins de la navigation et l'évitement d'obstacle le robot dispose de deux types de capteurs de distance et de proximité.

L'appareil permettant de mesurer les distances est appelé télémètre. On appelle télémétrie toute technique de mesure de distance par des procédés acoustiques, optiques ou radioélectriques. De même qu'il existe différentes techniques de mesure de distance (mesure du temps de vol d'une onde,

triangulation), il existe différentes technologies pour réaliser des télémètres. Tous les capteurs télémétriques, basés sur des mesures de l'environnement, sont bien évidemment actifs et extéroceptifs

2.6.2.1 La ceinture à ultrasons

La plate forme mobile est équipée également d'une ceinture d'ultrasons, qui est composée de 24 cellules ultrasoniques élémentaires (24 cellules émettrices réceptrices). Pour la disposition des capteurs sur la plate forme le lecteur peut se référer à l'annexe A.

Les télémètres à ultrason sont historiquement les premiers systèmes de mesures à avoir été utilisés. Ils utilisent des ondes sonores dont les fréquences ne sont pas perceptibles par l'oreille humaine. Les fréquences couramment utilisées dans ce type de technologie vont de 20 kHz à 200 kHz. Les ondes d'ultrasons émises se propagent dans l'air et sont réfléchis lorsqu'ils heurtent un corps solide devant elles. La distance entre la source et la cible peut être déterminée en mesurant le temps de vol séparant l'émission des ultrasons du retour de l'écho. Une telle cellule est représentée par la figure 2.9.

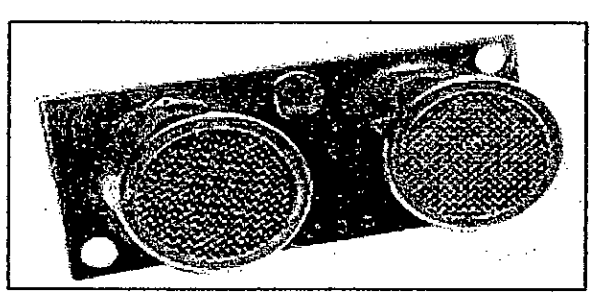


Figure 2.9 : Capteur ultrasonore (émetteur récepteur)

Le principe d'un télémètre à ultrason est très simple, il exploite le phénomène naturel de propagation et de réflexion de l'onde sonore (voir le schéma ci-après):

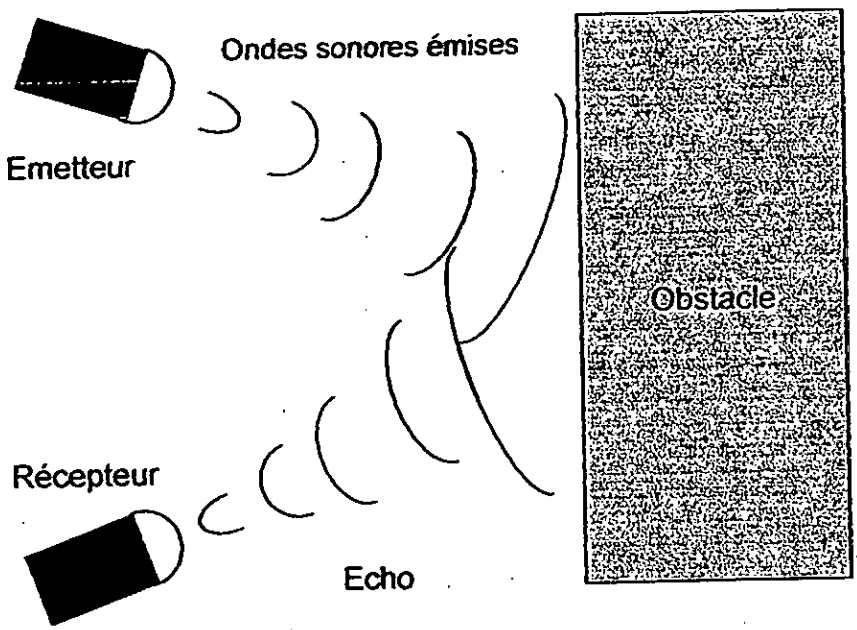


Figure 2.10 : Principe d'un télémètre à ultrasons.

La détermination de la distance repose sur la mesure du temps de vol entre l'émission de l'onde et la réception de son écho [22] [23]. La relation qui donne la distance en fonction du temps est la suivante:

$$D = \frac{V t}{2} \qquad (2.2).$$

Où:

D : est la distance en mètre.

t : le temps mesuré d'aller retour en second.

V : la vitesse du son en mètre / second. V ≈ 343m/s dans l'air à la température 20°C.

Ces télémètres sont très simples, peu cher et sont très répondus mais possèdent de nombreux inconvénients :

- l'onde émise est très sensible aux conditions environnementales. Un objet possédant des caractéristiques de texture non réfléchissantes ne sera pas détecté.

- deux télémètres voisins ne peuvent pas être utilisés simultanément, car il est impossible de savoir la source de l'onde réfléchie. Dans le cas d'une ceinture d'ultrasons, il faut activer les télémètres l'un après l'autre, ce qui implique un taux de rafraîchissement global de mesures relativement faible.

2.6.2.2 Le télémètre Laser

Ce capteur est placé sur la face avant du Robot mobile ce qui permet de scanner tout ce qui est devant le robot.

Ce type de télémètre utilise la lumière Laser, et donc l'onde optique comme moyen de mesure. Son principe est simple, un faisceau laser mis en rotation afin de balayer le plan (généralement horizontal), et qui permet de mesurer la distance des objets qui coupent ce plan [24]. Cette mesure peut être effectuée selon deux techniques, la première et basée toujours sur la mesure du temps de vol entre l'émission du signal et la réception de son écho, et la deuxième mesure le déphasage entre le signal émis et celui reçu.

Le télémètre Laser installé sur le robot est le SICK LMS200 (Figure.2.11), c'est un scanner de mesure à balayage deux dimensions (dans le plan horizontale avec un angle de 0° à 180°), il scrute son environnement et récupère les distances des objets se trouvant dans le plan. C'est une combinaison d'un télémètre à temps de vol avec un système de rotation du faisceau de mesure (Figure.2.12), grâce à cette technique on obtient une vision sous forme de radar. (Voir l'annexe B pour plus d'informations).

Figure 2.11: Le SICK LMS200 [24].

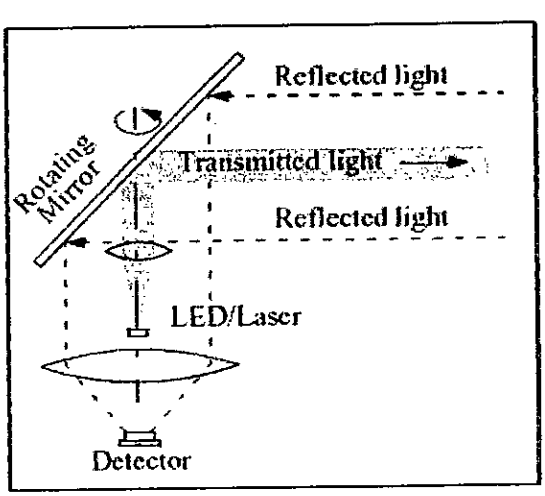


Figure 2.12 : Structure interne du LMS200 [22].

L'angle de balayage de ce système est représenté comme suit :

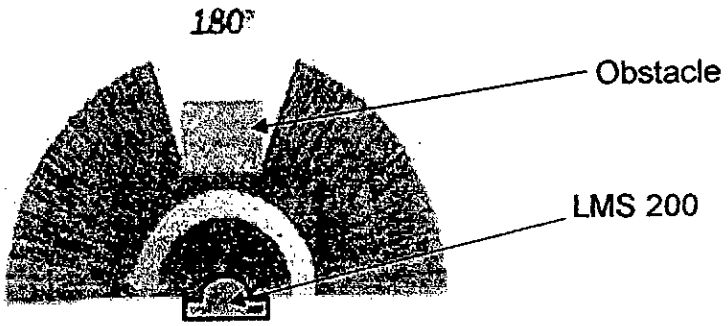


Figure 2.13 : Angle de balayage du LMS200. [24].

Généralement dans ce genre de système, deux approches sont utilisées pour la détermination des distances parcourues par le signal Laser :

- La première approche utilise le principe de mesure de temps de vol, une impulsion est émise par une diode laser et simultanément, une horloge est démarrée en parallèle. Cette impulsion lumineuse sera renvoyée par le premier obstacle rencontré sur son chemin. L'impulsion lumineuse, renvoyée par un obstacle, arrive sur un récepteur qui déclenche l'arrêt de l'horloge. A partir de ces informations, nous déduisons la distance séparant le télémètre et l'obstacle. Et connaissant la position angulaire du laser, nous arrivons à déterminer la position de l'objet dans le plan [25].

On a : le temps d'aller retour est exprimé en N impulsions, chaque impulsion étant cadencée par une fréquence d'horloge du système.

Donc : le temps = N.t.

Sachant que la vitesse du signal est celle de la lumière, et en appliquant la formule (2.2) on peut facilement obtenir la distance mesurée.

Le LMS 200 utilise cette technique, mais pour enrichir les informations des lecteurs on donne une idée sur le principe de la seconde technique.

- La seconde approche consiste à utiliser le principe de la mesure de déphasage entre le signal Laser émis modulé en fréquence et celui reçu. Les deux figures suivantes illustrent bien le phénomène [22].

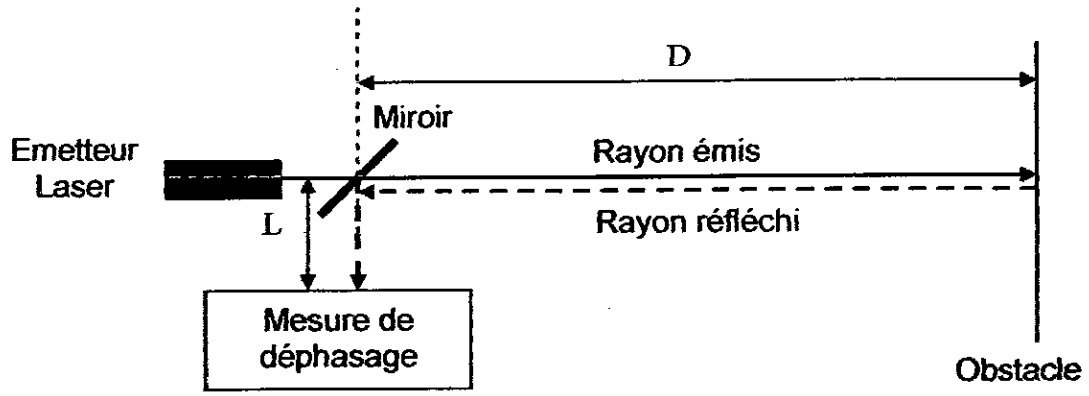


Figure 2.14 : Principe d'un télémètre Laser à mesure de déphasage [22].

Les deux signaux (émis et reçu) ont l'allure suivante :

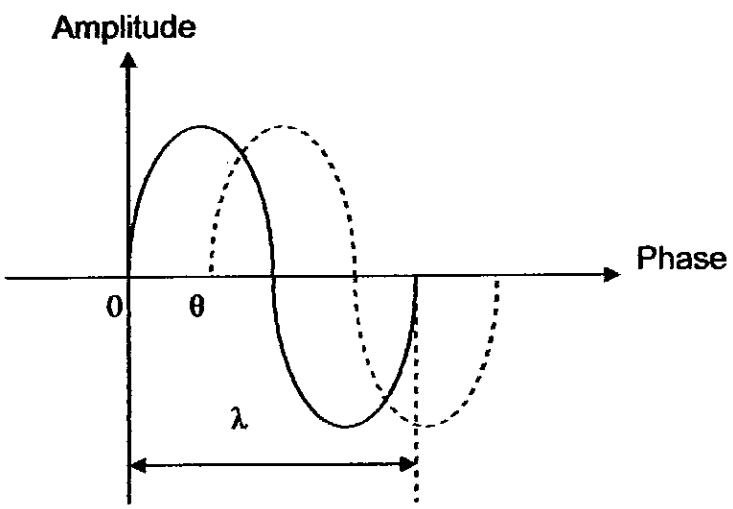


Figure 2.15: Mesure de phase entre le signal émis et le signal reçu [22].

La formule qui donne la distance mesurée en fonction de paramètres entrant en jeu est donnée par [22]:

Distance totale parcourue $= L + 2.D = L + \frac{\theta}{2.\pi}.\lambda$ (2.3)

Distance $= D = \frac{\lambda}{4.\pi}.\theta$

Avec : θ: le déphasage mesuré.

λ: la longueur d'onde.

On a $c = f.\lambda$ ou c la vitesse de la lumière et *f* la fréquence du signal.

Pour chaque angle on reçoit une distance proportionnelle, donc on peut calculer les coordonnées de l'objet trouvé par les deux formules 2.2 et 2.3 :

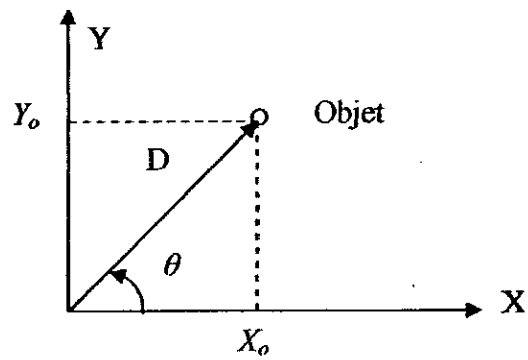


Figure 2.16 : Calcul de coordonnés d'un point connaissant la distance

$$X_o = D.Cos(\theta) \quad (2.4).$$

$$Y_o = D.Sin(\theta) \quad (2.5).$$

2.7 Description du bras manipulateur ultra léger ULM

ULM est un bras manipulateur ultra léger à six degrés de liberté, pesant 20kg, il se trouve porté et fixé sur la plate forme mobile. Pour plus de détails concernant le bras, le lecteur peut se référer à l'annexe A.

2.8 Système sensoriel du bras manipulateur

Le bras ultra léger dispose d'autres capteurs.

2.8.1 Capteurs de position des axes

Tous les codeurs incrémentaux installés sur le robot manipulateur sont des codeurs relatifs. Ils délivrent un nombre d'impulsion proportionnel à la rotation de l'axe effectué.

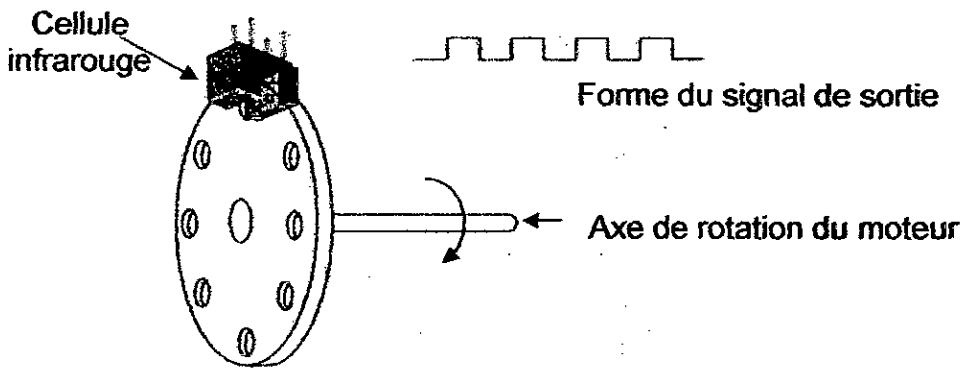
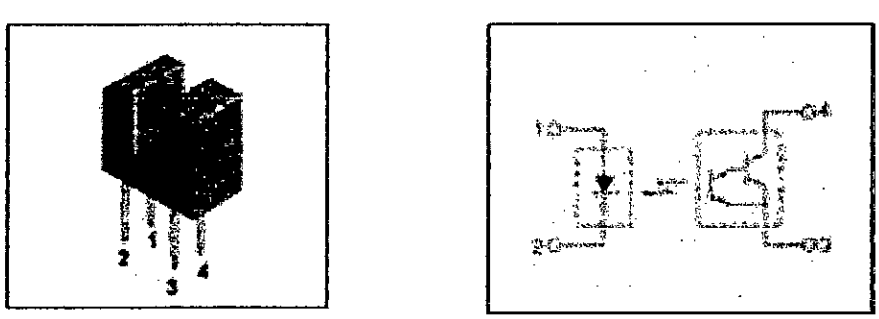


Figure 2.17 : Codeur incrémental

2.8.2 Capteurs de fin de cours

Il s'agit d'une cellule optique composée d'une LED émettrice infrarouge et d'un phototransistor récepteur d'infrarouge, comme le montre la figure ci-après. Ces cellules sont montées sur les butées des axes. Sur les butées des axes sont installés des indicateurs, si l'indicateur passe par la cellule, cette dernière nous informe de son passage.

Le récepteur est monté en face de l'émetteur à une distance de quelques millimètres, une barrière infrarouge sera donc établie par la cellule dans l'air qui sépare le récepteur de l'émetteur. La sortie du récepteur est traduite par un état logique, si un objet coupe la barrière optique un changement d'état logique sera signalé sur cette sortie.



a. Vue externe b. Structure interne

Figure 2.18 : Capteur de fin de cours.

2.8.3 La caméra

La caméra est installée sur la buté du bras, elle est de type CCD, elle délivre en permanence une image vidéo monochrome de l'environnement du robot.

2.9 Architecture logicielle du Robuter

Comme décrit précédemment, le robot mobile manipulateur est contrôlé par un PC embarqué sous Linux et par un système électronique de commande constitué de trois cartes à microcontrôleur MPC555. Pour comprendre l'organisation de cette architecture voici ce schéma [21] :

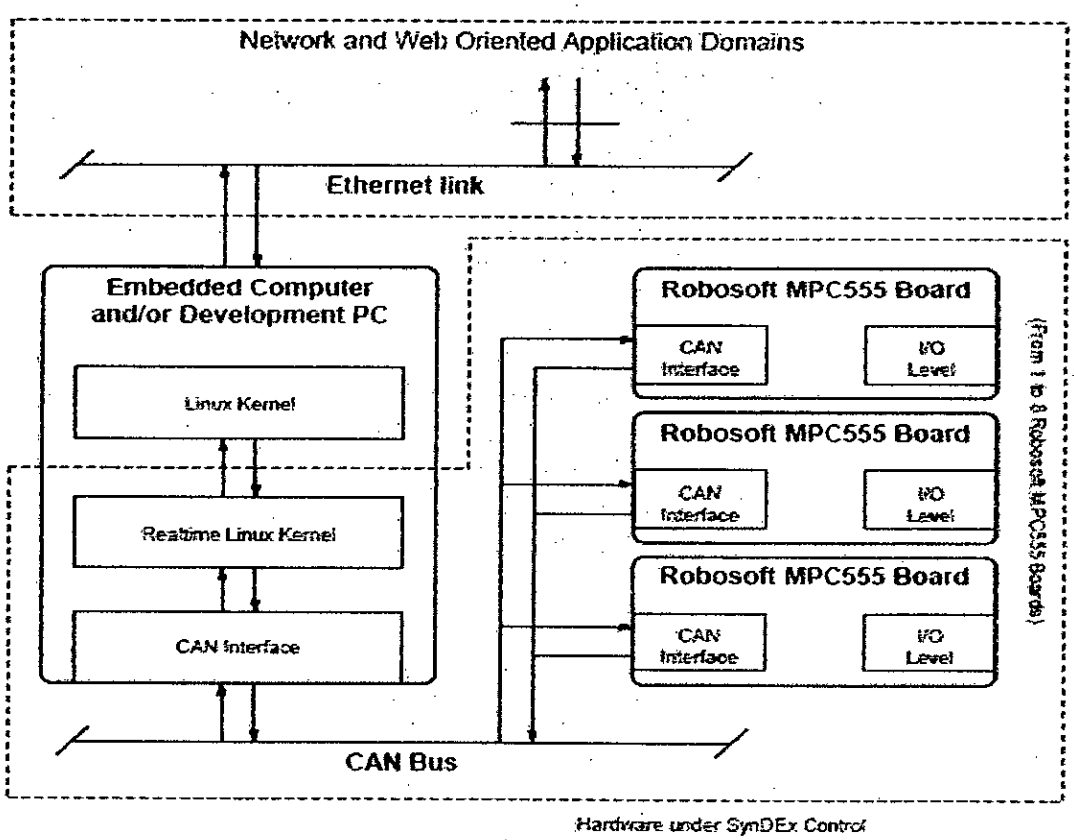


Figure 2.19 : Architecture Logiciel du Robuter_ULM.

De cette architecture on constate que pour faire actionner le robot et exploiter les capteurs, il faut développer trois applications, une application haut niveau détermine la tâche à faire et les lois de commande nécessaires, une

application bas niveau chargé d'exécuter ces commandes et une autre application assure la communication entre les deux applications et l'exécution des tâches en temps réel.

- L'application utilisateur se déroule sous Linux, généralement elle est écrite en C ou C++. Elle exprime ce que l'opérateur veut du robot de le faire. En effet, il s'agit d'un échange de données entre le programme de commande et le programme d'exécution bas niveau.

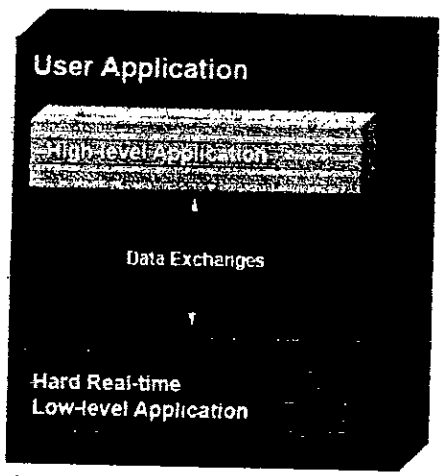


Figure 2.20 : Principe général d'une application utilisateur.

- L'application de bas niveau est chargée de dialoguer avec les cartes pour commander les moteurs et activer les capteurs (sur les cartes se trouvent les programmes élémentaires de bases comme la commande en PWM des moteurs). Le programme de bas niveau doit aussi pouvoir communiquer avec un programme haut niveau.

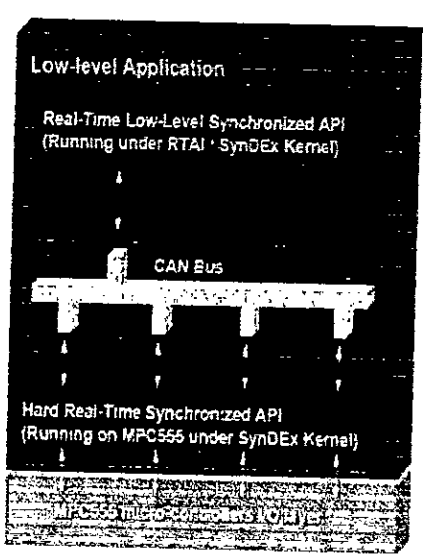


Figure 2.21 : Structure générale d'une application bas niveau.

- La dernière application assure la communication entre les différents modules. Il utilise pour cela le concept de la mémoire partagée, qui est une allocation et réservation des segments mémoire pour l'échange de données entre le haut niveau et le bas niveau. Cette tâche est assurée par SynDEx et le RTAI.

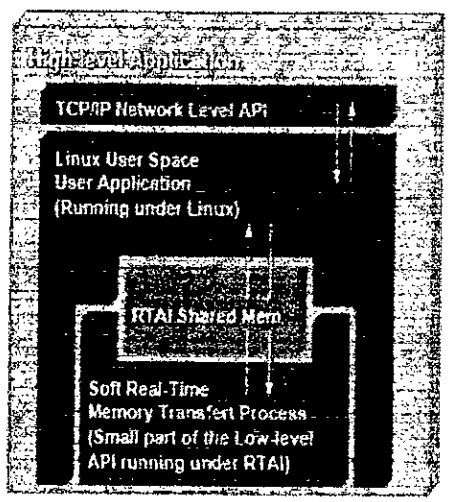


Figure 2.22 : Principe général d'une application haute niveau.

Après avoir expliqué la structure générale d'un programme sous PC embarqué, il est très nécessaire de donner quelques notions concernant cette architecture.

2.9.1 L'environnement Linux

LINUX est une version libre du système d'exploitation UNIX, écrit initialement par Linus Torvalds en 1991 [26]. Les origines de tous les systèmes UNIX remontent à la première version d'un système d'exploitation expérimental développé par Dennis Ritchie et Ken Thompson dans les laboratoires AT&T's Bell Laboratories en 1969 [27]. Ce système, avant tout, a été développé par des programmeurs pour des programmeurs.

Linux est un système fiable, fonctionnel et performant, il est multitâche, multiutilisateur et orienté réseau, il nous permet de réaliser les opérations les plus classiques, comme effectuer un travail bureautique, naviguer sur Internet, réaliser l'acquisition, la capture et le traitement d'images, réaliser des animations 3D ou encore programmer.

La version de Linux installée sur le PC embarquée est celle de RedHat.v.6, cette version assure une interface graphique (xwindows) de faible performance, qui peut gérer des graphismes minimales et simples, pratiquement il est utilisé en mode texte (en ligne de commande).

2.9.2. Linux Real Time Application Interface (RTAI)

Linux est marqué comme étant un système d'exploitation multitâches et multi utilisateurs qui supporte la philosophie temps réel.

On dit qu'un système fonctionne en "Temps Réel", s'il est capable d'absorber toutes les informations d'entrée sans qu'elles soient trop vieilles pour l'intérêt qu'elles représentent, et par ailleurs de réagir à temps pour que cette réaction ait un sens [27].

Le RTAI est vu comme une application qui se déroule en même temps avec les applications de l'utilisateur pour permettre la gestion et le partage de temps d'exécution entre eux.

2.9.3. SynDEx

Le robot est équipé par un logiciel de gestion de ressource, appelé SynDEx, ce dernier a été développé par les chercheurs du laboratoire INRIA.

SynDEx (**Syn**chronous **D**istributed **Ex**ecutifs) est un environnement logiciel graphique de développement niveau système pour applications temps réel de commande supportant la méthode Adéquation Algorithme Architecture (AAA), il fournit une aide à l'implantation temps réel multi processeur de ces algorithmes en déchargeant au maximum l'utilisateur des taches lourdes de programmation bas niveau système [28]. Plus d'information sont à la portée des lecteurs en Annexe B.

Le robot est livré avec la version 5 de SynDEx, dont la fenêtre de développement est illustrée ci-après :

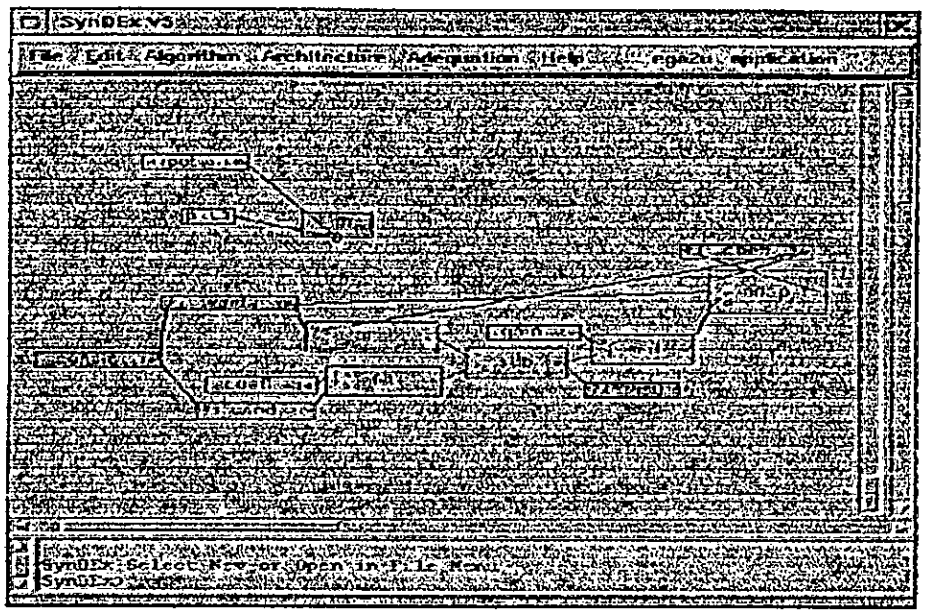


Figure 2.23 : Une fenêtre de développement de SynDEx V.5

2.9.4. Le GNUPro Toolchain

Les cartes de commande sont conçues à base des microcontrôleurs MPC555 de Motorola, ces cartes sont programmées et configurées en assembleur MPC555. Il est possible de programmer ces cartes avec leur langage Assembleur de bas niveau en utilisant l'utilitaire de développement GNUPro Toolchain qui fonctionne sous Linux. On peut aussi utiliser SynDEx pour programmer et charger le programme voulu sur les cartes, et bien sûr le listing doit être écrit en C.

2.10. Conclusion

Dans ce chapitre nous avons représenté le système de téléopération et le système robotique, objet de notre étude, qui se trouve au CDTA, pour plus de spécifications techniques concernant le robot, des détails sont à la portée du lecteur en annexe A. Le chapitre suivant est réservé à la modélisation de notre système téléopération.

CHAPITRE 3
MODELISATION GEOMETRIQUE DU SYSTEME ROBOTIQUE

3.1 Introduction

Notre travail consiste à manipuler par le robot manipulateur mobile à distance des objets situés sur un panneau vertical se trouvant devant le robot. Cela implique qu'il faut positionner le robot mobile devant le panneau et le bras manipulateur sur les objets. Le robot se trouve dans une position bien définie dans le repère de référence fixe et il va manipuler des objets qui doivent être bien définis dans le repère du bras manipulateur. Les différents repères mis en jeu sont représentés ci-dessous:

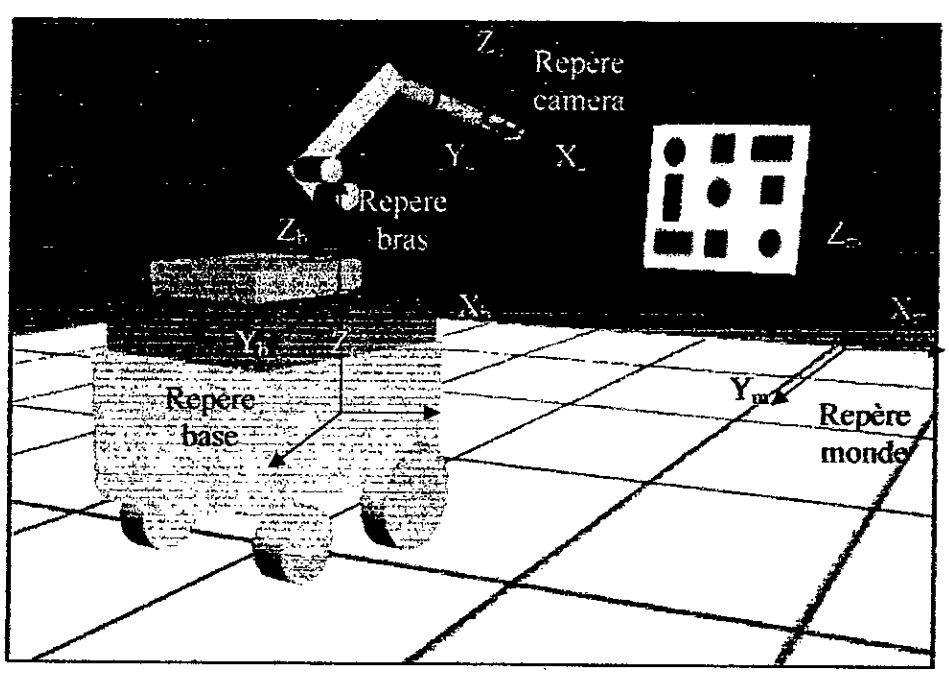


Figure 3.1: Les différents repères mis en jeu

Pour pouvoir manipuler un objet, il faut d'abord le localiser dans le repère 3D, c'est-à-dire connaître ses coordonnés cartésiennes dans un repère de référence fixe (repère lié à la scène), et connaître ses coordonnés dans le repère lié au robot. Cela ne peut se faire que si on connaît avec précision les différentes

transformations des repères mises en jeu. En effet, ces transformations élémentaires sont au nombre de quatre:

- une transformation du repère objet au repère caméra : pour exprimer les coordonnées d'un objet dans le repère caméra, cela se fait par une modélisation géométrique de la caméra.
- une transformation du repère caméra au repère pince.
- une transformation du repère pince au repère base par une modélisation géométrique du bras manipulateur.
- une transformation du repère base du bras au repère robot mobile.

Les différents repères et transformations mis en jeu sont illustrés dans la figure suivante:

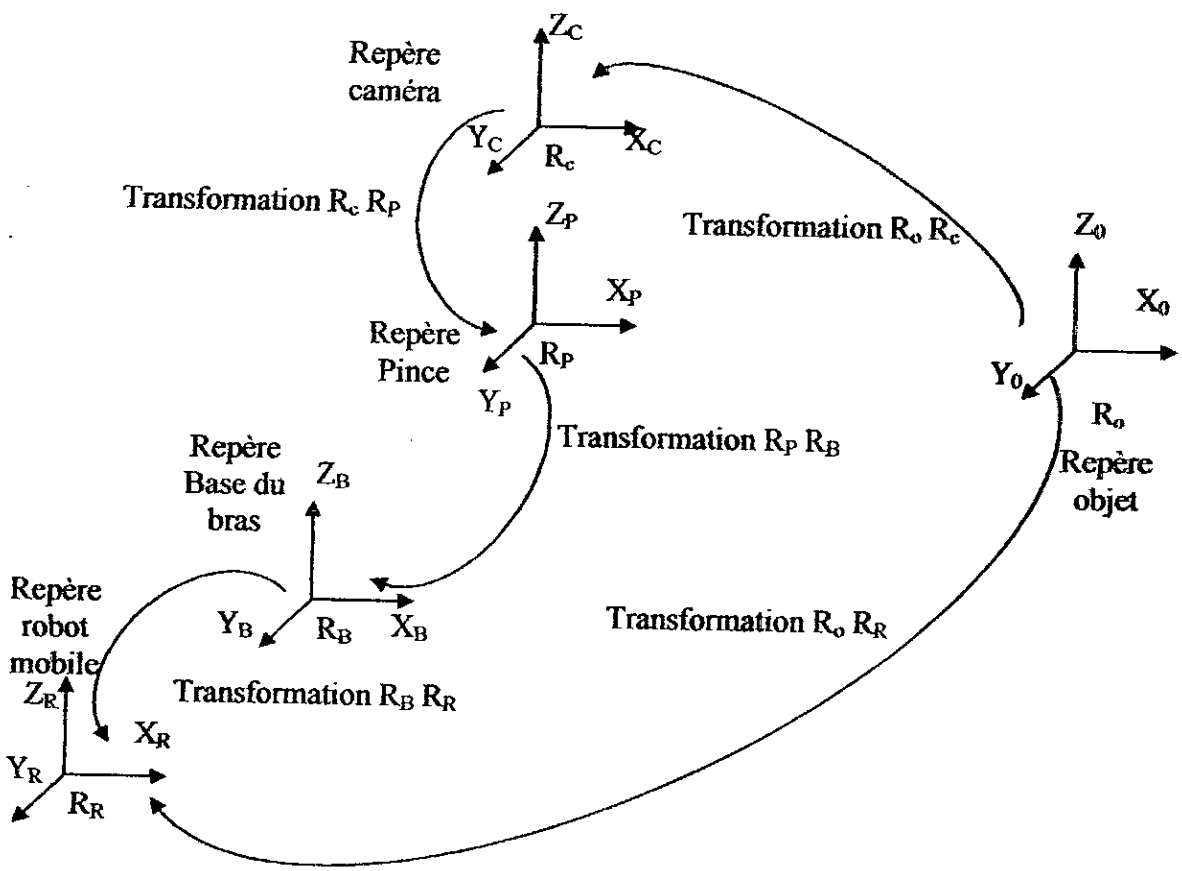


Figure 3.2 : Les transformations de repères nécessaires

Une transformation homogène entre deux repères, repère fixe et repère relatif s'exprime en général en un déplacement et une rotation. Si on adopte la notation matricielle cette transformation est composée d'une matrice de rotation notée R formée par les rotations autour des axes, et d'une matrice de translation notée T entre les deux repères, elle peut s'écrire sous la forme suivant :

$$M = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} [R] & [T] \\ 0 & 1 \end{bmatrix} \quad (3.1)$$

3.2 Transformation repère caméra/repère pince

La transformation repère camera/ repère pince est obtenue directement en mesurant les déplacement suivant x, y et z entre les repères, tandis que tous les angle de rotations sont nuls. On a : (Les mesures sont en mm).

$$M_c^p = \begin{bmatrix} 1 & 0 & 0 & 35 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 165 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.2)$$

3.3 Transformation repère bras/repère robot mobile

La transformation repère objet/ repère robot est obtenu en multipliant les transformations élémentaires cités auparavant. On a : (Les mesures sont en mm).

$$M_b^r = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 350 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.3)$$

3.4 Modélisation du robot mobile (Robuter)

La modélisation de ce robot consiste à établir les modèles géométriques qui entrent en jeu dans la définition de sa localisation et de son positionnement.

3.4.1 Modélisation géométrique du robot mobile

La plate forme mobile peut être représentée dans le plan comme suit:

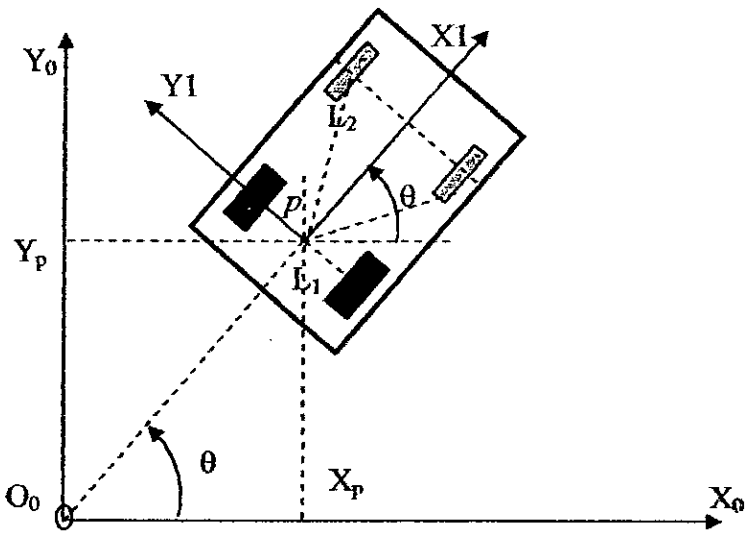


Figure 3.3 : Représentation dans le plan du robot mobile.

Pour spécifier la position du robot mobile, on a choisi un point *p* situé au centre de l'axe des roues motrice comme point de référence, ce point est muni d'un repère relatif R_1 (O_1, X_1, Y_1, Z_1) et d'un repère global fixe R_0(O_0, X_0, Y_0, Z_0) (figure 3.4). La position de *p* dans le repère R_0 est déterminée par leurs coordonnées cartésiennes X_p et Y_p, et θ l'angle de rotation entre le repère lié au chassi R_1 et le repère R_0 autour l'axe Z_0. La position de *p* peut être décrite par le vecteur :

$$P = \begin{pmatrix} x_p & y_p & \theta \end{pmatrix}^T \tag{3.4}$$

La matrice de rotation entre les deux repères (R_0 et R_1) est la suivante [30]:

$$R(\theta) = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.5}$$

3.4.2 Description des roues

On considère que les roues utilisées sur notre robot sont conventionnelles, parfaites et indéformables de plus on considère le contact des roues avec le sol ponctuel. Les roues conventionnelles sont divisées en trois classes [30]:

- Roues fixes notées (R_F).
- Roues orientables centrées notées (R_{OC}).

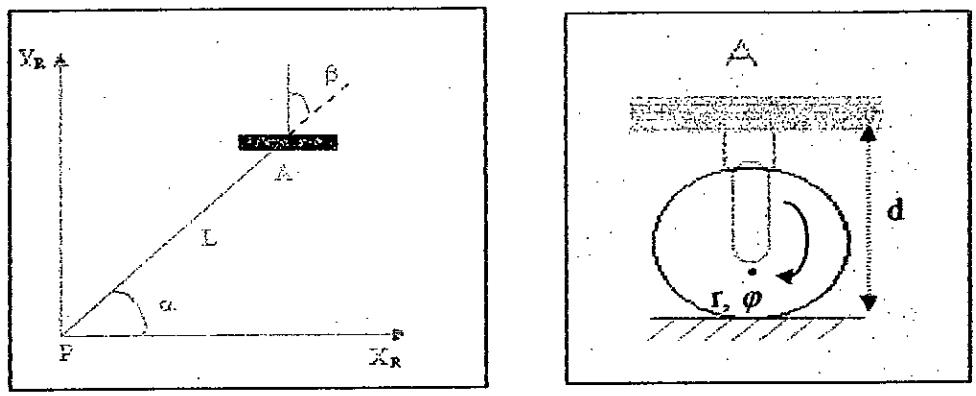


Figure 3.4: Roue conventionnelle orientable centrée.

- Roues orientables décentrées notées (R_{OD}).

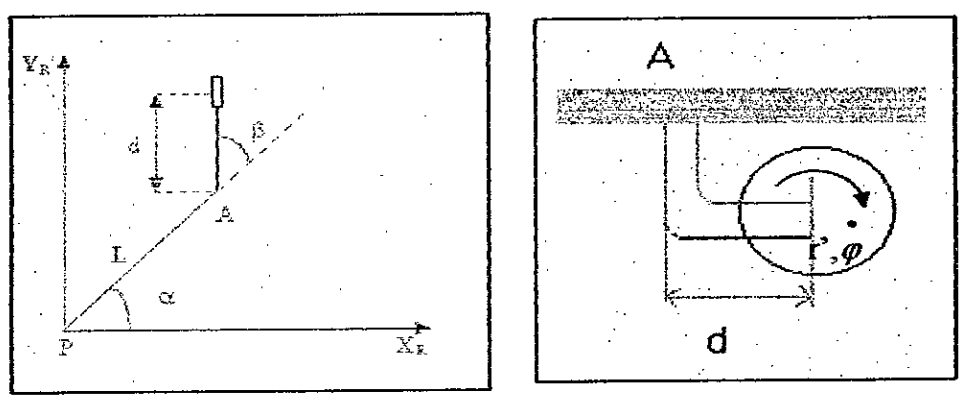


Figure 3.5: Roue conventionnelle orientable décentrée.

Elles sont généralement caractérisées par les paramètres suivants :

- Le rayon des roues centrées noté r, le rayon de la roue décentrée noté r'.
- Distance L (L= PA), entre le point P et le point de jonction A de la roue sur la plate forme.
- L'angle α formé par la droite (PA) et le vecteur X_1.
- L'angle β formé par la droite (PA) et l'axe passant par le centre de la roue. Cet angle est fixe pour les roues conventionnelles fixées et variables pour les roues conventionnelles orientables centrées ou décentrées.
- La distance d propre aux R_{OD}, représente la distance entre le point de jonction A et le centre de la roue.
- Les roues sont modélisées comme étant des disques idéaux, d'épaisseur zéro et de rayon *r*. De ce fait, le contact avec le sol est considéré ponctuel.

Chaque roue tourne dans l'espace avec une vitesse angulaire $\dot{\varphi}$ autour de son axe.

Les contraintes qui régissent le mouvement d'un roulement pur et sans glissement d'une roue sont [30], [31] :

$$\left[\sin(\alpha+\beta) \quad -\cos(\alpha+\beta) \quad -L.\cos\beta\right]R(\theta).\dot{P}-r.\dot{\vartheta}=0 \tag{3.6}$$

$$\left[\cos(\alpha+\beta) \quad \sin(\alpha+\beta) \quad d+L.\sin\beta\right]R(\theta).\dot{P}-d.\dot{\beta}=0 \tag{3.7}$$

Avec : $\dot{\beta}$ = 0 pour les roues fixes et d = 0 pour les roues centrées.

Si on applique ces formules dans le cas de notre robot, à savoir les deux roues arrière fixes et centrées et les deux roues folles avant qui sont aussi centrées, nous obtenons les résultats suivants:

Les roues du robot sont caractérisées par les angles suivants (*figure 3.3*) :

- Roue droite : $\alpha_d = -\pi/2$, $\beta_d = -\pi$.
- Roue gauche : $\alpha_g = \pi/2$, $\beta_g = 0$.
- Roue folle droite : $\alpha_{fd} = -\pi/4$, $\beta_{fd} = \beta_{fd}(t)$.
- Roue folle gauche : $\alpha_{fg} = \pi/4$, $\beta_{fg} = \beta_{fg}(t)$.

A partir des équations (3.3) et (3.4) et en tenant compte des angles α et β définis plus haut, on aura :

$$\begin{cases} \left[1 \quad 0 \quad L1\right]R(\theta).\dot{P}-r.\dot{\vartheta}_d=0 \\ \left[1 \quad 0 \quad -L1\right]R(\theta).\dot{P}-r.\dot{\vartheta}_g=0 \\ \left[\sin(\beta_{fd}(t)-\pi/4) \quad -\cos(\beta_{fd}(t)-\pi/4) \quad -L1.\cos\beta_{fd}(t)\right]R(\theta).\dot{P}-r.\dot{\vartheta}_{fd}=0 \\ \left[\sin(\beta_{fg}(t)+\pi/4) \quad -\cos(\beta_{fd}(t)-\pi/4) \quad -L1\cos\beta_{fg}(t)\right]R(\theta).\dot{P}-r.\dot{\vartheta}_{dg}=0 \end{cases} \tag{3.8}$$

Et aussi :

$$\begin{cases} \left[0 \quad -1 \quad 0\right]R(\theta).\dot{P}=0 \\ \left[0 \quad 1 \quad 0\right]R(\theta).\dot{P}=0 \\ \left[\cos(\beta_{fd}(t)-\pi/4) \quad \sin(\beta_{fd}(t)-\pi/4) \quad L1.\sin\beta_{fd}(t)\right]R(\theta).\dot{P}=0 \\ \left[\cos(\beta_{fg}(t)+\pi/4) \quad \sin(\beta_{fd}(t)-\pi/4) \quad L1\sin\beta_{fg}(t)\right]R(\theta).\dot{P}=0 \end{cases} \tag{3.9}$$

Ces deux systèmes d'équations peuvent être écrits sous la forme matricielle suivante :

$$\begin{bmatrix} -1 & 0 & -L_1 \\ 1 & 0 & -L_1 \\ \sin(\beta_{fd}(t)-\pi/4) & -\cos(\beta_{fd}(t)-\pi/4) & -L1.\cos(\beta_{fd}(t) \\ \sin(\beta_{fg}(t)+\pi/4) & -\cos(\beta_{fg}(t)+\pi/4) & -L1.\cos(\beta_{fg}(t) \end{bmatrix} R(\theta).\dot{P} - \begin{bmatrix} r & 0 & 0 & 0 \\ 0 & r & 0 & 0 \\ 0 & 0 & r & 0 \\ 0 & 0 & 0 & r \end{bmatrix} \begin{bmatrix} \dot{\vartheta}_d \\ \dot{\vartheta}_g \\ \dot{\vartheta}_{fd} \\ \dot{\vartheta}_{dg} \end{bmatrix} = 0 \tag{3.10}$$

$$\begin{bmatrix} 0 & -1 & -L_2 \\ 0 & 1 & -L_2 \\ \sin(\beta_{fd}(t)-\pi/4) & -\cos(\beta_{fd}(t)-\pi/4) & -L_2.\cos(\beta_{fd}(t) \\ \sin(\beta_{fg}(t)+\pi/4) & -\cos(\beta_{fg}(t)+\pi/4) & -L_2.\cos(\beta_{fg}(t) \end{bmatrix} R(\theta).\dot{P} - \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ \dot{\beta}_{fd}(t) \\ \dot{\beta}_{dg}(t) \end{bmatrix} = 0 \tag{3.11}$$

3.4.3 Modélisation cinématique du robot mobile

Le modèle cinématique donne la transformation qui permet de passer des vitesses angulaires des roues à la vitesse instantanée du robot.

Du système d'équation (3.7) si on considère que la 1ère et la 2ème ligne, on a :

$$\begin{bmatrix} 1 & 0 & L_1 \\ 1 & 0 & -L_1 \end{bmatrix} \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{X}_p \\ \dot{Y}_p \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} r & 0 \\ 0 & r \end{bmatrix} \begin{bmatrix} \dot{\vartheta}_d \\ \dot{\vartheta}_g \end{bmatrix} \tag{3.12}$$

Finalement on trouve :

$$V_d = \dot{X}_p.\cos\theta + \dot{Y}_p.\sin\theta + L_1.\dot{\theta} \tag{3.13}$$

$$V_g = \dot{X}_p.\cos\theta + \dot{Y}_p.\sin\theta - L_1.\dot{\theta} \tag{3.14}$$

Avec :

$V_d = r.\dot{\vartheta}_d$: représente la vitesse linéaire de la roue droite

$V_g = r.\dot{\vartheta}_g$: représente la vitesse linéaire de la roue gauche

Tandis que la deuxième ligne de l'équation (3.8) nous donne :

$$-\dot{X}_p.\sin\theta + \dot{Y}_p.\cos\theta = 0 \tag{3.15}$$

De l'équation (3.10), (3.11) et (3.12) on a :

$$\dot{X}_p = \frac{V_d + V_g}{2}.\cos\theta \tag{3.16}$$

$$\dot{Y}_p = \frac{V_d + V_g}{2}.\sin\theta \tag{3.17}$$

$$\dot{\theta} = \frac{V_d - V_g}{2.L_1} \tag{3.18}$$

La vitesse linéaire du point P (vitesse moyenne à ce point) est donnée par :

$$V_{moy} = \frac{V_d + V_g}{2} \tag{3.19}$$

Et la vitesse angulaire du robot est donnée par :

$$\dot{\theta} = \frac{V_d - V_g}{2.L_1} \tag{3.20}$$

Par intégration de l'équation (3.15) on a :

$$\theta = \int_{t0}^{t} \frac{1}{2.L_1}.(V_d - V_g)dt = \frac{1}{2.L_1}.(D_d - D_g) + \theta_0 \tag{3.21}$$

Où : D_d et D_g sont les distances parcourues par la roue droite et la roue gauche respectivement. θ_0 est la valeur initiale de l'angle.

A partir de ces dernières équations, on peut tirer les conclusions suivantes sur le modèle cinématique :

- Si $V_d = V_g$: le rayon de courbure ρ est égal à l'infini, la trajectoire du robot est une droite avec une vitesse $V = V_d = V_g$, vers l'avant ou vers l'arrière suivant le sens de rotation des roues.
- Si $V_d = -V_g$: le robot fait une rotation autour du point P avec une vitesse de rotation égale à V_d/L.
- Si $V_d > V_g$: le robot fait un braquage vers la gauche.
- Si $V_d < V_g$: le robot fait un braquage vers la droite.

- Si $\theta = \pi/2 + k\pi$: on aura $tg\theta = \infty$ c'est à dire $\dot{y}_p / \dot{x}_p = \infty$, ce qui veut dire que $\dot{x}_p = 0$, on n'a pas de mouvement suivant X mais suivant Y uniquement.

- Si $\theta = k\pi$: on aura $cotg\theta = \infty$ c'est à dire $\dot{x}_p / \dot{y}_p = \infty$, autrement dit $\dot{y}_p = 0$, on a donc un mouvement suivant X seulement.

3.5 Modélisation géométrique du bras ULM

Le but de cette modélisation est de pouvoir positionner et orienter la pince (l'organe terminale) du bras ULM en fonction des consignes de l'utilisateur. Rappelons que dans le cas du bras ULM qui est un assemblage de segments par des liaisons pivots, on caractérise la position relative entre deux segments successifs par un paramètre (l'angle de rotation) appelé *coordonnée (variable) généralisée.*

Un système mécanique articulé consiste en une chaîne cinématique de membres, ou corps rigides, reliés les uns aux autres en leurs extrémités par des articulations. En associant, conformément au formalisme des transformations homogènes, un référentiel de description à chacun des membres d'un système mécanique articulé, il est possible de décrire leurs postures relatives ou absolues dans l'espace 3D [31].

L'établissement du modèle géométrique consiste à établir, dans le cas général, les six paramètres indépendants qui suffisent pour déterminer le mouvement d'un corps rigide dans l'espace. Ces paramètres peuvent représenter le déplacement du centre d'un repère attaché au corps rigide (repère pièce) et les trois rotations de ce repère par rapport à un repère de référence. La coordonnée généralisée associée à un axe pour définir le mouvement permis par une liaison entre deux segments est un angle.

Les variables généralisés du bras ultra léger sont figurés comme suit :

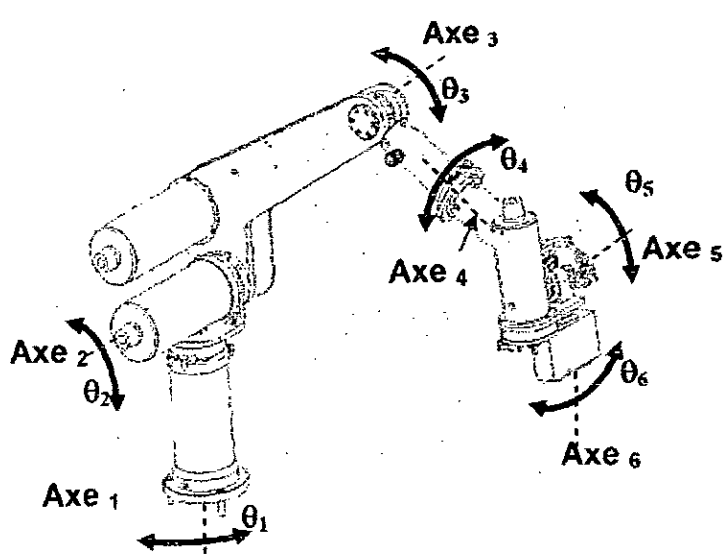


Figure 3.6 : Variables généralisées du bras ULM

3.5.1 Modélisation géométrique directe de ULM

La situation de l'élément final d'un mécanisme par rapport à un référentiel relié au bâti dépend des paramètres dimensionnels (géométriques) des segments successifs et des positions relatives entre ces segments, ce qui s'exprime par une relation utilisant les paramètres géométriques et les coordonnées généralisées. Cette relation est appelée *modèle géométrique direct.*

Le Modèle Géométrique Direct (MGD) de ULM exprime la *situation* (*position* et *orientation*) de l'élément final X (*coordonnées opérationnelles*) en relation avec les coordonnées généralisées q du mécanisme par rapport à son bâti.

$$X = f(q) \tag{3.22}$$

En effet, il existe différentes possibilités pour décrire la situation de l'organe terminal (la pince) par rapport au repère absolu du bâti. Concernant la position, les coordonnées cartésiennes, cylindriques et sphériques sont souvent utilisées. En ce qui concerne l'orientation, les possibilités sont plus nombreuses. Citons à titre d'exemple les angles de Bryant-Cardan, les angles d'Euler [32], les paramètres d'Euler [33], les cosinus directeurs [33], etc.

3.5.1.1 Mise en place de l'approche de Denavit-Hartenberg

La technique la plus répandue pour décrire la géométrie d'un bras manipulateur consiste à utiliser les *paramètres de Denavit-Hartenberg*, [32] [34][35].

Denavit et Hartenberg proposent en 1955 une méthode systématique d'attribution de référentiels aux membres d'une chaîne cinématique, permettant d'établir les transformations de passage entre articulations adjacentes. Cette méthode, communément appelée convention de Denavit et Hartenberg ou convention DH, concerne essentiellement les chaînes cinématiques ouvertes, dont chaque articulation possède un degré de liberté en rotation ou en translation [36] [37].

En effet, il y a deux formes de représentations des paramètres de Denavit Hartenberg, la première forme, c'est celle établie par Denavit Hartenberg, et la

seconde développée par Shetty et Ang en 1996, s'appelle représentation de Denavit Hartenberg modifiée [36]:

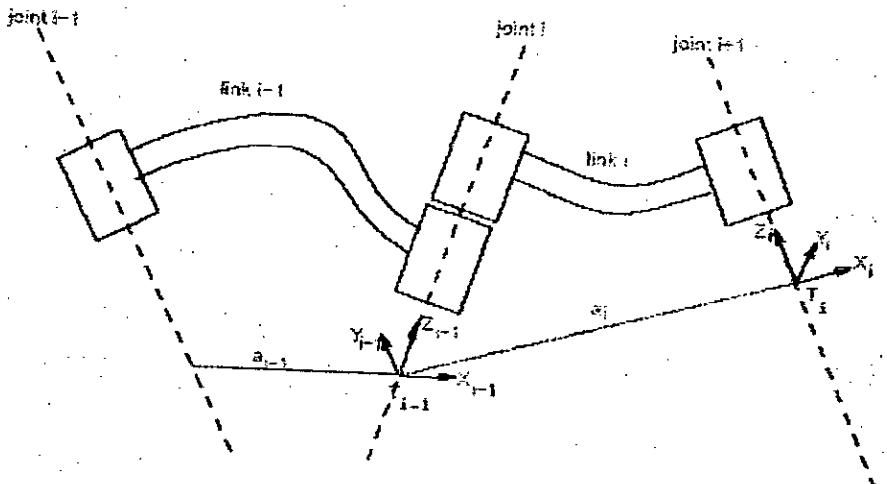


Figure 3.7: Paramètres de D&H d'un système articulé.

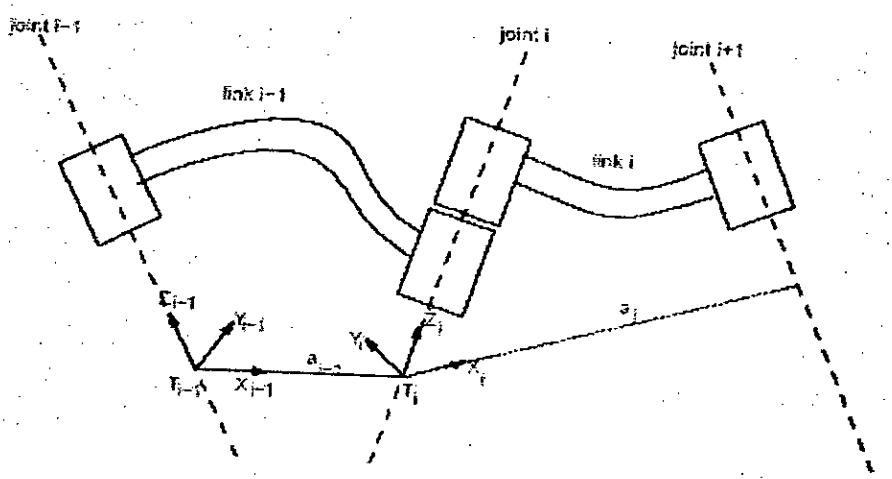


Figure 3.8: Paramètres modifiés de D&H d'un système articulé.

Selon la seconde approche, les paramètres de Denavit-Hartenberg des matrices de passages homogène sont :

1- O_i : le point d'intersection entre les segments (i) et (i+1).

2- α_{i-1} : l'angle de rotation autour de l'axe X_i entre les axes Z_i et Z_{i+1}.

3- a_{i-1} : la distance entre l'axe Z_i et l'axe Z_{i+1}, le long de X_i.

4- d_i : la distance entre l'axe X_{i-1} et l'axe X_i, le long de Z_i.

5- θ_i : l'angle de rotation entre les axes X_{i-1} et X_i autour de Z_i.

La matrice de passage homogène d'un repère à un autre des paramètres Denavit-Hartenberg modifié a la forme suivante [37]:

$$T_{i-1}^{i}=\begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 & a_{i-1} \\ \cos\alpha_i.\sin\theta_i & \cos\alpha_i.\cos\theta_i & -\sin\alpha_i & -d_i.\sin\alpha_i \\ \sin\alpha_i.\sin\theta_i & \sin\alpha_i.\cos\theta_i & \cos\alpha_i & d_i.\cos\alpha_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (3.23)$$

L'application de cette approche sur notre bras manipulateur, nous a permis de définir les repères suivants :

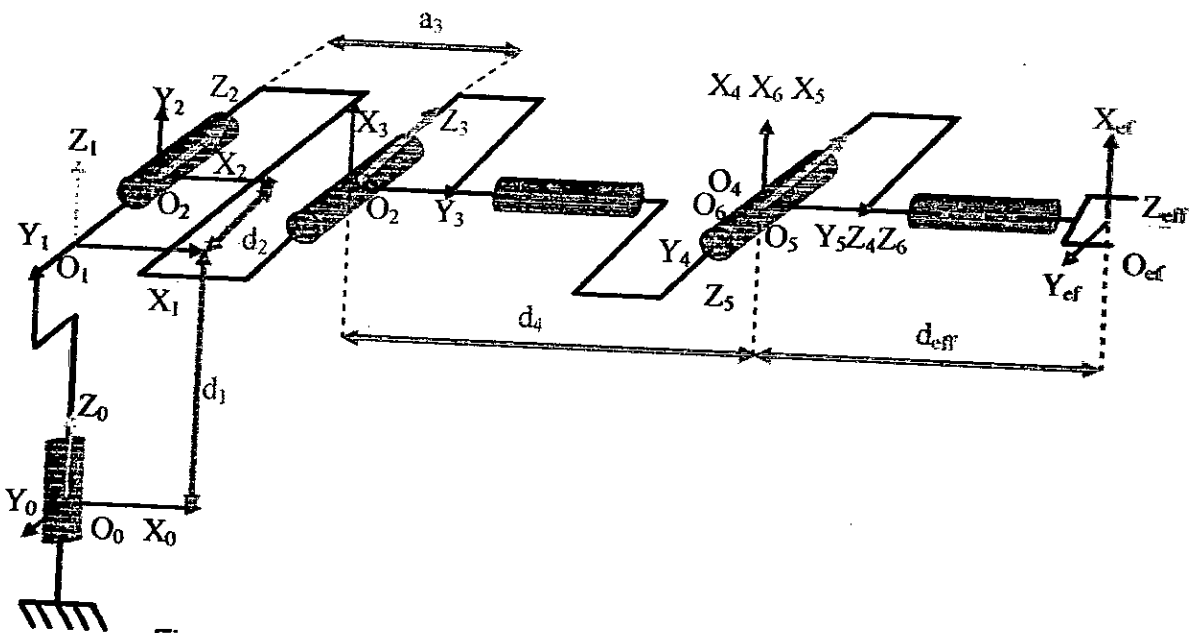


Figure 3.9 : Paramètres de Denavit Hartenberg du bras ULM

Après avoir placé les repères et désigné les différents paramètres du système, on peut écrire le tableau des paramètres D&H ci-après. Avec :

d_1 = 290mm.

d_2 = 108.5mm.

d_4 = 389.5mm.

d_{eff} = 138.12mm.

a_3 = 417.6mm.

Axe	α_{i-1}	a_{i-1}	d_i	θ_i
1	0	0	d_1	θ_1
2	$-\pi/2$	0	d_2	θ_2
3	0	a_3	0	θ_3
4	$-\pi/2$	0	d_4	θ_4
5	$\pi/2$	0	0	θ_5
6	$-\pi/2$	0	0	θ_6
7	0	0	d_{eff}	0

Tableau 3.1 : Paramètres de Denavit-Hartenberg du bras ULM.

Pour la vérification du modèle obtenu, nous avons simulé le modèle du D&H du bras ULM avec MATLAB et ses outils destinés à cet effet [35], [36], le modèle simulé ressemble bien à la structure réelle du robot ULM, et voici le résultat de ce que nous avons obtenu :

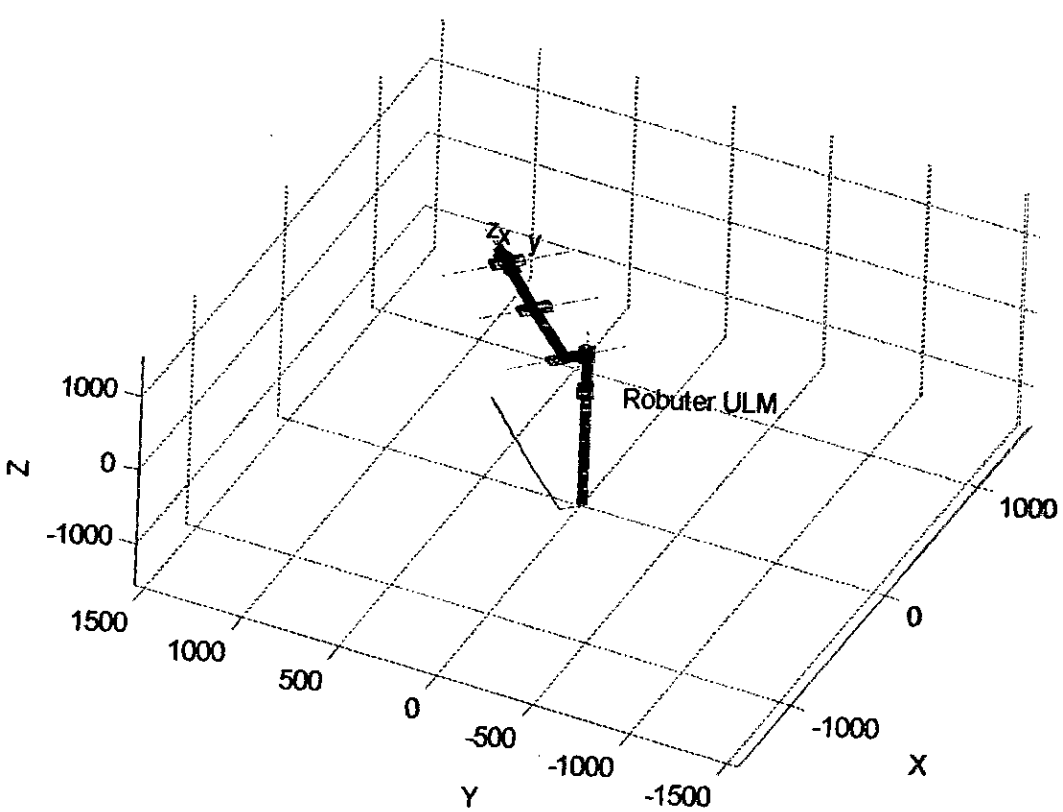


Figure 3.10 : Simulation et validation par MATLAB du MG du bras

3.5.1.2 Calcul des matrices homogènes

Les paramètres DH des robots donnés au tableau précédent nous permet de calculer les matrices de passages élémentaires d'un repère à un autre. Elles sont données comme suit :

$$T_0^1 = \begin{bmatrix} C_1 & -S_1 & 0 & 0 \\ S_1 & C_1 & 0 & 0 \\ 0 & 0 & 1 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad T_1^2 = \begin{bmatrix} C_2 & -S_2 & 0 & 0 \\ 0 & 0 & -1 & -d_2 \\ S_2 & C_2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_2^3 = \begin{bmatrix} C_3 & -S_3 & 0 & a_3 \\ S_3 & C_3 & 0 & d_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad T_3^4 = \begin{bmatrix} C_4 & -S_4 & 0 & 0 \\ 0 & 0 & -1 & -d_4 \\ S_4 & C_4 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_4^5 = \begin{bmatrix} C_5 & -S_5 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -S_5 & -C_5 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad T_5^6 = \begin{bmatrix} C_6 & -S_6 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ S_6 & C_6 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_6^{pince} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_{eff} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Avec : $C_i = Cos(\theta_i)$, $C_{ij} = Cos(\theta_{ij})$, $S_i = Sin(\theta_i)$ et $S_{ij} = Sin(\theta_{ij})$

Le modèle géométrique direct MGD du bras s'obtient en multipliant les six matrices élémentaires entre eux.

$$T_0^6 = T_0^1 . T_1^2 . T_2^3 . T_3^4 . T_4^5 . T_5^6 \qquad (3.24)$$

Après calcul, on trouve :

$$T_0^6 = \begin{bmatrix} a_x & n_x & s_x & p_x \\ a_y & n_y & s_y & p_y \\ a_z & n_z & s_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.25)$$

Où :

La position de l'organe terminal du robot ULM est donnée par les équations suivantes :

$$P_x = C_1.S_{23}.d_4 + C_1C_2a_3 - C_1S_2d_3 - S_1.d_2 + d_{eff}((C_1C_{23}C_4 + S_1C_4)S_5 + C_1S_{23}C_5) \quad (3.26)$$

$$P_y = S_1.S_{23}.d_4 + S_1C_2a_3 - S_1S_2d_3 - C_1.d_2 + d_{eff}((S_1C_{23}C_4 - C_1S_4)C_5 + S_1S_{23}C_5) \quad (3.27)$$

$$P_z = -C_{23}.d_4 + S_1a_3 + C_2d_3 + d_1 + d_{eff}(S_{23}C_4S_5 + C_{23}C_5) \quad (3.28)$$

et l'orientation du ULM par les neuf équations suivantes :

$$a_x = [(C_1.C_{23}.C_4 + S_1S_4)S_5 - C_1S_{23}S_5].C_6 + (-C_1C_{23}S_4 + S_1C_4)S_6 \quad (3.29)$$

$$a_y = [(S_1.C_{23}.C_4 - C_1C_4)C_5 - S_1S_{23}S_5].C_6 + (-S_1C_{23}S_4 - C_1C_4)S_6 \quad (3.30)$$

$$a_z = (S_{23}C_4C_5 + C_{23}S_5)C_5 + (-S_{23}S_{44})S_6 \quad (3.31)$$

$$n_x = -[(C_1.C_{23}.C_4 + S_1.S_4).C_5 - C_1.S_{23}.S_5]S_6 + (-C_1.C_{23}.S_4 + S_1.C_4).C_6 \quad (3.32)$$

$$n_y = -[(S_1.C_{23}.C_4 - C_1.C_4).C_5 - S_1.S_{23}.S_5]S_6 + (-S_1.C_{23}.S_4 - C_1.C_4).C_6 \quad (3.33)$$

$$n_z = -[(S_{23}.C_4.C_5 + C_{23}.S_5)]S_6 + (-S_{23}.S_4).C_6 \quad (3.34)$$

$$s_x = [C_1.C_{23}.C_4 + S_1.C_4]S_5 + C_1.S_{23}.C_5 \quad (3.35)$$

$$s_y = [S_1.C_{23}.C_4 - C_1.S_4]C_5 + S_1.S_{23}.C_5 \quad (3.36)$$

$$s_z = S_{23}.C_4.S_5 - C_{23}.C_5 \quad (3.37)$$

3.5.2 Modèle géométrique inverse du bras ULM

Le modèle géométrique inverse représente le problème inverse du modèle géométrique direct. Donc il consiste à calculer les coordonnées articulaires qui amènent l'organe terminal dans une situation désirée, spécifiée par ses coordonnées opérationnelles.

Pour obtenir le modèle géométrique inverse du bras manipulateur nous avons utilisé une approche analytique, cette approche se déroule en trois étapes [37] :

- Calcul de la matrice de passage homogène désirée.
- Ecriture d'un système linéaire de 9 équations fonction des angles θ_i (variables généralisés).
- Extraction des variables généralisés à partir des 9 équations.

Le détail de calcul et les formules des variables généralisées obtenus sont donnés en Annexe C, le lecteur peut le consulter pour plus d'informations.
Finalement on trouve que pour une position donnée on a généralement 16 configurations possibles :

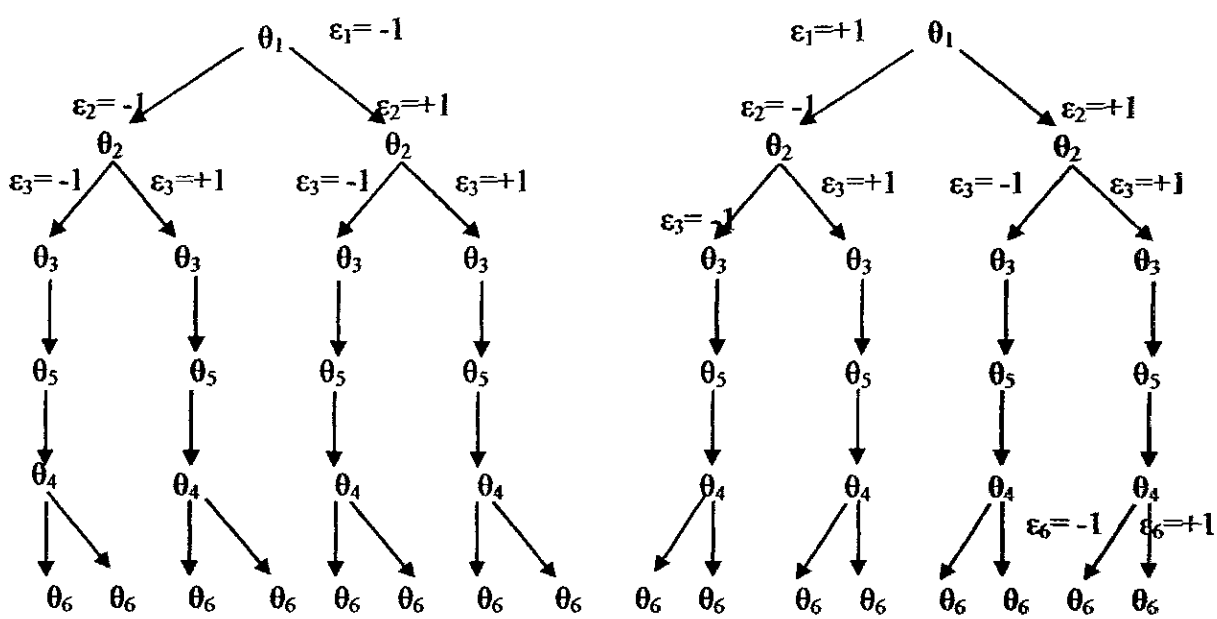


Figure 3.11. Les solutions possibles pour le MGI

3.6 Calibration de la caméra

La localisation de points dans un espace tridimensionnel au moyen d'un système de vision implique la connaissance des relations géométriques entre le dispositif impliqué, c'est-à-dire la caméra et l'objet. Ces relations géométriques se trouvent dans les paramètres des caméras, à savoir la taille des points dans l'image, la distance focale de la caméra, le centre de projection et la position de la

caméra dans la scène. Le processus de calcul de ces paramètres est appelé la calibration de caméra [38].

La calibration de caméra permet de déterminer le passage des cordonnées 2D images d'un point aux cordonnées 3D réelles de même point et inversement. La détermination de la relation 3D/2D passe par le choix préalable d'un modèle de caméra et par une phase dite de calibrage destinée à identifier les paramètres de ce modèle.

La modélisation des caméras consiste à obtenir les paramètres correspondant à la transformation directe et inverse effectuée pour passer des points d'un objet 3D dans l'espace aux points de sa projection 2D dans le plan image. Plusieurs méthodes sont proposés et étudier dans la littérature comme : la méthode à base d'ellipses [39], la méthode du modèle sténopé ou « de Pin-Hole » [40], méthode des points de fuites : [41], [42], [43], la méthode des multi-plans [44], [45], [46], et la méthode de Faugeras et Toscani [47].

La calibration de la caméra consiste à calculer les valeurs des paramètres du modèle choisi. Pour ce faire, il existe plusieurs approches [48], [49], [50], [51], la régression linéaire (moindres carrés), l'optimisation non linéaire, sont quelques unes parmi les méthodes utilisées.

3.6.1 Le modèle géométrique de la caméra

Dans notre cas on a utilisé le modèle sténopé, car c'est une représentation linéaire de la projection perspective et il présente une simplification considérable du calcul.

Ce modèle consiste à simplifier l'ensemble des lentilles que composent le système optique par un point où convergent tous les rayons lumineux pour aller se projeter sur le plan image par des droites sécantes en O_c (origine du repère de la caméra). L'image d'un objet est obtenue par sa projection perspective sur le plan image.

On représente généralement une caméra idéale par un modèle dit du sténopé ([38], [49], [52]) muni d'un plan image sans distorsions dont l'axe optique est considéré comme la normale au plan image. Le modèle sténopé, régi par les théorèmes de la géométrie projective centrale, est une représentation linéaire de

la projection perspective. On peut schématiser une telle modélisation par la figure suivante :

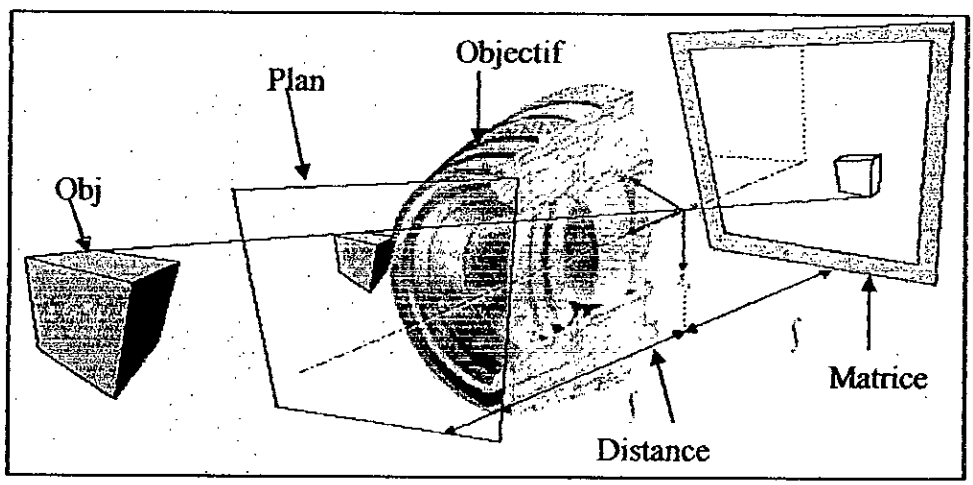


Figure 3.12: Modèle sténopé d'une caméra CCD. [49]

Les différents repères et projections perspectives mis en jeux sont représentés sur la figure suivante [49] :

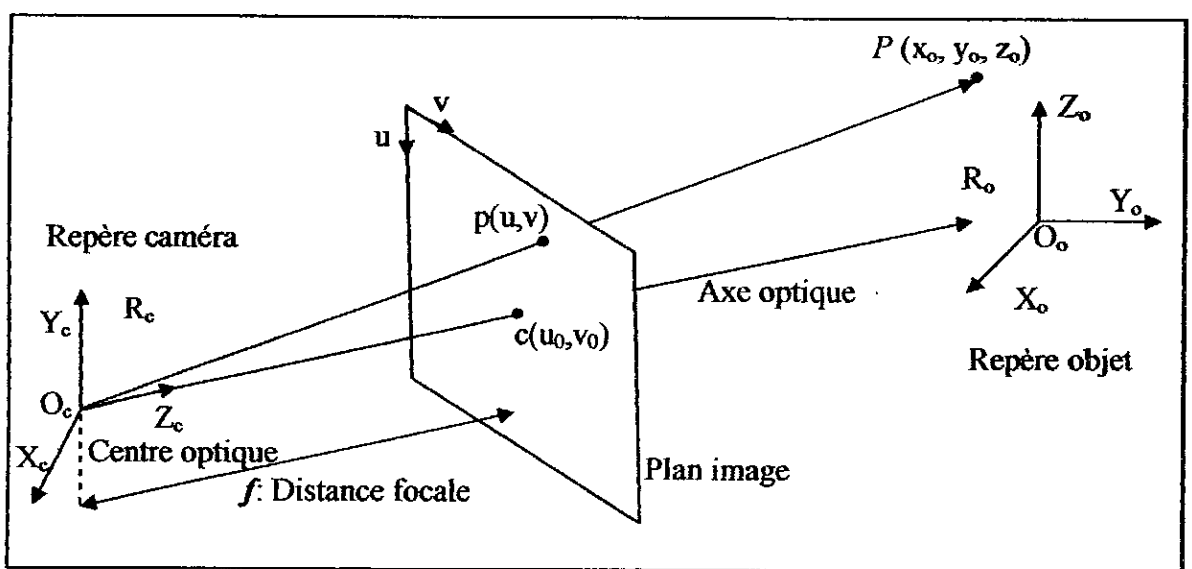


Figure 3.13 : Repères et projection perspective du modèle sténopé.

A partir de ce modèle nous avons par définition :

- Tout point de l'espace se projette sur le plan image suivant une droite passant par le centre de la lentille.
- Le point O_c (le centre optique ou de projection) représente l'origine du repère caméra.

- Le plan image est perpendiculaire à l'axe optique de la caméra.
- Le plan image se situe à une distance *f* (distance focale de la lentille) du centre optique.
- L'axe (O_c, X_c) représente l'abscisse du plan image.
- L'axe (O_c, Y_c) représente l'ordonnée du plan image.
- Le point c(u_0, v_0) est l'intersection de l'axe optique (O_c, Z_c) et le plan image.

Soit $P(x_o, y_o, z_o)$ un point de l'espace exprimé dans un référentiel fixe $R_0(O_o, X_o, Y_o, Z_o)$ et p(u,v) le pixel représentant sa projection sur l'image. Nous voulons trouver une relation qui modélise le passage entre P et p et calculer la matrice **M** traduisant cette relation, qui est telle que :

$$p = M.P \tag{3.38}$$

Donc $P = M^{-1}.p$

$$\begin{bmatrix} u \\ v \end{bmatrix} = M.\begin{bmatrix} x_o \\ y_o \\ z_o \end{bmatrix}$$

Donc : $$\begin{bmatrix} x_o \\ y_o \\ z_o \end{bmatrix} = M^{-1}.\begin{bmatrix} u \\ v \end{bmatrix}$$

Où M est la matrice de passage, ou encore la matrice jacobéenne. La matrice de passage M est le résultat du produit des deux matrices, la matrice de paramètres intrinsèques et la matrice de paramètres extrinsèques, M = Mint . Mext

Les transformations nécessaires entre les différents repères sont illustrées par la figure suivante :

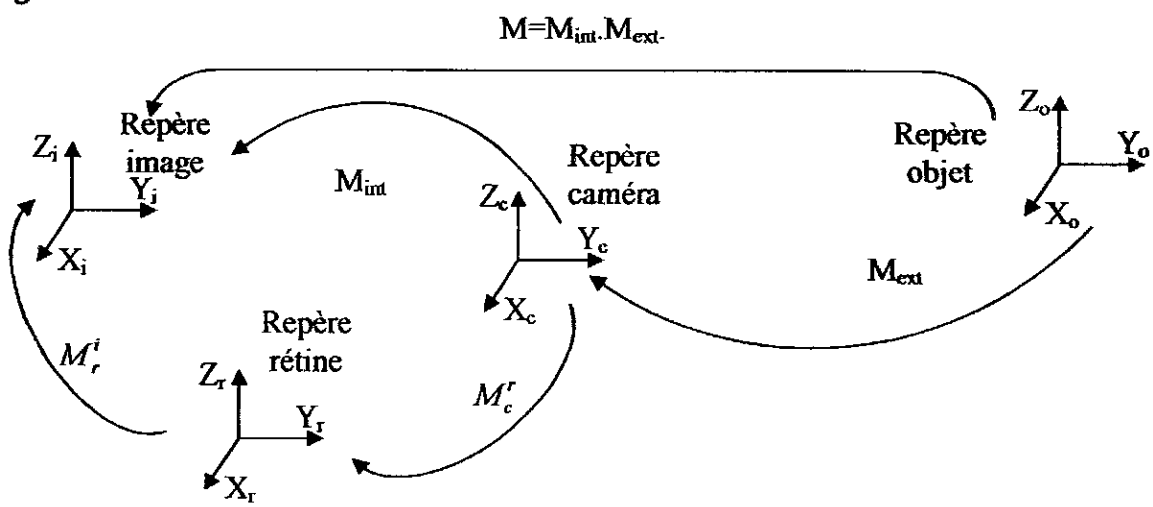


Figure 3.14: représentation des différents repères

3.6.1.1 Transformation Espace - Caméra (Paramètres extrinsèques)

Un point de l'espace *P* est défini par ses coordonnées (X_o, Y_o, Z_o) dans le repère objet et également il est défini par ses coordonnées (X_c, Y_c, Z_c) dans le repère caméra. La transformation utilisée pour passer du repère espace au repère caméra est une transformation homogène notée M_{ext}, elle est exprimée comme suite :

$$\begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} = M_{ext} \cdot \begin{bmatrix} x_o \\ y_o \\ z_o \\ 1 \end{bmatrix} \tag{3.39}$$

M_{ext} : appelée matrice des paramètres extrinsèques. La matrice M_{ext} peut s'écrire sous la forme :

$$M_{ext} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} [R] & [T] \\ 0 & 1 \end{bmatrix} \tag{3.40}$$

Etant donné l'orthogonalité de la matrice de rotation, l'inverse de *P* est l'expression suivante :

$$M_{ext}^{-1} = \begin{bmatrix} R^T & -R^T.T \\ 0 & 1 \end{bmatrix} \quad si \ \det(R) \neq 0. \tag{3.41}$$

3.6.1.2 Transformation Caméra -Image (Paramètres intrinsèques)

En projection perspective, un point M dans le repère caméra est transformé en son homologue m dans le plan image par la relation :

$$\begin{bmatrix} u \\ v \end{bmatrix} = M_{int} \cdot \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} \tag{3.42}$$

Cette transformation peut être mise en jeu par deux transformations : la première passe du repère caméra vers le repère rétine, et la seconde du repère rétine vers

le repère image. La figure suivante nous montre les différents paramètres intrinsèques mis en jeu :

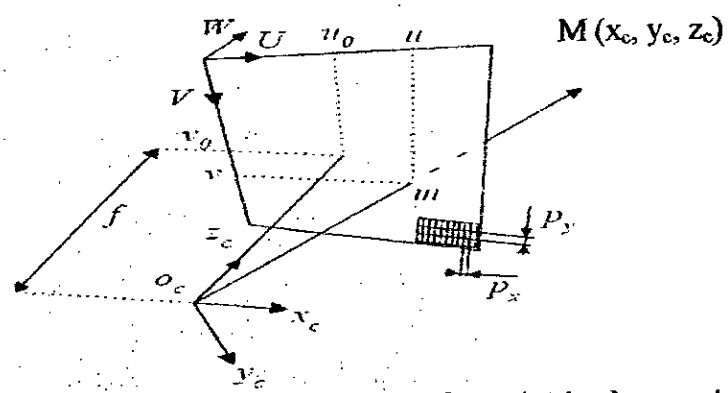


Figure 3.15: Représentation des paramètres intrinsèques de la caméra

3.6.1.3 Transformation Caméra –Rétine

Nous voulons trouver les coordonnées de la projection *m* du point *Mc* sur la rétine. Le point *Mc* est représenté sur le repère de la caméra. Le repère rétine et le repère image sont confondus, donc le point m dans le plan image se trouve sur le point m sur le plan rétine. La projection perspective d'un point M sur le plan rétine, établie à partir des triangles semblables, a une expression canonique suivante :

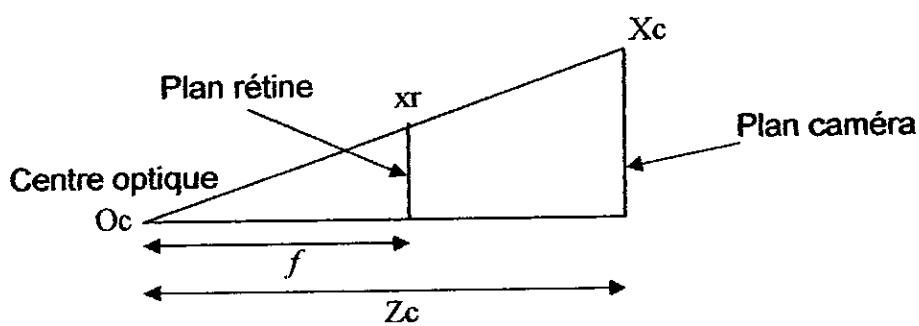


Figure 3.16 : Le triangle semblable

En utilisant les propriétés des triangles semblables (figure 3.19). Nous avons donc :

$$\begin{cases} \dfrac{X_c}{x_r} = \dfrac{Z_c}{f} \\ \dfrac{Y_c}{y_r} = \dfrac{Z_c}{f} \end{cases} \qquad (3.43)$$

De cette relation on a :

$$\begin{cases} X_c = \dfrac{Z_c}{f}.x_r \\ Y_c = \dfrac{Z_c}{f}.y_r \end{cases} \tag{3.44}$$

Soit en notation matricielle :

$$\begin{bmatrix} wx_r \\ wy_r \\ w \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \dfrac{1}{f} \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} \tag{3.45}$$

Avec $w = \dfrac{Z_c}{f}$ est le facteur d'échelle.

3.6.1.4 Transformation Rétine –Image

Les coordonnées du repère image sont en unité pixel. Pour rendre la transformation homogène, il est nécessaire d'introduire l'unité pixel (*px*, *py*) du système de coordonnées image :

$$\begin{cases} u = u_0 + \dfrac{x_r}{p_x} \\ v = v_0 + \dfrac{y_r}{p_y} \end{cases} \tag{3.46}$$

Ou encore :

$$\begin{cases} u = u_0 + f.\dfrac{x_r}{p_x.Z_c} \\ v = v_0 + f.\dfrac{y_r}{p_y.Z_c} \end{cases} \tag{3.47}$$

Soit en notation matricielle :

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{p_x} & 0 & u_0 \\ 0 & \frac{1}{p_y} & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_r \\ y_r \\ 1 \end{bmatrix} \tag{3.48}$$

Finalement on peut écrire :

$$\begin{bmatrix} U \\ V \\ w \end{bmatrix} = \begin{bmatrix} \frac{1}{p_x} & 0 & u_0 \\ 0 & \frac{1}{p_y} & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{1}{f} & 0 \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} \tag{3.49}$$

Avec : $\begin{cases} u = \frac{U}{W} \\ v = \frac{V}{W} \end{cases}$

Donc la matrice des paramètres intrinsèques M_{int} peut s'écrire sous la forme :

$$M_{int} = \begin{bmatrix} \frac{1}{p_x} & 0 & u_0 \\ 0 & \frac{1}{p_y} & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{1}{f} & 0 \end{bmatrix} = \begin{bmatrix} \frac{1}{p_x} & 0 & \frac{u_0}{f} & 0 \\ 0 & \frac{1}{p_y} & \frac{v_0}{f} & 0 \\ 0 & 0 & \frac{1}{f} & 0 \end{bmatrix} \tag{3.50}$$

Donc on a :

$$\begin{bmatrix} U \\ V \\ w \end{bmatrix} = \begin{bmatrix} \frac{1}{p_x} & 0 & \frac{u_0}{f} & 0 \\ 0 & \frac{1}{p_y} & \frac{v_0}{f} & 0 \\ 0 & 0 & \frac{1}{f} & 0 \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix}$$

Finalement on trouve la matrice jacobéenne de la caméra $M=M_{int}.M_{ext}$ sous la forme :

$$\begin{bmatrix} U \\ V \\ w \end{bmatrix} = \begin{bmatrix} \frac{1}{p_x} & 0 & \frac{u_0}{f} & 0 \\ 0 & \frac{1}{p_y} & \frac{v_0}{f} & 0 \\ 0 & 0 & \frac{1}{f} & 0 \end{bmatrix} \begin{bmatrix} \cos\beta\cos\gamma & \sin\alpha\sin\beta\cos\gamma-\cos\alpha\sin\gamma & \cos\alpha\sin\beta\cos\gamma+\sin\alpha\sin\gamma & t_x \\ \cos\beta\sin\gamma & \sin\alpha\sin\beta\sin\gamma+\cos\alpha\cos\gamma & \cos\alpha\sin\beta\sin\gamma-\sin\alpha\cos\gamma & t_y \\ -\sin\beta & \sin\beta\cos\beta & \cos\alpha\cos\beta & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_o \\ y_o \\ z_o \\ 1 \end{bmatrix} \quad (3.51)$$

La transformation projective complète, du repère objet (univers) au repère image est de la forme matricielle suivante :

$$\begin{bmatrix} U \\ V \\ W \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \end{bmatrix} \begin{bmatrix} x_o \\ y_o \\ z_o \\ 1 \end{bmatrix}$$

Où les différents coefficients qui composent la matrice de projection perspective sont :

$$m_{11} = \cos\beta\cos\gamma.\frac{1}{p_x} - \sin\beta.\frac{u_0}{f} \quad (3.52)$$

$$m_{12} = (\sin\alpha\sin\beta\cos\gamma - \cos\alpha\sin\gamma).\frac{1}{p_x} + \sin\beta\cos\beta.\frac{u_0}{f} \quad (3.53)$$

$$m_{13} = (\cos\alpha\sin\beta\cos\gamma + \sin\alpha\sin\gamma).\frac{1}{p_x} + \cos\alpha\cos\beta.\frac{u_0}{f} \quad (3.54)$$

$$m_{14} = t_x.\frac{1}{p_x} + t_z.\frac{u_0}{f} \quad (3.55)$$

$$m_{21} = \cos\beta\sin\gamma.\frac{1}{p_y} - \sin\beta.\frac{v_0}{f} \quad (3.56)$$

$$m_{22} = (\sin\alpha\sin\beta\sin\gamma + \cos\alpha\cos\gamma).\frac{1}{p_y} + \sin\beta\cos\beta.\frac{v_0}{f} \quad (3.57)$$

$$m_{23} = (\cos\alpha\sin\beta\sin\gamma - \sin\alpha\cos\gamma).\frac{1}{p_y} + \cos\alpha\cos\beta.\frac{u_0}{f} \quad (3.58)$$

$$m_{24} = t_y.\frac{1}{p_y} + t_z.\frac{v_0}{f} \quad (3.59)$$

$$m_{31} = -\sin\beta \tag{3.60}$$

$$m_{32} = \sin\beta\cos\beta \tag{3.61}$$

$$m_{33} = \cos\alpha\cos\beta \tag{3.62}$$

$$m_{34} = t_z \tag{3.63}$$

$$m_{41} = 0, \quad m_{42} = 0, \quad m_{43} = 0, \quad m_{44} = 1 \tag{3.64}$$

Faugeras et Toscani [47] font remarquer deux points :

- Premièrement, que les coefficients m_{31}, m_{32}, m_{33} ne sont pas influencés par les paramètres intrinsèques.
- Deuxièmement, que la contrainte d'orthogonalité de la matrice de rotation associée à ces coefficients reste vraie, $m^2_{31} + m^2_{32} + m^2_{33} = 1$ et permet d'extraire, d'un système d'équations non linéaires, les paramètres recherchés, c'est à dire, la distance focale et le centre optique.

La projection perspective s'obtient par les équations canoniques et homogènes :

$$u = \frac{U}{W} = \frac{m_{11}.x_o + m_{12}.y_o + m_{13}.z_o + m_{14}}{m_{31}.x_o + m_{32}.y_o + m_{33}.z_o + m_{34}}$$
$$v = \frac{V}{W} = \frac{m_{21}.x_o + m_{22}.y_o + m_{23}.z_o + m_{24}}{m_{31}.x_o + m_{32}.y_o + m_{33}.z_o + m_{34}} \tag{3.65}$$

On peut développer ces équations par :

$$m_{11}.x_o + m_{12}.y_o + m_{13}.z_o + m_{14} - u.(m_{31}.x_o + m_{32}.y_o + m_{33}.z_o) = u.m_{34}$$

$$m_{21}.x_o + m_{22}.y_o + m_{23}.z_o + m_{24} - v.(m_{31}.x_o + m_{32}.y_o + m_{33}.z_o) = v.m_{34}$$

On remarque que pour chaque point et sa projection on a deux équations avec 12 inconnues m_{ij}.

On prend m_{34}=1 pour simplifier la résolution du système, on trouve :

$$\begin{bmatrix} x_{o1} & y_{o1} & z_{o1} & 1 & 0 & 0 & 0 & 0 & -x_{o1}.u_1 & -y_{o1}.u_1 & -z_{o1}.u_1 \\ 0 & 0 & 0 & 0 & x_{o1} & y_{o1} & z_{o1} & 1 & -x_{o1}.v_1 & -y_{o1}.v_1 & -z_{o1}.v_1 \\ & & & & & & \vdots & & & & \\ x_{on} & y_{on} & z_{on} & 1 & 0 & 0 & 0 & 0 & -x_{on}.u_1 & -y_{on}.u_1 & -z_{on}.u_1 \\ 0 & 0 & 0 & 0 & x_{on} & y_{on} & z_{on} & 1 & -x_{on}.v_1 & -y_{on}.v_1 & -z_{on}.v_1 \end{bmatrix} \begin{bmatrix} m_{11} \\ m_{12} \\ m_{13} \\ m_{14} \\ m_{21} \\ m_{22} \\ m_{23} \\ m_{24} \\ m_{31} \\ m_{32} \\ m_{33} \end{bmatrix} = \begin{bmatrix} u_1 \\ v_1 \\ \vdots \\ u_n \\ v_n \end{bmatrix} \quad (3.66)$$

Cette relation est de la forme A.X=B [36]. Les éléments de la matrice *A* sont tirés des points dans l'espace $M_{oi} = (X_{oi}, Y_{oi}, Z_{oi}, 1)$ et leurs projections sur le plan image, les éléments de la matrice *B* sont tirés de leurs projections sur le plan image Bi $(u_i, v_i, 1)$ tandis que le vecteur X représente les 11 paramètres à déterminer. La solution de ce système est la pseudo inverse de X [38] :

$$x = (A^T.A)^{-1}.A^T.B \quad (3.67)$$

Où : $(A^T.A)^{-1}.A^T$ est la matrice pseudo inverse de la matrice *A*.

3.6.2 Calibration de la caméra

La calibration d'une caméra consiste à estimer la matrice de projection ***M*** en connaissant les coordonnées de plusieurs points dans l'espace et les coordonnées de leurs projections sur l'image. A cet effet deux systèmes de calibration sont utilisés : les grilles de calibration et les mires de calibration.

Les systèmes de calibration utilisés, grilles et mires de calibrations sont bien illustrés dans l'annexe C. les résultats de cette opération (exactement les matrices de passages) sont donnés et expliqués dans le dernier chapitre de ce manuscrit (chapitre 4 : mise en œuvre expérimentale et résultats).

3.7 Conclusion

Ce chapitre nous a permis de présenter la problématique de notre travail, qui est l'obtention des coordonnées réelles des objets perçus par le système de vision, la localisation du robot par rapport au plan des objets, l'élaboration des lois de commandes convenables du bras manipulateur pour agir sur l'environnement et le positionnement de l'organe terminal au dessus des objets à manipuler.
Pour cela nous avons présenté une modélisation mathématique de notre système robotique. La modélisation cinématique de la plate forme mobile nous a permis d'obtenir les équations nécessaires pour faire mouvoir cette dernière et faire de la navigation. La modélisation géométrique du bras nous a donné les équations nécessaires pour pouvoir positionner l'organe terminal dans l'espace manipulable et atteignable du bras. La calibration de la caméra décrite plus haut est sans doute l'outil clé de notre application, à travers laquelle nous pouvons savoir où se trouvent les objets à manipuler par rapport à la position actuelle du robot.

Afin de définir les relations mathématiques mises en jeu, on entamera l'implémentation logicielle du système de téléopération proposé, c'est ce que l'on va voir dans le chapitre suivant.

CHAPITRE 4
MISE EN OEUVRE EXPERIMENTALE ET RESULTATS

4.1 Introduction

La mise en place du système de téléopération nécessite plusieurs ressources matérielles et logicielles. Il est évident que le robot manipulateur mobile et ses accessoires représentent la partie la plus essentielle dans l'architecture de notre système de téléopération puisque c'est à travers eux que l'on puisse agir sur l'environnement.

Ce système robotique est en fait un assemblage de pièces mécaniques, il nécessite alors des logicielles de pilotage pour le mettre en service. D'où le développement d'une application logicielle, pour la gestion du matériel permettant ainsi à l'opérateur d'accomplir les tâches nécessaires.

Cette application logicielle devra être en mesure de traiter les images acquises par la caméra et les différentes informations issues des capteurs associés au robot, d'y extraire les informations utiles, d'effectuer les calculs nécessaires pour localiser les objets cibles et d'obtenir la configuration du robot manipulateur correspondant à la tâche voulue.

Dans la suite de ce chapitre nous montrerons cette application et nous décrirons les algorithmes de déroulement mis en jeu. Les résultats obtenus et les tests fait sont détaillés dans ce chapitre, ces résultats concernent la calibration de la caméra CCD, la simulation virtuelle des missions et configurations du robot sur le simulateur SimRobuter et la validation réelle sur le robot dans son environnement de travail.

Dans la page suivante on donne un diagramme fonctionnel explicatif résumant le fonctionnement du système de téléoperation.

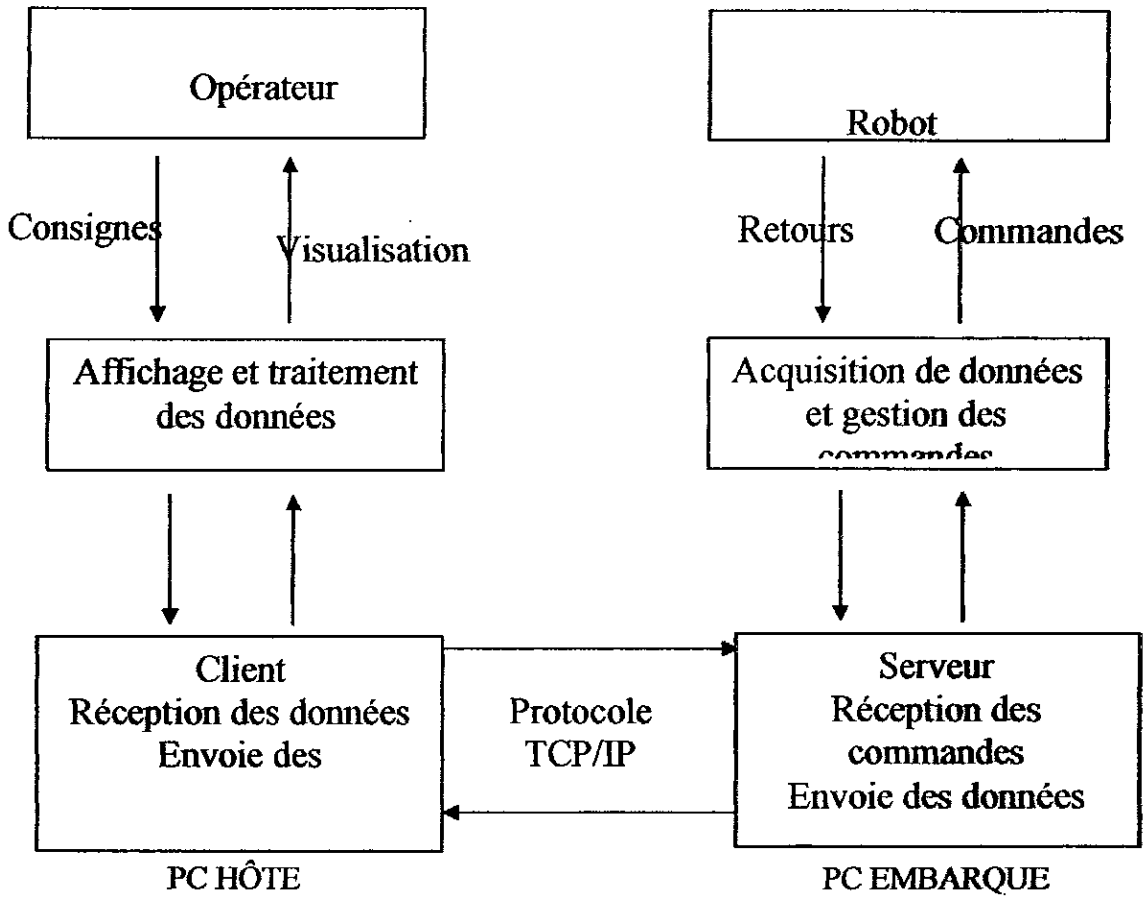


Figure 4.1 : Diagramme fonctionnel du système de téléopération.

L'utilisateur entre en communication indirecte avec le robot, par l'intermédiaire du PC hôte qui représente le Client et grâce à une interface de communication Homme machine, il agit sur le robot et décide comment ce dernier doit se déplacer et quelle tâche doit effectuer, tout en ayant la possibilité de recevoir des informations sur son état afin de contrôler son mouvement grâce à un système imageur qui a pour rôle l'acquisition des images concernant le robot et son environnement, Le PC embarqué, qui joue le rôle d'un serveur reçoit les informations qui sont envoyés via un réseaux de communication, élabore les lois de commande convenables pour le robot, assure leur exécution et reçoit les informations de retour venant des capteurs.

4.2 Algorithme global de l'application

De point de vue logiciel notre système de téléopération s'appuie essentiellement sur deux applications, une application client installée sur le PC

hôte et une application serveur sur le PC embarqué. Ces deux applications communiquent entre eux pour s'échanger les données nécessaires.

L'opérateur communique avec son système via une interface graphique IHM, cette dernière lui permet d'agir sur l'environnement et de valider des missions tout en visionnant des retours relatifs à l'état du robot. L'application de communication client permet à cette interface d'échanger les données nécessaires avec le robot.

D'un autre coté, le programme développé pour la machine embarquée est une application à trois niveaux :

- Une application serveur pour la gestion de la communication.
- Une application de bas niveau qui s'exécute sur les cartes électronique de commande, elle exploite les ressources système de façon directe.
- Une application SynDEx pour la mise en lien entre ces deux applications.

L'algorithme synthétisant ces fonctionnalités est illustré par le schéma suivant :

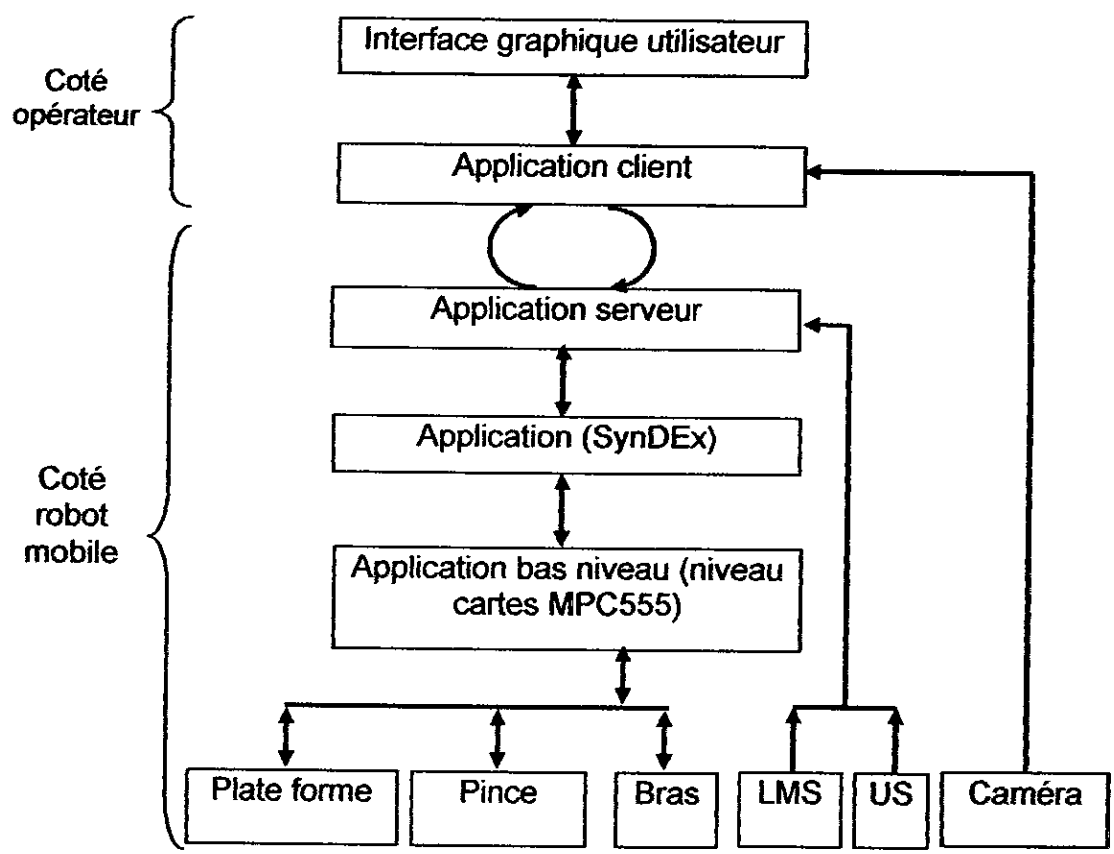


Figure 4.2 : l'organigramme de l'application de téléopération.

4.3 L'application serveur

Cette application est implémentée et exécutée sur la machine embarquée. Elle est écrite en langage C et compilée avec le compilateur GCC. Le rôle et les tâches à assurer sont expliqués par l'organigramme suivant:

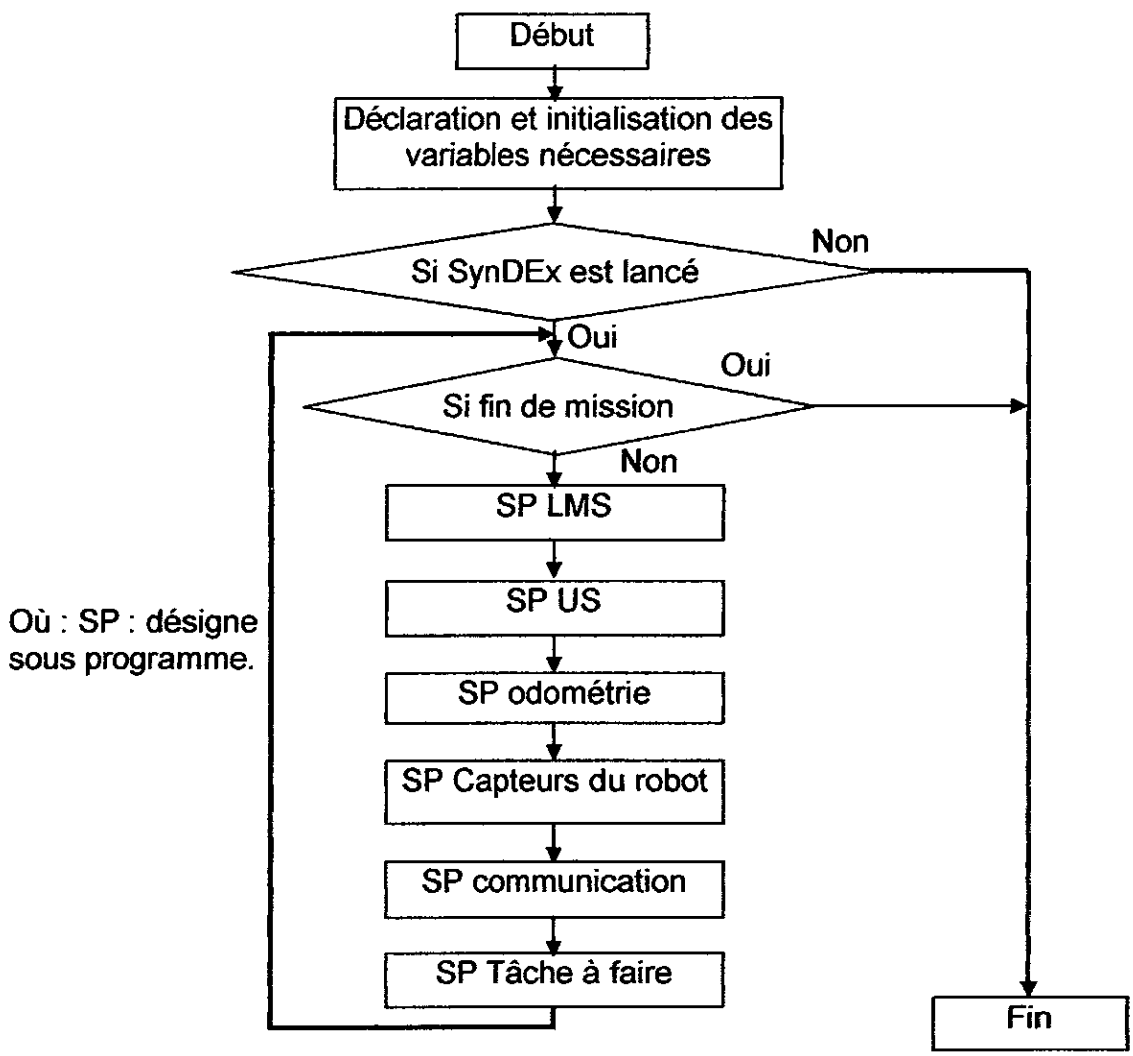


Figure.4.3 : L'organigramme de l'application serveur.

A l'exécution de l'application, le programme principale initialise les variables et les structures qu'il utilise, ensuite il teste la présence de l'application SynDEx, il fait une scrutation globale pour la mise à jour et la lecture de l'état de tous les capteurs installés sur le système, puis il se met en écoute pour attendre une éventuelle connexion avec le client. Dès qu'une demande de connexion de la part du client est reçue, le serveur accepte la communication et envoie les données au

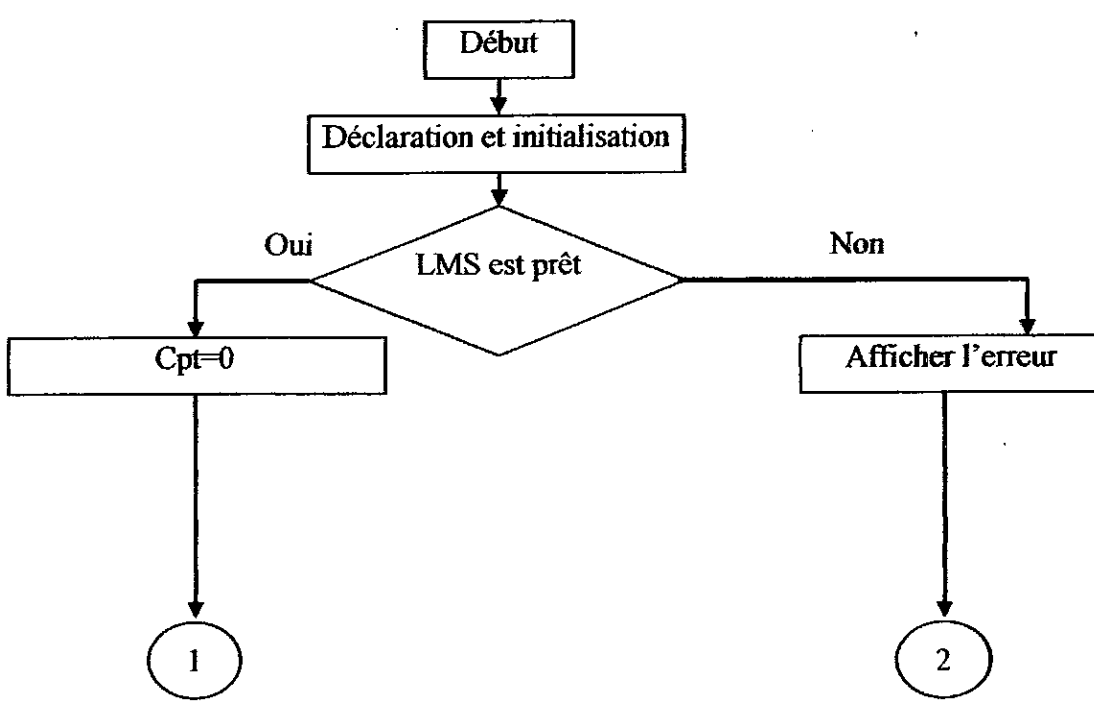
client, ensuite il se met à nouveau en écoute pour recevoir des données. L'organigramme illustré plus haut sera développé par l'explicitation des différents sous programmes et procédures donnés ci-après.

4.3.1 Déclaration et initialisation des variables nécessaires

Cette section représente l'entête du programme. Toutes les constantes et les variables nécessaires (zones mémoires, entiers, réels, tableaux, ...) sont déclarées et initialisées avec les valeurs qui conviennent.

4.3.2 Sous programme LMS

Le sous programme LMS se charge d'activer le télémètre Laser, de mesurer les distances et de sauvegarder ces mesures. Le LMS est connecté directement au port série du PC embarqué. Le programme principal échange les informations avec le LMS par le biais du port série en utilisant le protocole RS232. La forme développée de cette partie est donnée par :



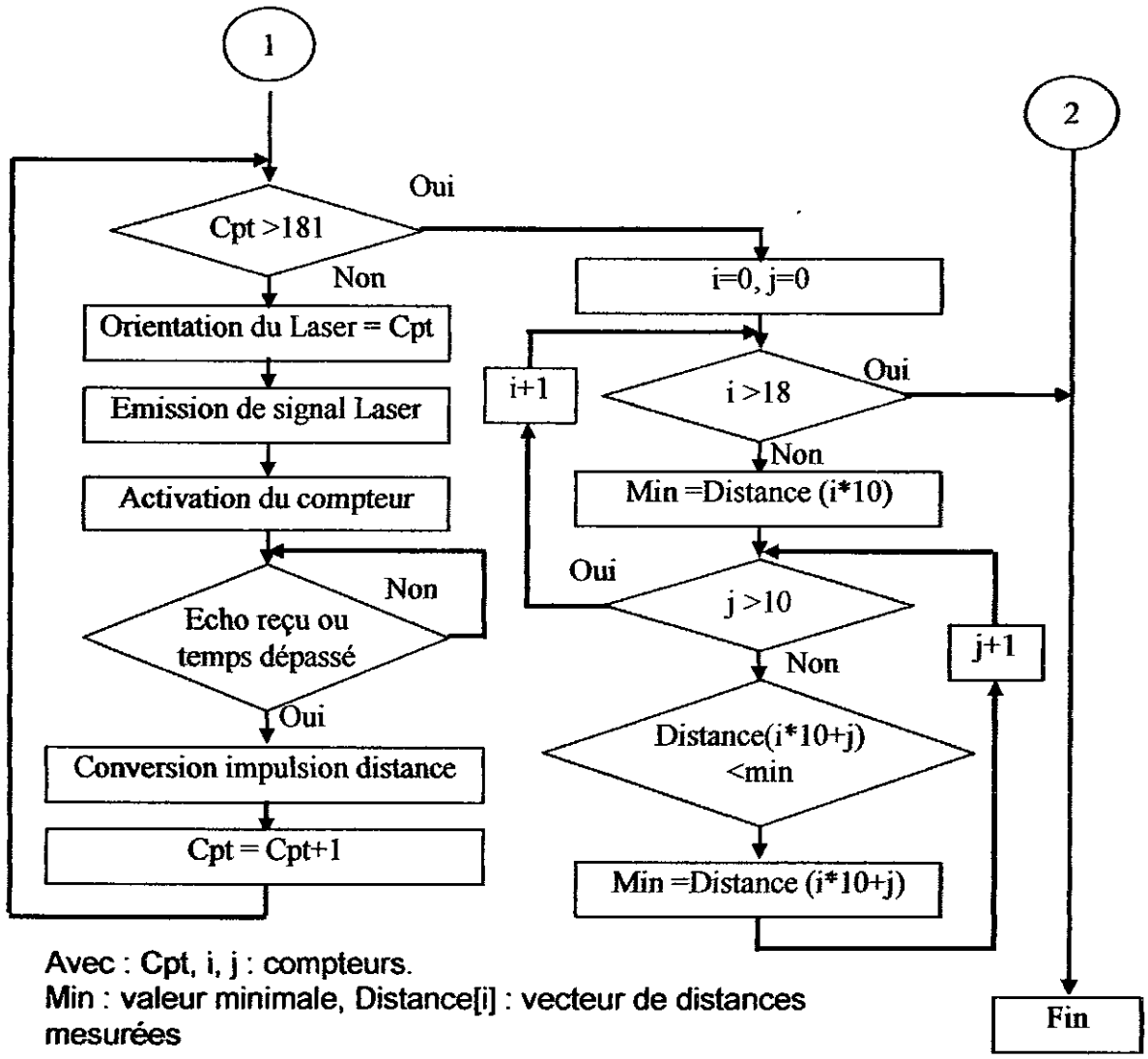


Figure.4.4 : L'organigramme de gestion du LMS.

Après les initialisations des variables et l'activation du LMS, le programme teste si ce dernier est prêt ou non, si non un message d'erreur signal le problème à l'opérateur, dans le cas contraire une acquisition de données sera faite. La mesure se déroule comme suit :

Pour tous les points du plan horizontal

- On envoie au système l'angle courant pour positionner le miroir interne.
- Un tir de laser est émis en même temps avec l'activation d'un compteur.
- Quand le circuit de réception reçoit le retour, on bloque le comptage, on lit la valeur et on calcul la distance mesurée (selon la formule donnée au chapitre 2).

4.3.3 Sous programme Ultrason (US)

La ceinture ultrasonique est connectée aux ports séries du PC embarqué, comme déjà mentionné, cette ceinture dénombre 24 cellules élémentaires, donc 24 capteurs ultrasoniques. Le programme d'exploitation de la ceinture est représenté par l'organigramme suivant :

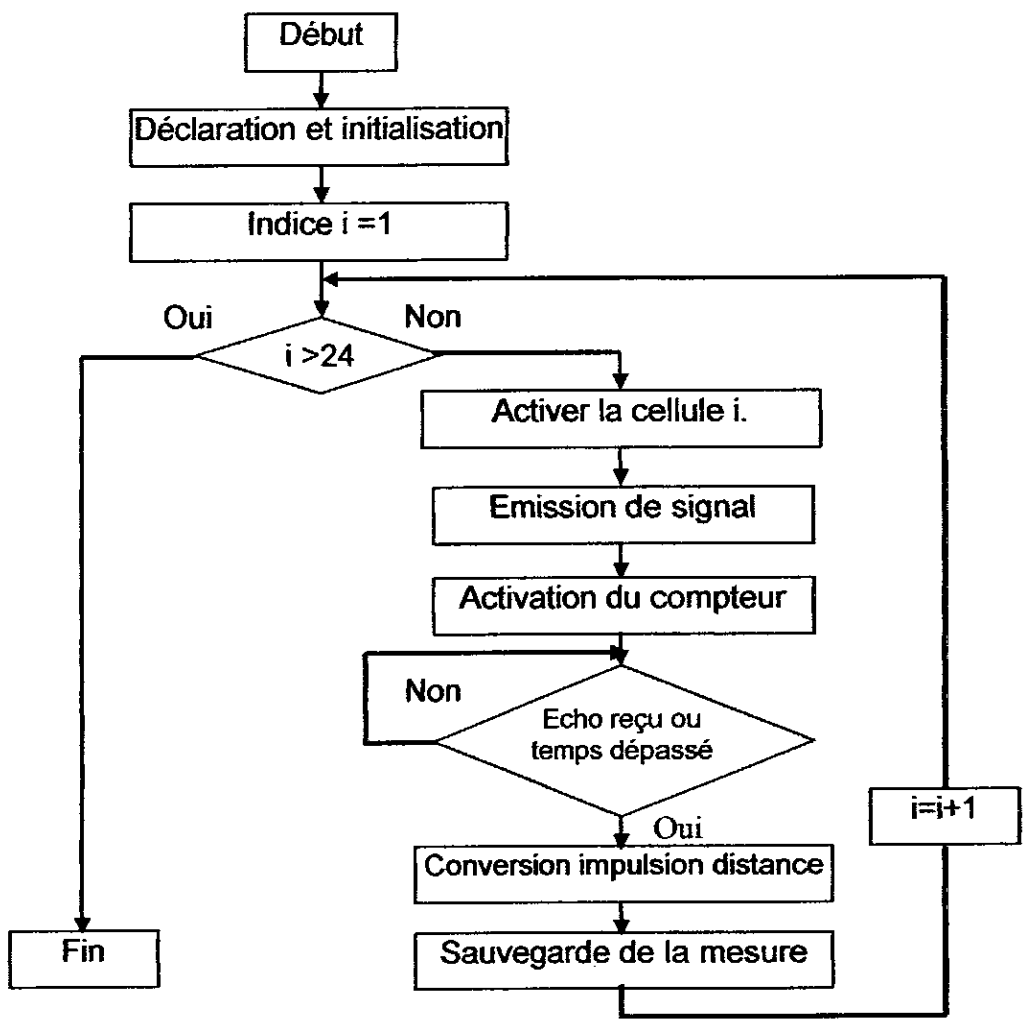


Figure.4.5: L'organigramme de gestion de la ceinture à ultrasons.

Les cellules sont activées l'une après l'autre pour éviter tout problème de chevauchement entre les ondes acoustiques, le principe de mesure de distance est le même que pour le LMS.

4.3.4 Sous programme odométrie

Ce programme lit l'état des codeurs incrémentaux installés au niveau des roues motrices et convertit les données lues en déplacement et en angle de rotation relatifs.

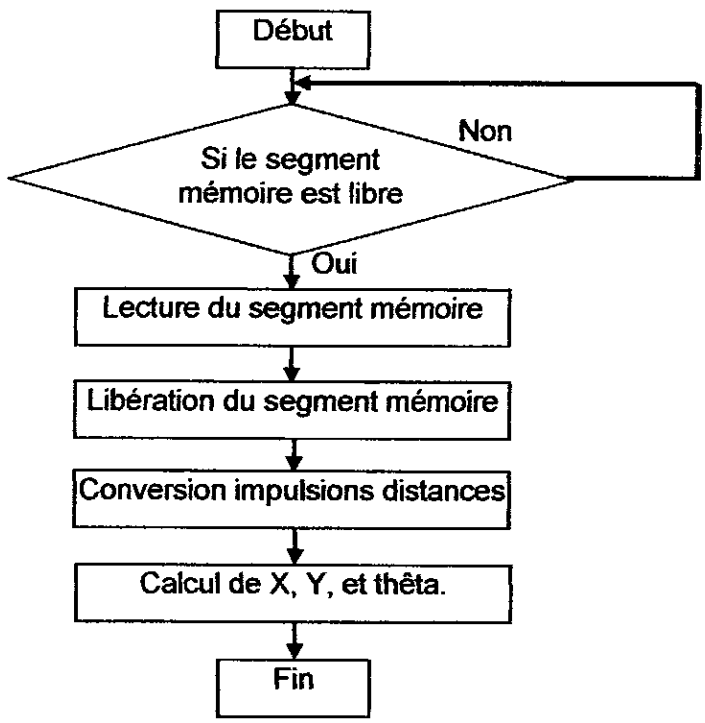


Figure.4.5 : L'organigramme de lecture de l'odométrie.

A noter ici que les segments de mémoire partagée sont déclarés au début du programme principal et SynDEx s'occupe de faire la mise à jour de leur contenu. Ces segments sont au nombre de quatre, un est réservé pour la pince, un autre pour la plate forme tandis que deux sont réservés pour le bras, dont un pour la lecture et l'autre pour l'écriture.

Ces segments mémoires sont le mécanisme d'échange de données entre le haut niveau et le bas niveau en temps réel. Ils sont partagés entre les deux applications.

4.3.5 Sous programme lecture des capteurs du bras

Ce programme permet de scruter les codeurs incrémentaux installés au niveau des actionneurs des segments du bras, lit également les informations issues du capteur d'effort et renvoie l'état de la pince (fermée ou ouverte).

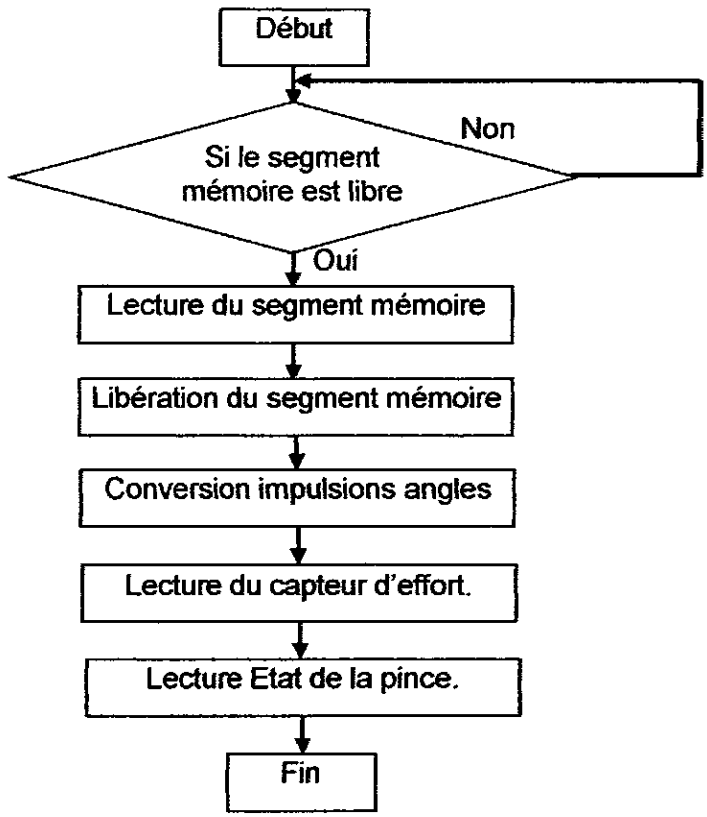


Figure.4.6 : L'organigramme de lecture des capteurs du bras.

4.3.6 Sous programme Communication

Cette sub-routine se charge de créer une Socket et se met en mode écoute pour attendre une éventuelle demande de connexion venant du client. Dès qu'une connexion est établie avec le client, l'échange de données se met en place. Le mécanisme de cette étape est bien expliqué dans le chapitre 2.

4.3.7 Sous programme tâche

A la réception des données issues du client, le serveur va les traiter, puis élabore les lois de commandes convenables. La stratégie d'exécution d'une tâche sera discuter plus loin.

4.4 L'application client

L'application client se voit comme une seule application qui a plusieurs tâches. Elle assure deux fonctions principales : la gestion de la communication avec le serveur, et la gestion d'une interface graphique de communication homme machine. L'interface de communication homme machine est l'application par laquelle l'opérateur dialogue avec le robot. Ce module est conçu pour assurer à l'opérateur les tâches essentielles suivantes:

- La prise en charge des commandes de l'utilisateur ;
- L'affichage et le traitement des images vidéo acquises en temps réel.
- Le traitement et l'affichage des informations sensorielles reçues.
- La fusion multi sensorielle des données reçues.
- La gestion du simulateur virtuel du robot pour permettre à l'utilisateur de voir les gestes que doit exécuter le robot.

La figure ci-après illustre la fenêtre principale de l'application développée.

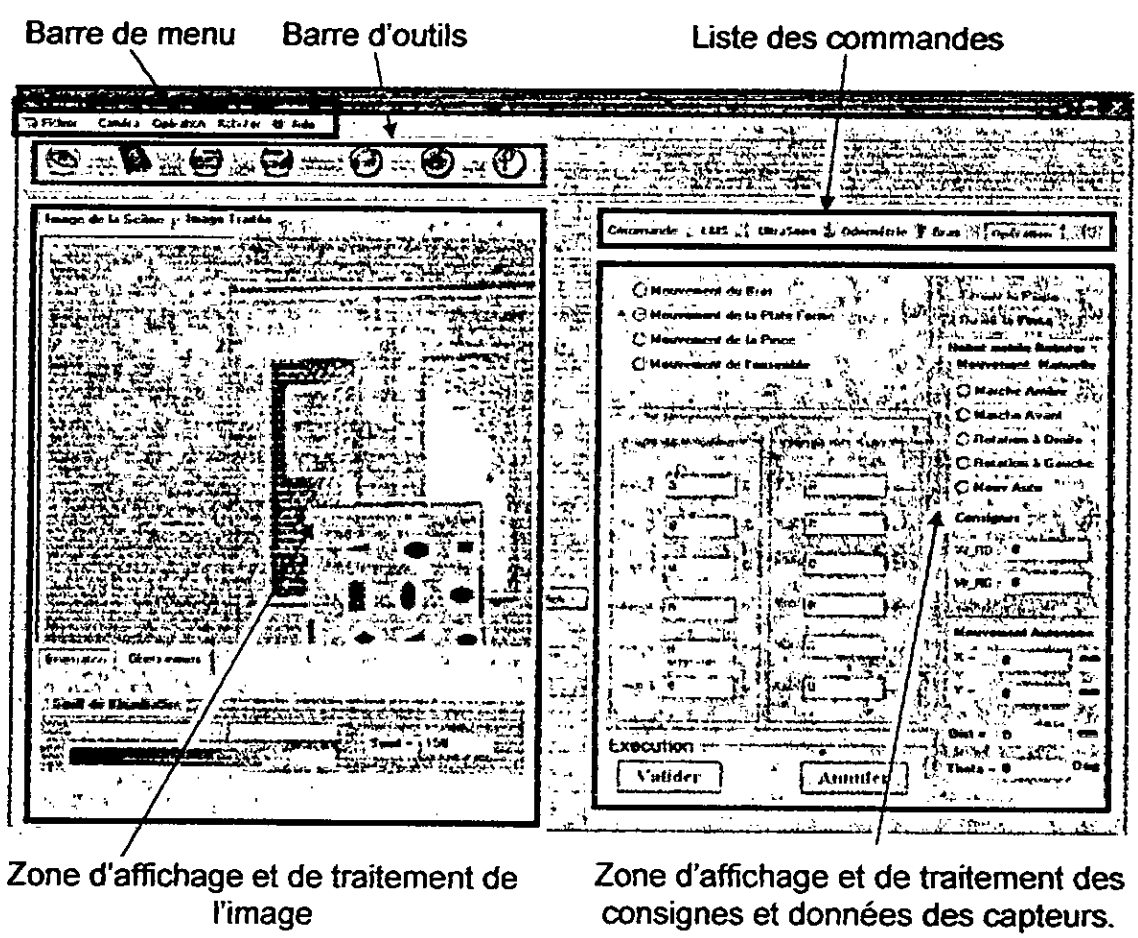


Figure.4.7 : Présentation de l'interface graphique.

4.4.1 Traitement de l'image

La caméra nous délivre en permanence des images vidéo de la scène. L'image fournie par la caméra vidéo n'est pas directement utilisable après numérisation, mais elle doit subir plusieurs opérations pour que nous puissions extraire l'information souhaitée. Le traitement comporte donc quatre phases essentielles : l'acquisition, le prétraitement, la segmentation et la caractérisation. Ces phases sont schématisées sur la figure suivante:

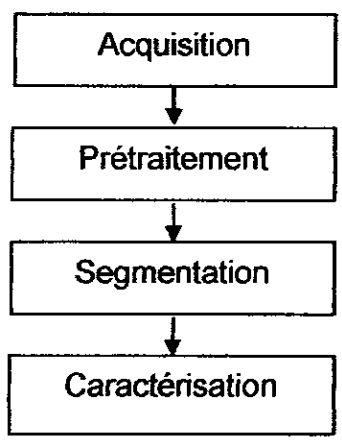


Figure.4.8 : Etapes de traitement de l'image.

4.4.1.1 Acquisition

Les prises de vue sont acquises par la caméra CCD installée sur la pince du robot, puis elles sont transmises vers le PC hôte où elles seront numérisées et stockées en mémoire tampon du bus PCI. Notre application prend en charge la lecture de ces images à partir du bus PCI.

4.4.1.2 Prétraitement de l'image

Dans le processus d'acquisition d'image, des dégradations peuvent apparaître (bruit, problèmes lors de la transmission de l'image, éclairage, etc...). Le rôle du prétraitement dans sa définition la plus générale est de remédier aux dégradations ayant affecté l'image et/ou de rendre cette image mieux adaptée à une application particulière [53].

4.4.1.3 Filtrage de l'image

Le filtrage est une opération qui consiste à éliminer ou réduire le bruit contenu dans une image. Ce bruit est dû à plusieurs sources: la chaîne d'acquisition, la chaîne de transmission ou à la scène elle-même. Dans une opération de filtrage, le pixel est considéré comme un individu (statistique) et on cherche son identité grâce à son voisinage. Le bruit est considéré comme issu d'un signal de haute fréquence. Pour supprimer les hautes fréquences, on utilise un filtre passe-bas, ce filtre ne laisse passer que les basses fréquences. Dans notre cas, et pour éliminer les incidences du bruit contenu dans l'image, on a appliqué deux types de filtres : le premier est linéaire: filtre moyen, et le deuxième un filtre non linéaire: filtre médian, qui nous ont donné de bons résultats.

4.4.1.4 Le Filtrage linéaire

Le filtrage linéaire appliqué est le filtre moyen qui correspond à une combinaison linéaire des pixels voisins. On utilise un noyau de convolution (ou masque) utilisant la moyenne non pondérée des pixels voisins.

$i(x-1,y-1)$	$i(x,y-1)$	$i(x+1,y-1)$
$i(x-1,y)$	$i(x,y)$	$i(x+1,y)$
$i(x-1,y+1)$	$i(x,y+1)$	$i(x+1,y+1)$

Tableau 4.1 : Représentation Matricielle d'un pixel image et son voisinage.

L'algorithme de calcul est le suivant :

Pour tous les pixels de l'image

I(x,y) = [I(x-1,y-1) + I(x,y-1) + I(x+1,y-1) + I(x-1,y) + I(x,y) + I(x+1,y) + I(x-1,y+1) + I(x,y+1) + I(x+1,y+1)] / 9.

Fin de pour

4.4.1.5 Le Filtrage non linéaire

Le deuxième filtre est de type non linéaire appelé filtre médian. On peut résumer son principe de la façon suivante :

Pour tous les pixels de l'image :

1. On range les pixels voisins et le pixel courant dans un tableau en ordre croissant.

2. Dans la nouvelle image, on remplace le pixel courant par la valeur située au milieu du tableau.

Fin de pour

A titre d'exemple de résultats obtenus, on prend comme image de départ l'image suivante :

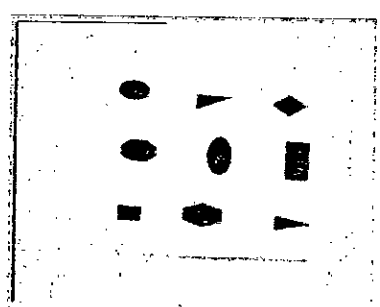


Figure.4.9 : Image originale.

Après filtrage, le résultat est une autre image à faible bruit. Le résultat de cette opération peut ne pas être remarqué à l'œil nu mais va être sûrement bien visualisée dans la suite du traitement :

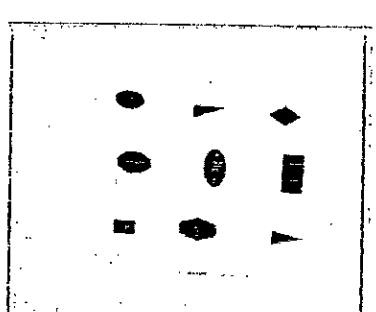


Figure.4.10 : Image filtrée

4.4.1.6 Binarisation de l'image

Après l'élimination du bruit, nous passons à une étape dite de Binarisation qui consiste à transformer l'image de niveau de gris en une autre à deux niveaux, 0 pour le fond et 1 pour les objets.

L'algorithme de calcul utilisé dans cette étape est le suivant :

```
Pour tous les pixels de l'image
    Si la valeur du pixels[x][y] >= seuil
                pixels[x][y] = 1.
    sinon   pixels[x][y] = 0.
Fin     de
pour
```

Les résultats obtenus après cette opération sont illustrés par :

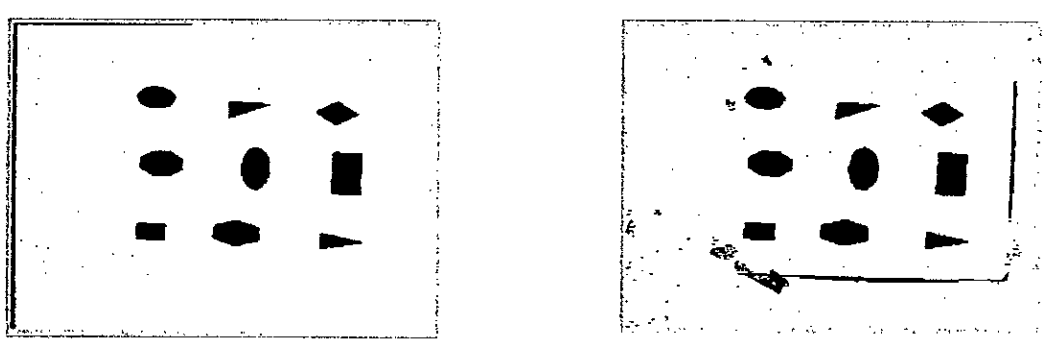


Figure.4.11 : Résultat de Binarisation, à droite image non filtrée, à gauche image filtrée.

Une fois le bruit éliminé et l'image restaurée, afin de compenser les déformations introduites par le milieu de transmission et l'optique d'acquisition, l'image est prête à être analysée. A ce niveau nous passons à l'étape de segmentation qui va permettre de réaliser une partition de l'image en ensembles connexes homogènes afin d'extraire tous les objets contenus dans la prise de vue.

4.4.1.7 Segmentation de l'image

Cette étape consiste à extraire les différentes formes correspondant aux objets qui constituent la scène.

Dans notre travail nous avons utilisé une méthode de segmentation basée sur la croissance de régions pour séparer les objets du fond de l'image et les objets entre eux. L'algorithme de segmentation utilisé est celui d'étiquetage des composantes connexes. Nous avons choisi une telle approche pour réduire le temps de calcul et répondre aux besoins de traitement en temps réel.

La notion de voisinage permet de définir la connexité (4 ou 8 connexités). Une composante 4-connexe (respectivement 8) est telle qu'entre tout couple de pixels de la composante, il existe un chemin 4-connexe (respectivement 8).

a : 4 composantes connexes b : 8 composantes connexes

Figure.4.12 : Voisinage d'un pixel.

L'étiquetage en composantes connexes d'une image attribue à tous les pixels d'une composante connexe une même étiquette (valeur entière). Chaque composante connexe est alors identifiée par son étiquette. Les algorithmes d'étiquetage en composantes connexes détectent les adjacences entre pixels et définissent l'étiquette du point courant en fonction de celles des points voisins.

Cette méthode de segmentation repose sur un balayage séquentiel de l'image. On considère pour chaque point P ses voisins déjà traités. Le déroulement de cette étape est comme suit:

1. Pour tous les pixels image : si un pixel n'appartient pas au fonts, on lui affecte une étiquette suivant la procédure suivante : si les pixels voisins ne sont pas étiquetés, on lui donne une nouvelle étiquette, sinon il prend l'étiquette de ses voisins.
2. L'image binaire obtenue à l'étape précédente va être balayée à la recherche des zones connexes de l'image. En effet à chaque zone connexe correspond une forme d'un objet (sauf si deux objets sont empilés ou accolés, dans ce cas l'algorithme ne verra qu'un objet là où il y a plusieurs). Une forme est déterminée par la liste des pixels qui la composent. Dès la détection d'un point ayant la valeur FOND comme niveau de gris, on utilisera un algorithme analogue aux méthodes

de remplissage des logiciels de dessin pour extraire tous les pixels appartenant à la même forme connexe que lui.

L'algorithme suivant se charge de construire à partir de l'image binaire une liste de forme, chaque forme étant elle-même une liste de pixels [54].

- ***Soient :***
  - *Lformes, la liste des formes extraites.*
  - *Lcandidats, la liste des pixels candidats pour appartenir à la forme en cours d'extraction.*
  - *Lacceptées, la liste des pixels acceptés comme appartenant à la forme en cours d'extraction.*
- *Initialisation des listes :*

Lformes := vide, Lcandidats := vide, Lacceptées := vide.

- ***Pour tout pixel p de l'image binaire faire :***
  - ***Si p != FOND et p != VU Alors :***
    - *Lcandidats := Lcandidats + p.*
    - *p := VU*
    - ***Tant que Lcandidats n'est pas vide faire :***
      - *extraire pixel p' de Lcandidats.*
      - *Lacceptées := Lacceptés + p'.*
      - ***Pour tout voisin p" de p' faire :***
        - ***Si p" != FOND et p" != VU alors :***
          - *Lcandidats := Lcandidats + p".*
          - *p" := VU.*
  - *Lformes := Lformes + Lacceptées*
  - *Lacceptées := vide*
- ***Fin***

L'implémentation de cette approche a démontré sa puissance, elle nous a donné de bons résultats, la procédure répond bien aux besoins de l'application temps réel.

Pour soigner l'affichage, nous avons attribué pour chaque étiquette une couleur, ainsi lors de l'affichage chaque forme apparaîtra avec une couleur donnée. Voici un exemple de résultat :

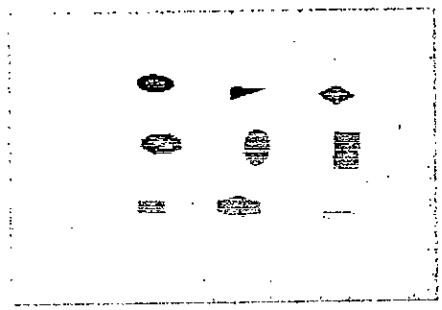


Figure.4.13: Segmentation de l'image.

4.4.1.8 Caractérisation

Cette phase correspond à l'analyse des résultats issus de l'étape précédente. Après la segmentation de l'image et l'extraction de tous les objets se trouvant dans la prise de vue, on s'intéresse à la détermination des coordonnées de chaque forme pour pouvoir manipuler ces objets. Pour se faire, on compte le nombre de pixels pour chaque objet et calcule ses coordonnées du centre de gravité et sa surface en utilisant les relations suivantes [54]:

$$X_G = \frac{1}{s} \cdot \sum P_x \tag{4.1}$$

$$Y_G = \frac{1}{s} \cdot \sum P_y \tag{4.2}$$

$$s = \sum P \tag{4.3}$$

Où : **S**: est la surface de l'objet = la somme de tous les pixels de l'objet.

$\mathbf{X_G}$ et $\mathbf{Y_G}$ sont les coordonnées du centre de gravité de l'objet.

La liste des objets obtenus doit subir un traitement supplémentaire afin d'éliminer les objets qui ne répondent pas aux conditions de sélection. Nous avons fixé pour la surface un seuil, les objets qui ont une surface inférieure au seuil fixé seront éliminés systématiquement. L'illustration suivante nous montre les résultats de cette étape.

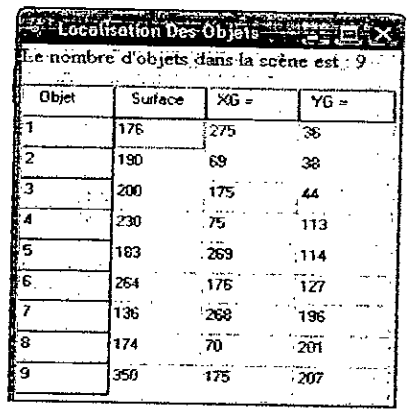


Figure.4.14 : Résultats de la caractérisation.

Après avoir extrait les objets dans l'image acquise et définit leurs paramètres (coordonnées du centre de gravité, surface), on a besoin de connaître les coordonnées réelles de ces objets dans un repère prédéfinie, donc il est primordial de transformer les cordonnées images en coordonnées réelles, le repère le plus utilisé est celui associé au robot manipulateur. Ainsi, il nous faut une transformation de repère pour passer des coordonnées image 2D aux coordonnées espace 3D pour le même objet. Pour trouver les coordonnées 3D réelles à partir des coordonnées 2D des points image, il faut passer par une modélisation de la caméra et une transformation de repère. Cette transformation est déjà expliquée dans le chapitre III.

4.4.2 Calibration de la caméra

Comme le montre la figure ci-après, l'application développée permet aussi une calibration de la caméra. Elle nous permet de capter l'image voulue et de sélectionner les points image manuellement en toute liberté.

Les valeurs saisies peuvent être enregistrées dans un fichier texte par l'utilisateur pour l'utiliser plus tard ou bien pour l'analyser.

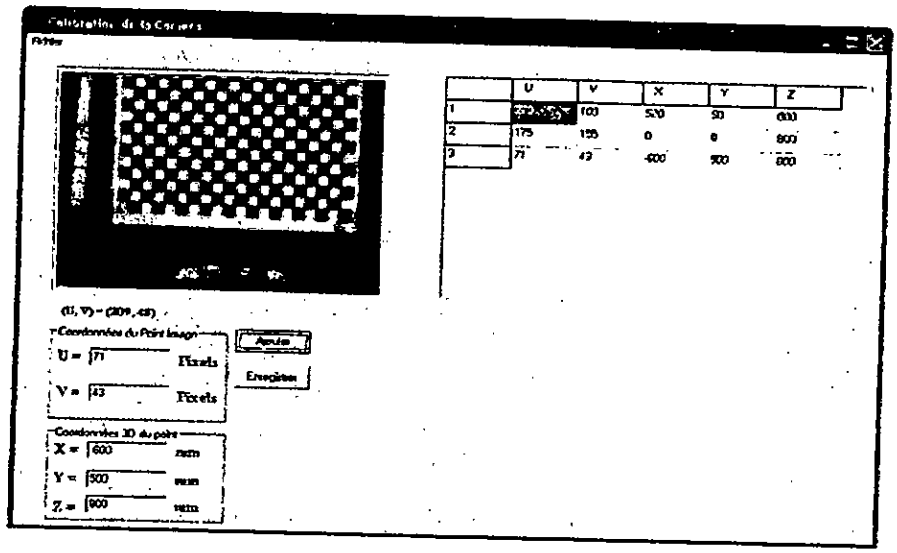


Figure.4.15: Calibration de la caméra.

4.4.3 Fusion multi sensorielle

Quand l'application client reçoit les données du robot, elle les enregistre dans un fichier texte, pour donner la main à une autre application pour lire et exploiter ces résultats. Le traitement et l'analyse des données issues des capteurs sont faits comme suit :

4.4.3.1 Représentation des ultrasons

La représentation des distances issues de la ceinture à ultrasons par des cônes où les sommets sont dirigés vers les points d'émission est une idée provenant de la forme réelle des ondes sonores émises par les télémètres à ultrasons (voir APPENDICE B).

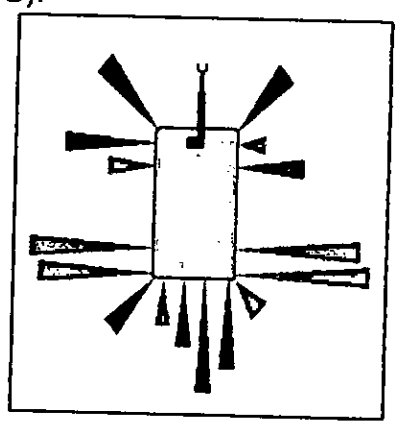


Figure.4.16 : Représentation graphique des données des Ultrasons.

4.4.3.2 Représentation des données Laser

Le LMS peut balayer l'environnement devant le robot avec une résolution qui peut atteindre 0.25°, cela veut dire qu'il nous donne 724 mesures. Pour notre cas, nous pouvons nous contenter d'une résolution de 1°, et l'espace de balayage est divisé en 18 zones. On prend pour chaque zone la valeur minimale de 10 mesures prises. Pour faciliter la compréhension et l'interprétation des résultats, on a la représentation graphique suivante:

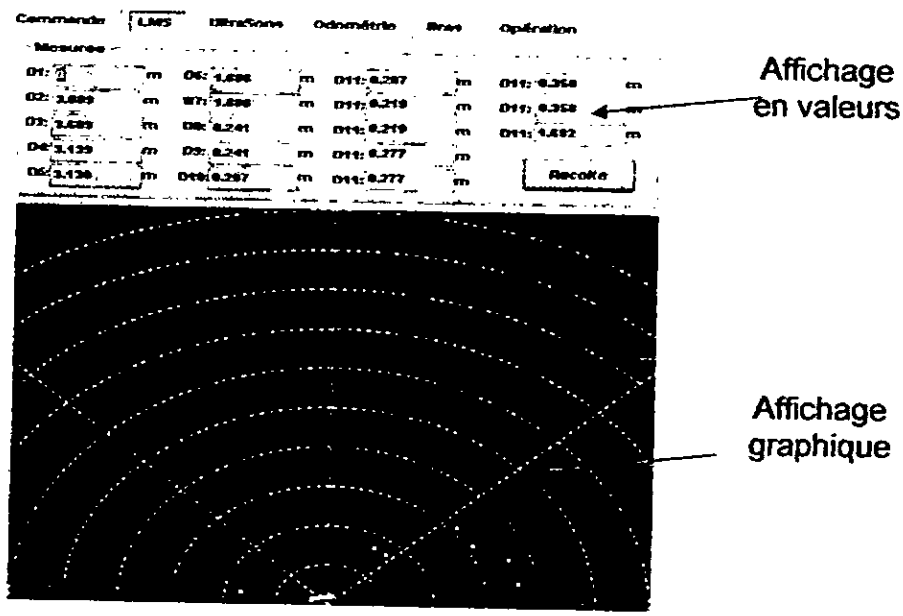


Figure.4.17 : Représentation graphique des données de LMS.

Les autres informations sont affichées sur l'interface IHM en valeur numérique telles quelles sont sans représentation graphique.

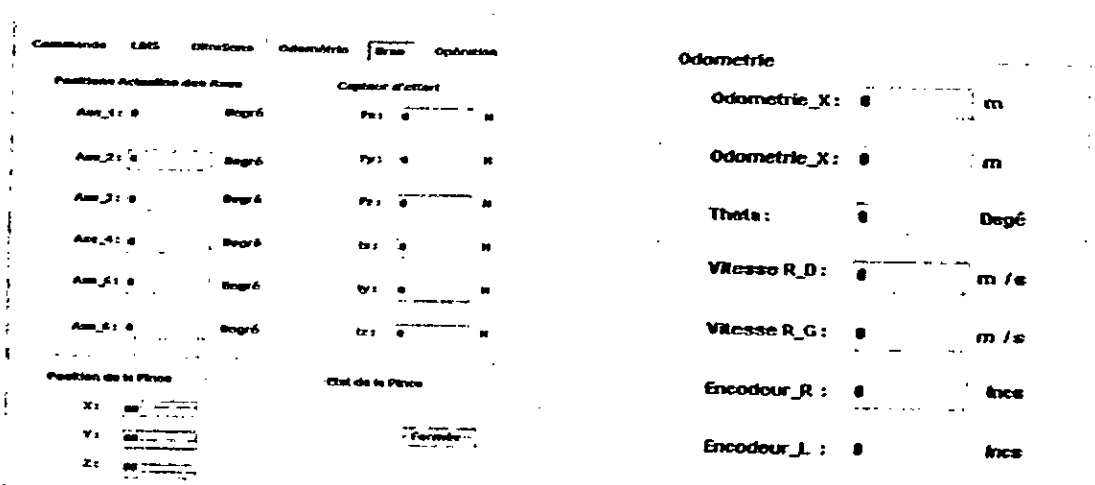


Figure.4.18 : Affichage des données du bras et de la plate forme mobile

4.4.3.3 La fusion entre ces données

Pour l'exploitation des différentes informations sensorielles, le raisonnement adopté est comme suit :

1. Le LMS nous donne une information concernant la distance séparant le robot des objets. La caméra permet l'acquisition de l'image et le système de traitement d'image définit et localise la (ou les) cible(s) à manipuler, tandis que les informations sur l'état interne du robot nous renseigne sur la position du robot et la configuration du bras.
2. Le MGI nous permet d'obtenir la configuration correspondant à la tâche désirée.
3. Elaboration d'une stratégie de mission.
4. Le simulateur nous permet de valider la mission à exécuter.
5. Si la tâche est valide et envisageable, les commandes sont envoyées au robot et on procède à la validation sur site réel.

4.5 Stratégie de téléopération utilisée

Les étapes 1, 2, 3 et 4 de la section précédente nous permettent de définir une stratégie d'exécution d'une tâche. Généralement nous avons deux situations :

a. La cible se trouve dans le champ du travail du bras : cette tâche ne nécessite que le mouvement du bras, car le bras peut l'atteindre sans déplacement de la plate forme mobile

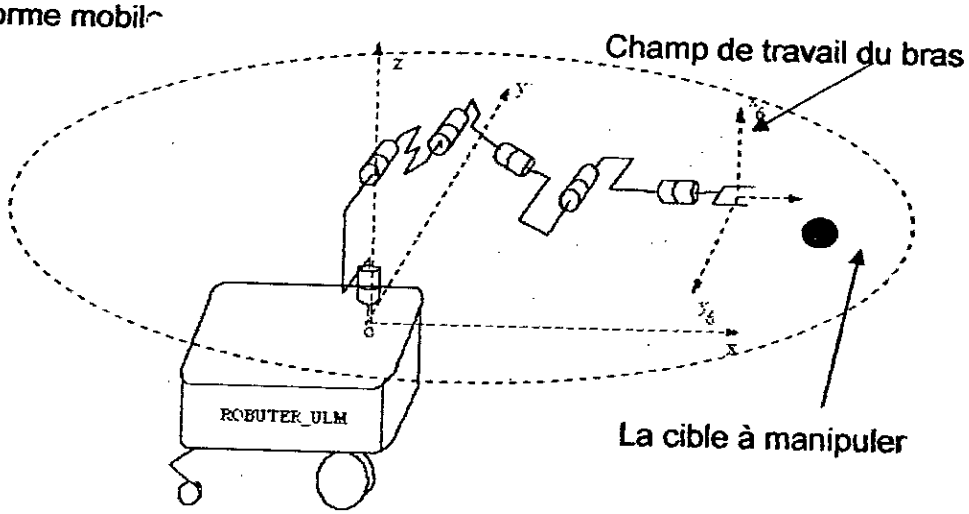


Figure.4.19 : La cible se trouve dans le champ de travail du robot.

b. La cible se trouvant en dehors la zone atteignable par le bras. Dans ce cas on procède de la manière suivante :

- On déplace la base mobile jusqu'à ce que la cible soit dans le champ du travail du bras.
- Ensuite on manipule la cible à l'aide du bras.

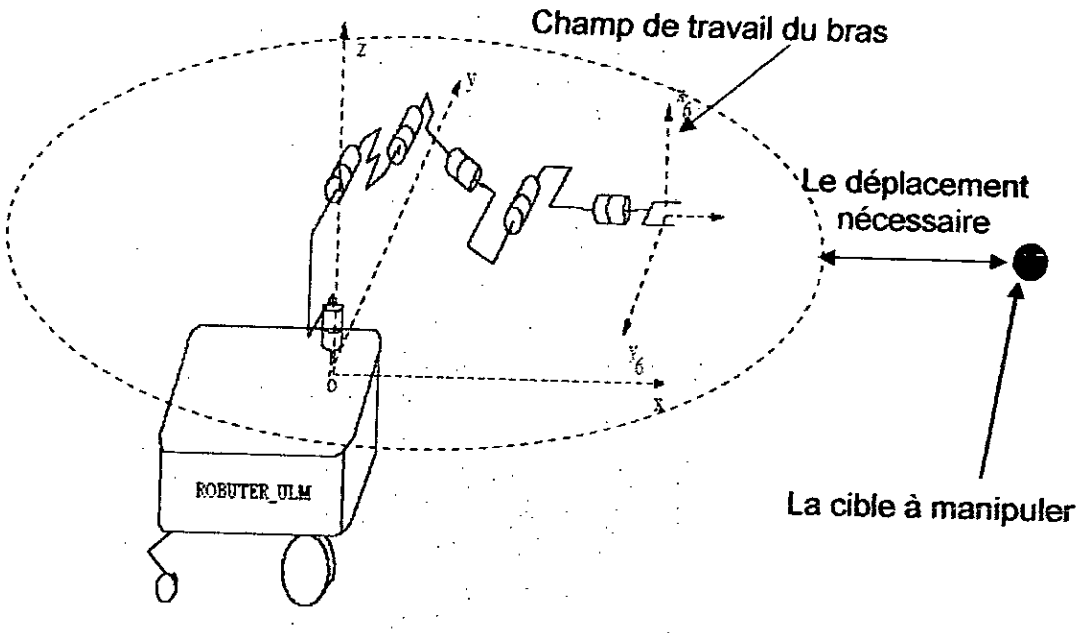


Figure.4.20 : La cible se trouve hors le champ de travail du robot.

4.6 Simulation des missions

La simulation graphique est une technique qui sert à visualiser le phénomène à étudier sur ordinateur avant de le réaliser expérimentalement.

En robotique, l'utilisation des simulateurs reste toujours l'outil de validation et d'expérimentation le plus répandu. On trouve toujours un prototype graphique représentant le système mécanique du robot. Il est virtuel, mais il a la même architecture physique et mécanique et peut se mouvoir et reproduire les mouvements du robot réel, avec un même nombre de degré de liberté, un même aspect et une reconstruction graphique 3D de la scène réelle (l'environnement où se trouve le robot). Les scènes sont des espaces 3D virtuels sur ordinateurs représentant les scènes réelles, elles ont les mêmes caractéristiques géographiques et la même structure physique.

La simulation informatique offre l'avantage de pouvoir attaquer la résolution d'un problème avant même d'avoir construit le mécanisme qui fait l'objet de l'étude, et même avant de fixer tous les choix technologiques. Cela doit permettre, au moment de la réalisation concrète, d'obtenir rapidement un robot fonctionnel, et surtout de ne pas faire courir de risque grave à un matériel coûteux. Beaucoup de simulateurs sont développés au cours du temps dans le monde, ces simulateurs sont efficaces mais présentent quelques inconvénients:

- Ces simulateurs sont réalisés par des outils de développements graphiques 3D, donc ce sont généralement des produits finis et ne supportent aucune modification.
- Ces simulateurs ne sont pas universels et sont à utilisation restreinte parce que les systèmes robotiques objets de la simulation sont des systèmes bien spécifiques et ne sont pas identiques, c'est pourquoi, les simulateurs sont personnalisés. Par exemple, un simulateur d'un robot de manutention pour une chaîne de production ne peut pas être utilisé pour la simulation d'un robot chirurgien par exemple.
- Chaque simulateur est une représentation graphique d'un système réel, et comme chaque système robotique a une tâche bien définie, le simulateur sera donc spécialisé.

4.6.1 Présentation de notre simulateur

Nous avons réalisé un simulateur spécifique du robot mobile manipulateur Robuter_ULM évoluant dans un environnement structuré pour des missions de type téléopéré. Il est utilisé pour valider et visualiser les mouvements du robot au sein d'un environnement crée dans le but de vérifier les erreurs et les anomalies et de voir les configurations possibles pour les tâches à envisager. Ce produit est appelé SimRobuter, qui signifie Simulateur du robot manipulateur mobile ROBUTER-ULM.

Ce simulateur est développé en pascal sous l'environnement de développement intégré (EDI) Delphi.5, associé à l'outil GLScene, qui est un ensemble de composants graphique pour Delphi permettant la création des graphismes et des animations 3D (le paquetage d'installation de GLScene se trouve à l'adresse électronique en [55]).

GLScene est le paquet de composants VCL (Visual Componant Library) orienté objet de la bibliothèque graphique ouverte OpenGL (Open Graphical Library), cette bibliothèque est très puissante pour la programmation graphique et l'animation 3D, elle offre pas mal d'outils de développement, et présente l'avantage d'être multi plateforme, elle supporte plusieurs environnements de développement Delphi [56], Builder C++ [57] et Kylix [58], [59]. Ainsi, un projet réalisé sous le système d'exploitation Windows sera compilé et exécuté sans aucun problème sous le système Linux d'où l'intérêt de cet outil.

L'utilisation de l'EDI Delphi n'était pas un choix, mais c'était une obligation, parce que la version libre de GLSene, disponible sur Internet, ne supporte que Delphi. Après avoir téléchargé et installé le composant GLSene, une barre de composant nommée GlScene sera ajoutée aux paquets de composant dans Delphi, qui a l'aspect suivant:

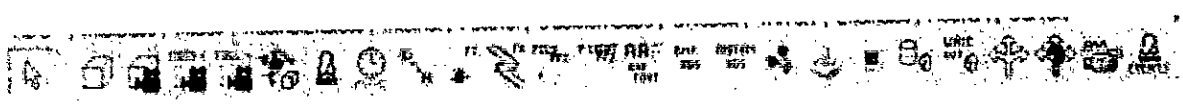


Figure.4.21: La barre de composant GLSene après son installation.

Comme le montre la figure ci-dessus, cet outil offre la possibilité de faire pas mal de choses, on cite quelques unes:

- la création et l'animation des objets 3D (cubes, sphères, mécanismes complexes).
- la possibilité de filmer cette animation.
- l'intégration des systèmes de commande comme le joystick.

L'exploitation de cet environnement nous a permis de développer l'application suivante:

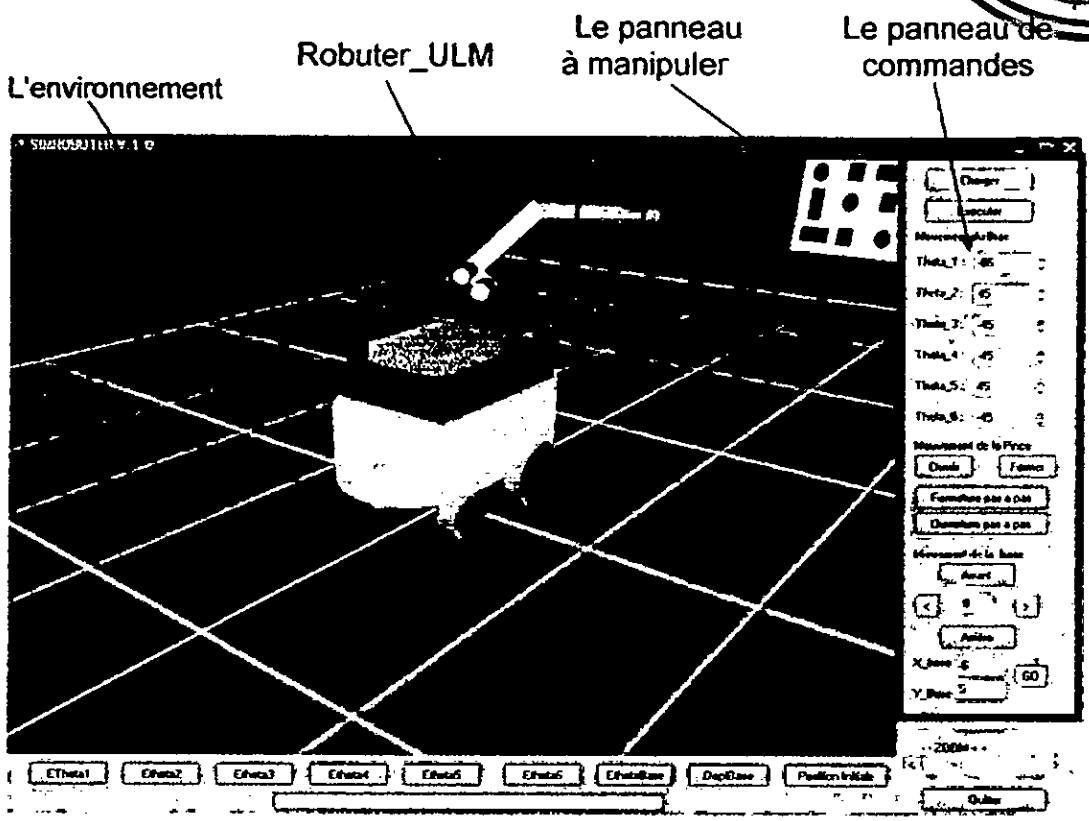


Figure.4.22: L'interface du simulateur SIMROBUTER.

L'application contient :

- un espace de simulation et de visualisation représentant la scène.
- des boutons de commande et des champs pour introduire les consignes manuellement.
- en appuyant sur la scène avec le bouton gauche de la souris et en faisant déplacer la souris, on pivote de la scène selon le mouvement de la souris.
- on peut aussi agir sur le zoom.

4.7 Résultats expérimentaux et validation

Dans cette section, nous allons présenter les différents résultats obtenus, les tests faits et les validations réalisées.

4.7.1 Résultats de calibration de la caméra CCD

Avant de citer les résultats de calibration, nous recommandons au lecteur de lire l'annexe C, qui contient une explication concernant les conditions de calibration.

Après avoir fait les tests et les mesures nécessaires, les matrices intrinsèques et extrinsèques obtenues sont les suivantes :

$$M_{int} = \begin{bmatrix} 7.2178 & 0 & -0.0062 & 0 \\ 0 & 8.0782 & 0.2774 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$M_{ext} = \begin{bmatrix} -0.9639 & 0.1634 & -0.2101 & 14.956 \\ 0.0216 & 0.9896 & -0.1378 & 53.674 \\ 0.0128 & 0.03 & -0.0313 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

La matrice de passage jacobéenne résultat du produit de ces deux matrices est la suivante:

$$M = M_{ext} \bullet M_{int} = \begin{bmatrix} -6.9574 & 1.1795 & -1.5166 & 107.9428 \\ 0.1783 & 8.0028 & -1.1218 & 433.8669 \\ 0.0128 & 0.0300 & -0.0313 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Les différents paramètres intrinsèques et extrinsèques calculés de la caméra après avoir obtenu les matrices M_{int} et M_{ext} (Pour la démonstration voir chapitre 3) sont :

P_x = 0.265mm. largeur du pixel mesurée.
P_y = 0.265mm hauteur du pixel mesurée.
f = 1mm. Distance focale.
u_0 = -0.0062mm.
v_0 = 0.2774mm.

P_x = 0.277mm. largeur du pixel estimée.
P_y = 0.250mm hauteur du pixel estimée.
$\alpha_U = 7.2178 mm^{-1}$.
$\alpha_v = 8.0782 mm^{-1}$.

Le tableau suivant nous donne un exemple des résultats obtenus par l'implémentation de cette matrice (toutes les mesures sont données en mm).
Avec: L'indice c : désigne la valeur calculée estimée en mm.

L'indice r : désigne la valeur réelle mesurée en mm.

Passage 3D → 2D								
X_r	Y_r	Z_r	U_c	V_c	U_r	V_r	ΔU	ΔV
-39	-19.5	300	38.96	25.91	40	26	1.04	0.09
19.5	0	500	205.01	31.89	207	32	1.99	0.11
58.5	-39	600	259.4	110.50	260	111	0.6	0.5

Passage 2D → 3D										
U_r	V_r	X_c	Y_c	Z_c	X_r	Y_r	Z_r	ΔX	ΔY	ΔZ
256	54	38.2	-19.82	399	39	-19.5	400	0.8	0.32	1
202	7	19.42	38.94	699.5	19.5	39	700	0.08	0.06	0.5
129	115	-39.2	-39.07	798.84	-39	-39	800	-0.2	-0.07	1.16

Tableau .4.2: Résultats de la calibration de la caméra.

Ce qui nous permet de calculer les erreurs moyennes suivantes :

E_U = 1.21mm

E_V = 0.23mm

E_X = 0.23mm

E_Y = 0.31mm

E_Z = 089mm

Pour une meilleure illustration de ces résultats, nous allons les représenter sous forme graphique, où les coordonnées image mesurées sont représentées par des croix bleues et les coordonnées calculées par des croix rouges.

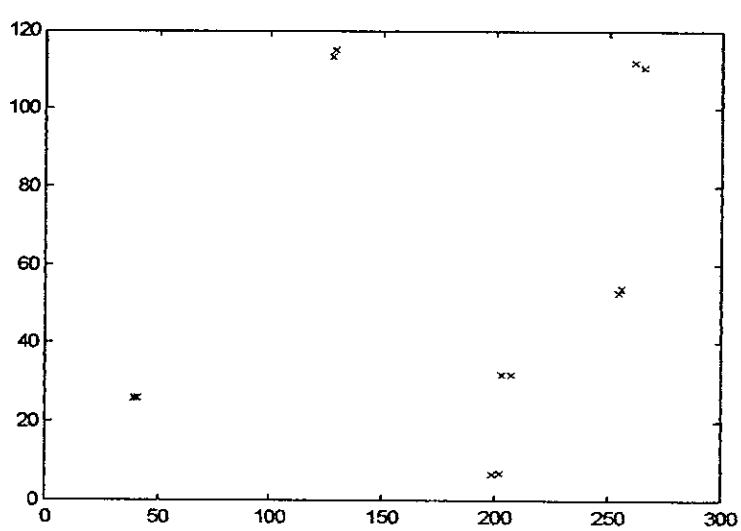


Figure 4.23 : Comparaison entre les coordonnées image mesurées et celles calculées.

Si on calcule les correspondants 3D des points 2D relevés sur l'image, on trouve les résultats suivants :

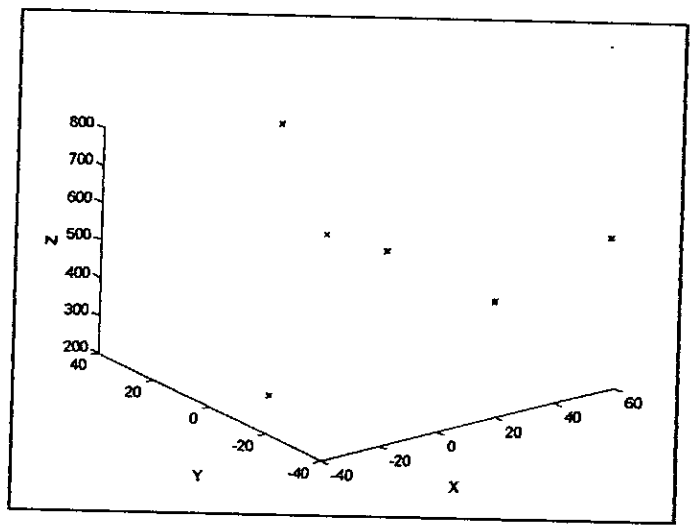


Figure 4.24 : Représentation des coordonnées image mesurées et celles calculées.

On remarque bien qu'en partant des points 2D et en calculant leurs correspondants 3D ou inversement, les valeurs calculées sont proches (parfois identiques) des valeurs réelles mesurées, ce qui est bien illustré par la coïncidence des points. Ceci confirme les bons résultats de calibration.

A noter ici, que les résultats de la calibration de la caméra que nous avons pu obtenir avec une précision de l'ordre du millimètre, sont meilleurs que ceux obtenus dans la littérature par d'autres chercheurs (voir le travail du chercheur Ait Aider et ses collègues reportés dans la référence [60], qui donnent des résultats de calibration avec une précision de l'ordre du centimètre).

4.7.2 Résultas de la simulation

Le simulateur nous offre la possibilité de faire une animation 3D d'un prototype virtuel du robot Robuter-ULM, qui reproduit fidèlement le comportement réel du robot et il peut avoir toutes les configurations possibles que peut avoir le robot réel, le simulateur offre aussi l'avantage de simuler et d'exécuter les gestes et les actions du robot virtuellement sur PC.

Il est conçu dans le but de pouvoir tester les différentes stratégies possibles dans les conditions réelles d'utilisation. Pour atteindre une position désirée on a

généralement plusieurs configurations possibles, lesquelles peuvent être testées et simulées comme dans le cas réel.

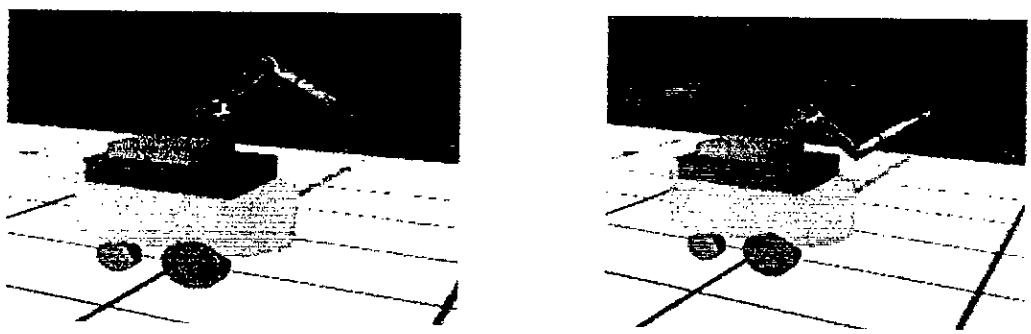


Figure.4.25: Les configurations possibles pour une position donnée.

Ou bien pour la simulation d'une mission, par exemple la mission d'atteinte d'une cible qui se trouve devant le robot, comme ci-dessous.

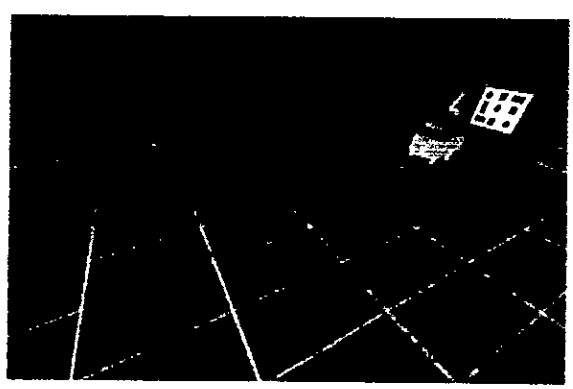


Figure.4.26: Simulation d'une mission.

Cet outil a été conçu en respectant les contraintes mécaniques du robot et les contraintes physiques de l'environnement.

Si une situation est impossible pour le robot, ou présente un certain danger, on annule la validation réelle. Dans la figure suivante apparaît une collision entre le bras et la plate forme mobile suite à une configuration non valide.

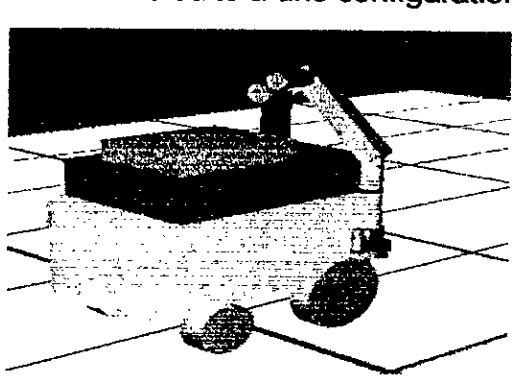


Figure.4.27: Collision entre le bras et la plate forme mobile

Le simulateur supporte aussi le zoom sur la scène, dans le cas où l'opérateur veut faire un zoom pour éclaircir la perception et se rapprocher du robot. Il permet aussi de changer le point de prise de vue de la caméra. Donc, on peut se positionner en n'importe quel point de l'environnement et percevoir n'importe quel aspect de la scène. La figure ci-dessous illustre une vue d'en haut:

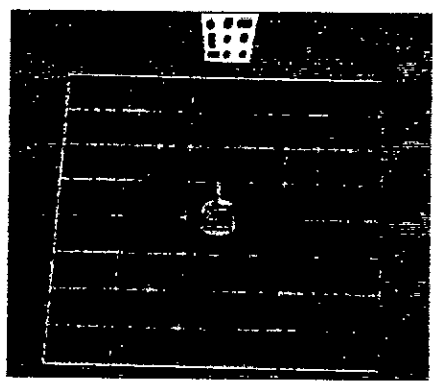


Figure.4.28. Opération du Zoom et pivote sur la scène

Une fois que la stratégie est vérifiée par simulation, il est possible de faire la validation sur site réel, il ne nous reste qu'à se connecter au serveur pour lui envoyer les consignes nécessaires à l'exécution.

4.7.3 Résultas de la téléopération (validation réelle)

Les tests de téléopération sont réalisés dans le laboratoire, où le robot a comme mission d'atteindre et de toucher un (ou plusieurs) objet (s) de type plan situé sur un plan vertical à une distance quelconque. La première des choses à vérifier au début est la communication entre le robot et sa station de contrôle, on opère ensuite la calibration du bras manipulateur, c'est-à-dire le positionner dans une configuration initiale.

A noter ici qu'il faut vérifier et s'assurer que le plan caméra soit en parallèle avec le plan des objets pour une meilleure localisation, comme il est illustré ci-dessous.

Figure 4.29 : vérification d'alignement de la caméra.

4.7.3.1 Mission à un seul objet cible

Par la suite, on montre un exemple de test de validation, fait sur site réel, où la tâche à faire est une mission d'atteinte un objet cible se trouvant en face du robot. La photo suivante nous montre un exemple, la cible à manipuler est celle désignée par l'opérateur et encerclée par rouge.

Figure 4.30 : l'opérateur nous montre la cible.

La station de contrôle reçoit les informations concernant la distance séparant le robot des objets et la prise de vue des objets, tandis que l'algorithme de fusion multi sensorielle calcule les commandes convenables.

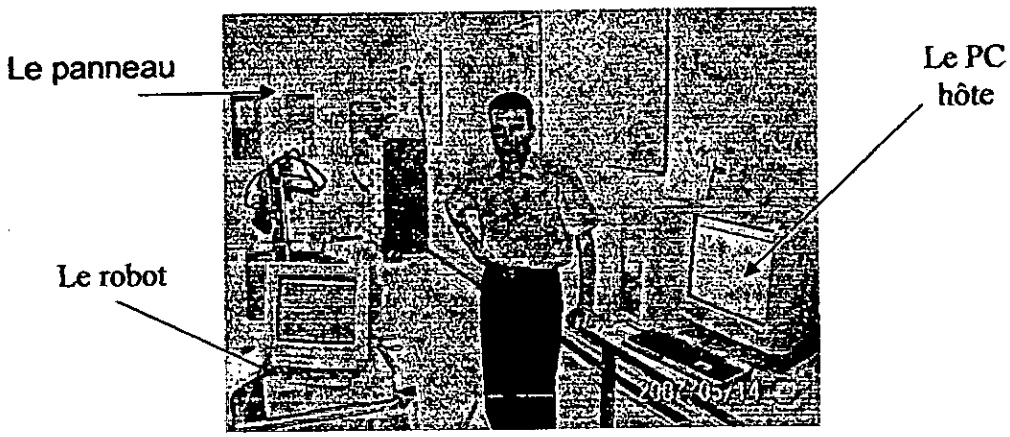


Figure.4.31: Représentation du système de téléopération.

Une fois la satisfaction des conditions de manipulation, on valide la mission et la plate forme mobile (Robuter) se met en mouvement en premier temps en exécutant une tâche de déplacement pour se rapprocher de la cible pour s'arrêter à une distance de 30cm de façon que les objets cibles se trouvent dans le champ de travail du bras manipulateur.

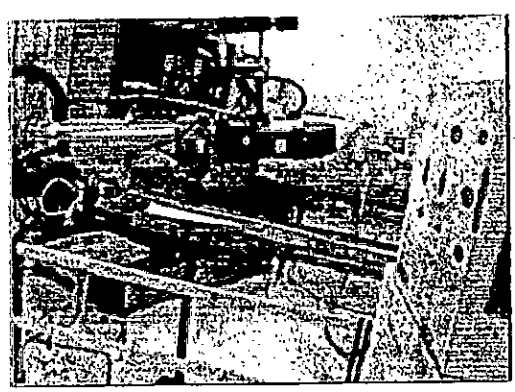


Figure.4.32: Le robot rapproche de la cible.

Le bras rentre en action pour atteindre la cible.

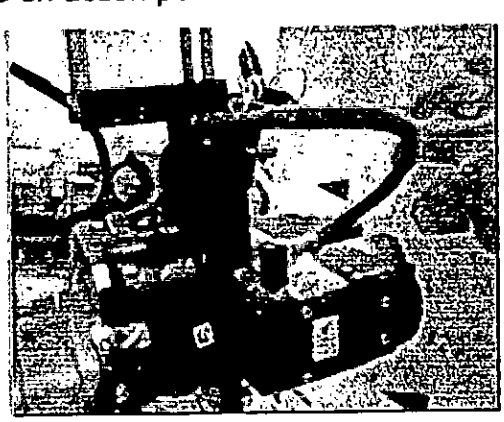


Figure.4.33 : Début de mouvement du bras.

Finalement, on voit que le robot a bien effectué la mission qui lui a été assignée.

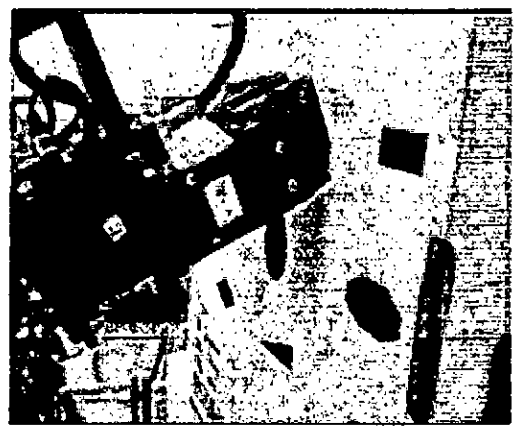


Figure.4.34 : Le robot atteint la cible.

En fin d'exécution, le bras revient à sa configuration initiale qui représente la configuration de départ pour chaque mission.

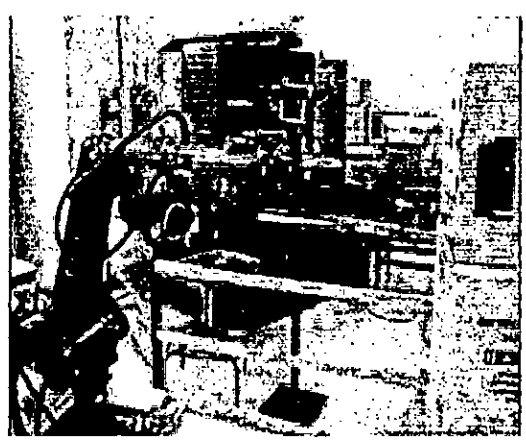


Figure.4.35 : Le bras reprend sa configuration de départ.

Le robot se met en statique pour signaler la fin de la mission et reste dans l'attente d'une nouvelle mission.

4.7.3.2 Mission à plusieurs objets cibles

Le système de téléintervention développé permet de réaliser n'importe quelle séquence selon un ordre choisi par l'opérateur sur les objets cibles. Pour illustrer cette caractéristique, nous envisageons un exemple d'une mission d'atteinte d'un ensemble de quatre points sur un panneau cible. Le robot va agir séquentiellement selon un ordre établi par l'opérateur sur un ensemble de quatre objets.

Initialement le robot est situé loin du panneau (figure ci-dessous) :

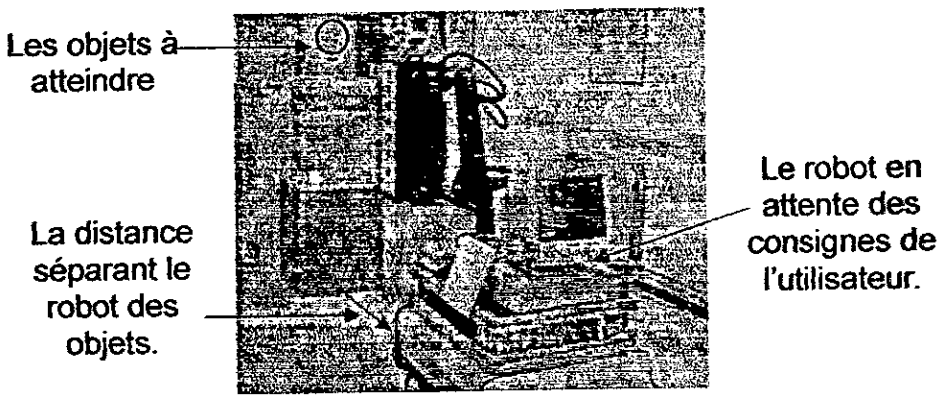


Fig.4.36. Position du départ du robot vis-à-vis du panneau.

Les données des capteurs sont disponibles au niveau du client (PC Hôte). Le serveur (PC embarqué) est en attente des ordres d'action. L'interface IHM sur le PC client permet à l'opérateur la perception, l'acquisition, la sélection et la définition de la mission. Une fois que la préparation est terminée, le système envoie les ordres d'action au PC embarqué pour la réalisation de la mission.

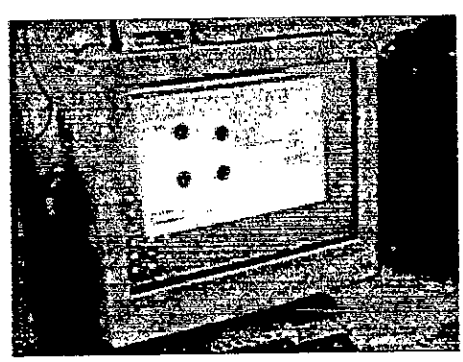


Fig.4.37. Traitement des données sur le PC hôte.

Dans cette mission, le robot doit agir séquentiellement sur les quatre points, comme illustré sur la figure suivante.

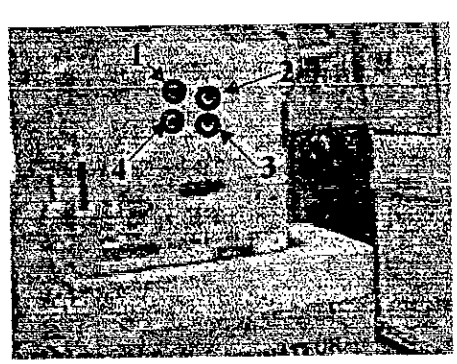


Fig.4.38. La séquence à atteindre par le robot.

Une fois les consignes reçues et traitées par le programme implanté sur le PC embarqué, le robot commence son action. Initialement le robot se trouve loin du panneau de 2.69m, il va se rapprocher de la cible, et s'arrête à une distance de 30 cm, permettant au bras de rentrer en action.

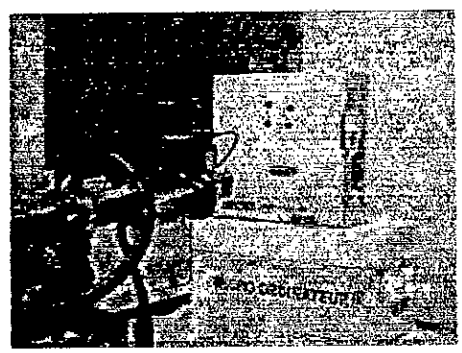


Fig.4.39. Déplacement du robot près de la cible.

L'action est dévolue maintenant au bras qui va se déplacer pour atteindre les objets selon la sélection réalisée par l'opérateur. Après chaque action, il prend une configuration intermédiaire, et à la fin de la séquence il prend sa configuration de départ. Voici le résultat de l'exécution de cette mission.

1. Atteinte du premier point :

Fig.4.40. Le premier point.

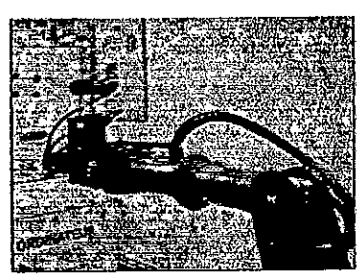


Fig.4.41. Position intermédiaire

2. Atteinte du deuxième point :

Fig.4.42. Le deuxième point.

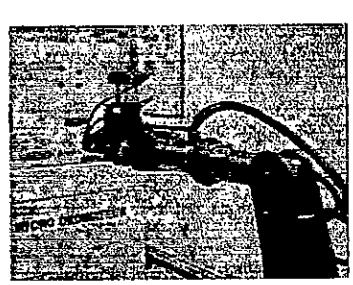


Fig.4.43. Position intermédiaire

3. Atteinte du troisième point :

Fig.4.44. Le troisième point.

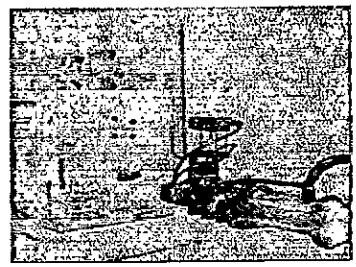


Fig.4.45. Position intermédiaire

4. Enfin atteinte du dernier point :

Fig.4.46. Le quatrième point.

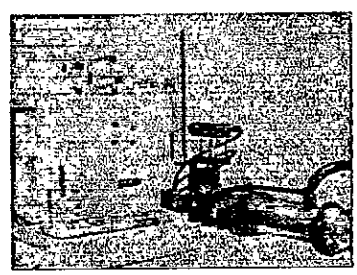


Fig.4.47. Retour à la position initiale.

4.8 Conclusion

Dans cette section, nous avons présenté les applications développées. Nous avons discuté de la communication établie entre le client et le serveur et le traitement des informations à faire. Les différentes approches utilisées en traitement d'images ont été également expliquées ainsi que les résultats de leur implémentation. Nous avons présenté le simulateur graphique 3D réalisé et nous avons développé largement la stratégie de missions que nous avons adoptée pour l'exécution des tâches.

Nous avons présentés aussi dans ce chapitre les résultats obtenus et les tests de validation expérimentale. Cependant, ces résultats sont encourageants et laissent espérer de pouvoir mettre en place d'autres stratégies et de faire des améliorations pour la perfection de ce système de téléintervention.

CONCLUSION

Nous avons abordé dans le cadre de ce travail l'élaboration et la mise en œuvre d'un système de téléopération construit autour du robot mobile manipulateur, Robuter_ULM, permettant de réaliser des tâches d'intervention à distance. La robotique d'intervention est au centre de plusieurs domaines de recherche : la robotique, la simulation graphique des systèmes réels, la vision en robotique, le traitement d'image proprement dit et l'asservissement.

Les principales contributions réalisées de ce travail se situent au niveau des aspects suivants :

1. la modélisation cinématique des robots manipulateurs et mobiles.
2. la transmission à distance, le traitement d'images et la calibration de caméra.
3. le protocole de communication Windows-Linux.
4. le développement de simulateur graphique.
5. la fusion multisensorielle d'informations issues de capteurs US, d'un système laser, d'un capteur d'effort et du système d'odométrie.
6. le système de téléintervention proprement dit.

Ainsi, une part non négligeable a été consacrée à la modélisation géométrique directe et inverse, que ce soit pour le robot Robuter ou le bras UL. Les modèles élaborés nous ont permis de définir la cinématique de l'ensemble résultant composé de ces deux robots hétérogènes à juste titre. Leur utilisation a été envisagée tant dans le cadre de l'élaboration du simulateur graphique que dans la mise œuvre du système temps réel de télérobotique.

La vision robotique a été abordée dans un cas complexe correspondant à l'utilisation d'une caméra embarquée placée sur la pince du robot et l'image issue de cette caméra est traitée sur un poste éloigné, par conséquent les contraintes liées à l'instabilité du bras et du décalage temporel devaient être prises en charge par un traitement adéquat.

La définition d'un protocole de communication à distance entre deux systèmes d'exploitation en l'occurrence Windows et Linux n'était pas une tâche

aisée. La mise en oeuvre du protocole de communication client serveur est sans doute quelque chose de nouveau puisque elle relie deux systèmes différents (du point de vue vitesse de traitement et de communication, système de fichier, codage d'informations,..). La communication réseau établie entre le PC hôte et le PC embarqué robot ROBUTER-ULM est parfaitement fonctionnelle, le flux d'information circule fidèlement entre les deux PC sans aucun problème, ROBUTER-ULM est parfaitement contrôlable à distance à travers le réseau local. Des extensions de la communication ont été opérées en direction de l'utilisation du réseau d'Internet avec des résultats comparables.

Le simulateur développé SIMROBUTER est un banc d'essai virtuel, il suscite pour nous beaucoup d'intérêts, il nos a permis de valider la configuration du robot avant, après et lors l'exécution d'une tâche. Avec l'utilisation de ce simulateur, l'opérateur aura une vision complète de l'action à exécuter par le robot. La conception de ce simulateur était une tâche ardue, elle nous a engendré beaucoup de problèmes complexes et variés que nous avons pu surmonter.

Les résultats de calibration ont été validés directement par des missions exercées par le Robuter_ULM. Ces résultats répondent bien à nous attentes, ils nous permettent de conclure que l'approche envisagée pour la localisation des objets est applicable si l'on s'est fixées bien en terme de précision.

Le système de fusion multisensorielle réalisé donne à l'utilisateur la faculté d'associer une ou plusieurs informations issues des capteurs proprioceptifs et extéroceptifs tant dans les manœuvres de déplacements que dans les manoeuvres d'approches lors de l'exécution d'une mission de téléintervention.

L'ensemble des contributions sus citées ont été intégrées à un systèmes de téléintervention avec une interface Homme-Machine conviviale donnant à l'opérateur la définition de mission d'une façon simple et permettant un suivi d'exécution des plus fiable.

En termes d'extension de ce travail, il est prévu de fusionner le système de téléintervention réalisé avec un autre système de navigation dans des milieux contraints en phase de développement.

APPENDICE A
LISTE DES SYMBOLES ET DES ABREVIATIONS

ARITI : Augmented Reality Interface for Telerobotic applications via Internet.

ARPH : Assistance Robotisée aux Personnes Handicapées

CCD : Charge Coupled Device.

CDTA : Centre de Développement des Technologies Avancées.

CEA : Commissariat à l'Energie Atomique.

EDI : Environnement de Développement Intégré

f : Distance focale de la caméra.

GUI : Graphical User Interface.

HF : Height Frequency.

IHM : Interface Homme Machine.

IP : Internet Protocol.

IRCAD : Institut de Recherche Contre le concert de l'appareil digestive.

LAN : Local Area Network.

LED : Light Emitting Diode.

LMS : Laser Mesurement System.

M : Matrice jacobéenne de la caméra.

M_{ext} : Matrice des paramètres extrinsèques.

M_{int} : Matrice des paramètres intrinsèques.

MGD : Modèle Géométrique Direct.

MGI : Modèle Géométrique Inverse.

OpenGL : Open Graphical Library.

PC : Personal Computer.

PWM : Pulse Width Modelation.

RA : Réalité augmentée.

RAM : Random Access Memory.

RPRV : Remotely Piloted Research Vehicles.

RPV : Remotely Piloted Vehicles.

RTAI : Real Time Application Interface.

RV : Réalité virtuelle.

SynDEx : Synchronous Distributed Executifs.

TAO : Téléopération assistée par ordinateur.

TCP : Transmission Control Protocol.

T_i^j : Matrice de transformation homogène du repère i vers le repère j.

UAV Unmanned Air Vehicles

ULM : Ultra Light Manipulateur.

VAHM : Véhicule Autonome pour Handicapés Moteur

VCL : Visual Componant Library

θ_i : Angle de rotation.

APPENDICE B

B.1 Architecture du robot mobile [21]

La structure et les dimensions du robot mobile sont données par les dessins techniques suivants, où toutes les mesures sont en millimètres:

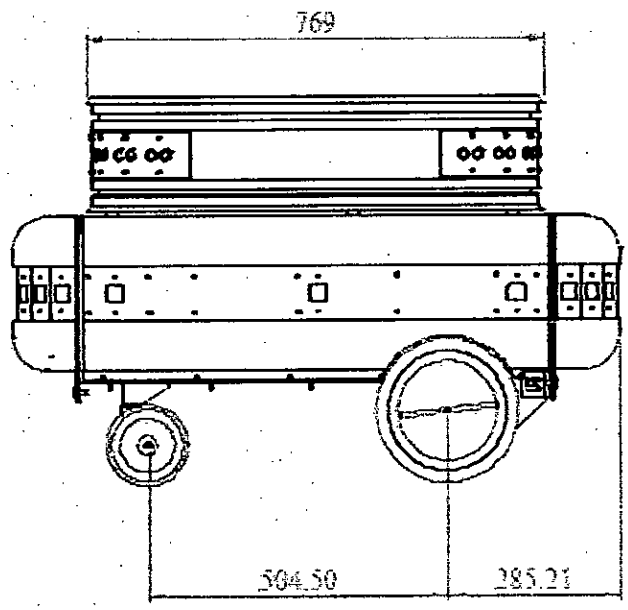


Figure.B.1 : Vue de coté du robot mobile.

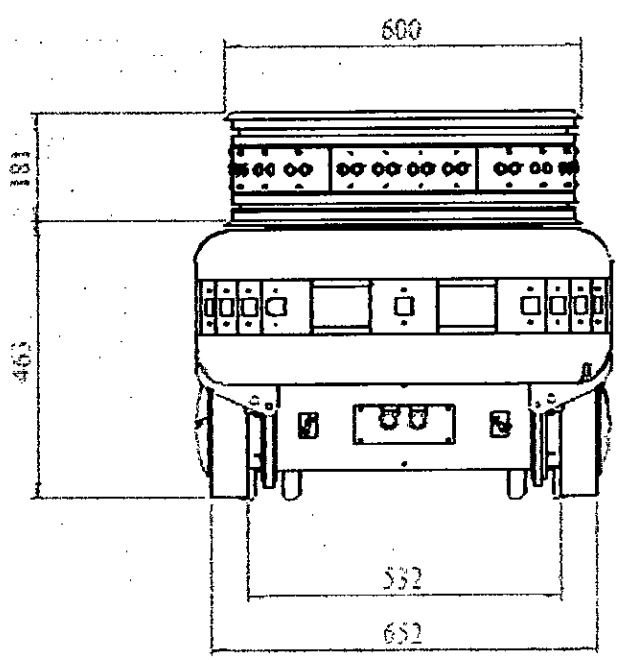


Figure.B.2 : Vue de l'avant du robot mobile.

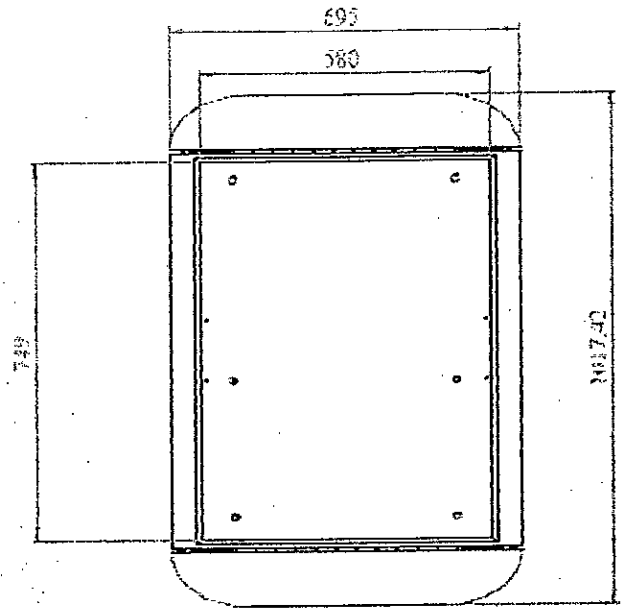


Figure.B.3 : Vue de dessus du robot mobile.

B.1 Architecture du bras manipulateur

Le bras ultra léger est constitué de six ensembles, il est décomposé suivant la figure suivante :

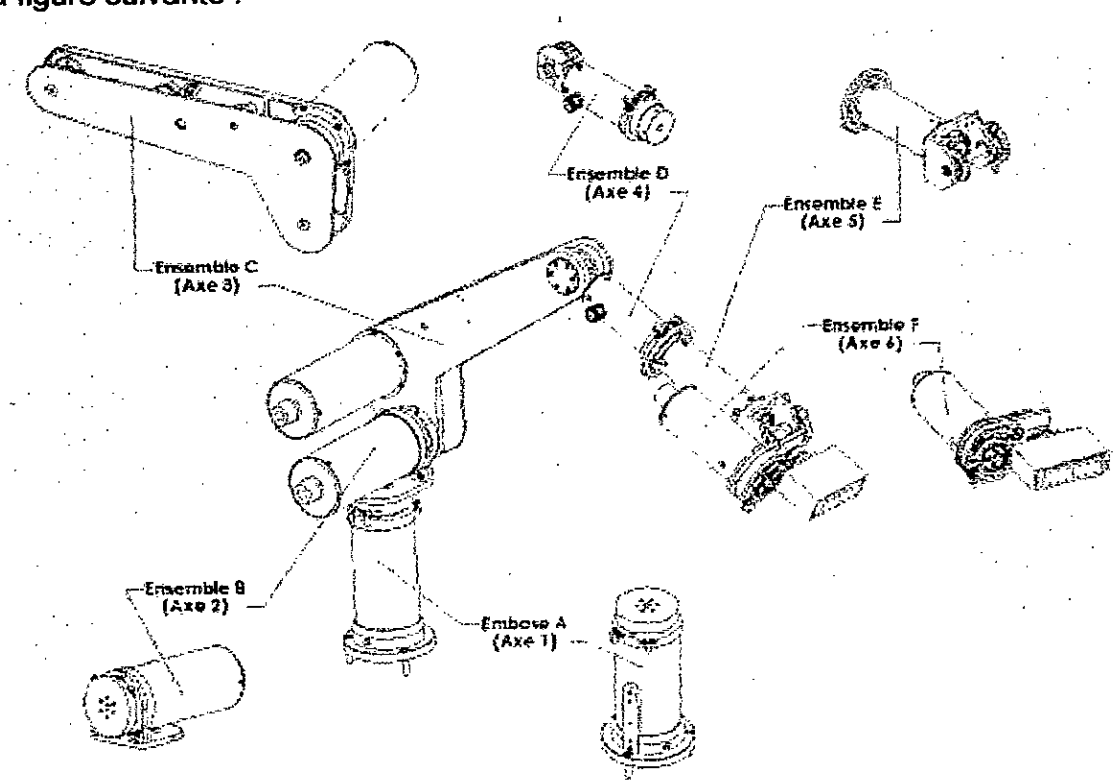


Figure.B.4 : Décomposition du bras manipulateur.

Les dimensions des segments et le débattement maximum que supporte chaque axe du bras ULM sont spécifiés et schématisés comme suite :

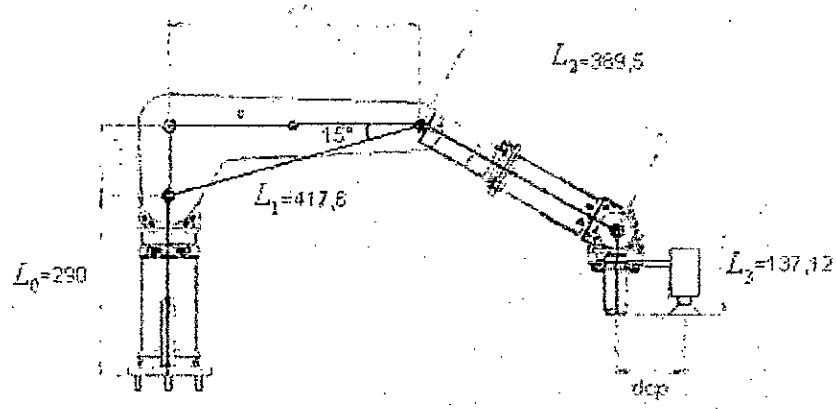


Figure.B.5 : Dimensions des axes.

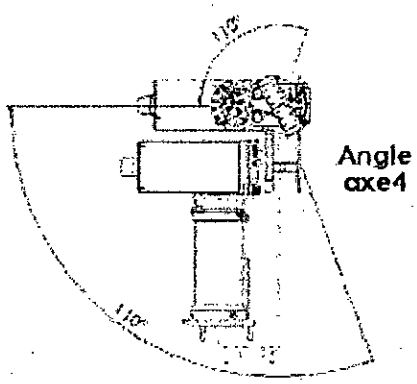


Figure.B.6 : débattements max de l'axe 4.

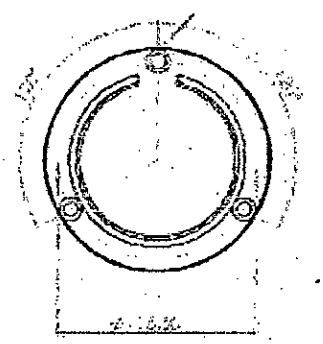


Figure.B.7 : débattements max de l'axe 1.

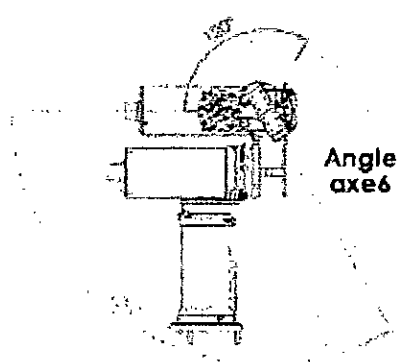


Figure.B.8 : débattements max de l'axe 6.

Le bras dans son évolution occupe un espace, ce dernier est limité par les dimensions et les débattements permis. L'espace de travail du bras est illustré ci-après :

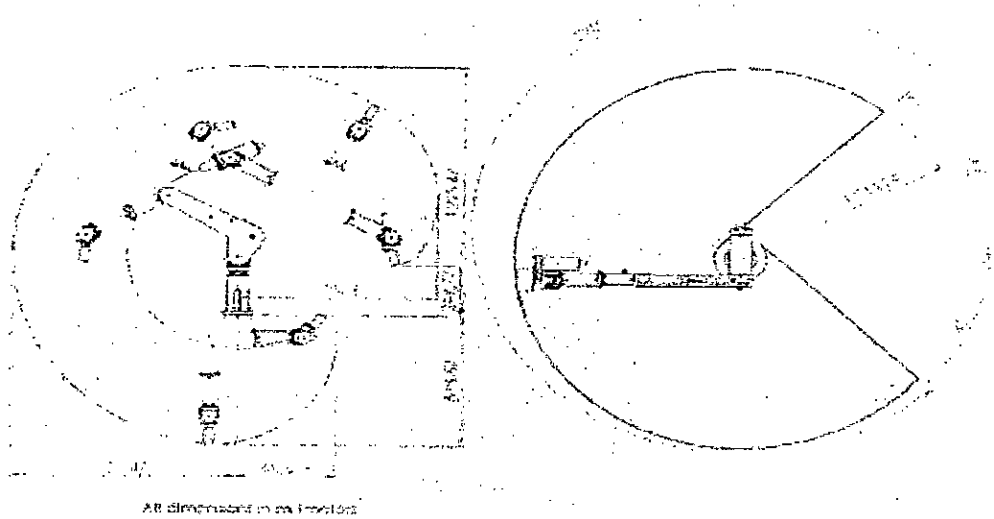


Figure.B.9 : Champ de travail du bras ULM

B.3 La ceinture à ultrason

La plate forme est dotée d'une ceinture à ultrasons, qui est composé de 24 capteurs à ultrasons. Chaque capteur est formé par un émetteur et un récepteur ultrason. Les caractéristiques d'émission d'un émetteur ultrason sont données par la figure ci après. Dans cette figure on voit que l'onde sonore émise se propage selon un cône dont le sommet est la source d'émission, et qui n'est pas concentrée et canalisée dans une seule direction.

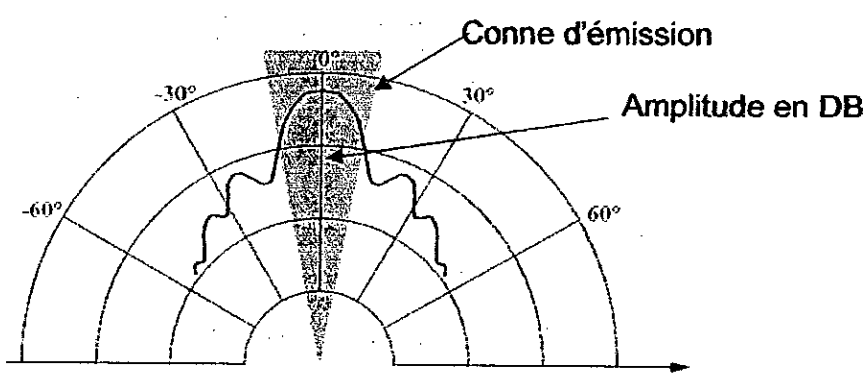


Figure.B.10 : Caractéristiques d'émission d'un télémètre à ultrason

Les 24 capteurs sont disposés sur la plate forme de la manière présentée dans le tableau suivant, mais avant de parler de la position de chaque cellule on va donner l'orientation du repère choisi pour pouvoir se positionner et mieux comprendre.

Axes	Direction	Sens
X	Axe du Robuter	Vers l'avant
Y	Verticale	Vers le haut
Z	Horizontale	Vers la droite
Thêta	0° suivant l'axe X	Trigonométrique

Tableau.B.1: repère lié au Robuter

Les axes ainsi définis sont repérés sur le robot de la manière suivante :

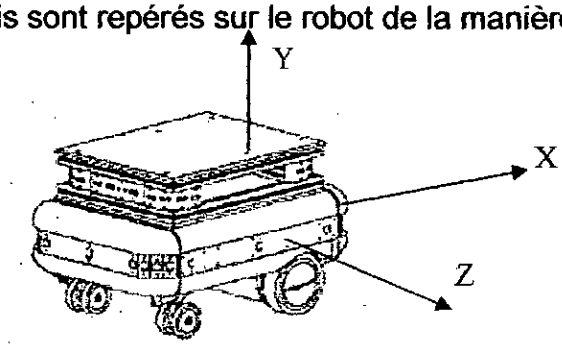


Figure.B.11: Repère considéré du Robuter

Capteur	X	Y	Z	Théta
1	485.96	428.50	-278.50	90
2	535.98	428.50	-265.10	60
3	572.60	428.50	-228.48	30
4	586.00	428.50	-178.46	0
5	586.00	428.50	-90.00	0
6	586.00	428.50	-30.00	0
7	586.00	428.50	30.00	0
8	586.00	428.50	60.00	0
9	586.00	428.50	178.46	0
10	572.60	428.50	228.48	330
11	535.98	428.50	265.10	300
12	485.96	428.50	278.50	270
13	-39.96	428.50	278.50	270
14	-89.98	428.50	265.10	240
15	-126.60	428.50	228.48	210
16	-140.00	428.50	178.46	180
17	-140.00	428.50	90.00	180
18	-140.00	428.50	30.00	180
19	-140.00	428.50	-30.00	180
20	-140.00	428.50	-90.00	180
21	-140.00	428.50	-178.46	180
22	-126.60	428.50	-228.48	150
23	-89.98	428.50	-265.10	120
24	-39.96	428.50	-278.50	90

Tableau.B.2 : Positions des capteurs à ultrasons dans le repère de Robuter [21].

B.2 Le Laser LMS200 [61]

Les performances du modèle SICK LMS200 sont données par le tableau suivant :

Angle d'ouverture	180°
Résolution angulaire	0.25°, 0.5° et 1°
Temps de réponse (fonction de la résolution)	13, 26, 52ms.
Résolution	10mm
Erreur systématique	±15mm
Erreur statique	5mm
Classe d'équipement laser	Classe 1.
Température de fonctionnement	0°c à 50°c.
Distance maximale de mesure	80m.
Interface	Rs422, Rs232.
Taux de transmission	9.6, 19.2, 38.4 et 500 kBaud.
Consommation	20w.
Poids	4.5kg.
Dimensions (L x l x h)	156x155x210mm

Tableau.B.3 : Caractéristiques du LMS200.

APPENDICE C

C.1 La communication réseau

Le but des réseaux est de faire communiquer plusieurs périphériques (ordinateurs) ensemble. Si les hommes communiquent entre eux grâce aux différentes langues, les ordinateurs utilisent différents protocoles.

C.1.1 Définitions et historique

Un réseau est une collection de périphériques permettant de stocker et manipuler des données, périphériques interconnectés entre eux de manière à ce que leurs utilisateurs puissent conserver, récupérer ou partager des informations. Les périphériques connectés peuvent être des micros ordinateurs, des terminaux, des imprimantes ou des appareils de stockage.

Les premiers protocoles réseau conçu ont une origine militaire. C'est en 1969 que le DARPA (*Defense Advanced Research Projects Agency*) finance un projet de recherche sur un réseau expérimental. Le réseau, baptisé ARPANET, a pour but d'interconnecter un ensemble de systèmes propriétaires développés par différents vendeurs en restant le plus indépendant possible du matériel. En 1969, 4 ordinateurs du département militaire US DARPA sont interconnectés entre eux pour faciliter l'échange d'informations entre bases militaires. Le but est aussi de créer un réseau capable par la décentralisation de ses données de survivre à un conflit avec le bloc soviétique. En 1972, ils ne sont encore que 37 serveurs à être reliés via le réseau ARPANET. Le projet s'ouvre progressivement à d'autres institutions scientifiques et académiques US. Dès 1973, s'établissent les premières connexions internationales entre les Etats-Unis et la Norvège [62].

C.1.2 Intérêt d'une communication réseau

L'intérêt d'une communication réseau est évident :

- Pouvoir combiner les compétences de plusieurs personnes ou machines, partager plus aisément l'information. Partager aussi les équipements et donc réaliser des économies substantielles.
- Sécurité : Le recours à des réseaux permet d'exercer un contrôle flexible et centralisé sur l'accès à des données ou équipements sensibles.
Communication mondiale : L'avantage le plus marquant est surtout cette possibilité de transmettre de l'information dans le monde entier quasi instantanément.

C.1.3 Protocole TCP/IP

Plusieurs protocoles de communication réseau sont développés dans la littérature au cours du temps, comme le Universal Naming Conventions, IPX/SPX, NETBEUI, TCP/IP.

TCP signifie Transmission Control Protocol, est un protocole sécurisé orienté connexion conçu pour s'implanter dans un ensemble de protocoles multicouches, supportant le fonctionnement de réseaux hétérogènes. TCP fournit un moyen d'établir une communication fiable entre deux tâches exécutées sur deux ordinateurs autonomes raccordés à un réseau de données. Le protocole TCP s'affranchit le plus possible de la fiabilité intrinsèque des couches inférieures de communication sur lesquelles il s'appuie. TCP suppose donc uniquement que les couches de communication qui lui sont inférieures lui procurent un service de transmission de paquet simple, dont la qualité n'est pas garanti.

TCP/IP est un modèle à cinq couches (figure 4.4) [62]. Les couches 1 et 2 (la couche physique et la couche Datalink) sont caractérisées par des unités d'information de bits et de frames ne sont pas définies par des RFC puisque l'essence même du TCP/IP est d'être indépendant des couches hardware. La couche 3 (IP) est la couche de l'Internet Protocol et de ses datagrammes. L'Internet Control Message Protocol (ICMP) est un service normalement fourni au niveau de cette couche. La couche 3 a besoin des protocoles ARP (Address Resolution Protocol) et RARP (Reverse Address Resolution Protocol) pour associer adresses IP et adresses de la couche MAC. La couche 4 permet de recourir au choix au protocole TCP (Transmission Control Protocol) ou au protocole UDP (User Datagram Protocol). Enfin, la cinquième couche fournit

différents services pour tout type d'applications désireuses de faire appel à des fonctions réseau. Ce serait par exemple le cas d'applications comme Telnet, FTP, du courrier électronique, un navigateur World Wide Web etc.

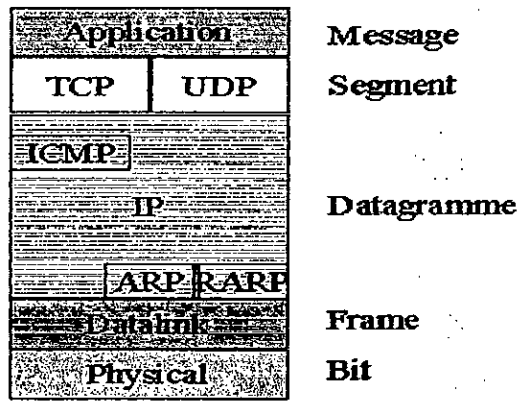


Figure C.1: Les couches du modèle TCP/IP

TCP est un protocole orienté connexion, fiable incluant détection et correction d'erreur, sans surcharge. Il se charge d'établir un handshake entre deux hôtes. TCP est également chargé d'adresser un datagramme à la bonne application via un nombre de 16 bits appelé le port. TCP découpe l'information par paquets puis numérote ces paquets, en calcule un checksum, et les place dans une enveloppe TCP. L'enveloppe TCP est ensuite placée dans une enveloppe IP (Internet Protocol) où sont mentionnées les adresses de l'émetteur et du destinataire. Le bloc de données que TCP passe à IP consiste en une entête TCP et les données en provenance de la couche application. Ce bloc est appelé 'segment'. Le protocole TCP/IP communique en paquet, les paquets TCP sont envoyés sous forme de datagrammes Internet, chaque paquets se compose d'une en-tête et la donné à envoyé proprement dite. Ce format est illustré comme suite [62].

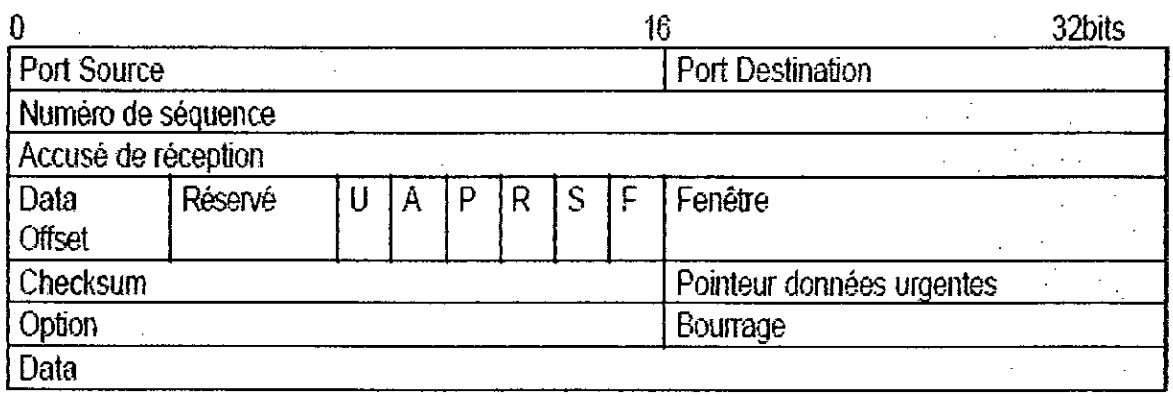


Figure C.2: Format des paquets TCP/IP

L'en-tête IP transmit un certain nombre de paramètres, tels que les adresses Internet source et destinataires. Les différents champs de l'en-tête sont :

- Port Source (16 bits): Le numéro de port de la source.
- port Destination (16 bits): Le numéro de port de la destination.
- Numéro de Séquence (32 bits) compte les octets du flux de transmission de manière à identifier la position du premier octet de données d'un segment dans le flot des données initiales.
- Accusé de réception (32 bits) : ce champ contient le numéro de séquence du prochain octet que le récepteur s'attend à recevoir. Une fois la connexion établie, ce champ est toujours renseigné.
- Data Offset (04 bits): La taille de l'en-tête TCP en nombre de mots de 32 bits. Il indique là ou commence les données. L'entête TCP, dans tous les cas à une taille correspondant à un nombre entier de mots de 32 bits.
- Réservé (06 bits) : Réservés pour usage futur. Ces bits doivent nécessairement être à 0.
- Bits de contrôle: 6 bits (de gauche à droite):

  URG: Pointeur de données urgentes significatif

  ACK: Accusé de réception significatif

  PSH: Fonction Push

  RST: Réinitialisation de la connexion

  SYN: Synchronisation des numéros de séquence

  FIN: Fin de transmission
- Window (16 bits) : annonce la quantité d'espace adressable alloué pour une connexion.
- Le checksum ou champ de contrôle de l'en-tête (16 bits) contient le « complément à un » du total « en complément à un » de tous les mots de 16 bits de l'en-tête.
- Urgent Pointer (16 bits) pointe à la fin d'un champ de données considéré comme urgent
- Options (longueur variable) ne comporte qu'une possibilité : le MSS ou Maximum Segment Size désignant la taille maximum du segment à envoyer.
- Le champ « padding » est habituellement bourré de 0 de manière à aligner le début des données sur un multiple de 32 bits.

C.2 SynDEx et la méthodologie AAA

SynDEx est un environnement de programmation graphique interactif pour des applications de traitement du signal et d'automatique s'exécutant en temps réel sur des machines multiprocesseur [63] [64] [65].

C.2.1 La méthodologie AAA

Afin de réduire le nombre d'erreurs de spécification des algorithmes et de limiter au max les tests matériels, on trouve la méthode AAA permettant d'aider à l'implantation d'un algorithme sur une architecture donnée, conduisant à proposer des modifications de l'architecture (dimensionnement), on à remettre en cause l'algorithme.

C.2.1.1 Algorithme [65]

L'algorithme est modélisé par un hypergraphe orienté, chaque sommet du graphe (nœuds) représente une opération de calcul, d'entrée sortie, de mémorisation ou de conditionnement. Chaque arc reliant deux nœuds traduit :

- un transfert itératif de données (flot de données) établissant une précédence entre deux actions
- un ordre partiel sur les opérations à réaliser, encore appelé ordre d'exécution.

L'ensemble des arcs et nœuds forme un graphe de dépendance des données.

C.2.1.2 Architecture [65]

Elle est modélisée par un graphe non orienté représentant un réseau de processeurs :

MIMD : Multiple Instruction Multiple Data (chaque processeur effectue son programme sur ses propres données) ou SPMD : Single Programme Multiple Data (plusieurs processeurs effectuent le même programme sur des données différentes) dont chaque sommet est un processeur et chaque arc est une

liaison physique de communication bidirectionnelle qui permet des transferts de données entre les mémoires des processeurs, au besoin par l 'intermédiaire d'une mémoire commune.

Chaque processeur comprend une unité de calcul, une unité d'interface avec l'environnement (E/S), une unité de communication pour chaque arc adjacent, une unité de mémoire partagée.

C.2.1.3 Adéquation [65]

Après avoir spécifier et caractériser les deux graphes de l'application, il suffit de presser sur un bouton pour exécuter l'heuristique de distribution et d'ordonnancement de SyndEx. Pour chaque processeur (resp SAM) SyndEx affiche verticalement, les opérations de calcul (resp communications) distribuées et ordonnancées par l'heuristique. La hauteur des rectangles, qui symbolisent les opérations de calcul ou de communication est proportionnelle à leur durée d'exécution.

Si la durée totale d'exécution ne satisfait pas les contraintes temps réel exigées par l'application, alors l'interface graphique de SyndEx permet à l'utilisateur d'agir sur l'heuristique en imposant des contraintes de placement (forcer une opération à être exécuter systématiquement sur un opérateur donné) sinon l'utilisateur peut modifier le graphe d'architecture et relance l'heuristique pour trouver une solution satisfaisante.

C.3 La caméra CCD

Une caméra CCD est composée d'une matrice de détecteurs photosensibles (photodiode) qui accumulent des charges électriques en proportion de la lumière qu'ils reçoivent. Les tailles des matrices sont données en nombres de pixels, qui définissent la résolution des images pour une optique donnée. Les matrices les plus courantes ont des tailles typiques de 340 x280 pixels; les plus grandes matrices actuelles vont jusqu'à 4000 x 4000 pixels. Les pixels ne sont pas toujours carrés, ils ont eux-mêmes des tailles variables selon les modèles, de 6 à 40 µm (distances entre centres des pixels).

Le principe de fonctionnement de ce type de caméra est très simple, l'éclairement de la matrice provoque l'accumulation de charges électriques dans chacune des photodiodes, où chaque photodiode traduit la quantité de luminance qu'il a reçue par un signal électrique positif compris entre 0,3v et 1v (0,3v : représente le noir, 1v : représente le blanc et entre les deux se situent les différents niveaux de gris). Ensuite un dispositif électronique de lecture se charge de récolter les charges, émis par les photodiodes, en séquence ligne par ligne vers la broche de sortie. Viens ensuite la conversion des signaux analogiques en données numériques positives comprises entre 0 et 255 (0 : noir, 255 : blanc et entre les deux se situent les différents niveaux de gris). La matrice et l'opération de lecture sont bien présentées dans la figure suivante:

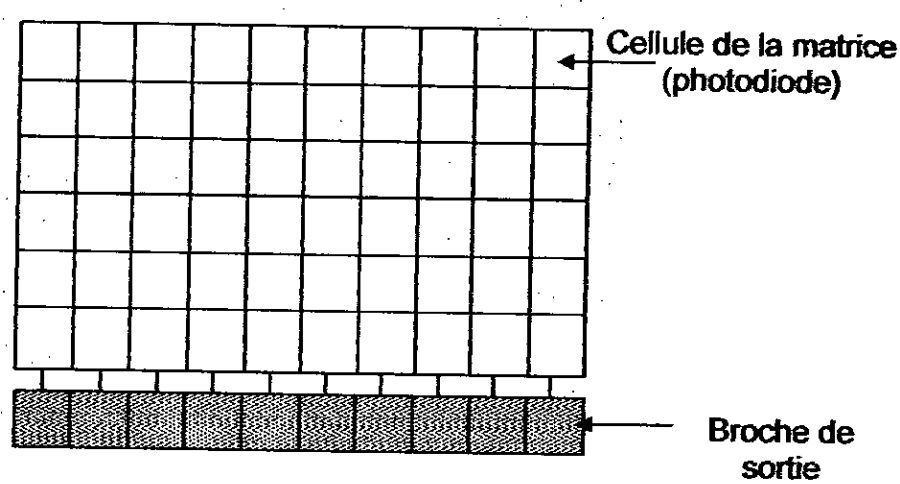
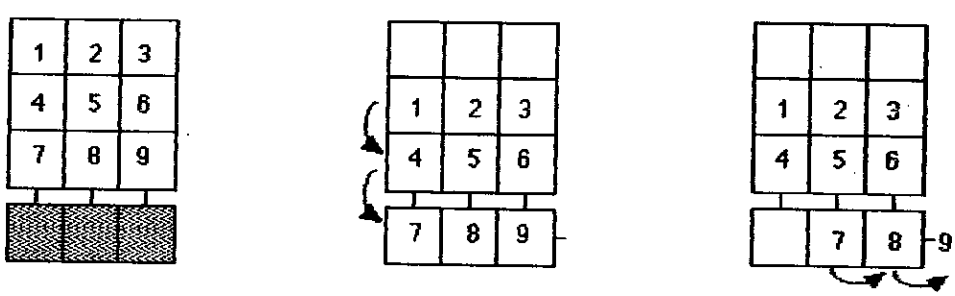


Figure C.3 : Schéma d'une matrice de caméra CCD

Pour bien expliquer le phénomène d'acquisition et de lecture d'image, on prend à titre d'exemple une matrice de 3x3 pixels.



Acquisition d'image = Accumulation des charges électriques

Décalage de ligne

Lecture de ligne

Figure C.4 : Etapes d'acquisition d'une image

APPENDICE D

D.1 Calcul du modèle géométrique inverse du bras manipulateur

Nous avons adopté pour le calcul du modèle géométrique inverse du bras ULM la méthode analytique expliquée et proposée en [37].

La position à atteindre par le bras manipulateur est définie par la matrice suivante :

$$T_{06desiré} = \begin{bmatrix} T_{11} & T_{12} & T_{13} & T_{14} \\ T_{21} & T_{22} & T_{23} & T_{24} \\ T_{31} & T_{32} & T_{33} & T_{34} \\ T_{41} & T_{42} & T_{43} & T_{44} \end{bmatrix} \qquad (D.1)$$

On pose : $T_{06}^{*} = T_{06desiré}$

Soient les notations suivantes :

$$c = \sqrt{T_{11} + T_{22} + T_{33} + 1}\ /2$$

$$p = (\mathrm{sgn}(T_{32} - T_{23})).\sqrt{T_{11} - T_{22} - T_{33} + 1}\ /2$$

$$q = (\mathrm{sgn}(T_{13} - T_{31})).\sqrt{-T_{11} + T_{22} - T_{33} + 1}\ /2$$

$$r = (\mathrm{sgn}(T_{21} - T_{12})).\sqrt{-T_{11} - T_{22} + T_{33} + 1}\ /2$$

Où : x, y, z : sont les coordonnées cartésiennes de la positions à atteindre.

c, p, q, r : paramètres à utiliser plus tard.

On pose :

$$T_{06}^{*} = \begin{bmatrix} T_{11}^{*} & x & T_{13}^{*} & T_{14}^{*} \\ T_{21}^{*} & x & T_{23}^{*} & T_{24}^{*} \\ T_{31}^{*} & x & T_{33}^{*} & T_{34}^{*} \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (D.2)$$

Pour le reste du calcul on adopte la notation :

$c_i = \cos(\theta_i)$ et $c_{ij} = \cos(\theta_{ij})$

$s_i = \sin(\theta_i)$ et $s_{ij} = \sin(\theta_{ij})$

Où :

$$T_{11}^* = c^{*2} + p^{*2} - q^{*2} - r^{*2}$$
$$T_{21}^* = 2.(q^*.p^* + c^*.r^*)$$
$$T_{31}^* = 2.(p^*.r^* - c^*.q^*)$$
$$T_{13}^* = 2.(p^*.r^* + c^*.q^*)$$
$$T_{23}^* = 2.(q^*.r^* - c^*.p^*)$$
$$T_{33}^* = c^{*2} - p^{*2} - q^{*2} + r^{*2}$$
$$T_{14}^* = x^* - d_7.T_{13}^*$$
$$T_{24}^* = y^* - d_7.T_{23}^*$$
$$T_{34}^* = z^* - d_7.T_{33}^*$$

D'autre part on a: $T_{06} = T_{03}.T_{36} \Rightarrow T_{36} = T_{03}^{-1}.T^*_{06}$ (D.3)

Si on pose : $T_{36} = T_{36}^*$ et on a : $T_{36}^* = T_{03}^{-1}.T_{06}^* = T_{03}^T T_{06}^*$ donc : $T_{03} = T_{01}.T_{12}.T_{23}$

Après calcul on trouve :

$$T_{03} = \begin{pmatrix} c_1.c_2.c_3 - c_1.s_2.s_3 & -c_1.c_2.s_3 - c_1.s_2.c_3 & s_1 & a_2.c_1.c_2 + s_1.d_2 \\ s_1.c_2.c_3 - s_1.s_2.s_3 & -s_1.c_2.s_3 - s_1.s_2.c_3 & -c_1 & a_2.s_1.c_2 - c_1.d_2 \\ s_2.c_3 + c_2.s_3 & -s_2.s_3 + c_2.c_3 & 0 & a_2.s_2 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad \text{(D.4)}$$

Donc :

$$T_{03}^T = \begin{pmatrix} c_1.c_2.c_3 - c_1.s_2.s_3 & s_1.c_2.c_3 - s_1.s_2.s_3 & s_2.c_3 + c_2.s_3 & 0 \\ -c_1.c_2.s_3 - c_1.s_2.c_3 & -s_1.c_2.s_3 - s_1.s_2.c_3 & -s_2.s_3 + c_2.c_3 & 0 \\ s_1 & -c_1 & 0 & 0 \\ a_2.c_1.c_2 + s_1.d_2 & a_2.s_1.c_2 - c_1.d_2 & a_2.s_2 & 1 \end{pmatrix} \quad \text{(D.5)}$$

Et on a : $T_{36} = T_{34}.T_{45}.T_{56}$ Après calcul on trouve :

$$T_{36}=\begin{pmatrix} c_4.c_5.c_6-s_4.s_6 & -c_4.c_5.s_6-s_4.c_6 & c_4.s_5 & 0 \\ s_5.c_6 & -s_5.s_6 & -c_5 & -d_4 \\ s_4.c_5.c_6+c_4.s_6 & -s_4.c_5.s_6+c_4.c_6 & s_4.s_5 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad \text{(D.6)}$$

Finalement par identification et en utilisant la formule (C.3) on peut écrire le système d'équations suivant :

$$c_{23}.(c_1 t^*_{11}+s_1 t^*_{21})+s_{23} t^*_{31}=c_4.c_5.c_6-s_4.s_6. \quad \text{(D.7.1)}$$

$$c_{23}.(c_1 t^*_{13}+s_1 t^*_{23})+s_{23} t^*_{33}=c_4.s_5 \quad \text{(D.7.2)}$$

$$c_{23}(c_1 t^*_{14}+s_1 t^*_{24})+s_{23} t^*_{34}=0 \quad \text{(D.7.3)}$$

$$c_{23} t^*_{21}-s_{23}.(c_1 t^*_{11}+s_1 t^*_{21})=c_5.s_6 \quad \text{(D.7.4)}$$

$$c_{23} t^*_{23}-s_{23}(c_1 t^*_{13}+s_1 t^*_{23})=-c_5 \quad \text{(D.7.5)}$$

$$c_{23} t^*_{24}-s_{23}(c_1 t^*_{14}+s_1 t^*_{24})=-d_4 \quad \text{(D.7.6)}$$

$$s_1 t^*_{11}-c_1 t^*_{21}=s_4.c_5.c_6+c_4.s_6 \quad \text{(D.7.7)}$$

$$s_1 t^*_{13}-c_1 t^*_{23}=s_4.s_5 \quad \text{(D.7.8)}$$

$$s_1 t^*_{14}-c_1 t^*_{24}=0 \quad \text{(D.7.9)}$$

$$c_2.a_2(c_1 t^*_{11}+s_1 t^*_{21})+s_1.d_2 t^*_{11}-c_1.d_2 t^*_{21}.+s_2.a_2 t^*_{31}=0 \quad \text{(D.7.10)}$$

$$c_2.a_2(c_1 t^*_{13}+s_1 t^*_{23})+s_1.d_2 t^*_{13}-c_1.d_2 t^*_{23}.+s_2.a_2 t^*_{33}=0 \quad \text{(D.7.11)}$$

$$c_2.a_2(c_1 t^*_{14}+s_1 t^*_{24})+s_1.d_2 t^*_{14}-c_1.d_2 t^*_{24}.+s_2.a_2 t^*_{34}=0 \quad \text{(D.7.12)}$$

Le modèle géométrique inverse est obtenu en résolvant ce système d'équation.

On a la solution d'une équation de la forme $A\cos\alpha+B\sin\alpha=c$ est donnée par :

Si $A^2+B^2\geq C^2$ alors :

$$\sin\alpha=(B.C+\varepsilon.A\sqrt{A^2+B^2-C^2})/(A^2+B^2)$$
$$\cos\alpha=(A.C+\varepsilon.B.\sqrt{A^2+B^2-C^2})/(A^2+B^2) \quad \text{(D.8)}$$

$$\Rightarrow \alpha=arctg(\sin\alpha/\cos\alpha) \quad \text{(D.9)}$$

Avec $\varepsilon=\pm 1$.

D.1.1. Calcul de $\theta 1$

De (C.7.9) on a : $s_1 t_{14}^* - c_1 t_{24}^* = 0$ alors $A1 = -t_{24}^*$; $B1 = t_{14}^*$; $C1 = 0$

Directement on a : $\theta_1 = arctg(\frac{s_1}{c_1}) = arctg(\frac{t_{24}^*}{t_{14}^*})$

$$\theta_1 = -arctg(\frac{s_1}{c_1}) = -arctg(\frac{t_{24}^*}{t_{14}^*})$$

D.1.2.Calcul de $\theta 2$

De (C.7.10) on a

$$A_2 = a_2.c_1\left(c_1 t_{14}^* + s_1 t_{24}^*\right) + s_1$$
$$B_2 = a_2 t_{34}^*$$
$$C_2 = c_1 d_2 t_{24}^* - s_1 d_2 t_{14}^*$$

$$\sin\theta_2 = (B_2 C_2 \pm A_1\sqrt{A_2^2 + B_2^2 - C_2^2})/(A_2^2 + B_2^2)$$
$$\cos\theta_2 = (A_2 C_2 \pm B_2\sqrt{A_2^2 + B_2^2 - C_2^2})/(A_2^2 + B_2^2)$$

1er cas : $\varepsilon 2 = +1$ $\quad \theta_2 = arctg(\frac{\sin\theta_2}{\cos\theta_2})$

2eme cas : $\varepsilon 2 = -1$ $\quad \theta_2 = arctg(\frac{\sin\theta_2}{\cos\theta_2})$

D.1.3.Calcul de θ_3

De (C.7.3) on a :

$$A_3 = c_1 t_{14}^* + s_1 t_{24}^*$$
$$B_3 = t_{34}^*.$$
$$C_3 = 0$$

$$\sin\theta_{23} = (B_3 C_3 \pm A_3\sqrt{A_3^2 + B_3^2 - C_3^2})/(A_3^2 + B_3^2)$$
$$\cos\theta_{23} = (A_3 C_3 \pm B_3\sqrt{A_3^2 + B_3^2 - C_3^2})/(A_3^2 + B_3^2)$$

1er cas : $\varepsilon 2 = +1$ $\quad \theta_2 = arctg(\frac{\sin\theta_2}{\cos\theta_2})$

2eme cas : $\varepsilon 2 = -1$ $\quad \theta_2 = arctg(\frac{\sin\theta_2}{\cos\theta_2})$

Pour l'angle 3 on a 2 valeurs $\varepsilon = +1$ $\quad \theta_3 = \theta_{23} - \theta_2$

$$\underline{\varepsilon = -1} \qquad \theta_3 = \theta_{23} - \theta_2$$

D.1.4.Calcule de θ_5

De (C.7.5) on a : $\theta_5 = a\cos\left[-c_{23}.t_{21}^* + s_{23}\left(c_1.t_{13}^* + s_1.t_{23}^*\right)\right]$

D.1.5.Calcul de θ_4

De (C.7.8) on a $\theta_4 = a\sin\left(\frac{s_1.t_{13}^* - c_1.t_{23}^*}{s_5}\right)$

D.1.Calcule de θ_6

$$A_6 = c_6.s_4.c_5$$

De (C.7.7) on a : $B_6 = c_4.s_6$

$$C_6 = s_1.t_{11}^* - c_1.t_{21}^*$$

$$\sin\theta_6 = (B_6.C_6 \pm \varepsilon_6.A_6.\sqrt{A_6^2 + B_6^2 - C_6^2})/(A_6^2 + B_6^2)$$

$$\cos\theta_6 = (A_6.C_6 \pm \varepsilon_6.B_6.\sqrt{A_6^2 + B_6^2 - C_6^2})/(A_6^2 + B_6^2)$$

1er cas : $\varepsilon_6 = +1$ $\qquad \theta_6 = arctg(\frac{\sin\theta_6}{\cos\theta_6})$

2eme cas : $\varepsilon_6 = -1$ $\qquad \theta_6 = arctg(\frac{\sin\theta_6}{\cos\theta_6})$

D.2 Calibration de la camera

Pour la calibration de la caméra nous avons conçu le système mécanique qui figure dans le schéma suivant:

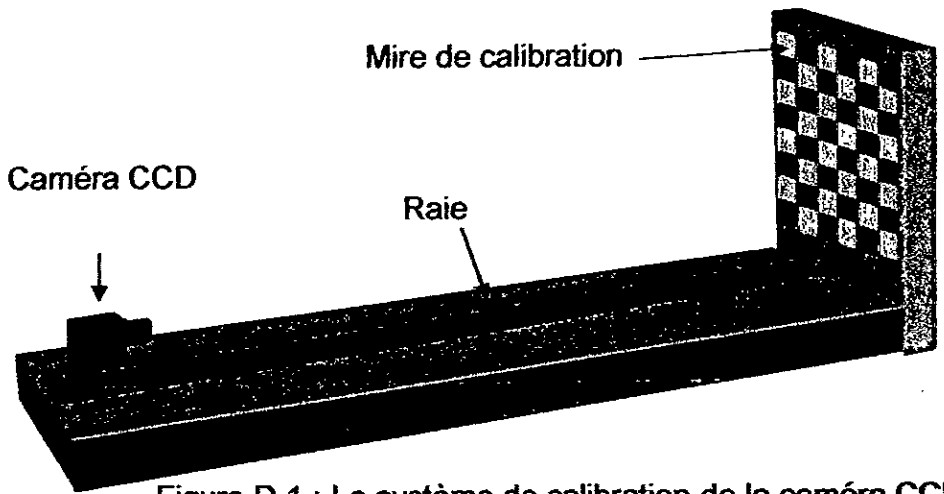


Figure D.1 : Le système de calibration de la caméra CCD

Ce système est composé d'un support plan positionné horizontalement, sur lequel est fixé un socle (une raie) pour pouvoir fixer et déplacer la caméra vers l'avant et vers l'arrière. Un deuxième support de type plan est fixé perpendiculairement par rapport au premier plan sert comme support de mire (ou grille) de calibration.

Le mécanisme réel de cet appareil est conçu en aluminium afin d'assurer le maximum de rigidité et le minimum de poids.

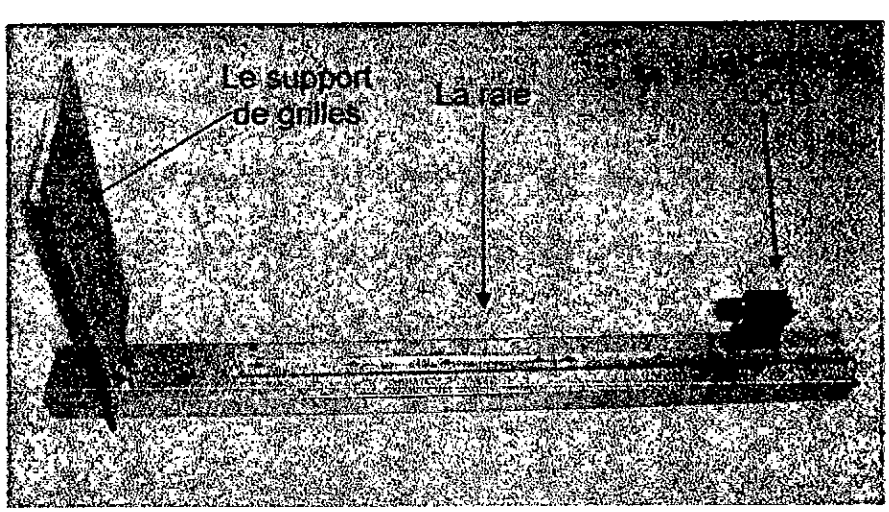


Figure D.2 : Le système réel de calibration de la caméra CCD.

D.2.1 Le principe d'utilisation

Le principe de cette étape est simple. Premièrement on fait déplacer la caméra vers l'avant ou vers l'arrière à une distance désirée, et on la fixe. On fait ensuite à l'aide du logiciel d'acquisition une capture de la prise de vue, et on sauvegarde l'image pour ne pas la perdre car cette opération nécessite une bonne concentration et un minimum d'erreur en linéarité et perpendicularité des repères mis en jeu. Ensuite on sélectionne des points 2D image sur l'écran et on mesure leurs coordonnées 3D correspondants. On répète la procédure autant de fois que nécessaire, bien sur on change à chaque fois la distance de la caméra vis-à-vis du plan devant elle.

L'expérience est répétée en moyenne une dizaine de fois, de cette façon on trouve dans notre base de données des centaines de point 3D<—>2D. Ensuite on exécute le programme qui calcule les paramètres intrinsèques et extrinsèques de la caméra. Finalement on obtient comme résultat une matrice de passage. Cette matrice nous permet de trouver les coordonnées 3D à partir d'un point image ou l'inverse.

D.2.2 La mire de calibration

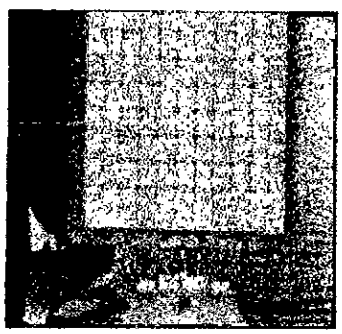
Elle est composée d'un quadrillage sur un plan, avec des carrés ombrés et d'autres blancs. Elle ressemble à une table de jeu d'échec. L'illustration suivante montre un exemple des prise de vues faites avec des distances variables.

Distance de 50cm. Distance de 80cm. Distance de 100cm.

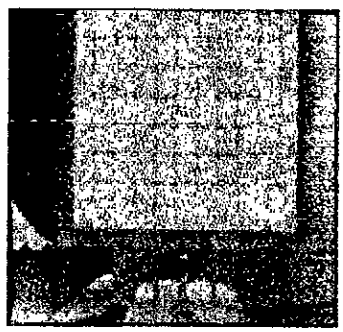
Figure D.3 : Exemple des prises de vues avec la mire de calibration.

D.2.3 La grille de calibration

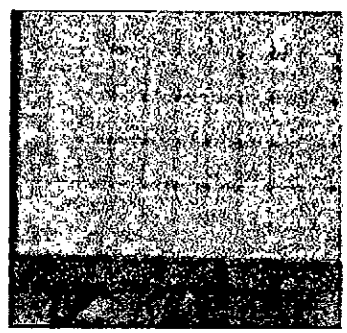
Le deuxième objet de calibration qu'on a utilisé est une grille composée d'un quadrillage. On donne aussi un exemple des prise de vues faites avec des distances variables en se limitant à quelque prise de vues.



Distance de 100cm.



Distance de 100cm.



Distance de 100cm.

Figure D.4 : Exemple des prises de vues avec la grille de calibration.

REFERENCES

1. Coiffet. P, « Télérobotique et téléopération », Edition Hermès Science Publication, Paris, France, 2002, p.19.
2. Claudio. M, « Robotic Telemanipulation an introduction », EURON Winter School on Telesurgery, University of Bologna, Benidorm, Spain, 26 –31 March2006.
3. Otmane. S, « Télétravail Robotisé et Réalité Augmentée : Application à la Téléopération via Internet », thèse de doctorat, université d'Evry val d'Essonne, France, decembre2000, P.11.
4. Otmane. S, « Téléopération, télérobotique et Internet : Techniques & applications », Laboratoire IBISC (Informatique, Biologie Intégrative et Systèmes Complexes), Université d'Evry Val d'Essonne (UFR S&T), CNRS-FR 2873, France, 2000, P.38.
5. Billinghurst. M., Grasset. R & Looser. J, « Designing Augmented Reality Interfaces », ACM SIGGRAPH Computer Graphics, 2005, P. 17-22.
6. Fuchs. P et Moreau. G, « Le traité de la réalité virtuelle », Les Presses de l'Ecole des Mines de Paris, V.1, 2ème édition, France, 2003.
7. « J'automatise » ; Revue scientifique, N°30, Septembre-Octobre 2003, P.41.
8. Klages. M, Drogou. JF & Michel. JL, « French-German cooperation in artic deep sea research: Experiences in the joint use of the deep sea ROV -VICTOR 6000- at a long term arctic station », InWater Tec 2001, Kiel, Germany, 2001.
9. Volpe. R, Ohm. T, Petras. R, Welch. R, Balaram. J & Ivlev. R, « A Prototype Manipulation System for Mars Rover Science Operations Proceeding of International Conference on Intellegent Robots and Systems, IROS'97, vol.3, Grenoble, France, Sept1997, p1486-1492.
10. Rybarczyk. Y, « Etude de l'appropriation d'un système de téléopération dans l'optique d'une Coopération Homme Machine », thèse de doctorat, université d'Evry val d'Essonne, France, Mars 2004, P.17.
11. Rapport d'activité, LSC FRE 2494 CNRS, Laboratoire des Systèmes Complexes, France, 2004, p. 86.

12. Rybarczyk. Y, Mestre. D, Hoppenot. P et Colle. E, « Implémentation télérobotique de l'anticipation sensorimotrice pour optimiser la Coopération Homme machine », Le Travail humain, Paris, France, Janvier 2003, p.10.
13. Rybarczyk. Y, Mestre. D, Hoppenot. P et Colle. E, « Commande d'un système d'assistance robotique aux personnes handicapées », Handicap'2002, Paris, France, juin 2002, p.13-14.
14. Horn. O, Courcelle. A et Kreutner. M, « Le projet VAHM (Véhicule Autonome pour Handicapés Moteur) : la localisation », Laboratoire d'Automatique des Systèmes Coopératifs (LASC), cedex 01, France, 2000.
15. Carbou. J.D, « Conception d'une architecture pour la commande à distance d'un robot d'intervention », thèse de doctorat, université Montpellier II, France, janvier 2004, P.14-20.
16. Otmane. S, Khezami. N & Mallem. M, « la réalité augmenté via Internet : méthodes et architectures pour la Téléopération collaborative », Laboratoire systèmes complexes, Université d'Evry Val d'Essonne, France, 2002.
17. Jacque.W, « Paradigme connexionniste pour l'apprentissage de fonctions robotiques : application au problème asservissement visuel de bras manipulateur», thèse de doctorat, université de Haute-Alsace, France, Mars 1992.
18. Nelson.B et Khosla.P.K, « Integrating sensor placement and visual tracking strategies», Proceedings of the IEEE International Conference on Robotics and Automation, 1994, pages: 1351-1356.
19. Gangloff.J, « Asservissements visuels et chirurgie robotisée », mémoire pour l'obtention de l'habilitation à diriger des recherches (Spécialité Robotique), Décembre 2004.
20. Breton.S, « Une approche neuronale du contrôle robotique utilisant la vision binoculaire par reconstruction tridimensionnelle », thèse de doctorat, université de Haute Alsace, U.F.R. des sciences et techniques, Spécialité Electronique Electrotechnique & Automatique, France, juin 1999, page 24-27.
21. Guérineau. N, Moignard. C et Pomiers. P, « RobuTER et Bras Ultra Léger: Manuel d'utilisation et de maintenance », Version.1, Robosoft, France, 1999.
22. Roland. S and Illah R.N, « Introduction to Autonomous Mobile Robots », A Bradford Book, the MIT Press, Cambridge, Massachusetts, London, England, 2004, P.89-116.

23. Izri. S, Eric. B, Laurent. D et Arnaud. C, « Détection de véhicules dans un environnement autoroutier à l'aide de données télémétriques et visuelles », département d'informatique, Université de Picardie Jules–Verne, CEDEX1, France, 2002.
24. Victorino. AC, Rives. P et Borrely. J, « mobile robot navigation using a sensor-based control strategy », in IEEE international conference in robotic and automation, may 2001.
25. Djarah. DJ, «application de réseaux de neurones pour la gestion d'un système de perception pour un robot mobile d'intérieur », mémoire de magister en électronique, option robotique, Université de Batna, 2006, P.28.
26. RCPU, RISC central processing unit reference manual, revision1, février 1999.
27. Viennet. M, « introduction aux systèmes UINX », France, 2001.
28. Pomiers. P, « The SynDEx linuxIO Macro: An Easy C/C++ Linux User's Application Interface », Robosoft, France, février 2002.
29. Sorel. Y, « SynDEx : un logiciel de CAO niveau système pour l'aide à l'implantation l'applications distribuées temps réel embarquées », INRIA, Cedex, France, 2003.
30. Compion. G, Bastin. G & D'Andrea-Novel.B, « structural properties and classification of kinematic and dynamic model of wheeled mobile robot », Proceedings of IEEE International conference on Robotics and Automation, 1993, pp 4626469.
31. Breton. S, « une approche neuronale du contrôle robotique utilisant la vision binoculaire par reconstruction tridimensionnelle », thèse de doctorat, université de Haute-Alsace U.F.R. des sciences et techniques, France, 1999, pp.65
32. BAYLE. B, « Cours de robotique », DESS Technologies et Stratégies Industrielles, Université Louis Pasteur de Strasbourg, page 17, année 2004/2005.
33. Kelly. A, « Essential Kinematics for Autonomous Vehicles », Carnegie Mellon University, 1994, P.10.
34. Corke. P, « Robotics Toolbox for MATLAB (Release6) », CSIRO, Manufacturing Science and Technology, Pinjarra Hills, Australia, avril 2001, p.7.
35. Kazan. A, « Intégration des aspects robotique, tolérancement et comportement mécanique pour la conception assistée par ordinateur de systèmes poly-articulés », thèse de doctorat, Institut national polytechnique de Gronoble,

Gronoble, France, 2003, page.17.

36. Denavit. J, Hartenberg. R.S, « A kinematic notation for lower-pair mechanisms based on matrices », ASME Journal of Applied Mechanics, Vol.17, 1955, pp. 215-221.

37. Renaud. M & Gorla. B, « Modèles des robots manipulateurs en vue de leur commande », IBID, S.E.E. (Société des électriciens et des radioélectriciens), 1984.

38. Rubén. S, Garcia. R, « Programmation bayesienne des bras manipulateurs », thèse doctorat, institut national polytechnique de Grenoble, France, p.165, 2003.

39. Tarel. J.P, « Calibration de camera fondée sur les ellipses », Rapport de recherche, Université Paris IX, France, 1994, p.13.

40. Kelly. R.B, Izaguirre. A et Kim. R, « Calibrage des caméras de vidéo. Application à l'étude de la dérivée temporelle d'une caméra », rapport de recherche LAAS, n83.066, 1983.

41. Caprile. B et Torre. V, « Using Vanishing Points for Camera Calibration », Journal of Computer Vision, 1990, pp. 127-140.

42. Cipolla. R, Drummond. T & Robertson. D; «Camera calibration from vanishing points in images of architectural scenes », In BMVC, Sep 1999.

43. Horaud. R & Monga. O, « Vision par ordinateur », Hermès, Paris, France, 1993.

44. Batista. J, Ara'ujo. H & Almeida. A, « Iterative Multi step explicit camera calibration », In Proc, Of the 6th International Conference on Computer Vision, Bombay, India, 1998.

45. Beardsley. P, Murray. D & Zisserman. A, « Camera Calibration Using Multiple Images », European Conf. Comp, 1992, pp. 312-320,

46. Martins. H.A, Birk. J.R & Kelly. R.B, « camera models based on data from two calibration planes », Computer graphics and image processing, vol 17, N°2, 1981, pp. 173-180.

47. Faugeras O.D. & Toscani. G, « The calibration problem for stereo », Proceedings of CVPR'86, 1986, pp.15-20.

48. Krupa. A, « Commande par vision d'un robot de chirurgie laparoscopique », thèse de doctorat de l'Institut National Polytechnique de Lorraine, Ecole doctorale IAEM Lorraine, France, 2003, pp.26-40.

49. Benallal. M, « Système de calibration de Caméra, localisation de forme polyédrique par vision monoculaire », thèse de doctorat de l'école des mines de paris, France, 2002, pp. 32-46.
50. Zhang. Z, « A flexible new technique for camera calibration ». IEEE Trans Pattern Anal. Mach. Intell, vol. 11, 2000, pp.1330-1334.
51. Devernay. F, « Vision par ordinateur, Cours dispensés au Mastère Photogrammétrie numérique », ENSG, 2001-2003.
52. Chaumette. F & Rives. P, « réalisation et calibration d'un système expérimental de vision composé d'une caméra mobile embarquée sur un robot manipulateur », Rapport de recherche N°994, IRISA, France, 1989, p. 6.
53. Moudache. S, Charfa.Y, Charfa.A, «Prétraitement et Segmentation en image couleur : application en microscopie cellulaire », Proceding JETIM : Journées d'Etude Algéro française en Imagerie Médicale, Alger, Algérie, 21-22 Novembre 2006, P.179.
54. « http://sde.eduvax.net/ », date de consultation 10/09.2006.
55. « http://glscene.sourceforge.net/ », date de consultation 10/11/2006.
56. « Guide du développeur Delphi 6 pour Windows », Borland Software Corporation, 2002, pp.158.192.
57. « Introduction à C++ Builder », Ours Blanc Des Carpates ™, ISIMA, (1999).
58. « http://www.developpez.com », date de consultation 15 /12/2006.
59. « Développez Magazine », revue du site developpez.com, Numéro 6, Septembre Octobre 2006, P.19.
60. Ait Aider. O, Hoppenot. P et Colle. E, « localisation caméra / objet par correspondances point 3D-pixel : application à la localisation d'un robot mobile », CEMIF-LSC, université d'Evry Val d'Essonne, France, 2002.
61. « www.sick.com», date de consultation: 15/01/2007.
62. Guillaume. D, « Recherche bibliographique: synthèse des protocoles courants », ESEO, 1999.
63. Grand pierre. T et Sorel. Y, « Modèle d'exécutif distribué temps réel pour SynDEx », Rapport de recherche, N° 3476, INRIA, France, Août 1998.
64. « www.inria.fr », site web de l'INRIA, date de consultation : 15/02/2006.
65. « www.SynDEx.org», site web officielle du SynDEx, date de consultation: 20/02/2006.