UNIVERSITE DE MONTREAL

Analyse par voie de simulation du systeme d'autopilotage d'un vehicule autonome

par

Monjy RABEMANANTSOA

DEPARTEMENT DE GENIE ELECTRIQUE

ECOLE POLYTECHNIQUE

MEMOIRE PRESENTE EN VUE DE L'OBTENTION

DU GRADE DE MAITRE ES SCIENCES APPLIQUEES (M.Sc.A)

Août 1989

c Monjy Rabemanantsoa 1989

National Library
of Canada

Canadian Theses Service

Ottawa, Canada
K1A 0N4

Bibliothèque nationale
du Canada

Service des thèses canadiennes

The author has granted an irrevocable non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of his/her thesis by any means and in any form or format, making this thesis available to interested persons.

The author retains ownership of the copyright in his/her thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without his/her permission.

L'auteur a accordé une licence irrévocable et non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de sa thèse de quelque manière et sous quelque forme que ce soit pour mettre des exemplaires de cette thèse à la disposition des personnes intéressées.

L'auteur conserve la propriété du droit d'auteur qui protège sa thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

ISBN 0-315-52708-0

Canada

UNIVERSITE DE MONTREAL

ECOLE POLYTECHNIQUE

Ce memoire intitule :

Analyse par voie de simulation du systeme d'autopilotage d'un vehicule autonome

présenté par: . Monjy. RABEMANANTSOA . . . . . .

en vue de l'obtention du grade de: Maitre es Sciences Appliquées

a été dûment accepté par le jury d'examen constitué de:

M. Roman BALDUR, M.Sc.A, Président

M. Romano DE SANTIS, Ph.D, Directeur de recherche

M. Kalyan GHOSH, D.Eng, Membre de jury

SOMMAIRE

L'objectif de ce mémoire est le développement d'une analyse par voie de simulation du comportement dynamique et statique d'un système d'autopilotage d'un véhicule autonome. Le véhicule considéré est constitué d'un module de locomotion du type employé sur les chaises roulantes pour handicapés; d'un contrôleur de vitesse et de position du type PI et PID; d'un système d'estimation de position hybride utilisant des mesures odométriques et inertielles.

Les questions considérées portent sur la modélisation du système, le développement d'un logiciel de simulation infographique, l'étude de la sensibilité du comportement du système relative aux paramètres qui la caractérisent: gains de contrôleurs, conditions initiales du véhicule, effets de perturbations, retards dans les mesures, précisions des capteurs etc ...

Les résultats confirment que le système d'autopilotage considéré a un comportement dynamique et statique très satisfaisant; il a une bonne robustesse aux variations des paramètres et à la présence de perturbations extérieures; il fonctionne bien en correspondance avec une grande variété d'opérations de travail.

ABSTRACT

The purpose of this project is a development analysis by mean of simulation of the dynamic and static behaviour of an autopilot system related to an autonomous vehicle. Design considerations consist of a motion module similar to the one required by a disabled person's rolling-chair; a PI controller for the speed and a PID controller for the position; readings from odometry measurement and sensors provide data to the vehicle's position estimate module.

The project involves therefore the design of the model required for an autopilot system, the software implementation for a real-time graphic simulation, the study of the sensitivity of the system behaviour as a function of its parameters : controller gains, starting conditions, disturbance effect, delays into measurements, sensors accuracies etc ....

Simulation outputs produce good results : dynamic and static behaviour of the autopilot system are satisfactory. The system is robust with respect to external disturbances and parameters variations. The performance is good under a large variety of operating conditions.

REMERCIEMENTS

Je désire exprimer ma reconnaissance envers mon Directeur de recherche, Dr. Romano M. DE SANTIS, professeur au département de Génie Electrique pour le thème de la recherche et son dévouement pour la supervision et l'orientation du projet tout au long de son développement.

L'infographie et l'animation graphique par ordinateur ont bénéficié de l'enseignement de M. Daniel THALMANN à qui je témoigne ma gratitude.

Un remerciement au Dr. Richard HURTEAU, professeur, pour l'accès à la station de travail CATIA relative à certaines simulations et un amical témoignage à mes collègues de travail, ceux qui m'ont aidé par leur conseil et la documentation.

TABLE DES MATIÈRES

PAGE

SOMMAIRE iv

ABSTRACT v

LISTE DES FIGURES ix

LISTE DES TABLEAUX xii

CHAPITRE 1 - INTRODUCTION 1

1.1 Introduction 1

1.2 Organisation du mémoire 7

CHAPITRE 2 - MODÈLE MATHÉMATIQUE ET SYSTÈME DE CONTRÔLE 9

2.1 Le système en étude 9

2.2 Modélisation des composantes 15

2.3 Modélisation du système de locomotion 17

2.4 Modélisation du système de mesure/estimation 20

2.5 Modélisation du système de contrôle 26

2.6 Modélisation de l'ensemble du système 33

PAGE

CHAPITRE 3 - DESCRIPTION DU LOGICIEL

3.1 Introduction 35
3.2 Structure du logiciel 37
3.3 Structure interne du logiciel 45

CHAPITRE 4 - ANALYSE DU SYSTÈME

4.1 Introduction 54
4.2 Performance dynamique du système 55
Expérience 1 60
Expérience 2 64
Expérience 3 70
Expérience 4 74
Expérience 5 79
Expérience 6 84
Expérience 7 88
4.3 Conclusion 94

PAGE

CHAPITRE 5 - CONCLUSION

5.1 Résultats obtenus 96

5.2 Améliorations possibles 98

BIBLIOGRAPHIE 100

APPENDICE A

a.1 Modules de calculs 105

APPENDICE B

b.1 Présentation des capteurs 118

b.2 Autres types de trajectoire 123

b.3 Simulation sur trajectoire courbe 130

LISTE DES FIGURES

Figure		Page
1.1	Configuration physique du robot mobile	2
2.1	Représentation physique du modèle	10
2.2	Lien entre les différents systèmes	12
2.3	L'ensemble mécanique du système en étude ...	13
2.4	Architecture des interfaces	14
2.5	Schéma du système de lomotion	17
2.6	Localisation du véhicule	19
2.7	Répartition des périodes de mesures	22
2.8	Evaluation de l'erreur de distance	25
2.9	Schéma du contrôleur de position/estimation.	29
2.10	Schéma du contrôleur de vitesse	32
2.11	Modélisation de l'ensemble du système	34
3.1	Schéma bloc du logiciel	38
3.3	Organigramme	46
3.4	Simulation infographique	53
4.1	Trajectoire de simulation	58
4.2	Concept de la trajectoire prédéfinie	59
4.7 @ 4.15	Courbes de simulation	62

Figure Page

b.1 Principe du dead-reckoning 118
b.2 Encodeur optique Sandtron 119
b.3 Principe de détection du capteur Ultrason .. 120
b.4 Diagramme de rayonnement du sonar 121
b.5 Interrupteur infra-rouge 122
b.6 Détermination de la succession de la route . 124
b.7 Trajectoire en courbe 126
b.8 @ b.10 Les types de trajectoire 127
b.11 @ b.14 Courbes de simulation 132

LISTE DES TABLEAUX

Tableau Page

2.1 Liste des équipements 13

2.2 Liste des notations utilisées 15

3.1 Menu principal 39

3.2 Exemple de menu de paramètres 40

3.3 Récapitulation des paramètres 44

3.4 Tableau des paramètres 48

4.1 Tableau des valeurs nominales 62

b.1 Les paramètres pour la faible vitesse 131

b.2 Les paramètres pour la haute vitesse 134

CHAPITRE I

INTRODUCTION

1.1 Introduction

Il existe à l'heure actuelle deux grandes catégories de véhicule autonome ou robot mobile avec de multiples usages discutés dans les références [16] [17] [18] [26]. La première catégorie est le robot sur pattes articulées et la seconde le vehicule sur roues. Ce dernier a eu deux générations dont la première est la famille des AGV (Automated guided vehicle) dont le déplacement s'effectue suivant une ligne peinte ou un cable; la deuxième génération est le robot mobile qui est un véhicule autonome necessitant une étude de trajectoire et de guidage.

Le véhicule considéré prototype du projet est présenté à la figure 1.1. Il fait partie de la seconde catégorie de véhicule sur roues dont deux roues pivotantes à l'avant et deux roues motrices à l'arrière. Il se déplace d'une façon autonome le long d'une route assignée en mémoire. Il peut servir ainsi soit comme véhicule de d'expérimentation d'ordre académique, soit comme véhicule d'assistance dans

les hopitaux pour transporter des appareils (Household robot, Nursing robot : ref [33]), soit pour une application manufacturière ou une application minière.

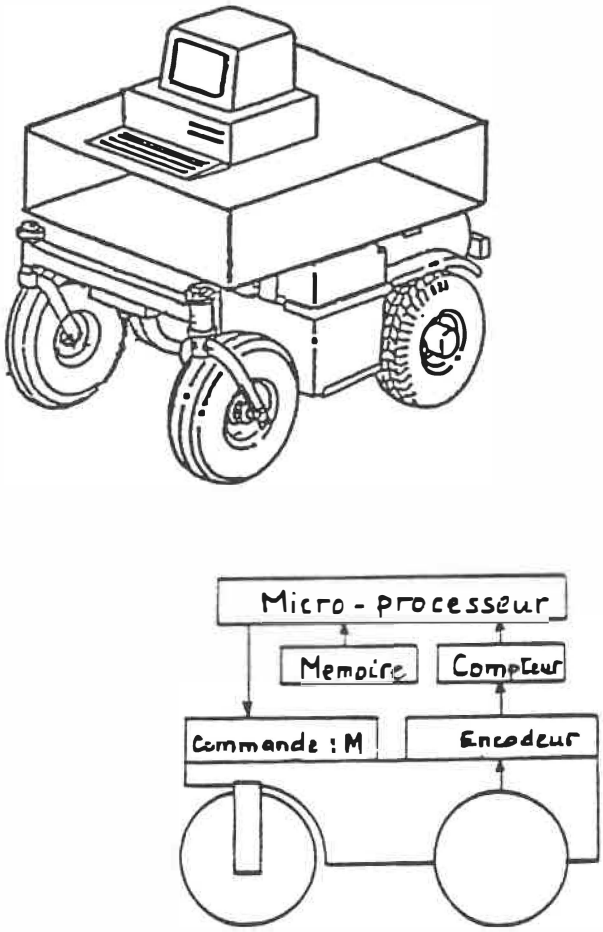


Figure 1.1 : Configuration physique du robot mobile

Le problème de navigation a fait l'objet de plusieurs études pour guider un véhicule mobile le long d'un chemin prédéfini et lui faire accomplir une tâche spécifique. Des approches basées sur l'identification de la trajectoire par vision artificielle ont été discutées par Moravec [15], Sobec et Chatila [16]. D'autres approches sur le suivi d'une trajectoire ont été abordées par Tsumura [18], Julière et Marce [27]; ensuite l'utilisation des capteurs ultrasonic pour la perception de l'environnement et la localisation du robot mobile ont fait l'objet d'étude par Bauzil, Briot, Ribes [17] puis Coles, Alan, Sinclair [20]. A côté de tout ceci il y a eu la technique des AGV dont l'approche consiste à contrôler le véhicule à distance et l'arrêter en cas de risque de collision; le déplacement se fait suivant un tracé prédéfini par cable ou par trace de peinture selon l'application désirée.

Toutefois toutes ces publications présentent des équations propres à leur modèle et dont le déplacement se rattache à leur application spécifique. Les réponses du système en fonction des variations des paramètres du modèle ne sont pas explicitées et on ne comprend pas très bien le fonctionnement des différentes étapes.

Ainsi plusieurs questions restent ouvertes : Comment le comportement dynamique du véhicule réagit-il sous l'influence de conditions nominales en correspondance avec diverses valeurs des erreurs de position initiale, du facteur d'erreur combinée β, de la vitesse désirée ? Quelle est l'influence d'une asymétrie au niveau de la constante de temps ou du gain statique des servomoteurs ? Comment se comporte la réponse du système en présence d'un délai ? L'intervalle de temps qui sépare deux mesures inertielles peut-il influencer le comportement dynamique du véhicule lors d'une bonne précision ou d'une faible précision dans les mesures odométriques ?

Afin de comprendre le phénomène réel de l'autopilotage nous avons développé un projet qui porte sur la modélisation du système, l'implantation d'un logiciel de simulation baptisé **SRM** (**S**imulation **R**obot **M**obile) qui permet de visualiser le déplacement du véhicule en temps réel puis l'étude de la sensibilité du comportement statique et dynamique du système relativement aux paramètres caractéristiques tels que : le gain des contrôleurs, les conditions initiales du véhicule, les effets de perturbations, les retards dans les mesures et les précisions des capteurs .

Le véhicule est propulsé par deux roues motrices indépendantes situées à l'arrière et actionnées chacune par un moteur DC monté sur l'axe de l'essieu arrière qui reçoit respectivement une commande à partir du système de contrôle. La position du véhicule peut être contrôlée avec une précision de 1% pour le mouvement linéaire et de 1 degré pour les orientations.

La trajectoire est divisée en plusieurs segments et le systéme de détection doit déterminer une direction précise par système de guidage autonome. Deux types d'erreurs interviennent lors du suivi de la trajectoire : l'erreur de position latérale ou de distance et l'erreur d'orientation.

L'analyse du déplacement nous ammène à considérer deux niveaux : global et local.

Au niveau global le module de navigation (ou mesure) choisit la position et l'orientation idéales avec les données en provenance des capteurs. Le niveau local guide le véhicule avec les données estimatives appelées données odométriques. Ces deux techniques seront expliquées en détail dans le chapître 2.

Le système de contrôle le long de la trajectoire se fait avec deux sortes de capteurs :

a) Capteurs proprioceptifs

Encodeur optique installé au niveau de chaque roue avant pour mesurer la vitesse. Cette mesure est effectuée avec la même période d'échantillonage qui caractérise le contrôle.

b) Capteurs extéroceptifs

Ce genre de capteur sert aux mesures de l'environnement: capteur ultrason (sonars) et capteur infra-rouge. Ils mesurent la position et l'orientation du véhicule par rapport à la route.

1.2 Organisation du mémoire

Le mémoire est divisé en 5 chapîtres.

Le chapître 1 présente les différentes approches utilisées pour le guidage de véhicules mobiles déja existants. On introduit ainsi un aperçu de la structure des AGV et des robots mobiles, ensuite celle du véhicule prototype du projet.

Le chapître 2 consiste à élaborer les différentes étapes de l'étude; Ceci consiste en premier lieu à la modélisation des composantes notamment le système de locomotion et le système d'estimation. En second lieu la modelisation du système de contrôle : le contrôleur de position/orientation et le contrôleur de vitesse.

Une description de l'architecture du logiciel se trouve au chapître 3 avec les différents accès aux modules et sous-modules par voie de menus. Nous y présentons également la structure interne du logiciel avec les principaux modes de calculs.

Le chapître 4 est consacré à l'analyse du système en fonction des résultats obtenus, ainsi que la présentation des courbes de simulation.

Les conclusions de la recherche sont tirées au chapître 5 suivies des améliorations éventuelles possibles.

CHAPITRE 2

MODÈLE MATHÉMATIQUE ET SYSTÈME DE CONTRÔLE

2.1 Le système en étude

Le système en étude est constitué d'une plate-forme sur laquelle est aménagé un chassis pouvant accueillir un ordinateur, des équipements electroniques, une série de capteurs et des équipements de contrôle.

L'ensemble du système comprend :

- Un système de locomotion avec divers caratéristiques physiques notamment la base motrice, la plate-forme et les accessoires pour l'autonomie.

- Les systèmes de mesure et d'estimation dans lesquels se trouvent les éléments de détection tels que les sonars et les encodeurs associés aux techniques utilisées pour l'autopilotage.

- Un système de contrôle constitué d'un contrôleur PID pour la position et d'un contrôleur PI avec "switch component" pour la vitesse.

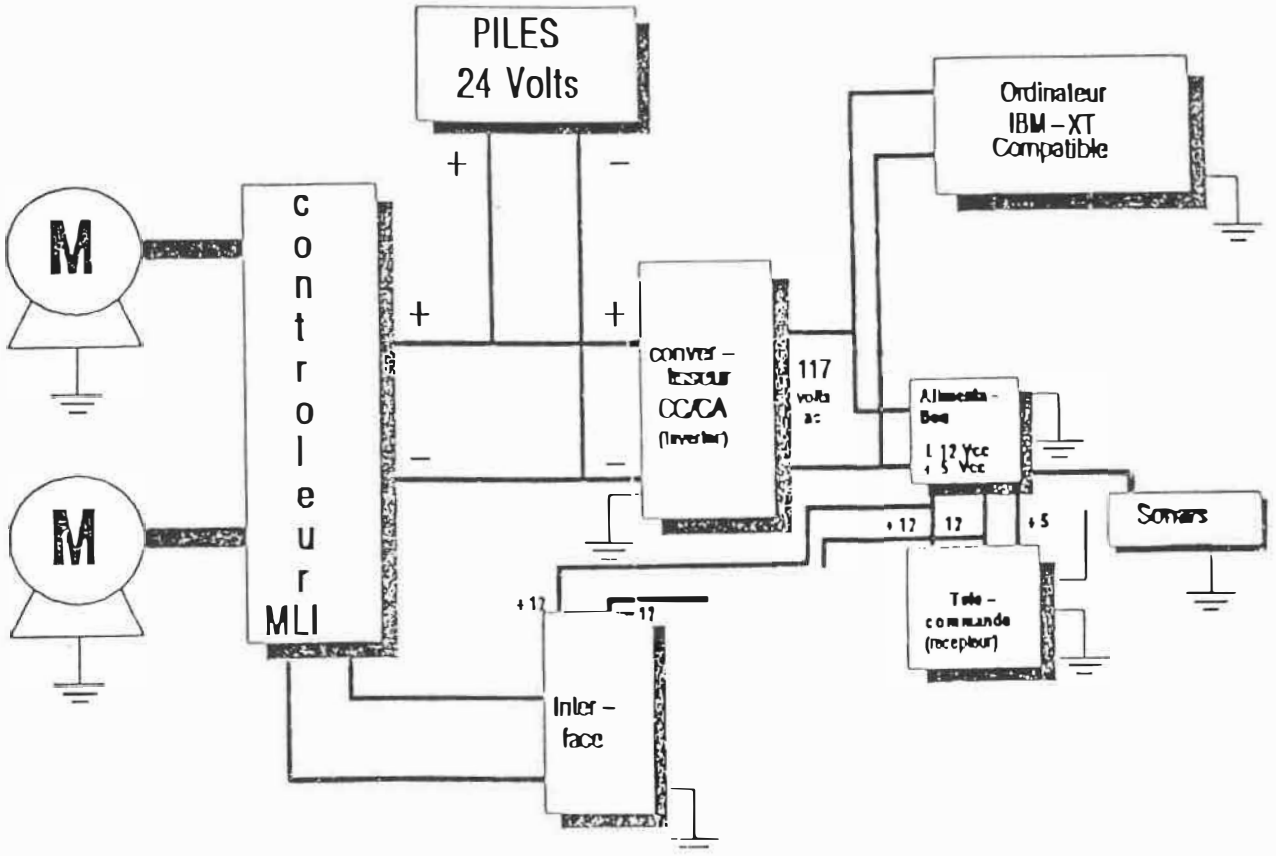


Figure 2.1 : Représentation physique du modèle

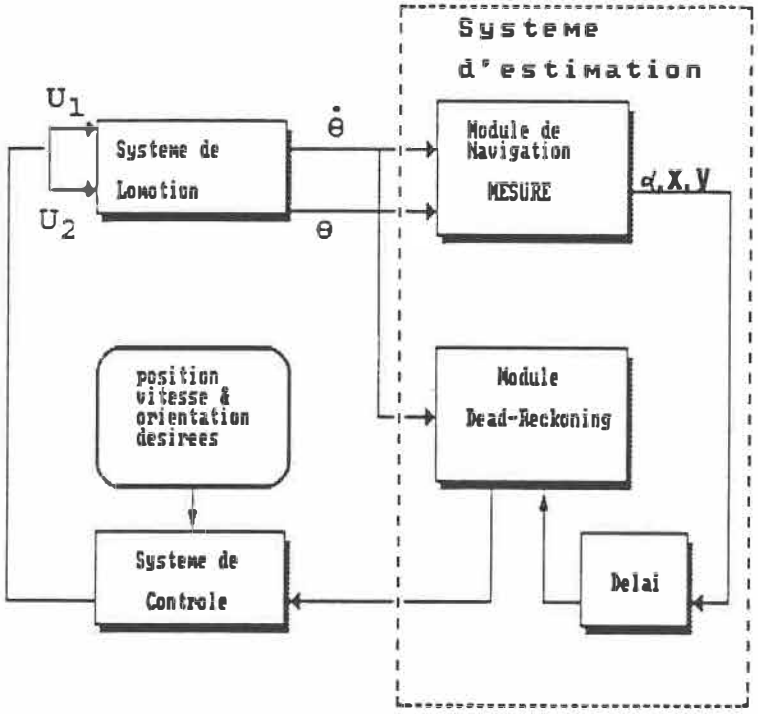


Figure 2.2 : Lien entre les différents systèmes

Une liste détaillée des composantes est donnée au tableau 2.1

1. Base mobile Fortress 6755 FS comprenant:
   - 2 moteurs DC montés sur l'axe de l'essieu arrière (puissance 250 Watts), vitesse maximale de 9 km/h,
   - un amplificateur de tension
   - 2 roues folles à l'avant.

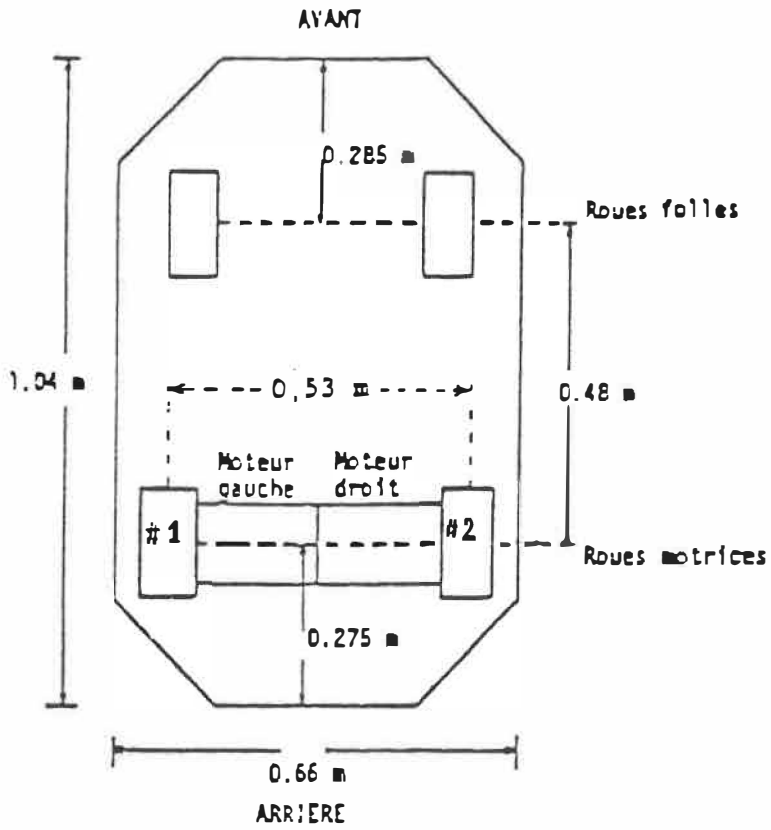
2. Convertisseur c.c. à c.a. (24 volts cc à 110 volts ac, onde carrée) de marque Tripplite d'une puissance de 1000 watts. Alimentation 5, 10 et 15 volts.

3. Capteurs : 2 encodeurs optiques Sandtron REX-44-5 (500 impulsions/tour)
   8 capteurs de proximité de type ultrason de marque Polaroid,
   2 interrupteurs infra-rouge de type 425RU à faisceau étroit et polarisé, sensible uniquement à des rétro-réflecteurs à coin de cube,
   2 boutons-poussoirs pour l'arrêt d'urgence

4. Micro-ordinateur IBM-XT (compatible) standard comprenant:
   - carte d'interface entrée-sortie LabMaster: 16 A/N, 2 N/A, 5 compteurs, 24 E/S parallèles et des accès d'interruption,
   - 2 cartes Base-Board comprenant 96 E/S parallèles.

Tableau 2.1 : Liste des équipements

La structure mécanique du système de locomotion est illustrée à la figure 2.3



Caractéristiques

Base motrice		Plate-forme	
Longueur	0,74 m	Longueur	1,04 m
Largeur	0,61 m	Largeur	0,66 m
Hauteur	0,41 m	Poids avec équip.	90 kg
Poids	80 kg		
Essieu avant	0,46 m	Performance	
Essieu arrière	0,53 m	Vitesse	2,5 m/s
Empattement	0,48 m		9 km/h
Rayon de giration	0,76 m	Distance max.	50 km
Dimension des roues	260 X 85 mm	Autonomie	5,5 h

Figure 2.3 : L'ensemble mécanique du système en étude

La partie électronique est illustrée par l'architecture des interfaces à la figure 2.4; l'autopilotage est supervisé par un micro-ordinateur qui coordonne les composantes énumérées précédemment telles que le système de locomotion, mesure, estimation et contrôle via un bus IBM: la carte LabMaster gère les entrées/sorties à travers une unité numérique/analogique dont 6 bits réservés pour chaque moteur. La carte BaseBoard fonctionne avec un port parallèle et émule avec une carte d'interface qui reçoit la sortie des encodeurs et deux autres cartes recevant la sortie des sonars.

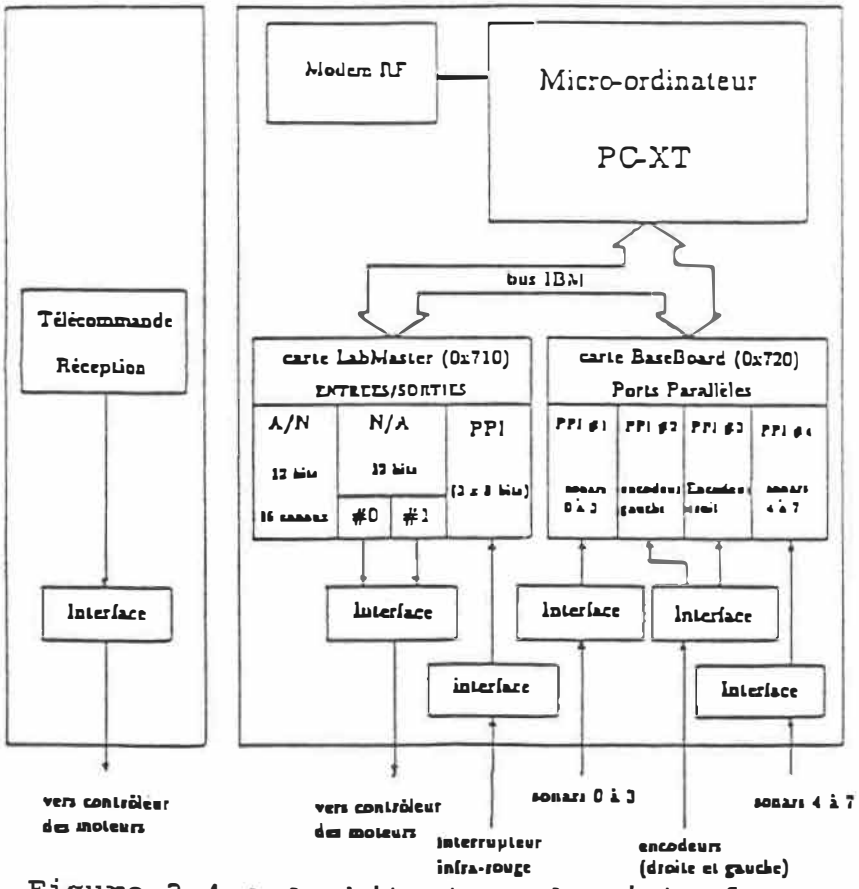


Figure 2.4 : Architecture des interfaces

2.2 Modélisation des composantes

La modélisation du système consiste à donner tout d'abord un modèle mathémathique de chacune de ses composantes notamment le système de locomotion, le système de mesure/estimation et le système de contrôle. Ensuite nous présenterons un modèle de l'ensemble. Pour ce faire, nous utilisons les notations indiquées au tableau 2.2

$x_v(t)$, $y_v(t)$	: Position du centre de l'essieu arrière à chaque instant t
$\alpha_v(t)$	: Angle de l'axe de symétrie longitudinale du véhicule par rapport à l'horizontal
ϵ_d	: Distance perpendiculaire entre la position $x_v(t)$, $y_v(t)$ et la trajectoire (erreur de position latérale ou de distance)
ϵ_α	: Différence entre l'angle de la trajectoire par rapport à l'horizontal et l'angle $\alpha_v(t)$
e	: Erreur totale égale à la somme de ϵ_d et ϵ_α avec leur coefficient respectif
ΔU	: Output du contrôleur PID
Um	: Output du contrôleur PI
U_1	: Tension appliquée à l'ampli pour la commande de la roue gauche
U_2	: Tension appliquée à l'ampli pour la commande de la roue droite
$\dot{\theta}_1$ et $\dot{\theta}_2$	: Vitesses angulaires des roues
$\ddot{\theta}_1$ et $\ddot{\theta}_2$	: Accélérations angulaires des roues

Paramètres de locomotion

TAU1	: Constante de temps du moteur gauche
TAU2	: Constante de temps du moteur droit
S1	: Facteur de glissement* de la roue gauche
S2	: Facteur de glissement* de la roue droite
P_1	: Perturbation sur le moteur gauche
P_2	: Perturbation sur le moteur droit
R1	: Rayon de la roue motrice gauche
R2	: Rayon de la roue motrice droite
L	: Distance entre les roues arrières

Tableau 2.2 : Liste des notations utilisées

(*) : Le facteur de glissement est défini comme etant le rapport de la distance réelle effectuée par la roue par la distance théorique.

2.3 Modélisation du système de locomotion

Le modèle du système de locomotion peut être pensé comme constitué de deux parties : Une partie dynamique et une partie cinématique.

La partie dynamique reçoit les tensions à appliquer aux servo-moteurs et détermine la position et la vitesse angulaires des roues. La partie cinématique calcule la position, orientation et vitesse du véhicule à partir de la position et vitesses des roues. (voir Figure 2.5)

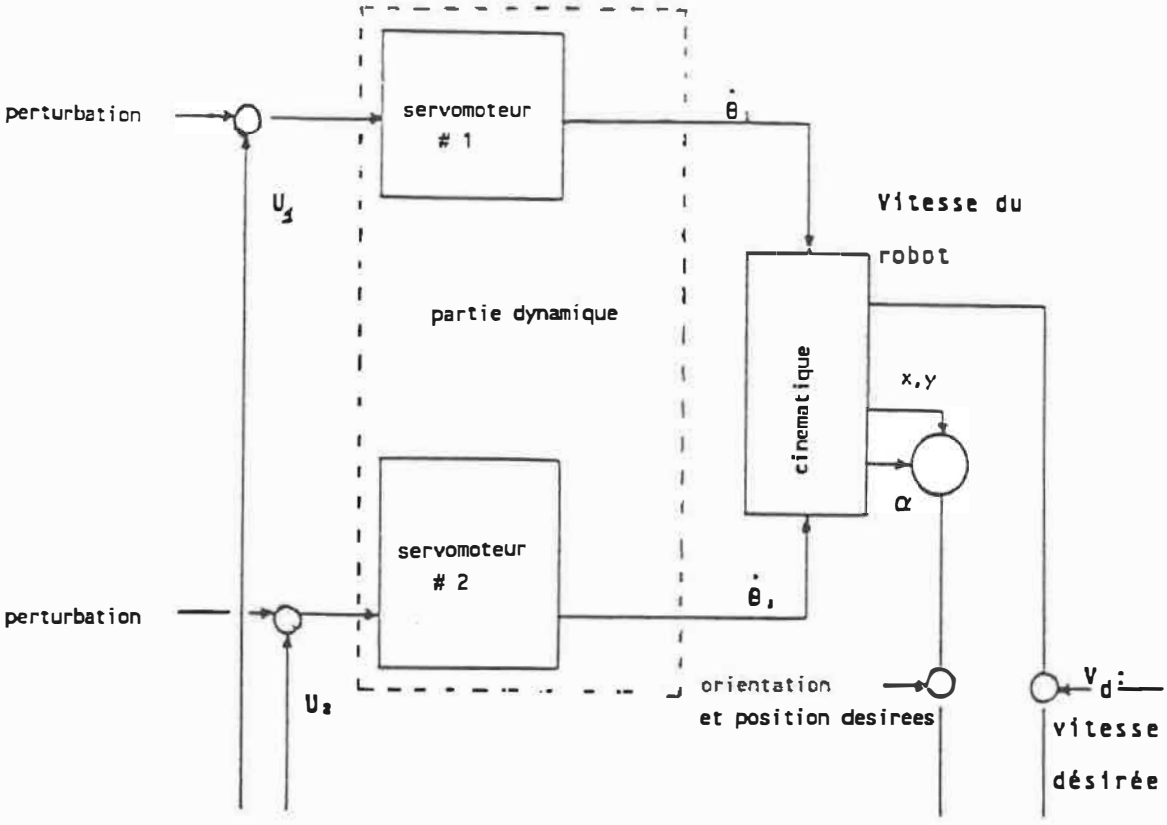


Figure 2.5 : Schéma du modèle du système de locomotion

La partie dynamique du système de locomotion sera modélisée en utilisant une "approche simplifiée" où l'on suppose que le comportement de l'un des 2 servomoteurs est assumé être découplé du comportement de l'autre. Cette hypothèse n'est satisfaite que dans des cas particuliers (i.e. $M = (4*I)/L^2$ [35]; où M est la masse du véhicule; I est le moment d'inertie par rapport à l'axe vertical attaché au véhicule : Figure 2.6). Cependant, elle nous permet d'analyser les phénomènes saillants qui caractérisent la dynamique de notre système d'autopilotage tout en simplifiant la tache de programmation.

En désignant les valeurs des constantes de temps "équivalentes" par TAU1 et TAU2 et les gains statiques par K_{m1} et K_{m2} nous avons

$$\ddot{\theta}_2(t) = -(1/TAU1) * \dot{\theta}_1(t) + (K_{m1}/TAU1)\,[U_1(t) + P_1]$$

$$\ddot{\theta}_1(t) = -(1/TAU2) * \dot{\theta}_2(t) + (K_{m2}/TAU2)\,[U_2(t) + P_2]$$

où : U_1 et U_2 sont les tensions à appliquer à l'ampli pour les commandes de la roue gauche et droite

P_1 et P_2 sont les perturbations

$\dot{\theta}_1$ et $\dot{\theta}_2$ sont les vitesses angulaires des roues

$\ddot{\theta}_1$ et $\ddot{\theta}_2$ sont les accélérations des roues

La relation entre la position, l'orientation et la vitesse du véhicule et les positions et vitesses angulaires des roues est caractérisée par les équations suivantes :

$$\dot{\alpha}_V(t) = \frac{1}{L} \left[R_2 s_2 \dot{\theta}_2(t) - R_1 s_1 \dot{\theta}_1(t) \right]$$

$$\dot{x}_V(t) = \frac{1}{2} \left[R_1 s_1 \dot{\theta}_1(t) + R_2 s_2 \dot{\theta}_2(t) \right] \cos \alpha_V(t)$$

$$\dot{y}_V(t) = \frac{1}{2} \left[R_1 s_1 \dot{\theta}_1(t) + R_2 s_2 \dot{\theta}_2(t) \right] \sin \alpha_V(t)$$

où : α_V : l'orientation du véhicule.

s_1 et s_2 représentent les facteurs de glissement,

R_1 et R_2 les rayons des roues motrices,

x_V, y_V : les coordonnées de l'origine du repère solidaire avec le véhicule par rapport à un repère inertiel

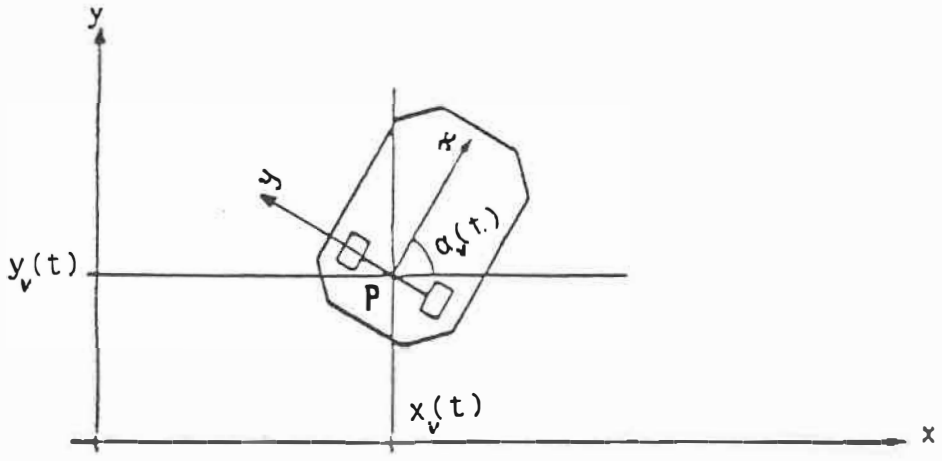


Figure 2.6 : Localisation du véhicule

2.4 Modélisation du système de mesure/estimation

La position et l'orientation du véhicule sont estimées à partir :

- des mesures de type inertiel effectuées à des intervalles de temps

$$T_{MES} = N.T$$

où : N est un nombre entier; T est la période d'échantillonnage correspondante à l'application des tensions de contrôle aux induits des servomoteurs. De telles mesures peuvent être réalisées par voie d'une caméra, système laser, sonars, balises infra-rouges etc ..

- des mesures de type odométrique effectuées à chaque période d'échantillonnage T. Les mesures sont fournies par des capteurs optiques montés sur l'axe des roues motrices.

Nous supposons que les mesures inertielles nous donnent directement les valeurs de x_v, y_v et α_v aux instants :

$$t = j.N.T, \qquad j = 1,2, \ldots$$

Les valeurs de x_v, y_v, α_v aux instants

$t = jNT + iT$, $i=1,2,3, \ldots, N-1$

sont obtenues à partir des mesures odométriques en utilisant les équations suivantes :

$$\alpha_v(jNT + iT) = \alpha_v(jNT) + \frac{1}{L} [R_2 \Delta\theta_2(iT) - R_1 \Delta\theta_1(iT)]$$

$$x_v(jNT + iT) = x_v(jNT+(i-1)T) + (T/2)*[R_1 \dot{\theta}_1(jNT+iT) + R_2 \dot{\theta}_2(jNT + iT)] \cos \alpha_v(jNT + iT)$$

$$y_v(jNT + iT) = y_v(jNT+(i-1)T) + (T/2)*[R_1 \dot{\theta}_1(jNT+iT) + R_2 \dot{\theta}_2(jNT + iT)] \sin \alpha_v(jNT + iT)$$

où :

$\Delta\theta_1(iT) = \theta_1 (jNT + iT) - \theta_1 (jNT)$

$\Delta\theta_2(iT) = \theta_2 (jNT + iT) - \theta_2 (jNT)$

La vitesse de translation du véhicule est estimée à partir des mesures odométriques fournies par les encodeurs optiques. En prenant égale à 1 les valeurs nominales des facteurs de glissement nous avons :

$$\text{VITESSE\_VEHI.} = \frac{1}{2} [R_1 \dot{\theta}_1 + R_2 \dot{\theta}_2]$$

La figure 2.7 montre la répartition des périodes dans le temps entre la mesure odométrique et la mesure inertielle.

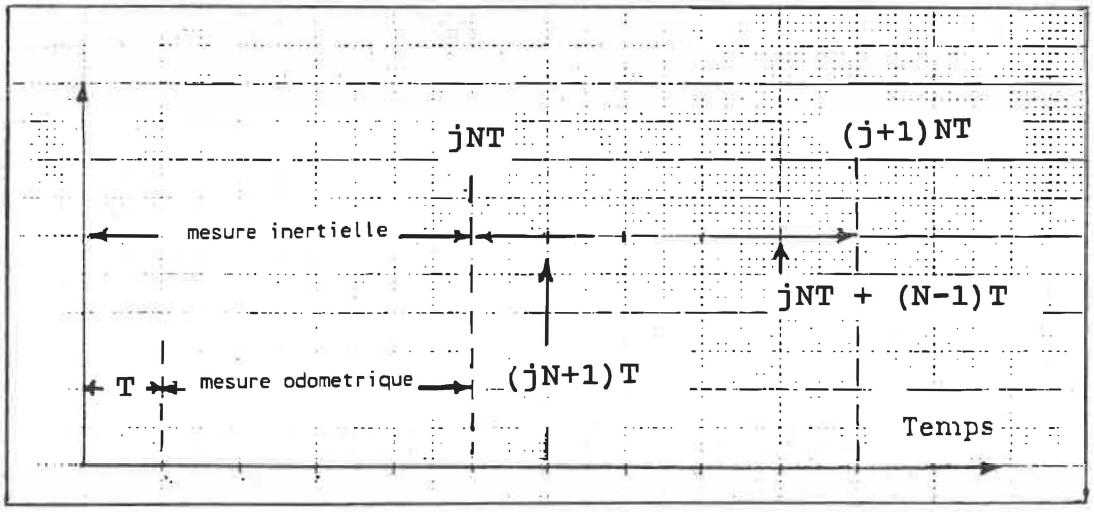


Figure 2.7 : Répartition des périodes de mesures

Le chemin à suivre est pensé comme etant constitué de segments de ligne droite. Les erreurs d'orientation et de vitesse de translation par rapport au chemin à suivre sont alors obtenues tout simplement à partir des équations :

$$\epsilon_\alpha = \alpha_C - \alpha(iNT + iT)$$

$$\epsilon_{vit} = \text{VITESSE\_DES} - \text{VITESSE\_VEHI}$$

où :

α_C est l'orientation du segment de droite du chemin le plus proche du véhicule;

VITESSE_DES est la vitesse désirée pour le véhicule.

L'erreur de position (latérale) est obtenue en utilisant l'équation

$$\epsilon_d = -(x_v - x_j) \sin \alpha_C + (y_v - y_j) \cos \alpha_C$$

où :

x_v, y_v représentent les coordonnées de l'origine du repère du véhicule,

x_j, y_j sont les coordonnées du point du segment les plus rapprochés du véhicule.

Il est facile de vérifier que les valeurs de x_j et y_j peuvent être calculées en utilisant les équations suivantes

$$x_j = \frac{x_v + a^2 x_2 + a(y_v - y_2)}{1 + a^2}$$

$$y_j = \frac{y_2 + a^2 y_v - a(x_2 - x_v)}{1 + a^2}$$

où :

$a = \text{tang } \alpha_v$

(x_1, y_1) et (x_2, y_2) sont les points extrêmes du segment de droite.

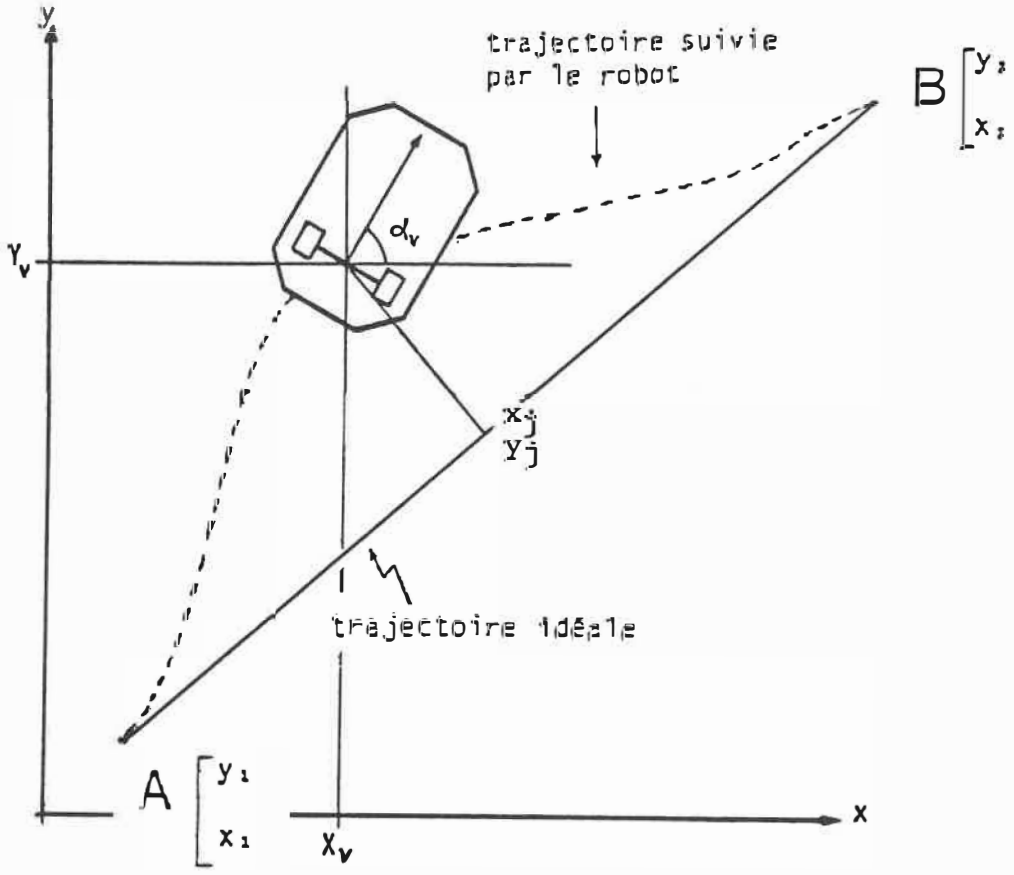


Figure 2.8 : Evaluation de l'erreur de position latérale ou de distance

2.5 Modélisation du système de contrôle

Le système de contrôle reçoit comme entrée les valeurs des erreurs de position, orientation et vitesse ϵ_d, ϵ_α et ϵ_{vit}; il fournit comme sortie les tensions U_1 et U_2 qui doivent être appliquées aux servomoteurs afin de réduire à zéro les valeurs de ces erreurs. Le système a une structure décentralisée constituée d'un contrôleur de position/orientation et d'un contrôleur de vitesse. En utilisant les notations :

$$U_1 = Um - \Delta U$$
$$U_2 = Um + \Delta U$$

le contrôleur de position/orientation calcule les valeurs ΔU en fonction des erreurs de position et d'orientation; le contrôleur de vitesse calcule la valeur de Um en fonction de l'erreur de vitesse.

2.5.1 Le contrôleur de position/orientation

Le contrôleur de position/orientation est du type PID (proportionnel, intégral, dérivé).

$$\Delta U = K_1.e + K_2 \frac{.de}{dt} + K_3 \int e.dt$$

où :

e représente une erreur "combinée" de position et d'orientation.

$$e \equiv (1-\beta)\ \epsilon_d + \beta.\ \epsilon_\alpha$$

avec β un coefficient de pondération à déterminer expérimentalement.

Les valeurs de K_1, K_2, K_3 peuvent être calculées en utilisant la technique du positionnement des pôles telle qu'expliquée dans la reférence [9]. Lors de la présence d'un délai, la valeur des gains peuvent être calculée d'une façon automatique en utilisant la méthode dans la référence [10]; le gain proportionnel a été détermine expérimentalement comme suit :

$$K_1 = \frac{C_1}{K} (\Theta/TAU)^{\alpha_1} + C$$

où :

K est le gain limite

Θ = délai

TAU : Constante de temps

$C_1 = 1.3$; $\alpha_1 = 0.8$; $C = 0.5$

La table de Cohen et Coon [9] permet ensuite de déduire les valeurs des gains K_2 et K_3 .

$$K_3/K_1 = \frac{1 + (0.6\ \Theta/TAU)}{2.5\ \Theta\ (1 + 0.2\ \Theta/TAU)}$$

$$K_2/K_1 = (0.37 * \Theta) / (1 + 0.2\ \Theta\ /TAU)$$

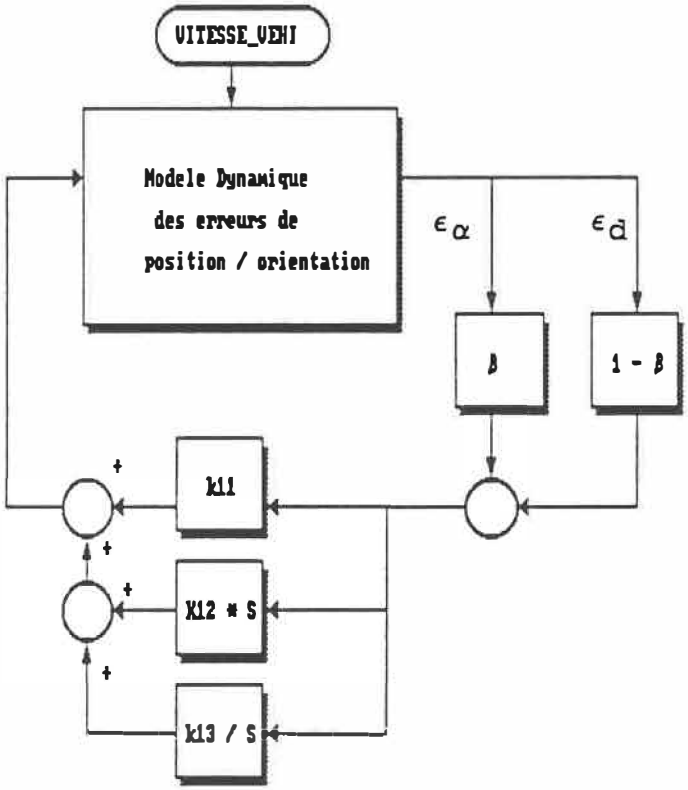


Figure 2.9 : Schéma du contrôleur de position/orientation

2.5.2 Le contrôleur de vitesse

Le contrôleur de vitesse est du type PI avec mode glissant. Pour une présentation complète de son opération et de conception nous renvoyons le lecteur a la référence [11]. Pour les objectifs du présent mémoire il suffit de se rappeler que le fonctionnement d'un tel contrôleur est régi par les équations suivantes (Figure 2.10)

$$Um = K_{21}\, e_v + K_{23} \left[\int e_v \, dt - K_{24} * VITESSE\_DES \right] + \Delta Um$$

où :

$$e_v = VITESSE\_DES - VITESSE\_VEHI$$

$$\Delta Um = - M * SIGN\ (slm)$$

la valeur de la constante M est choisie plus grande que la valeur maximale de la perturbation attendue; la fonction du mode glissant "slm" est choisie telle que

$$slm = \alpha_1 * VITESSE\_VEHI - \int e_v \, dt$$

En imposant le comportement dynamique décrit par la fonction de transfert

$$\frac{\text{VITESSE\_VEHI (s)}}{\text{VITESSE\_DES}} = -\frac{p_1 \cdot p_2}{z}\,\frac{(s - z)}{(s-p_1)\,(s-p_2)}$$

avec $z = p_2$

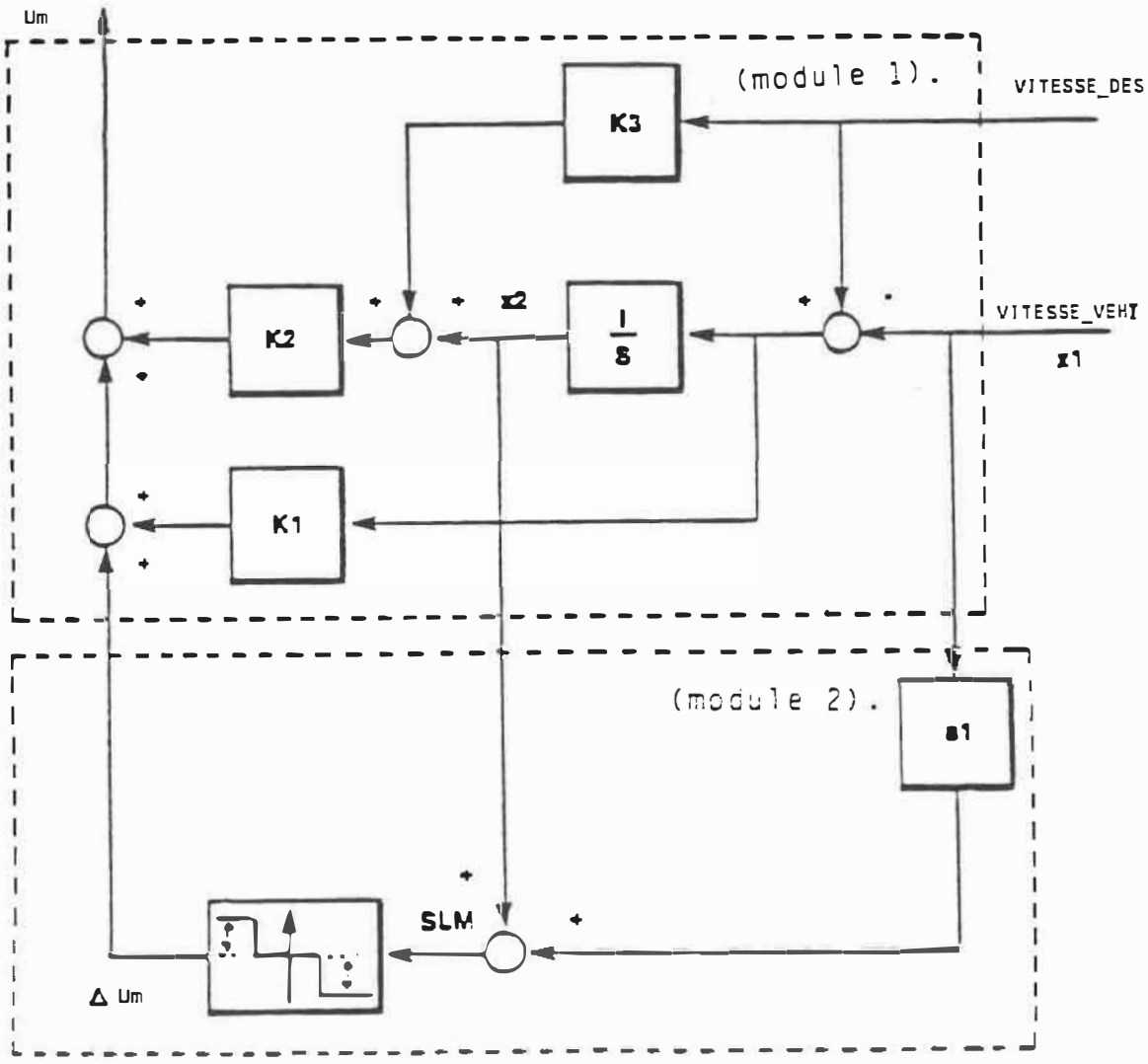
les valeurs de gain de ce contrôleur sont calculées en utilisant les formules

$$k_{21} = -((p_1+p_2) * \tau_{nom} + 1) / KM_{nom})$$
$$k_{23} = p_1 * p_2 * \tau_{nom} / KM_{nom}$$
$$k_{24} = (1+1 / (\tau * p_2)) / p_1$$

où

$\alpha_1 = -1/p_1$

KM_{nom} et τ_{nom} sont les valeurs nominales de τ et KM



Module 1: module PI classique

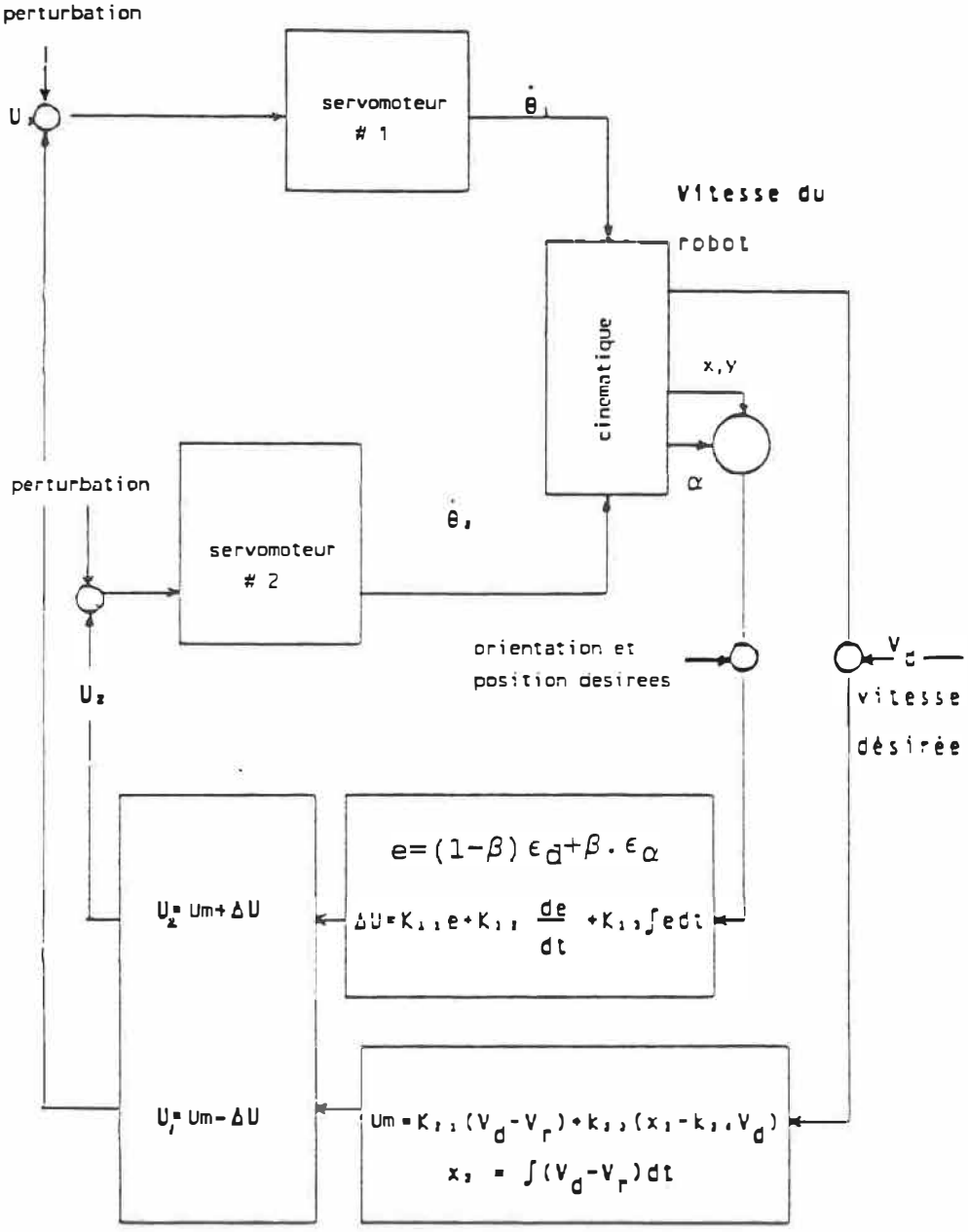
Module 2: module inverseur ou "switching component"
(SLM = sliding mode)

Figure 2.10 : Schéma du contrôleur PI

2.6 Modélisation de l'ensemble du système

Le modèle pour l'ensemble de notre système est obtenu en connectant d'une façon appropriée les divers modèles vus dans les sections précédentes (figure 2.11).

Nous remarquons principalement les deux modèles de contrôleurs connectés à la sortie du système de locomotion. A ceci s'ajoute le module de calcul propre aux contrôleurs qui donne les valeurs des tensions U_1, U_2.



U_1 : commande moteur droit V_d : vitesse désirée

U_2 : commande moteur gauche V_r : vitesse réelle

Figure 2.11 : Modélisation de l'ensemble du système

CHAPITRE 3

DESCRIPTION DU LOGICIEL DE SIMULATION

3.1 Introduction

L'objectif du logiciel **SRM** (Simulation Robot Mobile) est de simuler le comportement dynamique du robot mobile modélisé au chapitre 2 sous une diversité de conditions d'opérations :

- conditions initiales
- valeurs de gains de contrôleurs
- paramètres du module de locomotion
- modalités de mesure
- présence de perturbations extérieures
  et des variations des paramètres internes.
- présence de délai
- chemin à suivre.

Les principales caractéristiques du logiciel résident dans :

- **la structure interactive**
- **la gestion des fichiers au niveau des commutations des données**
- **la création d'une base de données de toute la simulation selon le choix de l'usager.**
- **la technique infographique.**

La structure du logiciel est constituée d'un module principal inter-connecté avec un certain nombres de sous-modules dont les contenus seront présentés dans la section de la structure interne. Parmi les plus importants, citons le module d'intégration numérique et celui de simulation de la trajectoire.

L'élaboration du logiciel suit le standard modulaire par menu utilisé par les collègues qui m'ont précédé dans la préparation de leur thèse de Maîtrise: Caroline Serfass [30], Anh-Tu Nguyen [31], Padnakhone Souphandavong [32].

Dans la section 3.2 nous présentons le logiciel tel que vu par l'utilisateur et dans la section 3.3 la structure interne avec la conception modulaire.

3.2 Structure du logiciel vu par l'utilisateur

La succession chronologique des différentes étapes rencontrées par l'utilisateur lors de l'éxécution du programme SRM est représenté par le schéma bloc à la figure 3.1 . En ce qui concerne le choix des fonctions ce schéma bloc est traduit sous forme de menu au tableau 3.1. Guidé par les directives interactives du progiciel, l'utilisateur n'a qu'à donner sa réponse aux choix offerts par les menus pour faire executer les opérations désirées.

3.2.1 Description du système

Ce module fait appel à un fichier texte externe au logiciel dans lequel se trouve une brève description du système d'autopilotage avec les équations dynamiques et cinématiques suivies de la description des variables.

3.2.1 Répertoire des fichiers

Ce module fonctionne comme la commande "Directory" du DOS ou "List" de UNIX; La liste des fichiers peut s'afficher par groupe d'extension (*.PAS) ou par ordre selon leur date de création ou tout simplement l'affichage de la liste complète .

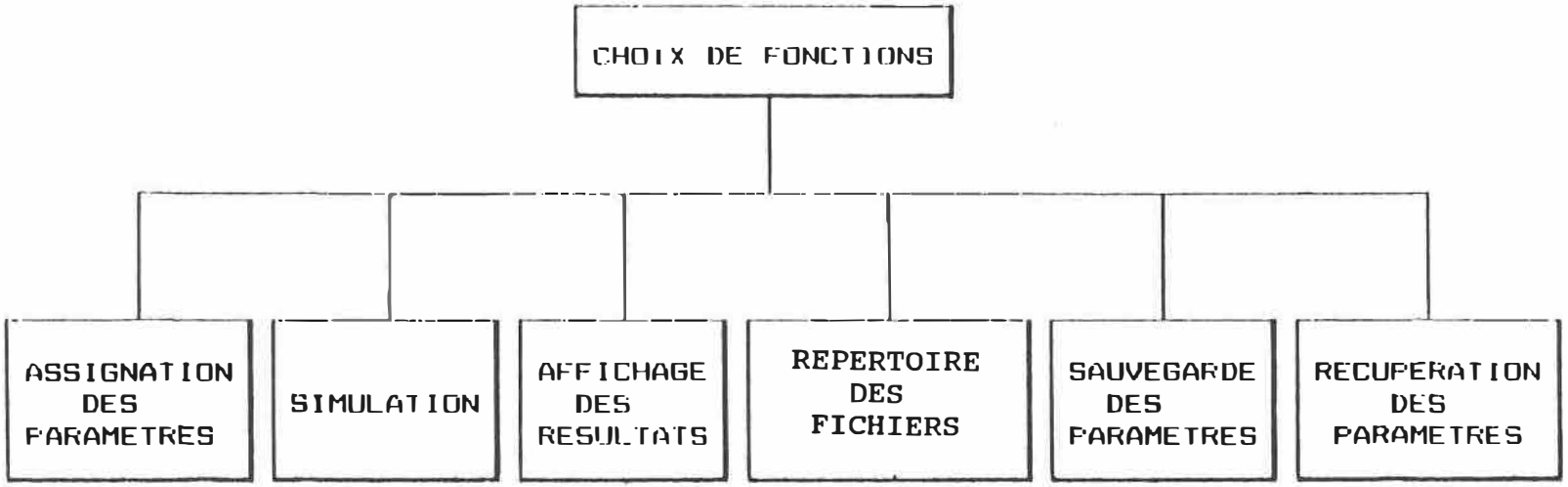


Figure 3.1 : Schéma bloc vu par l'utilisateur

MENU PRINCIPAL

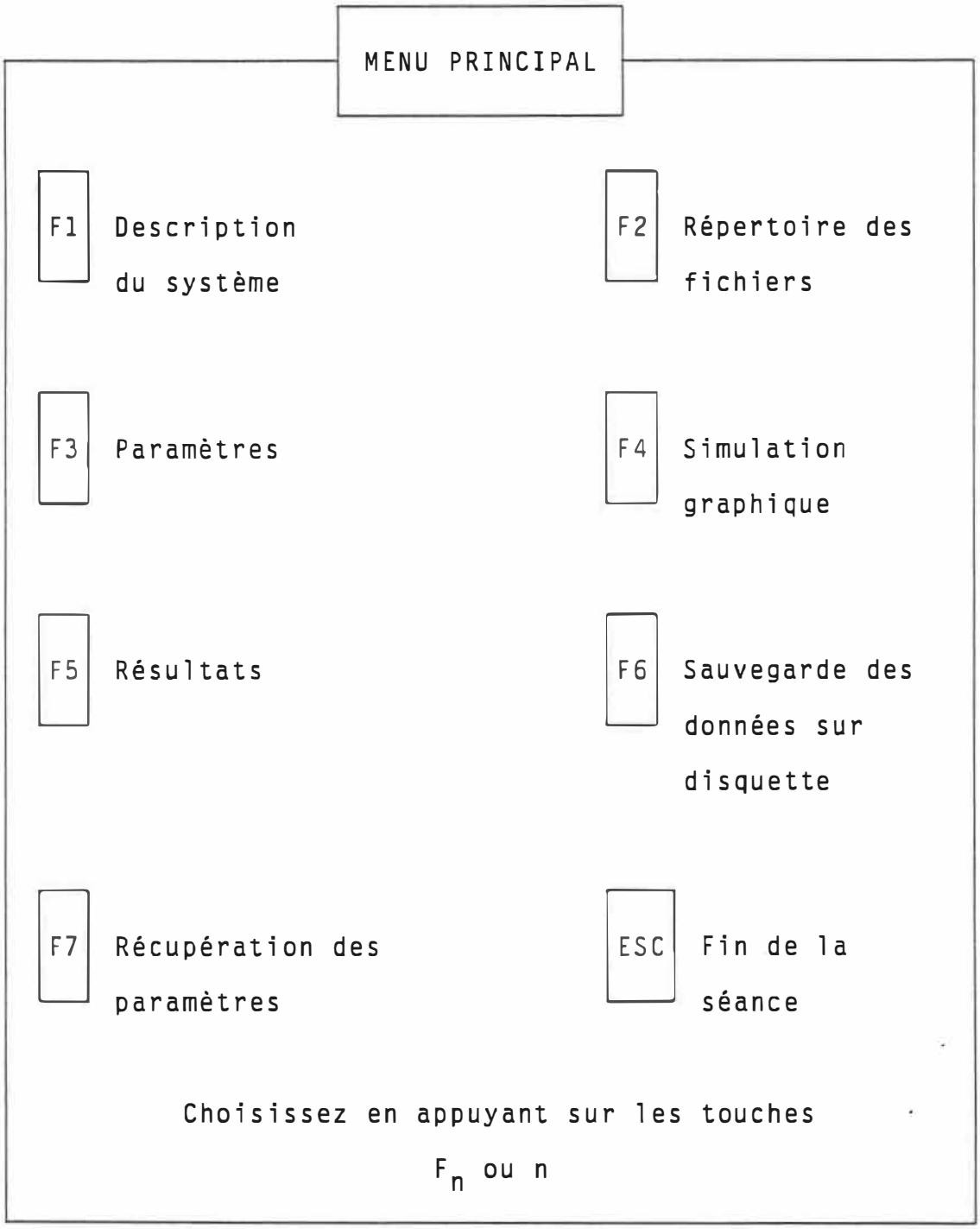


Tableau 3.1: Menu Principal

EXEMPLE DE MENU DE PARAMÈTRES

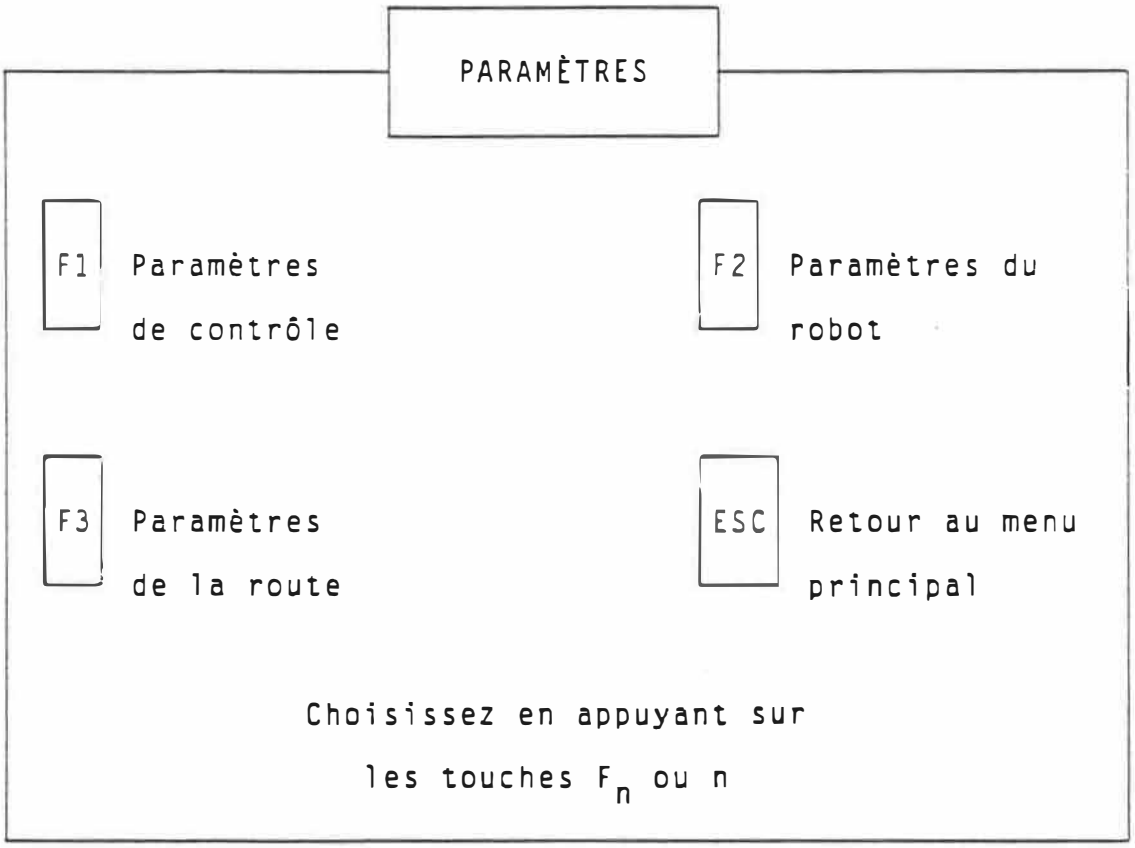


Tableau 3.2: Menu des paramètres

3.2.3 Les Paramètres

A priori, ce module prend les valeurs de paramètres préprogrammées lors de l'initialisation par le module pricipal. Ensuite il se connecte aux différents menus pour enregistrer les valeurs entrées par l'utilisateur selon :

* les paramètres de la route
* les paramètres du robot
* les paramètres de simulation
* les paramètres de contrôle.

Si la valeur enregistrée n'est pas conforme à la marge de valeur indiquée le sous-module se connecte directement à un module utilitaire qui active le "Errorsound" à l'intérieur de l'ordinateur. Après enregistrement des valeurs le contrôle est retourné au module principal pour les étapes subséquentes.

3.2.4 Simulation graphigue

Lors de la simulation, la route est affichée à l'écran sous forme de segments de droite représentant l'axe de la route. Le robot est symbolisé sous forme de fléche qui représente sa direction et la trace du robot est affichée avec une couleur différente de celle de la route pour pouvoir examiner les différentes déviations (position, orientation) lors du déplacement.

3.2.5 Présentation des Résultats

L'interprétation des résultats se fait à l'aide de trois différentes courbes relatives à chaque simulation. Ces courbes représentent les variables contrôlées en fonction du chemin parcouru.

- Erreurs de distance vs distance
- Erreurs d'orientation vs distance
- Erreurs de vitesse vs distance

L'échelle des courbes s'ajuste automatiquement selon la longueur maximum de la route choisie par l'utilisateur. En conséquence toutes les courbes sont tracées à plein écran pour permettre une meilleure visualisation et interprétation. Les données relatives à ces courbes peuvent être également enregistrées sous forme de fichier ASCII et mémorisées sur disque. Cela permet une analyse des résultats au moyen des possibilités de simulation fournies par le progiciel MATLAB.

3.2.6 Récupération et Sauvegarde des paramètres

Le module "Récupération" sert à l'assignation en mémoire de certaines valeurs de paramètres relatives à des simulations faites auparavant et stockées dans un fichier. Ainsi si on veut visualiser le comportement du robot et les courbes de simulation selon des valeurs spécifiques qui ont déja faites l'objet d'une expérience, il suffit de faire la récupération des paramétres via la touche F7 au lieu de rentrer à nouveau ces valeurs et calculer les nouveaux coefficients de gains.

L'action inverse de la récupération est la "sauvegarde" via la touche F6. L'action de sauvegarde consiste à sauvegarder les paramètres de la route, du robot, de la simulation et ceux du contrôle.

EXEMPLE DE MENU POUR LA SAUVEGARDE DES DONNÉES

Ceci est obtenu via la clé F7 du menu principal après avoir fait la simulation.

Paramètres de contrôle		Paramètres de la route
Vitesse	=	
TAU1,TAU2	=	X1, Y1 =
Delai1,Delai2	=	X2, Y2 =
Pertb1,Pertb2	=	X3, Y3 =
PID		
K1,K2,K3	=	
PI		
K1,K3,K4	=	

Paramètres du Robot		Paramètres de Simulation
largeur	=	δT =
R1,R2	=	dC, dV =
S1,S2	=	M , β =

SAUVEGARDE DES DONNEES [O/N] ?

Tableau 3.3: Récapitulation des paramètres

3.3 Stucture interne du logiciel

Le logiciel SRM a été structuré pour respecter la conception modulaire et la conception "top-down design analysis".

L'organigramme à la figure 3.2 montre la suite logique des calculs tandis que le schéma bloc des sous-modules montre les liens hiérarchiques à la figure 3.3. Nous montrerons dans cette section quelques images qui schématisent la représentation physique du calcul effectué par la sous-routine. Nous présenterons également quelques fonctions spécifiques associées aux systèmes modélisés dans le chapitre 2.

3.3.1 SRMMAIN : Module Principal

Le module principal initialise tous les menus présentés à la section précédente c'est-à-dire tous les variables identifiant les paramètres et ensuite initialise le tampon mémoire pour la simulation graphique et les courbes de résultats. Chaque sous-routine fait ensuite ses calculs appropriés dependamment du choix de l'utilisateur par l'intermédiaire des menus. Chaque menu se termine par le caractère <ESC> lequel fait appel à une routine utilitaire "Whistlesound" pour activer un son.

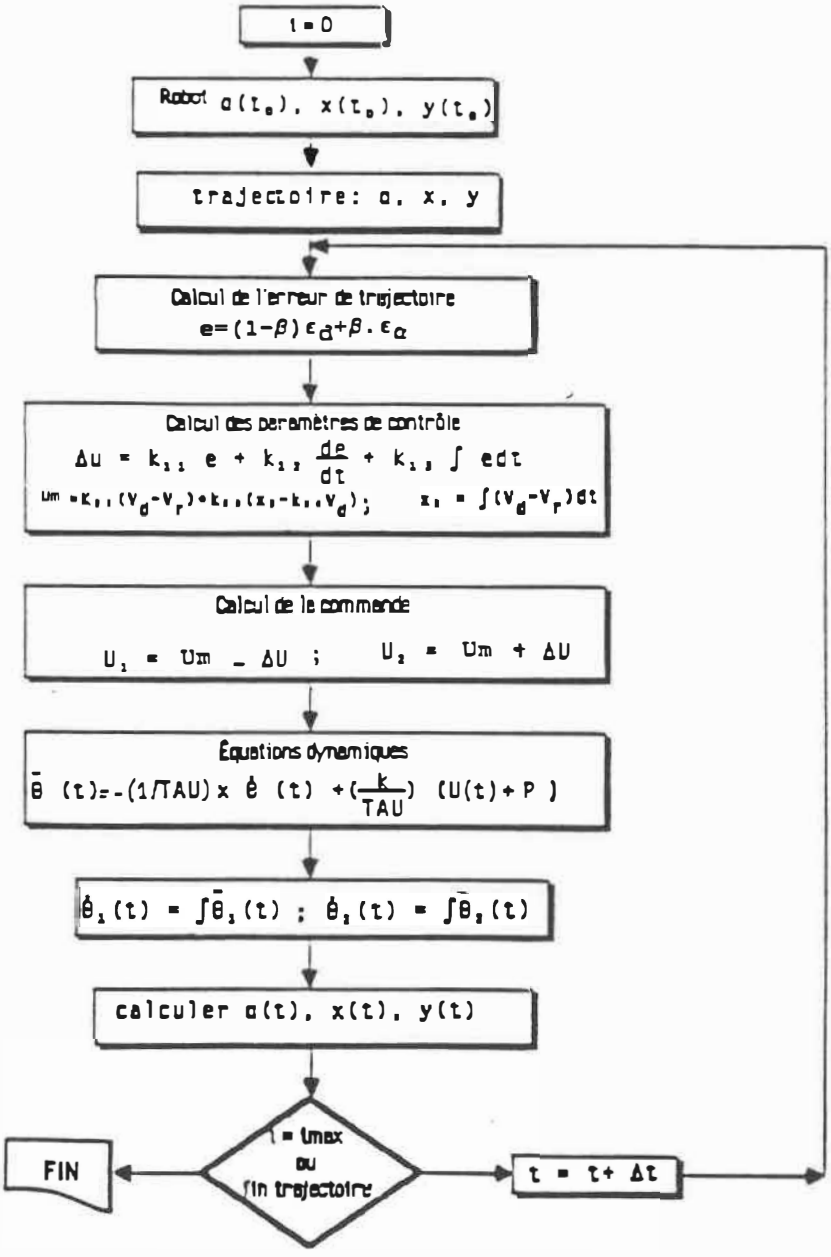


Figure 3.2 : ORGANIGRAMME

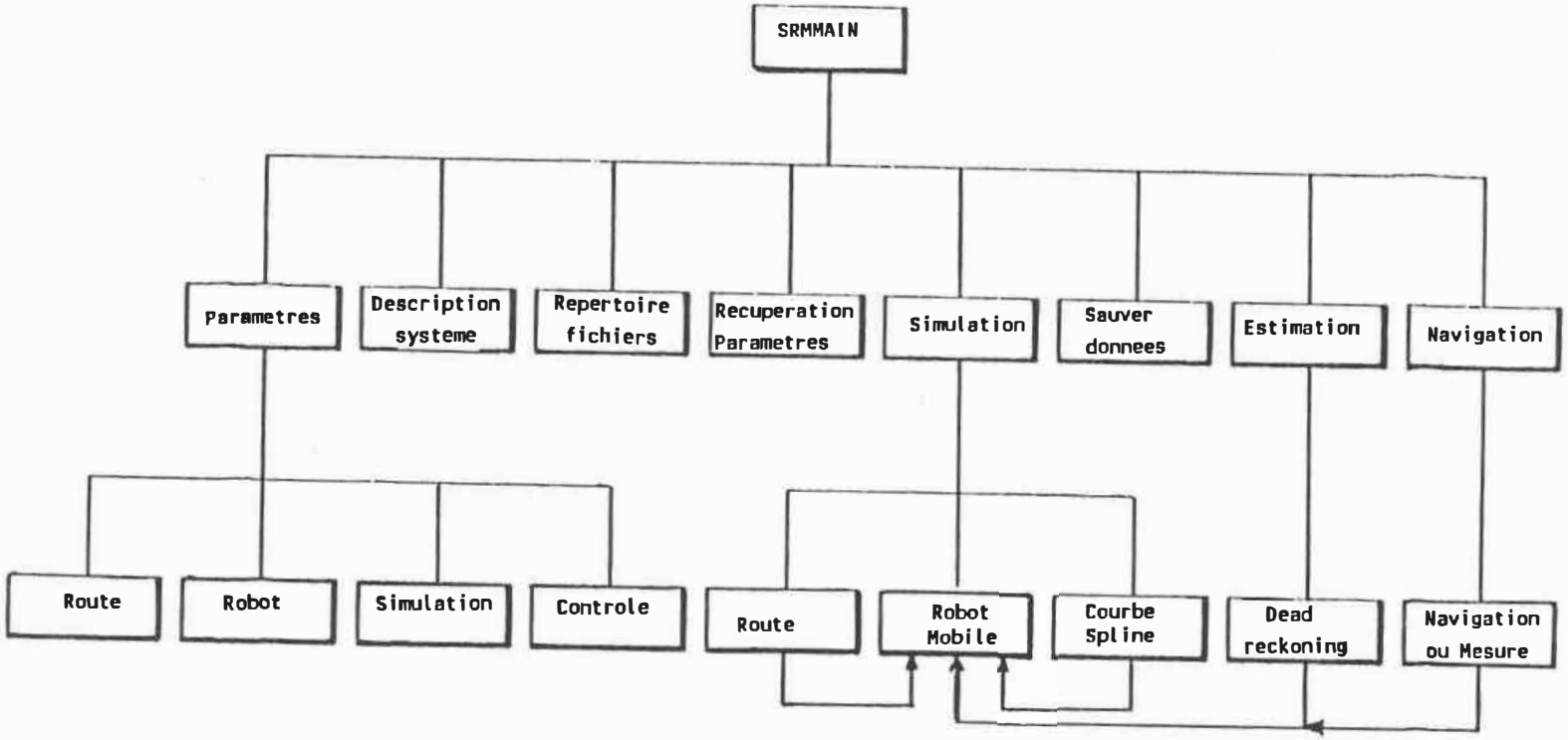


Figure 3.3 : Schéma bloc de la structure interne du logiciel

3.3.2 Paramètres

Dans ce paragraphe nous présentons les détails internes relatifs aux paramètres identifés dans le schéma bloc à la figure 3.3

```
                    PARAMETRES DE SIMULATION

     Coeff. d'erreur d'orientation: 0 ≤ Beta < 1   Beta = 0.50

     Coeff. du 'switch component'   : 0 ≤ M        M    = 1.00

     Periode d'echantillonnage      : 0 < dt(sec)  dt   = 0.020

     Per.de visualisation = dt * Fv : 0 < Fv       Fv   = 2

                Parametres pour le dead - reckoning

     Periode des capteurs           : 0 < Pc(sec) Pc    = 0.200

     Largeur estimee du robot       : 0 < Le( m ) Le    = 0.650

                Voulez-vous changer ces valeurs ? (O/N)
```

```
                      LA TRAJECTOIRE A SUIVRE

        Note :  Le robot va de (X[i],Y[i]) -> (X[i+1],Y[i+1])
                Il faut que  X[i] < X[i+1]
                et  X[i] <  30.000  Y[i] <  30.000

                      X1 = 1.000          Y1 = 1.000

                      X2 = 10.000         Y2 = 10.000

                      X3 = 25.000         Y3 = 25.000


NOTE : Si vous voulez creer un fichier MATLAB, Repondez "OUI" a la
       question ci-dessous et Retapez vos points meme s"il s"agit
       des memes coordonnees... afin de reinitialiser le buffer

                Voulez-vous Modifier les points ? (O/N)
```

PRESENTATION DES PARAMETRES DE CONTROLE

<1>	Vitesse desiree	0 < Vd <= 10 (m/s)	Vd = 2.50
	Constante de temps :		
<2>	Moteur gauche	0.1 < TAU1	TAU1 = 0.50
<3>	Moteur droite	0.1 < TAU2	TAU2 = 0.50
	Gain statique :		
<4>	Moteur gauche	0 < KM1	KM1 = 12.00
<5>	Moteur droite	0 < KM2	KM2 = 12.00
	Delai :		
<6>	Moteur gauche	0<= DL1 <=100 msec	DL1 = 100.00
<7>	Moteur droite	0<= DL2 <=100 msec	DL2 = 100.00
	Perturbation :		
<8>	Moteur gauche	0 < PM1 <= 2000 mv	PM1 = 50.00
<9>	Moteur droite	0 < PM2 <= 2000 mv	PM2 = 50.00

Tapez le numero correspondant pour changer les valeurs ou
<ESC> pour terminer.

->

Les parametres du robot

Le facteur de glissement de :			
<1>	La roue gauche	0 < S1 <= 0.95	S1 = 0.92
<2>	La roue droite	0 < S2 <= 0.95	S2 = 0.92
Le rayon de :			
<3>	La roue gauche	0 < R1 <= 0.20 m	R1 = 0.15m
<4>	La roue droite	0 < R2 <= 0.20 m	R2 = 0.15m
<5>	Sa largeur	0 < La <= 0.50 m	La = 0.50m
<6>	Sa longueur	0 < Lo <= 1.0 m	Lo = 0.90m
<ESC>	Pour terminer		

Tapez le numero correspondant pour changer les valeurs.

Tableau 3.4 : Tableau des Paramètres.

3.3.3 Estimation et Navigation

Ces deux sous-modules effectuent les phases de calculs associées aux modules dead-reckoning et Navigation (ou Mesure) que nous avons modélisés dansa le système d'estimation.

L'estimation de position se fait à l'aide des deux fonctions ci-dessous durant la période de mesure odométrique. Une autre fonction analogue calcule l'estimation pour l'orientation.

```
Function Pos_Xc (r1,r2,L,s1,s2   : Real;
                 Previous_Pos_Xc : Real;
                 Orient_robot    : vecteur;
                 Vit             : couple ) :Real;

Var Vit_trans : Real; { vitesse de translation }
Begin
  Vit_trans:= (1/2)*((r1 * s1 * vit[1][tn]) + (r1 * s2 * vit[2][tn]) );
  Pos_Xc   := Previous_Pos_Xc + (Vit_trans * cos(orient_robot[tn]) * dt);
End;

Function Pos_Yc (r1,r2,L,s1,s2   : Real;
                 Previous_Pos_Yc : Real;
                 Orient_robot    : vecteur;
                 Vit             : couple ) :Real;

Var Vit_trans : Real; { vitesse de translation }
Begin
  Vit_trans:= (1/2)*((r1 * s1 * vit[1][tn]) + (r1 * s2 * vit[2][tn]) );
  Pos_Yc   := Previous_Pos_Yc + (Vit_trans * sin(orient_robot[tn]) * dt);
End;
```

Les deux sous-modules utilisent les données communiquées par les fonctions d'intégration ci-dessous.

```
( Calcul de la vitesse de la roue gauche )
Function Dynamique_1( X12, tau1, km1, u1 : Real ) : Real ;
Var
 Per_ech: Real;
Begin
  Per_ech  := delta_t;
  F[2,2,1] := Exp (- Per_ech / Tau1 );
  B[2,1]   := km1 * ( 1- F[2,2,1]);
  Dynamique_1 :=  F[2,2,1] * X12 + B[2,1] * U1;
End ;  ( Output = teta_pt[1]  i.e : X[1,2]  )

( Calcul de la vitesse de la roue droite )
Function Dynamique_2( X22, tau2, km2, u2 : Real ) : Real ;
Var
 Per_ech: Real;
Begin
  Per_ech  := delta_t;
  F[2,2,2] := Exp (- Per_ech / Tau2 );
  B[2,2]   := km2 * ( 1- F[2,2,2]);
  Dynamique_2 :=  F[2,2,2] * X22 + B[2,2] * U2;
End ;  ( Output = teta_pt[2]  i.e : X[2,2]  )
```

3.3.4 Simulation infographique

Ce module effectue la simulation qui permet de visualiser en temps réel le comportement dynamique du robot. Pour ce faire, le véhicule est représenté dans un plan (x,y) par une flèche. Ainsi le déplacement de la flèche simule les séquences du suivi de la route.

La trace du véhicule est représentée à l'écran avec une couleur différente de celle de la route. Ce qui permet d'observer :

a) les erreurs de position latérale (ou de distance)
b) les erreurs d'orientation

La figure 3.4 illustre la simulation infographique. Pour plus de détails et de précision les erreurs (a) et (b) sont affichées ensuite sous forme de courbes (vs Temps et vs distance: X) dans le module " Résultat " .

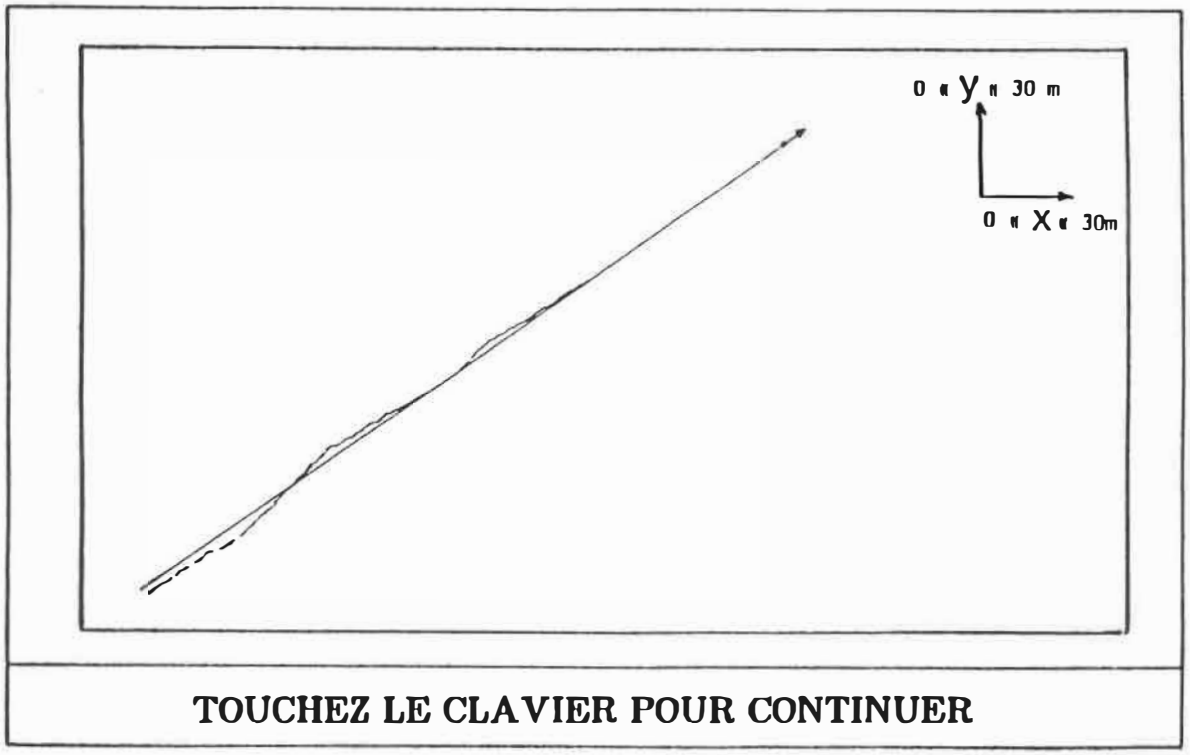


Figure 3.4 : Simulation Infographique

CHAPITRE 4

ANALYSE DU SYSTÈME

4.1 Introduction

Dans ce chapître nous exploitons le logiciel présenté au chapître 3 afin d'analyser le comportement dynamique du système d'autopilotage présenté au chapître 2.

Pour ce faire nous exécutons un certains nombres de tests permettant d'illustrer le comportement du système lors des conditions d'opérations idéales et ensuite lors des conditions d'opérations se rapprochant le plus possible du cas réel en variant les paramètres de simulation, les paramètres de contrôle et les paramètres caractéristiques du véhicule.

Ainsi l'analyse consiste à observer les effets des différentes erreurs appliquées à la position du robot et en déduire une conclusion concernant la stabilité du système et la réponse des contrôleurs.

4.2 Performance dynamique du système

Les différents tests que nous allons présenter ont été effectués avec une trajectoire en ligne droite (figure 4.1); le logiciel permet de simuler des trajectoires en ligne droite ou en lignes courbes telles que présentées dans l'appendice A.

La position de départ du véhicule peut être décalé de l'axe de la route avec différentes orientations selon le cas de l'étude. La figure 4.2 illustre le cas réel de la position du véhicule avec une vue du concept de la trajectoire prédéfinie qui est le chemin à suivre par le robot ou véhicule mobile; l'orientation du véhicule par rapport à l'horizontal est représentée par une flèche.

Les courbes de simulations qui vont suivre sont caractérisées par les conditions suivantes :

Réponse sous des conditions nominales avec une erreur de position/orientation initiale;

Réponse sous des conditions nominales en correspondance avec diverses erreurs de position initiale;

Réponse sous des conditions nominales en correspondance avec diverses valeurs de la vitesse désirée;

Réponse sous des conditions nominales en correspondance avec diverses valeurs du facteur d'erreur combinée β (vitesse désirée = 2.5 m/s);

Réponse sous des conditions nominales en correspondance avec diverses valeurs du facteur d'erreur combinée β (vitesse désirée = 10 m/s);

Influence d'une asymétrie au niveau de la constante de temps des servomoteurs;

Influence d'une asymétrie au niveau du gain statique des servomoteurs;

Influence de la présence d'un délai;

Influence de l'intervalle de temps qui sépare deux mesures inertielles lors d'une bonne précision dans les mesures odométriques;

Influence de l'intervalle de temps qui sépare deux mesures inertielles lors d'une faible précision dans les mesures odométriques.

REMARQUE

Dans toutes les simulations qui vont suivre les echelles utilisées pour les courbes sont :

Temps : en seconde

Errreur de vitesse : en mètre/sec

Erreur d'orientation : en degré

Erreur de position latérale ou de distance : mètre

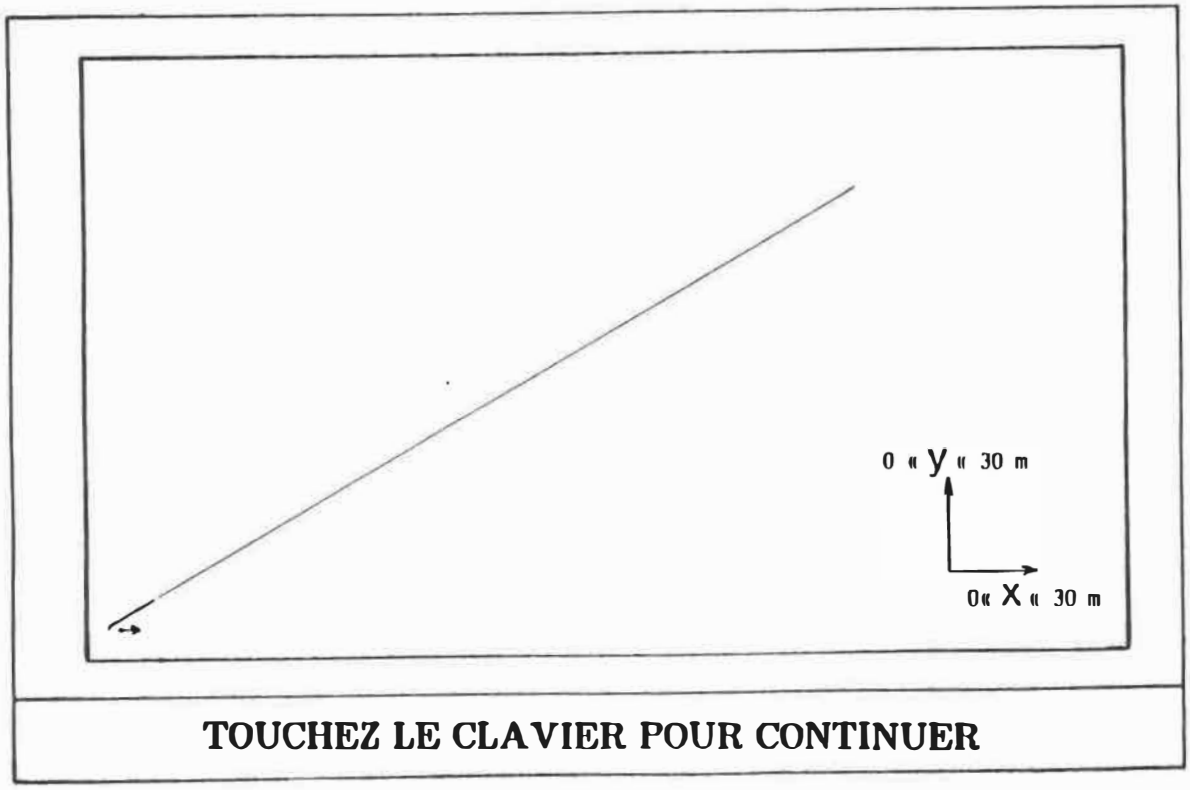


Figure 4.1 : trajectoire de simulation

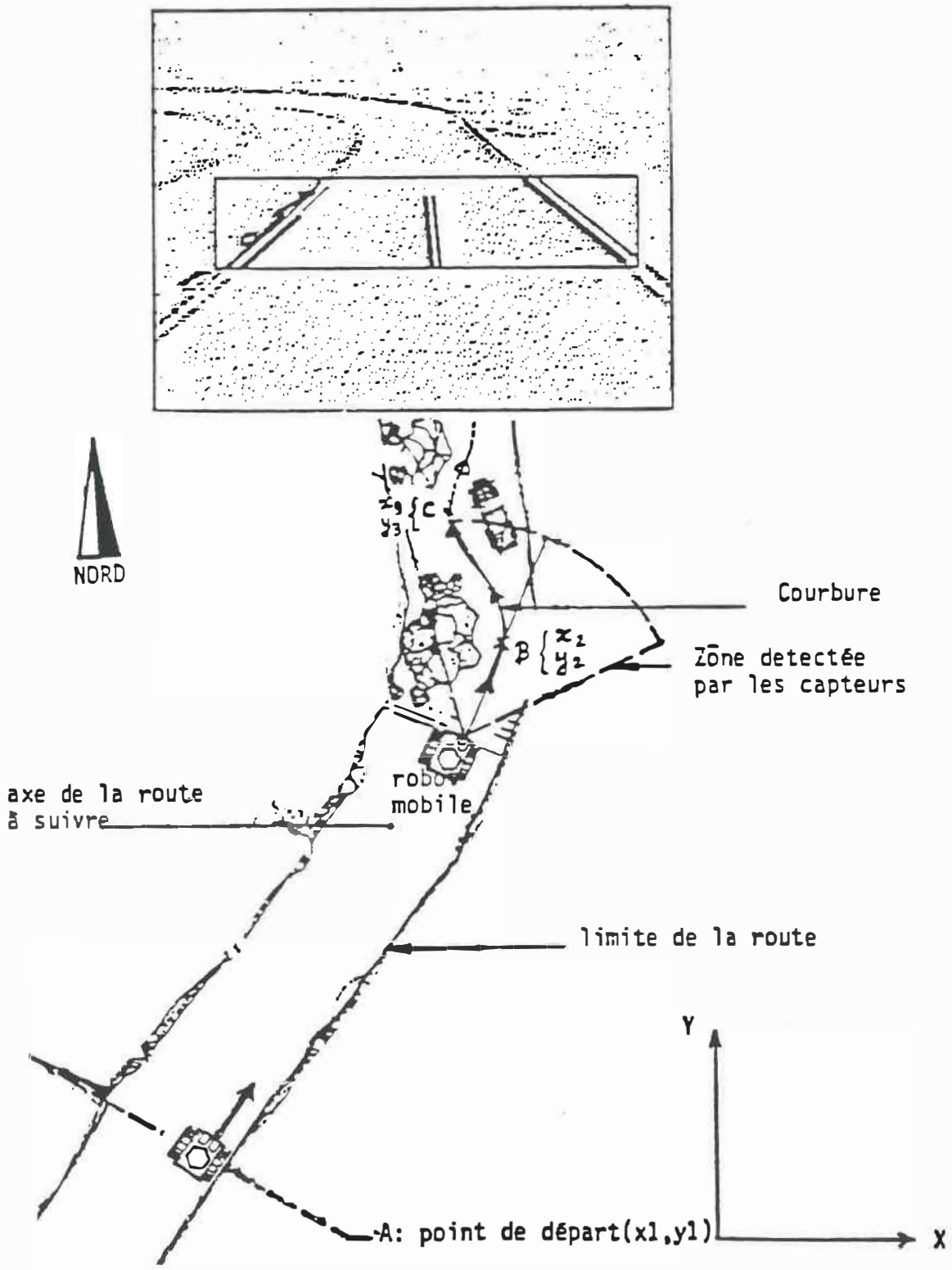


Figure 4.2 : Vue du concept de la trajectoire prédéfinie

EXPERIENCE No 1 : COMPORTEMENT DU SYSTEME SOUS DES CONDITIONS NOMINALES

Objectif : Illustrer le comportement dynamique du système sous des conditions d'opérations nominales

Modalités de simulation : Les paramètres du système ont les valeurs nominales indiquées au tableau 4.1. Le chemin à suivre est représenté par une ligne droite; une erreur de position et d'orientation initiales sont présentes; un échelon de vitesse est appliqué. Les gains du système sont déterminés en utilisant la technique du positionnement des pôles; les paramètres du véhicule sont supposés avoir une valeur symétrique; l'estimée de cette valeur est disponible avec une grande précision.

Résultats : Les courbes à la figure 4.7 a-b-c montrent l'évolution de la vitesse et de l'erreur de position et d'orientation du véhicule en fonction du temps.

Discussion : La Réponse dynamique du système est caractérisée par des valeurs de constantes de temps suggérées par une analyse fondé sur le modèle continu dans le temps, linéaire et découplé discuté au chapître 2. La précision, le temps de réponse et le comportement dynamique sont bien satisfaisants par rapport aux applications normalement envisagées pour le véhicule mobile.

CONDITIONS NOMINALES d'OPERATION

GAIN DU CONTROLEUR
Position/Orientation : K11 = 6.5; K12 = 7.0; K13 = 0.0; M1= 0;
Vitesse desiree : K21 = 10 ; K23 = 20 ; K24 = 0.0; M2= 0;
Facteur d'erreur combinée β = 0.7

SERVOMOTEUR
Constante de temps : TAU1 = 0.5; TAU2 = 0.5;
Gain statique : KM1 = 0.8; KM2 = 0.8;
Delai : Delai1= 0.0; Delai2= 0.0;
Perturbation Equiv. : Pert1 = 0.0; Pert2 = 0.0;

VEHICULE MOBILE
Fact.glissement : s1=1; s2=1; Rayon1=0.12; Rayon2=0.12; largeur = 0.5;

CONDITIONS DE SIMULATION
Err.position Lat_offset=0.5; Orient_offset = 15;
Echantillonnage Commande =0.02; Mesur.inertielle=0.2;
Larg.vehicule largeur_estimee= 0.5;

Tableau 4.1 : Tableau des valeurs nominales

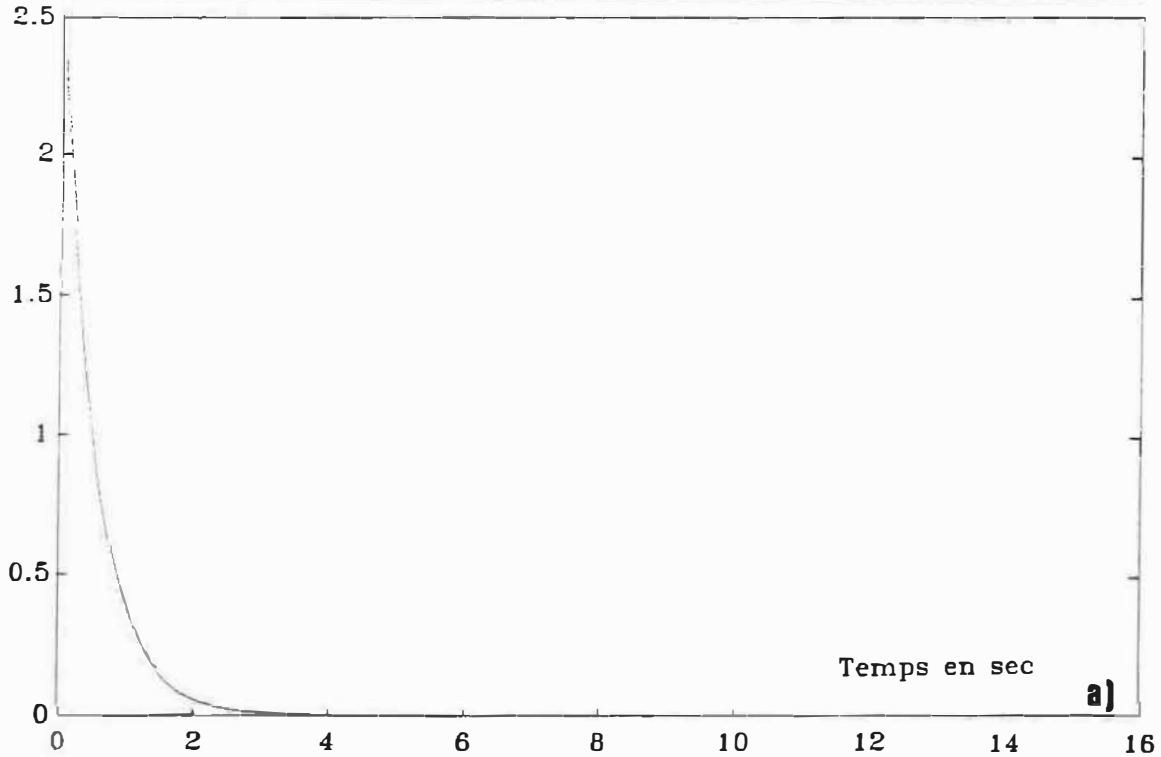


Figure 4.7a: Reponse sous des conditions nominales
(Erreur de vitesse vs Temps)

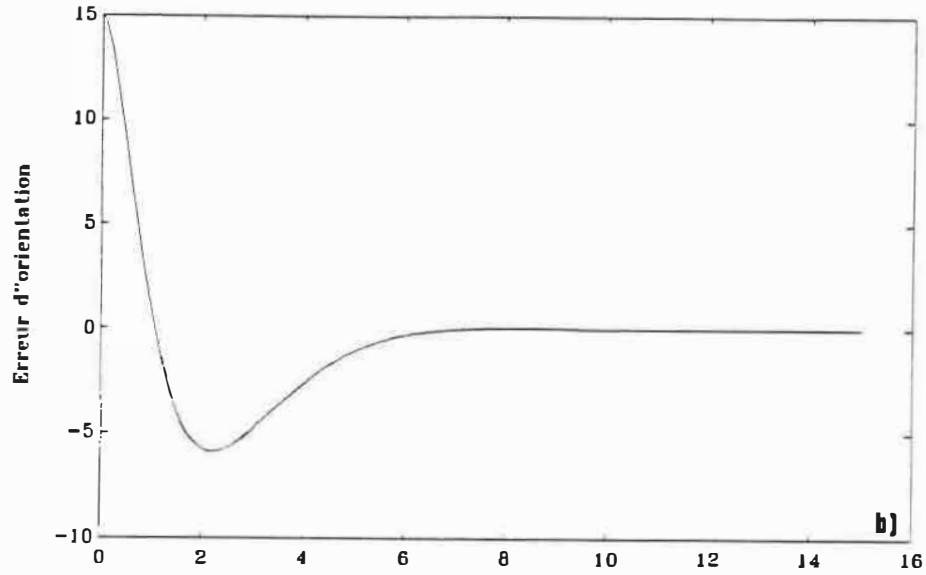


Figure 4.7b: Reponse sous des conditions nominales
(Erreur d'orientation vs Temps)

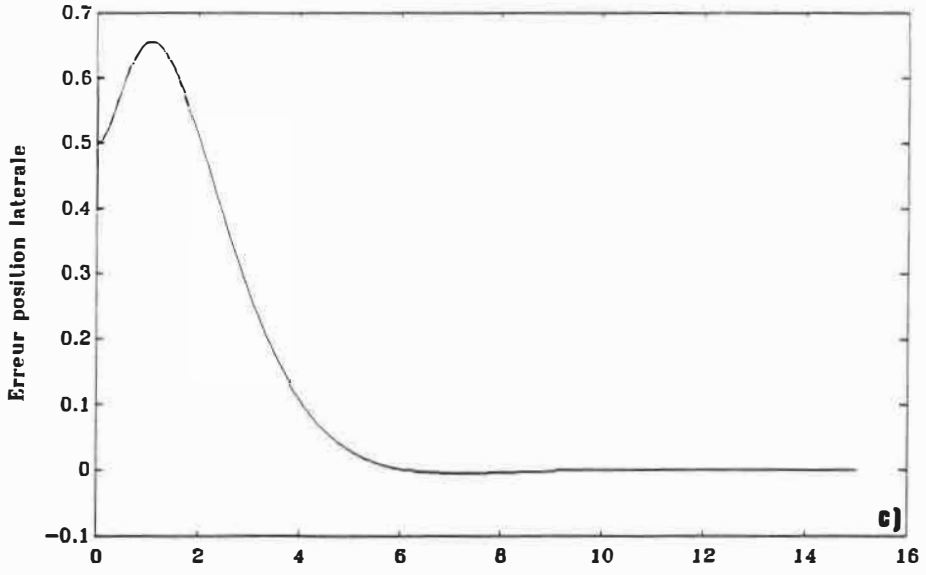


Figure 4.7c: Reponse sous des conditions nominales
(Erreur de position vs Temps)

EXPERIENCE No 2 : INFLUENCE DES CONDITIONS INITIALES

Objectif: Illustrer le comportement dynamique du système sous des conditions d'opérations nominales en correspondance avec diverses conditions initiales.

Modalités de simulation : Identiques à celles qui caractérisent l'expérience No 1 avec la variante que diverses valeurs de conditions initiales relatives aux erreurs de position sont maintenant considérées. En particulier :

Lat_offset : 0.25, 50, 100, 200 cm
Orientation_offset : 15, 45, 60 degrés

Résultats : Les courbes à la figure 4.8 montrent l'évolution de la vitesse et de l'erreur de position et d'orientation du véhicule en fonction du temps et des diverses conditions initiales.

Discussion : Lors d'erreurs de position initiale associées à une erreur d'orientation inférieure à 60 degrés le comportement dynamique du système semble bien respecter la réponse anticipée; en correspondance avec des erreurs d'orientation plus grandes l'influence des non-linéarités du système se manifeste et le comportement transitoire du système devient plus oscillant (Figure 4.8). Les diverses variations d'erreurs initiales n'influencent pas la vitesse.

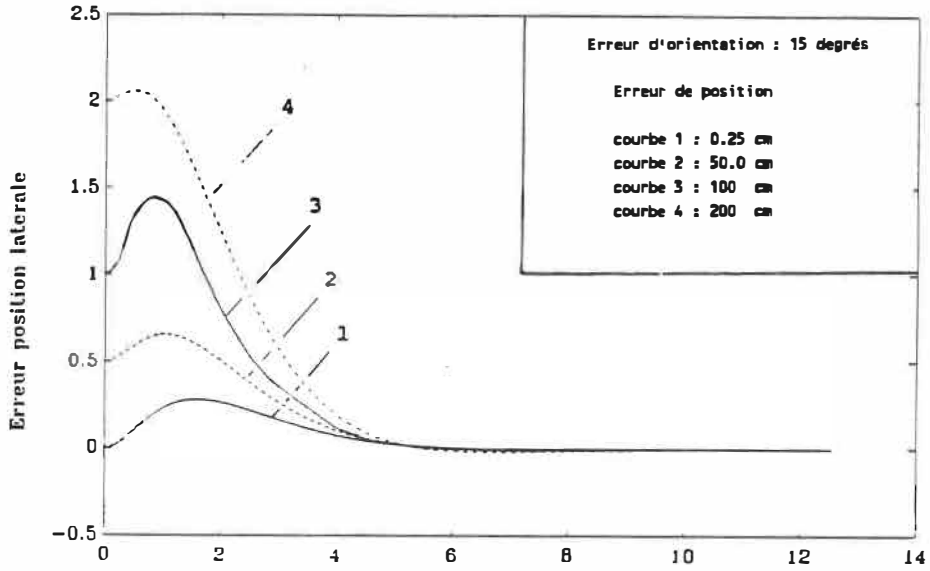


Figure 4.8a: Reponse sous des conditions nominales en correspondance avec diverses erreurs de position initiales (Erreur de position vs Temps)

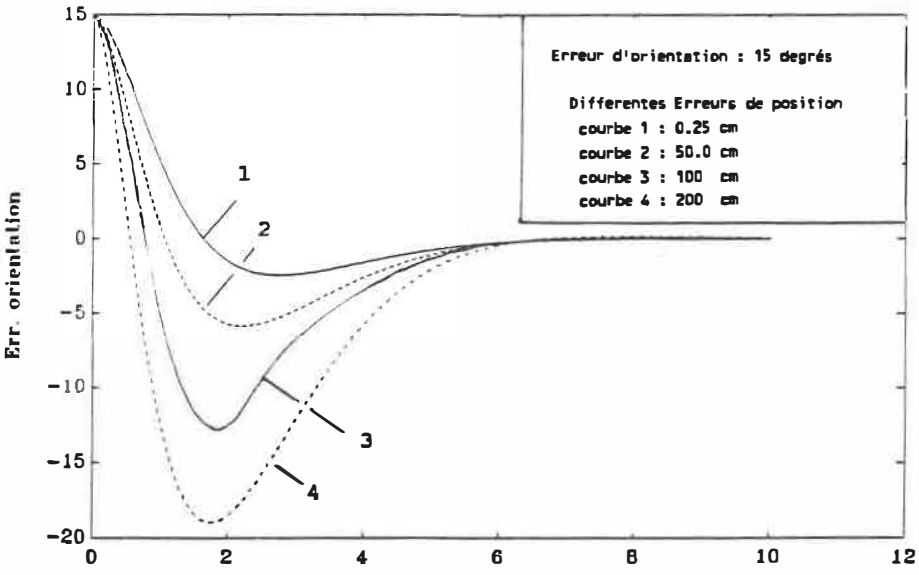


Figure 4.8b: Reponse sous des conditions nominales en correspondance avec diverses erreurs de position initiales (Erreur d'orientation vs Temps)

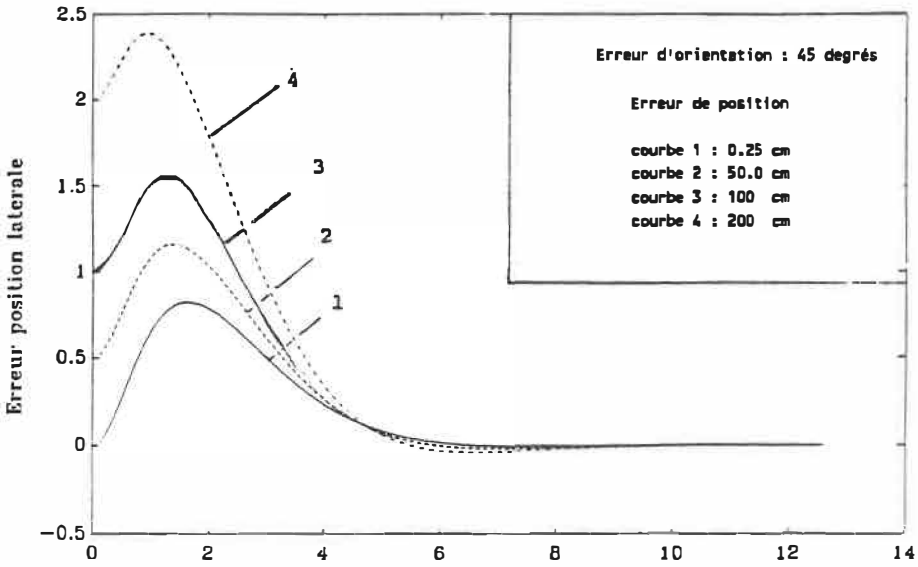


Figure 4.8c: Reponse sous des conditions nominales en correspondance avec diverses erreurs de position initiales (Erreur de position vs Temps)

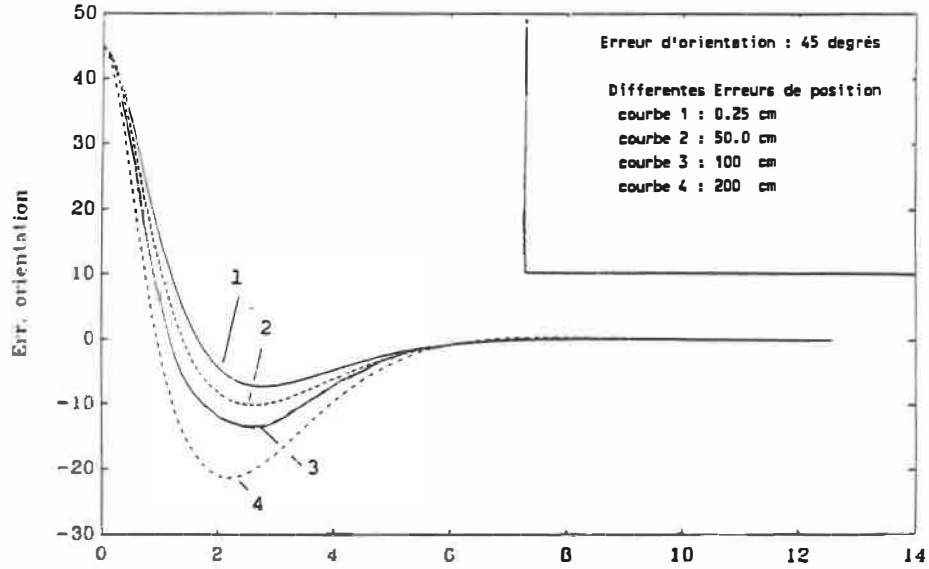


Figure 4.8d: Reponse sous des conditions nominales en correspondance avec diverses erreurs de position initiales (Erreur d'orientation vs Temps)

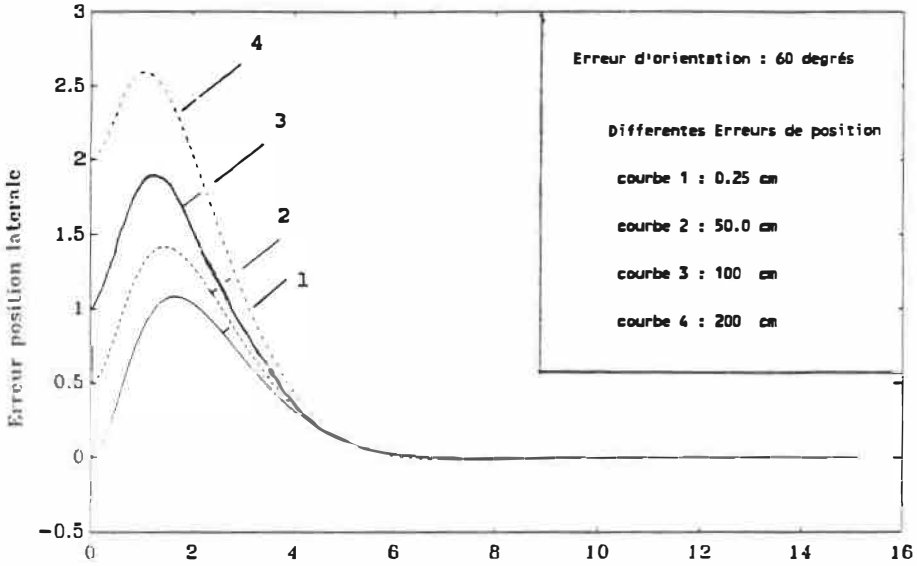


Figure 4.8e: Reponse sous des conditions nominales en correspondance avec diverses erreurs de position initiales (Erreur de position vs Temps)

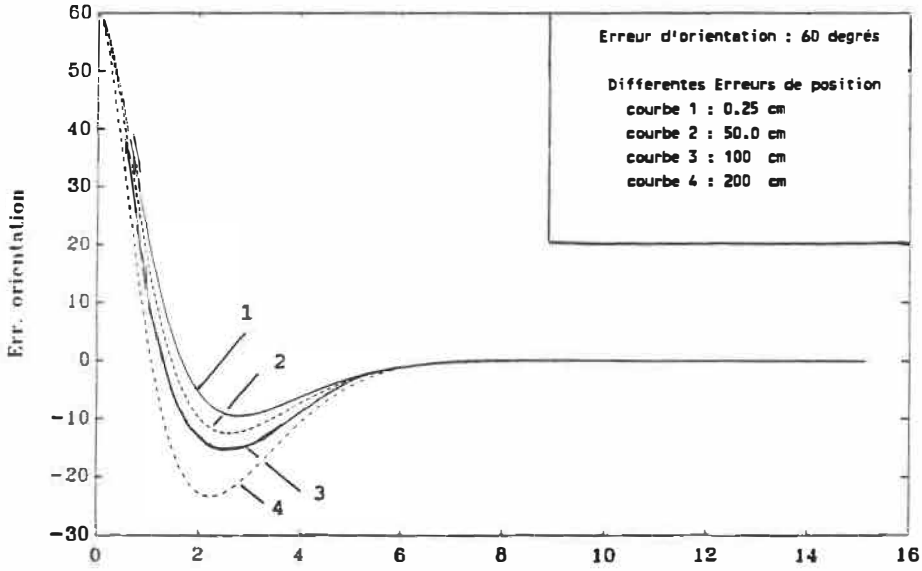


Figure 4.8f: Reponse sous des conditions nominales en correspondance avec diverses erreurs de position initiales (Erreur d'orientation vs Temps)

Les courbes de vitesse sont confondues pour les trois séries de simulations

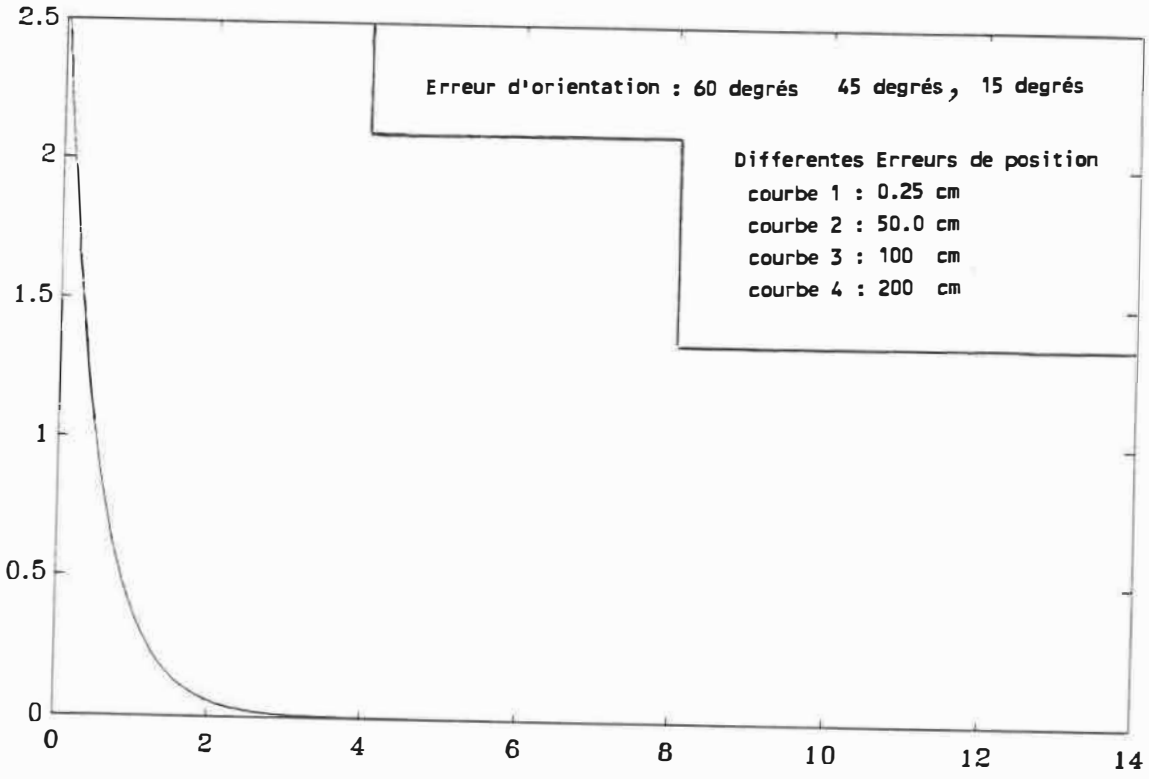


Figure 4.8g: Reponse sous des conditions nominales en correspondance avec diverses erreurs de position initiales (Erreur de vitesse vs Temps)

EXPERIENCE No 3: INFLUENCE DE LA VITESSE DESIREE

Objectif : Illustrer le comportement dynamique du système sous des conditions d'opérations nominales en correspondance avec diverses valeurs de la vitesse désirée.

Modalités de simulation : Identiques à celles ui caractérisent l'expérience No 1 avec la variante que diverses valeurs de vitesse désirée sont maintenant considérées. En particulier :

Vitesse désirée = 2.5, 5, 20, 50 m/s

Résultats : Les courbes à la figure 4.9 montrent l'évolution de la vitesse et de l'erreur de position et d'orientation du véhicule en fonction du temps et des diverses valeurs de la vitesse désirée.

Discussion : Lorsque la vitesse de consigne varie dans une plage de valeurs allant de 2.5 à 10 m/s, le comportement du système est suffisamment proche du comportement désiré. Cependant lorsque cette vitesse est trop faible par rapport à sa valeur nominale le comportement dynamique du système est plus amorti que prévu; lorsqu'elle est considérablement plus forte ce même comportement est sous amorti.

Tout cela suggère que sous des conditions où la vitesse ne varie pas trop par rapport à sa valeur nominale, les gains du contrôleur pourront conserver leur valeur nominale; dans des conditions où la vitesse désirée est prévue varier d'une façon importante, un ajustement adaptatif des gains doit être envisagé.

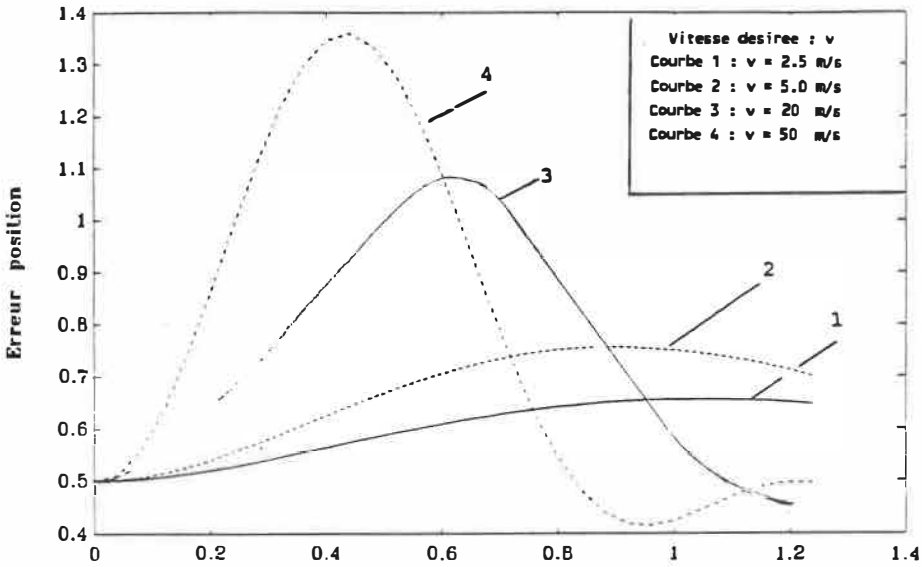


Figure 4.9a: Reponse sous des conditions nominales en correspondance avec diverses valeurs de la vitesse desiree (Erreur de position vs Temps)

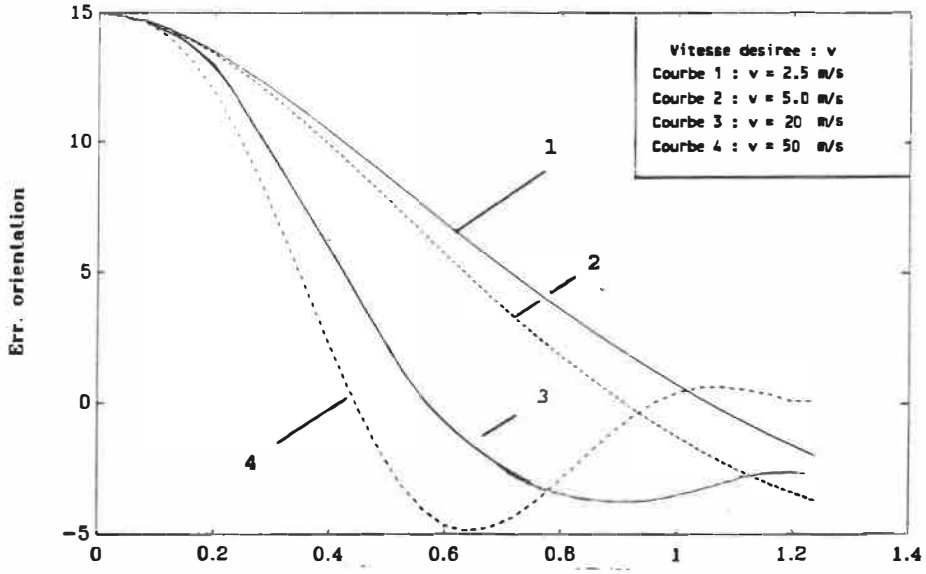


Figure 4.9b: Reponse sous des conditions nominales en correspondance avec diverses valeurs de la vitesse desiree (Erreur d'orientation vs Temps)

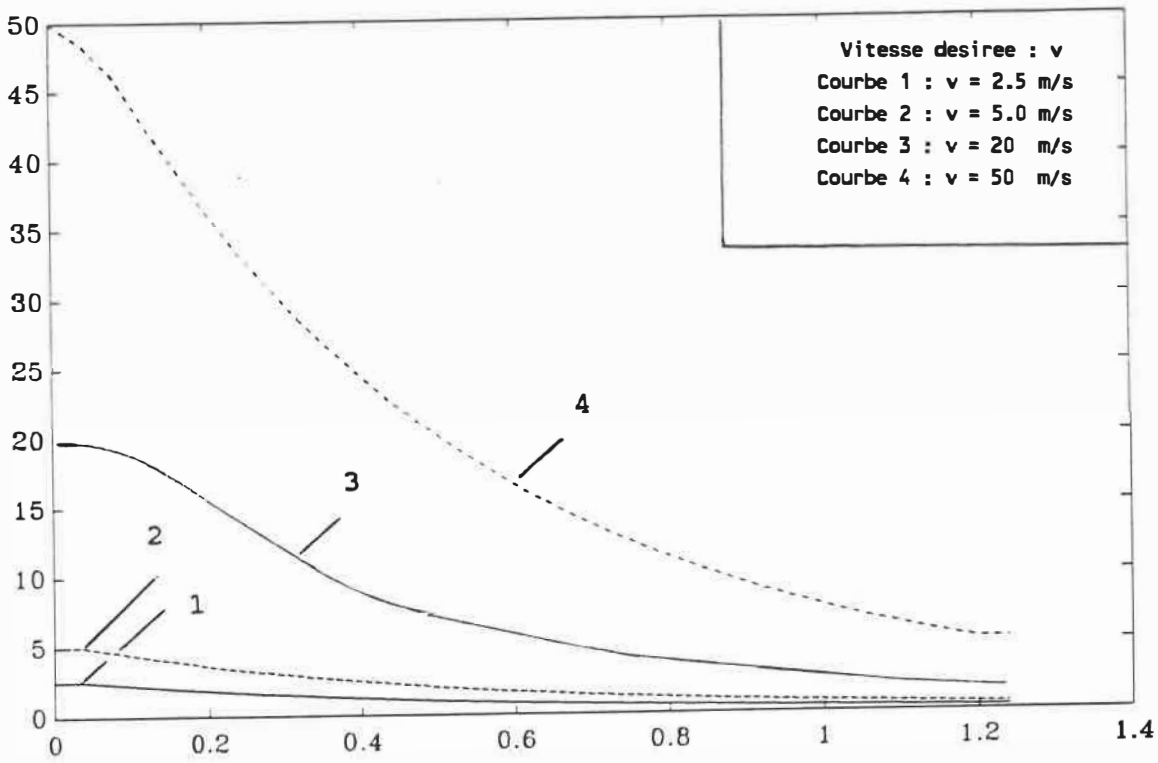


Figure 4.9c: Reponse sous des conditions nominales en correspondance avec diverses valeurs de la vitesse desiree (Erreur de vitesse vs Temps)

EXPERIENCE No 4 : INFLUENCE DU FACTEUR D'ERREUR COMBINEE β

Objectif : Illustrer le comportement dynamique du système sous des conditions d'opérations nominales en correspondance avec diverses valeurs du facteur d'erreur combinée β .

Modalités de simulation : Identitiques à celles qui caractérisent l'expérience No 1 avec la variante que diverses valeurs du facteur d'erreur combinée en correspondance avec deux différentes valeurs de la vitesse désirée sont maintenant considérées. En particulier :

Facteur d'erreur combiné β = 0.12, 0.7, 0.9
Vitesse désirée = 2.5 m/s, 10 m/s

Résultats : Les courbes à la figure 4.10 montrent l'évolution de la vitesse et de l'erreur de position et d'orientation du véhicule en fonction du temps et des diverses valeurs du facteur d'erreur combinée

Discussion : Le comportement du système présente des analogies avec ce que l'on a observé lors de l'expérience No 3. Lorsque le facteur d'erreur combinée est dans l'intervalle [0.45, 0.7], le comportement dynamique du système est bien satisfaisant par rapport aux applications envisagées pour le véhicule mobile. Cependant, lorsque cette valeur est trop faible par rapport à la plage de valeurs déterminée ci-dessus le comportement dynamique du système est plus amorti que prévu; lorsqu'elle est considérablement plus forte ce même comportement est plus sous amorti. Cela semble confirmer les conclusions relatives à l'expérience 3; sous des conditions où la vitesse désirée ne varie pas trop par rapport à sa valeur nominale, la valeur du facteur d'erreur combinée et des gains du contrôleur pourront conserver leur valeur nominale; dans des conditions où la vitesse désirée est prévue varier d'une façon importante, un ajustement adaptatif de ces valeurs doit être envisagé.

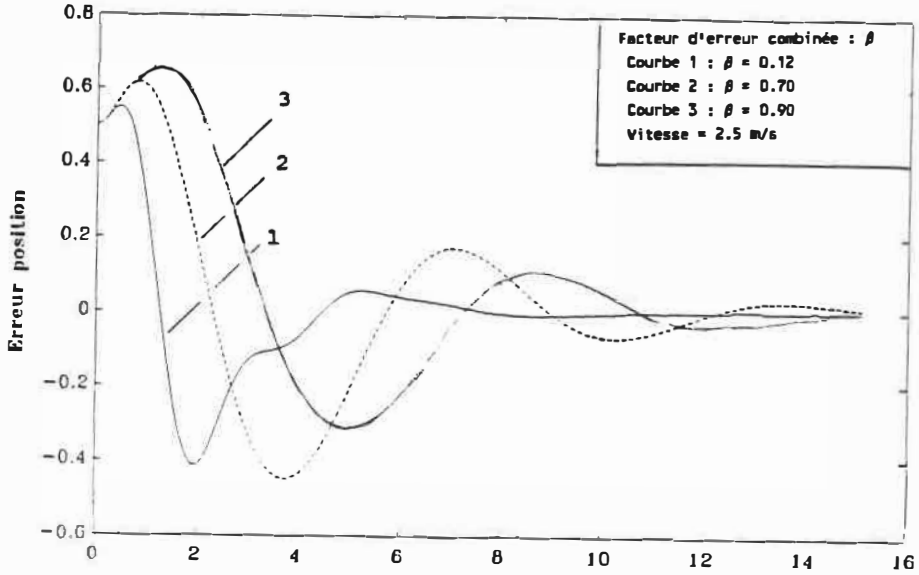


Figure 4.10a(serie 1: v = 2.5 m/s): Reponse sous des conditions nominales en correspondance avecs diverses valeurs du facteur d'erreur combinee (Erreur de position vs Temps)

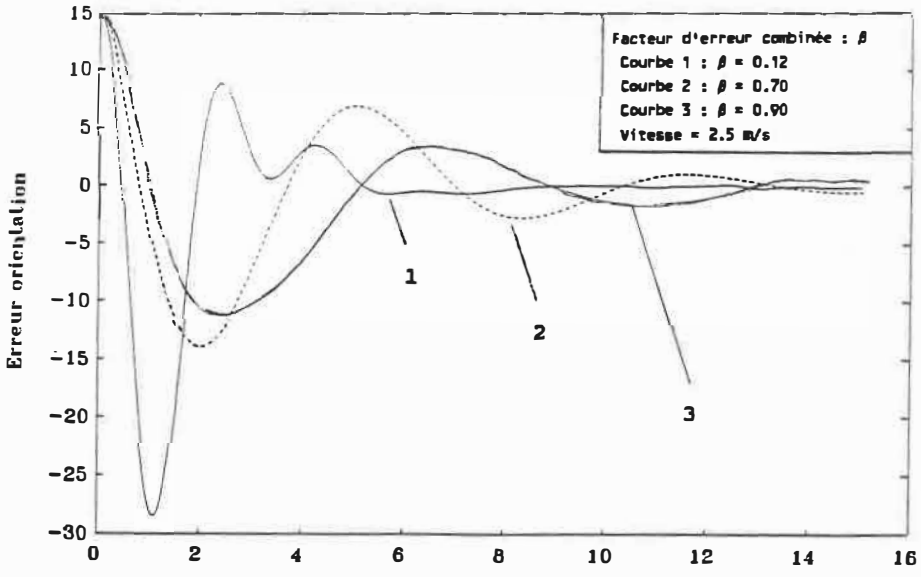


Figure 4.10b(serie 1: v= 2.5 m/s): Reponse sous des conditions nominales en correspondance avecs diverses valeurs du facteur d'erreur combinee (Erreur d'orientation vs Temps)

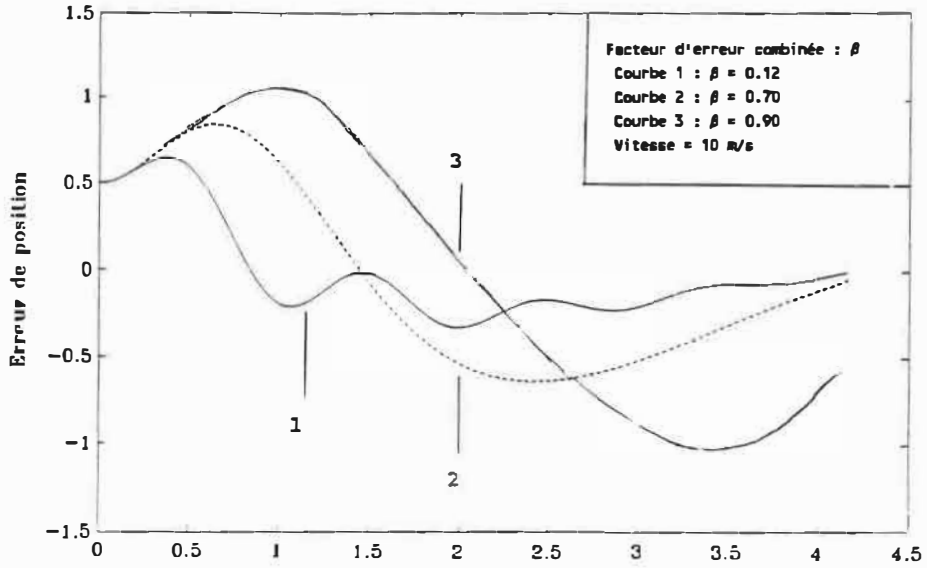


Figure 4.10a(serie 2: v = 10 m/s): Reponse sous des conditions nominales en correspondance avecs diverses valeurs du facteur d'erreur combinee (Erreur de position vs Temps)

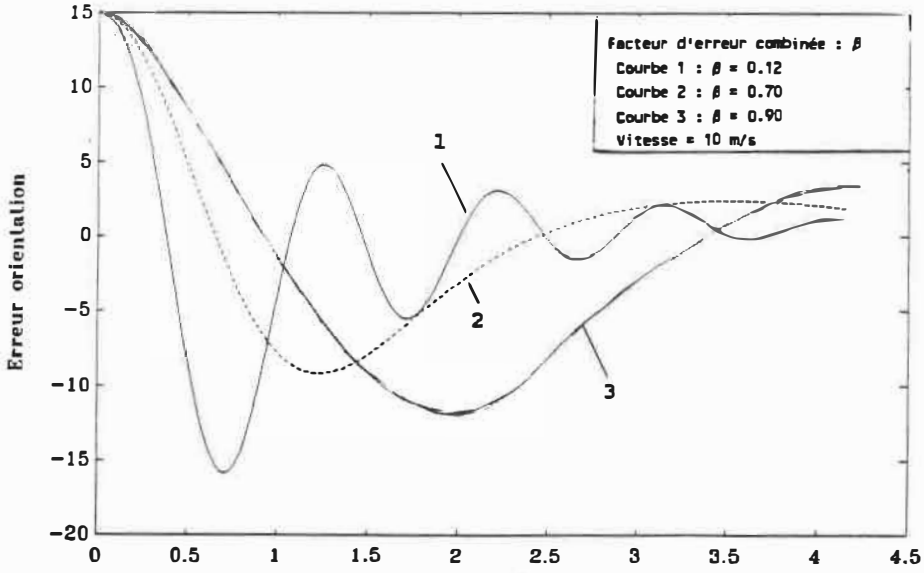


Figure 4.10b(serie 2: v = 10 m/s): Reponse sous des conditions nominales en correspondance avec diverses valeurs du facteur d'erreur combinee (Erreur d'orientation vs Temps)

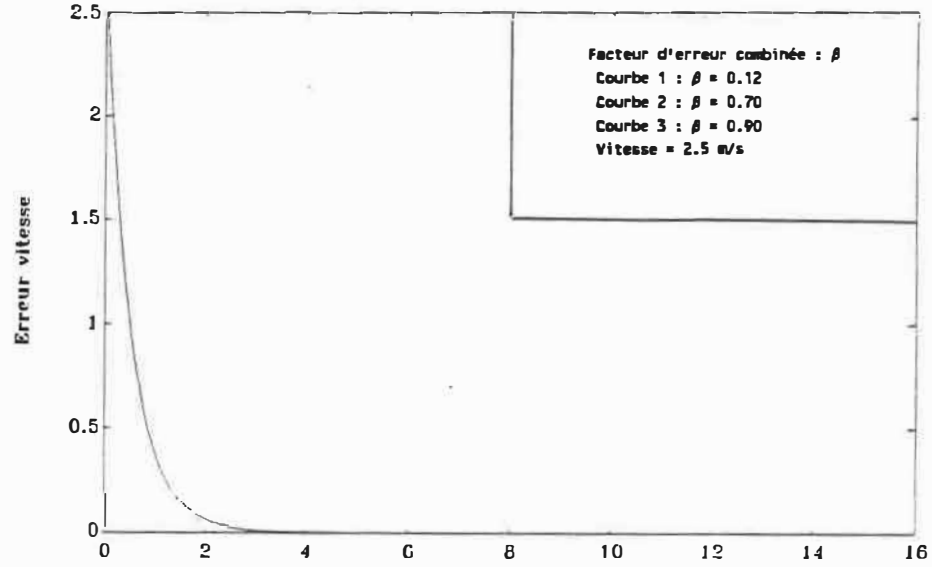


Figure 4.10c(serie 1: v = 2.5 m/s): Reponse sous des conditions nominales en correspondance avec diverses valeurs du facteur combinee (Erreur de vitesse vs Temps)

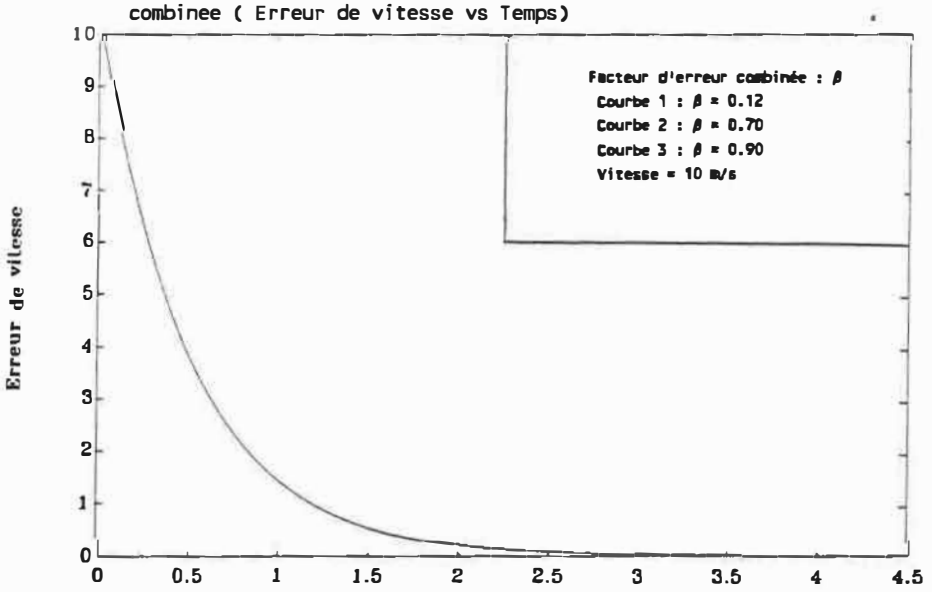


Figure 4.10c(serie 2: v = 10 m/s): Reponse sous des conditions nominales en correspondance avec diverses valeurs du facteur combinee (Erreur de vitesse vs Temps)

EXPERIENCE No 5 : INFLUENCE D'ASYMETRIES DANS LES PARAMETRES DU VEHICULE

Objectif : Illustrer l'inflence de la présence d'asymétries dans la valeur des paramètres du véhicule sur le comportement dynamique du système.

Modalités de simulation : Identiques à celles qui caractérisent l'expérience No 1 avec la variante que les paramètres des servomoteurs ne sont pas symétriques. En particulier, deux séries de tests sont considérées :

1ère série :

Constante de temps : Tau1 = 0.5; Tau2 = 0.75, 1
Gain statique : Km1 = 0.8; Km2 = 0.8

2ème série :

Constante de temps : Tau1 = 0.5; Tau2 = 0.5
Gain statique : Km1 = 0.8; Km2 = 1.0, 1.2

Résultats : Les courbes aux figures 4.11 et 4.12 montrent l'évolution de la vitesse et de l'erreur de position et d'orientation du véhicule en fonction du temps et des diverses asymétries dans la valeur des paramètres du véhicule.

Discussion : En général l'influence d'asymétries dans la valeur des paramètres du véhicule sur le comportement dynamique du système n'est pas très grande. Les figures 4.11(a-b) montrent qu'une variation de l'asymétrie de 50% sur les constantes de temps du moteur perturbe un peu la réponse transitoire du système tout en laissant inchangée la réponse en régime permanent. Les figures 4.12(a-b) montrent qu'une variation de l'asymétrie de 25% sur les gains statiques perturbe un peu la réponse en régime tout en laissant presque inchangée la réponse transitoire.

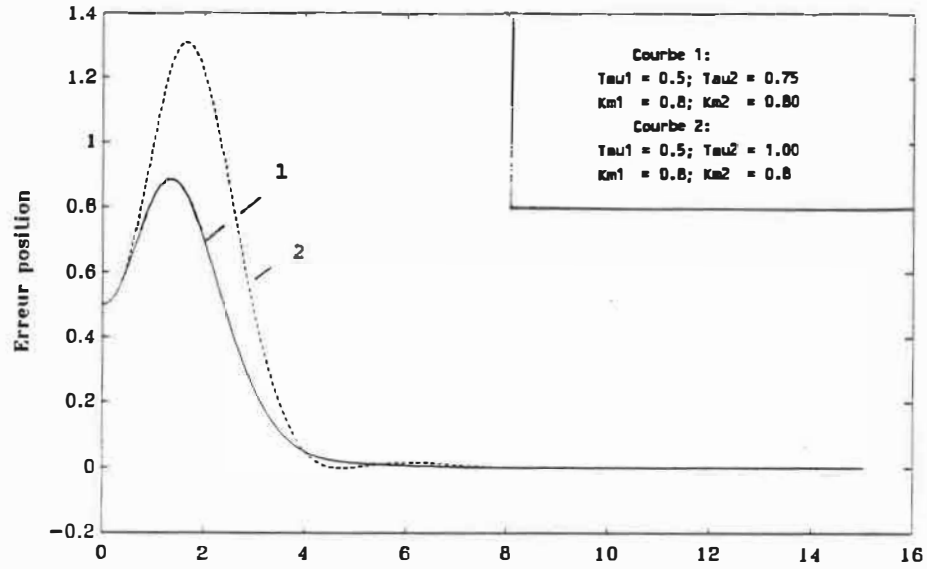


Figure 4.11a: Influence d'une asymetrie au niveau de la constante de temps des servomoteurs (Erreur de position vs Temps)

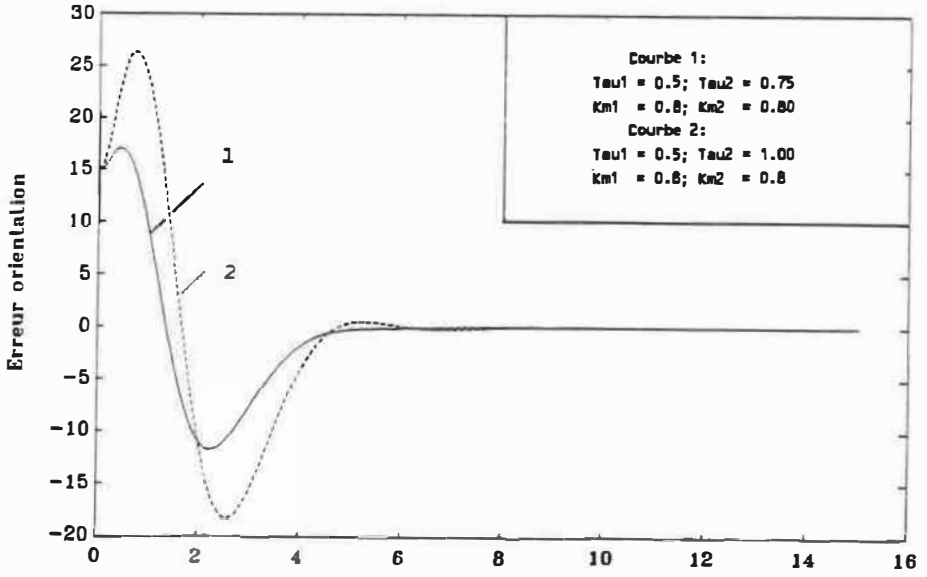


Figure 4.11b: Influence d'une asymetrie au niveau de la constante de temps des servomoteurs (Erreur d'orientation vs Temps)

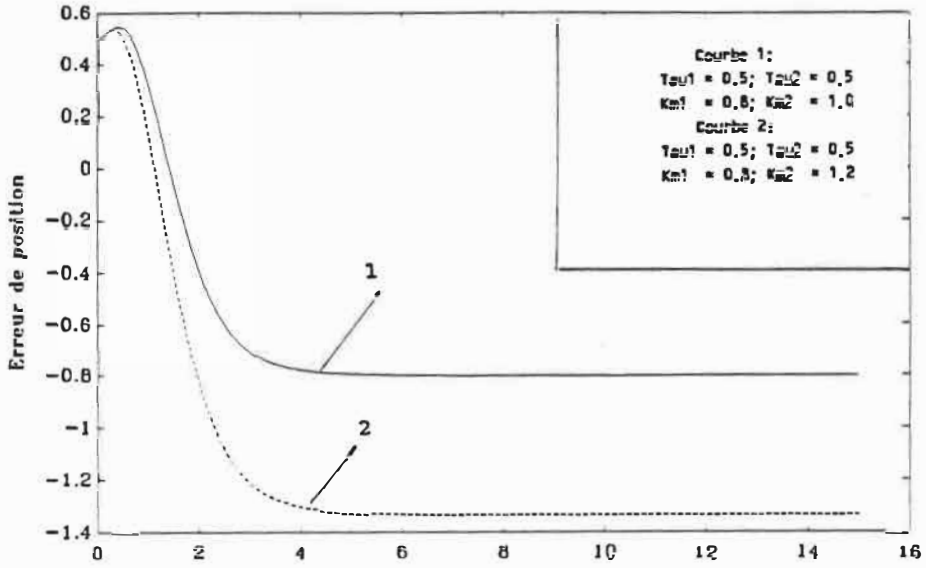


Figure 4.12a: Influence d'une asymetrie au niveau du gain statique des servomoteurs (Erreur de position vs Temps)

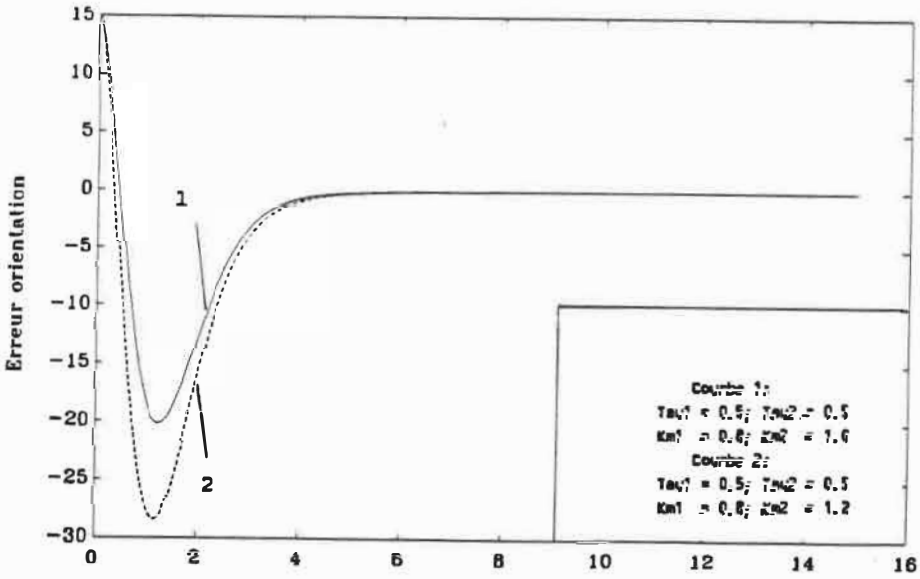


Figure 4.12b: Influence d'une asymetrie au niveau du gain statique des servomoteurs (Erreur d'orientation vs Temps)

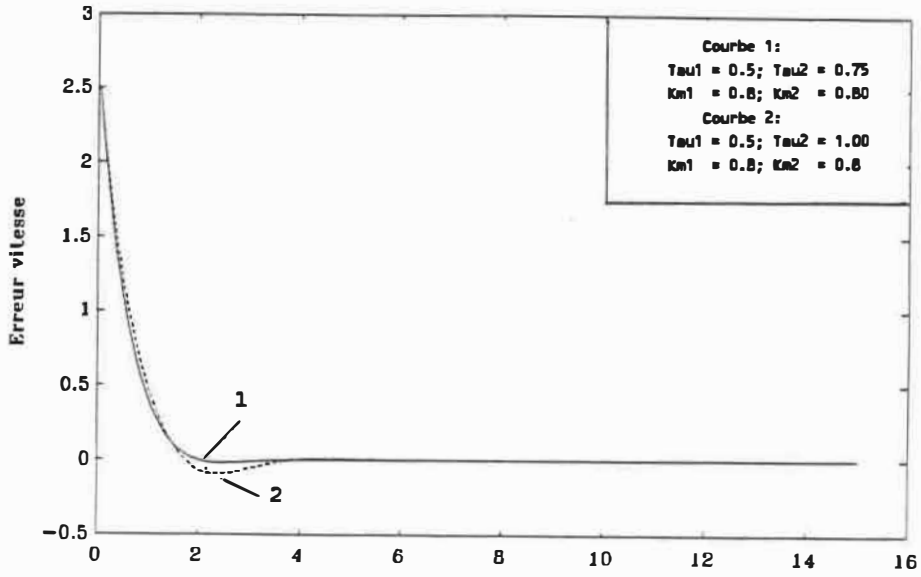


Figure 4.11c: Influence d'une asymetrie au niveau de la constante de temps des servomoteurs (Erreur de vitesse vs Temps)

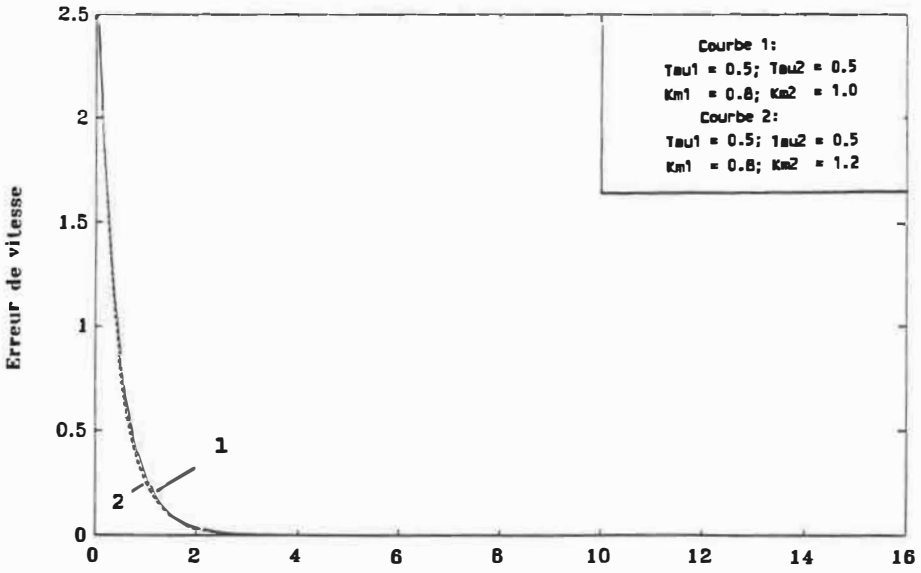


Figure 4.12c: Influence d'une asymetrie au niveau du gain statique des servomoteurs (Erreur de vitesse vs Temps)

EXPERIENCE No 6 : INFLUENCE D'UN DELAI DANS LA COMMANDE

Objectif : Illustrer l'influence de la présence d'un délai dans les calculs relatifs à la mesure et à la commande.

Modalités de simulation : Identiques à celles qui caractérisent l'expérience No 1 avec la variante que les mesures odométriques sont maintenant entachées d'erreurs; la fréquence des mesures inertielle est variable. L'expérience est conduite en utilisant les paramètres du tableau 1 avec les modifications indiquées ci-dessous

Délai
Délai1 = 0, 0.2, 0.35 s
Délai2 = 0, 0.2, 0.35 s

Résultats : Les courbes à la figure 4.13 montrent l'évolution de la vitesse et de l'erreur de position et d'orientation du véhicule en fonction du temps en correspondance avec plusieurs valeurs de délai.

Discussion : En correspondance avec la présence d'un délai inférieur à 300 msec, l'influence de ce délai sur le comportement dynamique du système est négligeable. Cependant des instabilités se manifestent lors de la présence de délais plus élevés.

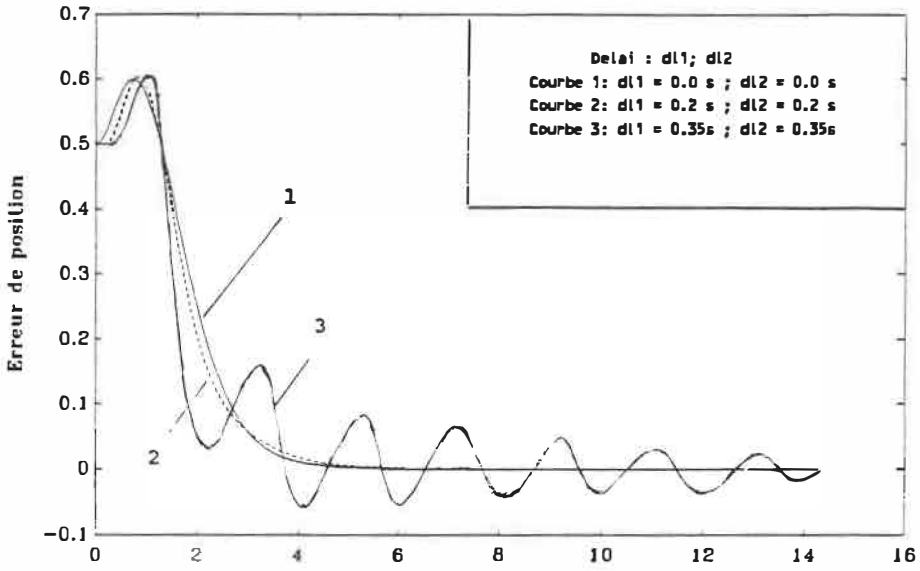


Figure 4.13a: Influence de la presence d'un delai
(Erreur de position vs Temps)

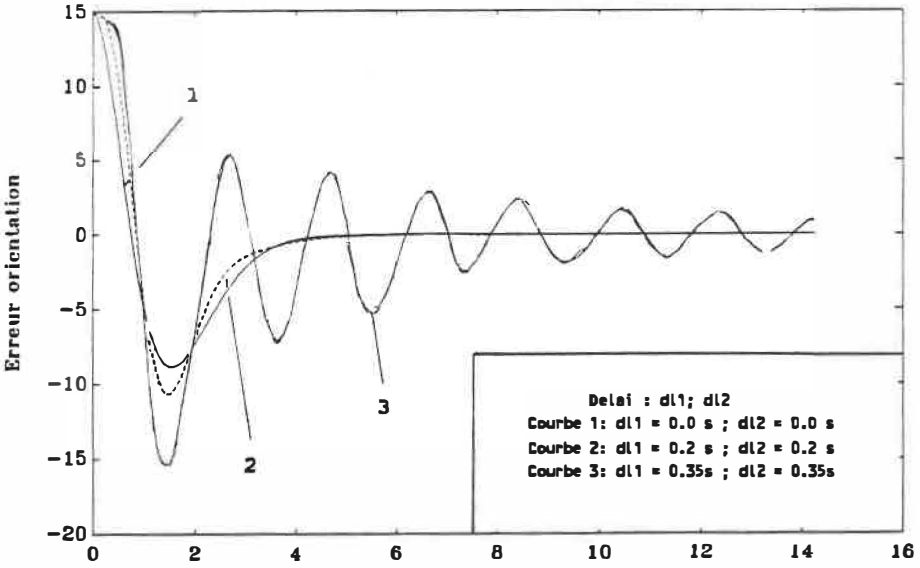


Figure 4.13b: Influence de la presence d'un delai
(Erreur d'orientation vs Temps)

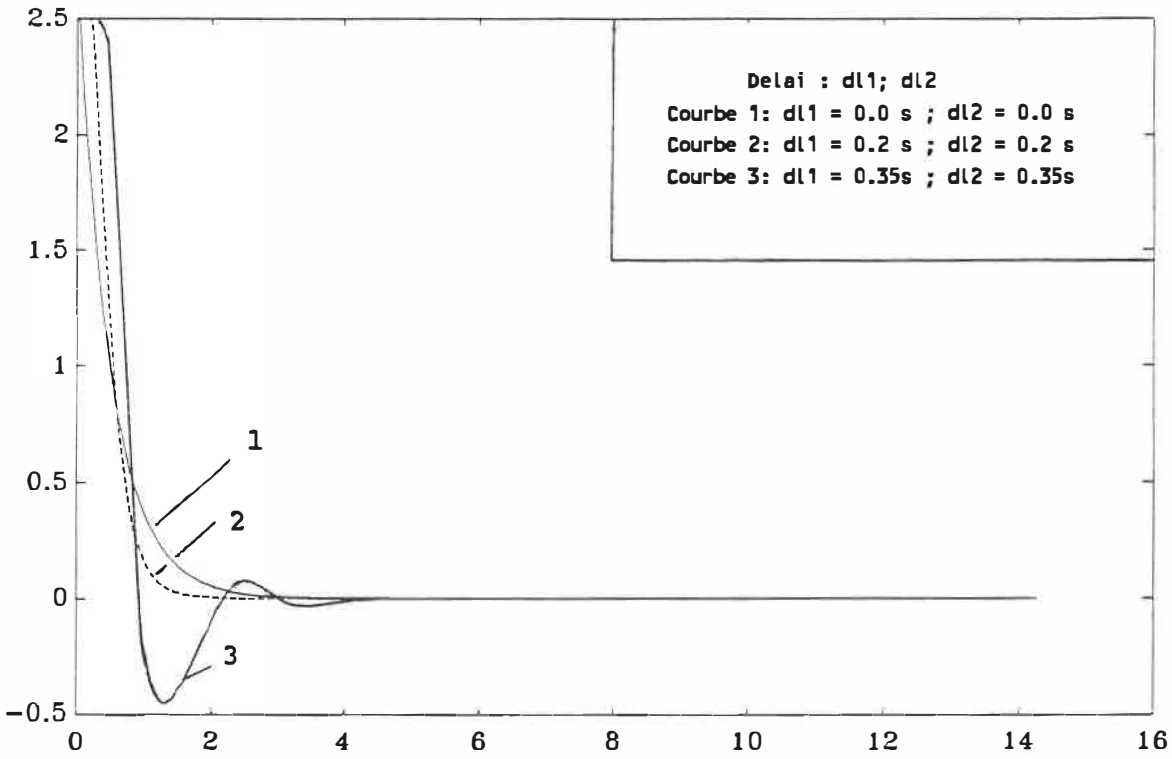


Figure 4.13c : Influence de la presence d'un delai
(Erreur de vitesse vs Temps)

EXPERIENCE No 7 : INFLUENCE DE LA PERIODE CARACTERISANT LA DISPONIBILITE DE MESURES INERTIELLES

Objectif : Illustrer l'influence de l'intervalle de temps qui sépare deux mesures inertielles par rapport à la précision des mesures odométriques.

Modalités de simulation : Identiques à celles qui caractérisent l'expérience No 1 avec la variante que les mesures odométriques sont maintenant entachées d'erreurs; la fréquence des mesures inertielles est variable. L'expérience est conduite en utilisant les paramètres du Tableau 1 avec les modifications indiquées ci-dessous. Deux séries de tests sont considérées :

1ère série (estimées odométriques de bonne qualité)

Rayon	:	R1 = 0.12, R2 = 0.12 m
largeur véhicule	:	largeur estimée = 0.60 m
Echantillonage	:	Commande = 0.01 Mesure inertielle = 0.2, 1, 2 s

2ème.série (estimées odométriques de mauvaise qualité)

Rayon	:	R1 = 0.12, R2 = 0.14 m
largeur véhicule	:	largeur estimée = 0.40 m
Echantillonage	:	Commande = 0.01 Mesure inertielle = 0.2, 1, 2 s

Résultats : Les courbes aux figures 4.14 et 4.15 montrent l'évolution de la vitesse et de l'erreur de position et d'orientation du véhicule en fonction du temps et en correspondance avec chaque série de tests.

Discussion : En correspondance avec des mesures odométriques suffisamment précises, l'intervalle de temps qui sépare deux mesures inertielles peut être considérablement plus grand que celui relatif à l'application de la commande sans que la performance du système soit détérioré d'une façon importante. L'inverse est vrai lorsque la précision des mesures odométriques se dégrade. Dans ce cas, pour que la performance du système ne soit détériorée d'une façon importante, il faut que l'intervalle de temps qui sépare deux mesures inertielles soit réduit d'une façon proportionnelle à la dégradation qui caractérise la précision des mesures odométriques.

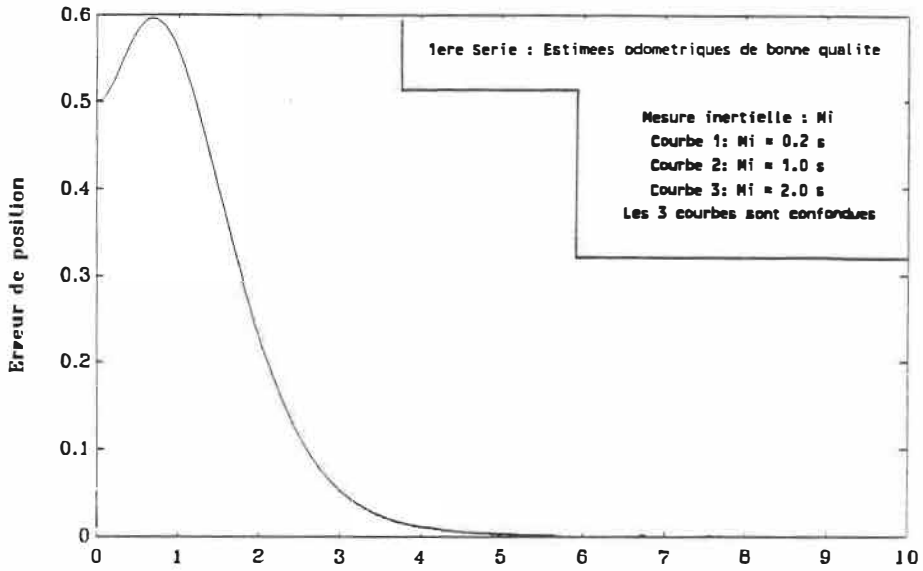


Figure 4.14a: Influence de l'intervalle de temps qui separe deux mesures inertielles lors d'une bonne precision dans les mesures odometriques (Erreur de position vs Temps)

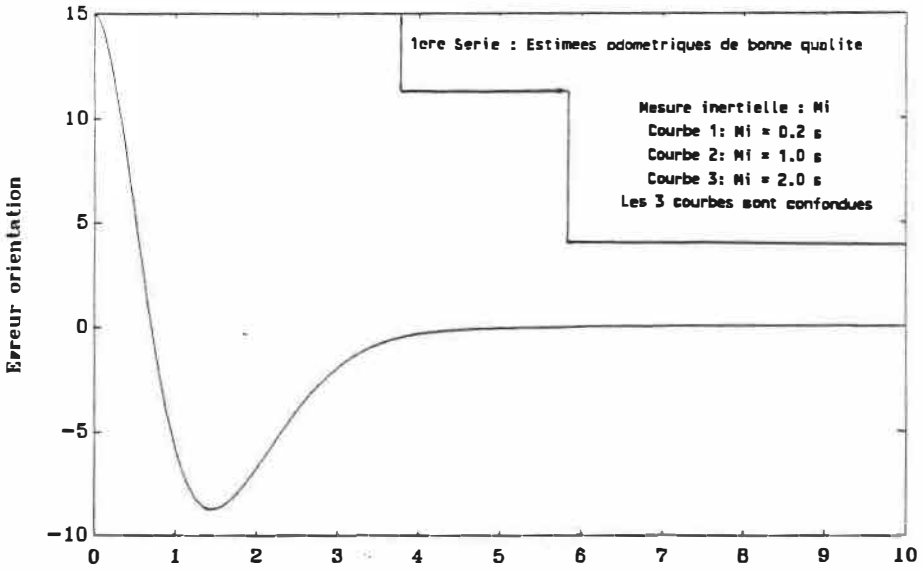


Figure 4.14b: Influence de l'intervalle de temps qui separe deux mesures inertielles lors d'une bonne precision dans les mesures odometriques (Erreur d'orientation vs Temps)

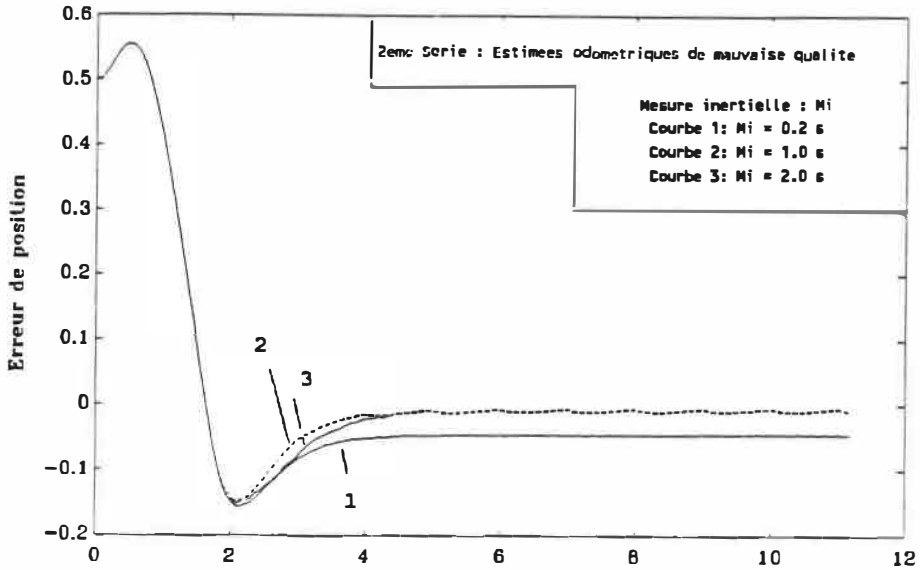


Figure 4.15a: Influence de l'intervalle de temps qui separe deux mesures inertielles lors d'une faible precision dans les mesures odometriques (Erreur de position vs Temps)

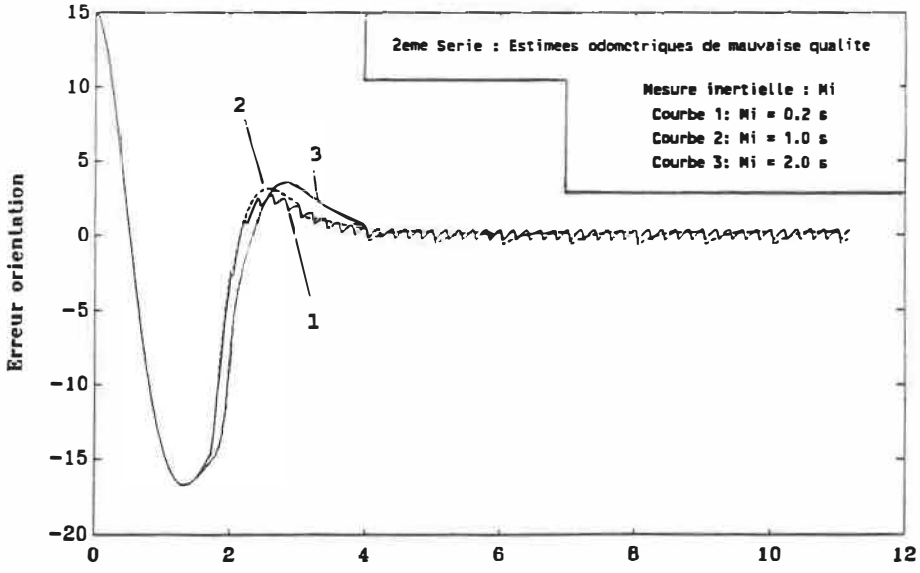


Figure 4.15b: Influence de l'intervalle de temps qui separe deux mesures inertielles lors d'une faible precision dans les mesures odometriques (Erreur d'orientation vs Temps)

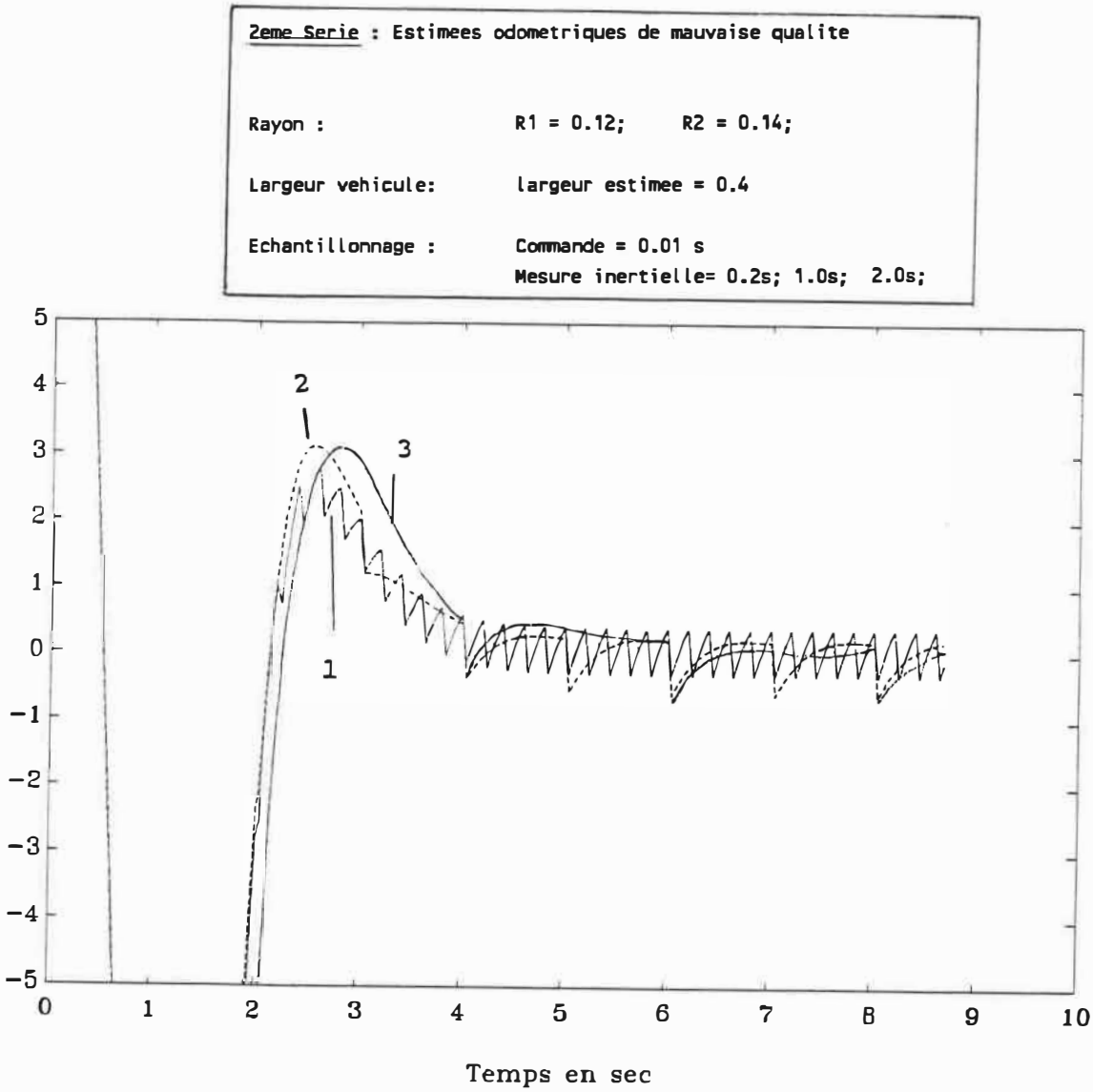


Figure 4.15b: Influence de l'intervalle de temps qui separe deux mesures inertielles lors d'une faible precision dans les mesures odometriques
(Erreur d'orientation vs Temps)

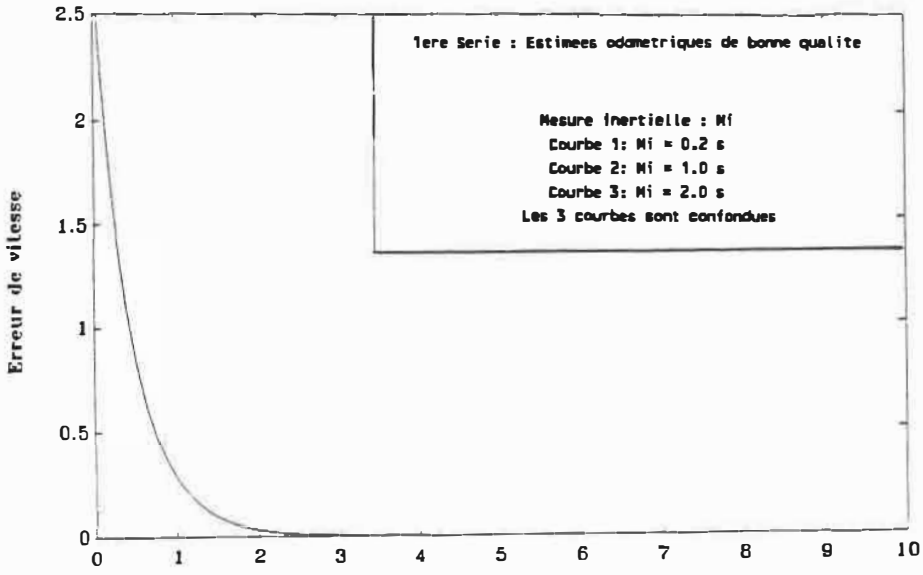


Figure 4.14c: Influence de l'intervalle de temps qui separe deux mesures inertielles lors d'une bonne precision dans les mesures odometriques (Erreur de vitesse vs Temps)

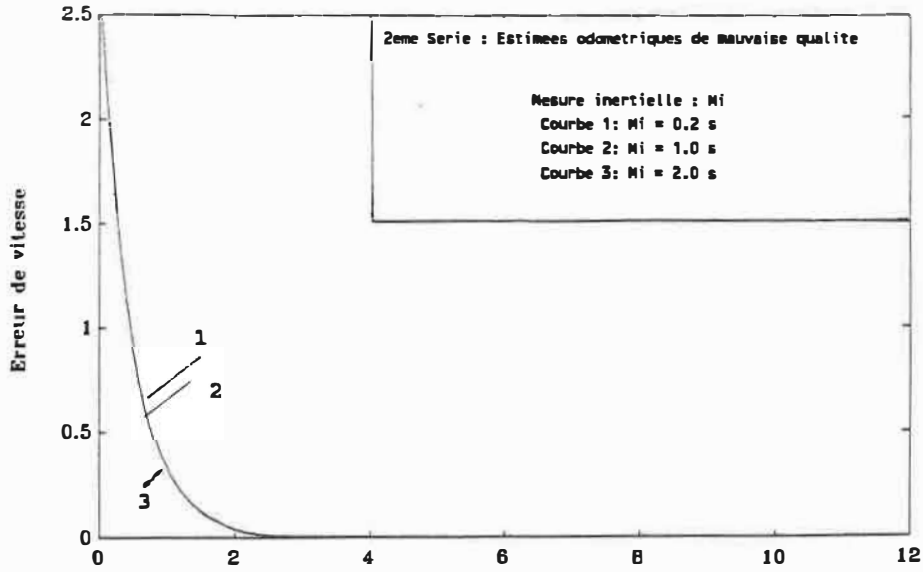


Figure 4.15c: Influence de l'intervalle de temps qui separe deux mesures inertielles lors d'une faible precision dans les mesures odometriques (Erreur de vitesse vs Temps)

4.3 Conclusion

Sous les conditions nominales la réponse du système au point de vue précision, temps de réponse et comportement dynamique est bien adéquate aux applications prévues. La vitesse du système est très peu influencée par les erreurs de position/orientation initiales (expérience No 2) et peut être augmentée jusqu'à 10 m/s sans causer trop de perturbations à la linéarité du système.

Cependant, l'influence de la non-linéarité se manifeste si certaines contraintes ne sont pas respectées. En particulier :

La valeur du facteur d'erreur combinée β doit-être établie en fonction de la valeur de la vitesse de croisière désirée;

Le délai ne peut excéder 300 msec pour éviter une réponse oscillatoire du système;

L'asymétrie des gains statiques (Km1, Km2) influence beaucoup la position latérale du véhicule mais peut se corriger en ajustant le gain de l'intégrateur du PID;

L'asymétrie des rayons de roues constitue une mauvaise estimée odométrique et cause des oscilllations tout au long de la trajectoire;

En effet, avec une bonne estimée odométrique (expérience 7) la grandeur de la période de mesure inertielle par rapport à celle de la commande ne détériore pas la performance du système.

Une description des capteurs utilisés pour la mesure inertielle se trouve à l'appendice A.

CHAPITRE 5

CONCLUSION

5.1 Résultats obtenus

L'objectif du projet a été atteint c'est-à-dire la conception d'un système d'autopilotage capable de guider un véhicule autonome (ou robot mobile) le long d'une route assignée en mémoire. Quelques tests effectués en collaboration avec des collègues sur le prototype de robot mobile au département du génie electrique ont été très performants pour l'application envisagée (minière).

La modélisation et l'étude de la sensibilité au point de vue comportement statique et dynamique du système ont donnés des résultats satisfaisants par rapport aux applications envisagées pour le véhicule. La précision sur le suivi de la trajectoire optimale s'avère efficace due à l'analyse du déplacement au niveau global et local par les techniques de la navigation et du dead-reckoning.

Dans une plage de valeurs de la vitesse comprise entre 0 et 10 m/s la réponse des contrôleurs ne manifeste

aucune oscillation. Ce qui signifie que les contrôleurs sont assez robustes aux effets de perturbations extérieures et sont capables de conserver la linéarité du système.

Dans la simulation nous avons essayer de simuler les différents phénomènes réels susceptibles d'intervenir dans le comportement d'un véhicule mobile autonome: réponse sous conditions nominales en correspondance avec diverses erreurs de position initiale, influence de l'erreur combinée β, variation de la vitesse désirée, présence d'un délai dans la commande, asymétrie des gains statiques ou de la constante de temps des servomoteurs, influence de l'intervalle de temps entre deux mesures inertielles lors d'une bonne ou faible précision des mesures odométriques ; les courbes de simulation (chapître 4) nous révèlent les bonnes performances du systéme au point de vue temps de réponse, amortissement, stabilisation.

En conséquence, le système d'autopilotage étudié dans ce projet peut servir d'outil pour guider un véhicule autonome (ou robot mobile) pour diverses applications : académique, transport, minière ou autres.

5.2 Extensions possibles

5.2.1 Système d'autopilotage

Certains points de l'étude pourraient faire l'objet d'extensions possibles pour une recherche supplémentaire :

1) Le robot pourrait être équipé de deux caméras combinés avec les capteurs pour faire une analyse d'images. La séquence du suivi de la route comprendrait à ce moment là un modèle mathémathique du dead-reckoning et de la navigation (ou mesure) qui interface avec un système de traitement numérique d'images.

2) Planification de la route avec la détection et la programmation des obstacles : recherche heuristique pour la détermination de la trajectoire idéale basée sur l'algorithme d'optimisation du chemin

3) Envisager l'étude des contrôleurs avec les méthodes de contrôle tels que le Dual ou Relais.

5.2.2 Conception Infographique

La méthode utilisée dans le cadre de ce projet est précise et permet de visualiser le déplacement du véhicule en temps réel.

Cependant, nous pourrions nous intéresser à un niveau d'animation graphique plus élevé qui consiste à un déplacement physique naturel d'un véhicule mobile. L'implantation comprendrait ainsi les modules suivants au niveau de la navigation :

a - module DEPART et ACCELERATION
b - module VITESSE
c - module DECELERATION et ACCELERATION avec le curseur
d - module DECELERATION et FREINAGE avant l'arrêt

Malheureusement, ces modules graphiques nécessitent un grand espace mémoire et un ordinateur plus rapide et plus élaboré avec des options graphiques qui dépassent les possibilités du PC.

BIBLIOGRAPHIE

[1] DE SANTIS R.M., "Les éléments fondamentaux de la robotique, cinématique, dynamique et commande", Edition Ecole Polytechnique de Montreal, (1986).

[2] DE SANTIS R.M., " Théorie des asservissements linéaires", Edition Ecole Polytechnique de Montreal,(1986)

[3] HURTEAU R., "Systèmes et simulation (2e edition)", Edition Ecole Polytechnique de Montreal, (dec. 1985).

[4] DORF C., "Modern Control System" Addison-Wesley (1983)

[5] TERROUX A., DE SANTIS R.M., "ASSER.POS" : Un programme pour la simulation de la commande d'un joint robotique, Ecole Polytechnique de Montreal, (1984).

[6] PUN, "Introduction to Optimization Practice", John Wisley, (1969).

[7] RABEMANANTSOA M., HOANG H., "Optimisation d'un réseau de communications", Projet de fin d'Etudes, Ecole Polytechnique de Montreal, (1977).

[8] PUN, "Optimisation: techniques & Théories", (1970)

[9] WRIGHT J.D., EDGAR T.F., "Digital Computer Control and Signal Processing Algorithms", (1985).

[10] RABEMANANTSOA M., "Système Expert pour l'optimisation des paramètres de gain d'un contrôleur PID", Annexe au mémoire de M.Sc.A, Ecole Polytechnique de Montreal, (1989).

[11] DE SANTIS R.M., "An Adaptative PI/Sliding Mode Controller for a Speed Drive", Journal of Dynamic Systems, Measurement and Control, (Sept 1989).

[12] THALMANN D., THALMANN N.,"Informatique Graphique: concepts et techniques", Gaétan Morin, (1983).

[13] CHAN S., PARK, "Interactive computer graphics", Addison-Wesley, (1985).

[14] WEINSTOCK,"Computer Animation", Addison-Wesley,(1986)

[15] MORAVEC P., "Rover Visual Obstacle Avoidance", Proc. 7th Int. Joint Conf. A.I, Vancouver, (1981 b).

[16] CHATILA R., "Path Planning and environment learning in a Mobile Robot system", Conf. on A.I, France , (1980).

[17] BAUZIL G., BRIOT M., and RIBES P., "A navigation Subsystem Using Ultrasonic Sensors for the Mobile Robot HILARE", Robot Vision and Sensory Controls Conference., France, (1980).

[18] TSUMURA T., and al. "An Experimental System for Automatic Guidance of Robot Vehicle following the Route stored in Memory", 11th Int. Symp. of Robot, Tokyo, (1981)

CHATTERGY R., "Some Heuristics for the Navigation of a Robot", Int. Robotics Research, Vol.4, No 1, (1985).

[20] COLES, ALAN, SINCLAIR, "Decision Analysis for an Experimental Robot Unreliable Sensors",Proc.4th IJCAI(1975)

[21] NAKAND, KOYACHI, AGARI, HIROOKA Y., "Sensor System of a Guideless Autonomous Vehicle in Flexible Manufacturing System", 15th ISIR, Japan, (1985).

[22] FERRER M., BAUZIL G., et BRIOT M., "Perception de l'Environnement et localisation du Robot Mobile HILARE", 2ème Congrès AFCET/IRIA, tome 3, Toulouse, (1979).

[23] GIRALT G., "A Multi-level Planning and Navigation System for a Mobile Robot", Proc. 6th IJCAI, (1979).

[24] GOUZENES L., "Strategies for Solving Collision-free Trajectories Problem for Mobile and Manipulator Robots", Int. Journal Robotics Research, Vol 3, No 4, (1984).

[25] HOLLIS, "NEWT: A Mobile Cognitive Robot", Byte Magazine, (1978).

[26] IIJIMA J., "Elementary Functions of a Self-contained Robot Yamabico ", 11th. Int. Symp. Robot, Japan, (1981).

[27] JULLIERE M., MARCE L., PERRICHOT H., "A Guidance System for a Vehicle which has to follow a Memorized Path", 2th International Conference AGV System, Stuttgart, (1983)

[28] JULLIERE M., MARCE L., PLACE H., "A Guided System for a Mobile Robot", 13th ISIR/Robots 7 Conf., Chicago, (1983)

[29] KHATIB O., "Real Time Obstacle Avoidance for Manipulators and Mobile Robots", Int. Journal Robotics Research, Vol. 5, No. 1, (1986).

[30] SERFASS C., DE SANTIS R. M., " Asservissement d'une articulation robotique avec un contrôleur à structure variable", EPM/RT-87/19, Ecole Polytechnique de Montreal, (1987).

[31] NGUYEN A. T., DE SANTIS R. M., " Etude de la sensibilité du comportement d'un manipulateur robotique", EPM/RT-87-2, Ecole Polytechnique de Montreal, (1987).

[32] SOUPHANDAVONG P., DE SANTIS R. M., "Robotisation de la tour d'hanoi par retroaction visuelle", EPM/RT-86-45, Ecole Polytechnique de Montreal, (1986).

[33] BORENSTEIN T., KOREN Y., "Motion Control Analysis of a Mobile Robot", Journal of Dynamic Systems, Measurement and Control, Vol 103/73, (June 1987).

[34] RABEMANANTSOA M., DE SANTIS R. M., "SRM : Logiciel de simulation pour l'analyse du système d'autopilotage d'un véhicule autonome", Annexe au Mémoire de M.Sc.A, Ecole Polytechnique de Montreal, (1989).

[35] DE SANTIS R.M., HURTEAU R., On Motion Controller Design for AGVs, Rapport Technique de l'Ecole Polytechnique de Montreal, (en cours).

APPENDICE A

Un "listing" détaillé du logiciel est disponible dans le document [34]; IL nous paraît néanmoins important de présenter dans cet appendice les principaux modules de calcul utilisés pour la navigation incluant les équations dynamiques et cinématiques du modèle.

On y trouvera également le module d'initialisation pour modeliser la route et l'appel de la procédure du "Turbo Numerical Toolbox".

```
Function  Atan2    ( cx, cy       : Real ) :Real;
Function  OR_ROUTE ( Pos_xc,Pos_yc: Real ) :Real;
Function  Sens_rot ( Pos_xc,Pos_yc: Real ) :Real;
Function  Err_dist  ( Pos_xc, Pos_yc, sens       : Real ) : Real ;
Function  ERR_ORIENT( Pos_xc,Pos_yc,orient_robot : Real ) : Real ;
Function  ERR_TOTAL ( orient_robot, Pos_xc, Pos_yc,sens : Real ) : Real ;
Function  Delta_U ( Err_total: Vecteur; Int_err_tot,K1, K2, K3 : Real  ) : Real ;
Function  OR_ROBOT( r1, r2, L, s1, s2 : Real    ;
                    orient_robot      : Vecteur ;
                    vit               : Couple ) : Real ;
Function  OR_ROBOT_ESTIME( rayon_moyen, Largeur_estime : Real    ;
                           orient_robot                : Vecteur ;
                           vit                         : Couple ) : Real ;

Function Pos_Xc ( r1, r2, L, s1, s2 : Real    ;
                     orient_robot : Vecteur ;
                           vit    : Couple  ;
                  Previous_Pos_Xc : Real )  : Real ;
Function Pos_Xc_estime ( rayon_moyen,largeur_estime  : Real    ;
                           orient_robot    : Vecteur ;
                           vit             : Couple  ;
                           Previous_Pos_Xc : Real )  : Real ;

Function Pos_Yc( r1, r2, L, s1, s2 : Real    ;
                     orient_robot : Vecteur ;
                           vit    : Couple  ;
                 Previous_Pos_Yc : Real )  : Real ;
Function Pos_Yc_estime ( rayon_moyen,largeur_estime : Real    ;
                         orient_robot               : Vecteur ;
                         vit                        : Couple  ;
                         Previous_Pos_Yc            : Real )  : Real ;

Function Dynamique_1( X12, tau1, km1, u1 : Real ) : Real ;
Function Dynamique_2( X22, tau2, km2, u2 : Real ) : Real ;
Function Position_1 ( X11, X12, tau1, km1, u1   : Real ) : Real ;
Function Position_2 ( X21, X22, tau2, km2, u2   : Real ) : Real ;
Function U_MED(Vit_desiree,Err_Vit,Int_err_vit,k21,k23,k24:Real): Real ;
Procedure Calcul_depart( X1, Y1, alfa1, De : Real;
                         Var  Kdx: Real;
                         Var  Kdy: Real);
Procedure La_commande( du          : Un_set ;
                       U_m         : Un_set ;
                       Vit_actuel  : real   ;
                       Int_err_vit : Real   ;
                       PC          : TparmsControle;
                       Var u       : Un_set ) ;

Procedure Batit_vect_a_parcourir( x_u, y_u : Vect_a_parcourir ;
                                  Var x_r  : Vect_a_relier ;
                                  Var y_r  : Vect_a_relier ;
                                  nb_seg   : Integer ) ;
```

```
(*****************************************************************************)

                    implementation

(*****************************************************************************)


{
┌──────────────────────────────────────────────────────────────────────────┐
│                                1-> Atan2                                 │
└──────────────────────────────────────────────────────────────────────────┘
}
Function Atan2( cx, cy : Real ) : Real ;

Var  angle,xx,yy,z : Real ;

Begin
  xx := Abs(cx) ;
  yy := Abs(cy) ;

  If cx = 0.0 then
    Begin
      If cy = 0.0 then angle := 0.0 ;
      If cy > 0.0 then angle := PI / 2.0 ;
      If cy < 0.0 then angle := 3.0 * PI / 2.0 ;
    End ;

  If cx > 0.0 then
    Begin
      z := yy / xx ;
      angle := ArcTan(Z) ;
      If ( cy < 0.0 ) then angle := ( 2.0 * PI - angle ) ;
    End ;

  If cx < 0.0 then
    Begin
      If cy = 0.0 then angle := PI ;
      If cy > 0.0 then angle := PI - angle ;
      If cy < 0.0 then angle := PI + angle ;
    End ;
  Atan2 := angle ;
 End ;

{
┌──────────────────────────────────────────────────────────────────────────┐
│                               OR_Route                                   │
└──────────────────────────────────────────────────────────────────────────┘
}
Function  OR_ROUTE (Pos_xc,Pos_yc: Real) :Real ;
```

```
 Begin
    If(Nb_Route = 1) Then { Route Droite }
      OR_ROUTE := Arctan ((y_arrive - y_depart)/ (x_arrive-x_depart));


    if (Nb_Route = 2) Then { Courbe spline }
    Begin

      OR_ROUTE := Atan2 (X_point[Point_de_depart+1] - X_point[Point_de_depart],
                         Y_point[Point_de_depart+1] - Y_point[Point_de_depart]);
    End;
 End;
 {
┌──────────────────────────────────────────────────────────────────────┐
│                              14-> Sens_rot                           │
└──────────────────────────────────────────────────────────────────────┘
}
Function  Sens_rot( Pos_xc, Pos_yc :Real ): Real ;
Var Slope, pente :real;
Begin

 If (Nb_Route = 1) Then
  If (Pos_xc <> x_depart) then
  begin
   slope := Arctan ( (Pos_yc - y_depart) / ( Pos_xc - x_depart) );
   Sens_rot := OR_ROUTE ( Pos_xc, Pos_yc ) - slope ;
  end;

   If (Nb_Route = 2) then
   Begin
      slope := Atan2 (Pos_xc - X_Point[Point_de_depart],
                      Pos_yc - Y_Point[Point_de_depart]);
      Sens_rot := OR_ROUTE ( Pos_xc, Pos_yc ) - slope;
   End;
End;
{
┌──────────────────────────────────────────────────────────────────────┐
│                        Err_dist : Erreur de Distance                 │
└──────────────────────────────────────────────────────────────────────┘
}
Function  Err_dist( Pos_Xc, Pos_yc,sens : Real ) : Real ;
{  xi, yi -> coordonnees au temps t du robot  }
const
  eps = 0.0001;
Var m, xj, yj, xi, yi, Route : Real ;
 Begin
   If (Nb_Route = 2) Then { Spline }
    begin
      x_arrive := X_point[Point_de_depart+1];
      y_arrive := Y_point[Point_de_depart+1];
    end;
```

```
      Route := OR_ROUTE(Pos_xc, Pos_yc);

      m := sin(ROUTE) / cos(ROUTE) ;
      xi:= Pos_xc; yi := Pos_yc;

      yj := (y_arrive + (Sqr(m) * yi) -
            (m * (x_arrive - xi))) / (1 + Sqr(m)) ;
      xj := (xi + (Sqr(m) * x_arrive) +
            (m * (yi - y_arrive)) ) / (1 + Sqr(m)) ;
    If ( Pos_yc <= y_depart ) then
      begin
        Err_dist := Sqrt(Sqr(xi-xj) + Sqr(yi-yj));
      end
    Else   { Pos_yc > y_depart }
      Begin
        if( (-eps <sens) and (sens<eps)) then
          Err_dist := 0.00
        else if ( sens >= eps) then
          Err_dist := Sqrt(Sqr(xi-xj) + Sqr(yi-yj))
        else
          Err_dist := -Sqrt(Sqr(xi-xj) + Sqr(yi-yj));
      End;

 End ;

{
┌──────────────────────────────────────────────────────────────┐
│                  ERR_ORIENT : Erreur d"orientation           │
└──────────────────────────────────────────────────────────────┘
}
Function  ERR_ORIENT( Pos_xc,Pos_yc,orient_robot : Real ) : Real ;
Begin
  Err_orient :=  OR_ROUTE (Pos_xc,Pos_yc) - Orient_Robot   ;
End;
{
┌──────────────────────────────────────────────────────────────┐
│                  ERR_TOTAL  : Erreur totale                  │
└──────────────────────────────────────────────────────────────┘
}
Function  ERR_TOTAL( orient_robot, Pos_xc, Pos_yc ,sens: Real ) : Real ;
Begin

 Err_total :=  (1 - Beta) * Err_dist(Pos_xc,Pos_yc,sens) +
               Beta * Err_orient (Pos_xc, Pos_yc,orient_robot) ;
End ;
{
┌──────────────────────────────────────────────────────────────┐
│                  DELTA_U                                     │
```

```
Function  Delta_U ( Err_total: Vecteur; Int_err_tot,K1, K2, K3 : Real  ) : Real ;
Var Der_Err_tot, Per_ech : Real;
Begin
 Per_ech := Delta_T;
 Der_Err_tot := ( Err_total[tn] - Err_total[tn-1] )/ Per_ech;
 Delta_U := (K1 * Err_total[tn]) + (K2 * Der_Err_tot) + (K3 * Int_Err_tot) ;
End ;

{
```

OR_ROBOT

```
}
Function  OR_ROBOT( r1, r2, L, s1, s2 : Real    ;
                    orient_robot      : Vecteur ;
                    vit               : Couple ) : Real ;

Var  Per_ech, Vit_Orient, Previous_Or_robot, X12, X22: Real ;
Begin
  Per_ech := Delta_T    ;
  X12     := Vit[1][tn] ;
  X22     := Vit[2][tn] ;

  Vit_Orient := (1/L) * ( (r2 * s2 * X22 ) - (r1 * s1 * X12 ) )  ;
  OR_ROBOT :=  Orient_robot[tn-1] + Vit_orient * Per_ech  ;
End ;
{
```

Pos_Xc

```
}
Function Pos_Xc( r1, r2, L, s1, s2 : Real    ;
                      orient_robot : Vecteur ;
                             vit   : Couple  ;
                  Previous_Pos_Xc : Real )  : Real ;

Var  Vit_trans, Per_ech, X12, X22: Real ;
Begin
 Per_ech := Delta_T;
 X12     := Vit[1][tn];
 X22     := Vit[2][tn];

 Vit_trans := (1/2) * ( ( r1 * s1 * X12 ) + ( r2 * s2 * X22 ) );
 Pos_Xc    := Previous_Pos_Xc +
            ( Vit_trans * cos (orient_robot[tn]) * Per_ech ) ;
End ;
{
```

Pos_Yc

```
Function Pos_Yc( r1, r2, L, s1, s2 : Real    ;
                      orient_robot : Vecteur ;
                             vit   : Couple  ;
                   Previous_Pos_Yc : Real )  : Real ;

Var  Vit_trans, Per_ech, X12, X22: Real ;
Begin
 Per_ech := Delta_T;
 X12     := Vit[1][tn];
 X22     := Vit[2][tn];

 Vit_trans := (1/2) * ( ( r1 * s1 * X12 ) + ( r2 * s2 * X22 ) );
 Pos_Yc    := Previous_Pos_yc +
            ( Vit_trans * sin (orient_robot[tn]) * Per_ech ) ;
End ;
{
```

OR_ROBOT_ESTIME

```
}
Function  OR_ROBOT_ESTIME( rayon_moyen, Largeur_estime : Real    ;
                           orient_robot                : Vecteur ;
                           vit                         : Couple ) : Real ;

Var  Per_ech, vit_orient,X12, X22: Real ;
Begin
  Per_ech := Delta_T    ;
  X12     := Vit[1][tn] ;
  X22     := Vit[2][tn] ;

  Vit_Orient := (1/Largeur_estime) * ((rayon_moyen*X22) - (rayon_moyen*X12));
  OR_ROBOT_ESTIME   :=  Orient_robot[tn-1] + Vit_orient * Per_ech  ;
End ;
{
```

Pos_Xc_estime

```
}
Function Pos_Xc_estime ( rayon_moyen,largeur_estime    : Real ;
                         orient_robot    : Vecteur ;
                         vit             : Couple  ;
                         Previous_Pos_Xc : Real )  : Real ;

Var  Vit_trans, Per_ech, X12, X22: Real ;
Begin
 Per_ech := Delta_T;
 X12     := Vit[1][tn];
 X22     := Vit[2][tn];

 Vit_trans      := (1/2) * ( ( rayon_moyen * X12 ) + ( rayon_moyen * X22 ) );
 Pos_Xc_estime  := Previous_Pos_Xc +
                   ( Vit_trans * cos (orient_robot[tn]) * Per_ech ) ;
End ;
```

```
┌──────────────────────────────────────────────────────────────────────┐
│                            Pos_Yc_estime                             │
└──────────────────────────────────────────────────────────────────────┘

)
Function Pos_Yc_estime ( rayon_moyen, largeur_estime : Real    ;
                         orient_robot                : Vecteur ;
                         vit                         : Couple  ;
                         Previous_Pos_Yc             : Real )  : Real ;

Var  Vit_trans, Per_ech, X12, X22: Real ;
Begin
 Per_ech := Delta_T;
 X12     := Vit[1][tn];
 X22     := Vit[2][tn];

 Vit_trans := (1/2) * ( ( rayon_moyen * X12 ) + ( rayon_moyen * X22 ) );
 Pos_Yc_estime  := Previous_Pos_yc +
                   ( Vit_trans * sin (orient_robot[tn]) * Per_ech ) ;
End ;

{
┌──────────────────────────────────────────────────────────────────────┐
│                             Dynamique                                │
└──────────────────────────────────────────────────────────────────────┘

)
{ Calcul de la vitesse de la roue gauche }
Function Dynamique_1( X12, tau1, km1, u1 : Real ) : Real ;
Var
 Per_ech: Real;
Begin
  Per_ech  := delta_t;
  F[2,2,1] := Exp (- Per_ech / Tau1 );
  B[2,1]   := km1 * ( 1- F[2,2,1]);
  Dynamique_1 :=  F[2,2,1] * X12 + B[2,1] * U1;
End ;  { Output = teta_pt[1]  i.e : X[1,2]  }

{ Calcul de la vitesse de la roue droite }
Function Dynamique_2( X22, tau2, km2, u2 : Real ) : Real ;
Var
 Per_ech: Real;
Begin
  Per_ech  := delta_t;
  F[2,2,2] := Exp (- Per_ech / Tau2 );
  B[2,2]   := km2 * ( 1- F[2,2,2]);
  Dynamique_2 :=  F[2,2,2] * X22 + B[2,2] * U2;
End ;  { Output = teta_pt[2]  i.e : X[2,2]  }

 (*************************************************************************)
             { Calcul de la Position angulaire }

{ Calcul de la position  de la roue gauche }
Function Position_1( X11, X12, tau1, km1, u1 : Real ) : Real ;
```

```
Var
 Per_ech: Real;
Begin
  Per_ech  := delta_t;
  F[2,2,1] := Exp (- Per_ech / Tau1 );
  F[1,2,1] := (1 + F[2,2,1]) * Per_ech;
  B[1,1]   := km1 * ( Per_ech - F[1,2,1]);
  Position_1 :=  X11 + F[1,2,1] * X12 + B[1,1] * U1;
End ;  { Output = teta[1]  i.e : X[1,1]  }

{ Calcul de la position  de la roue droite }
Function Position_2( X21, X22, tau2, km2, u2 : Real ) : Real ;
Var
 Per_ech: Real;
Begin
  Per_ech  := delta_t;
  F[2,2,2] := Exp (- Per_ech / Tau2 );
  F[1,2,2] := (1 + F[2,2,2]) * Per_ech;
  B[1,2]   := km2 * ( Per_ech - F[1,2,2]);
  Position_2 :=  X21 + F[1,2,2] * X22 + B[1,2] * U2;
End ;  { Output = teta[2]  i.e : X[2,1]  }

{
```

U_MED

```
}
Function  U_MED( Vit_desiree,Err_Vit, Int_err_vit, k21, k23, k24 : Real ) : Real ;

Begin
  U_MED := k21 * Err_vit + K23 * (Int_err_vit - (K24 * Vit_desiree) );
End ;
{
```

Calcul_depart

```
}
Procedure Calcul_depart( X1, Y1, alfa1, De : Real;
                         Var  Kdx: Real;
                         Var  Kdy: Real);

Var   Alfa2 :Real;
Begin
   Alfa2:= (Pi/2) - Alfa1;
   kdx  := De * cos (Alfa2);
   kdy  := De * sin (Alfa2);
 End;

{
```

La_commande

```
Procedure La_commande( du          : Un_set ;
                       U_m         : Un_set ;
                       Vit_actuel  : real   ;
                       Int_err_vit : Real   ;
                       PC          : TparmsControle;
                       Var u       : Un_set ) ;
Var
    I  : Integer;
    slm, M, a1, epsilon, sc : Real ;
    delta_U, U_MED  : Real ;

Begin

 delta_U   := ( dU[1] + dU[2] )/ 2;
 U_med     := ( U_m[1]+ U_m[2])/ 2;


(*  ******************   Calcul du Switch Component ********************)

  a1 := 0.5;
  epsilon := 0.1;

  slm := ( Vit_actuel * a1 ) - Int_err_vit;

  If ( (slm > -epsilon)  and  (slm < epsilon) ) then
     sc := 0.0                      { seuil }
  Else If ( slm >= epsilon ) then
         sc := - SC_M
       else
         sc :=  SC_M ;

  { La commande complete }

     U_MED := U_MED + SC ;  { SC = Switch Component }

  (************** Delai = 0 ***********************************************)

    U[1]:= ( U_MED - Delta_U) + PC.RG.pert; { RG.pert = perturbation gauche}
    U[2]:= ( U_MED + Delta_U) + PC.RD.pert; { RD.pert = perturbation droite}

  (************** Delai non Nul *******************************************)

  If (Ndelay1 > 0 ) then
   Begin
    For I:= 1 to Ndelay1-1 Do Xdelay1[Ndelay1-I+1] := Xdelay1[Ndelay1-I];
    Xdelay1[1] := U[1];
    U[1] := Xdelay1[Ndelay1];
  End;

 If (Ndelay2 > 0 )    then
  Begin
```

```
    For 1:= 1 to Ndelay2-1 Do Xdelay2[Ndelay2-1+1] := Xdelay2[Ndelay2-I];
    Xdelay2[1] := U[2];
    U[2] := Xdelay2[Ndelay2];
  End;

End ;

{
┌──────────────────────────────────────────────────────────────────────┐
│                              16-> Batit_vect_a_parcourir             │
└──────────────────────────────────────────────────────────────────────┘
}
Procedure Batit_vect_a_parcourir( x_u, y_u : Vect_a_parcourir ;
                                  Var x_r  : Vect_a_relier ;
                                  Var y_r  : Vect_a_relier ;
                                  nb_seg   : Integer ) ;


{-----------------------------------------------------------------------}
{-                                                                      -}
{-     Turbo Pascal Numerical Methods Toolbox                           -}
{-     (C) Copyright 1986 Borland International.                        -}
{-                                                                      -}
{-       Purpose: This program demonstrates interpolation with          -}
{-                a free cubic spline.                                  -}
{-                                                                      -}
{-  Include Files:  COMMON.INC        procedure DisplayWarning          -}
{-                                    procedure DisplayError            -}
{-                                    procedure IOCheck                 -}
{-                                    procedure GetOutputFile           -}
{-                  CUBE_FRE.INC      procedure CubicSplineFree         -}
{-                                                                      -}
{-   Version Date: 26 January 1987                                      -}
{-                                                                      -}
{-----------------------------------------------------------------------}

{$I-}                   { Disable I/O error trapping }
{$R+}                   { Enable range checking }

var
  XData, YData : TNvector;                  { Data points (X,Y) }
  NumPoints : integer;                      { Number of data points }
  Coef0, Coef1, Coef2, Coef3 : TNvector;    { Coefficients of the spline }
  NumInter : integer;                       { Number interpolating points }
  XInter, YInter : TNvector;                { Interpolating points }
  Error : byte;                             { Flags an error }
  d, xx, yy : Real ;
  teta, delta : real ;
  i, j, k, n : integer ;
```

```
procedure Initialize(var Coef0     : TNvector;
                     var Coef1     : TNvector;
                     var Coef2     : TNvector;
                     var Coef3     : TNvector;
                     var XData     : TNvector;
                     var YData     : TNvector;
                     var NumPoints : integer;
                     var NumInter  : integer;
                     var XInter    : TNvector;
                     var YInter    : TNvector;
                     var Error     : byte);

{-------------------------------------------------------------}
{- Output: Coef0, Coef1, Coef2, Coef3, XData, YData, NumPoints, -}
{-         NumInter, XInter, YInter, Error                      -}
{-                                                             -}
{- This procedure initializes the above variables to zero.     -}
{-------------------------------------------------------------}

begin
  FillChar(Coef0, SizeOf(Coef0), 0);
  FillChar(Coef1, SizeOf(Coef1), 0);
  FillChar(Coef2, SizeOf(Coef2), 0);
  FillChar(Coef3, SizeOf(Coef3), 0);
  FillChar(XData, SizeOf(XData), 0);
  FillChar(YData, SizeOf(YData), 0);
  FillChar(XInter, SizeOf(XInter), 0);
  FillChar(YInter, SizeOf(YInter), 0);
  NumPoints := 0;
  NumInter := 0;
  Error := 0;
end; { procedure Initialize }



Begin

  Initialize(Coef0, Coef1, Coef2, Coef3, XData, YData, NumPoints,
             NumInter, XInter, YInter, Error);

  If nb_seg = 2 then
    Begin
      {NumPoints := Nb_point_a_relier - 2 ;}
      NumPoints  := Nb_point_a_relier - 4 ;

      { Initialisation des points connus }
      XData[1] := x_u[1] ;
      YData[1] := y_u[1] ;

      xx := x_u[2] - x_u[1] ;
      yy := y_u[2] - y_u[1] ;
```

```
      d := sqrt( ( xx * xx ) + ( yy * yy ) );
      teta := atan2( x_u[2] - x_u[1], y_u[2] - y_u[1]) ;  { pente seg 1 }
      XData[2] := x_u[1] + ( 0.90 * d * cos(teta) ) ;
      YData[2] := y_u[1] + ( 0.90 * d * sin(teta) ) ;

      xx := x_u[3] - x_u[2] ;
      yy := y_u[3] - y_u[2] ;
      d := sqrt( ( xx * xx ) + ( yy * yy ) );
      teta := atan2( x_u[3] - x_u[2], y_u[3] - y_u[2]) ;  { pente seg 2 }
      XData[3] := x_u[2] + ( 0.10 * d * cos(teta) ) ;
      YData[3] := y_u[2] + ( 0.10 * d * sin(teta) ) ;
      XData[4] := x_u[2] + ( 0.90 * d * cos(teta) ) ;
      YData[4] := y_u[2] + ( 0.90 * d * sin(teta) ) ;
    {
      xx := x_u[4] - x_u[3] ;
      yy := y_u[4] - y_u[3] ;
      d := sqrt( ( xx * xx ) + ( yy * yy ) );
      teta := atan2( x_u[4] - x_u[3], y_u[4] - y_u[3]) ;
      XData[5] := x_u[3] + ( 0.10 * d * cos(teta) ) ;
      YData[5] := y_u[3] + ( 0.10 * d * sin(teta) ) ;

      XData[6] := x_u[4] ;
      YData[6] := y_u[4] ;
    }
     {  NumInter := 2 ;}
      NumInter  := 1 ;

      XInter[1] := x_u[2] ;
     {  XInter[2] := x_u[3] ;}

      CubicSplineFree(NumPoints, XData, YData,NumInter, XInter,
                      Coef0, Coef1, Coef2, Coef3, YInter, Error);

      x_r[1] := XData[1] ;              y_r[1] := YData[1] ;
      x_r[2] := XData[2] ;              y_r[2] := YData[2] ;
      x_r[3] := XInter[1] ;             y_r[3] := YInter[1] ;
      x_r[4] := XData[3] ;              y_r[4] := YData[3] ;
      x_r[5] := XData[4] ;              y_r[5] := YData[4] ;
    {
      x_r[6] := XInter[2] ;             y_r[6] := YInter[2] ;
      x_r[7] := XData[5] ;              y_r[7] := YData[5] ;
      x_r[8] := XData[6] ;              y_r[8] := YData[6] ;
    }
    End ;
End ;


end.   { fin de implementation }
```

APPENDICE B

Dans cet appendice nous présentons d'une façon plus approfondie un certain nombre de choses qui bien qu'importantes pour notre développement auraient peut être nui à la fléxibilité du texte si elles étaient insérées dans sa partie centrale. Il s'agit en particulier de la description des capteurs et la génération des trajectoires en courbe.

b.1 Présentation des capteurs

b.1.1 L'encodeur optique

Le sous-module dead-reckoning dont le rôle principal consiste à l'estimation de la position du robot lors de son déplacement utilise comme outil les données odométriques. La figure b.1 illustre brièvement le principe du dead-reckoning.

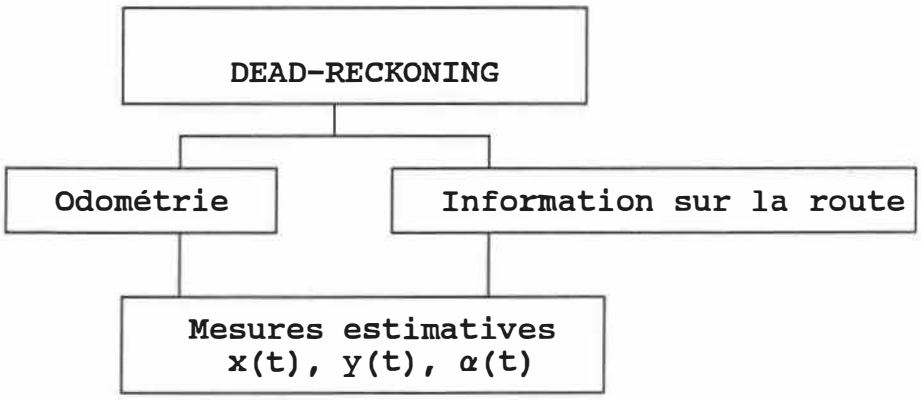


Figure b.1 : Principe du dead-reckoning

Ume mesure odométrique se fait à l'aide d'un encodeur optique permettant de mesurer les déplacements angulaires d'un arbre en rotation. Tel que montré sur la figure b.2 l'encodeur optique est constitué d'un disque circulaire possédant des zônes opaques et des zônes transparentes. Ce disque est solidaire à l'arbre en rotation. Un côté du disque est muni d'une photo-diode émettant une lumière. Lorsque le disque tourne, un signal binaire est généré et le nombre de changements de niveaux de signal indique le déplacement angulaire de l'arbre.

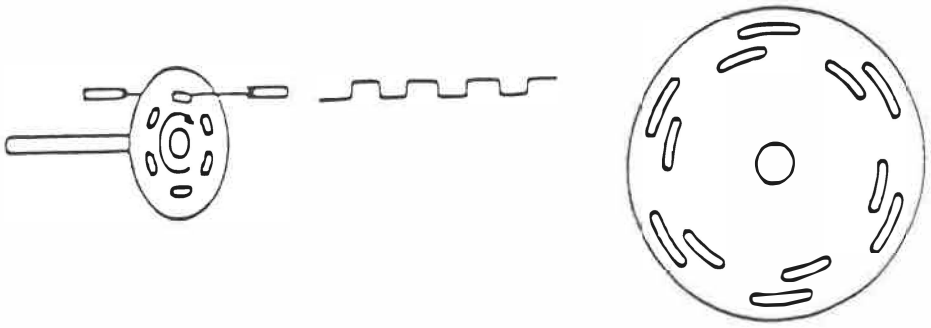


Figure b.2 : Encodeur optique Sandtron REX-44-5

b.1.2 Les capteurs Ultrason

Les capteurs à Ultrason installés à bord du véhicule sont utilisés comme mesure de distance entre le véhicule et l'environnement. Tel qu'illustré à la figure a.3, le principe de fonctionnement est simple. Il suffit de mesurer le temps écoulé entre l'émission et la réception du signal après reflexion. Ces capteurs sont caractérisés par le fait qu'une même pastille sert à la fois d'émetteur et de récepteur. Le signal d'exitation est d'une durée de 1 msec et est composé de 56 périodes d'un signal sonore comprenant 4 fréquences consécutifs allant de 50 à 60 Khz. Cette caractéristique du signal permet de réduire la probabilité d'une absence de réflexion causée par la configuration d'un obstacle et du capteur.

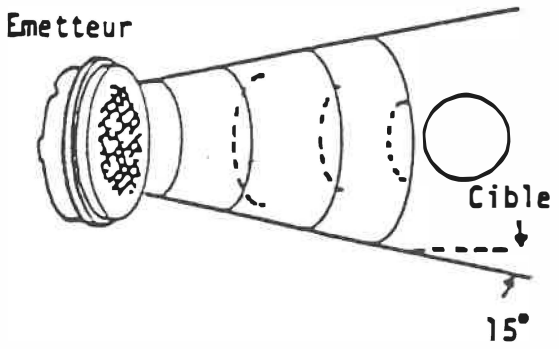


Figure b.3 : Principe de détection des capteurs Ultrason

Le diagramme de rayonnement du signal à une fréquence de 50 Khz est montré à la figure b.4

Tel qu'illustré le signal montre une atténuation de 3 db pour un angle de plus ou moins 7.5 degrés. Cette caractéristique du faisceau est liée à la géométrie de la pastille (diamètre de 38 mm ou 1.5 po). Ceci nous indique la directivité du capteur ultrason.

Compte tenu des caractéristiques du circuit électronique, ce capteur permet de mesurer des distances variant de 0.3 à 10.6 mètres (0.9 à 35 pi) avec une résolution de plus ou moins 30 mm (1.2 po). Un circuit numérique donne un affichage des valeurs 5 fois par seconde, donc avec une période de 200 msec.

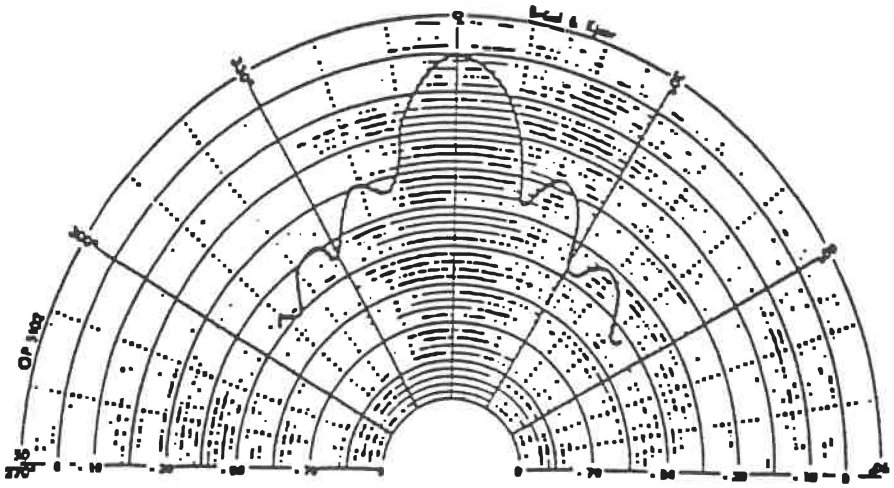


Figure b.4 : Diagramme de rayonnement du sonar

b.1.3 Les capteurs infra-rouges

La dernière série de capteurs est un interrupteur infra-rouge illustré à la figure b.5

Ce genre de capteur est installé sur le coté du véhicule: 1 capteur de chaque côte; Ce capteur émet un rayon infra-rouge qui réfléchit à 90 degré après avoir atteint le reflecteur. Cependant la distance maximale d'opération n'est que de 3 métres. Ces reflecteurs sont utilisés comme ds balises pour mesurer la localisation absolue du véhicule à des endroits précis du parcours.

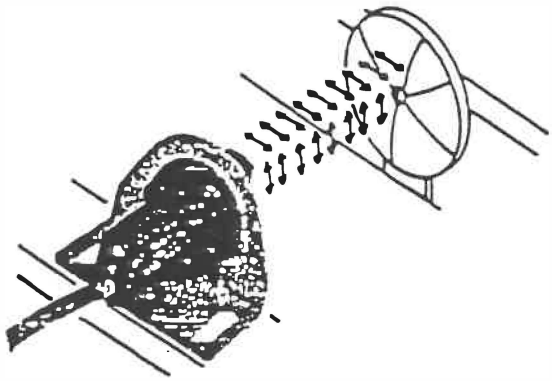


Figure b.5 : Interrupteur infra-rouge

b.2 Les autres types de trajectoire

Une attention particulière a été donnée à la solution des deux modules relatifs à la trajectoire : (a) la détermination de la succession de la route et (b) l'utilisation des courbes β-splines.

Remarque 1 : Détermination de la succession de la route

Le problème de la succession de la route se résume à déterminer le prochain point d'itération lors de la navigation.

Sur la figure a.6 la première portion de l'axe de la route est représentée par le segment P_{ck}- P_{ck+1}; la seconde portion par P_{ck+1}- P_{ck+2}; Dans le cas où le point de l'iteration suivante dépasse la route: P_s (x_s,y_s) est inadmissible (en dehors de la route) et on considère la ligne ℓ_{k+1} perpendiculaire au premier segment. La projection verticale de P_s coupe respectivement la ligne ℓ_{k+1} et le 2e segment en $P_{\ell_1}(x_{\ell_1},y_{\ell_1})$ et $P_{\ell_2}(x_{\ell_2},y_{\ell_2})$ avec $x_{\ell_1} = x_{\ell_2} = x_s$.

g_k et g_{k+1} représentent les gradients du $k^{ième}$ et $(k+1)^{ième}$

segments de la route. À cette méthode s'ajoute celle utilisée pour changer d'orientation et tourner lors des intersections de route: les B-splines

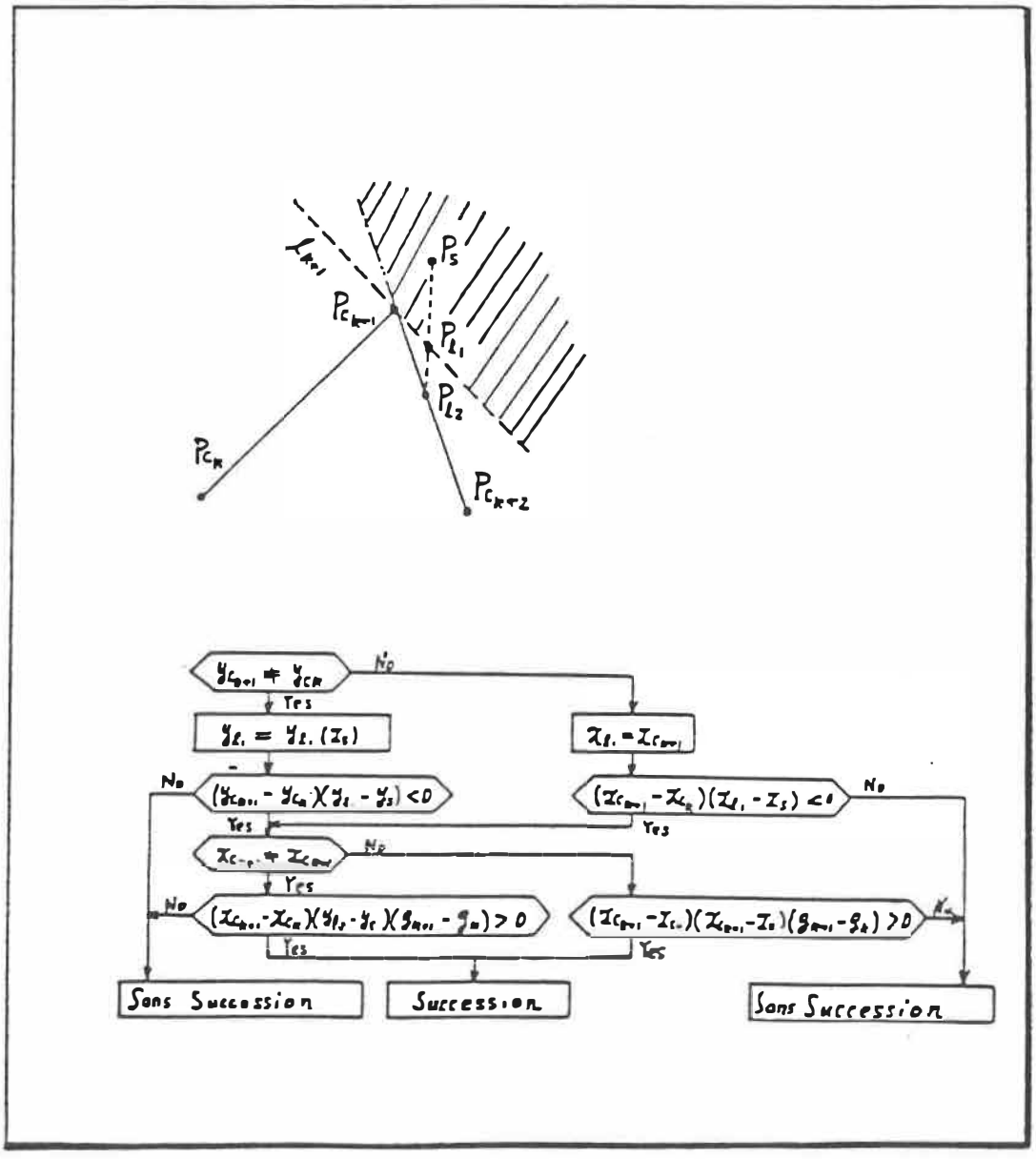


Figure b.6 : Détermination de la succession de la route

Remarque 2 : Utilisation des β-splines.

La route peut-être droite ou en courbe avec un angle de raccordement arrondi par la technique de la spline.

Les courbes β - splines se définissent par un certain nombre de points appelés points de contrôle. La principale caractéristique de ces courbes est qu'elles ne passent pas par les points de contrôle à part le premier et le dernier.

Les B-splines se définissent de manière similaire aux courbes de Bézier par la relation suivante:

$$B(t) = \sum_{i=0}^{n} P_i N_{i,k}(t)$$

ou k est l'ordre des B-splines et P les n+1 points de contrôle. Les $N_{i,k}$ sont définis récursivement comme:

$$N_{i,1}(t) = \begin{cases} 1 \text{ si } x_i \leq t \leq x_{i+1} \\ 0 \text{ autrement} \end{cases}$$

$$N_{i+k}(t) = \frac{(t-x_i)N_{i,k-1}(t)}{x_{i+k-1} - x_i} + \frac{(x_{i+k}-t)\, N_{i+1,k-1}(t)}{x_{i+k} - x_{i+1}}$$

On prend la convention 0/0 = 0.

Alors que pour les courbes de Bézier, t variait entre 0 et 1, pour les B-splines, il est défini par le vecteur X si des sommets sont multiples, on duplifie le x_i

p.e. x = [0 1 2 2 3 4 4 5]

Pour ce faire le programme fait appel à un module prédéfini dans le " Numerical Tool Box "; Toutefois l'initialisation de la courbe ainsi que l'interface de la route avec le " Tool Box " relève de la tâche du sous-module que nous appelons Courbe_spline (figure b.7). La figure ci-dessous montre l'effet de la courbe spline sur la route.

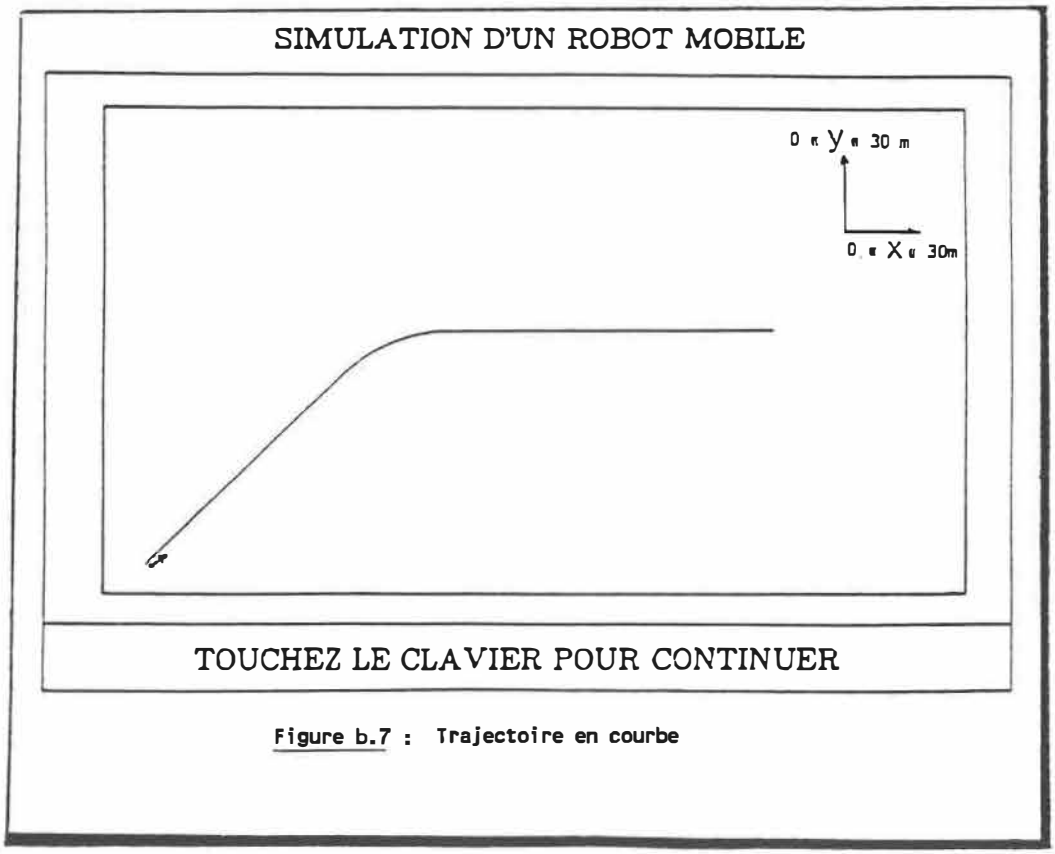


Figure b.7 : Trajectoire en courbe

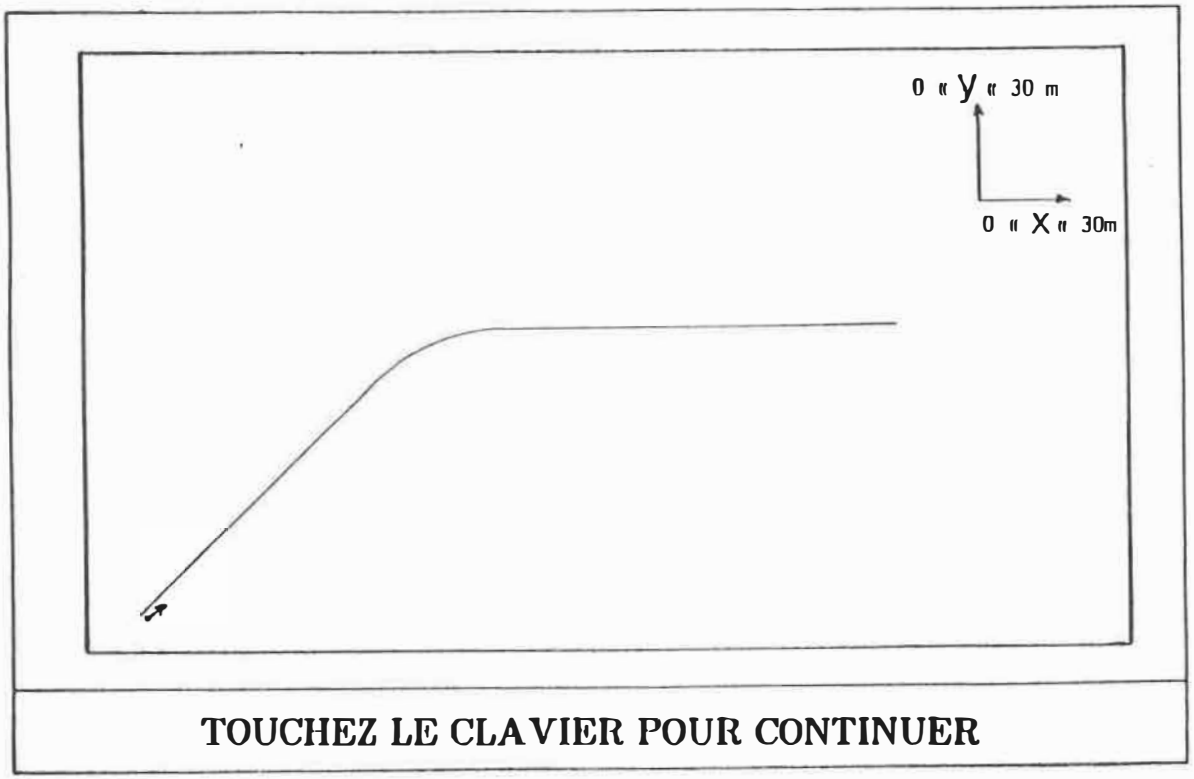


Figure b.8: Type de trajectoire # 1

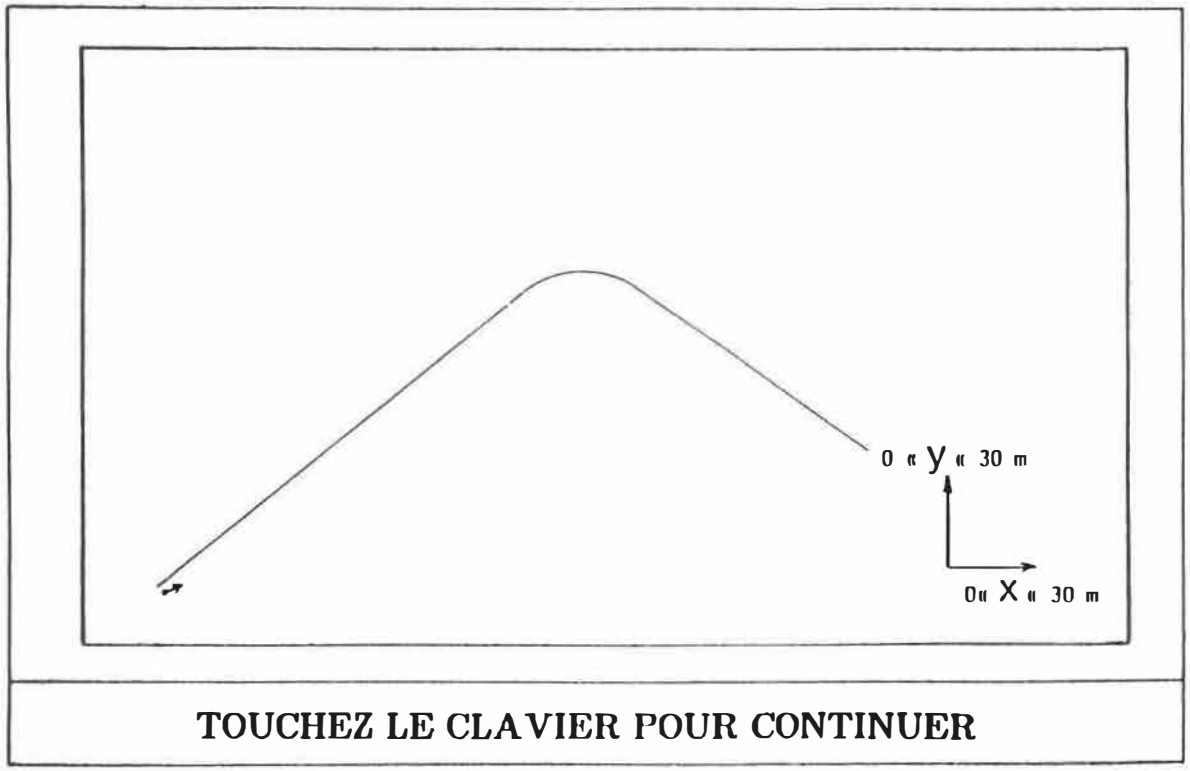


Figure b.9: Type de trajectoire # 2

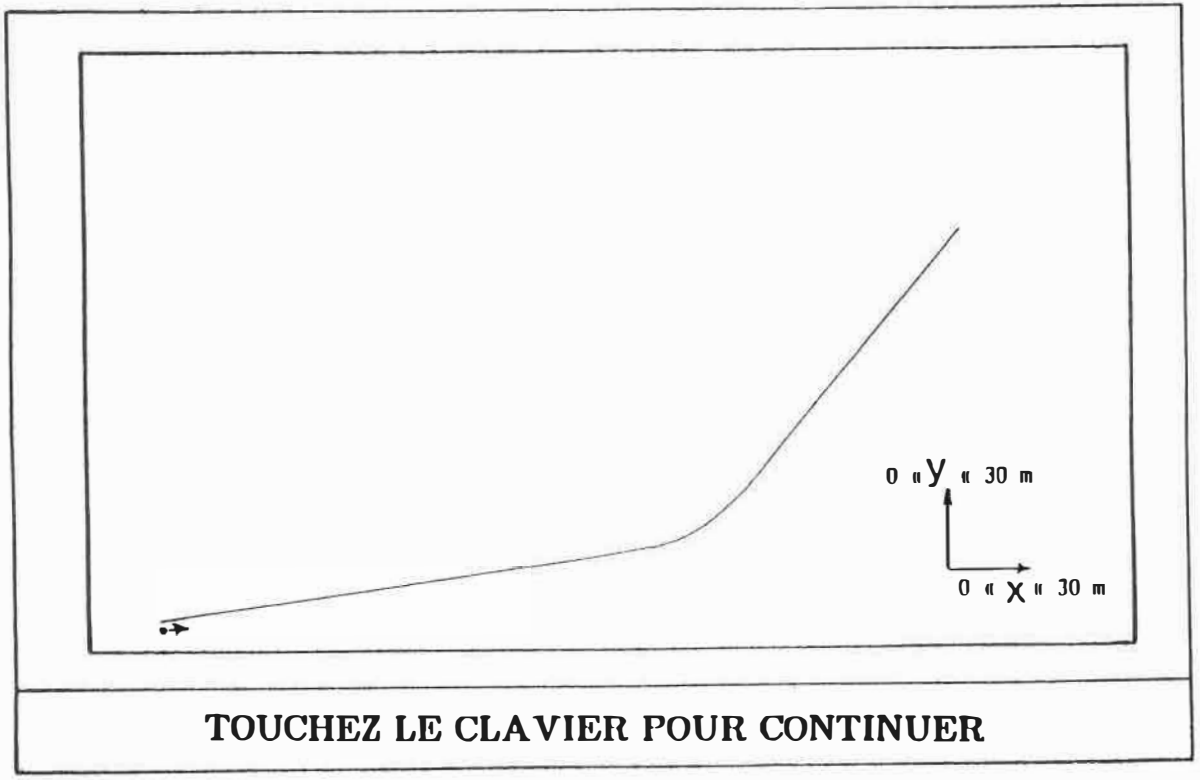


Figure b.10: Type de trajectoire # 3

b.3 Simulations sur trajectoire en courbe

Deux séries de simulation à différentes vitesses ont été effectuées sur le type de trajectoire de la figure b.8; Pour ce faire, nous avons adopté les valeurs de paramètres présentés au tableau b.1 pour la faible vitesse (2 m/s) et et ceux du tableau b.2 pour la haute vitesse (10 m/s).

Les courbes de simulation se trouvent respectivement aux figures b.11 @ b.14; les résultats confirment les observations déja mentionnées au chapître 4 particulièrement les courbes de vitesse où les trois courbes relatives aux différentes erreurs de position initiale restent confondues.

PRESENTATION DES PARAMETRES DE CONTROLE

<1>	Vitesse desiree	0 < Vd <= 10 (m/s)	Vd = 2.00
	Constante de temps :		
<2>	Moteur gauche	0.1 < TAU1	TAU1 = 0.50
<3>	Moteur droite	0.1 < TAU2	TAU2 = 0.50
	Gain statique :		
<4>	Moteur gauche	0 < KM1	KM1 = 12.00
<5>	Moteur droite	0 < KM2	KM2 = 12.00
	Delai :		
<6>	Moteur gauche	0<= DL1 <=100 msec	DL1 = 0.00
<7>	Moteur droite	0<= DL2 <=100 msec	DL2 = 0.00
	Perturbation :		
<8>	Moteur gauche	0 < PM1 <= 2000 mv	PM1 = 0.00
<9>	Moteur droite	0 < PM2 <= 2000 mv	PM2 = 0.00

Tapez le numero correspondant pour changer les valeurs ou <ESC> pour terminer.

->

MODIFICATION DES GAINS DES CONTROLEURS

Parametres de Position

Controleur PID :

<A> K11 = 6.75 <B> K12 = 0.67 <C> K13 = 15.12

Parametres de Vitesse

Controleur PI :

<G> K21 = 0.50 <H> K23 = 1.20 <I> K24 = 0.00

Faites votre selection en tapant la lettre de votre choix ou tapez <ESC> pour terminer.

Tableau b.1 : Valeurs des parametres pour la faible vitesse

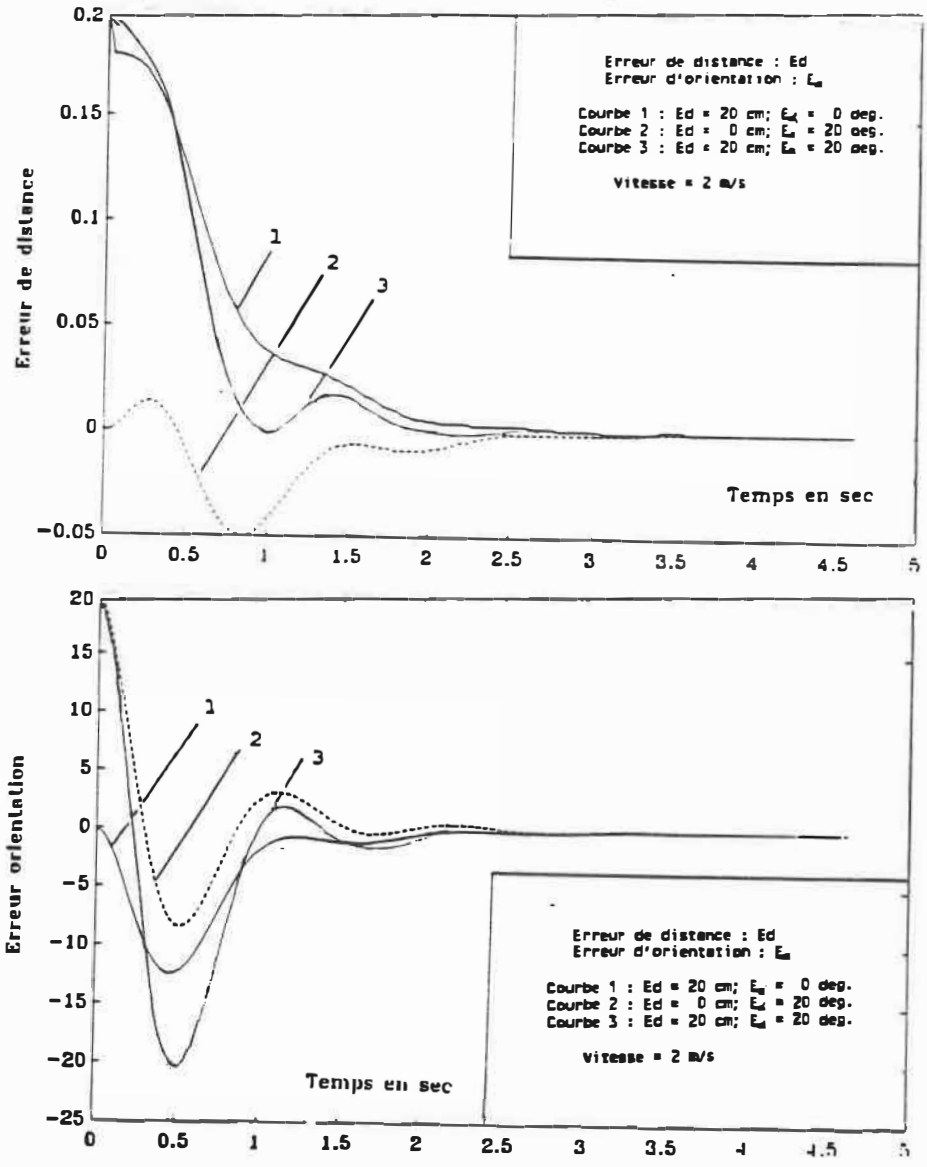


Figure b.11: Effet des erreurs initiales a faible vitesse

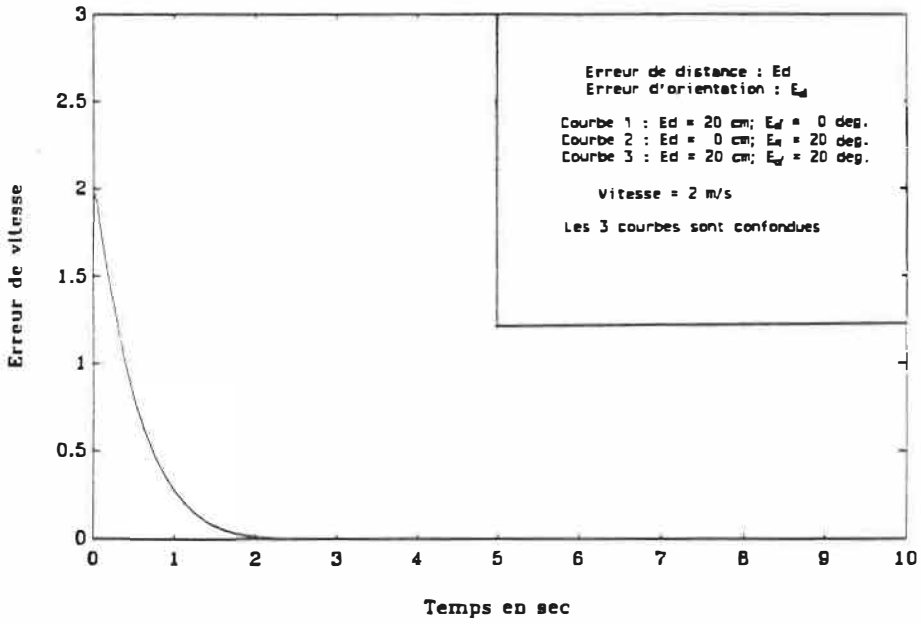


Figure b.12 : Courbe de vitesse (v = 2 m/s)

PRESENTATION DES PARAMETRES DE CONTROLE

<1>	Vitesse desiree	0 < Vd <= 10 (m/s)	Vd = 10.00
	Constante de temps :		
<2>	Moteur gauche	0.1 < TAU1	TAU1 = 0.50
<3>	Moteur droite	0.1 < TAU2	TAU2 = 0.50
	Gain statique :		
<4>	Moteur gauche	0 < KM1	KM1 = 12.00
<5>	Moteur droite	0 < KM2	KM2 = 12.00
	Delai :		
<6>	Moteur gauche	0<= DL1 <=100 msec	DL1 = 0.00
<7>	Moteur droite	0<= DL2 <=100 msec	DL2 = 0.00
	Perturbation :		
<8>	Moteur gauche	0 < PM1 <= 2000 mv	PM1 = 0.00
<9>	Moteur droite	0 < PM2 <= 2000 mv	PM2 = 0.00

Tapez le numero correspondant pour changer les valeurs ou <ESC> pour terminer.

->

MODIFICATION DES GAINS DES CONTROLEURS

Parametres de Position

Controleur PID :

<A> K11 = 7.50 <B> K12 = 1.20 <C> K13 = 15.12

Parametres de Vitesse

Controleur PI :

<G> K21 = 0.50 <H> K23 = 1.20 <I> K24 = 0.00

Faites votre selection en tapant la lettre de votre choix ou tapez <ESC> pour terminer.

Tableau b.2 : Valeurs des parametres pour la haute vitesse

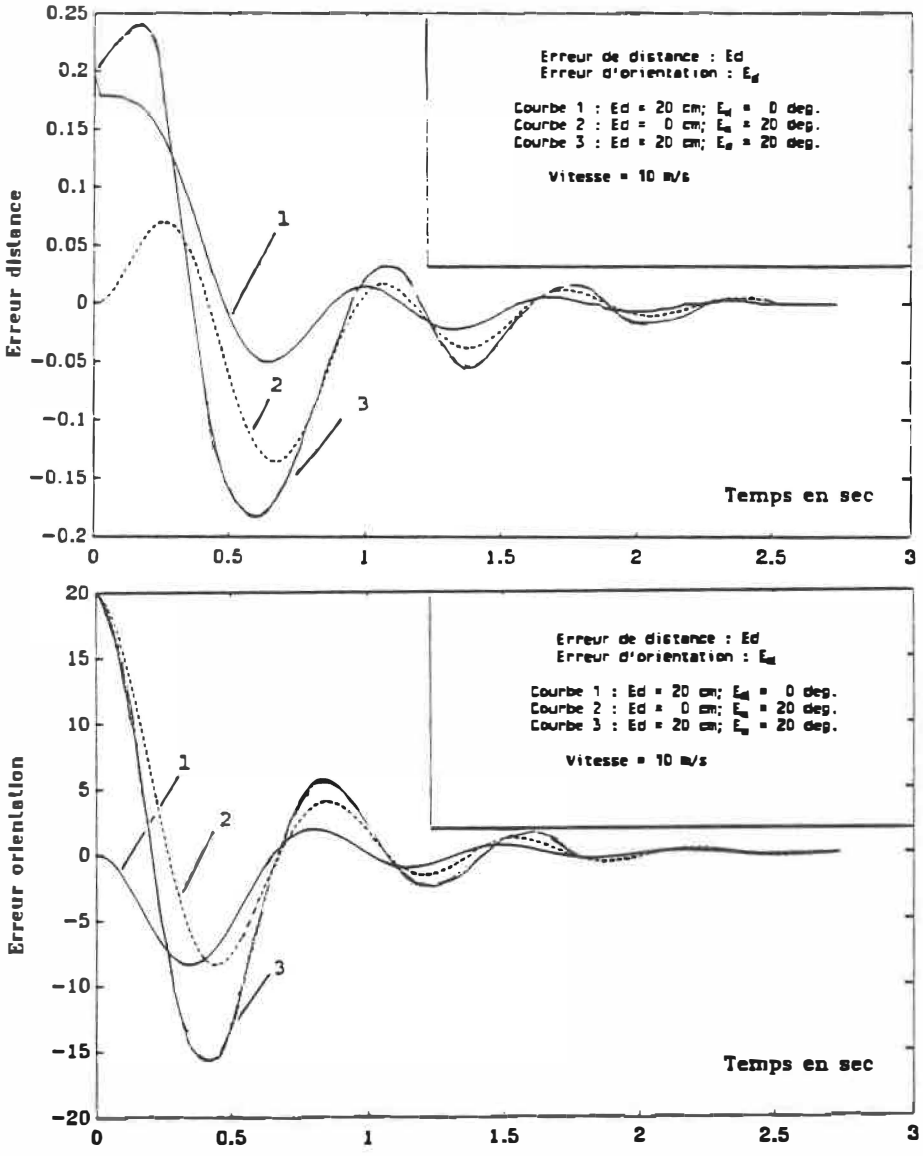


Figure b.13: Effet des erreurs initiales a faible vitesse

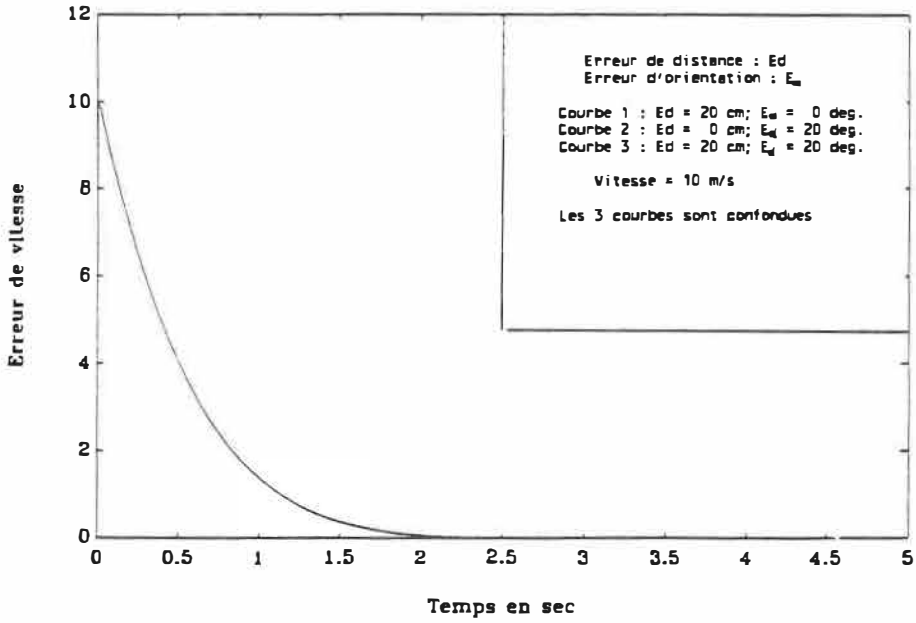


Figure b.14 : Courbe de vitesse (v = 10 m/s)

ÉCOLE POLYTECHNIQUE DE MONTRÉAL
3 9334 00290871 1