N° d'ordre : 08 / 2011-M / EL

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université des Sciences et de la Technologie Houari Boumediene
Faculté d'Electronique et Informatique

MEMOIRE

Présenté pour l'obtention du diplôme de

MAGISTER

En ELECTRONIQUE

Spécialité : Contrôle de Processus et Robotique

Par

M^r^ *TCHENDERLI-BRAHAM Smail Azzeddine*

PLANIFICATION ET COMMANDE EN POURSUITE DE TRAJECTOIRE DE ROBOTS MOBILES DE TYPE VOITURE

Soutenu publiquement le 19 Octobre 2011 devant le jury composé de :

Mme N. SAADIA	Maître de Conférences /A, d'USTHB	Présidente
Mme F. HAMERLAIN	Maître de Recherche / B, du CDTA	Directrice de mémoire
Mlle O. AZOUAOUI	Maître de Recherche /A, du CDTA	Examinatrice
Mr. F. FERGUENE	Maître de Conférences /A, d'USTHB	Examinateur
Mr. M.S. DJOUADI	Maître de Conférences / B à l'EMP	Invité

Remerciements

Le travail présenté dans ce mémoire a été mené au sein de l'équipe « Navigation et Commande des Robots Mobiles » (NCRM) de la Division Robotique et Productique du Centre de Développement des Technologies Avancées (CDTA) d'Alger.

Je tiens à remercier Mme F. Hamerlain pour m'avoir dirigé soutenu et aidé pour réaliser ce mémoire.

Je remercie Mme N.Saadia pour m'avoir fait l'honneur d'accepter de présider le jury et d'examiner mon travail.

Je remercie les membres du jury Mlle O. AZOUAOUI, Mr. F. FERGUENE, Mr. M.S. DJOUADI pour avoir accepté d'examiner mon mémoire.

Dédicaces

Je dédie ce travail à tous ceux qui me sont chers :

Mon père, ma mère, ma sœur, ainsi qu'à toute ma famille et mes amis et sans oublier tous ceux qui m'ont aidé durant ce mémoire.

Merci à tous.

Smail

Sommaire

Chapitre 3

Commande en Poursuite de Trajectoire

Chapitre 4

Simulations et Expérimentations

Liste des figures

Introduction générale

La robotique a beaucoup évoluée ces dernières années, du point de vue technologique et du point de vue des rôles qu'on lui attribut dans la vie de tous les jours. On remarque la présence des robots dans plusieurs domaines comme l'industrie manufacturière, la recherche scientifique, le domaine militaire ou encore dans la médecine etc. Les robots sont utilisés à plusieurs fins, d'une part comme outils de production permettant de remplacer l'homme dans les tâches pénibles, d'autre part, ils permettent d'accélérer la production comme c'est le cas dans l'industrie automobile. Les robots sont aussi utilisés pour la précision qu'ils apportent, par exemple dans le domaine chirurgical, des robots sont pilotés par des médecins pour réaliser des opérations chirurgicales très précises. D'autres tâches sont attribuées au robot comme l'exploration des milieux hostiles à l'homme, comme par exemple dans le domaine militaire ou encore le spatial. Il existe plusieurs catégories de robots : les robots manipulateurs, les robots marcheurs qu'ils soient à pattes ou bipèdes, les robots aériens (exemple : drones), les véhicules marins ou encore les robots mobiles à roues.

Dans ce mémoire, on s'intéresse aux robots mobiles à roues. Ces derniers existent sous plusieurs formes : l'unicycle, le tricycle, les véhicules de type voiture, les robots omnidirectionnels, les robots tractant une ou plusieurs remorques. Ces robots mobiles sont caractérisés par le fait qu'ils soient non holonomes (contrainte cinématique) [Laum.,01], non-linéaires et sous-actionnés (Le nombre d'entrée de commande est inférieur au nombre d'état à commander). Ces caractéristiques sont considérées comme des contraintes, rendant l'utilisation de ces robots difficiles à mettre en œuvre pour leurs permettre d'évoluer en toute sécurité dans les différents types d'environnements existants. Des recherches ont été entreprises pour l'étude de ces robots dans le but de les rendre autonomes. Les principaux axes de recherches dans le domaine de la robotique mobile concernent la localisation, la planification du mouvement, la navigation et la commande.

La localisation permet au robot mobile de se localiser dans l'environnement dans lequel il évolue en s'aidant de ses capteurs. En effet, en utilisant les informations récoltées par ses capteurs proprioceptifs (exemple : les capteurs odométriques) et ses capteurs extéroceptifs (exemple : caméra, GPS), le robot peut connaître sa position avec plus ou moins d'erreurs. On peut distinguer deux sortes de localisation : la localisation relative (en relation avec les informations obtenues grâce aux capteurs odométriques) et la localisation absolue connue pour apporter plus de précision que la précédente [Bour.,07].

La planification du mouvement est utilisée en robotique mobile pour évaluer un chemin ou une trajectoire qui permet au robot de se déplacer d'un point initial à un point final. Le planificateur prend en considération plusieurs types de contraintes, qu'elles soient liées au robot ou à l'environnement de ce dernier. Plusieurs méthodes de planification ont été élaborées pour s'adapter aux différents environnements dans lesquels évolue le robot. Parmi les environnements existants on cite : l'environnement avec connaissance a priori, sans connaissance a priori ou encore l'environnement dynamique [Prus.,96]. Pour un robot soumis à des contraintes de non holonomie comme le robot mobile de type voiture, le planificateur de trajectoire doit trouver des chemins admissibles et faisables pour ces robots. A ces fins, plusieurs méthodes ont été développées, par exemple : la modification des chemins holonomes en chemins non holonomes [Sche.,98], l'utilisation de la méthode de guidage en utilisant les propriétés des systèmes nilpotents, chaînés et plats [Laum.,01]. D'autres méthodes de planification utilisent les chemins à base de droite et d'arc de cercle, comme les chemins de Dubins [Dubi.,57], et les chemins de Reeds and Schepp [Frai.,04]. Ces méthodes présentent un inconvénient pour le robot mobile, ce dernier doit manœuvrer pour se déplacer le long de ces trajectoires. Des travaux sur la planification utilisent certaines courbes analytiques qui ont la propriété d'avoir une courbure continue et variable. Par exemple, on cite : courbes à coordonnées polaires [Sche.,98], les clothoïdes [Sche2.,98], les spirales cubiques [Kana.,89].

La commande permet de faire réaliser à un robot mobile à roues des tâches précises. Parmi les tâches importantes on cite : la stabilisation de configurations fixes, le suivi de chemin et la poursuite de trajectoire [laum.,98], [Wals.,94]. Le caractère non linéaire et non holonome des robots mobiles à roues nécessite l'élaboration d'une commande adaptée, robuste face aux erreurs de modélisation et aux perturbations. La loi de commande doit garantir la stabilité du système au sens de Lyapunov [Slot.,98]. Les commandes linéaires sont devenues obsolètes pour la commande des systèmes non linéaires et sous actionnés. Pour cela, des commandes non linéaires ont été développées, comme la commande par bouclage dynamique linéarisant [laum.,98], [sams.,85], les commandes neuronales [Boud.,09]. D'autres lois de commande connues pour leur robustesse on été développées comme les commandes par mode glissant (commande à structure variable) et par Backstepping.

La technique des modes glissants a été développée en URSS par l'équipe de recherche d'Emilyanov [Perr.,02]. Cette technique est inspirée des travaux de Fillipov sur les équations différentielles à second membre discontinue. Elle est utilisée pour plusieurs types de tâches (exemple : la stabilisation [Hame.,05], la poursuite de trajectoire [Hame1.,07]) et elle est

applicable à plusieurs systèmes comme les robots à muscle artificiel [Boud.,09], les robots mobiles à roues [Defo.,07], [Bena.,03], [Hame1.,07], [Yang.,99].
La commande par Backstepping à été développée par Kokotovic [Koko.,95]. Cette technique présente la particularité de permettre d'élaborer plus facilement des commandes qui garantissent la stabilité du système au sens de Lyapunov. Le backstepping est appliqué à plusieurs sortes de systèmes, comme les robots mobiles à roues [Hong.,02], les robots marins [Ghom.,08], etc.

Le travail présenté dans ce mémoire concerne la planification et la commande en poursuite de trajectoire des robots mobiles à roues de type voiture en double braquage. La plateforme mobile étudiée est celle du Robucar. Ce travail est réalisé au Centre de Développement des technologies Avancées (CDTA), au sein de l'équipe « Navigation et contrôle des Robots Mobiles (NCRM) ».

Le Robucar est un robot mobile de type voiture en double braquage. Il possède quatre roues motrices dont chacune est actionnée par un moteur électrique. Des capteurs proprioceptifs lui permettent de récolter les informations sur sa vitesse de translation et l'angle de braquage des roues avant et arrière. Ce robot est sous actionné, ses entrées de commande sont : la vitesse de translation et l'angle de braquage instantané.

Les problématiques auxquelles on se confronte dans ce mémoire sont :

- La non holonomie du Robucar qui l'empêche d'emprunter n'importe quel chemin. Certains chemins seulement sont réalisables.

- L'élaboration de la loi de commande dépend de la complexité du modèle cinématique du Robucar et de son modèle d'erreur de poursuite. Ce dernier, n'est pas une représentation d'état du système, il en résulte que la synthèse d'une commande par retour d'état est plus difficile. La commande doit permettre au robot de suivre la trajectoire désirée tout en assurant la stabilité du système à commander.

Pour faire face aux problématiques décrites ci-dessus, nous proposons des approches concernant la planification de trajectoire et la commande.

- On propose de générer une trajectoire grâce à des courbes analytiques ayant la propriété d'avoir une courbure variable. La courbe choisie est la spirale de cornu. La

génération de la trajectoire est réalisée en prenant en considération les contraintes de non holonomie propre au Robucar.

- On propose une mise en forme du modèle d'erreur de poursuite en effectuant une transformation par difféomorphisme de ce dernier. Cela nous permet d'exprimer le système à commander dans l'espace d'état, ce qui facilitera par la suite la synthèse de la loi de commande.

- On propose d'élaborer des lois de commande en poursuite de trajectoire en utilisant les techniques de commande du backstepping et des modes glissants.

Ce mémoire se compose de quatre chapitres organisés comme suit :

- Dans le chapitre 1, on introduira quelques notions et caractéristiques importantes liée aux robots mobiles à roues et plus précisément au robot de type voiture en double braquage.

- Dans le chapitre 2, on présentera l'approche de la génération de trajectoire utilisée. La trajectoire est ensuite présentée sous forme de résultat de simulation

- Dans le chapitre 3, on présentera en premier les principes de base des commandes robustes choisies pour exécuter la poursuite de trajectoire. Par la suite, on proposera une nouvelle expression du modèle de poursuite de l'erreur. Puis, on élaborera des commandes stabilisantes à base de backstepping et le mode glissant qui garantissent la stabilité du système au sens de Lyapunov.

- Dans le chapitre 4, on présentera les résultats de simulation de la commande en poursuite de trajectoire (développé dans le chapitre 3) le long de la trajectoire générée. Des tests de robustesse de la commande seront réalisés pour différents types de perturbations. Enfin, les algorithmes de commande seront implémentés sur la plateforme expérimentale Robucar, et les résultats d'expérimentation sont présentés.

Chapitre 1:
Etat de l'Art : La Robotique Mobile

1.1 Introduction

Les robots mobiles à roues ont fait l'objet de beaucoup de recherches dans différents domaines de la robotique. Ces robots ont un champ d'application très large, que ce soit pour des applications civiles comme les transports urbains automatisés, scientifique dans le cas d'exploration de milieux naturels hostiles ou encore militaire.

L'étude de ces robots se fait suivant plusieurs domaines. On cite en premier, la modélisation des robots en donnant une représentation mathématique du robot mobile. En second, la planification du mouvement et de la trajectoire pour le robot à commander. Et enfin, la commande des robots mobiles. Les robots mobiles à roues existants sont des systèmes complexes à cause de leur non-linéarité et non holonomie. En plus, ce sont des systèmes sous actionnés.

Dans ce chapitre, on présente certaines notions importantes liées à la robotique mobile et à ses applications. On introduit en premier ce que signifie la non holonomie. On présente juste après les caractéristiques liées aux différents types de robots existants dans la littérature ainsi que les différentes formes mathématiques existantes et pouvant les représenter. Par la suite, on expose certaines méthodes de planifications de trajectoire. Et enfin, on présente les différentes tâches de commandes pouvant être appliquées sur les robots mobiles à roues.

1.2 Les systèmes non holonomes

Un système non holonome est un système, dont les équations exprimant des relations vitesse/position des différentes variables du système ne sont pas intégrables. Les vitesses généralisées qui satisfont une égalité ne peuvent être représentées sous une forme de contrainte sur des positions généralisées. On ne peut donc pas relier toutes les configurations du système entre elles grâce à des relations algébriques. La non holonomie peut engendrer une réduction de l'espace des configurations du système mécanique. Il existe plusieurs classes des systèmes non holonomes, parmi eux les robots mobiles à roues.

1.3 Les robots mobiles à roues :

Les robots mobiles à roues représentent une classe de robots ayant une plateforme mobile contrairement aux autres types de robots (Marcheur, sous-marin, aérien…). Ces robots

présentent plusieurs contraintes, du fait qu'ils soirnt des systèmes sous actionnés, complexes et non holonomes. Ces contraintes rendent les robots difficiles à manipuler dans le cas où l'on veut élaborer un modèle mathématique représentatif adéquat, ou encore de les commander ou de leur planifier une trajectoire.

Dans le but de rendre le robot mobile plus simple à étudier, des hypothèses sont prises en considération comme celles des « Roulement sans Glissement ».

- **Hypothèse de roulement sans glissement**

L'hypothèse de roulement sans glissement permet d'élaborer plus facilement le modèle cinématique du robot mobile à roues. La roue en contact avec le sol est supposée rouler sans glissement. Ce phénomène se traduit par une vitesse nulle au point de contact I (voir figure (1.1)) entre la roue et le sol.

A titre d'exemple, la contrainte de roulement sans glissement pour les robots unicycle et tricycle est :

$$\dot{x}\, cos(\theta) - \dot{y}\, sin(\theta) = 0 \qquad (1.1)$$

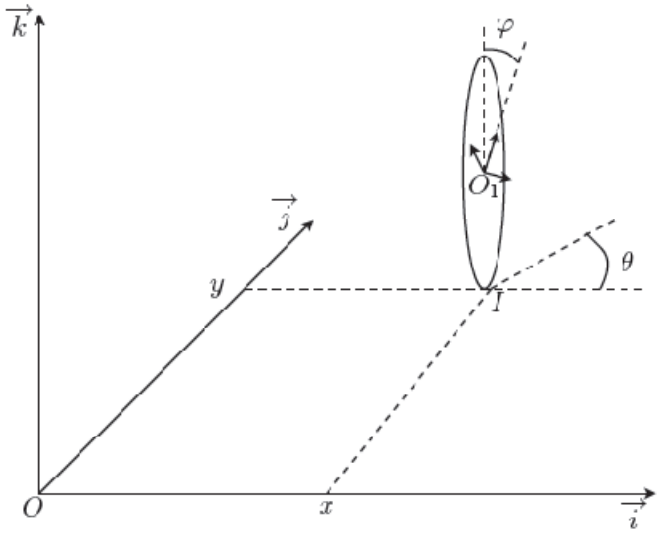

Figure 1.1: Roulement sans glissement

- **Les systèmes mécaniques sous-actionnés**

C'est un système dont la dimension de son espace de configuration est supérieure au nombre de ses entrées de commandes. Les contraintes non intégrables peuvent apparaître dans les systèmes sous-actionnés. Par exemple, des contraintes sur l'accélération qui ne peuvent

être intégrées pour avoir des conditions équivalentes en vitesse. On appelle ces contraintes des contraintes non holonomes du second ordre.

Dans ce qui suit on présente quelques robots mobiles à roues ainsi que leur modèle cinématique respectif.

1.3.1 Les robots mobiles de type unicycle

Ces robots sont aussi appelés robots mobiles différentiels. Ils sont constitués de deux roues indépendantes. L'accélération de chaque roue est commandée séparément par un moteur pour permettre au robot d'atteindre différentes configurations de son environnement (voir figure (1.2)). La stabilité de la plateforme est assurée par des roues folles.

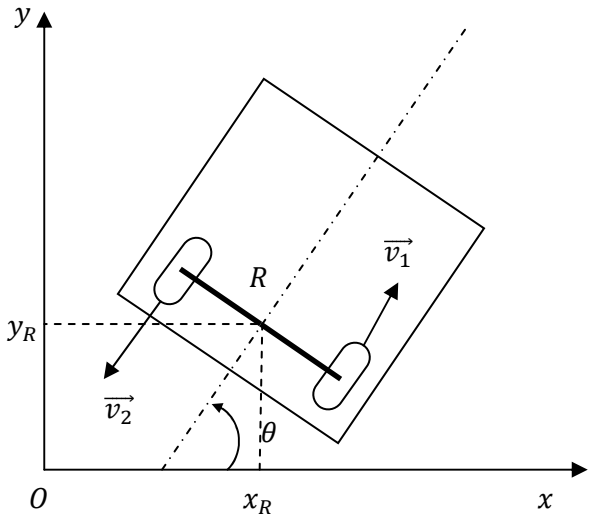

Figure 1.2 : Robot mobile de type unicycle

Le modèle cinématique est donné par le modèle mathématique suivant :

$$\begin{cases} \dot{x} = v_1 \, cos(\theta) \\ \dot{y} = v_1 \, sin(\theta) \\ \dot{\theta} = v_2 \end{cases} \tag{1.2}$$

Où (x, y) représentent les coordonnées cartésiennes du robot dans le plan, θ représente l'angle d'orientation du robot mobile par rapport à l'axe des abscisses. Les vitesses v_1 et v_2 représentent la vitesse appliquée à chaque roue.

1.3.2 Les robots mobiles de type voiture

1.3.2.1 Les robots de type voiture en simple braquage

C'est un robot constitué de quatre roues (voir figure (1.3)). Les roues de l'essieu arrière sont fixes. Les roues de l'essieu avant ont la capacité de braquer jusqu'à un angle

maximal de φ_{max}. Ce robot reçoit deux entrées de commande qui sont la vitesse de translation v et la vitesse de braquage u.

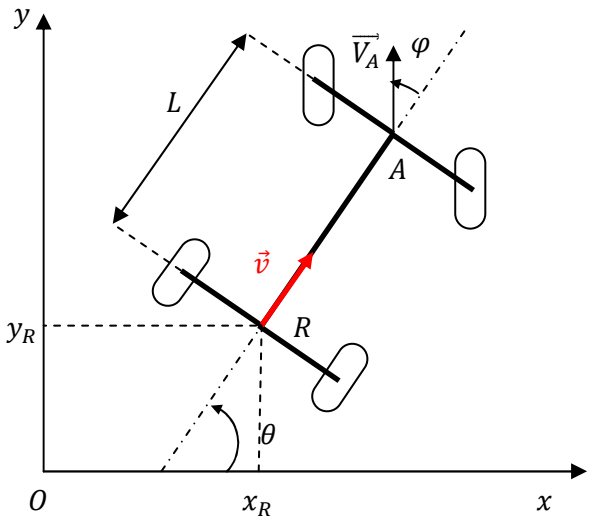

Figure 1.3 : Voiture en simple braquage

Le modèle cinématique du robot de type voiture en simple braquage est donné par:

$$\begin{cases} \dot{x} = v\,cos(\theta) \\ \dot{y} = v\,sin(\theta) \\ \dot{\theta} = \dfrac{v}{L}\tan(\varphi) \\ \dot{\varphi} = u \end{cases} \tag{1.3}$$

Les variables (x, y) représentent les coordonnées cartésiennes du robot dans le plan. θ représente l'angle d'orientation du robot mobile par rapport à l'axe des abscisses. φ représente l'angle de braquage des roues avant par rapport à l'axe du robot (figure (1.3)). Les vitesses v et u représentent respectivement la vitesse de translation du robot et la vitesse de rotation des roues avant.

1.3.2.2 Les robots de type voiture en double braquage

Le robot mobile étudié dans ce mémoire est un robot de type voiture en double braquage connu aussi sous le nom de « Bis-Car » (Bi-Steerable Car). C'est un système non holonome. Il est limité dans son mouvement par des contraintes cinématiques qui interviennent dans : le braquage maximum, une vitesse de braquage maximale et une vitesse de translation limitée.

Le système à commander compte une autre complexité liée au fait qu'il soit sous-actionnée. Le nombre d'entrées de commande est inférieur au nombre de sorties car le système comporte deux entrées de commande et quatre sorties représentant l'état du système.

Le robot mobile en double braquage étudié dans ce mémoire a la capacité de braquer les essieux avant et arrière en même temps. Ce qui apporte d'une part une plus grande manœuvrabilité du robot qui lui permet de passer par des virages étroits, une plus grande stabilité dans ses déplacements. D'autre part, le système devient plus complexe à représenter du point de vu mathématique.

- **Le modèle cinématique**

Le robot mobile de type voiture en double braquage (Cycab) a la caractéristique d'avoir un angle de braquage des roues arrière proportionnel à l'angle de braquage avant avec un facteur $-k$. Si les roues avant ont un angle de braquage de $\varphi_{av} = \varphi$, alors les roues arrière doivent avoir un angle de $\varphi_{ar} = -k\varphi$ par rapport à l'axe qui passe par les points R et F (voir figure 1.4). Les points R et F représentent respectivement le milieu de l'essieu arrière et le milieu de l'essieu avant du robot.

Le robot mobile peut être représenté par le point R [Sekh.,00] ou le point F .Le modèle cinématique du Cycab est différent si on l'exprime par rapport à l'un des deux points R et F.

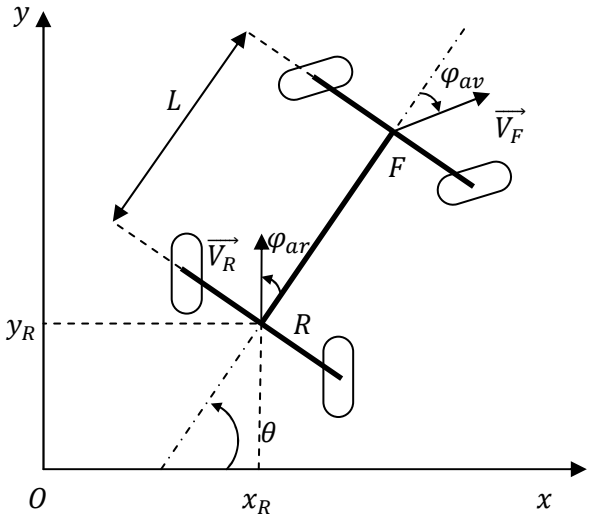

Figure 1.4: Modèle du Cycab

Le modèle cinématique exprimé par rapport au point R est donné par:

$$\begin{cases} \dot{x} = u\, cos\big(\theta + f(\varphi)\big) \\ \dot{y} = u\, sin\big(\theta + f(\varphi)\big) \\ \dot{\theta} = \dfrac{u}{L}\dfrac{\sin\big(\varphi - f(\varphi)\big)}{\cos(\varphi)} \end{cases} \tag{1.4}$$

Le modèle cinématique exprimé par rapport au point F est donné par:

$$\begin{cases} \dot{x} = u\, cos(\theta + \varphi) \\ \dot{y} = u\, sin(\theta + \varphi) \\ \dot{\theta} = \dfrac{u}{L}\dfrac{\sin\big(\varphi - f(\varphi)\big)}{\cos\big(f(\varphi)\big)} \end{cases} \tag{1.5}$$

Avec : u: vitesse de traction, (x, y) : les coordonnées cartésiennes

et : $f(\varphi) = -k\varphi$

Dans le but de diminuer la complexité du modèle cinématique du Cycab, on suppose que les angles de braquage des roues avant et arrière sont égaux et symétriques. Par conséquent, le facteur k est choisit comme suit : $k = 1$.

Le modèle cinématique est simplifié. Si l'on réécrit le modèle présenté dans l'équation (1.4) on obtient :

$$\begin{cases} \dot{x} = u\, cos(\theta - \varphi) \\ \dot{y} = u\, sin(\theta - \varphi) \\ \dot{\theta} = \dfrac{u}{L} 2\sin(\varphi) \end{cases} \tag{1.6}$$

L'utilisation du modèle cinématique (équation (1.6)) pour les tâches de commande est difficile. Cela est du à sa non linéarité. La mise en forme des modèles cinématiques en une forme canonique facilite leur utilisation.

1.3.3 La mise en forme

La transformation du modèle cinématique en une forme canonique facilite d'une part l'élaboration de la loi de commande [Mori.,00], [Mori2.,00], d'autre par elle facilite la planification de trajectoire [sams.,85], [laum.,98]. Les formes canoniques que l'on rencontre dans la littérature sont : la forme canonique de commandabilité généralisée, la forme chaînée [sams.,85], la forme de puissance. La forme canonique que l'on rencontre le plus souvent dans la littérature est la forme chaînée.

- **La forme chaînée**

La transformation d'un système non linéaire et non holonome en une forme chaînée permet d'avoir un système simplifié et linéarisé. Le changement du modèle se fait grâce à une transformation par difféomorphisme. En effet, si le système non holonome répond à certaines conditions [sams.,85], un changement de coordonnées adéquat dans l'espace d'état permet de le transformer en un système chaîné (voir le modèle ci-dessous).

$$\begin{cases} \dot{z}_1 = v_1 \\ \dot{z}_2 = v_2 \\ \dot{z}_3 = v_1 z_2 \\ \vdots \\ \dot{z}_n = v_1 z_{n-2} \end{cases} \quad (1.7)$$

Les variables z_i représentent les nouvelles variables d'états du système.
Les variables $v_1 \; et \; v_2$ représentent les nouvelles entrées de commandes.

1.4 La planification de trajectoire

La planification de trajectoire est utilisée dans la robotique mobile pour permettre au robot de se déplacer d'un point initial à un point final en suivant un chemin planifié. Le planificateur prend en considération plusieurs types de contraintes, qu'elles soient liées au robot mobile ou à l'environnement de ce dernier. Plusieurs méthodes de planification ont été élaborées pour s'adapter aux différents environnements dans lesquels évolue le robot, comme l'environnement avec connaissance a priori, sans connaissance a priori ou encore l'environnement dynamique.

Dans un environnement avec connaissance a priori, on suppose que les obstacles sont connus et statiques, parmi les méthodes utilisées pour la planification on cite : le Roadmap, les potentiels, la décomposition cellulaire [Prus.,96], [Lato.,91]. Le travail effectué grâce à l'une de ces trois méthodes se résume en trois tâches : la modélisation de l'environnement, la recherche de tous les chemins possibles que peut emprunter le robot mobile et enfin le choix du chemin optimal.

Dans un environnement sans connaissance a priori, on ne connait que la position du robot mobile ainsi que la position de la cible, on ne connait pas la position des obstacles mais on suppose qu'ils sont statiques.

Dans un environnement dynamique les obstacles sont mobiles, le robot doit détecter les obstacles à proximité à l'aide de capteurs pour qu'il n'y ait pas de collision.
Le planificateur de trajectoire doit non seulement répondre à des exigences liées à l'environnement mais aussi à des contraintes liées à la cinématique du robot. A cause de la non-holonomie des robots mobiles à roues comme les véhicules de type voiture, les trajectoires planifiées ou générées doivent être réalisables par le robot.
La planification de trajectoire peut être classifiée en deux grandes familles : la planification globale et la planification locale.

1.4.1 La planification globale

Le planificateur global est utilisé dans les environnements avec connaissance a priori. En général, c'est un algorithme itératif destiné à planifier le chemin du robot mobile dans un environnement connu et statique et qui permet au robot d'atteindre la position finale à partir de sa position initiale. la première étape consiste dans la modélisation de l'environnement puis la seconde étape consiste dans la planification du chemin optimal qui évite tout type de collision avec les obstacles statiques, exemple de planificateur : Dijksta [Prus.,96], A*,D*, Fil d'ariane [Sche.,98].

1.4.2 La planification locale

Le planificateur local est utilisé pour permettre au robot de générer une trajectoire locale, afin d'atteindre des sous-buts. On ne peut pas utiliser ce genre de planificateur pour planifier une trajectoire globale. Plusieurs applications sont possibles dans la planification locale comme par exemple :

- Des planificateurs pour l'évitement d'obstacle. On utilise les méthodes probabilistes pour la planification. Les informations issues des capteurs extéroceptifs sont nécessaires pour générer une trajectoire qui va permettre au robot d'éviter une collision un obstacle statique ou dynamique. Ces planificateurs sont généralement utilisés dans les environnements sans connaissance a priori et les environnements dynamiques.

- Des planificateurs pour la génération de chemins réalisables par les robots mobiles [Lami.,01]. Ces planificateurs prennent en considération certaines contraintes de non holonomie et cinématiques propres au robot mobile. Il existe plusieurs manières pour

générer une trajectoire. La génération dépend des contraintes propres du robot mais aussi des contraintes qui lui sont imposées, comme par exemple, la trajectoire à réaliser par le robot mobile peut obliger ce dernier à effectuer des manœuvres. Dans la littérature on peut trouver plusieurs méthodes comme par exemple : la déformation des chemins holonomes pour les rendre admissibles pour les robots non-holonomes [Sche.,98]. L'utilisation des chemins de Dubins [Dubi.,57] et des chemins de Reeds and Shepp [Frai.,04]. La génération des chemins à courbure continue ou encore l'utilisation de la méthode de guidage [Laum.,01].

1.4.2.1 La méthode de guidage

Cette méthode permet de résoudre le problème de planification de chemin dans des environnements sans obstacles. La mise en œuvre de cette méthode ressemble à la commande en boucle ouverte. On applique une entrée au système dont on veut planifier le chemin, et on récolte en sortie les positions du robot à chaque instant t. Cette méthode est applicable à certains types de systèmes dont la représentation mathématique répond à certains critères. Dans [laum.,98], la méthode de guidage est appliquée aux systèmes nilpotents, chainés et plats.

1.4.2.2 La méthode des chemins à courbure continue

Cette méthode est une extension des chemins de Reeds and Shepp et des chemins de Dubins. Ces derniers permettent de générer un chemin composé d'une combinaison de droites et de cercles. Ces chemins comportent une discontinuité dans la courbure lors du passage d'une droite à un cercle et vis versa. Cette discontinuité se répercute sur le mouvement réel du robot qui emprunte le chemin généré. Le robot doit s'arrêter ou manœuvrer ou encore il ne s'arrête pas et s'écarte du chemin planifié en passant d'une droite à un cercle par exemple. L'inconvénient est donc lié aux contraintes cinématiques du robot. Les chemins à courbure continue permettent de joindre deux configurations de la trajectoire qui ont une courbure différente. Ces chemins exploitent les caractéristiques de courbes mathématiques dont la courbure est variable sans présenter de discontinuité. Parmi ces courbes on cite : les spirales cubiques [Kana.,89] et les spirales de Cornu [Sche.,98], [Ran.,09].

1.5 Les problèmes de commandes des robots mobiles :

Les robots mobiles de type voiture sont des systèmes fortement non linéaires. Les commandes non linéaires sont recommandées pour le contrôle de ces systèmes. Plusieurs types de tâches peuvent être réalisées grâce aux commandes élaborées.

1.5.1 Les tâches à réaliser

Afin de synthétiser la commande la plus adaptée dans chaque cas, on classifie les tâches à exécuter par le robot en trois catégories [laum.,98]:

- **Stabilisation de configurations fixes**

Le robot mobile est représenté dans un repère de référence $\mathcal{R}_0$. Le repère lié au robot à commander est $\mathcal{R}_1$. L'objectif de la commande est de réguler à zéro la configuration $\zeta\big(x(t), y(t), \theta(t)\big)$ du point P qui représente le robot, ce qui lui permet d'atteindre le point désiré (voir figure 1.5). Les repères $\mathcal{R}_1$ et $\mathcal{R}_0$ seront alors confondus. Dans ce problème, les contraintes de non holonomie rendent la commande très difficile. Les méthodes classiques de la commande linéaire sont insuffisantes. La principale application de cette méthode est la manœuvre du parcking.

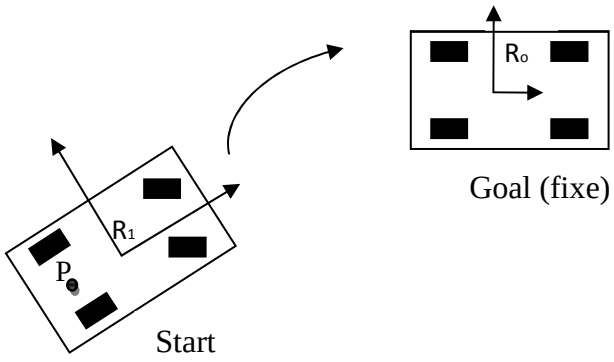

Figure 1.5: Stabilisation de configurations fixes

- **Suivi de chemin**

Soit une courbe $\mathcal{C}$ du plan. Le robot à commander est représenté par un point P dont la configuration est $\zeta\big(x(t), y(t), \theta(t)\big)$ par rapport à un repère de référence $\mathcal{R}_0$. La commande doit permettre au robot de parcourir la courbe $\mathcal{C}$ avec une vitesse v_0. Les variables que l'on doit réguler sont les variations de la configuration ζ du point P du robot avec les

configurations de la courbe $\mathcal{C}$. La figure (1.6) ci-dessous montre une représentation schématique du suivi de chemin.

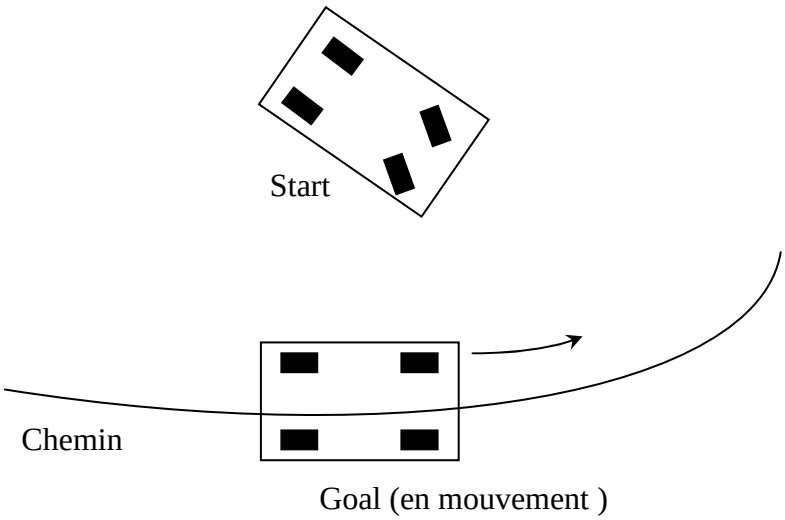

Figure 1.6: Suivi de chemin

- **La poursuite de trajectoire**

Dans ce problème, la vitesse de translation du robot n'est pas constante comme on l'a vu dans le suivi de chemin. Le robot doit atteindre chaque configuration du chemin en un temps t précis. Pour cela il faut doter la courbe $\mathcal{C}$ d'une loi horaire en la paramétrant par une variable temporelle t. Cela permet d'obtenir une trajectoire caractérisée par des coordonnées $\big(x_r(t), y_r(t)\big)$ dans un repère de référence $\mathcal{R}_0$. La commande va permettre de réguler le vecteur d'erreur $\big(x(t) - x_r(t), y(t) - y_r(t), \theta(t) - \theta_r(t)\big)$ des configurations du point P par rapport à la trajectoire à chaque instant t. Le problème de la poursuite peut être interprété d'une autre manière, on supposant que la commande doit asservir un véhicule à suivre un robot de référence, dont la trajectoire est donnée par $\big(x_r(t), y_r(t)\big)$ (voir la figure (1.7)).

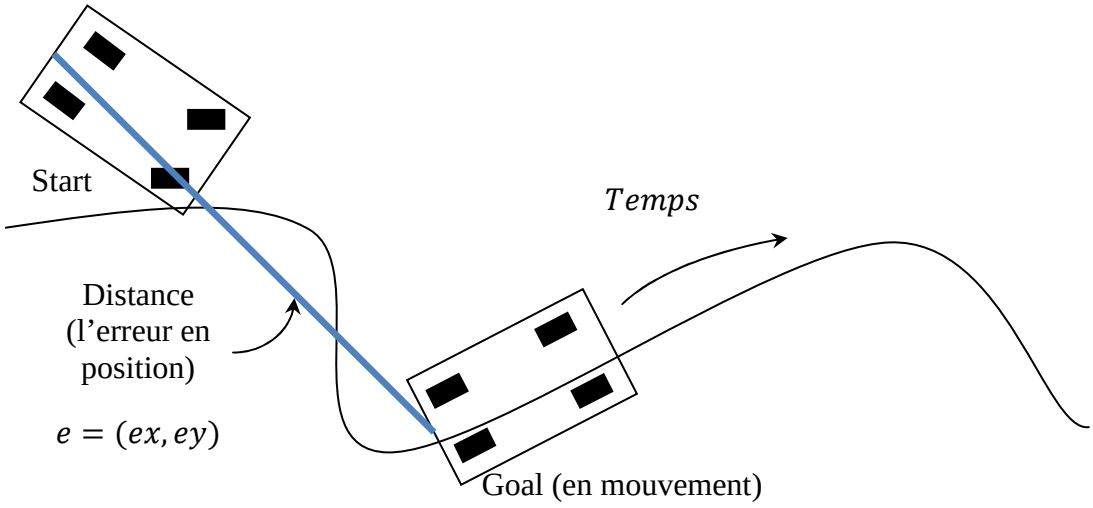

Figure 1.7: Poursuite de trajectoire

1.5.2 Les commandes utilisées

Plusieurs recherches ont été effectuées pour élaborer des stratégies de commande dans le but de contrôler les robots mobiles à roues. Le choix de la commande à appliquer dépend de la représentation mathématique du robot mobile. On présente ci-dessous quelques travaux concernant la commande des robots mobiles de type voiture pour la poursuite de trajectoire.

a. Dans [laum.,98], on utilise un modèle représentatif d'un robot mobile de type voiture en simple braquage. Ce modèle exprime l'erreur de poursuite du robot par rapport à la trajectoire désirée. Le modèle de poursuite est transformé via un difféomorphisme en un modèle équivalent au premier. La nouvelle expression mathématique du système permet d'exprimer ce dernier dans l'espace d'état. La commande synthétisée est une commande continue par retour d'état qui permet au système d'être stable au sens de Lyapunov.

b. Dans [Hame1.,07], la stratégie de commande développée est basée sur les commandes à structures variables. Cette commande discontinue permet de garantir la stabilité du système au sens de Lyapunov. La commande est plus robuste face aux perturbations et aux incertitudes de modélisation.

c. D'autres commandes à structures variables ont été élaborées pour la commande des robots mobiles à roues [Yang.,92] (comme l'unicycle ou le Car-Like (robot de type voiture en simple braquage)). La propriété de platitude du Car-Like [Fliess.,95] joue un rôle important dans la synthèse de la loi de commande. Dans [Bena.,03], ils utilisent un bouclage dynamique linéarisant pour commander le robot. L'élaboration de cette commande est possible grâce aux sorties plates du Car-Like.

d. D'autres stratégies de commandes basées sur le bouclage dynamique linéarisant sont développées dans [laum.,98], [D'And.,92], [Yang.,04], [Orli.,92]. Le modèle représentatif du robot est d'abord transformé en une forme chaînée. Cette forme canonique permet l'élaboration de la commande.

e. Les robots mobiles de type voiture en double braquage (Cycab) ont fait l'objet d'une étude dans [Herm.,03]. Le but étant d'utiliser les propriétés de platitudes du Cycab afin d'élaborer une commande par bouclage dynamique linéarisant.

1.6 Conclusion

On a présenté dans ce chapitre les généralités sur les robots mobiles à roue concernant les différents domaines de modélisation, de planification de trajectoire et de commande. Les robots mobiles sont des systèmes non holonomes et sous actionnés. Leur modèle cinématique complexe est parfois transformé via un changement de coordonnées pour avoir au final une forme canonique plus facile à manipuler dans le cas de l'élaboration de loi de commande ou encore de la planification de trajectoire. Le planificateur de trajectoire doit planifier une trajectoire admissible par un système contraint par sa non holonomie. Le planificateur de trajectoire doit prendre en considération les caractéristiques du modèle mathématique du robot, c'est-à-dire, ces contraintes de non holonomie ainsi que les contraintes liées à l'environnement extérieur. La commande destinée à contrôler le robot mobile dépend de la tâche à réaliser et de l'expression mathématique du robot à commander.

Chapitre 2 :
Planification de Trajectoire

2.1 Introduction :

Les méthodes de planification de trajectoire dans la robotique mobile sont nombreuses. Elles diffèrent les unes des autres par l'application réalisée par le robot. Les tâches d'un planificateur peuvent être une cible à atteindre, un obstacle à éviter, une trajectoire à optimiser du point de vu de la distance par rapport à un but, etc….

Dans ce mémoire, l'application principale qu'exécute le robot est la poursuite de trajectoire, par conséquent la trajectoire à développer doit répondre au critère nécessaire pour réaliser la tâche désirée. Dans ce chapitre, on génère une trajectoire faisable par le robot mobile de type voiture. On utilise pour cela des courbes mathématiques connues sous le nom de spirale de Cornu.

2.2 Etat de l'art :

Plusieurs méthodes de génération de trajectoire ont été développées pour prendre en considération les contraintes de non holonomie des robots afin que la trajectoire soit admissible pour ces derniers.

- La méthode la plus simple est de considérer un robot de référence caractérisé par un modèle cinématique identique à celui du système à commander. En injectant une commande à ce robot, il va se mouvoir dans l'espace en décrivant une trajectoire admissible pour le robot désiré. Les informations nécessaires à la poursuite sont récoltées facilement à chaque instant du début à la fin du chemin généré [Laum.,01].

- D'autres méthodes s'appuient sur la caractéristique de platitude [Bena.,03], [Hame2.,07] des systèmes comme pour les robots mobiles de type unicycle et tricycle. Cette méthode permet d'avoir une trajectoire de référence dans le plan décrite par des coordonnées (x, y). Cette trajectoire est choisie de manière à être faisable par le système à commander. A partir des coordonnées (x, y) et grâce à la caractéristique de platitude, on peut extraire les informations nécessaires à la poursuite de trajectoire en calculant les variables $v(t), \theta(t), \varphi(t)$ à partir des coordonnées (x, y) pour chaque instant de la trajectoire générée.

- La déformation des chemins holonomes en chemins non holonomes [Sche.,98], [Laum.,01]. Par exemple lors de l'utilisation des méthodes de planifications comme le graphe de visibilité [Prus.,96] ou la méthode des potentiels, le chemin généré n'est pas admissible pour un robot non holonome car il ne prend en considération aucune des contraintes cinématiques liées au robot. Pour cela on dit que le chemin généré est un chemin holonome. Pour pouvoir l'adapter au robot mobile de type voiture par exemple, il faut déformer le chemin holonome en un chemin non holonome. Les inconvénients de cette méthode sont, d'une part, la difficulté voire l'impossibilité de récupérer les informations nécessaires à l'exécution de la tâche de poursuite de trajectoire, et d'autre part, le robot est contraint à s'arrêter ou à manœuvrer.

- Des méthodes géométriques ont été développées afin de générer des chemins admissibles qui respectent la non holonomie des robots mobiles [Frai.,90]. Ces méthodes utilisent des courbes spécifiques. Les travaux de Dubins [Dubi.,57] ainsi que les travaux de Reeds & Shepp [Frai.,04] furent les premiers, Ils s'appuient essentiellement sur la combinaison des arcs de cercle et des droites afin de générer un chemin réalisable par le robot. Les chemins de Dubins permettent au robot de rouler et de s'arrêter mais pas de manœuvrer contrairement aux chemins de Reeds & Shepp qui permettent au robot de rouler, s'arrêter et de manœuvrer. Dans ces deux méthodes on note que la courbure de la trajectoire générée est discontinue, cela est dû à la concaténation des arcs de cercles et segments de droites dont la courbure est totalement différente. Si l'on prend l'exemple d'une voiture qui se déplace sur un chemin de Dubins, le passage du robot d'un segment de droite à l'arc de cercle peu se faire avec deux possibilités différentes, dans la première possibilité, le robot braque instantanément jusqu'à un angle α ce qui est physiquement irréalisable et peut endommager les actionneurs du robot. Dans la seconde possibilité, le robot s'arrête, braque jusqu'à atteindre un angle α, puis il poursuit sa course le long de la trajectoire.

- D'autres techniques mathématiques ont été développées afin de permettre au robot de se mouvoir dans l'espace le long d'un chemin sans s'arrêter ou manœuvrer. Ces chemins générés sont connus sous le nom de chemins à courbures continues [Sche.,98] [Frai.,04].

Position du problème :

Le type de planificateur de trajectoire est lié aux contraintes propres à la tâche que doit exécuter le robot ainsi qu'au robot lui même. Le planificateur doit générer une trajectoire d'où l'on peut extraire à partir des configurations de la trajectoire les données suivantes : $x(t), y(t), \theta(t), \varphi(t)$ à chaque instant t. Ces données représentent les configurations du robot de type voiture à chaque point du chemin généré. Ces configurations jouent un rôle important dans la poursuite de la trajectoire. La trajectoire générée doit être faisable par le robot mobile, ce qui veut dire que le planificateur doit intégrer les contraintes de non holonomie propres au robot à commander pour que le chemin planifié soit réalisable par le robot et que ce dernier puisse se mouvoir le long du trajet sans s'arrêter ou manœuvrer.

Approche proposée :

Dans ce chapitre on exploite les courbes dont la courbure est variable et continue pour construire une trajectoire admissible par le robot mobile de type voiture. Le chemin construit résultera de la concaténation de plusieurs types de courbes qui sont les segments de droite et les arcs de cercle à qui l'on ajoute des courbes dont le paramètre de courbure k est continu et variable, ce qui permettra de relier deux courbes qui ont des courbure différente. Le chemin à courbure continu résultant, doit être capable de prendre en considération les contraintes de non holonomie liées au robot. La contrainte liée au mouvement : rouler sans manœuvrer et sans s'arrêter. Et enfin, de pouvoir extraire les informations nécessaires à la poursuite de trajectoire.

2.3 La planification des chemins à courbure continue :

Les chemins à courbure continue permettent de générer un chemin faisable par le robot mobile tout en respectant ses contraintes de non holonomie et cinématique, tout en évitant les discontinuités de courbure du chemin. Dans les méthodes citées précédemment le chemin généré était constitué de segments de droites et d'arc de cercle, il en résulte beaucoup de discontinuité dans la courbure du chemin et le non respect des contraintes cinématiques du robot mobile. Pour remédier à ce problème on ajoute des chemins qui ont la caractéristique d'avoir une courbure variable, pour relier deux segments entre eux ou un segment et un arc de cercle.

Il existe deux familles des chemins à courbure continue, l'une est caractérisée par les coordonnées sous forme polynomiales décrivant la courbe, il en résulte une courbure variable.

L'autre est caractérisé par un paramètre de courbure sous forme polynomiale et qui décrit un chemin à courbure continue.

2.3.1 La méthode des coordonnées polynomiales :

La méthode des coordonnées polynomiales est une méthode très efficace pour générer les chemins à courbure continue pour les robots mobiles. Il existe plusieurs types de chemin à coordonnées polynomiales suivant le type de coordonnées exprimant la courbe comme : Les B-Splines, les courbes dont les coordonnées cartésiennes sont polynomiales et d'autres dont les coordonnées polaires sont polynomiales.
Cette méthode présente certains inconvénients, la vérification de la contrainte de courbure demande des calculs couteux, sans compter que les travaux précédents furent appliqués à des robots mobiles sans borne sur le rayon de braquage.

2.3.2 La méthode des courbures polynomiales :

Dans cette méthode, le paramètre de courbure décrivant la courbure du chemin généré s'écrit sous la forme d'un polynôme. L'intérêt de cette méthode est de générer un chemin à courbure continue qui prend en considération les contraintes cinématique de non holonomie du robot mobile. Le principe est d'utiliser une courbe mathématique qui permet de générer la courbe avec un paramètre de courbure K variable que l'on peut modifier selon nos contraintes, donc le paramètre de courbure doit s'exprimer en fonction des contraintes du robot, par exemple le braquage maximum que peut exécuter le robot.

Les contraintes cinématiques de non holonomie sont introduites en posant deux conditions principales qui sont : un paramètre de courbure K est borné et une dérivée de la courbure σ est aussi bornée.

L'inconvénient de cette méthode est que lors de la projection de cette courbe sur le plan lR^2, les coordonnées ne sont pas calculables, elles peuvent seulement être approchées.
Il existe plusieurs types de courbes polynomiales : les clothoïdes, les spirales cubiques, les courbes intrinsèques.

2.3.2.1 Les spirales cubiques :

Ce sont des fonctions dont la direction de la courbe est une fonction cubique de l'abscisse curviligne s. La courbure de cette fonction permet d'éviter les discontinuités de

mouvement pour les robots mobiles. Ces spirales ont fait l'objet d'une étude par Kanayama [Kana.,89] pour planifier un chemin pour le robot «YAMABICO » en évitant les discontinuités de mouvement de ce dernier grâce à la continuité du paramètre de courbure K du chemin.

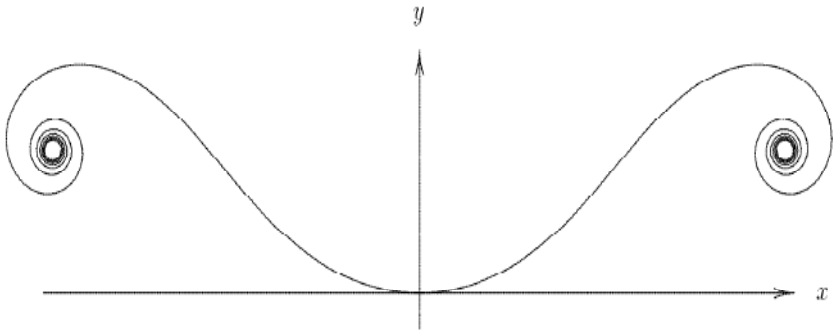

Figure 2.1: Spirale cubique

Cette courbe est caractérisée par sa direction et son paramètre de courbure qui s'expriment sous la forme suivante :

$$\begin{cases} K(s) = As^2 + Bs + C \\ \theta(s) = As^3 + Bs^2 + Cs + D \end{cases} \quad (2.1)$$

Avec : s : l'abscisse curviligne

K : Le paramètre de courbure

θ: L'angle entre la tangente à la courbe et l'axe des abscisses

L'inconvénient avec cette méthode c'est la difficulté de représenter le profil de braquage ainsi que d'introduire les contrainte de non holonomie du robot mobile de type voiture dans la génération de la courbe.

2.4 Les spirales de cornu :

La spirale de cornu est connue aussi sous le nom de clothoïde [Sche.,98]. C'est une courbe de dimension deux dont le paramètre de courbure varie linéairement avec l'abscisse curviligne (voir figure 2.2). Elle permet de raccorder deux courbes dont le paramètre de courbure est différent. Elle est utilisée pour le tracé des routes reliant des droites à des cercles. La spirale représente la courbe qu'effectue une voiture avec une vitesse de translation et une vitesse de braquage constantes. La clothoïde est caractérisée par un paramètre de courbure $K = \frac{1}{R}$.

Pour déterminer l'équation de la courbe, on suppose que le mobile qui se déplace tout au long de cette courbe de longueur L, présente une vitesse de translation v constante ainsi qu'une vitesse de braquage $\dot{\varphi}$ constante.

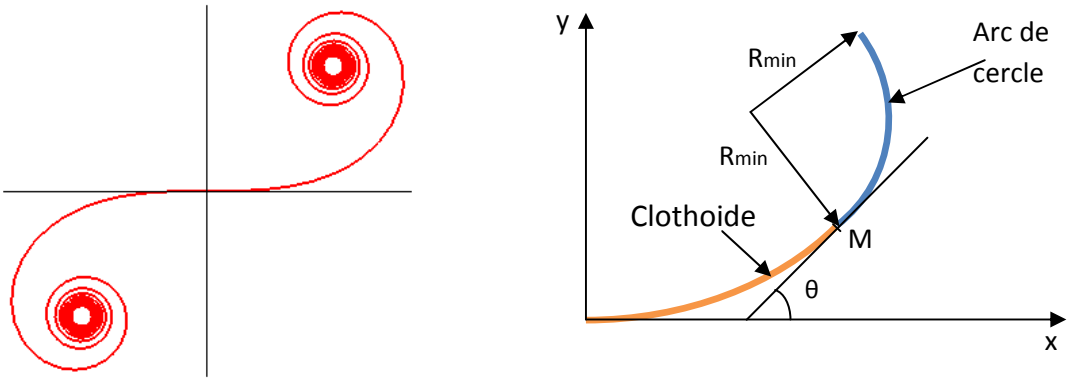

Figure 2.2 : les spirales de cornu

L'angle θ représente l'angle d'orientation de la tangente de la courbe au point M. Le paramètre de courbure K est exprimé par :

$$K = \sigma s \tag{2.2}$$

Où σ est La vitesse de courbure et s la longueur de l'abscisse curviligne. Cette relation montre que la courbure de la clothoide varie linéairement par rapport à l'abscisse curviligne.

Les coordonnées x et y du robot mobile changent en fonction de l'abscisse curviligne:

$$\begin{cases} x = \int_0^s cos\big(\theta(s)\big)\, ds \\ y = \int_0^s sin\big(\theta(s)\big)\, ds \end{cases} \tag{2.3}$$

En utilisant le développement limité de Taylor on peut obtenir les résultats suivants [Sche.,98]:

$$\begin{cases} x = \sum_0^\infty \dfrac{(-1)^n\ s^{4n+1}}{\sigma^{2n}\ 2^{2n}\ (4n+1)\ (2n)!} \\ y = \sum_0^\infty \dfrac{(-1)^n\ s^{4n+3}}{\sigma^{2n+1}\ 2^{2n+1}\ (4n+3)\ (2n+1)!} \end{cases} \tag{2.4}$$

Après la détermination des données (x,y,θ) exprimant la clothoîde, il faut exploiter cette courbe pour la génération du chemin à courbure continue. Ces derniers ne peuvent être uniquement composés de clothoide, il faut pour cela ajouter d'autres types de courbe afin de générer un chemin admissible pour le robot comme on détaillera ci-dessous.

La génération des chemins en utilisant des clothoîdes permet d'obtenir un chemin sans discontinuité de la courbure. Il en résulte que le robot mobile peut parcourir ce chemin en évitant les discontinuités du mouvement comme l'arrêt ou la manœuvre.
La clothoîde permet grâce à sa courbure variable et continue de relier deux courbes ayant l'un vis-à-vis de l'autre une courbure différente par rapport à l'axe des abscisses. Une seule clothoîde ne suffit pas pour éviter la discontinuité du mouvement du robot mobile, pour cela le chemin à courbure continu résultant sera composé par la concaténation de plusieurs courbes qui vont permettre le passage d'un point A à un point B et qui ont respectivement les orientations θ_A et θ_B par rapport à l'abscisse curviligne. On défini pour cela deux types de chemin qui peuvent nous permettre d'élaborer un chemin à courbure continue:

2.4.1 Les chemins optimaux

Les chemins à courbures continues admissible sont composés d'un ou plusieurs chemins élémentaires [Sche.,98]. Ces chemins peuvent décrire un ou plusieurs virages.

Définition [Sche.,98]*:*
Un chemin admissible de longueur L est appelé chemin élémentaire, si et seulement si, son profil de courbure vérifie la propriété suivante :

$$\exists \sigma \in \mathbb{R} / \begin{cases} \forall\, s \ \in [0, L\,/\,2]\,, K(s) = \sigma\, s \\ \forall\, s \ \in \ [L/2\,, L]\,, K(s) = \sigma\,(L - s) \end{cases} \tag{2.5}$$

Ce chemin est exprimé par le triplet (q_a, L, σ) qui représente respectivement : la configuration de départ $q_a = (x, y, \theta)$, la longueur du chemin L et la vitesse de courbure σ. Il est composé de deux clothoides identiques. Elles sont reliées l'une à l'autre au point P_M (voir figure 2.4). Ces deux clothoides donnent aux chemins optimaux la propriété de symétrie.
La courbure minimale $K_{\min}$ du chemin se trouve à ses extrémités $q_a \; et \; q_b$ comme le montre la figure 2.4 . La courbure maximale K_{max} est atteint au point P_M qui représente le milieu du chemin de longueur L. Pour une distance $s = \frac{L}{2}$, on obtient la courbure $K_{max} = \sigma L/2$.

Dans la figure (2.4), le chemin à courbure continue est composé d'une paire de clothoïde, qui relie deux configurations symétriques [Kana.,89], via la configuration P_M qui est le point de symétrie. La figure (2.3) représente la variation du paramètre de courbure K de manière linéaire.

Quand K varie de 0 à $\sigma L / 2$, la courbure est croissante. Elle représente la première clothoïde qui relie les configurations $q_a \; et \; q_m$.

Quand K varie de $\sigma L / 2$ à 0, la courbure est décroissante. Elle représente la seconde clothoïde qui relie les configurations $q_m \; et \; q_b$.

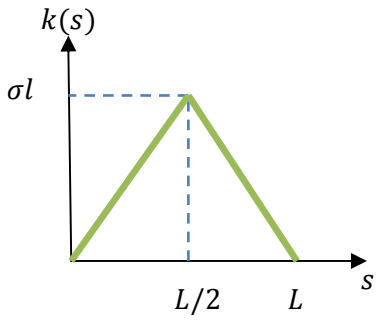

Figure 2.4 : Profil de la courbure K

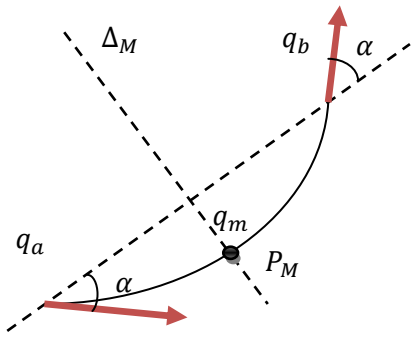

Figure 2.3 : Chemin élémentaire

Dans les chemins optimaux, la différence d'orientation maximale $\Delta\theta_{max}$ exprimant un virage reliant les configurations initiales q_a et final q_b est faible. Ce virage est dit « dégénéré » car il ne contient pas d'arc de cercle, il ne contient que des clothoides et permet de relier les segments de droites uniquement. La limite d'orientation $\Delta\theta_{\max}$ peut diminuer considérablement le nombre de chemins générés possibles et donc limité par le champ d'action du robot mobile. Il en résulte que la génération d'un chemin optimal ne permet pas au robot d'exploiter pleinement ses capacités de braquage, pour cela d'autres chemins sont pris en compte comme le montre le paragraphe ci-dessous.

2.4.2 Les chemins sous-optimaux

Définition *[Sche.,98]**:***

Un chemin admissible de longueur l est appelé « virage sous-optimal à courbure continue », si et seulement si, il est formé de deux portions de clothoïdes symétriques entourant un arcle de cercle [Sche2.,98].

La dérivé du profil de courbure d'un tel chemin vérifie par conséquent :

$$\exists\, \sigma_0 \in [-\sigma_{max}, \sigma_{max}] \Big/ \begin{cases} \forall\, s \in [0, l_0]\,, & \sigma(s) = \sigma_0 \\ \forall\, s \in [l_0, L - l_0]\,, & \sigma(s) = 0 \\ \forall\, s \in [L - l_0, L]\,, & \sigma(s) = -\sigma_0 \end{cases} \qquad (2.6)$$

Avec $l_0 = min(L/2\,, K_{max} / |\sigma_0|)$, $\sigma_0 \neq 0$

Lorsque $\sigma(s) = \sigma_0$, $\sigma(s) = -\sigma_0$ cela veut dire que la courbure K varie linéairement en fonction de l'abscisse curviligne s.

$\sigma(s) = 0$: La courbure K est constante, ce qui veut dire que le chemin généré est un arc de cercle.

L'association d'un arc de cercle permet de maintenir le paramètre de courbure K constant, il en résulte un angle de virage plus important. Cela permet d'exploiter le braquage maximum du robot mobile.

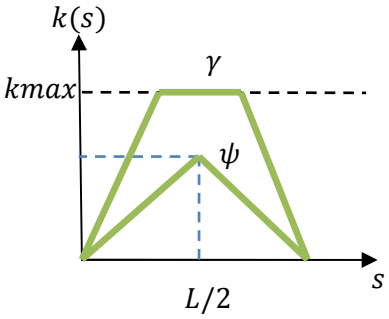

Figure 2.5 : Profil de la courbure K

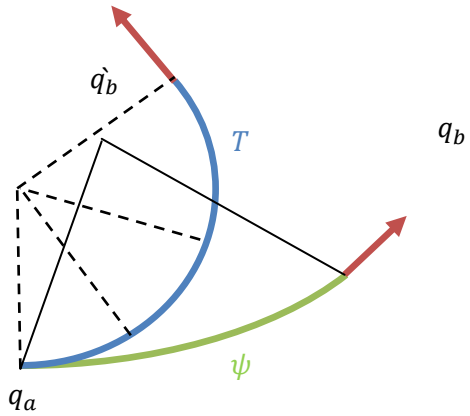

Figure 2.6 : Chemins sous-optimaux

La figure (2.6) représente deux courbes de type différent, la courbe ψ est un chemin élémentaire composé d'une paire de clothoïde, la seconde courbe T est un chemin sous optimal composé d'une paire de clothoïde et d'un arc de cercle. La figure (2.5) représente la variation de la courbure K du chemin T .Cette courbure est variable, lorsqu'elle est croissante, elle représente la variation de courbure de la première clothoïde, lorsqu'elle est constante, elle représente un arc de cercle et enfin si la courbure est décroissante, elle représente la seconde clothoïde.

Pour adapter les chemins à courbure continue à la non holonomie du robot mobile de type voiture en double braquage, on doit exprimer les contraintes cinématiques du robot à travers la courbure du chemin généré.

2.5 Contraintes cinématiques :

La génération du chemin à courbure continue est destinée à un robot mobile de type voiture. Ce dernier présente des contraintes de non holonomie l'empêchant d'emprunter n'importe quel chemin. Pour rendre le chemin généré admissible par le robot, il faut prendre en considération les contraintes de ce dernier.

Le robot de type voiture possède un angle de braquage limite φ_{max} , il ne peut pas prendre des virages dont le rayon de giration est inférieur à un rayon limite R_{min} . La Vitesse de braquage du robot est aussi bornée.

Ces contraintes sont prises en compte dans la génération des chemins en imposant des bornes pour la courbure du chemin et la vitesse de courbure.

Le modèle cinématique du robot mobile de type voiture en double braquage est donné par (voir paragraphe 1.3.2.2):

$$\begin{cases} \dot{x} = v\ cos(\theta - \varphi) \\ \dot{y} = v\ sin(\theta - \varphi) \\ \dot{\theta} = K \\ \dot{k} = \sigma \end{cases} \tag{2.7}$$

Avec:

$$\dot{\theta} = \frac{v}{l}\ 2\ sin(\varphi)$$

A partir des équations ci-dessus on déduit la relation entre l'angle de braquage et le paramètre de courbure K tel que [Fra.,04]:

$$K = \frac{v}{l}\ 2\ sin(\varphi) \tag{2.8}$$

L'angle de braquage ainsi que la vitesse de translation étant bornés à cause du braquage limite imposé par les contraintes non holonomes du robot mobile, le paramètre K devient lui aussi borné. Les propriétés inhérentes aux spirales de cornus imposent une vitesse de translation v constante.

Si les variations de l'angle de braquage sont très petites, on peut effectuer l'approximation suivante : $sin(\varphi) \approx \varphi$.

Il en résulte l'égalité suivante :

$$K \approx \frac{v}{l}\, 2\, \varphi \tag{2.9}$$

La vitesse angulaire de l'angle de braquage est elle aussi bornée, il en résulte que la dérivé du paramètre de courbure est majorée par une valeur max.

$$\sigma = \dot{K} = -2\, \frac{v}{l} \dot{\varphi}\, cos(\varphi) \quad avec \quad \sigma \leq \sigma_{max}$$

Si le braquage φ est très faible, on obtient $cos(\varphi) \approx 1$, il en résulte :

$$\sigma \approx -\, 2 \frac{v}{l} \dot{\varphi} = constante \tag{2.10}$$

L'intégration des contraintes de non holonomie est réalisée en imposant des bornes sur la courbure K et sa dérivée σ appartenant aux chemins à courbure continue générés comme on le montre ci-dessous.

- Contrainte sur la courbure

A partir de l'expression

$K \approx \frac{v}{l}\, 2\, \varphi$

on déduit les inégalités suivantes :

$$-\varphi_{max} < \varphi < \varphi_{max} \quad \Longrightarrow \quad K_{max} < K < K_{max}$$

Avec :

$K_{max} = \frac{v}{l}\, 2\, sin(\varphi_{max})$

- Contrainte sur la vitesse de courbure

A partir de l'expression

$\sigma \approx -\, 2 \frac{v}{l} \dot{\varphi}$

on déduit les inégalités suivantes :

$$-\dot{\varphi}_{max} < \dot{\varphi} < \dot{\varphi}_{min} \quad \Longrightarrow \quad \sigma_{max} < \sigma < \sigma_{max}$$

Avec :

$\sigma_{max} = \dot{K}_{max}$

2.6 La génération de trajectoire : Simulation

Dans les paragraphes précédents, on a développé l'idée de génération de chemin admissible pour le robot mobile de type voiture. Le passage de la notion de chemin à celle de trajectoire nécessite l'introduction du paramètre temporel. Ce dernier sera introduit dans l'algorithme de simulation que l'on verra ci-dessous.

Dans notre cas, l'intérêt de la génération de trajectoire est d'extraire les configurations $q(t) = \big(x(t), y(t), \theta(t), \varphi(t), v(t)\big)$ à chaque instant t le long du chemin à parcourir. Ces configurations représentent un robot virtuel qui parcourt la trajectoire désirée et qui par la suite sera suivi par un robot réel. Ces informations sont nécessaires pour réaliser la poursuite de trajectoire.

Dans ce paragraphe on présente l'algorithme de génération de trajectoire destiné à générer une trajectoire admissible pour le robot mobile de type voiture. On présente par la suite les résultats de simulation pour un circuit composé de plusieurs chemins sous-optimaux.

2.6.1 Algorithme de planification :

L'algorithme de planification permet à l'utilisateur de choisir

- le nombre de virage dans le chemin généré.
- L'angle du virage
- La configuration de départ du robot mobile :
  - en introduisant l'angle de départ, on détermine l'orientation de départ du robot mobile.
  - en introduisant la position initiale en coordonnées (x,y), on détermine la position (x,y) de départ du robot mobile.

L'algorithme permet de générer un chemin admissible en introduisant des bornes sur la courbure et la vitesse de courbure de la trajectoire générée.

Cet algorithme permet de visualiser la trajectoire générée sur le plan de dimension 2, les profils de l'orientation, de braquage ainsi que la vitesse correspondant à un robot virtuel se déplaçant le long de la trajectoire.

La configuration initiale de la trajectoire est :

$$x_0 = 0, \quad y_0 = 0, \quad \theta_0 = 0, \quad \varphi_0 = 0$$

On présente ci-dessous l'organigramme de génération de trajectoire :

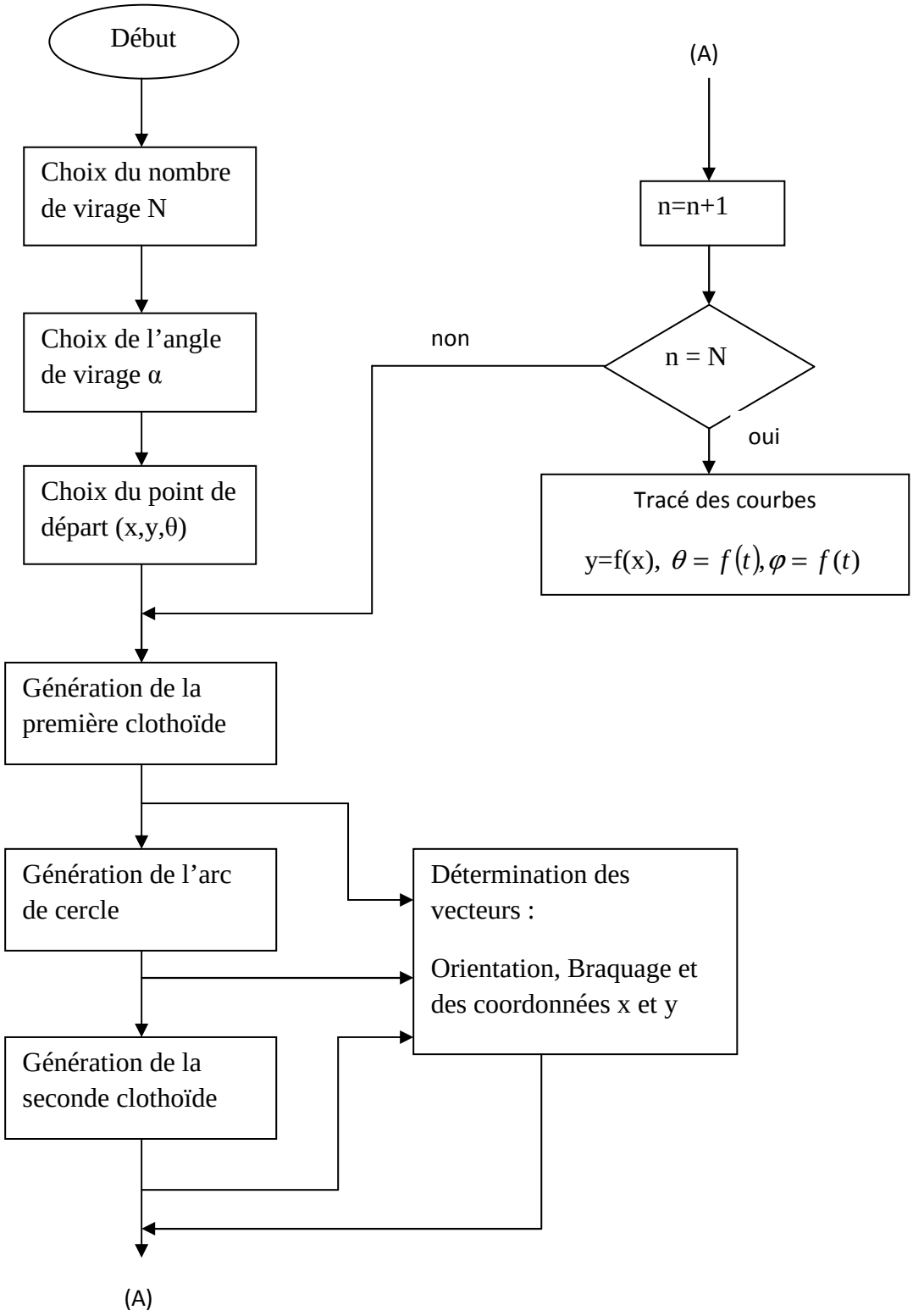

Figure 2.7 : Organigramme de l'algorithme de génération de la trajectoire

2.6.2 Les résultats de simulation :

- On présente ci-dessous dans la figure (2.8) la trajectoire généré grâce à l'algorithme de génération (voir paragraphe précédent). Les profils de braquage, d'orientation et de vitesse de traction sont présentés respectivement dans les figures (2.9) et (2.10).

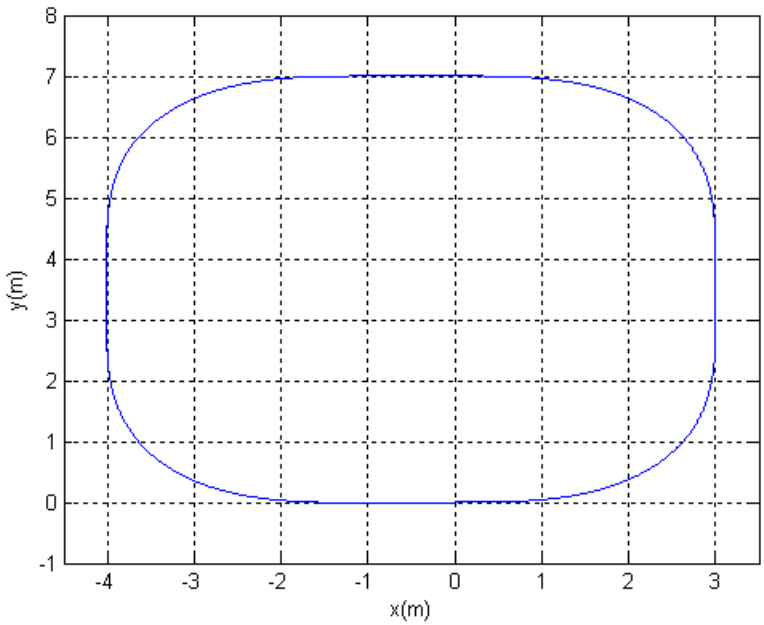

Figure 2.8 : Trajectoire générée

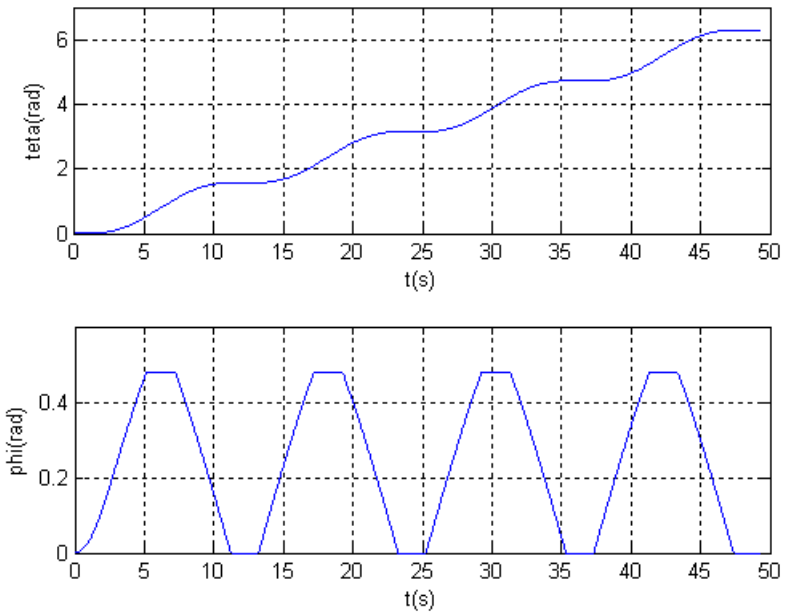

Figure 2.9 : Angles d'orientation $\boldsymbol{\theta}$ et de braquage $\boldsymbol{\varphi}$ total

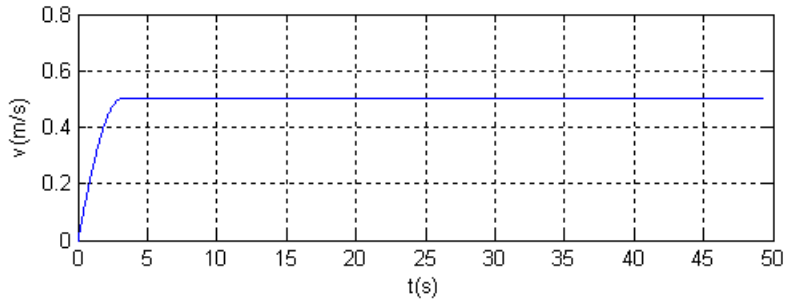

Figure 2.10 : Vitesse de traction

La trajectoire générée ci-dessus dans la figure (2.8) est composée de 4 virages, l'angle d'orientation de chacun d'eux est de $\pi / 2$. Chaque virage est composé d'un arc de cercle relié à une clothoïde par chacune de ses extrémités. Dans la figure (2.9), le profil de braquage exprime le braquage total des roues avant et arrière du robot virtuel, chaque virage est exprimé par un trapèze. L'orientation a une forme d'escalier dont chaque marche représente un virage.

- Dans les figures (2.11, 2.12 et 2.13) on inclut dans la trajectoire générée une petite déviation du chemin entre le deuxième et le troisième virage, cette portion de trajectoire est générée en utilisant le modèle cinématique du robot mobile de type voiture en double braquage. Ce dernier est commandé en boucle ouverte en injectant au modèle cinématique une vitesse de traction constante et un angle de braquage variable. Les résultats obtenus montrent la portion de trajectoire ajoutée. L'intérêt d'ajouter un changement dans le profil de trajectoire est d'apporter une perturbation au niveau de la trajectoire dans le cas d'une poursuite de trajectoire comme on le verra dans les chapitres suivants.

  Ce changement de trajectoire engendre un changement de profil de la trajectoire ainsi que de l'orientation et du braquage le long de la déviation.

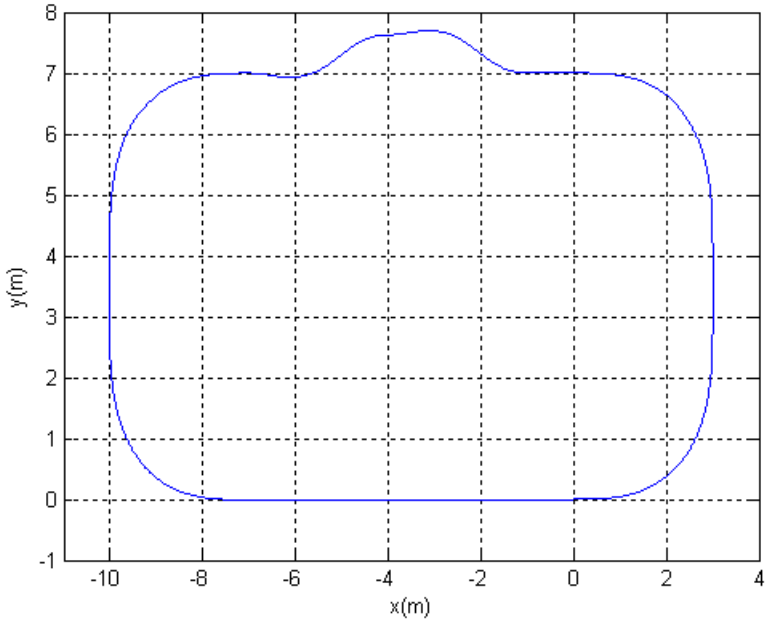

Figure 2.11 : Trajectoire générée

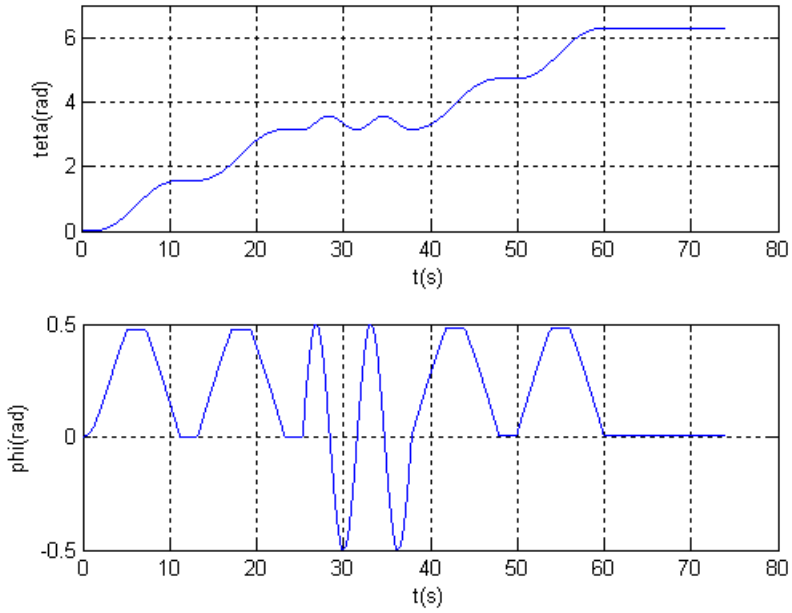

Figure 2.12 : Angles d'orientation $\boldsymbol{\theta}$ et de braquage $\boldsymbol{\varphi}$ total

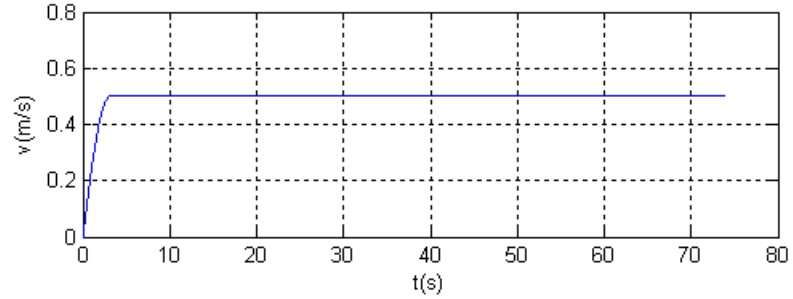

Figure 2.13 : Vitesse de traction

Dans la figure (2.12) le profil d'orientation de la trajectoire a une forme générale d'escalier, cependant, l'orientation prend la forme d'une sinusoïde au niveau du second escalier (entre le second et le troisième virage). Le profil de braquage prend une forme sinusoïdale après le second trapèze (Equivalent au deuxième virage).

2.7 Conclusion :

L'objectif de ce chapitre est de réaliser une génération de trajectoire admissible et faisable destinée à un robot mobile de type voiture en double braquage. Ce dernier étant soumis à des contraintes de non holonomie. Le chemin permettra au robot de se déplacer sans s'arrêter ou manœuvrer. L'approche présentée dans ce chapitre utilise les propriétés d'une courbe mathématique connue sous le non de spirale de cornu afin de générer des chemins à courbure continue. La trajectoire obtenue est constituée d'arcs de cercles, de segments de droites et de clothoides. L'intégration des contraintes de non holonomie a été effectuée en posant des bornes aux paramètres de courbure K ainsi qu'à la vitesse de courbure σ. Il en résulte une trajectoire sans discontinuité de courbure qui peut être parcourue par un robot mobile sans discontinuité de mouvement. L'algorithme de génération de trajectoire permet d'extraire les différents profils d'orientation, de braquage et de vitesse liés à la courbe à chaque instant t le long de la trajectoire générée.

Chapitre 3 :
Commande en Poursuite de Trajectoire

3.1 Introduction

La réalisation d'une poursuite de trajectoire nécessite la connaissance détaillée des configurations de la trajectoire à suivre à chaque instant t. Pour cela l'utilisation des informations obtenues grâce à la planification de trajectoire réalisée dans le chapitre précédent est incontournable. Ces informations représentent les configurations à chaque instant d'un robot virtuel. Ce dernier doit être suivi par le robot réel.

Dans ce chapitre, un modèle d'erreur représentatif de l'erreur de poursuite entre le robot mobile réel et le robot de référence est élaboré. Ce dernier permet une représentation du système dans l'espace d'état et l'estimation des nouvelles configurations du robot réel.

Nous développons des techniques de commandes en poursuite de trajectoire appliquées aux systèmes non holonomes, et plus précisément aux robots mobiles de type voiture en double braquage. On propose l'élaboration de lois de commandes par retour d'état, non linéaires et robuste face aux perturbations et aux erreurs de modélisation. Les commandes élaborées sont basées sur deux types principaux de commandes discontinues qui sont : la commande par Backstepping [Koko.,95] et la commande par Modes Glissants [Utki.,77]. Ces commandes devront garantir la stabilité du système au sens de Lyapunov.

Ces commandes doivent apporter stabilité et rapidité de convergence lors de la poursuite de trajectoire. On présente en premier une commande basée sur la méthode de Backstepping, en second on élabore des commandes basées sur les modes glissants d'ordre deux et on finit par le calcul d'une commande hybride utilisant les techniques du Backstepping et des modes glissants classiques. Le rôle de ces commandes est de stabiliser les variables d'états du modèle d'erreurs de poursuite autour de zéro le long d'une trajectoire admissible pour le robot de type voiture. La commande en poursuite de trajectoire est équivalente à une stabilisation des états du système (les états du modèle d'erreur) autour d'une valeur fixe.

3.2 La commande par Backstepping

L'approche de la commande par Backstepping a été développée par P.V. Kokotovic and al [Koko.,95]. Cette méthode est principalement destinée aux systèmes non linéaires avec des incertitudes paramétriques. L'algorithme de cette commande permet de calculer pas à pas

l'expression de la loi de commande en utilisant le principe de stabilité de Lyapunov (Fonction de Lyapunov) [Khal.,95] afin d'obtenir une commande qui garantit la stabilité du système au sens de Lyapunov et qui permet de réaliser la tâche de poursuite de trajectoire.

Dans cette méthode [Ghom.,08], on considère certaines variables d'états comme étant des commandes virtuelles qui sont des commandes intermédiaires pour la stabilisation de certains états du système. Ces commandes engendrent de nouvelles variables d'états qui sont considérées comme des commandes virtuelles jusqu'à apparition de la commande réelle de façon explicite. Le calcul récursif de la fonction de Lyapunov permet par la suite de calculer les véritables commandes appliquées au système.

Soit un système non linéaire et sous-actionné de la forme suivante :

$$\begin{cases} \dot{x}_1 = x_2 + f_1(x_1); \\ \dot{x}_2 = x_3 + f_2(x_1, x_2); \\ \vdots \\ \dot{x}_n = f_n(x_1, x_2, \dots, x_n) + u; \end{cases} \tag{3.1}$$

Avec :

x_i $(i = 1, \dots, n)$: Les variables d'états du système.

$f_i(x_1, x_2, \dots, x_i)$: Vecteurs.

u : Entrée de commande du système.

Dans la méthode de la commande par Backstepping, la variable d'état x_2 est une commande virtuelle servant à stabiliser x_1. L'expression de la variable x_2 est telle qu'elle stabilise la variable d'état x_1 au sens de Lyapunov.

La fonction candidate de Lyapunov égale à : $V_1 = \frac{1}{2} x_1^2$,

En posant l'expression de la variable x_2 comme suit : $x_2 = -x_1 - f_1(x_1)$.

Cela permet d'avoir la dérivée de la fonction de Lyapunov strictement négative :

$\dot{V}_1 < 0$, et donc, on obtient une stabilisation de x_1.

On pose l'expression suivante : $\alpha_1(x_1) = -x_1 - f_1(x_1)$,

α_1 qui représente l'expression désiré de x_2 qui permet de stabiliser la variable x_1.

Comme x_2 n'est pas une véritable commande, on introduit un changement de variable :

$$\begin{cases} z_1 = x_1 \\ z_2 = x_2 - \alpha_1(x_1) \end{cases} \tag{3.2}$$

avec $\alpha_1(x_1) = -x_1 - f_1(x_1)$

Où la variable z_2 exprime la différence en la variable x_2 et sa valeur désirée $\alpha_1(x_1)$.
La dynamique de ces deux variables z_1 et z_2 (voir équation (3.2)) est décrite par les expressions suivantes :

$$\begin{cases} \dot{z}_1 = x_2 + f_1(x_1); \\ \dot{z}_2 = x_3 + f_2(x_1, x_2) + \dot{\alpha}_1(x_1) \;\Leftrightarrow\; \dot{z}_2 = x_3 + \bar{f}_2(z_1, z_2); \end{cases} \tag{3.3}$$

En procédant de façon récursive, on remarque que la variable d'état x_3 (voir les équations (3.3)) représente une commande virtuelle pour la nouvelle variable z_2. Afin de stabiliser z_2 on pose la nouvelle fonction de Lyapunov augmentée : $V_2 = \frac{1}{2}(z_1^2 + z_2^2)$.

En posant un nouveau changement de variable, on obtient : $z_3 = x_3 - \alpha_2(z_1, z_2)$
Avec : $\alpha_2(z_1, z_2) = -z_1 - z_2 - \bar{f}_2(z_1, z_2)$
On obtient : $\dot{z}_2 = z_3 + \alpha_2(z_1, z_2) + \bar{f}_2(x_1, x_2)$
Cela nous permet d'obtenir $\dot{V}_2 < 0$, et donc d'assurer la stabilisation de z_2.

Récursivement, à l'étape i, on obtient les expressions suivantes :

$$\begin{cases} z_{i+1} = x_{i+1} - \alpha_i(z_1, z_2, \dots, z_i) \\ V_i = \frac{1}{2}\sum_{k=1}^{i} z_k^2 \end{cases} \tag{3.4}$$

On choisissant à chaque étape i l'expression de α_i comme suit :

$$\alpha_i(z_1, \dots, z_i) = -z_{i-1} - z_i - \bar{f}_i(z_1, \dots, z_i) \tag{3.5}$$

En dérivant les expressions précédentes (voir les équations (3.4)) on obtient:

$$\begin{cases} \dot{z}_i = z_{i+1} + \alpha_i(z_1, \dots, z_i) + \bar{f}_i(z_1, \dots, z_i) \\ \dot{V}_i = -\sum_{k=1}^{i-1} z_k^2 + z_{i-1}z_i + z_i\left(z_{i+1} + \alpha_i(z_1, \dots, z_i) + \bar{f}_i(z_1, \dots, z_i)\right) \end{cases} \tag{3.6}$$

En introduisant l'expression de α_i (voir l'équation (3.5)), on obtient :

$$\begin{cases} \dot{z}_i = -z_{i-1} - z_i + z_{i+1} \\ \dot{V}_i = -\sum_{k=1}^{i-1} z_k^2 + z_{i+1}z_i \end{cases} \tag{3.7}$$

A l'étape n, on a :

$$\dot{z}_n = \bar{f}_n(z_1, \dots, z_n) + u \tag{3.8}$$

En choisissant l'expression de la commande u comme suit:

$$u = \alpha_n(z_1, \dots, z_n) = -z_{n-1} - z_n - \bar{f}_n(z_1, \dots, z_n) \tag{3.9}$$

En remplaçant l'expression de u dans l'équation (3.8), on obtient :

$$\dot{z}_n = -z_{n-1} - z_n \tag{3.10}$$

La dérivée de la fonction de Lyapunov prend la forme suivante :

$$\dot{V}_n = -\sum_{i=1}^{n} z_n^2 \qquad Avec: \qquad \dot{V}_n < 0 \tag{3.11}$$

Ce qui prouve que le système est stable et les variables d'états tendent asymptotiquement vers les valeurs désirées.

D'autres méthodes de Backstepping existent :

- **Le Backstepping Intégrateur** [Koko.,95]**:**

Soit le système multivariable non linéaire suivant :

$$\dot{x} = f(x) + g(x)\, u$$

La commande u est considérée comme un état ou encore commande virtuelle servant à stabiliser certaines variables. L'expression de la commande élaborée v représente la dérivée de $u(x)$: $v = \dot{u}$

- **Le Backstepping adaptatif** [Khal.,95]**:**

Cette méthode est destinée aux systèmes dont les paramètres ne sont pas connus, ce qui nécessite une estimation des paramètres ainsi que leur ajustement durant l'élaboration progressive des lois de commandes.

3.3 La commande par Mode Glissant

La commande par mode glissant est une commande à structure variable [Agra.,08], discontinue. Elle peut commuter entre deux valeurs différentes suivant le signe d'une fonction contrainte connue aussi sous le nom de surface de glissement [Utki.,77]. Cette dernière est exprimée en fonction des variables d'états du système à commander. Elle est représentée par un hyperplan dans l'espace d'état du système. La commutation de la commande permet d'atteindre et de maintenir la fonction contrainte autour de zéro ce qui permet de stabiliser les variables d'états du système autour des valeurs désirées. La commande à structure variable est simple à réaliser, robuste vis-à-vis des perturbations, des non linéarités et des variations paramétriques des systèmes à commander [Perr.,02].

3.3.1 La commande par mode glissant classique

Dans les modes glissant d'ordre un, la discontinuité de la commande intervient dans la dérivée première de l'expression de la surface de glissement.

- ***Conception de la commande par modes glissants*** [Utki.,77]

Soit le système à commander :

$$\dot{x} = f(x) + g(x)\, u(x,t) \tag{3.12}$$

Avec : x : les variables d'états, f et g : des champs de vecteurs, u : entrée de commande.

L'élaboration d'une commande à structure variable [Perr.,02] nécessite en premier lieu le choix des surfaces de glissement exprimées en fonction de tous les états du système (3.12). Le nombre de surfaces de glissement peut aller jusqu'à $(2^m - 1)$ surfaces, m étant le nombre d'entrée du système. En deuxième lieu, on doit établir les conditions de convergence pour lesquelles la commande ainsi que la loi de commutation rend le système stable et fait converger des variables d'états du système vers les positions désirées. Enfin, la commande $u(x,t)$ qui dépend de la surface de glissement $s(x,t)$ est de la forme suivante :

$$u(x,t) = \begin{cases} u^+(x,t) & si \quad s(x,t) > 0 \\ u^-(x,t) & si \quad s(x,t) < 0 \end{cases} \qquad u^+(x,t) > u^-(x,t) \tag{3.13}$$

Avec : $u^+(x,t), u^-(x,t)$ sont des fonctions continues.

Le schéma synoptique de la commande à structure variable en boucle fermé est représenté par la figure (Figure 3.1) ci-dessous.

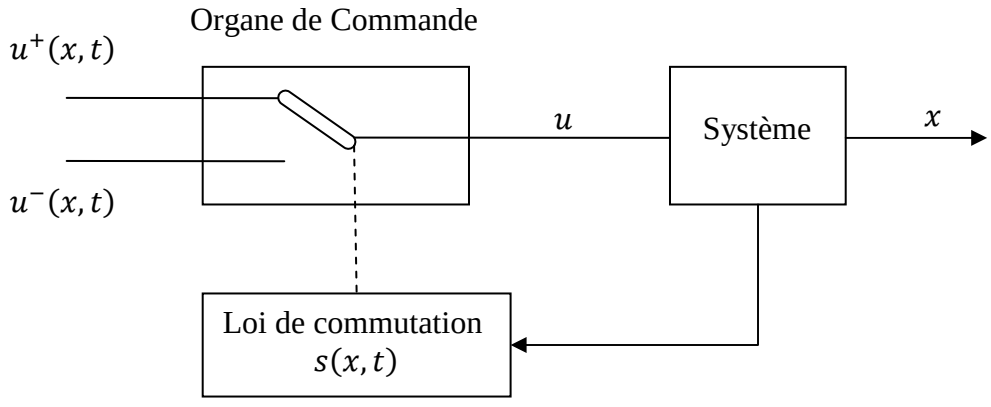

Figure 3.1 : Schéma de principe d'un système à structure variable

La commande va permettre au système d'atteindre le régime glissant défini dans le cas idéal par $s(x,t) = 0$, comme on le voit dans la figure 3.2 .

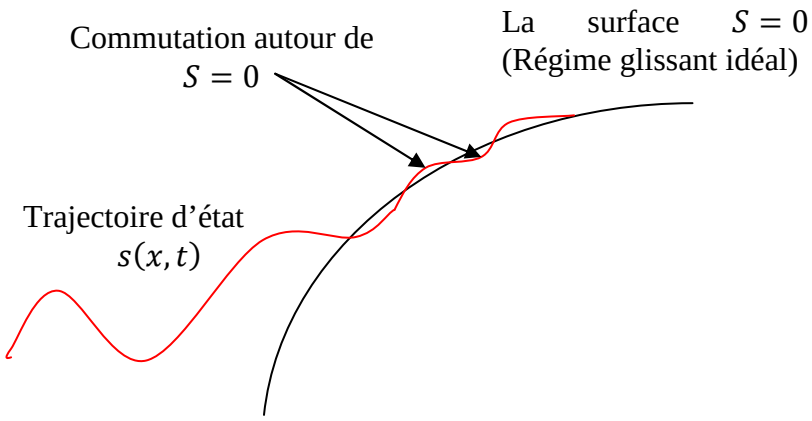

Figure 3.2 : Convergence du système vers le régime glissant

Cette méthode a un inconvénient principal qui est le phénomène du chattering que l'on détaillera ci-dessous, ce phénomène qui peut endommager les actionneurs du système à commander.

- **Condition d'existence du régime glissant** [Utki.,77]**:**

Le régime glissant existe lorsque l'on a l'inégalité suivante :

$$s\,\dot{s} < 0 \qquad (3.14)$$

Elle est aussi appelée condition d'attractivité.

Un régime glissant existe sur une surface de glissement si, et seulement si, dans un voisinage de la surface de glissement toutes les trajectoires du système sont dirigées vers elle (voir Figure 3.2).

En d'autres termes.

$$\lim_{s(x)\to 0^-} \dot{s}(x) > 0 \quad \text{et} \quad \lim_{s(x)\to 0^+} \dot{s}(x) < 0$$

- **La commande équivalente :**

Soit le système multivariable et non linéaire (3.12) :

$$\dot{x} = f(x) + g(x)\,u(x,t)$$

Le calcul de la commande équivalente permet au système à commander d'évoluer quand le régime glissant idéal est établi. Cette commande représente la valeur moyenne que

peut prendre la grandeur de la commande lors de la commutation rapide entre $u^+(x,t)$ et $u^-(x,t)$. Cette commutation permet de maintenir la trajectoire d'état sur la surface de glissement $s = 0$. Le comportement du système est régi par deux conditions :

$$s(x,t) = 0 \quad et \quad \dot{s}(x,t) = 0$$

A partir de l'équation $\dot{s}(x,t) = 0$, on déduit l'expression de la commande équivalente qui est donnée par:

$$\frac{ds(x,t)}{dt} = \left(\frac{ds}{dx}\right)^T . \frac{dx}{dt} + \frac{ds}{dt} = \left(\frac{ds}{dx}\right)^T . \left[f(x,t) + g(x,t)\, U_{eq}\right] + \frac{ds}{dt} = 0 \qquad (3.15)$$

Si on pose la condition d'existence suivante :

$$\left[\left(\frac{ds}{dx}\right)^T g(x,t)\right] \neq 0 \qquad (3.16)$$

La commande équivalente est donnée par :

$$U_{eq}(x,t) = -\left[\left(\frac{ds}{dx}\right)^T g(x,t)\right]^{-1} \left[\left(\frac{ds}{dx}\right)^T . f(x,t) + \frac{ds}{dt}\right] \qquad (3.17)$$

En remplaçant $u(x,t)$ par l'expression de $U_{eq}(x,t)$ dans l'équation (3.12), le vecteur d'état $x(t)$ devient solution du système (3.12), ce dernier est exprimé par l'équation différentielle suivante:

$$\frac{dx}{dt} = \left\{I - g(x,t)\left[\left(\frac{ds}{dx}\right)^T . g(x,t)\right]^{-1}\left(\frac{ds}{dx}\right)^T\right\} f(x,t) - g(x,t)\left[\left(\frac{ds}{dx}\right)^T g(x,t)\right]^{-1}\left(\frac{ds}{dx}\right)^T \qquad (3.18)$$

- **Propriété de robustesse**

L'avantage de la commande par mode glissant est la robustesse face aux perturbations. Soit le système perturbé suivant :

$$\dot{x} = f(x,t) + g(x,t)\, u(x,t) + p(x,t) \qquad (3.19)$$

Où :

x : représente les variables d'états

f et g : des champs de vecteurs.

$u(x,t)$: l'entrée de commande.

$p(x,t)$: des perturbations.

Le mode glissant idéal est indépendant du champ de vecteur $g(x,t)$. On en déduit que si un champ de vecteur $g_i(x,t)$ appartient à l'espace engendré par la base $g(x)$ ($span\{g(x)\}$), le mode glissant idéal est indépendant de $g_i(x,t)$.

Théorème :

Un régime glissant sur s, du système perturbé, est indépendant du signal de perturbation $p(x,t)$, si et seulement si, celui–ci est borné et vérifie :

$$p(x,t) \in span\{g(x)\} \tag{3.20}$$

Cette condition est appelée condition de recouvrement ou dans la dénomination internationale "matching condition" [Hame2.,07].

- **Phénomène du Chattering**

Pour maintenir la trajectoire d'état du système sur la surface de glissement afin d'atteindre le régime glissant idéal, la commande doit commuter à une fréquence infinie, ce qui est physiquement irréalisable. En pratique la commutation à haute fréquence autour de la surface (voir la Figure 3.3 ci-dessous) crée un phénomène de réticence au niveau des actionneurs et qui se traduit par des vibrations. Ce phénomène est appelé chattering ou broutement.

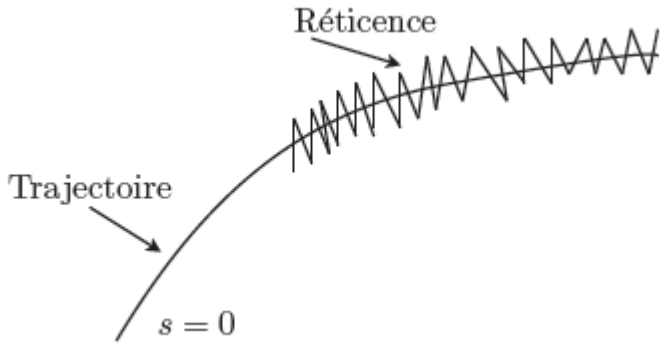

Figure 3.3 : Phénomène de Chattering

3.3.2 La commande par mode glissant d'ordre supérieur

La commande par mode glissant d'ordre supérieur apporte une alternative intéressante pour diminuer l'effet du chattering sans pour autant oublier le principal atout des modes

glissant qui est la robustesse. La commande discontinue agit sur la dérivée d'ordre supérieur de la surface de glissement [Leva.,93].

Une commande par mode glissant d'ordre ρ doit maintenir à zéro l'ensemble de glissement S_ρ qui est constitué de la contrainte de glissement $s(x,t)$ ainsi que ses $\rho-1$ dérivés.

$$S_\rho = \{x \in X : s = \dot{s} = \ddot{s} = \cdots = s^{\rho-1} = 0\} \tag{3.21}$$

Définition : [Leva.,93]

On dit que la loi de commande u est un algorithme glissant idéal d'ordre ρ par rapport à S_ρ si elle génère une solution au sens de Filippov sur S_ρ , i.e.

$$s = \dot{s} = \ddot{s} = \cdots = s^{\rho-1} = 0 \tag{3.22}$$

Pour contrôler un système par une commande à mode glissant d'ordre ρ, il faut contraindre le système à glisser sur une surface qui représente une combinaison de la surface de glissement et de ces $(\rho-1)$ dérivées.

Les algorithmes de commande par modes glissants d'ordre supérieur sont essentiellement connu pour $(\rho=2)$. Il existe néanmoins des travaux qui permettent d'élaborer des commandes pour des régimes glissants d'ordre quelconque.

Définition : (degré relatif).

Le degré relatif représente le nombre de fois que doit être dérivée la sortie par rapport au temps, pour faire apparaître l'entrée de façon explicite.

3.3.2.1 Commande par mode glissant d'ordre deux

La commande par mode glissant d'ordre deux [Leva.,06] est utilisée dans le cas où le degré relatif du système est égal à 1 ou 2. Le système atteint le régime glissant lorsque l'ensemble de glissement S_2 est aux alentours de zéro, c'est-à-dire $S_2 = \{x \in X : s = \dot{s} = 0\}$ (voir Figure 3.4).

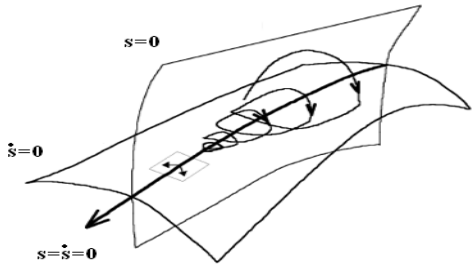

Figure 3.4 : Ensemble de glissement d'ordre 2

Soit le système à commander suivant :

$$\dot{x} = f(x,t) + g(x,t)\, v \tag{3.23}$$

La discontinuité de la commande intervient dans l'expression de la dérivée seconde $\ddot{s}$ qui s'écrit sous la forme suivante :

$$\ddot{s} = \alpha(x,t) + \beta(x,t)\, v \tag{3.24}$$

Avec

$v = \dot{u}$ dans le cas où le système est de degré relatif $\rho = 1$ par rapport à s.

$v = u$ dans le cas où le système est de degré relatif $\rho = 2$ par rapport à s.

Cette commande entraine le système de degré relatif un ou deux vers un régime glissant sous certaines hypothèses :

$$|\alpha| < \Phi, 0 < \Gamma_m \leq \beta \leq \Gamma_M, \Phi > 0 \ .$$

Plusieurs algorithmes ont été élaborés dans le cas d'une commande par mode glissant d'ordre deux. Parmi ces algorithmes, on cite : le Drift, le sub-optimal, le twisting, le super twisting [Frid.,02].

L'algorithme du twisting :

Dans cet algorithme, on a besoin d'information concernant le signe de la surface de glissement ainsi que sa dérivée.

Pour les systèmes de degré relatif égal à un, l'expression de la commande est la suivante :

$$\dot{u} = \begin{cases} -u & si \quad |u| > U_M > 0 \\ -V_m\, sign(s) & si \quad s\dot{s} \leq 0;\ |u| \leq U_M \\ -V_M\, sign(s) & si \quad s\dot{s} > 0;\ |u| \leq U_M \end{cases} \tag{3.25}$$

Les conditions de convergence en temps fini sont données par: [Perr.,02]

$$V_M > V_m, \qquad V_M > \frac{\Phi}{\Gamma_m}, \qquad \Gamma_m V_M - \Phi > \Gamma_M V_m + \Phi$$

La commande u obtenue va permettre d'annuler s et $\dot{s}$ (voir Figure 3.5).

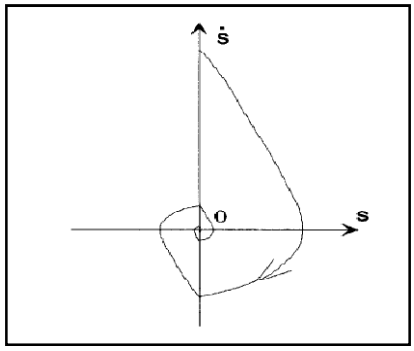

Figure 3.5 : Trajectoire du système avec l'algorithme du twisting

L'algorithme du super twisting [Frid.,02]**:**

L'algorithme du super twisting a été développé pour appliquer la commande par mode glissant d'ordre deux sur les systèmes dont le degré relatif est égal à 1 tout en évitant le chattering causé par les modes glissants classiques.

L'expression de la commande u est composée d'une partie continue et d'une seconde partie discontinue.

Cet algorithme nécessite des informations sur le module est le signe de la surface de glissement uniquement. Lorsque le système atteint le régime glissant les variables s et $\dot{s}$ s'annulent (voir Figure 3.6).

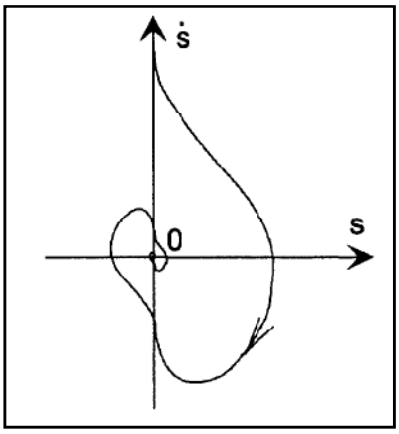

Figure 3.6: Trajectoire du système avec l'algorithme du super-twisting

L'expression de la commande est la suivante :

$$u(t) = u_1(t) + u_2(t) \tag{3.26}$$

Avec :

$$\dot{u}_1(t) = \begin{cases} -u & si \quad |u| > 1 \\ -W\ sign(s_1) & si \quad |u| \leq 1 \end{cases} \tag{3.27}$$

$$u_2(t) = \begin{cases} -\lambda\ |s_0|^{\rho} sign(s_1) & si \quad |s_1| > s_0 \\ -\lambda\ |s_1|^{\rho} sign(s_1) & si \quad |s_1| \leq s_0 \end{cases} \tag{3.28}$$

Avec :

W, λ : des gains de la loi de commutation.

s_1 : la surface de glissement

Sachant que les conditions suffisantes pour une convergence en temps fini vers le régime glissant sont :

$$W > \frac{\Phi}{\Gamma_m}$$

$$\lambda^2 > \frac{4\ \Phi}{\Gamma_m^2} \frac{\Gamma_M\ (w + \Phi)}{\Gamma_m\ (w - \Phi)}$$

$$0 < \rho \leq 0.5$$

L'expression de la commande est simplifiée lorsque le système à commander est linéairement dépendant de la commande, $u(t)$ n'a pas besoin d'être bornée est $s_0 = 0$. Ce qui donne :

$$\begin{cases} u(t) = -\lambda\ |s|^{\rho} sign(s) + u_1(t) \\ \dot{u}_1(t) = -W\ sign(s) \end{cases} \tag{3.29}$$

Les différentes commandes qui vont être élaborées seront appliquée à un système non holonome de type voiture en double braquage. Pour cela l'élaboration d'un modèle représentatif du système à commander dans l'espace d'état est nécessaire.

3.4 Application au robot mobile de type voiture : Le Modèle de poursuite [Laum.,01]

Le modèle de poursuite est une représentation des erreurs de poursuite du système par rapport à un robot de référence. Ce dernier exprime la trajectoire de référence que doit suivre

le robot mobile. Dans ce chapitre on développe le modèle d'erreur du robot mobile de type voiture en double braquage, puis on élabore un nouveau modèle équivalent au précédent. Le nouveau modèle exprime le système dans l'espace d'état, afin de nous permettre par la suite d'élaborer des commandes par retour d'état.

3.4.1 Le modèle d'erreur de poursuite

Il représente l'erreur des états du système par rapport au robot de référence qu'il poursuit. Le robot de référence représente la configuration liée à la trajectoire que doit suivre le robot réel à chaque instant (voir Figure 3.7).

Soit le modèle cinématique du robot de référence :

$$\begin{cases} \dot{x}_r = u_{1r}\, cos(\theta_r + \varphi_r) \\ \dot{y}_r = u_{1r}\, sin(\theta_r + \varphi_r) \\ \dot{\theta}_r = \dfrac{u_{1r}}{L} 2\sin(\varphi_r) \\ \dot{\varphi}_r = u_{2r} \end{cases} \tag{3.30}$$

Où (x_r, y_r) représentent les coordonnées cartésiennes du robot de référence. L'angle θ_r représente l'angle d'orientation par rapport à l'axe des abscisses. L'angle φ_r représente l'angle de braquage des roues avant et arrière. u_{1r} et u_{2r} représentent les entrées de commande.

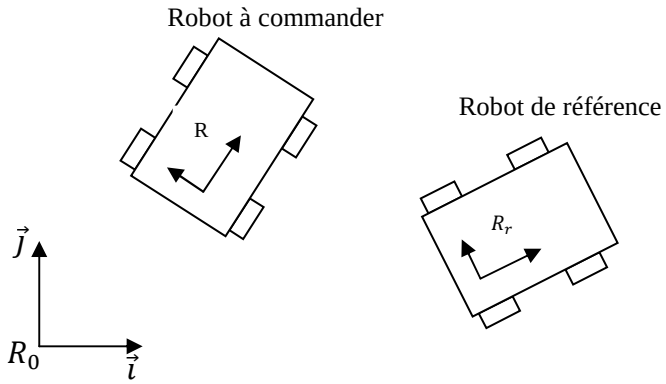

Figure 3.7 : Représentation du robot de réf et du robot réel

L'erreur du système est donnée par le vecteur d'erreur « e » suivant :

$$e = (x - x_r, y - y_r, \theta - \theta_r, \varphi - \varphi_r)$$

L'erreur de suivi en position est exprimée dans le repère du robot de référence.

$$\begin{bmatrix} e_{1r} \\ e_{2r} \\ e_{3r} \\ e_{4r} \end{bmatrix} = \begin{bmatrix} cos(\theta_r) & sin(\theta_r) & 0 & 0 \\ -sin(\theta_r) & cos(\theta_r) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} e_1 \\ e_2 \\ e_3 \\ e_4 \end{bmatrix} \tag{3.31}$$

On obtient :

$$\begin{cases} e_{1r} = cos(\theta_r)e_1 + sin(\theta_r)e_2 \\ e_{2r} = -sin(\theta_r)e_1 + cos(\theta_r)e_2 \\ e_{3r} = e_3 \\ e_{4r} = e_4 \end{cases} \tag{3.32}$$

La dynamique de l'erreur est exprimée par le système d'équations (3.33) suivant :

$$\begin{cases} \dot{e}_{1r} = v\ cos(e_{3r} + \varphi) - v_r cos(\varphi_r) + e_{2r}\ 2\ \dfrac{v_r}{L} sin(\varphi_r) \\ \dot{e}_{2r} = v\ sin(e_{3r} + \varphi) - v_r sin(\varphi_r) - e_{1r}\ 2\ \dfrac{v_r}{L} sin(\varphi_r) \\ \dot{e}_{3r} = \dfrac{2}{L}\left(v\ sin(\varphi) - v_r sin(\varphi_r)\right) \\ e_{4r} = \varphi - \varphi_r \end{cases} \tag{3.33}$$

L'expression de la dynamique de l'erreur n'est pas une représentation d'état du système à commander. Elle ne peut être exploitée dans le cas où l'on applique une commande dans l'espace d'état. Un changement de coordonnées est nécessaire pour obtenir une nouvelle expression du système équivalente à la première et exploitable dans le cas d'une commande par retour d'état.

- **Le difféomorphisme** [Slot.,98] [Levi.,04]

Le diffeomorphisme est une généralisation du changement de variable, il est utilisé pour transformer un système non linéaire en un autre système non linéaire avec nouvelle variable d'état. Ce qui permet d'analyser et de résoudre certains problèmes de contrôle comme la stabilisation et la stabilité du système à commander.

Soit le système non linéaire suivant :

$$\dot{x} = f(x) + g(x)u$$

On pose :

$$z(x) = \Phi(x) = \begin{bmatrix} \Phi_1(x_1, \dots, x_n) \\ \Phi_2(x_1, \dots, x_n) \\ \vdots \\ \Phi_n(x_1, \dots, x_n) \end{bmatrix} \tag{3.34}$$

tels que : $\begin{cases} \Phi(x) \text{ est inversible} \\ \Phi(x) \text{ est de classe } \subset^k \ (\Phi(x) \ est \ dérivable \ k \ fois) \end{cases}$

et : $x(z) = \Phi^{-1}(\mathrm{z})$

Cette transformation est dite un difféomorphisme local de $\mathbb{R}^n$.

Le nouveau système exprimé par les variables d'états z prend la forme suivante :

$$\dot{z} = \frac{dz}{dt} = \frac{d\Phi}{dx}\frac{dx}{dt} \tag{3.35}$$

$$\dot{z} = \frac{d\Phi}{dx}\, f(x)|_{x=\Phi^{-1}(\mathrm{z})} + \frac{d\Phi}{dx}\, g(x)\, u|_{x=\Phi^{-1}(\mathrm{z})} \tag{3.36}$$

3.4.2 Mise en forme du modèle de poursuite

Afin de pouvoir commander le système dans l'espace d'état. On propose une mise en forme du modèle d'erreur en appliquant un changement de variable par difféomorphisme. En s'inspirant du travail effectué sur la mise en forme par difféomorphisme du modèle de poursuite du robucar simple braquage [Hame2.,07] [Laum.,01], nous proposons dans notre cas (Double braquage) le changement de variable suivant, donné par les nouvelles variables d'états:

$$\begin{cases} z_1 = e_{1r} \\ z_2 = e_{2r} \\ z_3 = tan(e_{3r}) \\ z_4 = (tan(e_3 + \varphi) - z_3)\sqrt{(z_3^2 + 1)} - tan(\varphi_r)(z_3^2 + 1) + k\, z_2 \end{cases} \tag{3.37}$$

Ce changement de variable nous permet de démontrer la nouvelle expression du modèle d'erreur de poursuite

Démonstration :

La transformation du modèle de poursuite (3.33) est réalisée en utilisant le changement de variable (3.37). A partir des équations du modèle (3.33), de nouvelles équations sont déduites. Ces équations doivent s'exprimer uniquement avec les variables d'erreurs z_i ($i = 1,2,3,4$).

Expression de la dérivée de z_1 :

A partir du changement de variable (3.37), on déduit que la dérivée de la variable z_1 est égale à la dérivée de e_{1r} (voir l'équation (3.33)). Il en résulte :

$$\dot{z}_1 = v\ cos(e_{3r} + \varphi) - v_r\ cos(\varphi_r) + e_{2r}\dot{\theta}_r \tag{3.38}$$

On choisit la variable ω_1 comme étant la nouvelle entrée de commande qui prend la forme suivante :

$$w_1 = v\ cos(e_{3r} + \varphi) - v_r\ cos(\varphi_r) \tag{3.39}$$

Comme les variables z_2 et e_{2r} sont identiques (voir le changement de variable (3.37), on déduit que l'expression de $\dot{z}_1$ s'écrit sous la forme suivante.

$$\dot{z}_1 = w_1 + z_2\dot{\theta}_r \tag{3.40}$$

Expression de la dérivée de z_2 :

L'expression de z_2 est la suivante (voir les équations (3.32),(3.37)):

$$z_2 = -sin(\theta_r)e_1 + cos(\theta_r)e_2 \tag{3.41}$$

La dérivée de z_2 prend la forme suivante :

$$\dot{z}_2 = v\ sin(e_{3r} + \varphi) - v_r\ sin(\varphi_r) - e_{1r}\dot{\theta}_r \tag{3.42}$$

Dans le but de faire apparaitre l'expression de l'entrée de commande ω_1 dans l'équation (3.42), on développe l'équation (3.42) comme suit :

$$\dot{z}_2 = v\ cos(e_{3r} + \varphi)\ tan(e_{3r} + \varphi) - v_r\ sin(\varphi_r) - z_1\dot{\theta}_r \tag{3.43}$$

On déduit de l'équation (3.39) que :

$$v\ cos(e_{3r} + \varphi) = w_1 + v_r\ cos(\varphi_r) \tag{3.44}$$

En remplaçant l'équation (3.44) dans l'équation (3.43) on obtient :

$$\dot{z}_2 = \big(w_1 + v_r\ cos(\varphi_r)\big)\ tan(e_{3r} + \varphi) - v_r\ sin(\varphi_r) - z_1\dot{\theta}_r \tag{3.45}$$

Expressions des dérivées de z_3 et z_4 :

La variable z_3 s'exprime comme suit (voir le changement de variable (3.37)) : $z_3 = tan(e_{3r})$.

En dérivant la variable z_3 on obtient l'expression suivante :

$$\dot{z}_3 = \frac{2}{L}\frac{v\sin(\varphi) - v_r\sin(\varphi_r)}{cos^2(e_{3r})} \tag{3.46}$$

Dans ce qui suit, on développe l'expression (3.46), jusqu'à faire apparaitre les nouvelles variables d'états :

En développant les expressions de $[v_r\ sin(\varphi_r)]$ et $[v\ sin(\varphi)]$, on obtient :

$$v_r\ sin(\varphi_r) = \frac{v\ cos(e_{3r}+\varphi) - w_1}{cos(\varphi_r)} sin(\varphi_r) = v\ cos(e_{3r}+\varphi)\ tan(\varphi_r) - w_1 tan(\varphi_r) \tag{3.47}$$

$$v\ sin(\varphi) = v\ sin(e_{3r}+\varphi-e_{3r}) = v\,[\sin(e_{3r}+\varphi)\cos(e_{3r}) - \cos(e_{3r}+\varphi)\sin(e_{3r})] \tag{3.48}$$

En remplaçant les expressions de $[v_r\ sin(\varphi_r)]$, $[v\ sin(\varphi)]$ par les expressions (3.47), (3.48), on obtient :

$$\dot{z}_3 = \frac{2}{L}\frac{v\,[\sin(e_{3r}+\varphi)\cos(e_{3r}) - \cos(e_{3r}+\varphi)\sin(e_{3r})] - [v\ cos(e_{3r}+\varphi)\ tan(\varphi_r) - w_1 tan(\varphi_r)]}{cos^2(e_{3r})} \tag{3.49}$$

A l'aide de relations trigonométriques, on simplifie l'expression (3.49), on aboutit à l'équation on suivante :

$$\dot{z}_3 = \frac{2}{L}\left[\left(w_1 + v_r\ cos(\varphi_r)\right)\frac{[tan(e_{3r}+\varphi)\ \cos(e_{3r}) - \sin(e_{3r}) - tan(\varphi_r)]}{cos^2(e_{3r})} + w_1\frac{tan(\varphi_r)}{cos^2(e_{3r})}\right] \tag{3.50}$$

$$\dot{z}_3 = \frac{2}{L}\left[\left(w_1 + v_r\ cos(\varphi_r)\right)\left[\frac{tan(e_{3r}+\varphi)}{cos(e_{3r})} - \frac{tan(e_{3r})}{cos(e_{3r})} - \frac{tan(\varphi_r)}{cos(e_{3r})}\right] + w_1\frac{tan(\varphi_r)}{cos^2(e_{3r})}\right] \tag{3.51}$$

On sait que $\{z_3 = tan(e_{3r})\}$, on en déduit que :

$$\frac{1}{cos^2(e_{3r})} = 1 + tan^2(e_{3r}) = 1 + z_3^2 \tag{3.52}$$

En remplaçant l'expression du $[cos(e_{3r})]$ (voir l'équation (3.52)) dans l'équation (3.51) on obtient :

$$\dot{z}_3 = \frac{2}{L}\left(w_1 + v_r\ cos(\varphi_r)\right)\left[(tan(e_{3r}+\varphi) - z_3^2)\sqrt{1+z_3^2}\ tan(\varphi_r)(1+z_3^2)\right] + \frac{2}{L}w_1 tan(\varphi_r)(1+z_3^2) \tag{3.53}$$

Dans le but d'exprimer la dérivée de z_3 en fonction de la variable z_2 on introduit le paramètre $\{k\ z_2\}$ dans l'équation (3.53):

$$\dot{z}_3 = \frac{2}{L}\Big(w_1 + v_r\ cos\big(\varphi_r\big)\Big)\left[(tan(e_{3r}+\varphi) - z_3^2)\sqrt{1+z_3^2}\ tan\big(\varphi_r\big)(1+z_3^2) + k\ z_2 - k\ z_2\right] + \frac{2}{L}w_1 tan\big(\varphi_r\big)(1+z_3^2) \quad (3.54)$$

Afin d'éviter l'apparition des variables d'erreurs de poursuite e_{3r} (voir l'équation (3.32)), on exprime la quatrième variable d'état z_4 comme suit :

$$z_4 = (tan(e_{3r}+\varphi) - z_3^2)\sqrt{1+z_3^2}\ tan(\varphi_r)(1+z_3^2) + k\ z_2 \quad (3.55)),$$

En remplaçant l'expression (3.55) de la variable z_4 dans l'expression (3.54), on obtient :

$$\dot{z}_3 = \frac{2}{L}\Big(\omega_1 + v_r\ cos\big(\varphi_r\big)\Big)\left[z_4 - k\ z_2\right] + \frac{2}{L}w_1 tan\big(\varphi_r\big)(1+z_3^2) \quad (3.56)$$

En développant l'expression (3.56), on obtient :

$$\dot{z}_3 = \frac{2}{L}\{v_r\ z_4\ cos(\varphi_r) - v_r\ k\ z_2\ cos(\varphi_r) + w_1[z_4 - k\ z_2 + tan(\varphi_r)(1+z_3^2)]\} \quad (3.57)$$

En posant :

$$h_1 = z_4 - k\ z_2 + tan(\varphi_r)(1+z_3^2) \quad (3.58)$$

On en déduit que :

$$\dot{z}_3 = v_r\ z_4\ cos(\varphi_r) - v_r\ k\ z_2\ cos(\varphi_r) + \omega_1 h_1 \quad (3.59)$$

A partir de l'expression (3.55), on déduit l'expression de $\{tan(e_{3r}+\varphi)\}$:

$$h_2 = tan(e_{3r}+\varphi) = \frac{z_4 - k\ z_2 + tan(\varphi_r)(1+z_3^2)}{\sqrt{1+z_3^2}} + z_3 \quad (3.60)$$

La dérivée de la variable z_4 est considérée comme la seconde entrée de commande w_2

On obtient la nouvelle représentation du modèle de poursuite :

$$\begin{cases} \dot{z}_1 = w_1 + z_2\dot{\theta}_r \\ \dot{z}_2 = v_r\big(h_2\ cos(\varphi_r) - cos(\varphi_r)\big) - z_1\dot{\theta}_r + w_1 h_2 \\ \dot{z}_3 = v_r\ z_4 \cos(\varphi_r) - v_r\ k\ z_2 \cos(\varphi_r) + w_1 h_1 \\ \dot{z}_4 = w_2 \end{cases} \quad (3.61)$$

Avec :

$$\begin{cases} h_1 = z_4 - k\, z_2 + tan(\varphi_r)(z_3^2 + 1) \\ h_2 = \dfrac{z_4 - k\, z_2 + tan(\varphi_r)(z_3^2 + 1)}{\sqrt{(z_3^2 + 1)}} + z_3 \end{cases} \quad (3.62)$$

Et :

$$\begin{cases} w_1 = v\, cos(e_3 + \varphi) - v_r\, cos(\varphi_r) \\ w_2 = \dot{z}_4 \end{cases} \quad (3.63)$$

Avec : $k > 0$, $e_3 \in \left]-\frac{\pi}{2}, \frac{\pi}{2}\right[$, $\varphi_r \in \left]-\frac{\pi}{2}, \frac{\pi}{2}\right[$, Et : $vr \geq 0$

L'élaboration du modèle de poursuite sous forme d'une représentation d'état (voir le système (3.61)) est dédié au robot mobile de type voiture en double braquage. Ce modèle est équivalent au modèle d'erreur du robot. Il est obtenu grâce à un changement de variable par difféomorphisme qui nous permet d'exprimer le système à commander dans l'espace d'état.

La poursuite de trajectoire qu'exécute le robot réel pour atteindre les configurations du robot de référence à chaque instant t est équivalente à la stabilisation des états du modèle d'erreur de poursuite autour de zéro, c'est-à-dire que les variable z_i $(i = 1,2,3,4)$ tendent vers zéro.

3.5 Elaboration des lois commandes

3.5.1 La commande par Backstepping

Soit le système à commander représenté par le modèle de poursuite (voir le système (3.61)). Ce système présente un inconvénient lié à l'entrée de commande w_1 qui apparaît de façon explicite dans plusieurs variables d'état. En stabilisant progressivement tous les états du système, on obtient plusieurs expressions de la commande w_1. Afin d'avoir une seule expression, on applique le principe du Backstepping intégrateur (voir le paragraphe (3.2)) sur les commandes w_1, w_2.

L'élaboration des commandes dépend en partie du choix des variables d'états à stabiliser. On présente ci-dessous trois méthodes à partir desquels on élabore les expressions des commandes par Backstepping.

Méthode 1 :

Dans cette méthode, on utilise le principe du Backstepping intégrateur uniquement pour la première entrée de commande pour avoir w_1 pour avoir une nouvelle entrée de commande v_1 telle que : $v_1 = \dot{w}_1$. La seconde entrée de commande w_2 reste la même.

Etape 1 : stabilisation de z_1

L'expression de la fonction de Lyapunov est comme suit :

$$V(z_1) = \frac{1}{2} z_1^2 \tag{3.64}$$

La dérivée de la fonction de Lyapunov est :

$$\dot{V}(z_1) = \dot{z}_1 z_1 \tag{3.65}$$

L'expression de la dérivée de la variable z_1 est comme suit (voir le système (3.61)):

$$\dot{z}_1 = w_1 + z_2\dot{\theta}_r$$

On en déduit que la dérivée de la fonction de Lyapunov prend la forme suivante:

$$\dot{V}(z_1) = z_1\left(w_1 + z_2\dot{\theta}_r\right) \tag{3.66}$$

Pour que la dérivée de la fonction de Lyapunov soit strictement négative, on pose :

$$w_1 + z_2\dot{\theta}_r = -z_1 \tag{3.67}$$

On remplaçant l'expression (3.67) dans l'expression (3.66), on obtient :

$$\dot{V}(z_1) = -z_1^2 \tag{3.68}$$

On déduit de l'expression (3.68) que la variable z_1 est stable au sens de Lyapunov.

On en déduit un nouveau changement de variable en posant :

$$X_1 = w_1 + z_2\dot{\theta}_r - \Phi(z_1) \text{ avec } \Phi(z_1) = -z_1$$

On obtient la nouvelle variable d'état :

$$X_1 = w_1 + z_2\dot{\theta}_r + z_1 \tag{3.69}$$

Etape 2 : stabilisation de X_1

Soit l'expression de la fonction de lyapunov augmentée :

$$V(z_1, X_1) = \frac{1}{2} z_1^2 + \frac{1}{2} X_1^2 \tag{3.70}$$

La dérivée de la fonction de Lyapunov est :

$$\dot{V}(z_1, X_1) = \dot{V}(z_1) + \dot{X}_1 X_1 \tag{3.71}$$

Comme : $\{ \dot{V}(z_1) = -z_1^2 \}$ (voir équation (3.68)), on en déduit que :

$$\dot{V}(z_1, X_1) = -z_1^2 + \dot{X}_1 X_1 \tag{3.72}$$

Avec:

$$\dot{X}_1 = \dot{w}_1 + \dot{z}_2\dot{\theta}_r + z_2\ddot{\theta}_r + \dot{z}_1 \tag{3.73}$$

Pour que $\dot{V}(z_1, X_1)$ (voir l'équation (3.72)) soit strictement négative, on pose :

$$\dot{w}_1 + \dot{z}_2\dot{\theta}_r + z_2\ddot{\theta}_r - z_1 = -X_1 \tag{3.74}$$

Il en résulte que $\dot{V}(z_1, X_1)$ devient strictement négative :

$$\dot{V}(z_1, X_1) = -z_1^2 - X_1^2 \tag{3.75}$$

En remplaçant X_1 par sont expression (3.69) dans l'équation (3.74), On en déduit l'expression de la commande v_1 :

$$v_1 = -\dot{z}_2\dot{\theta}_r - z_2\left(\ddot{\theta}_r + \dot{\theta}_r\right) - w_1 \tag{3.76}$$

Avec : $v_1 = \dot{w}_1$

Etape 3 : stabilisation de z_2

Soit l'expression de la fonction de Lyapunov augmentée :

$$V(z_1, X_1, z_2) = \frac{1}{2}\, z_1^2 + \frac{1}{2}\, X_1^2 + \frac{1}{2}\, z_2^2 \tag{3.77}$$

La dérivée de la fonction de Lyapunov est:

$$\dot{V}(z_1, X_1, z_2) = \dot{V}(z_1, X_1) + \dot{z}_2\, z_2 \tag{3.78}$$

Comme : { $\dot{V}(z_1, X_1) = -z_1^2 - X_1^2$ } (voir équation (3.75)), on en déduit que :

$$\dot{V}(z_1, X_1, z_2) = -z_1^2 - X_1^2 + \dot{z}_2\, z_2 \tag{3.79}$$

Avec : $\dot{z}_2 = w_1 h_2 + v_r\left(h_2 cos(\varphi_r) - sin(\varphi_r)\right) - z_1\dot{\theta}_r$

Pour que $\dot{V}(z_1, X_1, z_2)$ (voir l'équation (3.79)) soit strictement négative, on pose :

$$w_1 h_2 + v_r\left(h_2 cos(\varphi_r) - sin(\varphi_r)\right) - z_1\dot{\theta}_r = -z_2 \tag{3.80}$$

Il en résulte que $\dot{V}(z_1, X_1, z_2)$ devient strictement négative :

$$\dot{V}(z_1, X_1, z_2) = -z_1^2 - X_1^2 - z_2^2 \tag{3.81}$$

On pose le changement de variable suivant :

$$X_2 = w_1 h_2 + v_r\left(h_2 cos(\varphi_r) - sin(\varphi_r)\right) - z_1\dot{\theta}_r - \Phi(z_2) \text{ avec } \Phi(z_2) = -z_2$$

On obtient la nouvelle variable d'état :

$$X_2 = w_1 h_2 + v_r\left(h_2 cos(\varphi_r) - sin(\varphi_r)\right) - z_1\dot{\theta}_r + z_2 \tag{3.82}$$

Etape 4 : stabilisation de X_2

Soit l'expression de la fonction de Lyapunov augmentée :

$$V(z_1, X_1, z_2, X_2) = \frac{1}{2}\, z_1^2 + \frac{1}{2}\, X_1^2 + \frac{1}{2}\, z_2^2 + \frac{1}{2}\, X_2^2 \tag{3.83}$$

La dérivée de la fonction de Lyapunov est:

$$\dot{V}(z_1, X_1, z_2, X_2) = \dot{V}(z_1, X_1, z_2) + \dot{X}_2\, X_2 \tag{3.84}$$

Comme : $\{\, \dot{V}(z_1, X_1, z_2) = -\, z_1^2 -\, X_1^2 -\, z_2^2 \,\}$ (voir équation (3.81)), on en déduit que :

$$\dot{V}(z_1, X_1, z_2, X_2) = -\, z_1^2 -\, X_1^2 -\, z_2^2 + \dot{X}_2\, X_2 \tag{3.85}$$

Avec :

$$\dot{X}_2 = w_2 \left(\frac{w_1 + v_r cos(\varphi_r)}{\sqrt{(z_3^2+1)}} \right) + A1 = \Gamma 1 \tag{3.86}$$

Et :

$$A1 = z_4 \left(\frac{\dot{w}_1 + \dot{v}_r cos(\varphi_r) - v_r \dot{\varphi}_r sin(\varphi_r)}{\sqrt{(z_3^2+1)}} \right) + z_4 \left(\frac{w_1 + v_r cos(\varphi_r)}{(z_3^2+1)\sqrt{(z_3^2+1)}} \right) \dot{z}_3\, z_3 - k \left[\dot{z}_2 \left(\frac{w_1 + v_r cos(\varphi_r)}{\sqrt{(z_3^2+1)}} \right) + z_2 \left(\frac{\dot{w}_1 + \dot{v}_r cos(\varphi_r) - v_r \dot{\varphi}_r sin(\varphi_r)}{\sqrt{(z_3^2+1)}} \right) - z_2 \left(\frac{w_1 + v_r cos(\varphi_r)}{\sqrt{(z_3^2+1)}} \right) \dot{z}_3\, z_3 \right] + \frac{\dot{\varphi}_r}{cos^2(\varphi_r)} \sqrt{(z_3^2+1)} \big(w_1 + v_r cos(\varphi_r) \big) + tan(\varphi_r) \sqrt{(z_3^2+1)}\, \big(\dot{w}_1 + \dot{v}_r cos(\varphi_r) - v_r \dot{\varphi}_r sin(\varphi_r) \big) + tan(\varphi_r) \left(\frac{w_1 + v_r cos(\varphi_r)}{\sqrt{(z_3^2+1)}} \right) \dot{z}_3\, z_3 + \dot{z}_3 \big(w_1 + v_r cos(\varphi_r) \big) + z_3 \big(\dot{w}_1 + \dot{v}_r cos(\varphi_r) - v_r \dot{\varphi}_r sin(\varphi_r) \big) - \dot{v}_r sin(\varphi_r) - v_r \dot{\varphi}_r cos(\varphi_r) - \dot{z}_1 \dot{\theta}_r - z_1 \ddot{\theta}_r + \dot{z}_2$$

Les variables $X_1\, et\, X_2$ sont fonction des variables d'états z_1, z_2, z_3, z_4, il en résulte que la fonction de Lyapunov finale pour le nouveau système est équivalente à la fonction de Lyapunov du système original.

$$V(z_1, X_1, z_2, X_2) = V(z_1, z_2, z_3, z_4) \tag{3.87}$$

Pour que $\dot{V}(z_1, X_1, z_2, X_2)$ (voir l'équation (3.85)) soit strictement négative, on pose :

$$\Gamma 1 = -X_2 \tag{3.88}$$

Il en résulte que $\dot{V}(z_1, X_1, z_2, X_2)$ devient strictement négative :

$$\dot{V}(z_1, X_1, z_2, X_2) = -\, z_1^2 -\, X_1^2 -\, z_2^2 -\, X_2^2 \tag{3.89}$$

On en déduit l'expression de la commande v_2 :

$$v_2 = (-A1 - X_2) \Big/ \left(\frac{w_1 + v_r cos(\varphi_r)}{\sqrt{(z_3^2+1)}} \right) \tag{3.90}$$

Avec : $v_2 = w_2$

On obtient au final deux commandes par retour d'état v_1, v_2 qui vérifient la condition de stabilité pour le système à commander.

La nouvelle représentation d'état du système devient :

$$\begin{cases} \dot{z}_1 = w_1 + z_2\dot{\theta}_r \\ \dot{X}_1 = v_1 + \dot{z}_2\dot{\theta}_r + z_2\ddot{\theta}_r + \dot{z}_1 \\ \dot{z}_2 = v_r\big(h_2\, cos(\varphi_r) - cos(\varphi_r)\big) - z_1\dot{\theta}_r + w_1 h_2 \\ \dot{X}_2 = v_2\left(\dfrac{w_1 + v_r cos(\varphi_r)}{\sqrt{(z_3^2+1)}}\right) + A1 \end{cases} \quad (3.91)$$

Méthode 2 :

Dans cette méthode, les deux premières étapes restent les mêmes, la différence par rapport à la méthode précédente apparaît dans le choix de la variable d'état que l'on fait intervenir dans la nouvelle représentation d'état. Dans l'étape 3, on choisit d'introduire la variable d'état z_3 à la place de z_2.

Etape 3 : stabilisation de z_3

Soit l'expression de la fonction de lyapunov augmentée :

$$V(z_1, X_1, z_3) = \frac{1}{2}\, z_1^2 + \frac{1}{2}\, X_1^2 + \frac{1}{2}\, z_3^2 \quad (3.92)$$

La dérivée de la fonction de Lyapunov est :

$$\dot{V}(z_1, X_1, z_3) = \dot{V}(z_1, X_1) + \dot{z}_3\, z_3 \quad (3.93)$$

Comme : $\{\ \dot{V}(z_1, X_1) = -\, z_1^2 - \, X_1^2\ \}$ (voir équation (3.75)), on en déduit que :

$$\dot{V}(z_1, X_1, z_3) = -\, z_1^2 - \, X_1^2 + \dot{z}_3\, z_3 \quad (3.94)$$

Avec : $\dot{z}_3 = v_r\, z_4 \cos(\varphi_r) - v_r\, k\, z_2 \cos(\varphi_r) + w_1 h_1$, (voir l'équation (3.61)).

Pour que $\dot{V}(z_1, X_1, z_3)$ (voir l'équation (3.94)) soit strictement négative, on pose :

$$v_r\, z_4 \cos(\varphi_r) - v_r\, k\, z_2 \cos(\varphi_r) + w_1 h_1 = -z_3 \quad (3.95)$$

Il en résulte que $\dot{V}(z_1, X_1, z_3)$ devient strictement négative :

$$\dot{V}(z_1, X_1, z_3) = -\, z_1^2 - X_1^2 - \, z_3^2 \quad (3.96)$$

En posant le changement de variable suivant :

$$X_2 = v_r\, z_4 \cos(\varphi_r) - v_r\, k\, z_2 \cos(\varphi_r) + w_1 h_1 - \Phi(\mathrm{z}_3) \;\text{ où }\; \Phi(\mathrm{z}_3) = -\mathrm{z}_3$$

On obtient la nouvelle variable d'état :

$$X_2 = v_r\, z_4 \cos(\varphi_r) - v_r\, k\, z_2 \cos(\varphi_r) + w_1 h_1 + \mathrm{z}_3 \tag{3.97}$$

Etape 4 : stabilisation de X_2

Soit l'expression de la fonction de Lyapunov augmentée :

$$V(z_1, X_1, z_3, X_2) = \frac{1}{2}\, z_1^2 + \frac{1}{2}\, X_1^2 + \frac{1}{2}\, z_3^2 + \frac{1}{2}\, X_2^2 \tag{3.98}$$

La dérivée de la fonction de Lyapunov est :

$$\dot{V}(z_1, X_1, z_3, X_2) = \dot{V}(z_1, X_1, z_3) + \dot{X}_2\, X_2 \tag{3.99}$$

Comme : $\{\, \dot{V}(z_1, X_1, z_3) = -\, z_1^2 -\, X_1^2 -\, z_3^2 \,\}$ (voir équation (3.96)), on en déduit que :

$$\dot{V}(z_1, X_1, z_2, X_2) = -\, z_1^2 -\, X_1^2 -\, z_2^2 + \dot{X}_2\, X_2 \tag{3.100}$$

Avec :

$$\dot{X}_2 = w_2\big(w_1 + v_r cos(\varphi_r)\big) + A2 = \Gamma 2 \tag{3.101}$$

Et :

$$A2 = z_4\big(\dot{w}_1 + \dot{v}_r cos(\varphi_r) - v_r \dot{\varphi}_r sin(\varphi_r)\big) + k\big[\dot{z}_2\big(w_1 + v_r cos(\varphi_r)\big) + z_2\big(\dot{w}_1 + \dot{v}_r cos(\varphi_r) - v_r \dot{\varphi}_r sin(\varphi_r)\big)\big] + \dot{w}_1 tan(\varphi_r)(z_3^2 + 1) + w_1 \frac{\dot{\varphi}_r}{cos^2(\varphi_r)}(z_3^2 + 1) + w_1 tan(\varphi_r)\, 2\, \dot{z}_3 z_3 + \dot{z}_3$$

Pour que $\dot{V}(z_1, X_1, z_3, X_2)$ (voir l'équation (3.100)) soit strictement négative, on pose :

$$\Gamma 2 = -X_2 \tag{3.102}$$

Il en résulte que $\dot{V}(z_1, X_1, z_3, X_2)$ devient strictement négative :

$$\dot{V}(z_1, X_1, z_3, X_2) = -\, z_1^2 -\, X_1^2 -\, z_3^2 -\, X_2^2 \tag{3.103}$$

On en déduit l'expression de la commande v_2 :

$$v_2 = (-A2 - X_2)/\big(w_1 + v_r cos(\varphi_r)\big) \tag{3.104}$$

Avec : $v_2 = w_2$

La nouvelle représentation d'état du système étant :

$$\begin{cases} \dot{z}_1 = w_1 + z_2 \dot{\theta}_r \\ \dot{X}_1 = \; v_1 + \dot{z}_2 \dot{\theta}_r + z_2 \ddot{\theta}_r + \dot{z}_1 \\ \dot{z}_3 = v_r\, z_4 \cos(\varphi_r) - v_r\, k\, z_2 \cos(\varphi_r) + w_1 h_1 \\ \dot{X}_2 = v_2\big(w_1 + v_r cos(\varphi_r)\big) + A2 \end{cases} \tag{3.105}$$

Méthode 3 :

Dans cette méthode, on choisit de stabiliser la variable z_4 dans l'étape 3. La méthode du Backstepping intégral est appliquée aux deux entrées de commandes $w_1 \; et \; w_2$.

Etape 3 : stabilisation de z_4

Soit l'expression de la fonction de Lyapunov augmentée :

$$V(z_1, X_1, z_4) = \frac{1}{2} z_1^2 + \frac{1}{2} X_1^2 + \frac{1}{2} z_4^2 \tag{3.106}$$

La dérivée de la fonction de Lyapunov est :

$$\dot{V}(z_1, X_1, z_4) = V(z_1, X_1) + \dot{z}_4 \, z_4$$

Comme : $\{ \; \dot{V}(z_1, X_1) = - z_1^2 - X_1^2 \; \}$ (voir équation (3.75)), on en déduit que :

$$\dot{V}(z_1, X_1, z_4) = - z_1^2 - X_1^2 + \dot{z}_4 \, z_4 \tag{3.107}$$

Avec : $\dot{z}_4 = w_2$, (voir l'équation (3.61)).

Pour que $\dot{V}(z_1, X_1, z_4)$ (voir l'équation (3.107)) soit strictement négative, on pose :

$$w_2 = -z_4 \tag{3.108}$$

Il en résulte que $\dot{V}(z_1, X_1, z_4)$ devient strictement négative :

$$\dot{V}(z_1, X_1, z_4) = - z_1^2 - X_1^2 - z_4^2 \tag{3.109}$$

En posant le changement de variable suivant :

$$X_2 = w_2 - \Phi(\mathrm{z}_4) \;\; \text{où} \;\; \Phi(\mathrm{z}_4) = -\mathrm{z}_4$$

On obtient la nouvelle variable d'état :

$$X_2 = w_2 + \mathrm{z}_4 \tag{3.110}$$

Etape 4 : stabilisation de X_2

Soit l'expression de la fonction de Lyapunov augmentée :

$$V(z_1, X_1, z_4, X_2) = \frac{1}{2} z_1^2 + \frac{1}{2} X_1^2 + \frac{1}{2} z_4^2 + \frac{1}{2} X_2^2 \tag{3.111}$$

La dérivée de la fonction de Lyapunov est :

$$\dot{V}(z_1, X_1, z_4, X_2) = \dot{z}_1 \, z_1 + \dot{X}_1 \, X_1 + \dot{z}_4 \, z_4 + \dot{X}_2 \, X_2 \tag{3.112}$$

Comme : $\{ \; \dot{V}(z_1, X_1, z_4) = - z_1^2 - X_1^2 - z_4^2 \; \}$ (voir équation (3.107)), on en déduit que :

$$\dot{V}(z_1, X_1, z_4, X_2) = - z_1^2 - X_1^2 - z_4^2 + \dot{X}_2 \, X_2 \tag{3.113}$$

Avec : $\dot{X}_2 = v_2 + w_2$, et $v_2 = \dot{w}_2$

Pour que $\dot{V}(z_1, X_1, z_4, X_2)$ (voir l'équation (3.113)) soit strictement négative, on pose :

$$v_2 + w_2 = -X_2 \tag{3.114}$$

Il en résulte que $\dot{V}(z_1, X_1, z_4, X_2)$ devient strictement négative :

$$\dot{V}(z_1, X_1, z_3, X_2) = -z_1^2 - X_1^2 - z_3^2 - X_2^2 \tag{3.115}$$

On en déduit l'expression de la commande v_2 :

$$v_2 = -2w_2 - \mathrm{z}_4 \tag{3.116}$$

Avec : $v_2 = \dot{w}_2$

La nouvelle représentation d'état du système étant :

$$\begin{cases} \dot{z}_1 = w_1 + z_2\dot{\theta}_r \\ \dot{X}_1 = v_1 + \dot{z}_2\dot{\theta}_r + z_2\ddot{\theta}_r + \dot{z}_1 \\ \dot{z}_4 = w_2 \\ \dot{X}_2 = v_2 + w_2 \end{cases} \tag{3.117}$$

3.5.2 La commande par les Modes glissants

Dans ce paragraphe, on élabore les lois de commande par mode glissant d'ordre deux pour stabiliser le système (3.61). D'abord, on choisit les lois de commutations qui nous permettent d'entrainer le système vers le régime glissant. On montre par la suite, que le système est stable au sens de Lyapunov lorsqu'il atteint le régime glissant. Puis, on élabore différentes lois de commande pour la poursuite de trajectoire, à l'aide d'algorithme glissant d'ordre deux.

3.5.2.1 Surfaces de glissement et stabilité du système

Soit le système (3.61) à commander :

$$\begin{cases} \dot{z}_1 = w_1 + z_2\dot{\theta}_r \\ \dot{z}_2 = v_r\big(h_2\, cos(\varphi_r) - cos(\varphi_r)\big) - z_1\dot{\theta}_r + w_1 h_2 \\ \dot{z}_3 = v_r\, z_4 \cos(\varphi_r) - v_r\, k\, z_2 \cos(\varphi_r) + w_1 h_1 \\ \dot{z}_4 = w_2 \end{cases}$$

Avec :

$$\begin{cases} h_1 = z_4 - k\, z_2 + tan(\varphi_r)(z_3^2 + 1) \\ h_2 = \dfrac{z_4 - k\, z_2 + tan(\varphi_r)(z_3^2 + 1)}{\sqrt{(z_3^2 + 1)}} + z_3 \end{cases}$$

Choix des surfaces de glissement

Les lois de commutation appliquée au système sont décrites par deux surfaces de glissement :

$$s_1 = \left(z_1 + z_2\, h_2 + \frac{h_1 z_3}{k}\right) v_r + w_1 \tag{3.118}$$

$$s_2 = \left(\frac{z_2 + z_3}{\sqrt{(z_3^2 + 1)}}\right) v_r + k z_4 v_r + w_2 \tag{3.119}$$

Démonstration de la stabilité au sens de Lyapunov

La fonction de Lyapunov est une fonction définit positive qui exprime l'erreur quadratique du système. La loi de commande permet au système d'être stable au sens de Lyapunov si la dérivée de la fonction de Lyapunov est strictement négative.

Soit la fonction de Lyapunov données par la forme suivante :

$$V = \frac{1}{2}\left(z_1^2 + z_2^2 + \frac{z_3^2}{k} + z_4^2\right) \tag{3.120}$$

La dérivée de la fonction de Lyapunov s'écrit de la forme suivante :

$$\dot{V} = w_1\left(z_1 + z_2\, h_2 + \frac{h_1 z_3}{k}\right) + \frac{z_2 + z_3}{\sqrt{(z_3^2 + 1)}}\, z_4 \cos(\varphi_r) - k\frac{z_2^2}{\sqrt{(z_3^2 + 1)}}\, cos(\varphi_r)$$
$$+ z_2\left(\sqrt{(z_3^2 + 1)} - 1\right) sin(\varphi_r) + z_4 w_2 \tag{3.121}$$

Quand le système atteint le régime glissant idéal, les surfaces de glissement données par les équations (3.118) et (3.119) sont nulles :

$$s_1 = 0 \Leftrightarrow w_1 = -v_r\left(z_1 + z_2\, h_2 + \frac{h_1 z_3}{k}\right) \tag{3.122}$$

$$s_2 = 0 \Leftrightarrow w_2 = -\left(\frac{z_2 + z_3}{\sqrt{(z_3^2 + 1)}}\right) v_r - k\, z_4\, v_r \tag{3.123}$$

En remplaçant ces deux équations (3.122) et (3.123) dans l'équation (3.121) on obtient :

$$\dot{V} = -v_r\left(z_1 + z_2\, h_2 + \frac{h_1 z_3}{k}\right)^2 - k\, z_4^2 - k\frac{z_2^2}{\sqrt{(z_3^2 + 1)}}\, cos(\varphi_r)$$
$$+ z_2\left(\sqrt{(z_3^2 + 1)} - 1\right) sin(\varphi_r) \tag{3.124}$$

Dans cette expression, le signe du terme suivant $\left[z_2\left(\sqrt{(z_3^2+1)}-1\right)\ sin(\varphi_r)\right]$ est inconnu, s'il est positif et suffisamment grand, $\dot{V}$ peut devenir positif et exprimer l'instabilité du système. Une condition doit être posée, pour que la loi de commande reste stabilisante pour le système.

- **Etude de la stabilité :**

Soit $k > 0$.

Le but de cette étude est de trouver une condition sur k en fonction des variables du système afin de maintenir la stabilité du système face aux perturbations du système.

L'expression étudiée est la suivante :

$$fs = -k\frac{z_2^2}{\sqrt{(z_3^2+1)}}\ cos(\varphi_r) + z_2\left(\sqrt{(z_3^2+1)}-1\right)\ sin(\varphi_r) \qquad (3.125)$$

Soit $k > 0$ et $\varphi_r \in [-\varphi_{max}, \varphi_{max}]$;

avec $\varphi_{max} = 18°$ (Contrainte de braquage maximum du Robucar).

Cas 1 : lorsque $z_2 = 0$:

$fs = 0, on\ en\ déduit\ que\ \dot{V} < 0$. Donc le système est stable quelque soit la valeur de k.

Cas 2 : lorsque $z_2\ et\ sin(\varphi_r)$ sont de signe différent, avec $z_2 \neq 0\ et\ sin(\varphi_r) \neq 0$.

$fs < 0, on\ en\ déduit\ que\ \dot{V} < 0$. Donc le système est stable quelque soit la valeur de k.

Cas 3 : lorsque $z_2\ et\ sin(\varphi_r)$ sont de même signe avec $z_2 \neq 0\ et\ sin(\varphi_r) \neq 0$.

On obtient l'inégalité suivante :

$$k > \left|\frac{\sqrt{(z_3^2+1)}\ \left(\sqrt{(z_3^2+1)}-1\right)\ tan(\varphi_r)}{z_2}\right| \qquad (3.126)$$

Soit l'ensemble $E_{\varepsilon/2} = \left\{z_2 \in R,\ |z_2| \leq \frac{\epsilon}{2}\right\}\ avec\ \epsilon > 0$.

Si $z_2 \in E_{\varepsilon/2}$ on en déduit que $k_M = Max\ (k_{minimum})$ est exprimé par l'inégalité suivante :

$$k_M > \left| \frac{\sqrt{(z_{3\,max}^2 + 1)}\ \left(\sqrt{(z_{3\,max}^2 + 1)} - 1\right)\ tan(\varphi_{r\,max})}{\epsilon/2} \right| \qquad (3.127)$$

$z_{3\,max}$ est la valeur maximum tolérée.

$\varphi_{r\,max}$ le braquage maximum du robot de référence.

(La démonstration de la convergence asymptotique du système est montrée dans l'Annexe 2)

- **Position du problème :**

– Le modèle exprimant l'erreur de poursuite du système à commander possède deux entrées de commandes $w_1 et\ w_2$. Ces dernières apparaissent de façon explicite dans l'expression des surfaces de glissement $s_1 et\ s_2$. Le système est considéré comme ayant un degré relatif nul. L'application d'une commande par mode glissant nécessite au moins que le système soit de degré relatif égal à un.

– Le système réel à commander est le ROBUCAR. Il possède quatre roues motrices, chacune est actionnée par un moteur électrique. L'utilisation d'une commande discontinue comme les modes glissants classiques crée beaucoup de chattering, elle pourrait entrainer la dégradation des moteurs électriques.

- **solutions proposées :**

– Pour rendre le système de degré relatif égal à un, il faut étendre le système en considérant de nouvelles commandes $v_1 et\ v_2$ sur lesquelles la commande discontinue de type mode glissant agit. Ces commandes seront par la suite intégrées pour obtenir des commandes $w_1 et\ w_2$ plus lisses. Les nouvelles commandes $v_1 et\ v_2$ apparaissent de façon explicite dans la dérivée première de la surface de glissement. Le modèle de poursuite (3.61) prend la forme suivante :

$$\begin{cases} \dot{z}_1 = w_1 + z_2\dot{\theta}_r \\ \dot{w}_1 = v_1 \\ \dot{z}_2 = v_r\left(h_2\ cos(\varphi_r) - \ cos(\varphi_r)\right) - z_1\dot{\theta}_r + w_1 h_2 \\ \dot{z}_3 = v_r\, z_4 \cos(\varphi_r) - v_r\, k\, z_2 \cos(\varphi_r) + w_1 h_1 \\ \dot{z}_4 = w_2 \\ \dot{w}_2 = v_2 \end{cases} \qquad (3.128)$$

- Afin de réduire les effets du chattering sur les actionneurs du Robucar, le choix d'une commande par mode glissant d'ordre supérieur est intéressant, et plus particulièrement les modes glissants d'ordre deux. Il existe des algorithmes glissants d'ordre 2 applicables au système dont le degré relatif est égal à un comme celui du système (3.128). Parmi ces algorithmes on cite : le twisting et le super twisting.

- ***Application de l'algorithme du super twisting*** [Tche.,11]***:***

La loi de commutation du système à commander est décrite par deux surfaces de glissements s_1, s_2. Chacune des commandes w_1, w_2 doit stabiliser respectivement les surfaces s_1, s_2 autour de zéro.

L'utilisation de l'algorithme du super-twisting est intéressant, car la commande est robuste, facile à mettre en œuvre et à implémenter.

Les commandes v_1, v_2 sont exprimées comme suit :

$$\begin{cases} v_i(z) = -\lambda\ |s_i|^{\rho} sign(s_i) + u_i(z_i) \\ \dot{u}_i(z) = -W\ sign(s_i) \end{cases} \tag{3.129}$$

Avec : $i = 1,2$

Et:

z_i: les variables d'états du système

s_i : les surfaces de glissement.

- ***Application de l'algorithme du twisting :***

On utilise deux lois de commandes v_1, v_2 pour stabiliser les surfaces de glissement autour de zéro.

$$v_i = \begin{cases} -u & si \quad |u| > U_M > 0 \\ -V_m\ sign(s_i) & si \quad s_i \dot{s}_i \leq 0;\ |u| \leq U_M \\ -V_M\ sign(s_i) & si \quad s_i \dot{s}_i > 0;\ |u| \leq U_M \end{cases} \tag{3.130}$$

$Avec$: $i = 1,2$.

L'utilisation de cette commande nécessite la connaissance des valeurs de $\dot{s}_i$ pour pouvoir connaitre leur signe.

$$\dot{s_1} = \left(\dot{z}_1 + \dot{z}_2\ h_2 + \dot{h}_2\ z_2 + \frac{\dot{z}_3\ h_1}{k} + \frac{\dot{h}_1\ z_3}{k} \right) v_r + v_1 \tag{3.131}$$

$$\dot{s_2} = \left(\frac{(\dot{z}_2 + \dot{z}_3)cos(\varphi_r)}{\sqrt{(z_3^2+1)}} - 2\,\dot{z}_3\,z_3 \frac{(\dot{z}_2 + \dot{z}_3)cos(\varphi_r)}{(z_3^2+1)\sqrt{(z_3^2+1)}} - u_r \frac{(\dot{z}_2 + \dot{z}_3)sin(\varphi_r)}{\sqrt{(z_3^2+1)}} + kp\,w_2 \right) v_r + v_2 \quad (3.132)$$

- ***Application de l'algorithme du twisting associé à une commande équivalente Ueq* [Chet.,08] *:***

L'utilisation du Ueq (voir paragraphe 3.3.1) permet d'ajouter une composante continue à l'expression de la commande ce qui permet de diminuer encore plus le chattering.

L'expression des nouvelles entrées de commandes (v_1', v_2') sont données par les formes suivantes :

$$\begin{cases} v_1' = v_1 + Ueq_{\,1} \\ v_2' = v_2 + Ueq_{\,2} \end{cases} \quad (3.133)$$

Calcul du $Ueq_1\, et\, Ueq_2$:

Quand le système atteint le régime glissant régi par : $s_1 = s_2 = 0\ et\ \dot{s}_1 = \dot{s}_2 = 0$, les commandes équivalentes sont calculées à partir des l'équations (3.131) et (3.132), lorsque:

$$\begin{cases} \dot{s}_1 = 0 \\ \dot{s}_2 = 0 \end{cases}$$

On obtient :

$$Ueq_1 = -\left(\dot{z}_1 + \dot{z}_2\,h_2 + \dot{h}_2\,z_2 + \frac{\dot{z}_3\,h_1}{k} + \frac{\dot{h}_1\,z_3}{k} \right) v_r \quad (3.134)$$

$$Ueq_2 = -\left(\frac{(\dot{z}_2 + \dot{z}_3)cos(\varphi_r)}{\sqrt{(z_3^2+1)}} - 2\,\dot{z}_3\,z_3 \frac{(\dot{z}_2 + \dot{z}_3)cos(\varphi_r)}{(z_3^2+1)\sqrt{(z_3^2+1)}} - u_r \frac{(\dot{z}_2 + \dot{z}_3)sin(\varphi_r)}{\sqrt{(z_3^2+1)}} + kp\,w_2 \right) v_r \quad (3.135)$$

Les expressions de Ueq_1 et Ueq_2 nous permettent de déduire les expressions de v_1' et v_2'.

3.5.3 La commande hybride (Mode Glissant-Backstepping)

Cette méthode permet d'associer deux commandes : le Backstepping et les modes glissants classiques [Hong.,02]. Elle permet d'élaborer plus facilement une commande stable au sens de Lyapunov et ce, grâce à la propriété de la commande par Backstepping. De plus, la

commande élaborée possède la propriété de robustesse liée au mode glissant classique. Ce qui permet d'éviter le chattering tout en ayant une commande stable et robuste.

On introduit les modes glissants dans les étapes 2 et 4 vues dans l'élaboration de la commande par Backstepping (Voir paragraphe 3.5.1).

Dans l'étape 2, la nouvelle variable d'état X_1 (voir l'équation (3.69)) est assimilée à une surface de glissement S_1, on obtient :

$$S_1 = w_1 + z_2\dot{\theta}_r + \mathrm{z}_1 \tag{3.136}$$

La dérivé de la surface de glissement prend la forme suivante :

$$\dot{S}_1 = \dot{w}_1 + \dot{z}_2\dot{\theta}_r + z_2\ddot{\theta}_r + \dot{z}_1 \tag{3.137}$$

On pose :

$$\dot{S}_1 = -q_1\, sign(S_1) - r_1\, S_1 \tag{3.138}$$

Avec : q_1, r_1 : les gains de la commande par mode glissant d'ordre un

En associant les deux équations (3.137) et (3.138), on en déduit l'expression de la commande v_1 :

$$v_1 = -\dot{z}_2\dot{\theta}_r - z_2\ddot{\theta}_r - z_1 - q_1\, sign(S_1) - r_1\, S_1 \tag{3.139}$$

Dans l'étape 4, la nouvelle variable d'état X_2 est assimilée à une surface de glissement S_2, on obtient :

Méthode 1 :

Dans cette méthode les résultats sont obtenus en choisissant z_2 comme variable d'état (voir méthode 1 backstepping, page 52).

$$S_2 = w_2 h_2 + v_r\big(h_2 cos(\varphi_r) - sin(\varphi_r)\big) - z_1\dot{\theta}_r + \mathrm{z}_2 \tag{3.140}$$

La dérivée de la surface de glissement est : $\dot{S}_2 = \Gamma 1$ (voir l'équation (3.86)).

On pose :

$$\dot{S}_2 = -q_2\, sign(S_2) - r_2\, S_2 \tag{3.141}$$

Avec : q_2, r_2 : les gains de la commande par mode glissant d'ordre un

En associant les deux équations (3.86) et (3.141), on en déduit l'expression de la commande v_2 :

$$v_2 = (A1 - q_2\ sign(S_2) - r_2\ S_2) \Big/ \left(\frac{w_1 + v_r cos(\varphi_r)}{\sqrt{(z_3^2 + 1)}}\right) \quad (3.142)$$

Méthode 2 :

Dans cette méthode les résultats sont obtenus en choisissant z_3 comme variable d'état (voir méthode 2 backstepping, page 54).

$$S_2 = v_r\ z_4 \cos(\varphi_r) - v_r\ k\ z_2 \cos(\varphi_r) + w_1 h_1 + z_3 \quad (3.143)$$

La dérivée de la surface de glissement est : $\dot{S}_2 =$ Γ2 (voir l'équation (3.101)).

On pose :

$$\dot{S}_2 = -q_2\ sign(S_2) - r_2\ S_2 \quad (3.144)$$

En associant les deux équations (3.101) et (3.144), on en déduit l'expression de la commande v_2 :

$$v_2 = (A2 - q_2\ sign(S_2) - r_2\ S_2)/\big(w_1 + v_r cos(\varphi_r)\big) \quad (3.145)$$

Méthode 3 :

Dans cette méthode les résultats sont obtenus en choisissant z_3 comme variable d'état (voir méthode 2 backstepping, page 55).

$$S_2 = w_2 + z_4 \quad (3.146)$$

Et :

$$\dot{S}_2 = v_2 + w_2 \quad (3.147)$$

On pose :

$$\dot{S}_2 = -q_2\ sign(S_2) - r_2\ S_2 \quad (3.148)$$

En associant les deux équations (3.147) et (3.148), on en déduit l'expression de la commande v_2 :

$$v_2 = -w_2 - q_2\ sign(S_2) - r_2\ S_2 \quad (3.149)$$

- ***Etude de la stabilité du système au sens de Lyapunov***

Calcul de la fonction candidate de lyapunov :

$$V_j(z_1, S_1, z_i, S_2) = V_j(z_1, z_2, z_3, z_4) = \frac{1}{2}\left(z_1^2 + S_1^2 + m_j^2 + S_2^2\right) \quad (3.150)$$

Avec $j = 1,2,3$

La fonction V_j représente la fonction de lyapunov pour chacune des trois méthodes utilisées plus haut.

La variable m_i représente les variables d'états z_i utilisées dans chacune des trois méthodes exposées ci-dessus pour synthétiser une loi de commande hybride, de sorte à avoir :

$$m_1 = z_2, m_2 = z_3, m_3 = z_4.$$

$V_j(z_1, S_1, z_i, S_2)$ est une fonction définie positive.

Montrons $\dot{V}_j(z_1, S_1, z_i, S_2)$ est une fonction strictement négative tel que:

$$\dot{V}_j(z_1, S_1, z_i, S_2) = z_1\dot{z}_1 + S_1\dot{S}_1 + z_i\dot{z}_i + S_2\dot{S}_2 \qquad (3.151)$$

Dans le paragraphe (3.2), on choisit les variables $\dot{z}_1$ et $\dot{z}_i$ comme suit :

$$\dot{z}_1 = -z_1 \text{ et } \dot{z}_i = -z_i.$$

On choisit les variables $\dot{S}_1$ et $\dot{S}_2$ comme suit (voir les équation (3.138), (3.141)):

$$\dot{S}_1 = -q_1\, sign(S_1) - r_1\, S_1 \quad \text{et} \quad \dot{S}_2 = -q_2\, sign(S_2) - r_2\, S_2$$

L'expression de $\dot{V}_j$ devient :

$$\dot{V}_j(z_1, S_1, z_i, S_2) = -z_1^2 - q_1 S_1\, sign(S_1) - r_1 S_1^2 - z_i^2 - q_2\, S_2 sign(S_2) - r_2 S_2^2 \qquad (3.152)$$

Donc :

$\dot{V}_j(z_1, S_1, z_i, S_2) \leq 0$, la commande hybride permet d'avoir un système stable au sens de Lyapunov.

3.6 Conclusion :

Dans ce chapitre, nous avons élaboré et mis en forme un modèle d'erreur de poursuite qui nous a permis d'élaborer des lois de commande par retour d'état de type mode glissant et backstepping. Ces commandes sont robustes et assurent la stabilité du système au sens de Lyapunov. Elles sont dédiées à la commande en poursuite de trajectoire du robot mobile de type voiture en double braquage. En premier lieu, on a déterminé des commandes stabilisantes par Backstepping. L'élaboration des lois de commande a été réalisée par trois méthodes différentes qui dépendent du choix des variables d'états à stabiliser et qui permettent de

définir les expressions finales des commandes. Un nouveau changement de variable des états du système est opéré, donnant pour chaque méthode un nouveau modèle de poursuite équivalent au premier élaboré dans le paragraphe (3.4). Le principe du Backstepping intégrateur a été utilisé partiellement dans les deux premières méthodes, tandis que dans la troisième méthode, il a été appliqué totalement. En second lieu, on a exploité les modes glissants d'ordre deux pour élaborer une commande robuste et discontinue. Ainsi, La loi de commande a été synthétisée de plusieurs manières en exploitant les différents algorithmes par modes glissants d'ordre deux, comme le twisting et le super-twisting. La stabilité du système au sens de Lyapunov a été démontrée et une condition de stabilité a été déduite. Enfin, une méthode hybride basée sur le backstepping et les modes glissants classiques a été utilisée pour élaborer une loi de commande en poursuite de trajectoire. La robustesse de ces lois de commandes doit être testée par le robot le long d'une trajectoire admissible. Ces commandes sont testées en simulation et expérimentation dans le chapitre suivant.

Chapitre 4 :
Simulation et Expérimentation

4.1 Introduction

Dans ce chapitre on présente les résultats de simulation et d'expérimentation pour certaines commandes élaborées dans le chapitre précédent. La commande est appliquée à une poursuite de trajectoire sur un robot mobile de type voiture en double braquage, le long d'une trajectoire admissible. La plateforme expérimentale sur laquelle nous appliquons ces commandes est le ROBUCAR. Les commandes simulées sont le Backstepping, le super-twisting, le twisting associé à une commande équivalente et la commande hybride (Backstepping et Mode glissant classique). On testera par la suite la robustesse de quelques commandes en introduisant des perturbations (erreur en position de départ, changement de profil de la trajectoire, bruit blanc gaussien). Enfin, on présentera les résultats d'expérimentation des commandes appliquées au Robucar.

4.2 Les simulations des Commandes :

Les tests des performances des différents contrôleurs élaborés sont représentés pour certaines commandes, le long d'une trajectoire en circuit fermé généré dans le chapitre 2. Dans ces simulations l'état du robot réel est représenté par un trait vert et l'état du robot de référence est représenté par un trait rouge.

Les conditions initiales sont données par :

- Le robot de référence : $x_r(0) = 0m\ , y_r(0) = 0m, \theta_r(0) = 0rad\ , \varphi_r(0) = 0rad$
- Le robot réel : $x(0) = 0.2\ m\ , y(0) = 0.3\ m, \theta(0) = 0\ rad\ , \varphi(0) = 0\ rad$

Le robot de référence se déplace avec une vitesse de traction de : $v_r = 0.5\ m/s$.

La période d'échantillonnage est égale à : $T_e = 10\ ms$.

Le choix du paramètre k (Voir l'équation 3.6 du modèle de poursuite) est différent pour chaque commande, ce choix permet d'améliorer la commande en stabilité et en temps de réponse. Dans ce qui suit, les résultats présentés en simulation correspondent au choix des paramètres k ainsi que les coefficients liés à la commande par mode glissant. Cela permet d'obtenir le meilleur résultat possible pour chacune des commandes simulées.

4.2.1 Algorithme de simulation :

Dans ce paragraphe on présente l'organigramme général du programme de simulation qui permet la génération de la trajectoire admissible et la commande en poursuite de trajectoire pour un robot mobile de type voiture. On note qu'à chaque itération les nouvelles valeurs des états (x, y, θ, φ) sont calculées grâce aux variables d'états du modèle de poursuite et des variables de commandes w_1, w_2 .

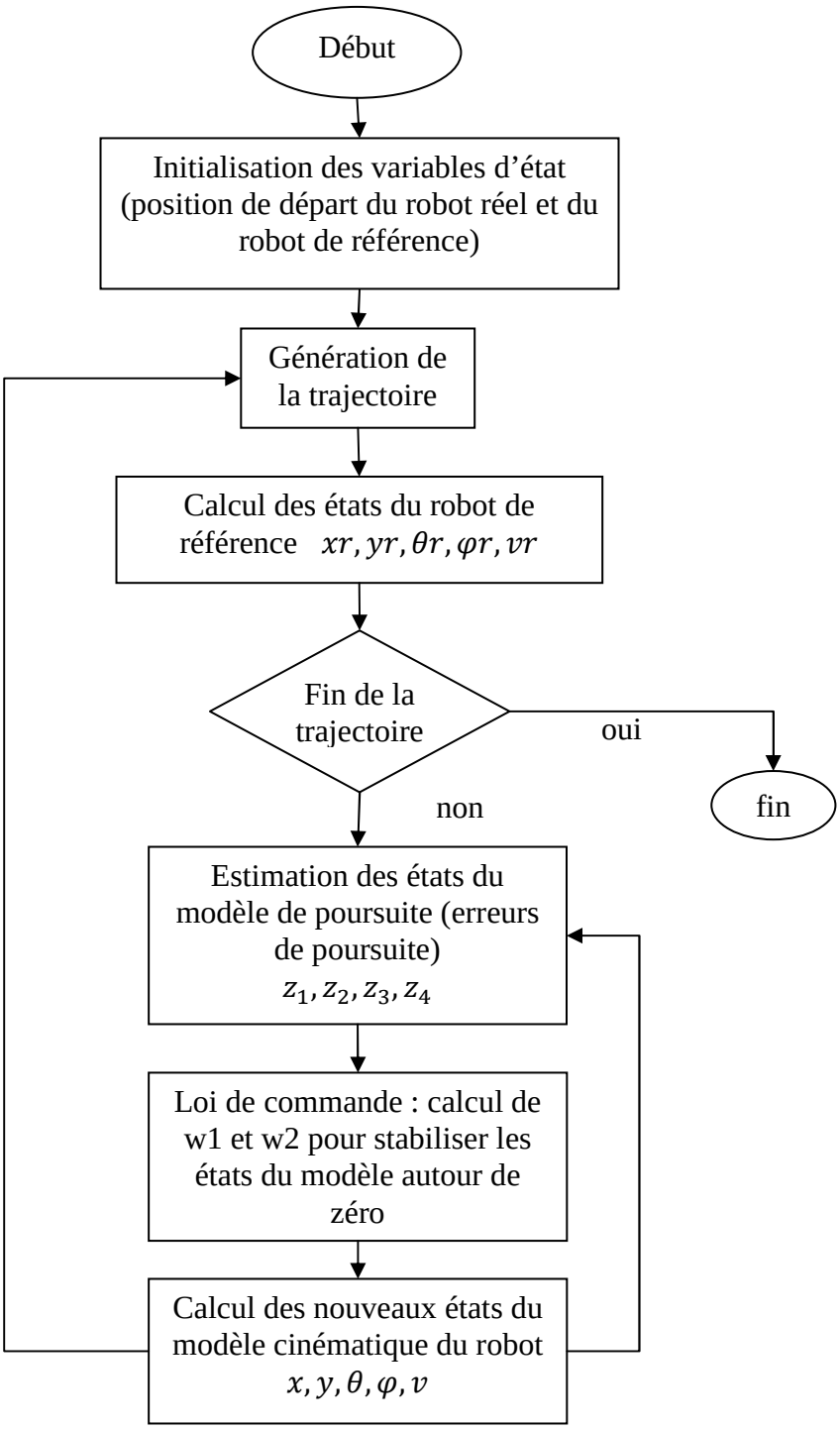

Figure 4.1 : Organigramme de simulation de l'algorithme de poursuite de trajectoire

4.2.2 Résultats de simulation :

Dans ce paragraphe, on teste en simulation les commandes de Backstepping, les commandes par modes glissants d'ordre deux et la commande hybride (Backstepping-mode glissant). On utilise pour cela un modèle de poursuite représentant un robot mobile de type voiture en double braquage, par la suite on teste la robustesse de ces commandes sur le système en ajoutant des perturbations liées au système à commander et à la trajectoire à suivre.

4.2.2.1 La commande par Backstepping :

Dans le chapitre 3, la commande par Backstepping a été synthétisée de trois façons différentes. La différence entre chaque commande est liée au choix des variables d'états qui interviennent dans la stabilisation du système comme on l'a vu dans le paragraphe (3.5.1). On présente ci-dessous la simulation de la poursuite de trajectoire en utilisant la commande élaborée par la méthode 3 (voir paragraphe (3.5.1), page 59). Cette commande ne comporte pas de singularité, contrairement aux deux méthodes précédentes, où les commandes sont inapplicables à $t = 0s$, lorsque la vitesse de translation v_r du robot de référence et la commande en vitesse w_1 sont nulles. L'expression des commandes vues dans la méthode 3 (voir les équations (3.116) et (3.76), chapitre 3) sont plus courtes et moins complexes que celles des méthodes 1 et 2 (voir les équations (3.104) et (3.90)). Le temps de calcul est plus court, ce qui la rend plus intéressante à utiliser dans le cas des systèmes en temps réel. D'autre part les résultats obtenus sont plus intéressants car ils sont plus performants du point de vue du temps de réponse, période transitoire et robustesse.

On présente ci-dessous dans les figures (4.2) et (4.3) l'évolution des variables d'états z_i du modèle de poursuite ainsi que les variables d'erreurs e_i (voir paragraphe (3.4.1), page 47) du robot réel par rapport au robot de référence. Les variables (e_1, e_2, e_3, e_4) représentent respectivement les erreurs sur la position "x", la position "y", l'angle d'orientation et l'angle de braquage. La figure (4.4) représente les profils de vitesse de traction et des angles de braquage total et d'orientation ainsi que les trajectoires parcourues par les deux robots (réel et virtuel). Le choix de la constante $k = 3$ (voir le paragraphe (3.4.2) du chapitre 3) nous permet d'avoir de bons résultats pour cette commande.

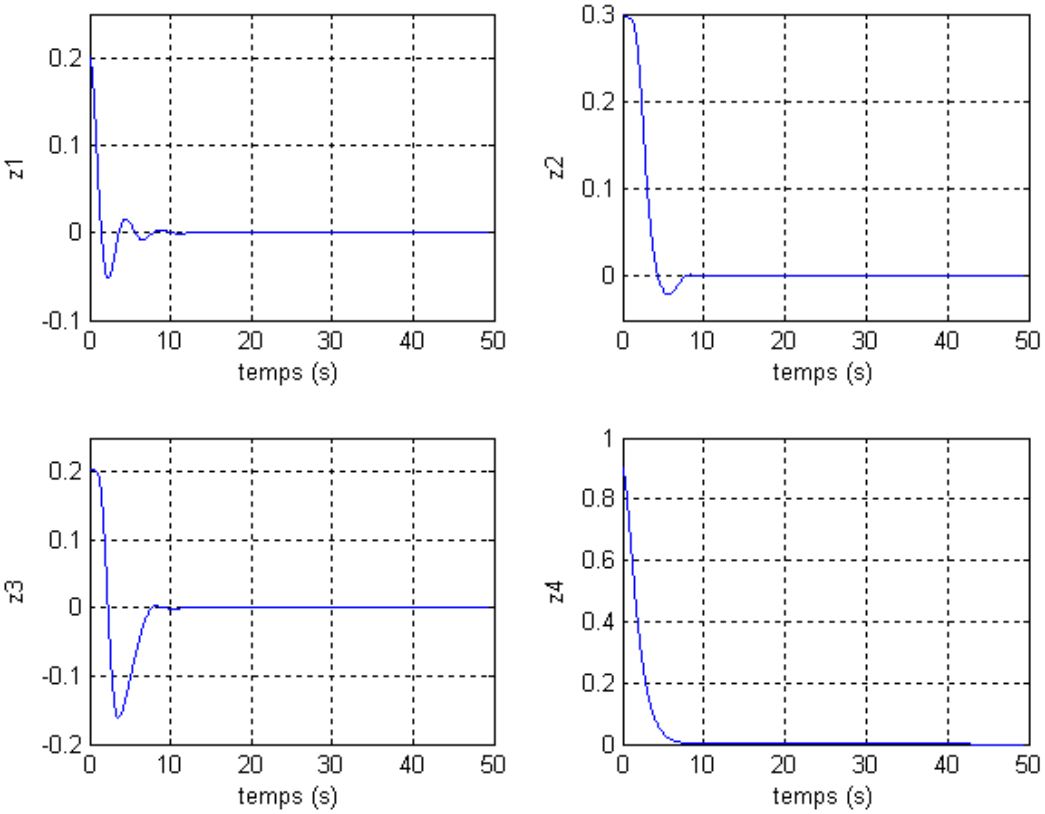

Figure 4.2: Variables d'états du modèle de poursuite

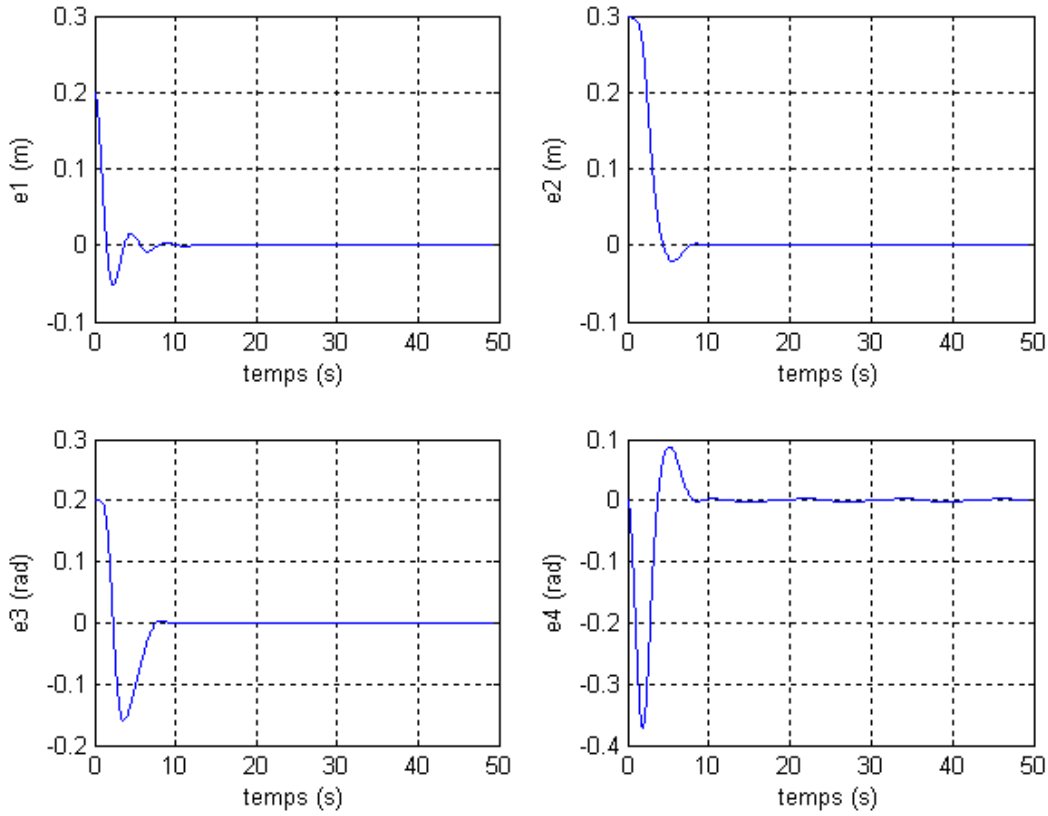

Figure 4.3: Les variables d'erreurs

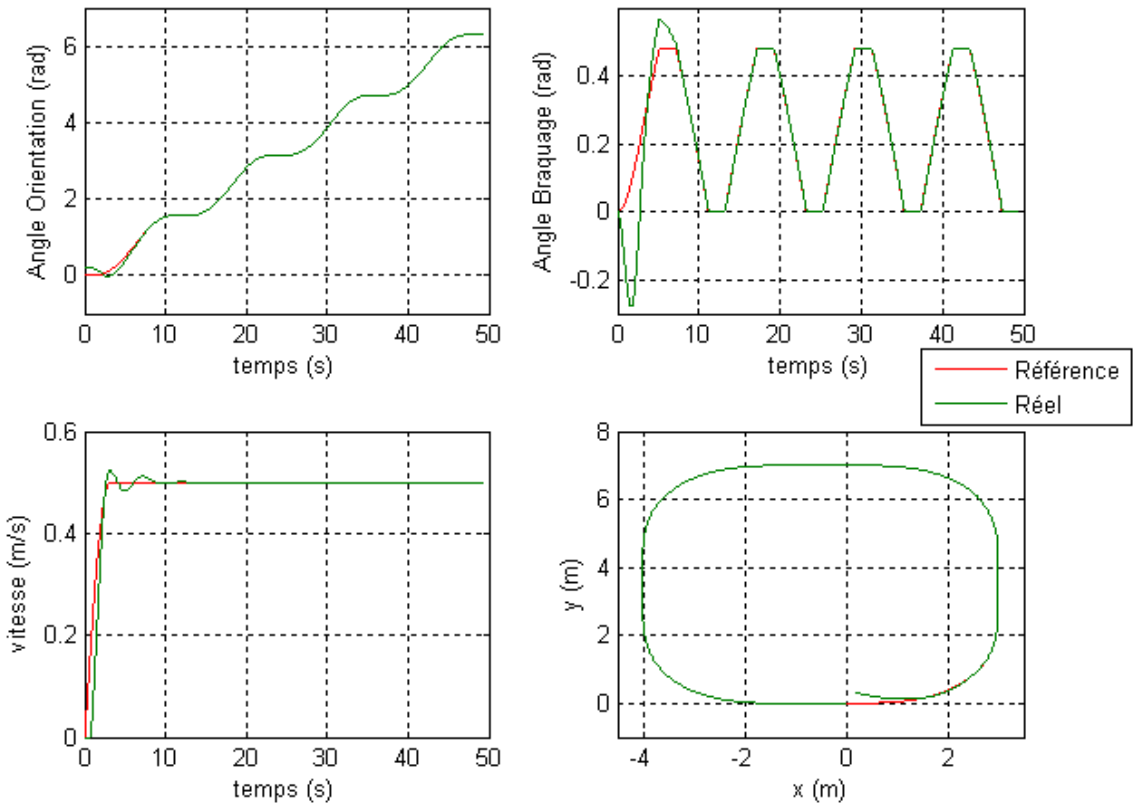

Figure 4.4: La poursuite de trajectoire

Dans la figure (4.2), les variables d'états (z_1, z_2, z_3, z_4) du modèle de poursuite sont stabilisées autour de zéro. Notons que de faibles fluctuations apparaissent durant la période transitoire qui dure moins de $10s$. Dans la figure (4.3), on remarque que l'évolution des erreurs (e_1, e_2, e_3) ressemble à l'évolution des variables (z_1, z_2, z_3), cela est du au changement de variable choisit (voir paragraphe (3.4.1), page 47). La variable e_4 représente l'erreur de l'angle de braquage. L'évolution de e_4 montre la variation de l'angle de braquage du robot réel par rapport à celui du robot de référence nécessaire pour que le robot réel puisse rejoindre la trajectoire désirée, à partir de sa position initiale.

Dans la figure (4.4), on observe la bonne poursuite de trajectoire du système qui est visible au niveau des variables inhérentes au robot. Le profil de vitesse de traction converge vers la valeur désirée de $0.5\ m/s$ au bout de $10\ s$, Les profils des angles d'orientation et de braquage convergent aussi vers leur valeur désirée respective. Bien qu'il y ait un dépassement visible pour l'angle de braquage total par rapport au braquage du robot de référence, ce dépassement reste acceptable tant que l'angle de braquage total ne dépasse pas une valeur limite de

$0.6\ rad$. Les faibles fluctuations qui apparaissent au niveau des variables d'états du modèle de poursuite et des autres variables d'états du modèle cinématique du robot sont engendrées par le choix du paramètre k.

Remarque :
Lorsque le paramètre k augmente ($k >> 100$), des fluctuations apparaissent sur le profil de vitesse. Des dépassements importants apparaissent au niveau de l'angle de braquage durant le premier virage seulement. Ces fluctuations et dépassements peuvent entrainer l'instabilité du système.
Quand k diminue ($k << 1$), le régime transitoire devient plus long. Des fluctuations importantes apparaissent sur des variables d'états z_i ainsi que les profils de vitesse, d'orientation et de braquage. La poursuite de la trajectoire est très mauvaise.

Dans ce qui suit, on présentera les résultats de simulation de la poursuite de trajectoire pour les commandes par mode glissant d'ordre supérieur.

4.2.2.2 Commandes par Mode Glissant d'ordre 2 :

Dans ce paragraphe on teste en simulation les commandes par mode glissant développées dans le paragraphe (3.5.2) du chapitre 3. On présente ci-dessous les résultats pour les algorithmes suivant :

- L'algorithme du Twisting avec la composante continue U_{eq} (commande équivalente) : cette commande donne de meilleurs résultats par rapport à l'algorithme du twisting seul. Ce dernier engendre de faibles oscillations (effet de chattering) au niveau des surfaces de glissement ($s_1\ et\ s_2$), des commandes (w_1, w_2) et du profil de vitesse. L'ajout d'une composante continue U_{eq} permet de réduire considérablement l'effet de chattering. L'inconvénient étant que la période transitoire est plus longue, cela veut dire que le système met plus longtemps à atteindre le régime glissant.
- L'algorithme du super Twisting permet d'avoir une commande robuste est rapide du point de vu de la période transitoire et du temps de réponse. L'expression de la commande est courte, ce qui la rend très intéressante pour la commande des systèmes temps réel.

Cas de l'Algorithme du Twisting avec Ueq :

On présente ci-dessous dans les figures (4.5) et (4.7) l'évolution des variables d'états z_i du modèle de poursuite ainsi que les variables d'erreurs e_i (voir paragraphe (3.4.1), page 47) du robot réel par rapport au robot de référence. La figure (4.8) représente les profils de vitesse de traction et des angles de braquage total et d'orientation ainsi que les trajectoires parcourues par les deux robots (réel et virtuel). La figure (4.6) exprime l'évolution des lois de commutation $s_1 et\ s_2$. Pour avoir une bonne commande en poursuite, les paramètres appartenant à l'expression de la commande sont choisit comme suit :

$$k = 2 \ , \ Um = 0{,}5 \ , \ a_m = 0.035 \ , \ a_M = 1.1,$$

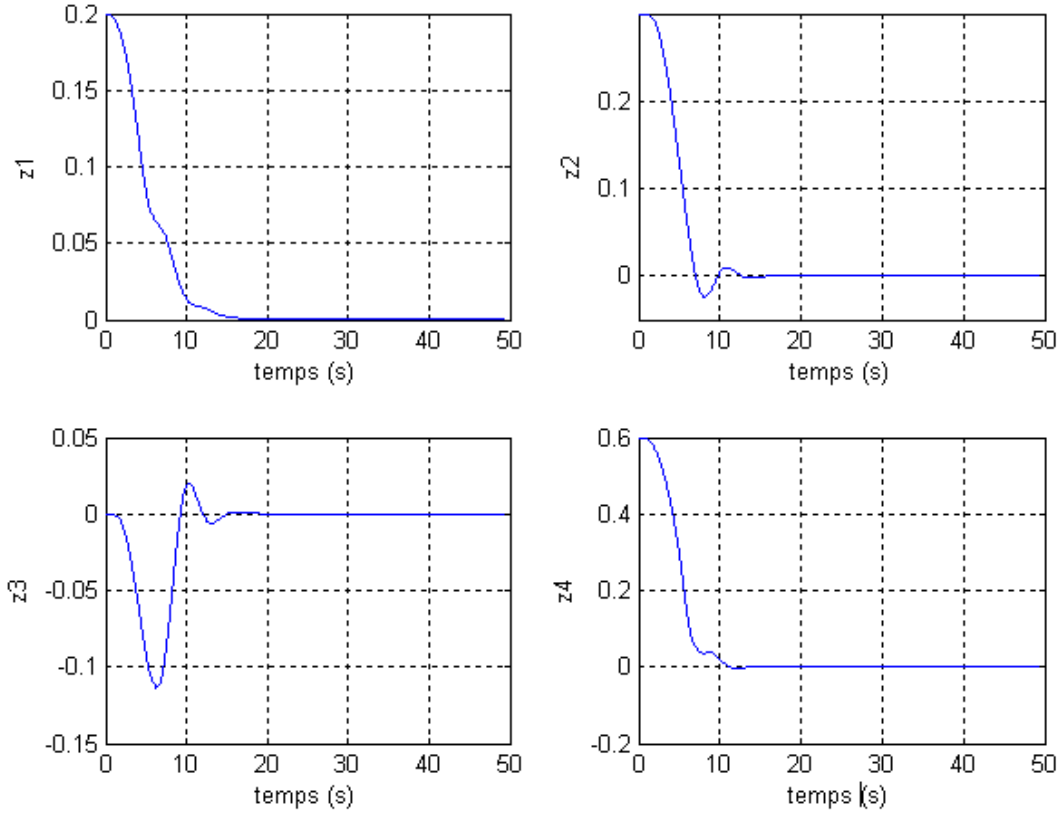

Figure 4.5: Variables d'états du modèle de poursuite

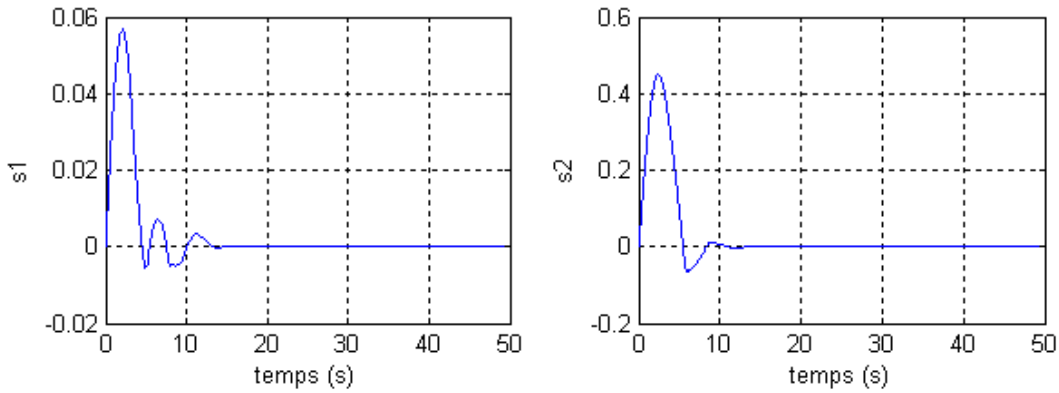

Figure 4.6: les surfaces de glissement

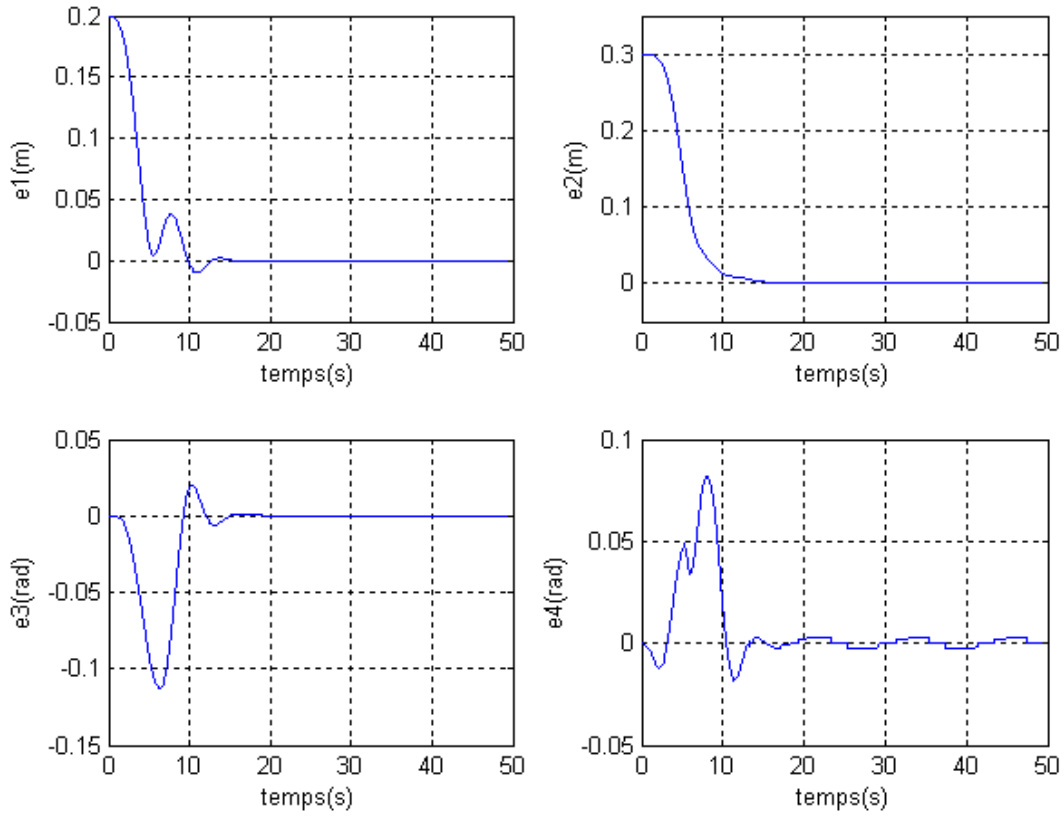

Figure 4.7: Les variables d'erreurs

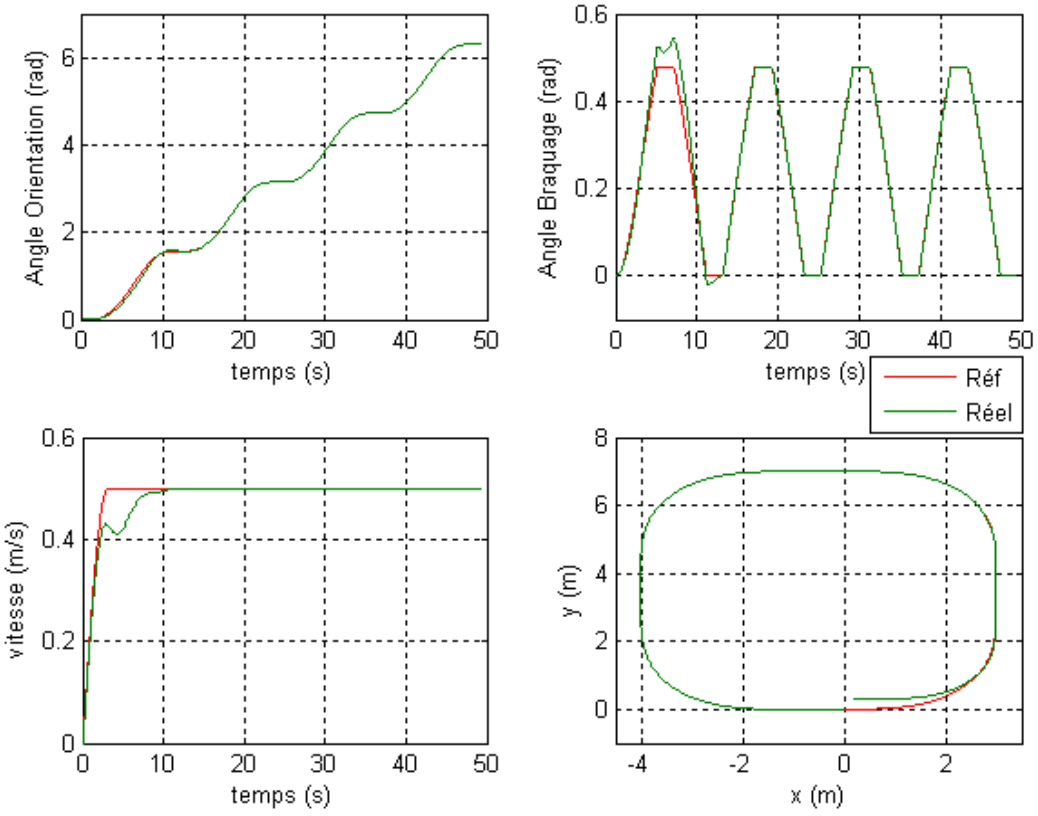

Figure 4.8: La poursuite de trajectoire

Les variables d'états (z_1, z_2, z_3, z_4) du modèle de poursuite (voir figure (4.5)), ainsi que les variables d'erreurs (e_1, e_2, e_3, e_4) (voir figure (4.7)) convergent vers zéro avec l'apparition de très faibles fluctuations durant la période transitoire, l'atténuation des ces fluctuations est du à l'association de la composante continue (Commande équivalente). Le régime glissant est atteint par le système au bout de $15s$, comme le montre la figure (4.6), où les lois de commutation $(s_1 \; et \; s_2)$ s'annulent à la fin de la période transitoire. Cette dernière est plus longue à cause de l'ajout de la composante continue. La poursuite de trajectoire est réalisée correctement (voir figure (4.8)). Les profils des angles d'orientation et de braquage ainsi que le profil de la vitesse de traction se stabilisent autour de leur valeur de référence respective à la fin du premier virage. Notons que le dépassement qui intervient sur l'angle de braquage durant le premier virage est atténué, et ce, grâce à l'ajout de la composante continue.

Remarque :

Quand le paramètre k augmente $(k >> 20)$, la période transitoire devient plus longue, le régime glissant n'est atteint qu'à la fin de cette période. La poursuite de trajectoire est mauvaise surtout pour les variables de braquage et de position (x, y).

Ce qui est très mauvais pour le robot à cause de ses conditions de non holonomie qui imposent un angle de braquage borné.

Lorsque le paramètre k diminue $(k << 1)$, la période transitoire est plus longue. Des oscillations d'amplitudes importantes apparaissent sur les variables d'états (z_1, z_2, z_3, z_4) , les variables de commandes (w_1, w_2) et les variables de glissement (s_1, s_2). Si k est encore plus petit (k<<0.1), cela peut entrainer l'instabilité du système.

Cas de l'Algorithme du Super-twisting

On présente ci-dessous les résultats de simulation. Les figures allant de (4.9) à (4.12) représentent respectivement les variables (z_1, z_2, z_3, z_4) du modèle de poursuite, les variables d'erreurs (e_1, e_2, e_3, e_4), les lois de commutation (s_1, s_2), les trajectoires réalisées, ainsi que les profils: d'Orientation, de braquage et de vitesse de traction inhérents au modèle cinématique du robot mobile.

Pour avoir des résultats optimaux dans la poursuite de trajectoire, les paramètres appartenant à l'expression de la commande sont choisis comme suit : $k = 3$, $w = 20$, $\lambda = 4$

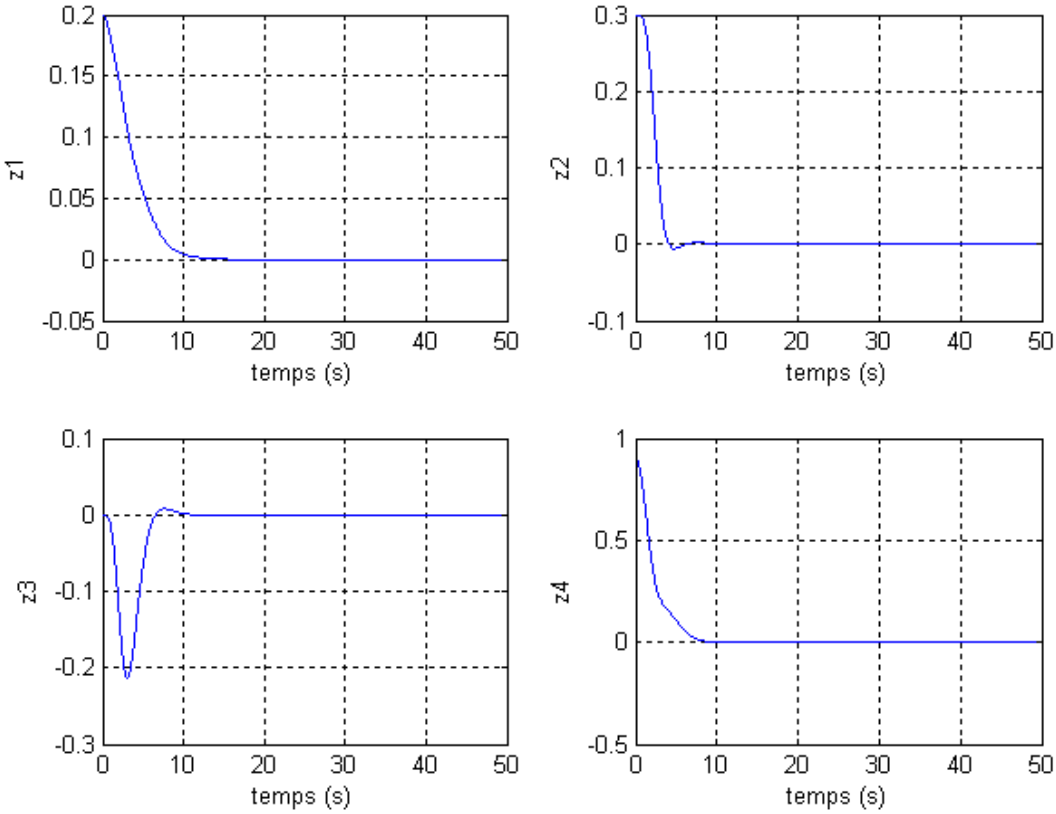

Figure 4.9: Variables d'états du modèle de poursuite

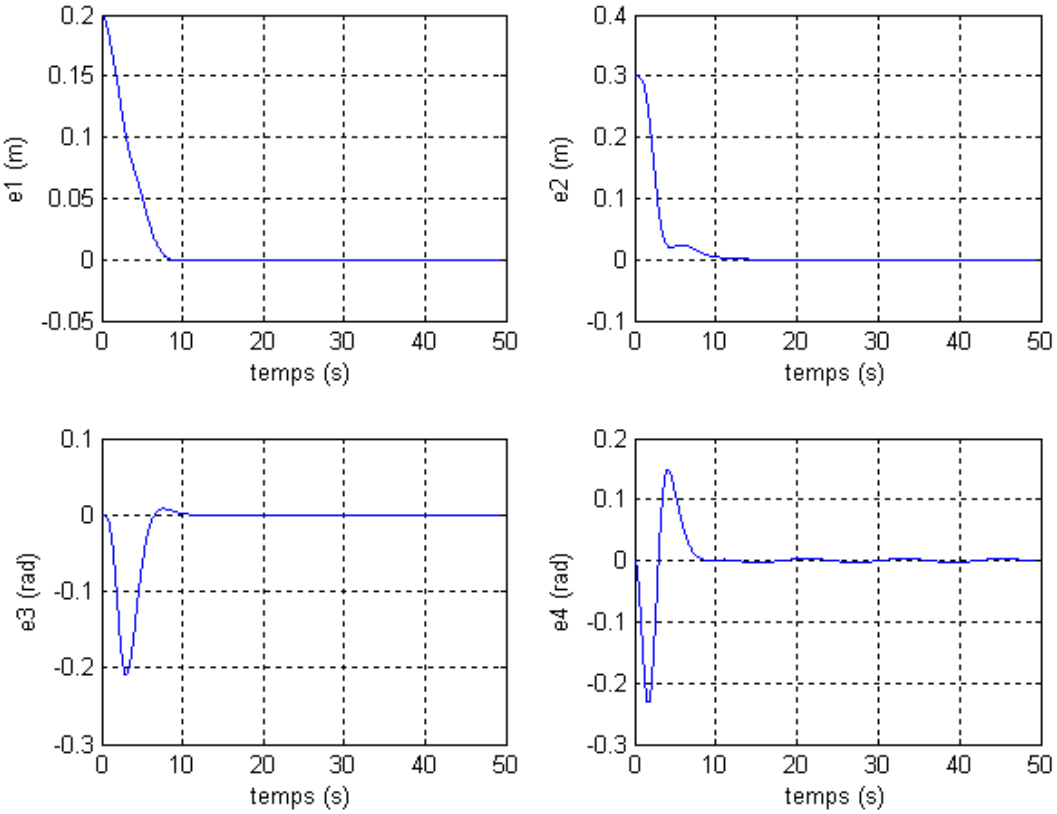

Figure 4.10: Les variables d'erreurs

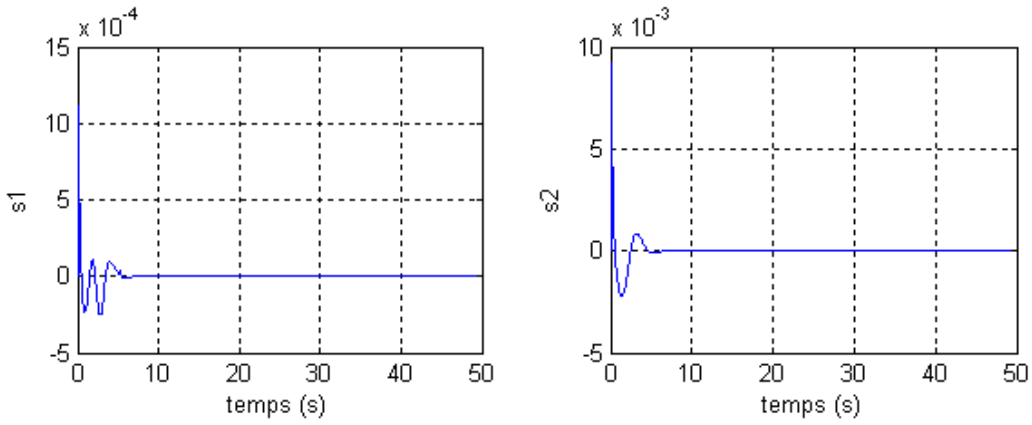

Figure 4.11: les surfaces de glissement

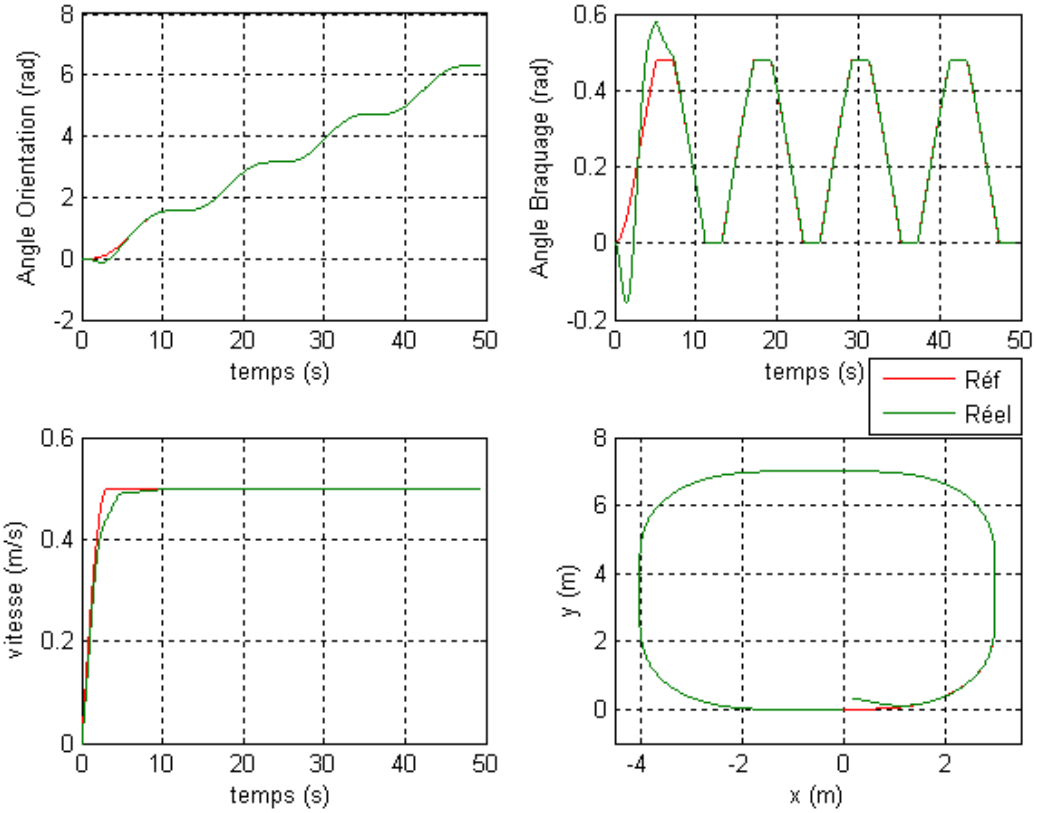

Figure 4.12: La poursuite de trajectoire

La commande a permis de stabiliser les variables d'états (z_1, z_2, z_3, z_4) et les variables d'erreurs (e_1, e_2, e_3, e_4) autour de zéro (voir les figures (4.9) et (4.10)) sans engendrer du chattering ou des fluctuations. L'algorithme du Super Twisting permet de réaliser une bonne poursuite de trajectoire (voir figure (4.12)). Le régime glissant est atteint au bout de $5s$ (durée du régime transitoire) (voir figure (4.11)). Cependant, la rapidité de réponse de la commande a engendré un dépassement important qui apparait sur l'angle de braquage durant le premier virage seulement (voir figure (4.12)).

Remarque :
Plus le paramètre k augmente et plus la réponse devient plus rapide, les variables d'états convergent plus vite vers l'état désiré. Le système atteint le régime glissant en moins de $5s$. La rapidité de la réponse engendre des dépassements supérieurs aux valeurs maximales tolérées, ces dépassements apparaissent au niveau de l'angle de braquage total, ce dernier étant borné à $\pm 0.6\, rad$ à cause des contraintes de non holonomie. Si k est très grand, le système entre dans l'instabilité.
Lorsque k diminue ($k << 1$), la période transitoire est plus longue. Le système met plus longtemps à entrer dans le régime glissant. Des fluctuations et des oscillations importantes apparaissent sur les variables du modèle de poursuite (z_1, z_2, z_3, z_4), les lois de commutations (s_1, s_2) et la vitesse de translation de robot réel. La poursuite de trajectoire est très mauvaise et des dépassements importants apparaissent sur le profil de braquage. Si k est encore plus petit le système entre dans l'instabilité.

La commande présentée dans le paragraphe suivant représente les résultats de simulation d'une commande hybride dont l'intérêt est de diminuer le chattering en associant les principes de la commande à structures variables classiques et du Backstepping

4.2.2.3 La Commande hybride :

Cette commande a été développée dans le paragraphe (3.5.3) du chapitre 3. Elle associe deux types de commande : les Modes glissants et le Backstepping. Cela permet d'avoir la robustesse d'une commande à structure variable tout en diminuant l'effet de chattering (avoir une commande lisse) grâce au Backstepping. L'expression de la commande est synthétisée de trois façons différentes, comme pour le Backstepping (voir le paragraphe (3.5.1)). Les figures présentées ci-dessous montrent les résultats de simulation de la troisième méthode car elle

présente une expression de la commande plus courte et donc plus adaptée au contrôle des systèmes en temps réel. De plus, elle présente de meilleurs résultats que ceux obtenus avec les méthodes 1 et 2.

On présente ci-dessous les résultats de simulation. Les figures allant de (4.13) à (4.16) expriment l'évolution des variables d'états du modèle de poursuite (z_1, z_2, z_3, z_4), les variables d'erreurs (e_1, e_2, e_3, e_4), les variables liées au robot mobile en double braquage $(x, y, Orientation, Braquage, Vitesse)$ et enfin les lois de commutations (s_1, s_2) liées à la loi de commande par mode glissant classique.

Afin d'obtenir une bonne poursuite on choisit la valeur des paramètres intervenant dans les expressions de la commande comme suit : $k = 1.25$, $q = 0.1$, $ks = 1.75$.

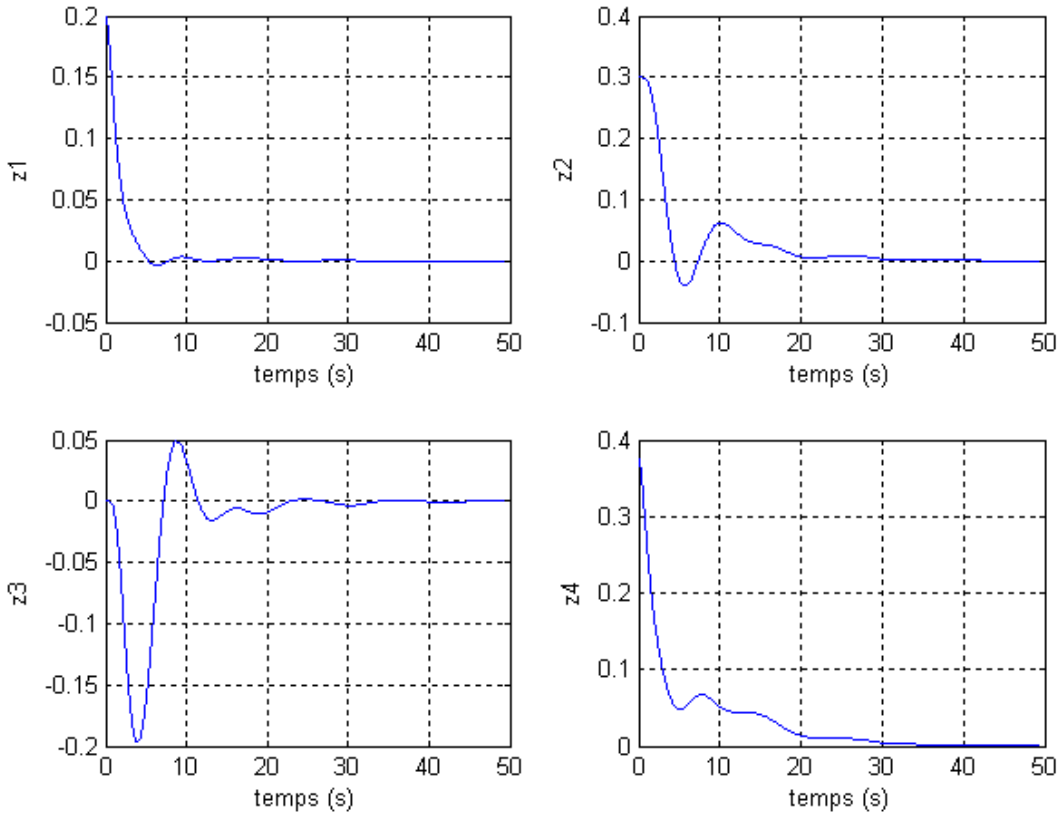

Figure 4.13:Variables d'états du modèle de poursuite

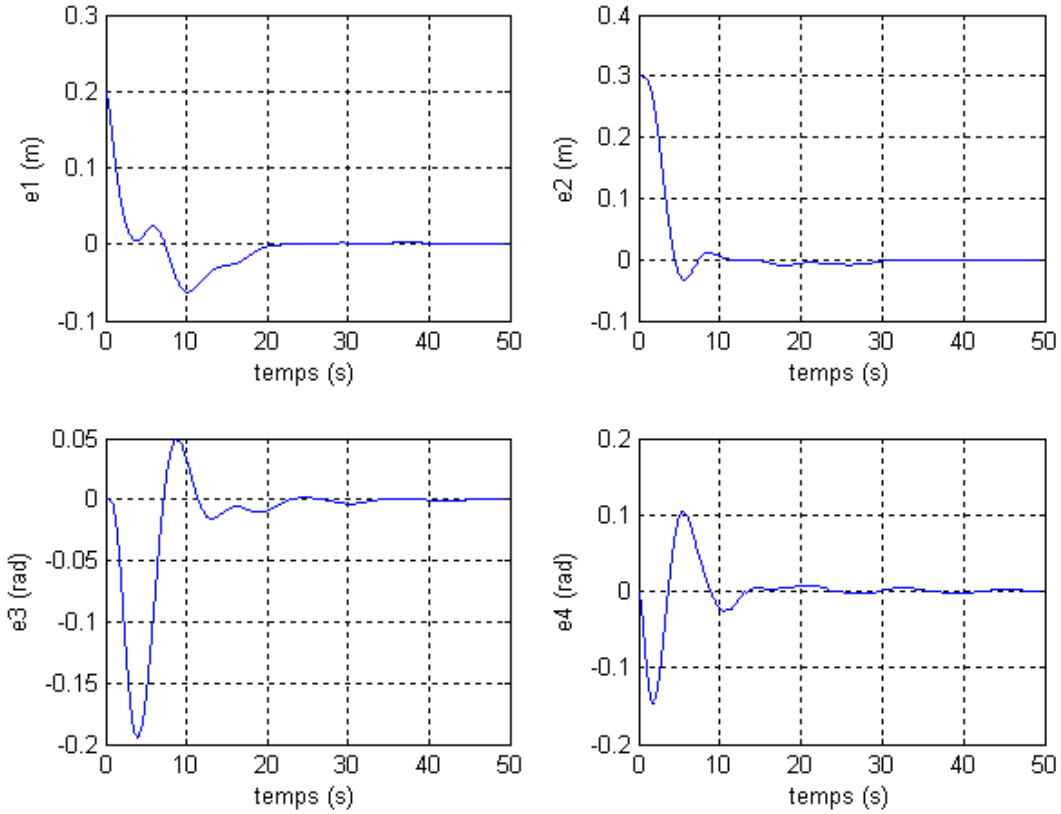

Figure 4.14: Les variables d'erreurs

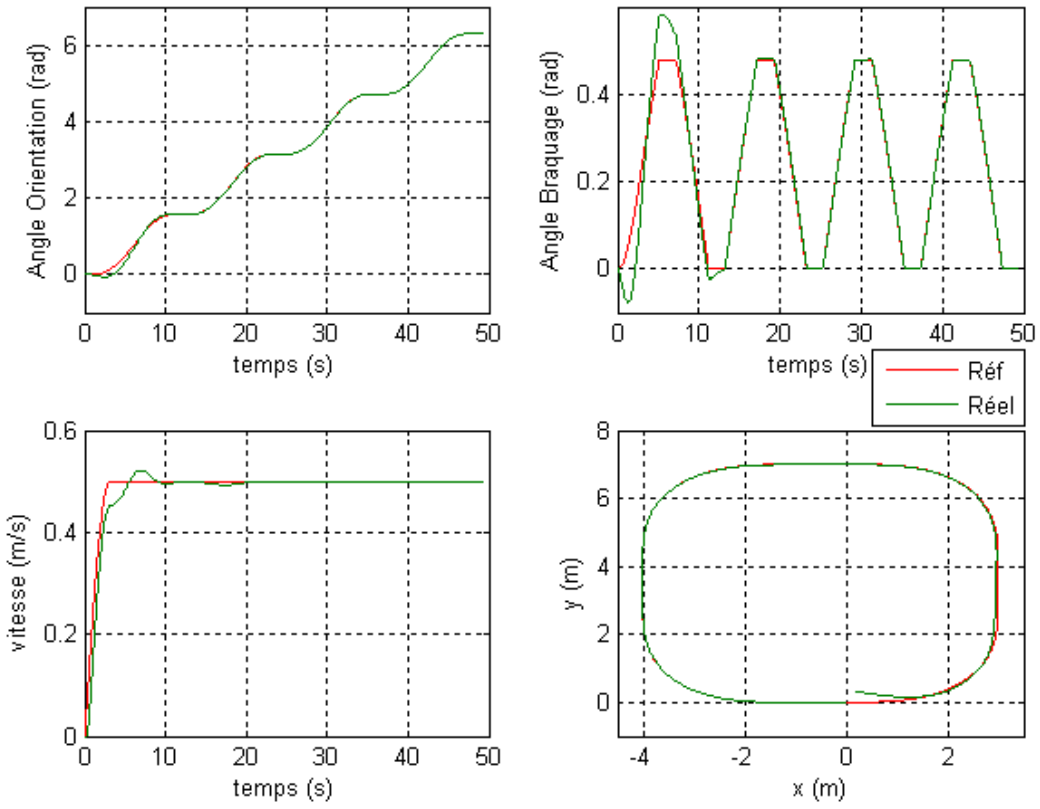

Figure 4.15: La poursuite de trajectoire

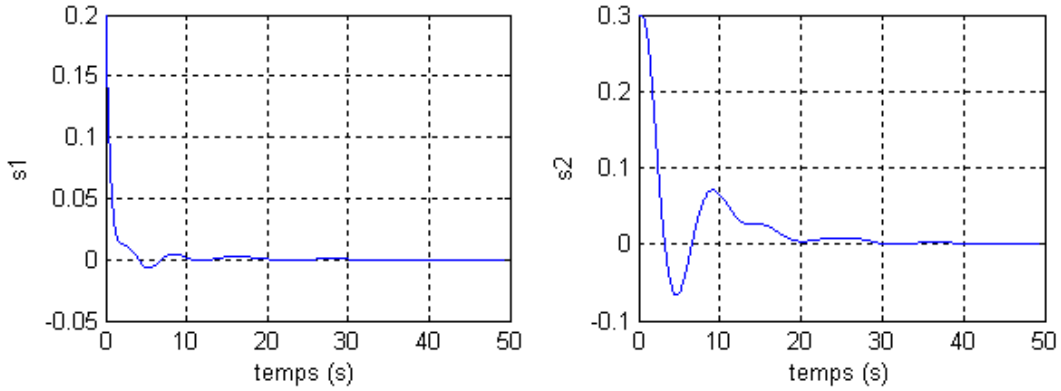

Figure 4.16: Les surfaces de glissement

Les variables d'états (z_1, z_2, z_3, z_4) du modèle de poursuite (voir figure (4.13)), ainsi que les variables d'erreurs (e_1, e_2, e_3, e_4) (voir figure (4.14)) convergent vers zéro avec l'apparition de très faibles fluctuations durant la période transitoire.La poursuite de trajectoire est effectuée correctement (voir la figure (4.15)). L'angle de braquage réel dépasse la valeur de référence durant le premier virage, Il est à la limite de la valeur maximale de $+\,0.6\,rad$ (les contraintes de Nonholonomie). Le temps nécessaire pour atteindre le régime glissant est de $20s$ (voir la figure (4.16)). Cela est du à la composante continu qui constitue la loi de commande (voir paragraphe (3.5.3)). On remarque, que le chattering est inexistant dans les résultats de simulation. Cela est du à l'association de la commande par Backstepping avec la commande par mode glissant classique.

Remarque :

Quand k augmente $(k > 5)$, la période transitoire est plus longue. Le système peut ne jamais atteindre le régime glissant. Cela entraine une très mauvaise poursuite de trajectoire de sorte que le robot réel n'atteint jamais la trajectoire de référence.

Lorsque k diminue $(k << 1)$, la période transitoire augmente. Des oscillations et des fluctuations apparaissent au niveau des commandes (w_1, w_2) et se répercutent sur toutes les variables exprimant le système comme les lois de commutations, les variable d'états du modèle de poursuite ou encore les états du robot mobile. Le système devient très sensible et entre très vite dans l'instabilité.

4.2.2.4 Test de robustesse des commandes :

Les simulations des commandes en poursuite de trajectoire réalisées ci-dessus prennent en considération une perturbation à l'état initial de la poursuite. Le robot réel est écarté de la trajectoire de référence au temps $t = 0\ s$, avec une erreur sur la position (x, y). Dans ce qui suit, on ajoute de nouvelles perturbations au robot réel. La première est une perturbation de type bruit blanc gaussien. La seconde perturbation intervient dans le changement instantané du profil de la trajectoire de référence.

- ***Système avec perturbation de type bruit blanc gaussien***

Cette perturbation introduit des erreurs aléatoires au niveau de chaque variable de l'erreur de poursuite. Elle est caractérisée par une puissance de $-50DB$, d'une moyenne nulle et d'une variance égale à un.

Le système perturbé est de la forme suivante :

$$\begin{cases} \dot{z}_1 = w_1 + z_2\dot{\theta}_r + p \\ \dot{w}_1 = v_1 \\ \dot{z}_2 = v_r\big(h_2\ cos(\varphi_r) - \ cos(\varphi_r)\big) - z_1\dot{\theta}_r + w_1h_2 + p \\ \dot{z}_3 = v_r\, z_4 \cos(\varphi_r) - v_r\, k\, z_2 \cos(\varphi_r) + w_1h_1 + p \\ \dot{z}_4 = w_2 + p \\ \dot{w}_2 = v_2 \end{cases} \tag{4.1}$$

$\boldsymbol{p}$ représente la perturbation de type bruit blanc gaussien.

On présente ci-dessous les figures (4.17) (4.18) (4.19) (4.20) qui représentent respectivement les commandes de : Backtepping, Algorithme du Twisting avec Ueq, Algorithme du Super-Twisting et la commande Hybride.

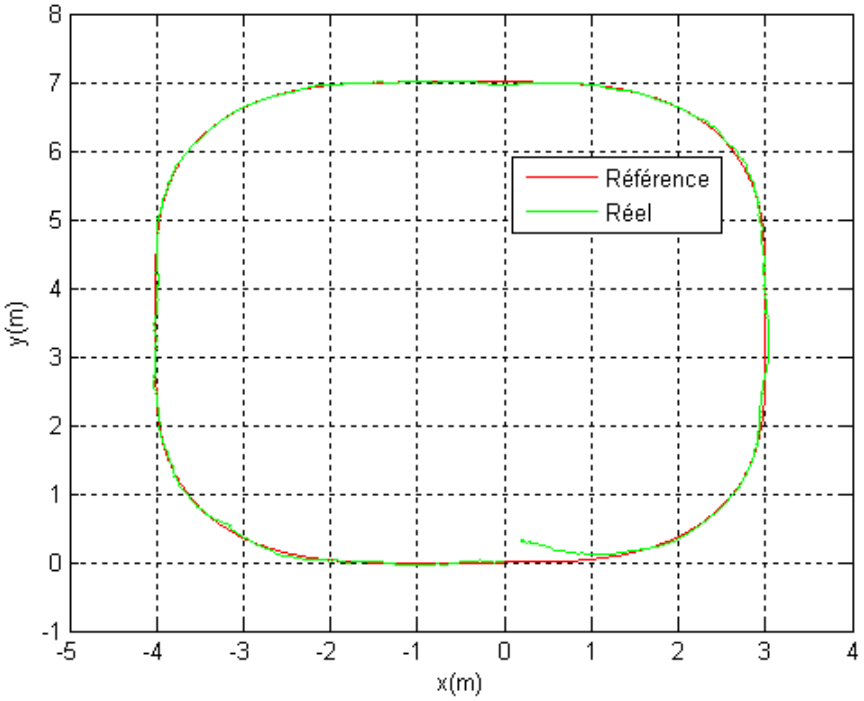

Figure 4.17: Trajectoire réalisée avec la commande du Backstepping

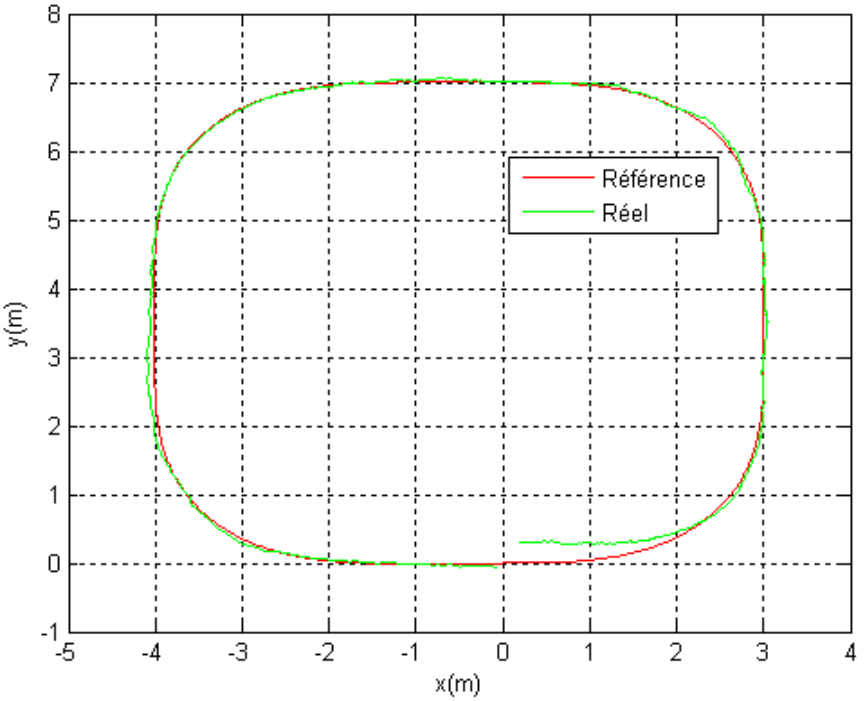

Figure 4.18: Trajectoire réalisée avec l'algorithme du Twisting et Ueq

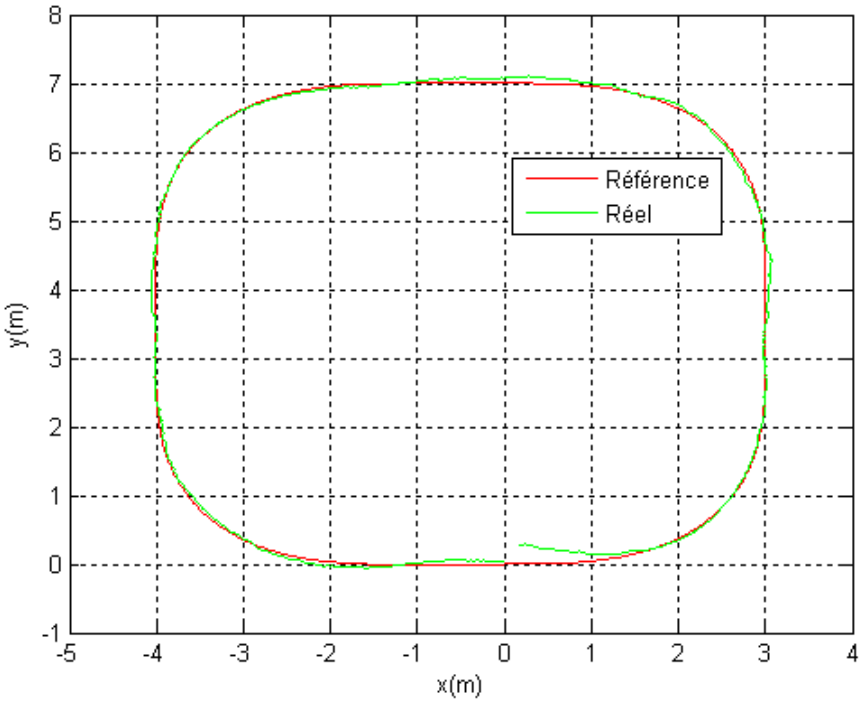

Figure 4.19: Trajectoire réalisée avec l'algorithme du Super-Twisting

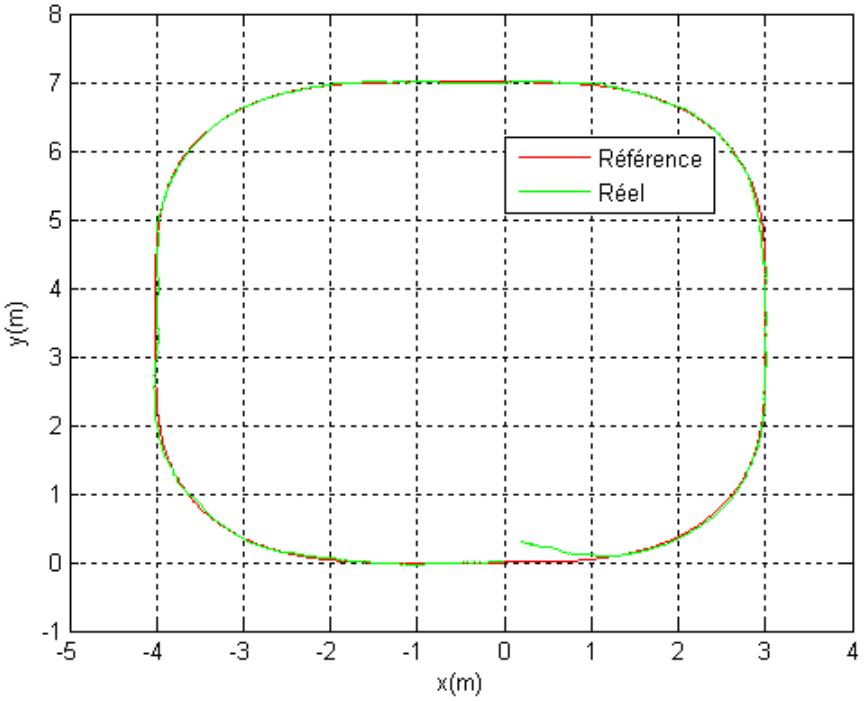

Figure 4.20: Trajectoire réalisée avec la commande Hybride

Malgré l'introduction de perturbation aléatoire de type bruit blanc gaussien dans le modèle de poursuite du système à commander, on remarque que la poursuite de trajectoire est réalisée correctement pour toutes les commandes simulées. Le robot ne rentre pas dans l'instabilité et ne dévie pas de la trajectoire de référence. Cependant les perturbations engendrent des fluctuations au niveau de la vitesse est de l'angle de braquage, ces parasites traduisent une réponse brutale des actionneurs, ce qui peut entrainer dans la pratique la dégradation des actionneurs.

- ***Cas du changement de profil de la trajectoire*** :

Dans le chapitre 2, nous avons présenté les simulations de la génération d'une trajectoire avec un changement brutal du profil de la trajectoire. Ce nouveau profil engendre des modifications au niveau de toutes les variables exprimant l'état du robot virtuel, c'est-à-dire, $(x, y, Orientation, Braquage, Vitesse)$. On présente ci-dessous, dans les figures (4.21) (4.22) (4.23) (4.24), les résultats de simulation de toutes les commandes simulées précédemment. Afin de ne pas saturer le document les résultats présentés ne concernent que les trajectoires réalisées.

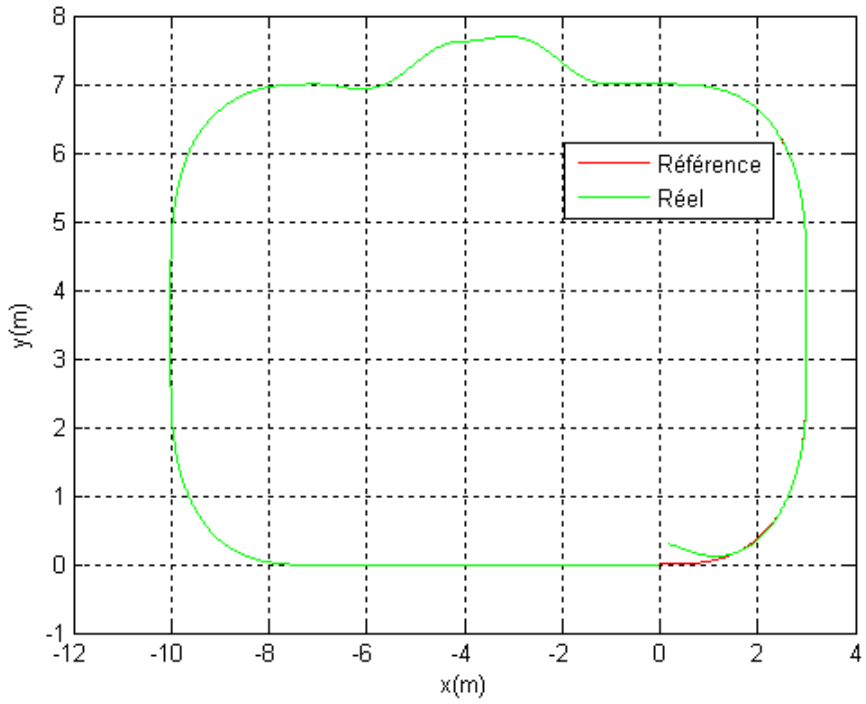

Figure 4.21: Trajectoire réalisée avec la commande par Backstepping

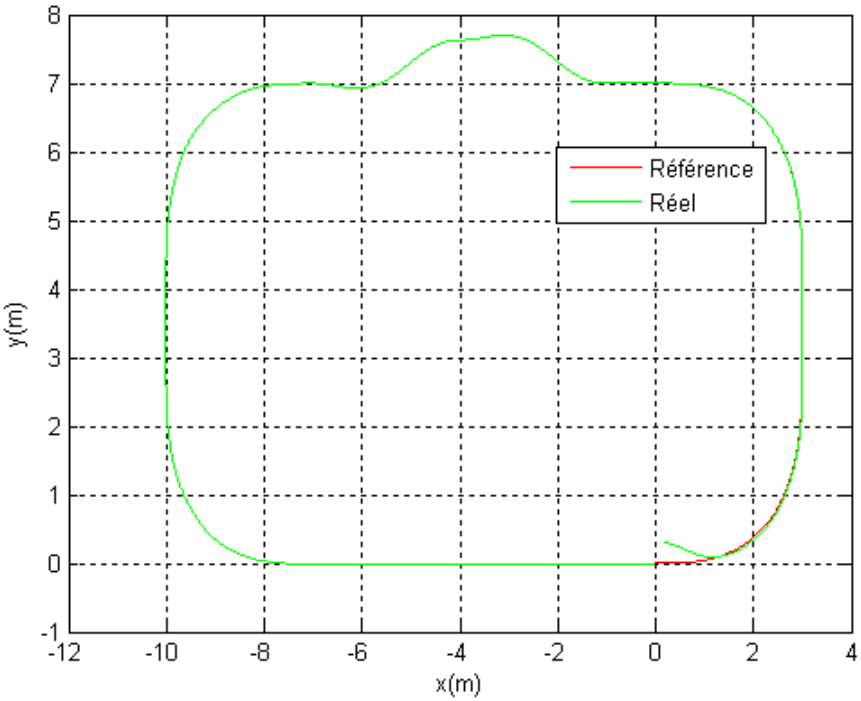

Figure 4.22: Trajectoire réalisée avec l'algorithme du Twisting et Ueq

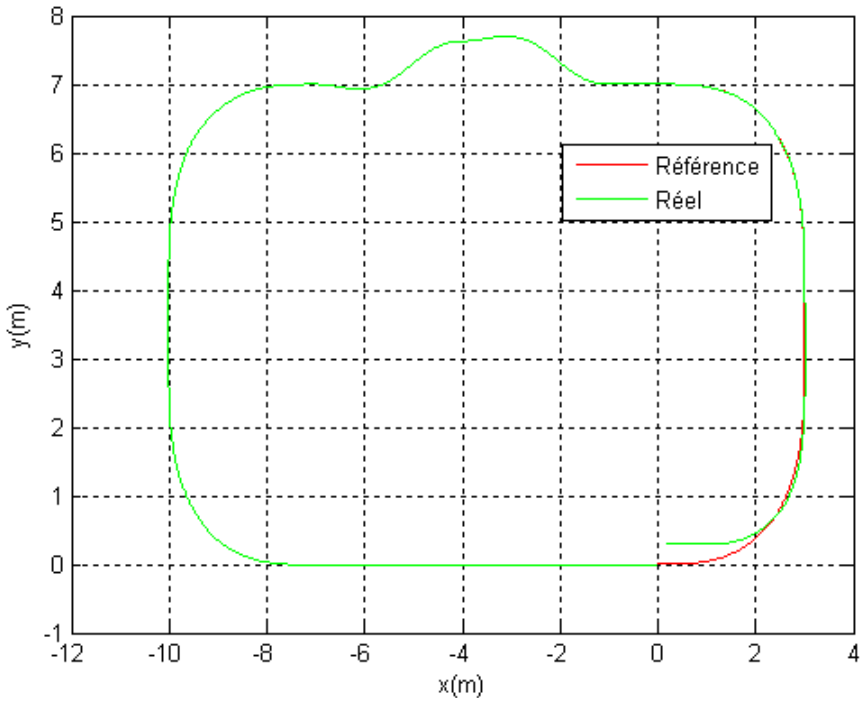

Figure 4.23: Trajectoire réalisée avec l'algorithme du Super-Twisting

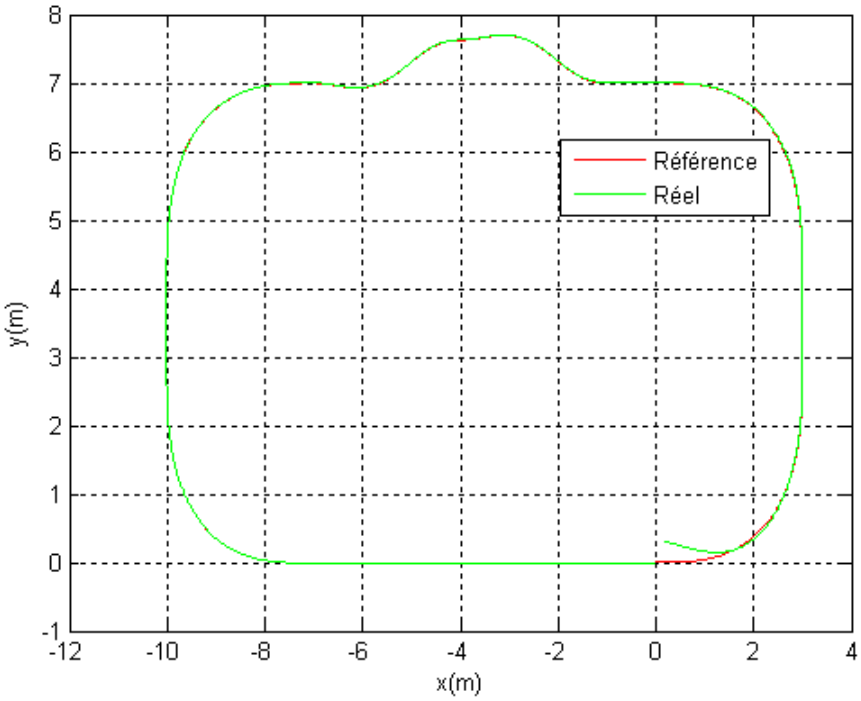

Figure 4.24: Trajectoire réalisée avec la commande par Backstepping

On remarque que toutes les commandes sont robustes face à la perturbation introduite en changeant le profil de la trajectoire après le second virage.

4.2.3 Discussion

La valeur du paramètre *k* est choisie différemment pour chaque commande simulée ci-dessus, de manière à avoir un résultat optimal à chaque fois. Les méthodes de commandes présentées permettent d'avoir une bonne poursuite en général, pour cela le choix du paramètre *k* est crucial pour assurer la stabilité ainsi que pour avoir un bon compromis entre le temps de réponse, la durée de la période transitoire et l'importance des fluctuations durant cette période.

On remarque une différence pour le temps nécessaire à annuler les erreurs de poursuite ainsi que le temps nécessaire pour atteindre le régime glissant (pour les commandes par mode glissant et la commande hybride). Cette différence est en partie due à l'association d'une composante continue à la commande à structure variable, il en résulte une commande plus

lente, et donc, un temps de réponse et un régime transitoire plus long comme le montre le tableau récapitulatif ci-dessous.

Les commandes	MGOS2 Super-Twisting	MGOS2 Twisting - Ueq	Backstepping	Hybride
Régime Transitoire	10 s	15 s	10 s	30 s
Régime Glissant	5 s	15 s	----	20 s

L'association de la commande à base de l'algorithme du Twisting et d'une commande équivalente (voir paragraphe 4.2.2.2) à permis de réduire considérablement le chattering engendré par la commande de l'algorithme du Twisting. D'autre part l'association de la commande par backstepping et de la commande à structure variable classique à permis de réduire le chattering obtenue avec la méthode des modes glissants classiques (voir paragraphe 4.2.2.3). Cependant, la commande par Mode glissant d'ordre 2 « Algorithme du Super-Twisting » (voir paragraphe 4.2.2.2) donne une meilleure poursuite de trajectoire, le régime glissant est atteint très rapidement « 5 seconde », ce qui rend cette commande plus rapide que les autres commandes à structures variables testées. L'effet du chattering n'apparaît pas dans les résultats de simulations. En d'autres termes, les résultats obtenus en simulation, pour l'algorithme du Super-Twisting sont plus satisfaisants.

Afin de valider les résultats de simulation, il faut tester ces commandes sur une plateforme robotisée réelle : cas du Robucar, disponible au CDTA, division robotique et productique.

4.3 Expérimentation des commandes

Dans ce paragraphe on présente les résultats expérimentaux, de la commande en poursuite de trajectoire sur la plateforme robotisée « ROBUCAR ». La commande implémentée est la commande par Mode Glissant d'Ordre 2 « algorithme du super-twisting ». Elle est facile à mettre en œuvre et à implémenter, elle présente de très bons résultats en simulation par rapport aux autres commandes simulées. La simplicité de l'expression de la commande permet une plus grande rapidité de calcul et donc de réponse, ce qui est très intéressant dans les systèmes temps réel comme le ROBUCAR.

4.3.1 Algorithme pour l'implémentation:

Dans cet algorithme, les informations sur la position (x, y, θ, φ) du robot réel sont directement obtenues grâce à des capteurs Odométriques, ces derniers estiment à chaque période d'échantillonnage la vitesse v de translation du robot ainsi que son angle de braquage φ .

Par la suite on calcule les variables d'états du robot grâce au modèle Odométrique suivant :

$$\begin{cases} x_k = x_{k-1} + \delta d \cos\left(\theta_{k-1} - \varphi + \dfrac{\Delta\theta}{2}\right) \\ y_k = y_{k-1} + \delta d \sin\left(\theta_{k-1} - \varphi + \dfrac{\Delta\theta}{2}\right) \\ \theta_k = \theta_{k-1} + \delta d \end{cases} \tag{4.2}$$

Ces variables vont permettre d'estimer les variables d'état du modèle de poursuite pour calculer les variables de commandes w_1, w_2 permettant le contrôle du robot mobile.

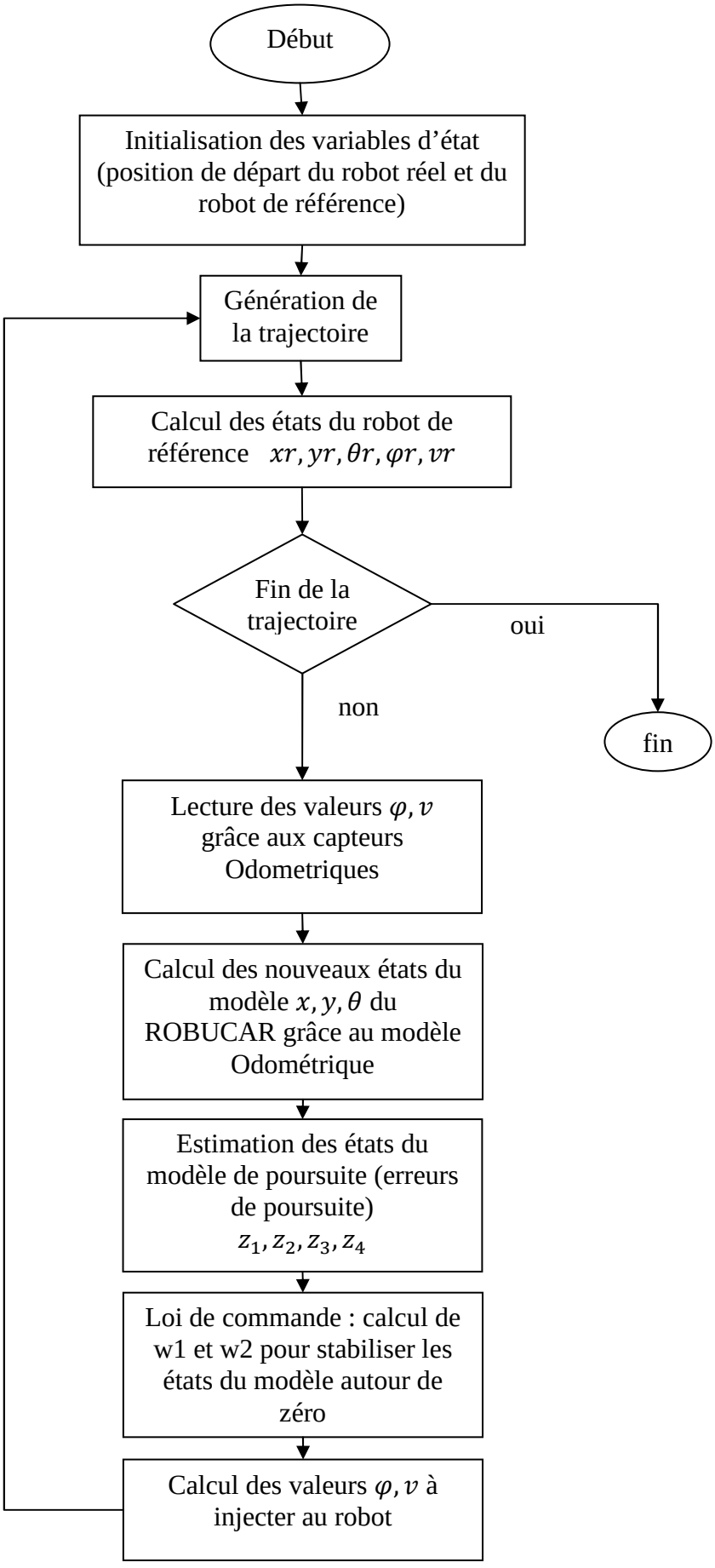

Figure 4.25 : Organigramme de l'algorithme de poursuite de trajectoire

4.3.2 Présentation du ROBUCAR

Le ROBUCAR est un robot de type voiture conçu par Robotsoft (voir figure 4.26). C'est un prototype de voiture électrique dont le châssis ressemble à celui des petites voitures utilisé sur les terrains de golf. Il est caractérisé par :

- 4 moteurs électriques de 1200 Watts.
- 4 roues motrices et directrices.
- Un frein à commande électrique par roue.
- vitesse maximale : 18 Km/h (5m/s).
- autonomie : 2 heures d'utilisation continue.
- capacité d'accueil : 2 personnes avec bagages.
- conduite automatique ou manuelle.

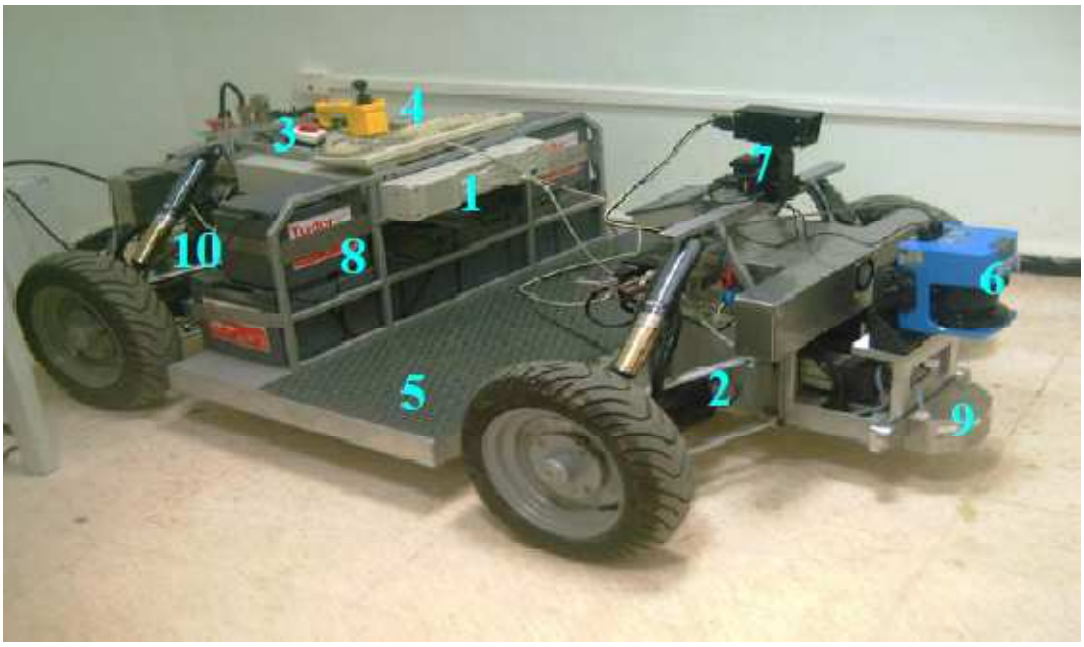

Figure 4.26: Le Robucar

Il est composé des éléments suivants :

1. Pc embarqué
2. Axe avant (2roues motrices, vérin direction, et un µCrtl Mpc555)
3. Arrêt d'urgence.
4. Joystick.
5. Châssis nid d'abeille.
6. LMS Sick200.
7. Caméra CCD.
8. Batteries.
9. Ultrasons
10. Axe arrière (2roues motrices, vérin direction)

Le ROBUCAR possède des capteurs proprioceptifs qui lui permettent de calculer sa position actuelle à chaque instant. Ces capteurs sont appelés aussi des capteurs odométriques. Le codeur incrémental qui se trouve au niveau de chaque roue lui permet de calculer sa vitesse, d'autres capteurs permettent de déterminer l'angle de braquage du robot. A partir de ces deux informations, on peut estimer la position du robot et donc la trajectoire réalisée par ce dernier. Le robot peut être commandé par deux grandeurs : la vitesse de translation et le braquage instantané. Ces informations sont calculées à chaque instant par l'algorithme de commande (PC embarqué), puis elles sont misent dans une mémoire partagée (voir figure 4.27) entre le programme qui gère le bas niveau (gestion des actionneurs et capteurs) et le PC embarqué. Ces grandeurs sont ensuite injectées dans les actionneurs.

Le pilotage du ROBUCAR en temps réel est réalisé grâce à un logiciel: SYNDEX, il gère la partie bas niveau du ROBUCAR ainsi que les tâches temps réel.

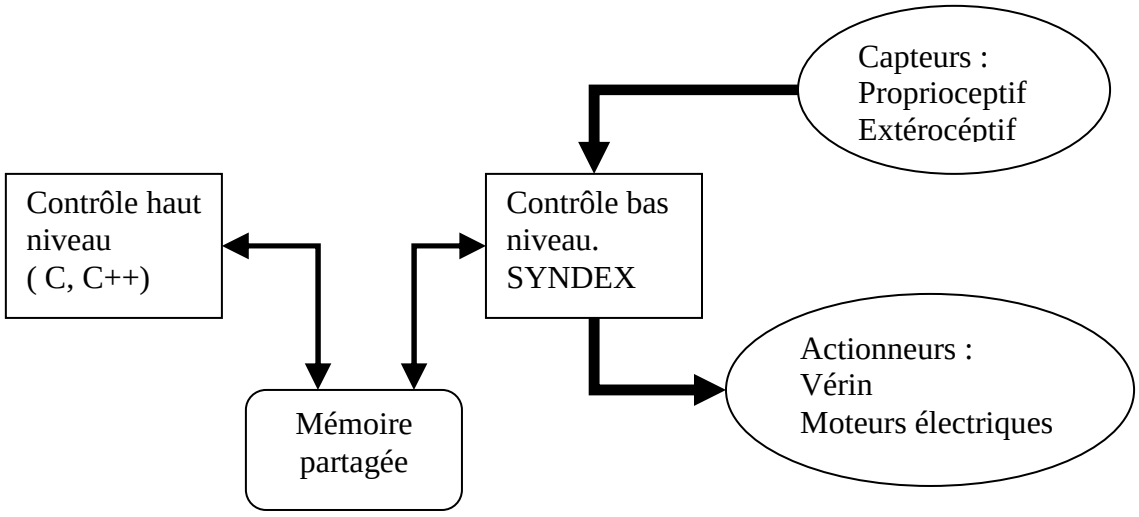

Figure 4.27: Principe de la mémoire partagée

4.3.3 Résultats expérimentaux:

On présente ci-dessous les résultats d'expérimentations d'une commande en poursuite de trajectoire le long d'un arc de cercle, le long d'un circuit fermé à quatre virages. De plus on présente les résultats obtenus lors de l'ajout d'une perturbation au niveau l'angle de braquage ainsi qu'au niveau de la vitesse.

a) Trajectoire en forme d'arc de cercle

Cas de la commande par mode glissant d'ordre 2 (Le super-twisting) :

La figure (4.28) représente les profils de vitesse et des angles de braquage et d'orientation. Ainsi que la trajectoire parcourue par les robots réel et virtuel. La figure (4.29) exprime les erreurs (e_1, e_2, e_3, e_4) des variables cinématiques du robot réel par rapport au robot virtuel.

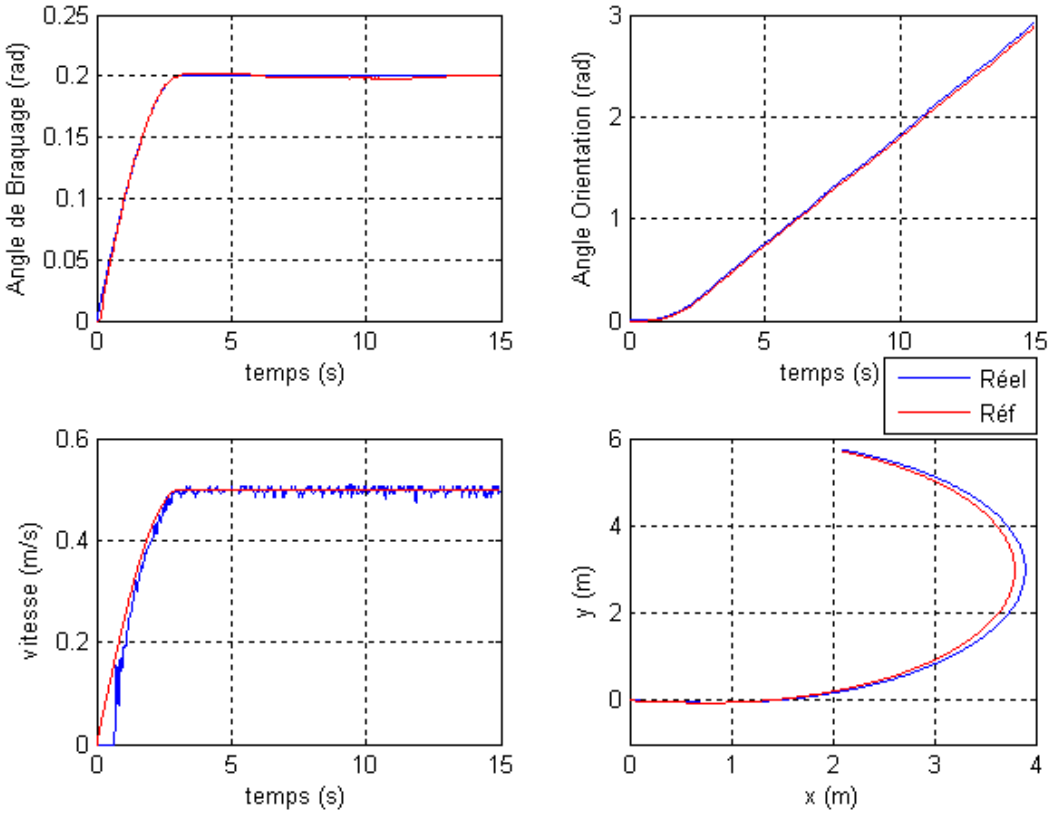

Figure 4.28: La poursuite de trajectoire

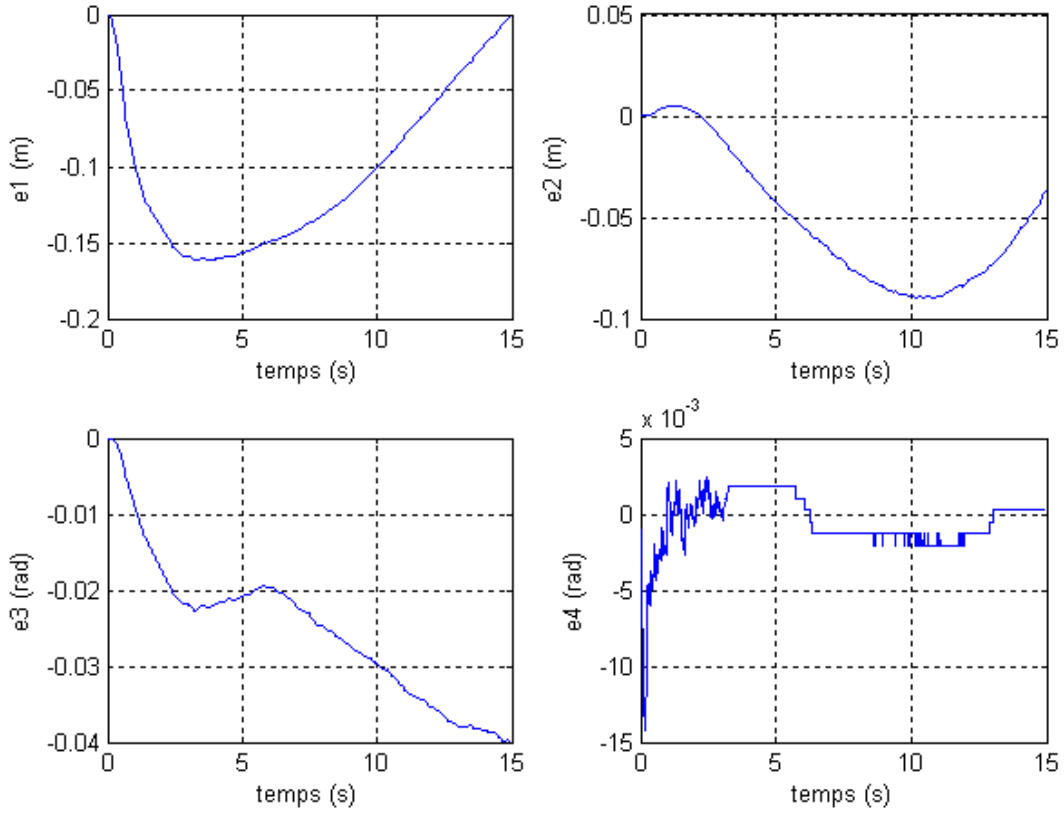

Figure 4.29: Les variables d'erreurs

Cas de la commande par backstepping :

La figure (4.30) représente les profils de vitesse et des angles de braquage et d'orientation. Ainsi que la trajectoire parcourue par les robots réel et virtuel. La figure (4.31) exprime les erreurs (e_1, e_2, e_3, e_4) des variables cinématiques du robot réel par rapport au robot virtuel.

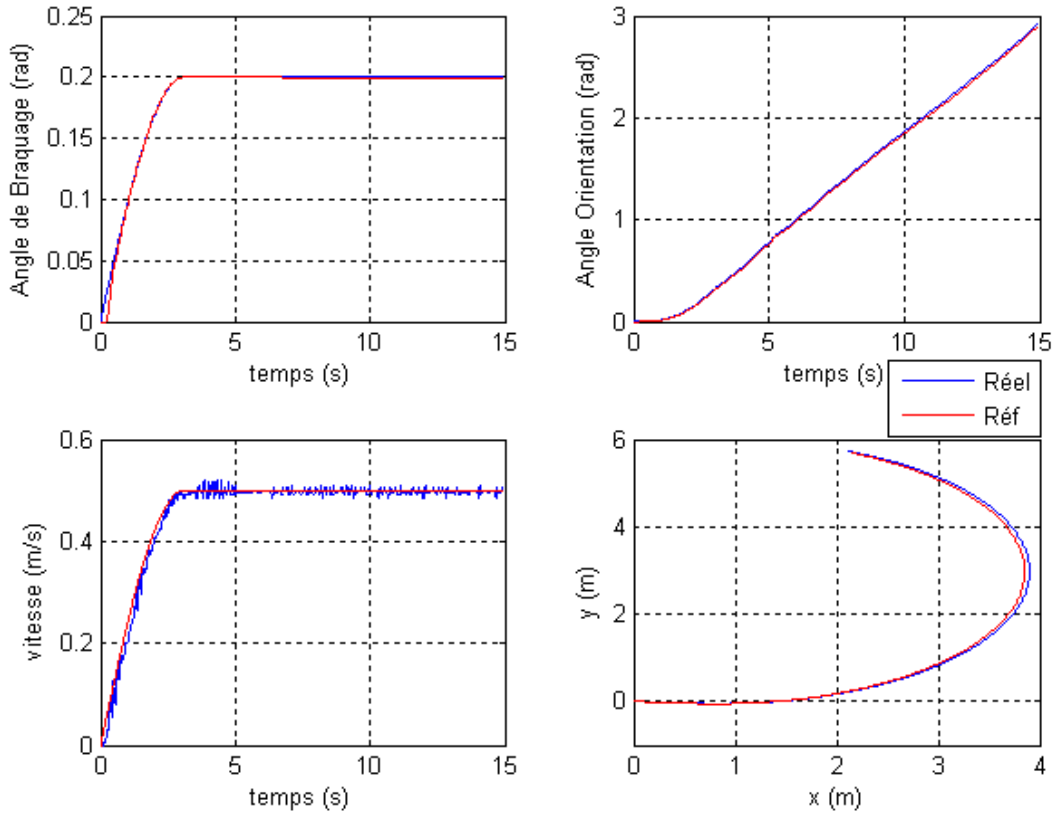

Figure 4.30: La poursuite de trajectoire

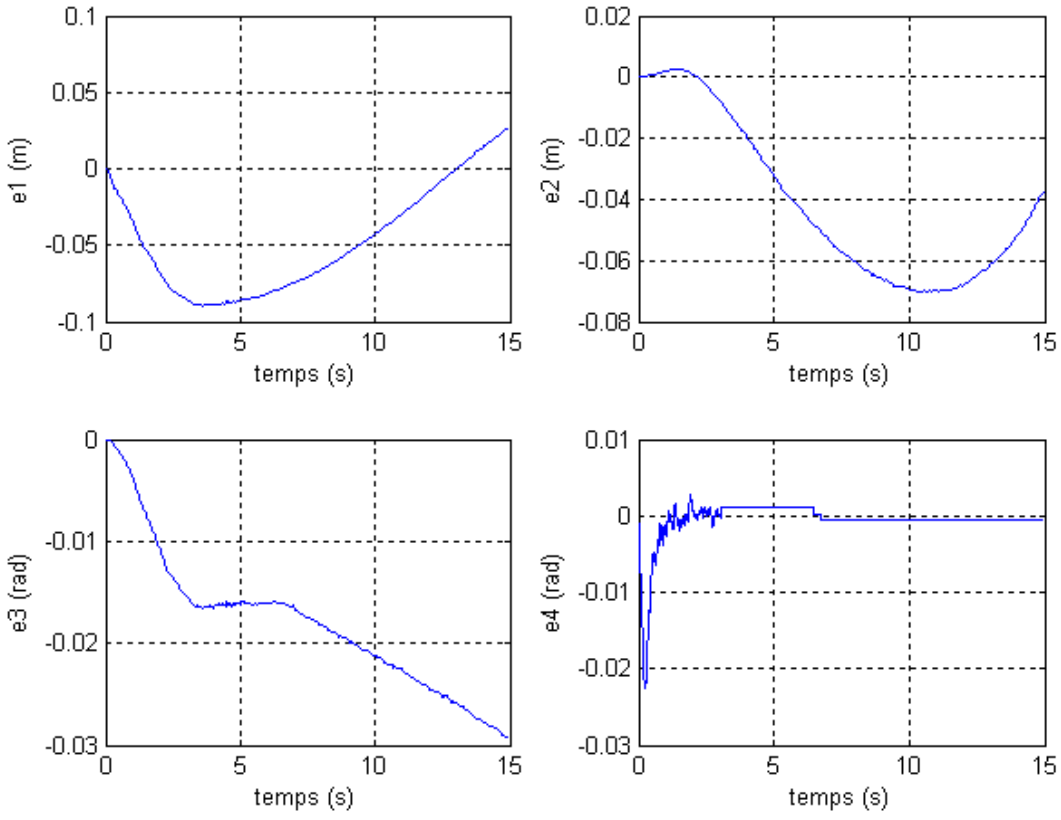

Figure 4.31: Les variables d'erreurs

Cas de la commande par Hybride (Backstepping - Mode glissant classique) :

La figure (4.32) représente les profils de vitesse et des angles de braquage et d'orientation. Ainsi que la trajectoire parcourue par les robots réel et virtuel. La figure (4.33) exprime les erreurs (e_1, e_2, e_3, e_4) des variables cinématiques du robot réel par rapport au robot virtuel.

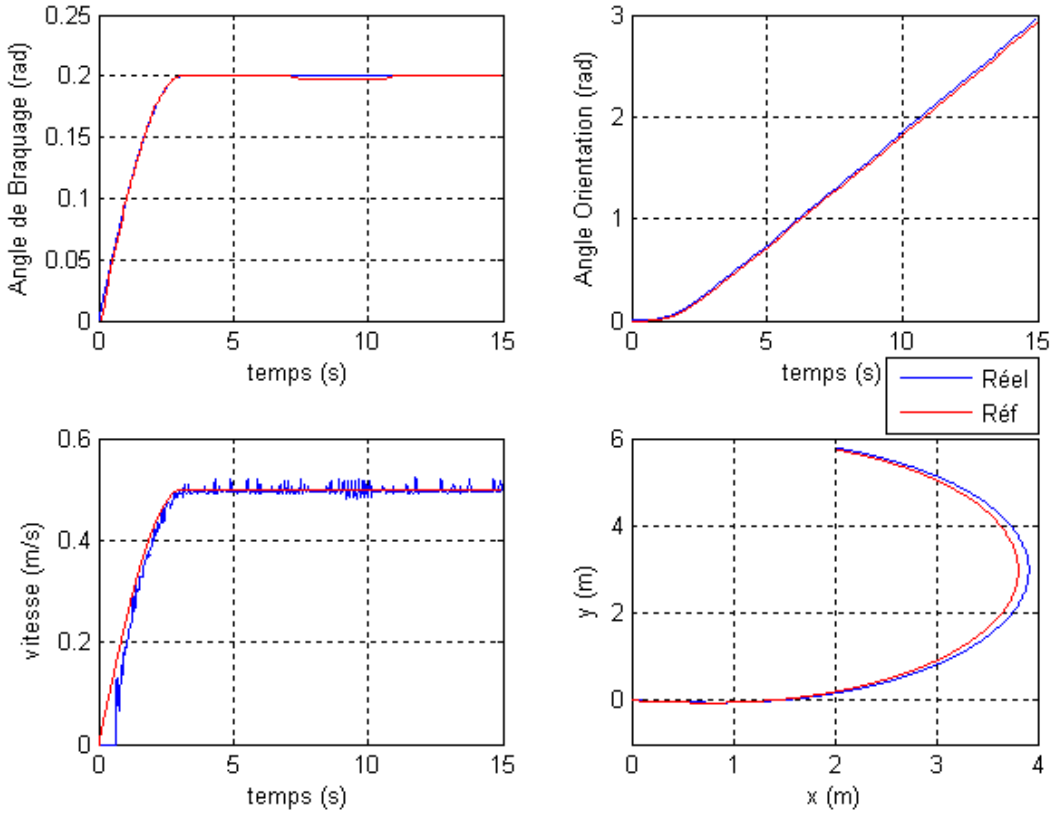

Figure 4.32: La poursuite de trajectoire

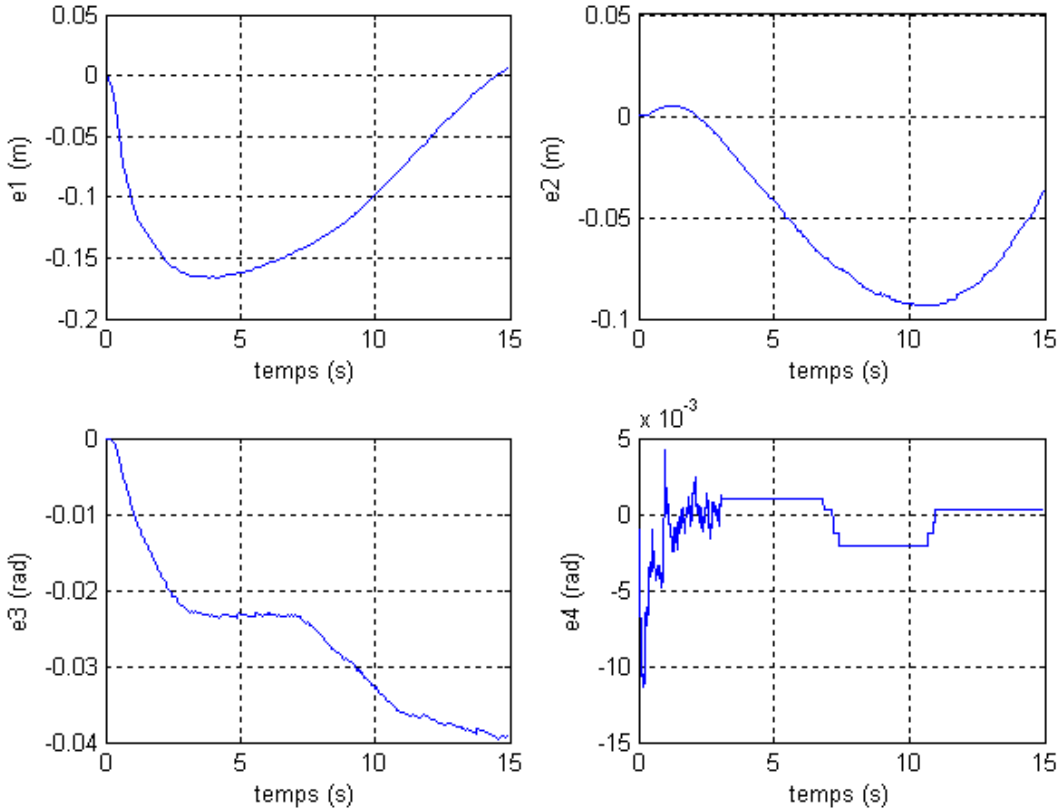

Figure 4.33: Les variables d'erreurs

Discussion

Des fluctuations apparaissent au niveau des entrées de commande : La vitesse de traction et l'angle de braquage (voir les figures (4.28),(4.30),(4.32)) du Robucar. Elles résultent d'une part, de l'estimation faite par les lois de commande pour assurer la poursuite de trajectoire, d'autre part, elles résultent des erreurs engendrées par les capteurs odométriques. Ces fluctuations engendrent des erreurs (e_1, e_2, e_3, e_4) dans la poursuite de trajectoire qui apparaissent au niveau de la position (x, y) du robot de sa vitesse de traction, de son angle d'orientation et de son angle de braquage (voir les figures (4.29),(4.31),(4.33)).

b) Trajectoire en forme de circuit fermé à quatre virages

Pour réaliser la poursuite de trajectoire le long du circuit fermé, on applique la commande par mode glissant d'ordre 2 en utilisant l'algorithme du Super-Twisting.

La figure (4.34) ci-dessous représente la trajectoire réalisée ainsi que les profils de vitesse, de braquage et d'orientation. La figure (4.35) représente les erreurs (e_1, e_2, e_3, e_4) des variables du modèle cinématique du robot réel par rapport au robot virtuel.

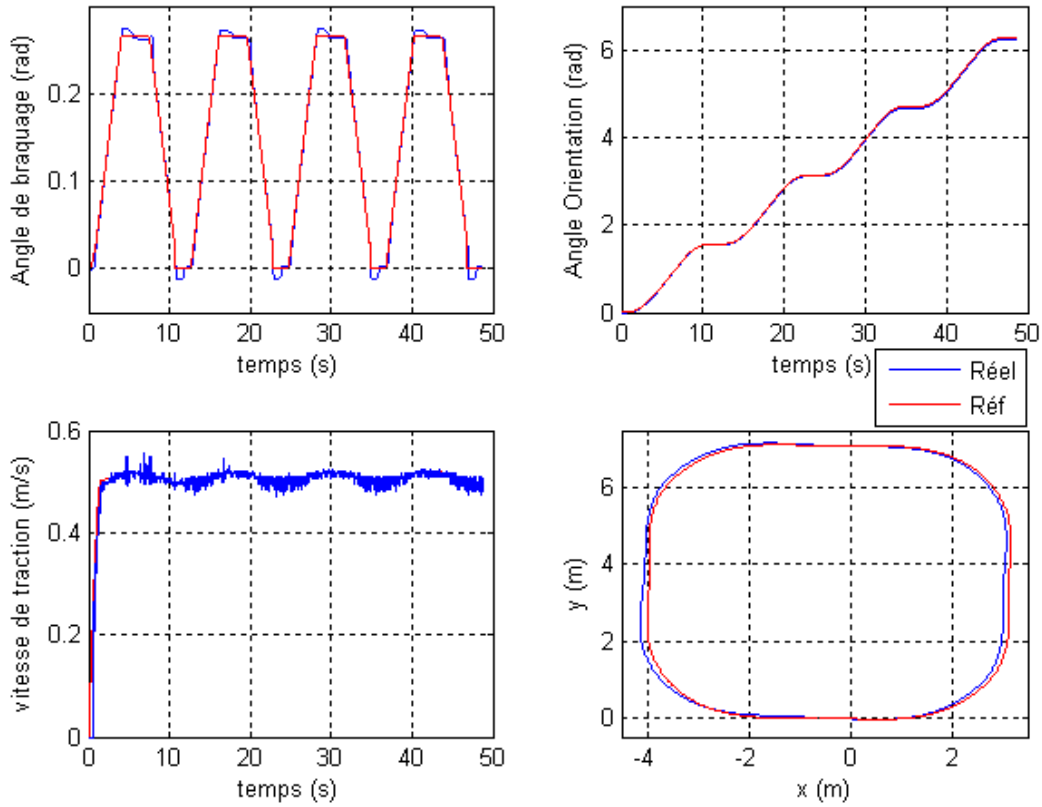

Figure 4.34: Poursuite de trajectoire : cas du circuit fermé

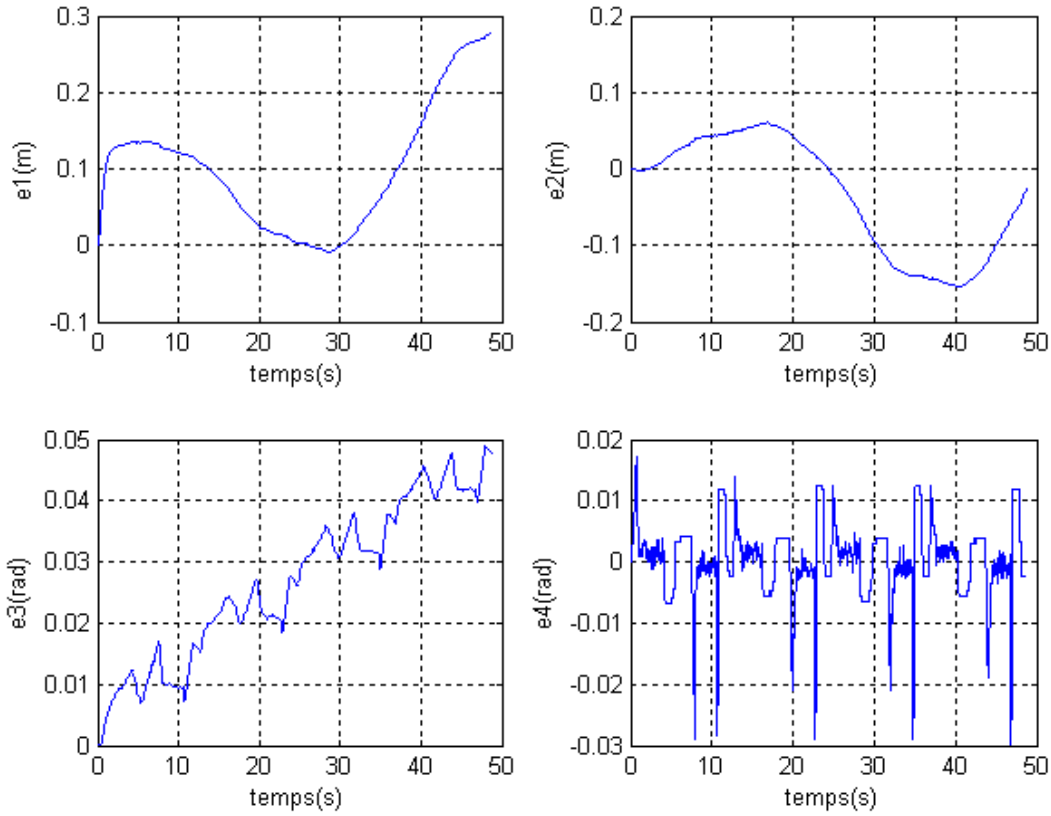

Figure 4.35: Les variables d'erreurs: cas du circuit fermé

Discussion :

De faibles dépassements apparaissent à chaque virage sur le profil de l'angle de braquage (voir la figure (4.34)), elles sont dues aux faibles erreurs engendrées par la loi de commande. Cependant, ces erreurs sont corrigées, ce qui permet d'avoir un profil de braquage du robot réel qui se superpose avec le profil de braquage du robot de référence. De faibles fluctuations apparaissent sur le profil de la vitesse de traction (voir la figure (4.34)), elles sont dues d'une part à la loi de commande générée pour réaliser la poursuite de trajectoire, d'autre part, elles sont dues essentiellement aux erreurs odometriques engendrées par le robot lui-même. Ces fluctuations engendrent des erreurs (e_1, e_2, e_3, e_4) (voir la figure (4.35)) visibles dans la trajectoire réalisée par le robot (voir la figure (4.34)). Les erreurs odometriques qui interviennent sur la vitesse et le braquage engendrent des erreurs sur l'orientation du robot (voir l'erreur e_3 dans la figure (4.34)), ces fluctuations engendrent aussi des erreurs de positions (x, y) (voir respectivement les erreurs e_1 et e_2 dans la figure (4.34)).

c) Tests de robustesse

On présente les tests de robustesse dans le cas d'une commande en poursuite de trajectoire le long d'un arc de cercle. La commande appliquée est la commande par mode glissant d'ordre 2 en utilisant l'algorithme du Super-Twisting. Les tests de robustesse sont réalisés en ajoutant des perturbations au robot de référence, ce qui entrainera une déviation de la trajectoire de référence que doit suivre le ROBUCAR. La première perturbation intervient dans le changement momentané du profil de vitesse de traction du robot de référence (virtuel) (voir la figure (4.36)). La seconde perturbation intervient dans le changement momentané du profil de braquage du robot de référence (virtuel) (voir la figure (4.37)).

Cas de la perturbation sur la vitesse de traction :

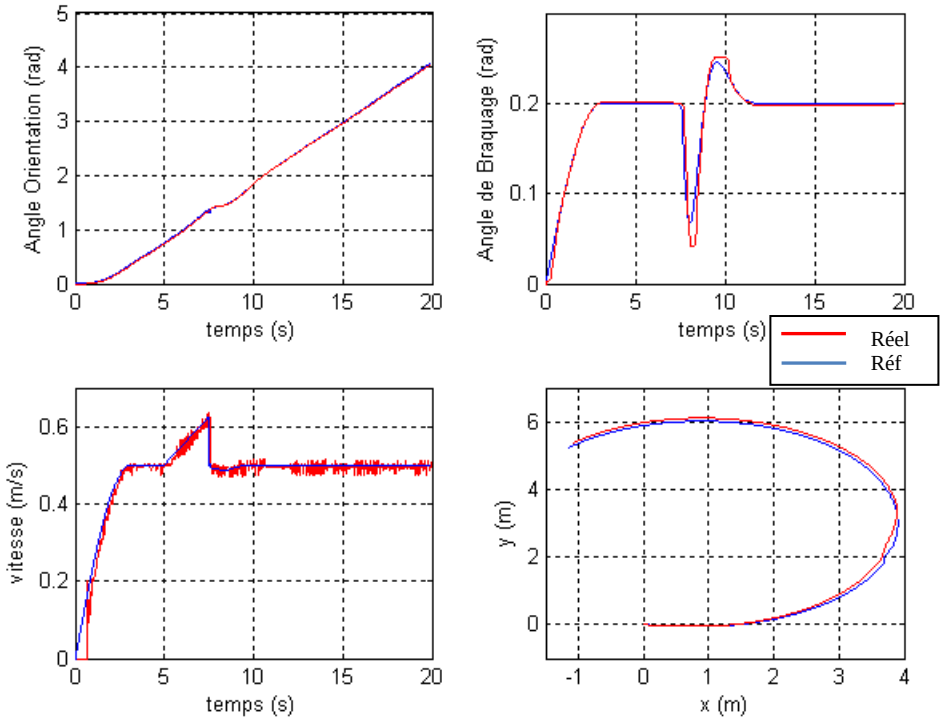

Figure 4.36: Poursuite de trajectoire : cas de la perturbation sur la vitesse de traction

Cas de la perturbation sur le profil de l'angle de braquage :

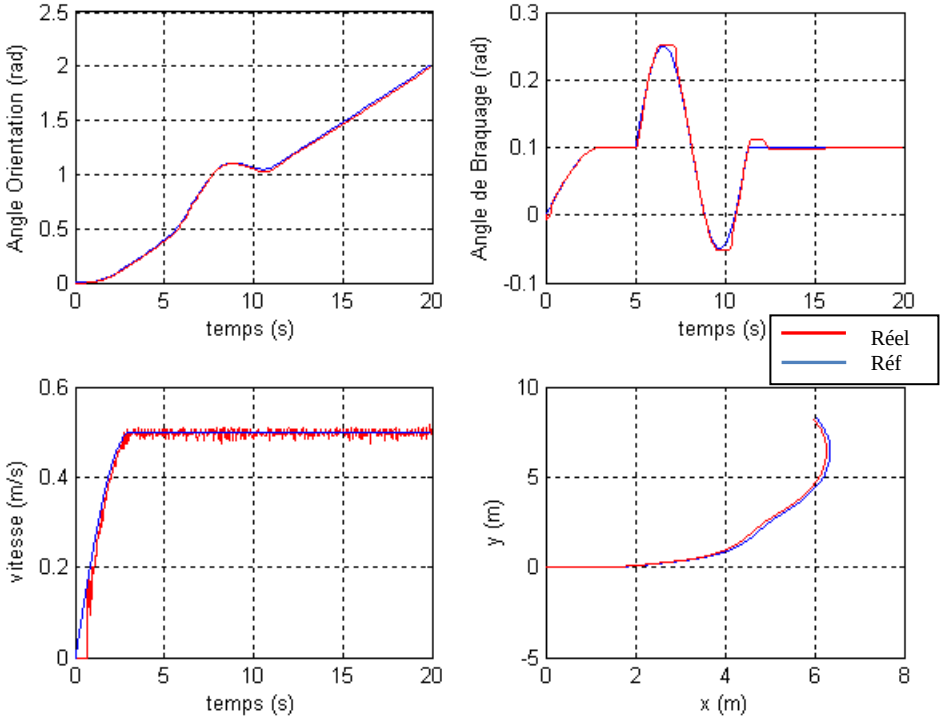

Figure 4.37: Poursuite de trajectoire : Cas de la perturbation sur le profil de l'angle de braquage

La perturbation introduite dans le profil de vitesse de traction du robot de référence (voir la figure (4.36)) engendre une légère déviation de la trajectoire du ROBUCAR, cette déviation est visible sur le profil d'angle d'orientation ainsi que la position (x, y) du ROBUCAR. Ce dernier suit la déviation de la trajectoire désirée grâce aux variations de l'angle de braquage.

La perturbation introduite dans le profile de braquage du robot de référence (voir la figure (4.37)) engendre une déviation plus importante de la trajectoire du ROBUCAR. Cependant les variations de l'angle de braquage du ROBUCAR permettent de suivre la trajectoire déviée.

4.4 Conclusion

La validation des commandes en poursuite de trajectoire élaborées est effectuée en simulation et en expérimentation. Dans les simulations, les résultats des tests de commandes en poursuite de trajectoire présentés dans ce chapitre sont satisfaisants quelque soit la perturbation introduite (erreur de départ en position, bruit blanc gaussien, changement de profil de la trajectoire). Les tests expérimentaux sont aussi satisfaisants, bien que l'on remarque des erreurs dans la poursuite de trajectoire que l'on attribue essentiellement aux erreurs engendrées par les mesures effectuées par les capteurs odométriques.

Conclusion générale

Dans ce mémoire, nous nous somme intéressé à la génération de trajectoire et à la commande en poursuite de trajectoire pour le Robucar. Ce dernier étant un robot mobile non-holonome, sous actionné de type voiture choisi en mode double braquage.

Les objectifs de notre travail sont d'une part la génération d'une trajectoire réalisable pour le Robucar, en prenant en considération ses contraintes cinématiques. D'autre part, l'élaboration de lois de commande robustes qui permettent au robot de réaliser la poursuite de trajectoire et de garantir sa stabilité au sens de Lyapunov.

Pour cela, on a commencé par présenter les caractéristiques liées à la cinématique de quelques robots mobiles et plus précisément les robots de type voiture en double braquage. Dans le but d'introduire les approches de planification et de commande développées dans ce mémoire, quelques généralités liées aux méthodes de planification de trajectoire ont été présentées, ainsi que certains travaux liés à la commande en poursuite de trajectoire pour les robots mobiles de type voiture en simple et en double braquage.

Pour la génération d'une trajectoire réalisable pour le Robucar, nous avons opté pour l'utilisation des courbes analytiques ayant la caractéristique d'avoir une courbure variable. La courbe utilisée est connue sous le nom de spirale de Cornu. Cette continuité dans la trajectoire se traduit par une continuité du mouvement du robot mobile. Ce dernier peut se déplacer le long du chemin sans manœuvrer. La courbe générée prend en considération les contraintes cinématiques du Robucar qui sont : un angle de braquage borné pour les roues avant et arrière, des vitesses de translation et de rotation des roues bornées. La trajectoire générée est un assemblage de courbes (ligne droite, arc de cercle, spirale de cornu). A partir de la courbe générée on extrait les profils de braquage, d'orientation, et des positions représentant la trajectoire parcourue par le robot virtuel.

Afin de développer une commande par retour d'état pour la poursuite de trajectoire, la mise en forme du modèle d'erreur de poursuite est nécessaire pour exprimer ce dernier dans l'espace d'état. Grâce à une transformation par difféomorphisme, on a obtenu un modèle d'erreur équivalent au premier à la différence que le nouveau modèle est une représentation d'état.

Les commandes élaborées pour exécuter une poursuite de trajectoire sont les modes glissants d'ordre deux et le Backstepping. Les commandes par modes glissants ont la caractéristique d'être facile à mettre en œuvre et robuste face aux perturbations. Les algorithmes glissants d'ordre deux utilisés sont le Twisting et le Super-Twisting. A cause du

caractère discontinu des modes glissants, une troisième méthode est élaborée en utilisant le Twisting avec une commande équivalente continue dans le but de diminuer le chattering. Ces commandes assurent la stabilité au sens de Lyapunov du système. La commande par backstepping utilisée a permis d'élaborer une commande robuste où l'on démontre plus facilement la stabilité du système au sens de Lyapunov. La dernière commande élaborée est basée sur les modes glissants d'ordre un et le backstepping. Cette commande a garanti en effet, la stabilité du système grâce aux propriétés des commandes par backstepping tout en étant robuste face aux perturbations internes et externes et cela grâce aux propriétés des modes glissants.

La validation de ces commandes est effectuée en simulation et en expérimentation. Dans les simulations, les résultats des tests des différentes commandes élaborées en poursuite de trajectoire sont satisfaisants quelque soit la perturbation introduite (erreur de départ en position, bruit blanc gaussien, changement de profil de la trajectoire). Les tests expérimentaux sont aussi satisfaisants, bien que l'on remarque des erreurs dans la poursuite de trajectoire que l'on attribue aux erreurs engendrées par les capteurs odométriques.

En perspective, il serait intéressant de tester les commandes élaborées dans le cas où l'on prend en considération les hypothèses de glissement du robot Bis-Car, cela permettrait l'élaboration de commandes plus robustes face aux perturbations de glissement que subit le Robot dans la réalité. Dans le cas où l'on prend en considération les erreurs de modélisation inhérentes au robot, l'élaboration d'une commande par Backstepping adaptatif serait intéressante pour augmenter la robustesse de la commande. Une autre perspective intéressante et propre au robot, est celle d'utiliser les propriétés de platitude de ce dernier pour faciliter l'élaboration de la commande et la planification de trajectoire. En effet la propriété de platitude permet d'une part de transformer le système plus facilement en une forme canonique telle que la forme chainée, ce qui rend la synthèse de la loi de commande plus facile. D'autre part la propriété de platitude du robot facilite la détermination des postures du robot de référence à partir d'une trajectoire de référence.

Annexe 1 : Invariance des régimes glissants par rapport aux perturbations [Hame.,93]

Soit le système dynamique suivant :

$$\frac{dx}{dt} = f(x) + g(x)\,u(x) + \xi \qquad A1.1$$

Avec : $\xi = \delta . f(x)$.

ξ représente les perturbations paramétriques du champ de vecteur $f(x)$. Le régime glissant est invariant par rapport au vecteur perturbation ξ, si ce dernier satisfait la définition suivante :

Définition :

Le régime glissant idéal jouit de propriété d'invariance forte par rapport au signal de perturbation ξ chaque fois que la dynamique du système en mode de glissement idéal est indépendante du signal de perturbation ξ.

En sachant que le champ de vecteur $g(x)$ n'intervient par dans le mode glissant idéal, soit le théorème suivant :

Théorème :

Le régime glissant sur la variété S du système perturbé (A1.1), satisfait les propriétés d'invariance vis à vis de ξ, si et seulement si le vecteur de la perturbation ξ satisfait la condition suivante :

$$\xi \in Span\{g(x)\}$$

Preuve :

Pour le système dynamique perturbé (A1.1), le régime glissant idéal est gouverné par :

$$\frac{dx}{dt} = F.(f + \delta . f)$$

Avec : $F = [I - g(x).(\langle \nabla S, g(x) \rangle)^{-1}.(\nabla S)]$

La condition nécessaire vient du fait que si

$$F.\xi = [I - g(x).(\langle \nabla S, g(x) \rangle)^{-1}.(\nabla S)].\xi = 0$$

Alors on obtient :

$$\xi = [g(x).(\langle \nabla S, g(x) \rangle)^{-1}.(\nabla S)].\xi = g(x).\mu(x)$$

c'est à dire que $\xi \in Span\{g(x)\}$ pour toute fonction scalaire continue $\mu(x)$. Dans ce cas l'opérateur F annule l'influence de ξ sur la dynamique équivalente du régime glissant idéal.

Annexe 2 : Etude de la convergence asymptotique du système à commander

Soit le système à commander régi par le modèle d'erreur de poursuite (3.5.2.1) donné par :

$$\begin{cases} \dot{z}_1 = w_1 + z_2\dot{\theta}_r \\ \dot{z}_2 = v_r\big(h_2\, cos(\varphi_r) - \ cos(\varphi_r)\big) - z_1\dot{\theta}_r + w_1 h_2 \\ \dot{z}_3 = v_r\, z_4 \cos(\varphi_r) - v_r\, k\, z_2 \cos(\varphi_r) + w_1 h_1 \\ \dot{z}_4 = w_2 \end{cases} \qquad (A2.1)$$

Avec :

$$\begin{cases} \ h_1 = z_4 - k\, z_2 + tan(\varphi_r)(z_3^2 + 1) \\ h_2 = \dfrac{z_4 - k\, z_2 + tan(\varphi_r)(z_3^2 + 1)}{\sqrt{(z_3^2 + 1)}} + z_3 \end{cases}$$

Démontrer la convergence du système $(A2.1)$ ci-dessus, revient à montrer que la fonction de Lyapunov tend vers zéro lorsque le temps tend vers l'infini [Hame1.,07]. On applique au système $(A2.1)$ la loi de commande qui est caractérisée par les surfaces de glissement suivantes :

$$s_1 = \left(z_1 + z_2\, h_2 + \frac{h_1 z_3}{k}\right) v_r + w_1 \qquad (A2.2)$$

$$s_2 = \left(\frac{z_2 + z_3}{\sqrt{(z_3^2 + 1)}}\right) v_r + k z_4 v_r + w_2 \qquad (A2.3)$$

Supposons que le système $(A2.1)$ atteint le régime glissant lorsque $s_1 = s_2 = 0$:

Soit la fonction de Lyapunov données par la forme suivante :

$$V = \frac{1}{2}\left(z_1^2 + z_2^2 + \frac{z_3^2}{k} + z_4^2\right)$$

La dérivée de la fonction de Lyapunov s'écrit de la forme suivante :

$$\dot{V} = w_1\left(z_1 + z_2\, h_2 + \frac{h_1 z_3}{k}\right) + \frac{z_2 + z_3}{\sqrt{(z_3^2 + 1)}}\, z_4 \cos(\varphi_r) - k\frac{z_2^2}{\sqrt{(z_3^2 + 1)}}\, cos(\varphi_r)$$
$$+ z_2\left(\sqrt{(z_3^2 + 1)} - \ 1\right) sin(\varphi_r) + z_4 w_2$$

Quand le système atteint le régime glissant idéal, les surfaces de glissement données par les équations $(A2.2)$ et $(A2.3)$ sont nulles :

$$s_1 = 0 \Leftrightarrow \ w_1 = -v_r\left(z_1 + z_2\, h_2 + \frac{h_1 z_3}{k}\right) \qquad (A2.4)$$

$$s_2 = 0 \Leftrightarrow\ w_2 = -\left(\frac{z_2 + z_3}{\sqrt{(z_3^2 + 1)}}\right) v_r - k\ z_4\ v_r \qquad (A2.5)$$

En remplaçant ces deux équations $(A2.4)$ et $(A2.5)$ dans l'équation de $\dot{V}$ on obtient :

$$\dot{V} = -v_r \left(z_1 + z_2\ h_2 + \frac{h_1 z_3}{k}\right)^2 - k\ z_4^2 - k \frac{z_2^2}{\sqrt{(z_3^2 + 1)}}\ cos(\varphi_r)$$
$$+z_2 \left(\sqrt{(z_3^2 + 1)} - \ 1\right) sin(\varphi_r) \qquad (A2.6)$$

Etant donnée que $\dot{V}$ est négatif et V est positif. V est alors décroissante jusqu'à atteindre un seuil $V_0 \geq 0$ donc les variables d'états z_i sont bornées.

v_r est strictement positif. $v_r\ et\ z_i$ sont bornés et uniformément continus. On en déduit que $\dot{V}$est uniformément continue. En appliquant le lemme de barbalat [Slot.,98]. On obtient :
$lim_{t\to\infty} \frac{dV}{dt} = 0.$

Il en résulte que :
$lim_{t\to\infty} v_r z_2 = 0$. On en déduit que $lim_{t\to\infty} z_2 = 0$.

$lim_{t\to\infty} v_r z_4 = 0$. On en déduit que $lim_{t\to\infty} z_4 = 0$.

$lim_{t\to\infty} v_r \left(z_1 + z_2\ h_2 + \frac{h_1 z_3}{k}\right) = 0$. On en déduit que $lim_{t\to\infty} \left(z_1 + z_2\ h_2 + \frac{h_1 z_3}{k}\right) = 0$.
Soit l'expression : $v_r^2 z_4$

$$\frac{d v_r^2 z_4}{dt} = 2\dot{v}_r v_r z_4 + v_r^2 w_2$$

On remplace l'expression de w_2 dans le cas du régime glissant idéal :

$$\frac{d v_r^2 z_4}{dt} = 2\dot{v}_r v_r z_4 + v_r^3 \left(-\left(\frac{z_2 + z_3}{\sqrt{(z_3^2 + 1)}}\right) - k\ z_4\right)$$

$$\frac{d v_r^2 z_4}{dt} + v_r^3 \left(\frac{z_2 + z_3}{\sqrt{(z_3^2 + 1)}}\right) = (2\dot{v}_r v_r - v_r^3 k)\ z_4$$

Comme $lim_{t\to\infty} z_4 = 0$. On en déduit que : $lim_{t\to\infty}\left[v_r^3\left(\frac{z_2+z_3}{\sqrt{(z_3^2+1)}}\right)\right] = 0$.

Comme $lim_{t\to\infty} z_2 = 0$. On en déduit que : $lim_{t\to\infty} z_3 = 0$.

Et enfin comme z_2 et z_3 converge vers zéro et que $lim_{t\to\infty}\left(z_1 + z_2\, h_2 + \frac{h_1 z_3}{k}\right) = 0$. On en déduit que $lim_{t\to\infty} z_1 = 0$.

On obtient alors $lim_{t\to\infty} V = 0$.

Les états z_i du système convergent asymptotiquement vers zéro quand le temps tend vers l'infini. Il en résulte que la trajectoire du système converge asymptotiquement vers la trajectoire désirée.

Bibliographie :

[Agra.,08] A. A. Agrachev, and al, “Nonlinear and optimal control theory”. Edition Springer, 2008.

[Bena.,03] A. Benalia, M. Djemai, J. P. Barbot “Control of the kinematic car using trajectory generation and high orders sliding mode control”. dans Systems, Man and Cybernetics 2003 IEEE Inernational Conference. vol.3, pp. 2455–2459,2003.

[Boud.,09] S. Boudoua, “Robust control for an artificial muscles robot arm”. 6th International Conference on Informatics in Control, Automation and Robotics, (INCINCO 2009), Juillet 2009.

[Bour.,07] S. Bourain, “ Contribution à la localisation dynamique du robot mobile d’intérieur B21r en utilisant la plateforme multi sensorielle ”. Mémoire de magister, Université de Saad Dahleb, Blida, Novembre 2007.

[Chet.,08] M. Chettouh, “ Commande d’un robot à muscle artificiels par les régimes glissants et réduction du broutement”. Thèse Phd, USTHB, Decembre 2008.

[D’And.,92] B. D’Andréa-Novel, G. Bastin, G. Campion, “Dynamic Feedback Linearization of Nonholonomic Wheeled Mobile Robots”. Proceeding, Robotics and Automation, 1992 IEEE, International Conference, pp. 2527–2532, Mai 1992.

[Defo.,07] M, Defoort, “ Contributions à la planification et à la commande pour robots mobiles coopératifs ”. Thèse Phd, Ecole centrale de lille, France, Octobre 2007.

[Dubi.,57] L. E. Dubins, “ On curve of minimal length with a contrain on average curvature, and with prescribed initial and terminal positions and tengents ”. American Journal of Mathematics, Vol. 79, No. 3, pp. 497-516, 1957.

[Frai.,90] T.Fraichard Ch. Laugier, G. Liévin, “Robot motion planning: the case of non holonomic mobiles in a dynamic world”. Proceeding, Intelligent Robots and Systems,vol.2, pp.757–764, 1990.

[Frai.,04] T. Fraichard, A. Scheuer, “ From Reeds and Shepp’s to continuous curvature path ”. IEEE Trans. On Robotics, Vol. 20, No. 6, pp. 1025-1035, Decembre 2004.

[Frid.,02] L. Fridman, A. Levant, “High-Order Sliding Modes”. In Sliding Mode Control in Engineering, W. Perruquetti, and J. P. Barbot, Ed. Marcel Dekker, pp. 62–95,2002.

[Fliess.,95] M. Fliess, J.Lévine, P.Martin, P.Rouchon, “Design of Trajectory stabilizing feedback for driftless flat systems”. Proceeding, 3rd European Control Conference, Rome, Italy, ECC’95, pp. 1882-1887, 1995.

[Ghom.,08] J. Ghommam. “ Commande Nonlinéaire et Navigation des Véhicules Marins Sous-actionnés ”. Thèse PhD, Université d’Orléans, France, Janvier 2008.

[Hame.,93] M. Hamerlain, “ Commande hiérarchisée à modèle de référence et à structure variable d’un robot manipulateur à muscles artificiels ”. Thèse de Doctorat. Toulouse, 1993.

[Hame.,05] F. Hamerlain, “ Stabilisation par modes glissants d’ordre supérieur de systèmes non holonomes : cas de robots mobiles à roues ”. Mémoire de Magister. USTHB, 2005.

[Hame1.,07] F. Hamerlain, K. Achour, T. Floquet, W. Perruquetti, “Trajectory tracking of a car-like robot using second order sliding mode control”. Proceedings European Control Conference, Kos, Greece, pp. 4932-4936, 2007.

[Hame2.,07] F. Hamerlain, “ Stratégies de commande par modes glissants d.ordre supérieur appliquées à des robots mobiles à roues” . Thèse PhD, Ecole centrale de lille, France, Décembre 2007.

[Herm.,03] J. Hermosillo, S. Sekhavat, “Feedback control of a bi-steerable car using flatness application to trajectory tracking”. In Proc. American Control Conference, Denver, Colorado, vol.1, pp. 312–317, 2003.

[Hong.,02] C.K. Li, Y.M. Hongmin Chao Hu, A.B. Rad, " Output tracking control of mobile robots based on adaptive backstepping and high order sliding modes ". Systems, Man and Cybernetics, IEEE International Conference, Vol. 5, 2002.

[Kana.,89] Y. Kanayama, B. Hartman, " Smouth local path planning for Autonomous Vehicules". Robotics and Automation, Proceedings, IEEE International Conference, 1989.

[Koko.,95] P. Kokotovic, M. Krstic, " Nonlinear and adaptive control design ". Edition Wiley-Interscience, 1995.

[Lami.,01] F. Lamiraux, J.-P. Laumond, «Smooth Motion Planning for Car-Like Vehicles». IEEE Trans. On Robotics and Automation, Vol. 17, No. 4, pp. 498-502, Aout 2001.

[Lato.,91] J.C. Latombe, " Robot motion planning ". Edition Kluwer Academic Publishers, 1991.

[Laum.,98] A. De Luca, G. Oriolo, C. Samson, " Feedback Control of a Nonholonomic Car-Like Robot " . In Robot Motion Planning and Control (Lectures Notes in Control and Information Sciences 229), J. P. Laumond , Ed. Springer, 1998.

[Laum.,01] J. P. Laumond, " La robotique mobile ". Edition Hèrmes, 2001.

[Leva.,93] A. Levant, "Sliding order and sliding accuracy in sliding mode control". International journal of control, vol. 58, No. 6, pp. 1247-1263, Octobre 2006.

[Leva.,06] A. Levant, "Construction principles of 2-sliding mode design". Automatica, vol. 43, No. 4, pp. 576-586, Octobre 2006.

[Mori.,00] P.Morin C. Samson, "Control of Nonlinear Chained Systems: From the Routh–Hurwitz Stability Criterion to Time-Varying Exponential Stabilizers". IEEE Trans. Automatic Control,vol.45, pp.141–146, Janvier 2000.

[Mori2.,00] P.Morin C. Samson, “Practical stabilization of a class of nonlinear systems. Application to chain systems and mobile robots”. Proceeding, Decision and control, vol 3, pp.2989–2994, December 2000.

[Orli.,92] G. Orlio, A. De Luca, M. Vendittelli, “WMR Control Via Dynamic Feedback Linearization: Design, Implementation, and Experimental Validation”. IEEE Trans. Control systems and technology, vol.10, No.6, pp.835–852, Novembre 2002.

[Perr.,02] A. J. Fossart, T. Floquet, “Introduction: An overview of Classical Sliding Mode Control”. In Sliding Mode Control in Engineering, W. Perruquetti, and J. P. Barbot, Ed. Marcel Dekker, 2002.

[Prus.,96] A. Pruski, “ Robotique mobile : la planification de trajectoire ”. Edition Hermès, 1996.

[Ran.,09] Ran Dai, J.E. Cochran, “Path planning for multiple unmanned aerial vehicles by parameterized cornu spirals”. In Proc. American Control Conference, St Louis, MO, USA, pp. 2391–2396, 2009.

[Sams.,85] C. Samson, “ Control of Chained systems Application to Path Following and Time-Varying Point-Stabilization of Mobile Robots”. IEEE Trans. Automatic Control,vol.40, pp.64–77, Janvier 1985.

[Sekh.,00] S. Sekhavat, J.Hermosillo, “ The cycab robot : a Differentially flat system ”. Proceeding, Intelligent Robots and Systems, 2000 IEEE/RSJ, International Conference. vol.1, pp. 312–317, 2000.

[Sche.,98] A. Scheuer, “ Planification de chemins à courbure continue pour robot mobile non holonome ”. Thèse PhD, INPG, France, 1998.

[Sche2.,98] A. Scheuer, Ch. Laugier, “Plannig sub-optimal and continuous-curvature paths for car-like robots”. Proceeding, Intelligent Robots and Systems, 1998 IEEE/RSJ, International Conference. vol.1, pp. 25–31, Octobre 1998.

[Slot.,98] J-J. E. Slotine, W. Li, "Applied Nonlinear Control". Edition Prentice Hall, 1998.

[Tche.,11] S. A. Tchenderli-Braham, F. Hamerlain. " Commande en poursuite de trajectoire par Mode Glissant d'Ordre 2 Appliquée au Bis-Car ". CGE'07. EMP, Alger, Algérie. Avril 2011.

[Utki.,77] V. I. Utkin, "Variable structure Systems with sliding modes". IEEE Trans. Automatic Control, vol.22, No.2, pp.212–222, Avril 1977.

[Yang.,04] E. Yang, D. Gu, T. Mita, and H. Hu, "Nonlinear Tracking Control of a Car-like Mobile Robot via Dynamic Feed back Linearization". Proceedings of Control 2004, Bath, UK, pp 6-9, Septembre 2004.

[Yang.,99] J.-M. Yang, J.-H Kim, "Sliding Mode Control for Trajectory Tracking of Nonholonomic Wheeled Mobile Robots". IEEE Trans. On Robotics and Automation, Vol. 15, No. 3, pp. 578-587, Juin 1999.

[Wals.,94] G. Walsh, D. Tilbury, S. Sastry, R. Murray, J.-P. Laumond, "Stabilization of trajectories for systems with nonholonomic constraints". IEEE Trans. Automatic Control, vol.39, pp.216–222, Janvier 1994.

[Levi.,04] J. Levine, " Analyse et commande des systèmes non linéaires". http://cas.ensmp.fr/~levine/Enseignement/CoursENPC.pdf

Résumé

Ce mémoire a consisté premièrement dans la génération d'une trajectoire admissible et réalisable prenant en compte les contraintes cinématiques et de non holonomie du robot mobile de type voiture en double braquage. La seconde partie a traité le problème de la commande en poursuite de trajectoire par différentes techniques (Mode glissant d'ordre 2, Backstepping, Hybride). Le modèle d'erreurs de poursuite du robot a été développé et des lois de commande ont été élaborées et validées en simulation et en expérimentation.

Summary

This work deals firstly about admissible and achievable trajectory generation taking into account the kinematic constraints and non-holonomy of the mobile robot car in double steering mode. The second part deals with the problem of trajectory tracking control by different techniques (sliding mode of order 2, Backstepping, Hybrid). The model of tracking errors of the robot was developed and control laws were developed and validated in simulation and experimentation.