



Contributions à la planification et à la commande pour les robots mobiles coopératifs

Michael Defoort

► To cite this version:

Michael Defoort. Contributions à la planification et à la commande pour les robots mobiles coopératifs. Automatic. Ecole Centrale de Lille, 2007. French. <tel-00196529>

HAL Id: tel-00196529

<https://tel.archives-ouvertes.fr/tel-00196529>

Submitted on 13 Dec 2007

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

ÉCOLE CENTRALE DE LILLE

THÈSE

présentée en vue d'obtenir le grade de

DOCTEUR

Spécialité : Automatique et Informatique Industrielle

par

Michael Defoort

Ingénieur diplômé de l'ISEN

**CONTRIBUTIONS A LA PLANIFICATION
ET A LA COMMANDE POUR LES
ROBOTS MOBILES COOPERATIFS**

Doctorat délivré par l'École Centrale de Lille

Soutenue le 22 octobre 2007 devant le jury constitué de :

M. Michel Fliess	<i>Président</i>	Directeur de recherche CNRS, Ecole Polytechnique de Paris
M. Philippe Fraisse	<i>Rapporteur</i>	Professeur, Université Montpellier II
M. Hugues Mounier	<i>Rapporteur</i>	Maître de Conférences (HdR), Université Paris Sud XI
Mme Sarah Spurgeon	<i>Rapporteur</i>	Professeur, University of Leicester, U.K.
M. Sergey V. Drakunov	<i>Examinateur</i>	Professeur, Embry-Riddle Aeronautical University, USA
M. Wilfrid Perruquetti	<i>Codirecteur</i>	Professeur, École Centrale de Lille
M. Thierry Floquet	<i>Codirecteur</i>	Chargé de recherche CNRS, LAGIS
Mme Annemarie Kökösy	<i>Codirecteur</i>	Maître de Conférences, ISEN

Thèse préparée au Laboratoire d'Automatique, de Génie Informatique et Signal
L.A.G.I.S., UMR 8146 - Ecole Centrale de Lille

Table des matières

Remerciements	7
Notations	9
Introduction générale	13
1 Problématique et état de l'art	21
1.1 Introduction	21
1.2 Modélisation de la cinématique des véhicules à roues	21
1.2.1 Hypothèses de modélisation	21
1.2.2 Roulement sans glissement et non holonomie	22
1.2.3 Modélisation des robots à roues	24
1.3 Planification de trajectoire	27
1.3.1 Chemins et trajectoires	27
1.3.2 Planification de trajectoire : définition	27
1.3.3 Evitement réactif d'obstacles	27
1.4 Poursuite de trajectoire	30
1.4.1 Rappel sur la commande en boucle ouverte et fermée	30
1.4.2 Le problème de poursuite de trajectoire	31
1.5 Système multi-robots : les problématiques	33
1.5.1 Planification de trajectoire	34
1.5.2 Poursuite de trajectoire	36
1.6 Conclusion	37
I Planification de trajectoire	39
2 Planification de trajectoire : cadre mono-robot	41
2.1 Introduction	41
2.2 Description du problème	42

2.2.1	Propriétés de platitude	42
2.2.2	Description du problème de planification	44
2.3	Algorithme hors ligne de génération de trajectoire	46
2.3.1	Mise sous forme d'un problème de commande optimale	46
2.3.2	Platitude et planification de trajectoire	47
2.3.3	Spécification de la sortie plate par une fonction spline	48
2.3.4	Transformation en un problème de programmation non linéaire	51
2.4	Algorithme en ligne de génération de trajectoire	53
2.4.1	Principe de la planification sur un horizon glissant	53
2.4.2	Mise en œuvre	54
2.5	Applications numériques sur un robot unicycle	58
2.5.1	Exemple 1 : Connaissance complète de la carte	58
2.5.2	Exemple 2 : Connaissance partielle de la carte	60
2.6	Conclusion	61
3	Planification de trajectoire : cadre multi-robots	63
3.1	Introduction	63
3.2	Description du problème dans le cadre multi-robots	64
3.3	Architecture centralisée de planification	66
3.3.1	Mise en œuvre de l'algorithme centralisé	66
3.3.2	Amélioration de l'algorithme centralisé	67
3.4	Architecture décentralisée de planification	70
3.4.1	Description des conflits	70
3.4.2	Principe de l'algorithme décentralisé	72
3.4.3	Mise en œuvre de l'algorithme	74
3.5	Présentation d'autres algorithmes décentralisés	76
3.6	Résultats numériques comparatifs	77
3.6.1	Scénario 1 : 2 robots qui se croisent	78
3.6.2	Scénario 2 : Flottille de 5 robots dans un environnement inconnu	83
3.7	Conclusion	86
II	Poursuite de trajectoire	91
4	Commande par modes glissants	93
4.1	Introduction	93
4.2	Etat de l'art	94
4.2.1	Commande par modes glissants d'ordre un	94

4.2.2	Commande par modes glissants avec action intégrale (CMGI)	99
4.2.3	Commande par modes glissants d'ordre supérieur	101
4.3	Une solution originale pour la CMG d'ordre supérieur	105
4.3.1	Formulation du problème	105
4.3.2	Rappel sur la stabilisation en temps fini d'une chaîne de ρ intégrateurs	107
4.3.3	Synthèse de la commande robuste en temps fini [Defoort et al., 2006b]	110
4.4	Applications de la CMG d'ordre supérieur	113
4.4.1	Commande d'un aéroglissoir [Defoort et al., 2006b]	113
4.4.2	Commande d'un moteur pas-à-pas [Defoort et al., 2006d]	117
4.5	Conclusion	125
5	Stabilisation et suivi de trajectoire pour un robot	129
5.1	Introduction	129
5.2	Poursuite de trajectoires non stationnaires	132
5.2.1	Formulation du problème	132
5.2.2	Synthèse de commande robuste stabilisante	133
5.2.3	Résultats expérimentaux	135
5.3	Approche unifiée pour la poursuite de trajectoire	137
5.3.1	Motivations pour la stabilisation pratique	137
5.3.2	Mise sous forme du “système de Heisenberg”	139
5.3.3	Stabilisation pratique du “système de Heisenberg”	141
5.3.4	Résultats expérimentaux	148
5.4	Conclusion	151
6	Suivi coordonné de trajectoire pour la flottille	153
6.1	Introduction	153
6.2	Formulation du problème de commande collaborative	154
6.2.1	Positionnement “meneur / suiveur”	154
6.2.2	Objectif de commande	156
6.3	Algorithmes coordonnés de suivi de trajectoire	157
6.3.1	CMGI d'ordre un	158
6.3.2	CMGI d'ordre deux	160
6.4	Extension à une architecture à deux meneurs	162
6.5	Résultats expérimentaux	164
6.5.1	Description de la plate-forme Miabot	164
6.5.2	Résultats expérimentaux	165
6.6	Conclusion	167

Conclusion générale et perspectives	171
A Le robot Pekee	177
A.1 Présentation du robot	177
A.2 Communication entre les différents modules du robot	178
A.3 Système d'exploitation temps réel	179
A.4 Linux comme système temps réel	179
A.5 Linux temps réel sur la carte compact-flash de Pekee	180
B Classification des véhicules à roues	183
B.1 Classification et description des roues	183
B.2 Classification des plates-formes mobiles	185
Table des figures	189
Liste des tableaux	193
Index	194
Bibliographie	195

Remerciements

Le travail que nous présentons dans ce mémoire a été effectué au LAGIS sous la direction de Monsieur Wilfrid Perruquetti, Professeur à l'Ecole Centrale de Lille, de Monsieur Thierry Floquet, Chargé de recherche à l'Ecole Centrale de Lille et de Madame Annemarie Kökösy, Enseignant chercheur à l'ISEN.

Je tiens à remercier très vivement Monsieur Wilfrid Perruquetti, Monsieur Thierry Floquet et Madame Annemarie Kökösy pour avoir accompagné mon travail durant ces trois années de thèse. Les judicieux conseils qu'ils m'ont prodigués, leur enthousiasme envers mon travail ainsi que leur grande disponibilité m'ont permis de progresser dans mes études et d'achever ce travail dans les meilleures conditions. Je tiens à leur exprimer toute mon amitié et ma reconnaissance.

Qu'il me soit ensuite permis de remercier très vivement Monsieur le Professeur Philippe Fraisse, Monsieur le Professeur Hugues Mounier et Madame le Professeur Sarah Spurgeon pour l'honneur qu'ils m'ont fait en acceptant d'être les rapporteurs de ce mémoire. Je les remercie de l'intérêt qu'ils ont montré pour mes travaux.

Que Monsieur Michel Fliess, Directeur de Recherche CNRS, reçoive l'expression de ma très vive reconnaissance d'avoir accepté de présider ce jury.

Qu'il me soit permis de remercier Monsieur le Professeur Sergey Drakunov, pour l'honneur qu'il m'a fait en acceptant d'être membre du jury de ma thèse.

C'est avec sympathie que je souhaite témoigner ma reconnaissance à Monsieur Jean-Pierre Richard, Professeur à l'Ecole Centrale de Lille et Monsieur Michel Dambrine, Professeur à l'Université de Valenciennes et du Hainaut-Cambrésis, pour la pertinence de leurs remarques et leurs conseils.

Je tiens à exprimer aussi toute ma gratitude envers tous les membres du LAGIS pour leur sympathie. Ils ont rendu très agréables ces trois années. Je pense particulièrement à Philippe Vanheeghe, Professeur à l'Ecole Centrale de Lille et Directeur du LAGIS, mais aussi Hilaire, Gilles, Bernard,

Jacques, Brigitte, Régine et Patrick.

Bien sûr je souhaite aussi remercier les doctorants ou jeunes docteurs qui sont devenus plus que des collègues de travail. Un grand merci à Romain, Nima, Frédéric, Adrien, Delphine, Wenjuan, Alexandre, François, Emmanuel et tous les autres. Un remerciement particulier à Georges pour notre fructueuse et amicale collaboration.

Je souhaite aussi dire un grand merci à tous mes amis pour leur soutien.

Enfin, je termine ces remerciements par quelques mots plus personnels. Merci à vous Papa, Maman. Merci pour votre soutien, vos encouragements. Et oui, vous n'aviez pas deviné que je serai encore étudiant à 26 ans! Grâce à vous je me suis épanoui. Merci aussi à ma soeur Sandrine pour ses encouragements de tous les jours.

Notations

Notations générales :

- \mathbb{R} : ensemble des nombres réels.
- \mathbb{R}^+ : ensemble des nombres réels positifs ou nuls.
- \mathbb{R}^{+*} : ensemble des nombres réels strictement positifs.
- \mathbb{R}^n : espace vectoriel de dimension n construit sur le corps des réels.
- \mathbb{N} : ensemble des nombres entiers naturels.
- \mathcal{V} : voisinage non vide de l'origine dans \mathbb{R}^n .
- $[a, b]$: intervalle fermé de \mathbb{R} d'extrémités a et b .
- (a, b) : intervalle ouvert de \mathbb{R} d'extrémités a et b .
- $[a, b)$: intervalle semi-ouvert de \mathbb{R} d'extrémités a et b .
- $C^0(E, F)$: ensemble des fonctions continues de E dans F .
- $C^k(E, F)$: ensemble des fonctions de classe k de E dans F .
- $\mathcal{R}_O = (O, \vec{i}, \vec{j}, \vec{k})$: repère fixe de l'espace lié au point O et de base $(\vec{i}, \vec{j}, \vec{k})$.
- $t \in \mathbb{R}$: variable temporelle.
- $t_{initial} \in \mathbb{R}$: instant initial fixé. Pour les systèmes stationnaires $t_{initial} = 0$.
- $t_{final} \in \mathbb{R}$: instant final.
- $\dot{x} = \frac{dx}{dt}$: dérivée de la variable x par rapport au temps.
- $\ddot{x} = \frac{d^2x}{dt^2}$: seconde dérivée de la variable x par rapport au temps.
- $x^{(j)} = \frac{d^jx}{dt^j}$: $j^{\text{ème}}$ dérivée de la variable x par rapport au temps.
- x^T : transposé du vecteur x .
- $x \in \mathbb{R}^n$: vecteur de composantes x_j .
- $\cdot \wedge \cdot$: produit vectoriel de deux champs de vecteurs.
- $|.|$: valeur absolue d'un nombre réel.
- $\|.\|$: norme euclidienne sur \mathbb{R}^n .
- $\det(A)$: déterminant de la matrice A .
- $\|A\|$: norme euclidienne de la matrice A .
- I_n : matrice identité de $\mathbb{R}^{n \times n}$.
- 0_n : matrice nulle de $\mathbb{R}^{n \times n}$.

- Soient $f(x)$ et $g(x)$ des champs de vecteurs de dimension n , suffisamment différentiables. Le *produit de Lie* ou *crochet de Lie* de $f(x)$ et $g(x)$ et l'opérateur ad sont définis par :

$$\begin{aligned}[f, g](x) &= \frac{\partial g}{\partial x}(x)f(x) - \frac{\partial f}{\partial x}(x)g(x) \\ ad_f^k g(x) &= [f, ad_f^{k-1}g](x), \quad k \geq 1\end{aligned}$$

avec $ad_f^0 g(x) = g(x)$.

- Soit une fonction réelle différentiable $\lambda(x)$. En notant $d\lambda(x) = \left(\frac{\partial \lambda}{\partial x_1}, \dots, \frac{\partial \lambda}{\partial x_n}\right)$, la dérivée de λ le long de f est donnée par :

$$\begin{aligned}L_f \lambda(x) &= d\lambda(x)f(x) = \sum_{i=1}^n \frac{\partial \lambda}{\partial x_i}(x)f_i(x) \\ L_f^k \lambda(x) &= dL_f^{k-1} \lambda(x).f(x)\end{aligned}$$

avec $L_f^0 \lambda(x) = \lambda(x)$.

- $\text{sign}(a)$: fonction signe réelle définie par :

$$\text{sign}(a) = \begin{cases} -1 & \text{si } a < 0 \\ 1 & \text{si } a > 0 \end{cases}$$

Par extension, $\text{sign}(x)$ où $x = [x_1, \dots, x_n]^T \in \mathbb{R}^n$ sera le vecteur de composantes $\text{sign}(x_j)$.

- $\text{Vect}\{g_1, \dots, g_m\}$: distribution engendrée par les champs de vecteurs g_1, \dots, g_m .

Notations spécifiques à la flottille :

- $N_a \in \mathbb{N}$: nombre de robots de la flottille page 64
- $\mathcal{N} \in \mathbb{N}$: ensemble des robots composant la flottille page 64
- $\mathcal{A} \subset \mathcal{N} \times \mathcal{N}$: ensemble des couples de robots qui échangent des informations page 65

Notations spécifiques au robot d'indice i :

- $\mathcal{Q}_i \subset \mathbb{R}^n$: espace des configurations page 27
- $\mathcal{U}_i \subset \mathbb{R}^m$: ensemble des commandes admissibles page 44
- $q_i \in \mathcal{Q}_i$: vecteur d'état page 42
- $u_i \in \mathbb{R}^m$: vecteur de vitesse page 42
- $f_i : \mathbb{R}^n \times \mathbb{R}^m \rightarrow \mathbb{R}^n$: application décrivant le modèle cinématique du robot page 42
- $z_i \in \mathbb{R}^m$ sortie plate page 42
- $l_i \in \mathbb{N}$ ($l_i < n$) : nombre de fois qu'il faut dériver z_i pour exprimer q_i et u_i en fonction de la sortie plate et de ses dérivées page 43
- $(x_i, y_i) \in \mathbb{R}^2$: coordonnées du milieu de l'axe des roues motrices page 44
- $\rho_i \in \mathbb{R}^+$: rayon de la forme géométrique du robot page 44
- $d_{i,com} \in \mathbb{R}^+$: portée de diffusion des informations du robot i page 65

Notations spécifiques à la planification :

- $T_p \in \mathbb{R}^+$: horizon de planification page 46
- $T_c \in \mathbb{R}^+$ ($T_c \leq T_p$) : horizon de calculs page 54
- $T_d \in \mathbb{R}^+$ ($T_d \geq T_p$) : horizon d'intuition page 72
- $\tau_k \in [t_{initial}, t_{final}]$: instant de mise à jour des problèmes d'optimisation page 54
- $q_{i,ref}(t, \tau_k)$: trajectoire optimale de référence commençant à l'instant τ_k page 55
- $u_{i,ref}(t, \tau_k)$: vitesse optimale de référence commençant à l'instant τ_k page 55
- $q_i(t, \tau_k)$: trajectoire que l'on cherche à optimiser commençant à l'instant τ_k page 55
- $u_i(t, \tau_k)$: vitesse que l'on cherche à optimiser commençant à l'instant τ_k page 55
- $\widehat{q}_i(t, \tau_k)$: trajectoire intuitive commençant à l'instant τ_k page 72
- $\widehat{u}_i(t, \tau_k)$: vitesse intuitive commençant à l'instant τ_k page 73
- $q_{i,initial} \in \mathcal{Q}_i$: configuration initiale page 44
- $q_{i,final} \in \mathcal{Q}_i$: configuration finale page 44
- $u_{i,initial} \in \mathcal{U}_i$: vitesse initiale page 44
- $u_{i,final} \in \mathcal{U}_i$: vitesse finale page 44
- $\mathcal{D}_i(\tau_k) \subset \mathcal{Q}_i$: zone de détection du robot déterminée à l'instant τ_k page 45
- $\mathcal{O}_i(\tau_k) \subset \mathcal{D}_i(\tau_k)$: zone obstacle déterminée à l'instant τ_k page 45
- $M_i \in \mathbb{N}$: nombre de cercles nécessaires pour recouvrir la zone obstacle $\mathcal{O}_i(\tau_k)$ page 45
- $O_{m_i} \in \mathcal{O}_i(\tau_k)$: obstacle d'indice m_i statique et circulaire de centre (X_{m_i}, Y_{m_i}) et de rayon r_{m_i} page 46
- $d_{i,O_{m_i}}(\tau_k) \in \mathbb{R}^+$: distance entre le robot et l'obstacle O_{m_i} à l'instant τ_k page 46
- J : fonctionnelle à minimiser page 46
- $\{\text{nœud}_0 \leq \dots \leq \text{nœud}_M\}$: suite de nœuds page 49
- $B_{j,d}(t) \in \mathbb{R}^+$: $j^{\text{ième}}$ B-spline d'ordre d page 49
- $n_{\text{knot}} \in \mathbb{N}$: nombre d'intervalles de longueur non nulle du vecteur de noeuds page 50
- $\{\delta_0 = t_{initial} < \delta_1 < \dots < \delta_{n_{\text{knot}}} = t_{final}\}$: subdivision uniforme de $[t_{initial}, t_{final}]$ page 51
- $C_{j,i} \in \mathbb{R}^m$: vecteurs contenant les points de contrôle de la fonction spline page 52
- $N_{ech} \in \mathbb{N}$: nombre d'échantillons pour la discréttisation page 52
- $d_{ip}(\tau_k) \in \mathbb{R}^+$: distance entre les trajectoires planifiées des robots i et p à l'instant τ_k page 65
- $d_{\text{sécurité}} \in \mathbb{R}^+$: distance de sécurité pour éviter la collision entre deux robots page 65
- $\mathcal{R}_i(\tau_k) \subset \mathcal{Q}_i$: zone d'accessibilité maximale du robot page 70
- $\mathcal{C}_{i,collision}(\tau_k) \subset \mathcal{N}$: ensemble des robots risquant de provoquer une collision avec le robot i sur l'horizon T_p page 70
- $\mathcal{C}_{i,com}(\tau_k) \subset \mathcal{N}$: ensemble des robots risquant de provoquer une rupture de communication avec le robot i sur l'horizon T_p page 70
- $\mathcal{C}_i(\tau_k) \subset \mathcal{N}$: ensemble des robots risquant de provoquer un conflit avec le robot i page 71
- $\xi \in \mathbb{R}^+$: déformation admissible entre la trajectoire intuitive

et la trajectoire optimale planifiée

page 74

Acronymes

- CMG : commande par modes glissants
- CMGI : commande par modes glissants avec action intégrale
- OS : système d'exploitation

Introduction générale

Ce travail de doctorat a été préparé au sein de l'équipe SyNeR¹ (Systèmes Non linéaires et à Retards) du Laboratoire d'Automatique, Génie Informatique et Signal (LAGIS, UMR CNRS 8146). Mon travail de recherche s'inscrit dans le cadre du projet ROBOCOOP², soutenu par le Conseil Régional Nord-Pas de Calais et l'Union Européenne. Ce projet concerne le développement de robots autonomes collaboratifs. Un robot autonome est un système automoteur, disposant à la fois de moyens de traitement de l'information permettant une capacité décisionnelle suffisante et de moyens matériels adaptés, de façon à pouvoir exécuter, sous contrôle humain réduit, un certain nombre de tâches précises, dans un environnement variable inconnu. Ces robots autonomes devront réaliser une action commune. Cette collaboration suppose un partage de tâches et d'informations, ce qui implique la présence d'un réseau de communication. Les objectifs principaux du projet ROBOCOOP sont de proposer et de mettre en œuvre des outils pour la modélisation, l'analyse et la synthèse de commandes spécifiques à la coopération de robots distants.

Motivations

Enjeux

Au-delà de l'intérêt scientifique propre de l'objectif affiché de faire coopérer des robots interagissant rationnellement avec leur environnement et des problèmes de recherche fondamentaux qu'un tel objectif nous conduit à aborder, la problématique de ce projet couvre des enjeux sociaux et économiques importants et profonds. La robotique autonome n'en est que la partie apparente. De nombreux autres domaines applicatifs sont naturellement concernés par de telles recherches. Aujourd'hui, les robots mobiles deviennent de plus en plus complexes, intègrent des capacités de perception, de communication et d'adaptation à divers domaines de fonctionnement et visent des exigences toujours plus grandes de robustesse, d'ergonomie et de sécurité. De plus, plusieurs agents ont la possibilité de résoudre plus efficacement et plus rapidement une mission.

Certains des enjeux de notre problématique portent sur des applications de la robotique socialement utiles (voir par exemple [Fraisse et al., 2007]). Il s'agit en premier lieu de robots dans des environne-

¹Le site de l'équipe SyNeR est disponible sur <http://syner.free.fr/>

²Le site du projet ROBOCOOP est disponible sur <http://syner.ec-lille.fr/robocoop/>

ments hostiles réalisant des tâches dangereuses ou très pénibles pour l'homme. La robotique d'exploration planétaire ou la robotique sous-marine en sont des exemples typiques [Rekleitis et al., 2000]. La robotique minière, certaines applications de la robotique de chantier (e.g. le transport d'objets encombrants ou lourds à l'aide de plusieurs véhicules autonomes) [Miyata et al., 2002], ou la robotique d'exploration de zones dangereuses (e.g. le déminage, l'intervention en milieu radioactif) relèvent également de ce type d'applications. On peut également citer des applications environnementales, où des formations de drones permettraient une surveillance en temps continu de zones à risque (incendies, pollution) [Inalhan et al., 2002]. La robotique de service (manutention, nettoyage, surveillance) couvre certains besoins non traités aujourd'hui car difficilement réalisables par des opérateurs humains. Enfin, la robotique chirurgicale télé-opérée permettrait à plusieurs chirurgiens de pouvoir intervenir à distance sur un même patient au moyen d'interfaces robotisées et par l'intermédiaire d'un réseau. D'autres applications, ayant généralement des effets néfastes sur la société, peuvent être envisagées sur le plan militaire notamment au niveau de la collecte d'informations ennemis et lors des phases offensives.

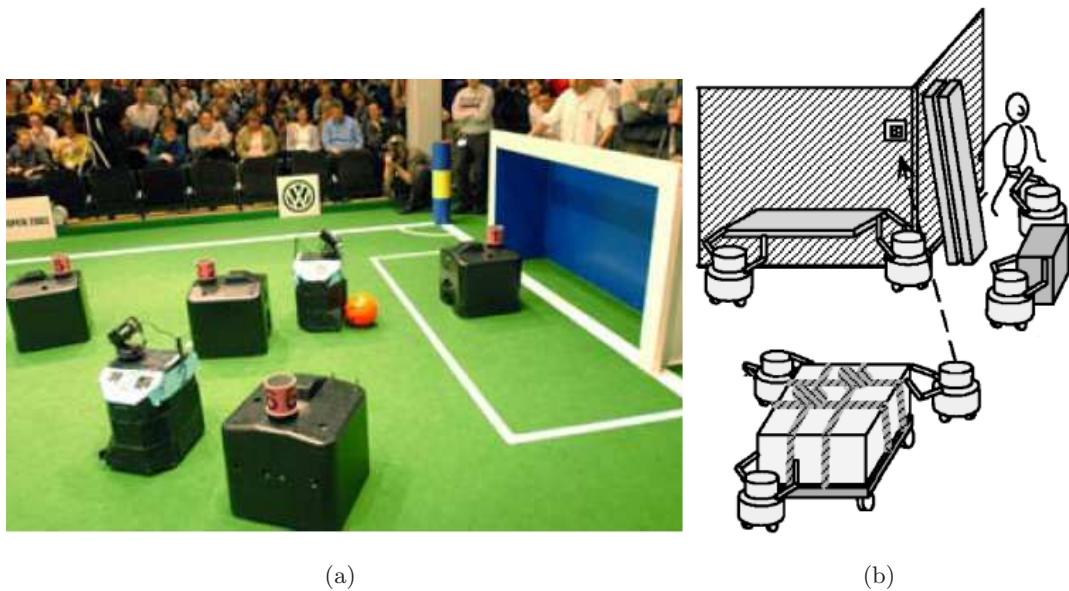


FIG. 1 – Exemples d'application de commande coordonnée de robots mobiles (a) jeu d'équipe [Kim et al., 2004] (b) transport coopératif [Miyata et al., 2002].

Nouveaux challenges techniques

Le problème de la coopération entre différentes entités autour d'un objectif commun constitue une problématique scientifique très intéressante. Elle offre un champ privilégié de développement et d'expérimentation de concepts et de méthodes relevant de l'informatique, de l'automatique et du traitement du signal. De par la confrontation aux contraintes et à la complexité de l'interaction avec

un environnement concret et non modélisable complètement, la coopération entre des véhicules autonomes est un défi pour ces sciences de l'ingénieur. Elle fait découvrir des problèmes essentiels dans ces disciplines et à leur intersection (e.g. planification et commande, modélisation multi-agents, traitement de l'information, protocoles de communication, logique et décision, ordonnancement, théorie des graphes, ...). Enfin, elle permet des validations probantes des différentes approches.

Dans ce mémoire, nous nous attacherons à développer des outils permettant la navigation coopérative autonome d'une formation de robots mobiles dans un environnement complexe.

Caractéristiques de notre approche

Les spécificités de notre approche sont de deux types : celles liées aux systèmes auxquels on s'intéresse et les celles liées aux applications.

Les spécificités liées au système sont :

- on s'intéresse dans notre étude aux robots mobiles à roues,
- le système est soumis à des contraintes cinématiques,
- il est également soumis à des perturbations et des incertitudes (dues, par exemple, aux erreurs de modélisation, aux phénomènes de dérapage et de glissement des roues du robot sur le sol, aux frottements, etc.),
- la puissance disponible à bord des véhicules est limitée,
- la portée de diffusion des informations entre deux robots est limitée et dépend de l'antenne d'émission et de la puissance embarquée.

Ainsi le type de système auquel on s'intéresse se range dans la classe des véhicules non holonomes, tels les robots Pekee et les robots Miabot (figure 2) disponibles aux LAGIS et utilisés dans les expérimentations. En effet, la condition de roulement sans glissement des roues sur le sol a pour effet de contraindre le véhicule à se déplacer tangentiellement à sa direction principale. Cette contrainte non intégrable, réduisant l'ensemble des vitesses accessibles à chaque instant, est connue sous le nom de contrainte de non holonomie. Les problématiques associées à de tels systèmes sont fondamentalement différentes de celles associées à des robots de type omni-directionnel (ils ont une totale mobilité dans le plan, c'est-à-dire qu'ils peuvent bouger dans n'importe quelle direction sans aucune réorientation à chaque instant). Cette caractéristique implique qu'une trajectoire quelconque n'est pas nécessairement admissible pour le type de système auquel on s'intéresse. Par exemple, un robot de type unicycle ne peut pas effectuer de déplacements latéraux sans exécuter des manœuvres. Par conséquent, la contrainte de non holonomie doit être prise en compte dans la génération de trajectoire, qui plus est lorsque l'environnement dans lequel évolue le robot est également contraint par des obstacles.

Une autre raison rendant la tâche de navigation autonome difficile est la présence de perturbations et d'incertitudes qui induisent un écart entre la trajectoire planifiée et la trajectoire réellement effectuée. En boucle ouverte, cette dérive est incontournable. Par conséquent, il est nécessaire de

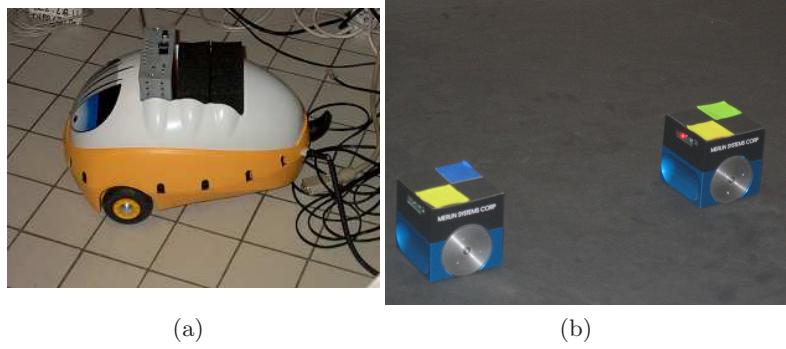


FIG. 2 – A gauche, le robot Pekee. A droite, le robot Miabot.

réduire son effet en utilisant les informations fournies par les capteurs pour localiser le véhicule.

Pour ces principales raisons, le problème du déplacement des robots mobiles est généralement traité en deux étapes. La première consiste à planifier une trajectoire admissible, c'est-à-dire satisfaisant les contraintes de non holonomie. La seconde consiste à élaborer des stratégies de commande robuste en boucle fermée garantissant un suivi précis de la trajectoire planifiée en boucle ouverte. Cette approche en deux étapes diffère de l'approche purement réactive dans laquelle le robot utilise un système de perception lui fournissant des informations sur l'environnement, à partir desquelles il définit, à chaque instant, son déplacement immédiat de manière dynamique [Arkin, 1998], [Balch et Arkin, 1998].

Le problème s'annonce encore plus délicat lorsque l'on envisage le déplacement simultané de plusieurs robots dans le même espace de travail. En effet, aux deux problèmes précédents s'ajoute le problème de la coopération multi-robots.

Les spécificités de notre approche sont également liées aux applications envisagées :

- les obstacles, ici supposés statiques, ne sont pas nécessairement connus à l'avance. Ils peuvent être détectés au fur et à mesure que les robots se déplacent dans leur environnement,
- les distances entre certains robots sont naturellement bornées afin d'éviter les collisions et les ruptures de communication,
- les robots échangent des informations sur leur position et leurs intentions,
- selon la stratégie choisie, la résolution des conflits est réalisée soit par un superviseur soit individuellement par chaque robot,
- on peut être amené à effectuer des manœuvres,
- l'asservissement doit être performant (précision, rapidité, stabilité, robustesse vis-à-vis d'incertitudes paramétriques et de perturbations ...).

Les robots doivent avoir ainsi la capacité de calculer ou de mettre à jour leur trajectoire en temps réel tant que la mission n'est pas achevée, de manière centralisée ou non. Le cadre général de notre travail est donc celui de la navigation d'une formation de “véhicules intelligents”, mais nous mettons

l'accent sur le fait que les spécificités de notre approche entraînent des problématiques singulières et requièrent des méthodes de résolutions originales.

Nos contributions

Nos contributions portent sur la résolution de certaines fonctionnalités de la navigation autonome pour une flottille de robots mobiles. Le diagramme, représenté figure 3, résume les différentes fonctionnalités nécessaires à la navigation autonome et permet de situer nos contributions, représentées en gras, qui résident dans le développement de méthodes originales de résolutions pour la planification et le suivi de trajectoire, et dans l'intégration et l'implémentation de l'ensemble.

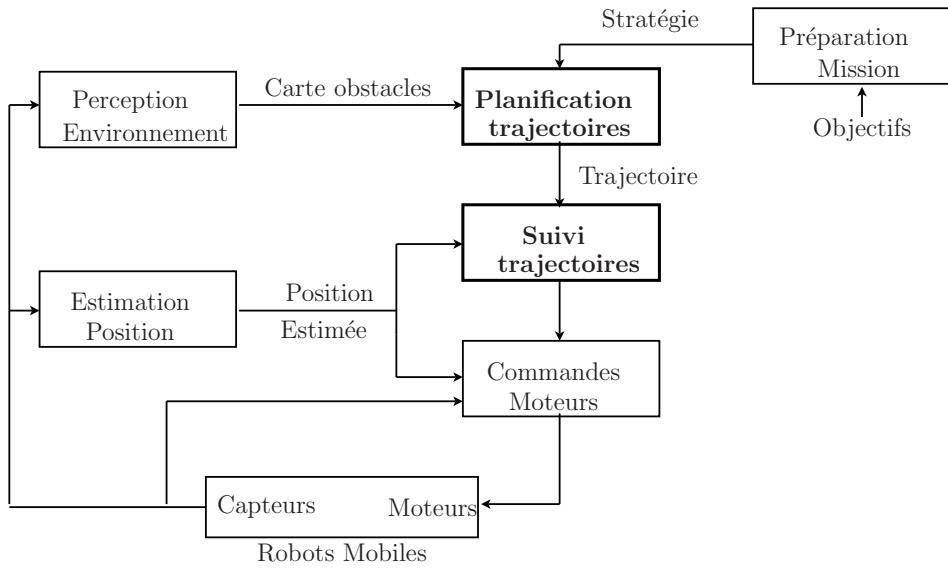


FIG. 3 – Fonctionnalités de la navigation autonome. Nos contributions théoriques sont représentées en gras.

Notre méthodologie repose sur deux principes. Tout d'abord, la volonté de développer des méthodes génériques qui s'appliquent à une classe de systèmes plutôt qu'à un robot en particulier. Une conséquence intéressante est que des méthodes développées pour résoudre un problème dans un contexte applicatif particulier trouvent parfois des applications dans des domaines très différents. Le second principe est l'implémentation sur des systèmes réels, afin de se confronter à des problèmes réels.

Organisation du mémoire de thèse

Le Chapitre 1 donne un rapide tour d'horizon de la question de la planification et de la commande pour une flottille de robots mobiles. Nous y proposons également une introduction à la robotique mobile, dont certains aspects méritent quelques précisions.

Nous avons choisi de diviser le reste de ce mémoire en deux parties. La première, développée dans les Chapitres 2 et 3, est consacrée à la planification de trajectoire et la deuxième, développée dans les Chapitres 4, 5 et 6, à la poursuite robuste de trajectoire (commande en boucle fermée).

Le Chapitre 2 de ce mémoire introduit le problème de planification pour un robot mobile. Différents concepts tels que la commande optimale sous contraintes, la platitude, les fonctions splines, la commande sur un horizon glissant y sont développés. A partir de ces concepts, un algorithme de génération en ligne de trajectoire est présenté et validé expérimentalement sur un robot mobile.

Le Chapitre 3 est un prolongement du Chapitre 2 dans le cadre d'une planification de plusieurs robots mobiles coopératifs. Après avoir décrit le problème, deux algorithmes y sont détaillés. Le premier est un algorithme centralisé qui nécessite l'utilisation d'un superviseur et peut être vu comme une extension directe du cas d'un seul robot. Le second est un algorithme décentralisé dans lequel chaque robot planifie uniquement sa propre trajectoire à partir d'informations locales disponibles. Ces deux algorithmes sont testés expérimentalement et comparés avec d'autres méthodes existantes.

Le Chapitre 4 donne une présentation générale des concepts de base de la commande par modes glissants d'ordre supérieur. Les propriétés de robustesse, le phénomène de chattering et les solutions pour l'éliminer y sont présentés. Un nouvel algorithme de commande par modes glissants d'ordre quelconque basé sur une extension dynamique du système est proposé pour des systèmes non linéaires incertains. Cet algorithme permet de garantir l'établissement d'un régime glissant d'ordre quelconque avec un réglage simple des paramètres de la loi de commande en vue d'atteindre les performances désirées. L'efficacité de cette stratégie de commande est illustrée par un exemple académique de commande d'un aéroglissoir et par des tests expérimentaux pour la commande par modes glissants d'un moteur pas-à-pas.

Le Chapitre 5 présente des commandes par modes glissants permettant de stabiliser les erreurs de suivi de trajectoire d'un robot mobile. Dans une première partie, une loi de commande discontinue en vitesse est proposée afin de stabiliser asymptotiquement les erreurs de suivi sous certaines conditions. Puis, d'autres commandes, qui tiennent compte des dynamiques des actionneurs, sont développées afin de garantir une stabilisation pratique des erreurs de suivi malgré la présence de perturbations sur le modèle. Pour réaliser cet objectif, le système est mis sous la forme dite du "système de Heisenberg" avec ajout d'intégrateurs dans les chaînes d'entrée. Ces algorithmes sont illustrés par le biais de résultats expérimentaux.

Dans le chapitre 6, nous proposons, dans le cadre de la poursuite de trajectoire pour une flottille de robots mobiles, un mécanisme décentralisé de coordination de type "meneur / suiveur". Permettant de

s'affranchir de la connaissance de la position de l'ensemble des robots par rapport à un repère fixe, il assure la stabilisation de la flottille dans une certaine configuration ainsi que l'évitement de collisions inter-robots depuis l'instant initial.

Ce mémoire se termine par une conclusion générale où l'accent sera mis sur la contribution des techniques proposées quant à la résolution des problèmes de planification et de commande par modes glissants appliqués à une formation de robots mobiles, et sur les travaux s'inscrivant dans le prolongement de cette thèse et à mener ultérieurement.

Chapitre 1

Problématique et état de l'art

1.1 Introduction

La recherche d’algorithmes de planification et de stratégies de commande pour des véhicules non holonomes constitue aujourd’hui l’un des principaux axes de recherche de la robotique moderne. Plusieurs raisons contribuent à cet engouement. La première est que les véhicules sur roues constituent de nos jours le moyen de transport individuel principal. Leur automatisation, précédemment limitée aux expérimentations en laboratoire, est maintenant envisagée pour des applications grand public (convois de véhicules sur autoroute, systèmes de transport urbain intelligent, etc.). Ces nouvelles applications, qui nécessitent de coordonner les mouvements de plusieurs véhicules, donnent lieu à de nouveaux problèmes d’automatique. Une autre raison plus technique tient au fait que les équations régissant le déplacement des véhicules non holonomes revêtent un intérêt théorique particulier dans le domaine de l’automatique non linéaire. L’objectif de ce chapitre est de présenter un certain nombre de caractéristiques spécifiques à la planification et à la commande de ces systèmes. Nous nous intéresserons dans un premier temps à la modélisation de tels systèmes. Puis nous rappellerons diverses méthodes de planification et de synthèse de commandes.

1.2 Modélisation de la cinématique des véhicules à roues

1.2.1 Hypothèses de modélisation

La problématique de la commande des robots mobiles étant trop vaste pour pouvoir être présentée de façon exhaustive, nous introduisons dans cette partie un certain nombre d’hypothèses simplificatrices :

- les véhicules sont considérés comme rigides et évoluant sur un plan,
- les véhicules sont dotés de roues conventionnelles : le point de contact entre la roue et le sol est réduit à un point I et la roue est soumise à la contrainte de roulement sans glissement.

1.2.2 Roulement sans glissement et non holonomie

Beaucoup de systèmes mécaniques sont sujets à des contraintes de position et/ou de vitesse, c'est-à-dire que plusieurs relations entre les positions et/ou les vitesses des différents points du système doivent être satisfaites pendant tout le mouvement. Ces contraintes sont dites holonomes s'il est possible de les intégrer et elles aboutissent à des relations algébriques liant les paramètres de configuration. Ces relations peuvent être éliminées par un changement de variables approprié et le système est dit **holonome**. Dans le cas de contraintes non intégrables, l'élimination n'est plus possible et le système est dit **non holonome**.

Dans le cas des véhicules sur roues, ces contraintes cinématiques résultent de l'hypothèse de roulement sans glissement. Considérons une roue verticale qui roule sans glisser sur un sol plan (voir figure 1.1). Le roulement sans glissement se traduit par la vitesse nulle du point I de la roue en contact avec

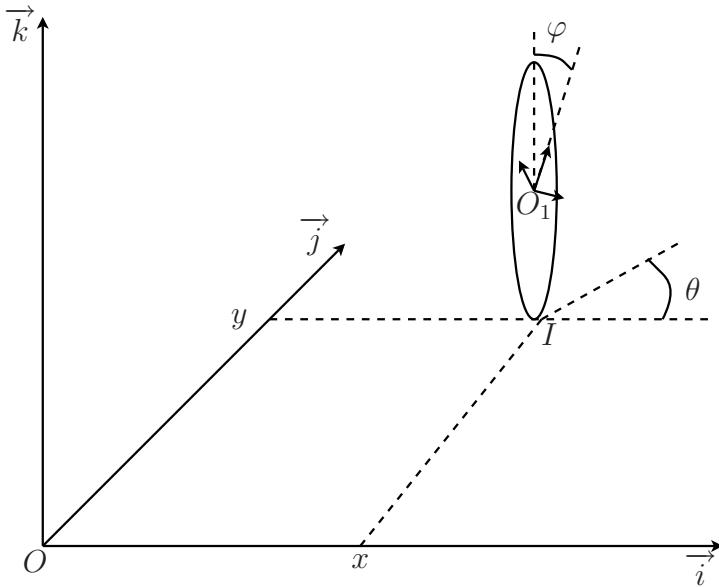


FIG. 1.1 – Description d'une roue.

le sol. Avec les notations de la figure 1.1, on obtient :

$$\begin{aligned}\vec{V}(I/\mathcal{R}_O) &= \dot{x}\vec{i} + \dot{y}\vec{j} + \left(\dot{\theta}\vec{k} + \dot{\varphi}(-\sin\theta\vec{i} + \cos\theta\vec{j})\right) \wedge (-r\vec{k}) \\ &= (\dot{x} - r\dot{\varphi}\cos\theta)\vec{i} + (\dot{y} - r\dot{\varphi}\sin\theta)\vec{j} \\ &= 0\end{aligned}$$

où r est le rayon de la roue et (x, y) est la coordonnée du point O_1 dans le repère fixe $\mathcal{R}_O = (O, \vec{i}, \vec{j}, \vec{k})$. On en déduit deux contraintes :

$$\begin{cases} \dot{x} - r\dot{\varphi}\cos\theta = 0 \\ \dot{y} - r\dot{\varphi}\sin\theta = 0 \end{cases} \quad (1.2.1)$$

qui peuvent se réécrire de la manière suivante :

$$\begin{cases} \dot{x} \cos \theta + \dot{y} \sin \theta &= r\dot{\phi} \\ -\dot{x} \sin \theta + \dot{y} \cos \theta &= 0 \end{cases}$$

Il est intéressant de noter qu'en introduisant $v = r\dot{\phi}$ la vitesse de roulement de la roue et $w = \dot{\theta}$ sa vitesse de rotation autour de l'axe \vec{k} , on forme le modèle suivant :

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos \theta & 0 \\ \sin \theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ w \end{bmatrix} \quad (1.2.2)$$

Ce modèle non linéaire, possède, du point de vue de la commande, deux propriétés fondamentales.

Propriété 1.1 *Le linéarisé du système non linéaire (1.2.2) autour d'un point d'équilibre n'est pas commandable.*

Démonstration.

L'approximation linéaire du système (1.2.2) autour d'un état d'équilibre (x_0, y_0, θ_0) , est donné par :

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos \theta_0 & 0 \\ \sin \theta_0 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ w \end{bmatrix}$$

Le rang de la matrice de commandabilité étant inférieur à la dimension de l'état du système (1.2.2), le critère de commandabilité de Kalman n'est pas vérifié et le linéarisé n'est donc pas commandable.

■

Propriété 1.2 *le système non linéaire (1.2.2) est commandable.*

Démonstration.

Le système (1.2.2) peut se noter sous la forme :

$$\dot{q} = g_1(q)v + g_2(q)w \quad (1.2.3)$$

avec $q = [x, y, \theta]^T$, $g_1(q) = [\cos \theta, \sin \theta, 0]^T$ et $g_2(q) = [0, 0, 1]^T$. Le résultat suivant donne alors une condition suffisante pour que le système (1.2.3) soit commandable (voir [Nijmeijer et Schaft, 1991] par exemple).

Théorème 1.3 *Considérons le système*

$$\dot{q} = \sum_{j=1}^m g_j(q)u_j \quad (1.2.4)$$

où $q \in \mathbb{R}^n$, u_j ($j = 1, \dots, m$) sont des variables de commande et $g_j(q)$ ($j = 1, \dots, m$) sont des champs de vecteurs différentiables sur \mathbb{R}^n .

Soit l'algèbre d'accessibilité notée $Lie(g_1, \dots, g_m)$ définie comme la plus petite sous-algèbre de champs de vecteurs analytiques définis sur \mathbb{R}^n qui contienne les champs de vecteurs engendrés par $\{g_1, \dots, g_m\}$:

$$Lie(g_1, \dots, g_m) = \{[h_k, [h_{k-1}, [\dots, [h_2, h_1]]]], \quad h_j \in \{g_1, \dots, g_m\}, \quad j = 1 \dots k, \quad k = 0 \dots \infty\}$$

où $[h_i, h_j] = \frac{\partial h_j}{\partial q} h_i - \frac{\partial h_i}{\partial q} h_j$. Pour $q \in \mathbb{R}^n$, la distribution d'accessibilité A_c est la distribution engendrée par $Lie(g_1, \dots, g_m)$:

$$A_c(q) = \text{vect} \{h(x), \quad h \in Lie(g_1, \dots, g_m)\}$$

Le système (1.2.4) est commandable si :

$$\dim(A_c(q)) = n, \quad \forall q \in \mathbb{R}^n$$

La propriété 1.2 est un corollaire immédiat de ce théorème. En effet, il suffit de vérifier qu'en tout point q , les vecteurs $g_1(q)$, $g_2(q)$, et $[g_1, g_2](q)$ forment une base de \mathbb{R}^3 . ■

1.2.3 Modélisation des robots à roues

Généralités

La modélisation précédente peut s'étendre à tous les véhicules sur roues. Etant donné un espace de dimension n correspondant à l'espace des configurations du système initial, en éliminant l'ensemble des contraintes complètement intégrables du système, on réduit l'espace initial à une sous-variété :

$$\dot{q} = \sum_{i=1}^m g_i(q) u_i \tag{1.2.5}$$

où $q \in \mathbb{R}^n$, $m < n$, u_i ($i = 1, \dots, m$) sont des variables de commande et $g_i(q)$ ($i = 1, \dots, m$) sont des champs de vecteurs différentiables du système. ($m - n$) est le nombre de contraintes cinématiques non intégrables. Le système (1.2.5) est une généralisation du modèle cinématique (1.2.2) de la roue soumise aux contraintes de roulement sans glissement. Une des caractéristiques de ces systèmes non holonomes est que le linéarisé n'est pas commandable alors que le système réel l'est [Campion et al., 1996].

Par la suite, nous allons nous focaliser sur deux types de robots mobiles non holonomes couvrant la majeure partie des robots mobiles existants¹.

Le robot de type unicycle

On considère le robot mobile de *type unicycle* schématisé figure 1.2. Ce robot est équipé de deux roues fixes motrices commandées indépendamment et de roues folles assurant sa stabilité.

¹Pour plus de détails sur les modèles de robots existants, le lecteur peut se référer au chapitre 7 de [Canudas-De-Wit et al., 1996] ainsi qu'à l'Annexe B.

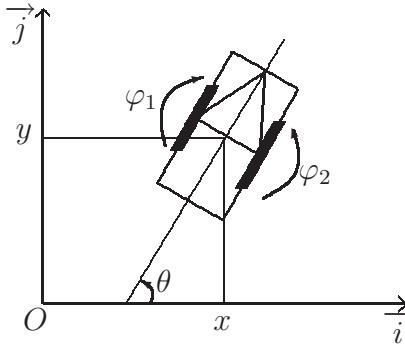


FIG. 1.2 – Robot de type unicycle.

Soit l'abscisse et l'ordonnée (x, y) du milieu de l'axe des deux roues motrices, θ l'orientation du robot, r le rayon des roues et $2R$ la distance entre les deux roues motrices. En reprenant les équations (1.2.1), on montre aisément que les contraintes de roulement sans glissement de chacune des roues commandées s'écrivent :

– pour la roue gauche

$$\dot{x} - R\dot{\theta} \cos \theta - r\dot{\varphi}_1 \cos \theta = 0 \quad (1.2.6)$$

$$\dot{y} - R\dot{\theta} \sin \theta - r\dot{\varphi}_1 \sin \theta = 0 \quad (1.2.7)$$

– pour la roue droite

$$\dot{x} + R\dot{\theta} \cos \theta - r\dot{\varphi}_2 \cos \theta = 0 \quad (1.2.8)$$

$$\dot{y} + R\dot{\theta} \sin \theta - r\dot{\varphi}_2 \sin \theta = 0 \quad (1.2.9)$$

Ces quatre contraintes ne sont pas indépendantes puisque la différence des membres de gauche des égalités (1.2.6) et (1.2.8) est proportionnelle à celle associée à (1.2.7) et (1.2.9). On peut donc omettre, par exemple, la dernière contrainte. En outre, une contrainte est complètement intégrable. En effet, on a :

$$\begin{cases} \dot{x} \cos \theta + \dot{y} \sin \theta - R\dot{\theta} = r\dot{\varphi}_1 \\ \dot{x} \cos \theta + \dot{y} \sin \theta + R\dot{\theta} = r\dot{\varphi}_2 \end{cases}$$

Ainsi, obtient-on :

$$2R\dot{\theta} = r(\dot{\varphi}_2 - \dot{\varphi}_1)$$

Ce qui implique :

$$2R\theta = r(\varphi_2 - \varphi_1) + \text{constante}$$

Par conséquent, il ne reste plus que deux contraintes indépendantes, (1.2.6)-(1.2.7), dont une écriture équivalente est :

$$\begin{cases} \dot{x} = v \cos \theta \\ \dot{y} = v \sin \theta \\ \dot{\theta} = w \end{cases} \quad (1.2.10)$$

où

- la vitesse linéaire du robot est

$$v = \frac{r}{2}(\dot{\varphi}_1 + \dot{\varphi}_2) \quad (1.2.11)$$

- la vitesse angulaire du robot est

$$w = \frac{r}{2R}(\dot{\varphi}_2 - \dot{\varphi}_1) \quad (1.2.12)$$

Le fait que ce système soit le même que le modèle (1.2.2) obtenu pour une roue unique, justifie le qualificatif unicycle souvent employé dans la littérature.

Le robot de type voiture

Du point de vue du conducteur, un robot de *type voiture* possède deux commandes : l'accélération et la direction. Prenons comme point de référence (x, y) le milieu de l'axe des roues de l'essieu arrière où se trouve les deux roues motrices (voir figure 1.3). Ici, on introduit la notion de roue directrice centrale. Cette roue, virtuelle dans le cas d'une voiture, correspond à la roue directrice d'un tricycle équivalent. Son introduction permet de simplifier les équations en faisant abstraction du mécanisme de couplage des roues directrices servant à respecter les contraintes de roulement sans glissement et en ne considérant qu'un seul angle de direction.

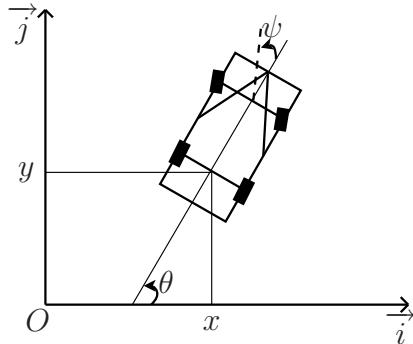


FIG. 1.3 – Robot de type voiture.

Le modèle cinématique correspondant est :

$$\begin{cases} \dot{x} &= v \cos \theta \\ \dot{y} &= v \sin \theta \\ \dot{\theta} &= v \frac{\tan \psi}{L} \\ \dot{\psi} &= w_\psi \end{cases} \quad (1.2.13)$$

où L est la distance entre les axes des roues avant et arrière, ψ est l'angle de braquage formé par les roues avant et l'axe principal de la voiture, v est la vitesse linéaire et w_ψ est la vitesse angulaire selon l'axe vertical de la roue directrice par rapport au corps du véhicule.

Comme pour le robot de type unicycle, on peut aisément vérifier que le système (1.2.13) est commandable puisqu'il vérifie les conditions du théorème 1.3.

1.3 Planification de trajectoire

1.3.1 Chemins et trajectoires

La plupart des travaux de planification de trajectoire sont basés sur le concept d'espace des configurations du robot introduit dans [Lozano-Pérez, 1983] au début des années 80. Une *configuration* désigne l'ensemble des paramètres caractérisant d'une manière unique le robot dans son environnement ou espace de travail. L'ensemble des configurations du robot est l'*espace des configurations* \mathcal{Q} , qui a une structure de variété différentielle. On note n la dimension de \mathcal{Q} .

Une *trajectoire* est une fonction continue de $[t_{initial}, t_{final}] \subset \mathbb{R}$ dans \mathcal{Q} qui à toute valeur $t \in [t_{initial}, t_{final}]$ associe une configuration :

$$\begin{aligned} q : [t_{initial}, t_{final}] &\rightarrow \mathcal{Q} \\ t &\mapsto q(t) \end{aligned}$$

Une trajectoire est dite *admissible* si elle est solution du système d'équations différentielles correspondant au modèle cinématique du robot, incluant les contraintes sur les commandes, pour des conditions initiales et finales données. Un *chemin* est l'image d'une trajectoire dans \mathcal{Q} . Un *chemin admissible* est l'image d'une trajectoire admissible.

1.3.2 Planification de trajectoire : définition

Une partie essentielle de l'autonomie des véhicules réside en la capacité à planifier des trajectoires admissibles assurant des déplacements sans collision dans un environnement particulier. Cet environnement contient généralement des zones dans lesquelles le robot ne peut pas se déplacer. Ces zones peuvent être détectées lorsque le robot se déplace. Un cas particulier d'évitement d'obstacles non stationnaires est l'évitement de collisions avec d'autres robots. Le robot doit ainsi avoir la capacité de calculer ou de mettre à jour sa trajectoire en temps réel tant que la mission n'est pas achevée. Ainsi, dans notre cas, le problème de planification de trajectoire pour un robot mobile peut se définir de la manière suivante.

Définition 1.1 *On appelle planification de trajectoire, le calcul d'une trajectoire admissible et sans collision pour un robot entre une configuration de départ et une configuration d'arrivée données.*

1.3.3 Evitement réactif d'obstacles

Dans cette partie, nous faisons un bref état de l'art des principales méthodes d'évitement réactif d'obstacles afin de mettre en lumière leurs avantages et leurs limites par rapport aux spécificités de notre problématique.

Méthodes analytiques

Même en l'absence d'obstacle, commander un système non holonome pour l'amener d'une configuration de départ à une configuration d'arrivée n'est pas une chose aisée. En effet, il n'existe pas aujourd'hui d'algorithme général permettant de résoudre le problème pour n'importe quel système non holonome. Des méthodes analytiques sont connues seulement pour certaines classes de systèmes (voir [Lafferriere et Sussmann, 1992, Jacob, 1991] par exemple). Pour les autres, on ne dispose que de méthodes numériques.

En outre, la présence d'obstacles rend les méthodes analytiques inapplicables à des systèmes non holonomes [Schwartz et Sharir, 1988, LaValle, 2006].

Méthode par décomposition en cellules

Une première méthode approchée permettant de déterminer un chemin sans collision est basée sur la décomposition cellulaire de l'environnement [Latombe, 1991]. Elle consiste à partitionner l'espace des configurations libres du robot en un ensemble de régions connexes adjacentes. Différentes techniques de décomposition de l'espace existent. On peut citer par exemple la partition de Voronoï [Choset, 1996] ou les graphes de visibilité [Chazelle et Guibas, 1989]. La décomposition obtenue est alors capturée dans un graphe de connectivité dont les noeuds correspondent aux différentes régions et les arcs aux relations d'adjacence entre elles. Le problème consistant à trouver un chemin dans l'espace des configurations est alors remplacé par le problème de recherche d'un chemin dans un graphe, problème qui peut être facilement résolu par des algorithmes classiques basés sur la programmation dynamique [Shin et McKay, 1986].

Cependant, ces méthodes se basent sur la structuration de l'espace des configurations et la modélisation *a priori* de sa connexité. En outre, les discréétisations en grille de l'espace de travail, proposées par ces méthodes, limitent l'espace des trajectoires admissibles.

Champs de potentiel

Les méthodes de champs de potentiel pour la navigation en robotique, initialement proposées par [Khatib, 1986] pour un bras manipulateur, consiste à assimiler le robot à une particule contrainte à se déplacer dans un champ de potentiel fictif obtenu par la composition d'un premier champ attractif (atteindre la configuration désirée) et d'un ensemble de champs répulsifs modélisant la présence d'obstacles dans l'espace du robot. À chaque position du robot, une force résultant de l'action conjuguée des obstacles et du but est calculée. Elle correspond à la direction à suivre.

De nombreuses adaptations de cette technique ont été proposées. On peut par exemple citer [Borenstein et Koren, 1991], qui calcule une direction de mouvement à partir d'informations proximétriques. Il faut cependant noter que ces méthodes purement réactives sont sujettes à des minima

locaux. Par ailleurs, elles peuvent entraîner un mouvement oscillatoire du robot dans certaines situations (des passages étroits par exemple).

Ces problèmes peuvent être résolus de différentes façons. Il est par exemple possible de déclencher un comportement particulier lorsque l'on rencontre un tel minimum (déplacement aléatoire, suivi de murs ...) [Barraquand et al., 1992], [Kavraki et al., 1996], [Statheros et al., 2006]. Il est également possible d'imposer que la fonction représentant le champ de potentiel soit une fonction harmonique [Kim et Khosla, 1992], [Rimon et Koditschek, 1992], ce qui garantit qu'il n'y ait pas de minimum local, mais complexifie beaucoup son calcul, rendant son implémentation délicate.

Méthode de la fenêtre dynamique

Cette technique proposée dans [Fox et al., 1997] travaille dans l'espace des commandes du robot. La taille du domaine de recherche des vitesses accessibles (c'est-à-dire n'entraînant pas de collisions) est réduite par la prise en compte explicite du modèle cinématique du système. Les commandes envoyées au robot sont le résultat de la maximisation sur ce domaine de recherche d'une fonction coût liée à la configuration finale. L'utilisation de cette méthode est très intéressante pour un robot se déplaçant rapidement ou pour un robot ayant des capacités d'accélération et de décélération limitées. Elle permet alors de produire un déplacement du robot sûr et régulier. Son extension au cadre multi-robots est cependant très délicate du fait de son manque de flexibilité.

Bande élastique

Le concept de bande élastique a été proposé dans [Quinlan, 1994] pour les plates-formes mobiles. Il a été étendu aux manipulateurs mobiles dans [Brock et Khatib, 1998]. Le but est de déformer localement une trajectoire donnée pour prendre en compte d'éventuels obstacles en temps réel. Cette trajectoire initialement planifiée est représentée par une série de boules adjacentes dans l'espace des configurations. Le rayon d'une boule centrée en une configuration est la distance de cette configuration à l'obstacle le plus proche. Ainsi une trajectoire est sans collision lorsque les boules qui la composent se recouvrent. Développée pour des systèmes sans contrainte cinématique, cette technique considère la trajectoire comme une bande élastique se modifiant sous l'action de forces répulsives générées par les obstacles et de forces internes de contraction ou d'élasticité.

Cette technique a été étendue à un robot de type voiture par [Khatib et al., 1997]. Le lissage de la courbe joignant les centres des boules est réalisé en utilisant une courbe de Bézier. Cependant, cette technique ne peut s'appliquer qu'au prix d'approximations sur la forme des trajectoires (combinaisons d'arcs de cercle et de lignes droites), ce qui rend impossible la navigation en environnement fortement contraint.

L'exposé de ces différentes méthodes, de leurs hypothèses et leurs limitations fait immédiatement

apparaître leur inadéquation à une flottille de robots évoluant dans des environnements inconnus fortement contraints. Cela motive le développement d'une méthode d'évitement réactif d'obstacles pour des systèmes non holonomes.

1.4 Poursuite de trajectoire

1.4.1 Rappel sur la commande en boucle ouverte et fermée

Après avoir générée la trajectoire désirée du robot mobile, la question qui se pose est de savoir comment le système physique réalise les mouvements planifiés, via la commande des actionneurs dont il est équipé. Une première approche consisterait à appliquer directement les commandes obtenues durant l'étape de planification. En pratique, si le système présente très peu d'imperfections (erreurs de modélisation, erreurs de mesure, ...) et s'il est très peu perturbé, la commande en boucle ouverte produira des résultats satisfaisants. Par contre, si ces imperfections et ces perturbations ne peuvent pas être négligées, même si elles sont petites, l'application de la commande en boucle ouverte produira un écart important entre le résultat désiré et la réponse obtenue. Ainsi, un léger glissement des roues sur le sol entraîne une dérive du véhicule pratiquement inévitable (voir figure 1.4).

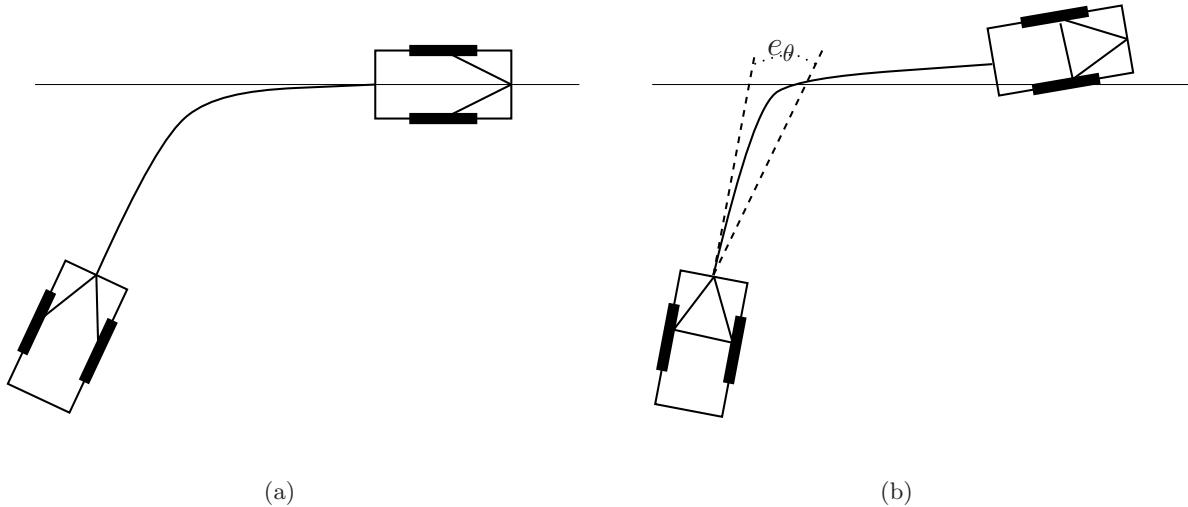


FIG. 1.4 – (a) Trajectoire planifiée. (b) Trajectoire réellement effectuée par le robot.

Ce type de comportement peut être évité si la commande en boucle ouverte est corrigée en fonction de la position et de l'orientation du robot par rapport à la trajectoire planifiée. La commande résultante est appelée commande en boucle fermée. Elle permet de rendre le système moins sensible aux perturbations, aux simplifications et approximations dans la modélisation, aux variations des paramètres du modèle ainsi qu'aux incertitudes sur les variables physiques mesurées. Toutefois, si les perturbations sont trop fortes et si on ne les prend pas en compte dans la conception de la commande, rien ne permet d'assurer que le système aura le comportement désiré.

De manière générale, l'objectif de la théorie de la commande robuste est de déterminer des commandes de manière à :

- prendre explicitement en compte les incertitudes de modélisation du système ainsi que les perturbations afin de les compenser ou les atténuer,
- garantir le bon fonctionnement du système par rapport à un ensemble de critères donnés,
- réaliser une tâche désirée avec une précision donnée.

1.4.2 Le problème de poursuite de trajectoire

Introduction

Considérons le système non linéaire suivant :

$$\dot{q} = f(q, u) \quad (1.4.1)$$

où $q \in \mathcal{Q} \subset \mathbb{R}^n$ représente l'état du système avec \mathcal{Q} un ensemble ouvert de \mathbb{R}^n , $u \in \mathbb{R}^m$ est l'entrée de commande et $f : \mathcal{Q} \times \mathbb{R}^m \rightarrow \mathbb{R}^n$ est une application suffisamment différentiable.

Soit une trajectoire admissible q_{ref} du système (1.4.1), c'est-à-dire qu'il existe une commande de référence u_{ref} telle que :

$$\dot{q}_{ref} = f(q_{ref}, u_{ref})$$

Le problème de **poursuite de trajectoire** consiste à déterminer une commande permettant de stabiliser asymptotiquement l'**erreur de suivi** $q_e = q - q_{ref}$.

Par exemple, pour un robot de type unicycle, le problème est de stabiliser à l'origine l'erreur :

$$q_e = (x - x_{ref}, y - y_{ref}, \theta - \theta_{ref})$$

comme l'illustre la figure 1.5.

Condition nécessaire de Brockett

Un aspect essentiel du problème de stabilisation des systèmes non holonomes est lié à l'impossibilité d'obtenir une stabilisation asymptotique par le biais de retours d'état statiques, continus et stationnaires. Ceci découle d'un résultat important montré dans [Brockett, 1983].

Théorème 1.4 [Brockett, 1983] *Considérons le système non linéaire (1.4.1) où f est différentiable et $f(0, 0) = 0$. Une condition nécessaire pour qu'il existe un retour d'état $u(q)$ continu qui rende l'origine du système bouclé localement asymptotiquement stable est que l'image par f de tout voisinage de l'origine de $\mathbb{R}^{n \times m}$ soit un voisinage de 0 dans \mathbb{R}^n .*

Cette condition implique que de nombreux systèmes non linéaires commandables ne sont pas asymptotiquement stabilisables par retour d'état continu. C'est en particulier le cas de tous les robots

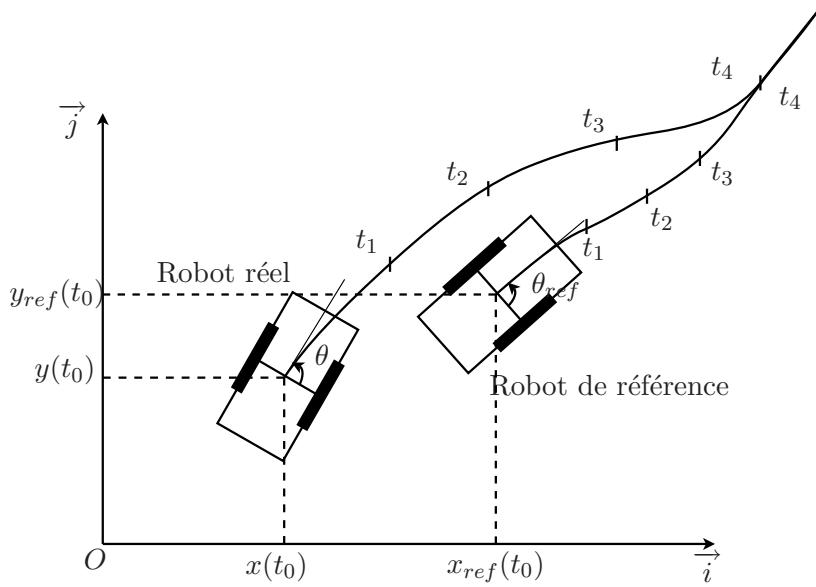


FIG. 1.5 – Suivi d'un véhicule de référence.

mobiles non holonomes. Ceci a conduit à développer d'autres stratégies de commande afin de résoudre ce problème de stabilisation asymptotique. Trois types de commandes ont principalement été considérés [Kolmanovsky et McClamroch, 1995] :

- les bouclages dynamiques et, en particulier, quasi-statiques [Delaleau et da Silva, 1998],
- les retours d'état non stationnaires continus, qui sont des retours d'état qui dépendent de la variable temporelle (c'est-à-dire $u(q, t)$ au lieu de $u(q)$ pour un retour d'état classique),
- les retours d'état discontinus, qui sont des retours d'état présentant au moins une discontinuité.

Retours d'état non stationnaires continus

L'utilisation de retours d'état non stationnaires continus pour la stabilisation asymptotique d'un robot mobile trouve son origine dans les travaux de Samson au début des années 90 [Samson, 1990]. Un des premiers résultats d'existence de telles commandes stabilisantes est donné dans [Coron, 1992]. Par la suite, diverses approches basées sur la méthode directe de Lyapunov [Pomet, 1992], [Samson, 1995], [Goddavn et Egeland, 1997], [Dixon et al., 2000b], sur les commandes sinusoïdales et polynomiales [Murray et Sastry, 1993], sur les techniques de backstepping [Jiang et Nijmeijer, 1999] conduisent à la construction de retours d'état continus non stationnaires permettant de stabiliser asymptotiquement un robot mobile. Bien que connues pour avoir d'assez bonnes propriétés de robustesse, ces commandes présentent généralement des inconvénients d'ordre pratique concernant les taux de convergence, le réglage délicat des paramètres de commande ou les trajectoires générées. Le problème de la génération d'une loi de commande stabilisante prenant en compte les saturations des entrées est également traité dans [Jiang et al., 2001]. La convergence de l'état du système vers l'origine est rapide. Cependant, cette

commande ne présente pas de bonnes propriétés en termes de robustesse vis-à-vis des perturbations.

Retours d'état discontinus

Quant à l'utilisation de commandes discontinues pour la robotique mobile, elle trouve son origine dans les travaux de Bloch à la fin des années 80 [Bloch et McClamroch, 1989, Bloch et al., 1992]. De nombreux travaux proposent des commandes continues par morceaux [Hespanha et al., 1999] ou des commandes discontinues [Bloch et al., 1992], mais ne traitent pas explicitement du problème de robustesse. Des commandes permettant d'obtenir une convergence exponentielle sont proposées dans [Canudas-De-Wit et Sordalen, 1992]. Cependant, elles sont sensibles aux erreurs initiales et aux perturbations. Inspirées par ces résultats, des stratégies robustes basées sur une transformation du système en coordonnées polaires [Astolfi, 1996, Chwa, 2004] assurent une convergence exponentielle du système mais présentent une singularité à l'origine. D'autres types de commande notamment par modes glissants [Floquet et al., 2003, Drakunov et al., 2005] permettent d'obtenir de bons résultats mais font apparaître le phénomène de réticence, qui conduit à l'usure rapide des actionneurs.

Conclusion

De nombreuses méthodes pour la synthèse de commandes permettant de résoudre le problème de stabilisation asymptotique pour un système non holonome ont été développées. Cependant, l'étude de ces méthodes a mis en évidence un certain nombre de limitations (phénomène de réticence, singularités, convergence lente, problèmes de robustesse vis-à-vis de dynamiques non modélisées, ...). Cela motive le développement d'une synthèse générique de commandes robustes évitant ces limitations.

Notre approche s'inspirera de la technique de la commande par modes glissants. Ce choix est guidé par le fait que cette méthode a déjà fait ses preuves dans le cadre de la commande des robots mobiles notamment dans [Floquet et al., 2003] et qu'elle ouvre des pistes de recherches intéressantes pour la synthèse de commandes robustes, facilement implantables et présentant un bon compromis entre robustesse et performance.

1.5 Système multi-robots : les problématiques

Elaborer un système multi-robots conduit nécessairement à s'interroger sur l'architecture individuelle des robots : dans quelle mesure des architectures de planification et de commande données peuvent-elles répondre aux exigences et contraintes posées par un tel système ? Suffit-il d'étendre une architecture mono-robot dans un cadre multi-robots, ou alors faut-il considérer la nature des problématiques multi-robots dans les fondements même de l'architecture ?

1.5.1 Planification de trajectoire

La présence de contraintes communes telles que l'évitement de collisions ou le maintien des liaisons de communication, engendre une interaction entre robots. Ces interactions peuvent représenter une menace pour l'intégrité du système, ou tout au moins risquent-elles de dégrader ses performances. Cependant, elles peuvent également être mises à profit pour réaliser plus efficacement des tâches mono-robot (redondance, partage de tâches sécables), ou bien réaliser des tâches qui, par nature (opérations distantes simultanées, explorations …), ne pourraient être réalisées par un seul robot.

Mécanismes de coordination

Dans le cas d'absence de coopération, les robots n'échangent pas d'informations sur leurs intentions et ne devinent pas les intentions des autres. Par conséquent, une approche du pire cas est souvent adoptée (par exemple, une approche basée sur la théorie des jeux dans [Tomlin et al., 1998]). Cependant, elle devient délicate à mettre en œuvre lorsque le nombre de robots dépasse trois.

Mettre à profit les interactions dans un système multi-robots consiste à y introduire des mécanismes de coordination, afin de rendre cohérentes entre elles les actions des robots. En étudiant les communautés humaines, Mintzberg [Mintzberg, 1979] a identifié trois processus de coordination :

- la ***coordination par ajustements mutuels*** où les individus s'accordent pour partager des ressources en vue d'atteindre un but commun (aucun individu n'a de contrôle sur les autres individus, et le processus de décision est conjoint),
- la ***coordination par “leadership”*** ou supervision où une relation hiérarchique existe entre des individus (certains individus exercent alors un contrôle sur d'autres),
- la ***standardisation*** où des procédures sont prédefinies en vue de situations d'interactions particulières (par exemple des règles dont l'application peut limiter les conflits).

Les différents mécanismes de coordination exploités dans les systèmes multi-robots sont tous des manifestations d'un ou de plusieurs de ces processus fondamentaux. Nous exposons ci-après les deux principaux mécanismes appliqués à la planification de trajectoire.

Processus centralisé

Les approches centralisées visent à synthétiser pour les robots de la flottille, un ensemble de trajectoires via un superviseur. Ce superviseur dispose de toutes les informations relatives aux différents véhicules de la flottille et construit les trajectoires individuelles pour chaque robot, en résolvant typiquement un problème d'optimisation de grande dimension. Ces stratégies permettent théoriquement d'obtenir les trajectoires optimales, compte tenu des connaissances, et de faire face à des problèmes complexes. Le prix à payer est la complexité en termes de calculs et la dépendance des robots vis-à-vis du superviseur. En pratique, le nombre de calculs augmente très rapidement en fonction de la taille de la formation et le problème devient rapidement insoluble pour une flottille de plus de cinq robots.

Dans cette catégorie, des approches basées sur les champs de potentiel [Loizou et Kyriakopoulos, 2002, Olfati-Saber et al., 2003, Tanner et al., 2003], sur la méthode de la fenêtre dynamique [Ogren, 2003] ou sur la commande optimale [Schouwenaars et al., 2001, Dunbar et Murray, 2002, Hu et al., 2002] ont été proposées.

Processus décentralisé

Des stratégies décentralisées ont été récemment introduites. Elles nécessitent généralement un flux de communications assez élevé afin de transmettre des requêtes informatives aux autres individus. Le protocole peut inclure des notions d'intention et d'engagement à partir desquelles chaque robot élabore sa propre trajectoire en prenant en considération les activités des autres robots.

Les avantages d'une telle stratégie sont multiples : absence de superviseur, diminution des temps de calcul par rapport à une approche centralisée, plus grande robustesse (un problème par exemple une panne ou une destruction sur un ou plusieurs robots n'entraîne généralement pas une destruction de l'ensemble du système). Le prix à payer est bien sûr une incapacité à délivrer une performance optimale puisqu'à chaque instant, chaque robot ne dispose que d'informations limitées et incomplètes sur les autres robots de la flotte.

L'une des approches décentralisées consiste à décomposer le problème de planification en deux [Guo et Parker, 2002]. D'abord, chaque robot détermine un chemin afin d'éviter les obstacles stationnaires. Puis, une fois les chemins fixés, les vitesses des robots sont ajustées de manière à éviter les collisions. Cependant, cette approche ne permet pas de remplir la contrainte sur le maintien des liaisons de communication. D'autres stratégies décentralisées basées sur les champs de potentiel [Gazi et Passino, 2004], les fonctions de navigation (champs de potentiel sans minimum local) [Dimarogonas et al., 2003, Tanner et Kumar, 2005, Gennaro et Jadbabaie, 2006], sur la décomposition en cellules [Lindhe et al., 2005] ont été développées. Cependant, elles ne sont pas applicables à des systèmes non holonomes.

Récemment, d'autres approches basées sur la commande optimale et pouvant être appliquées à des systèmes non holonomes ont été introduites [Dunbar et Murray, 2006, Keviczky et al., 2006, Kuwata et al., 2006]. Dans [Dunbar et Murray, 2006], une solution permet de résoudre le problème de planification décentralisée dans un environnement sans obstacle. Dans [Keviczky et al., 2006], une stratégie d'évitement réactif d'obstacles est formulée où chaque système planifie sa trajectoire ainsi que celle des véhicules voisins. Cependant, cela entraîne une diminution du niveau de décentralisation et des commandes d'urgence doivent être introduites afin de garantir l'évitement de collisions. Dans [Kuwata et al., 2006], la stratégie de planification est basée sur une architecture de type "meneur / suiveur". Une relation hiérarchique existe entre les robots et caractérise l'ordre de planification. L'utilisation de ce type d'architecture facilite la mise en œuvre des algorithmes. Cependant, du fait de non retour d'information du suiveur vers le meneur, la structure est sensible aux perturbations et à tous les problèmes sur les suiveurs.

Ce tour d'horizon des principaux mécanismes de coordination laisse entrevoir la diversité des paradigmes étudiés et exploités dans le contexte multi-robots. Cependant, il faut noter que les mécanismes énumérés ne présentent pas à la fois des performances élevées et un temps de calcul faible.

1.5.2 Poursuite de trajectoire

En ce qui concerne la fonctionnalité de poursuite de trajectoire pour une flottille de robots mobiles, plusieurs stratégies peuvent être mises en place. La plus simple est d'étendre l'architecture de poursuite de trajectoire mono-robot dans un cadre multi-robots. Dans ce cas, seule la fonctionnalité de planification intègre les mécanismes de coordination entre robots.

Depuis quelques années se développent d'autres stratégies [Desai et al., 2001, Das et al., 2002, Lawton et al., 2003, Tanner et al., 2004, Orqueda et Fierro, 2006, Chen et Serrani, 2006]. Ces travaux, initiés dans [Desai et al., 1998], permettent d'intégrer un mécanisme assurant la coordination d'un groupe de robots dans la fonctionnalité de poursuite de trajectoire. Introduites au départ pour maintenir une forme géométrique spécifique pour la flottille dans le cadre d'une architecture de type "meneur / suiveur" [Desai et al., 2001], ces techniques ont été généralisées pour des architectures quelconques [Lawton et al., 2003]. Par rapport aux stratégies non coopératives de poursuite de trajectoire, elles présentent l'avantage de s'affranchir de la connaissance de la position des robots par rapport à un repère fixe.

Ces travaux très récents sont liés au développement de capteurs, notamment les caméras omni-directionnelles, et d'algorithmes de vision permettant de déterminer les positions relatives entre les robots. Par exemple, dans [Vidal et al., 2004], les auteurs présentent un algorithme basé sur des techniques de segmentation. Une loi de commande, basée sur la vision et assurant la coordination des robots, est donnée dans [Das et al., 2002] en utilisant un retour linéarisant. Dans [Mariottini et al., 2005], les auteurs proposent une loi de commande centralisée en supposant que les robots suiveurs peuvent recevoir la vitesse des robots meneurs et estimer leur configuration relative par l'intermédiaire de caméras panoramiques. Dans [Chen et Serrani, 2006], une architecture décentralisée de commande utilisant uniquement les positions et vitesses relatives est construite. Récemment, un algorithme décentralisé, basé sur des caméras omni-directionnelles est donné dans [Orqueda et Fierro, 2006]. Il ne nécessite pas de communication entre les robots puisque des observateurs grands gains sont construits afin d'estimer les dérivées des positions relatives.

Deux points très importants sont à noter :

- Les stratégies coopératives de poursuite de trajectoire ne peuvent être utilisées pour l'ensemble de la flottille. En effet, pour l'un au moins des robots, la stratégie doit être non coopérative afin de pouvoir le localiser par rapport à un repère fixe.
- Lorsque les stratégies de poursuite de trajectoire sont coopératives, une attention particulière doit être portée aux propriétés de robustesse des commandes stabilisantes. En effet, une erreur sur l'un des robots se répercute sur les autres. Cependant, relativement peu de solutions ont été

apportées pour résoudre ce problème.

1.6 Conclusion

Ce chapitre propose une vision d'ensemble des caractéristiques et techniques de planification et de poursuite de trajectoire associées aux systèmes non holonomes dans un cadre mono-robot puis multi-robots.

Le domaine est très riche et de multiples algorithmes sont proposés dans la littérature, parfois bien formalisés et éprouvés, mais parfois aussi à l'état de théorie ou au contraire construits sur des bases empiriques.

D'une part, il convient de noter que les différentes méthodes de planification, exposées pour un seul robot, font généralement apparaître leur inadéquation à une flottille de robots évoluant dans des environnements inconnus fortement contraints. Cela motive le développement d'une méthode d'évitement réactif d'obstacles, pour un système non holonome, suffisamment flexible pour pouvoir être adaptée au cas multi-robots. Ainsi, dans la première partie de ce mémoire, après avoir élaboré un planificateur de trajectoire pour un robot, nous développerons un planificateur pour une flottille s'inspirant des techniques existantes de coordination.

D'autre part, la simplicité apparente des équations régissant le mouvement des robots mobiles sur roues masque la complexité des problèmes de commande sous-jacents. En particulier, il semble qu'aucune solution au problème de poursuite asymptotique de trajectoires admissibles ne soit complètement satisfaisante au niveau de la gestion du compromis entre robustesse et performance. Néanmoins, cette considération est primordiale dans le cadre multi-robots. C'est pourquoi il est intéressant de développer une méthode générique permettant d'atteindre l'objectif de poursuite de trajectoire de manière robuste et efficace. Ainsi, dans la seconde partie de ce mémoire, nous élaborerons des algorithmes génériques de commande par modes glissants.

Tous les algorithmes seront testés sur une plate-forme expérimentale.

Première partie

Planification de trajectoire

Chapitre 2

Planification de trajectoire : cadre mono-robot

2.1 Introduction

Nous avons vu dans le chapitre précédent que les différentes méthodes de planification pour un seul robot (méthodes analytiques, méthodes par décomposition en cellules, champs de potentiel, méthode de la fenêtre dynamique, méthode de la bande élastique), ne peuvent généralement pas être étendues à une flottille de robots évoluant dans des environnements inconnus fortement contraints.

Ces considérations conduisent à envisager un algorithme de planification de trajectoire pour un système non holonome basé sur une optimisation sous contraintes. En effet, la formulation du problème de planification sous forme d'un problème de commande optimale permet une flexibilité de l'algorithme afin de pouvoir être adapté au cadre multi-robots.

Notons qu'une carte complète de l'environnement n'est généralement pas connue par avance puisque les obstacles sont détectés au fur et à mesure du déplacement du robot. Ainsi, seule une certaine zone autour du robot est prise en compte. Dans cette région, nous supposerons que l'environnement est statique et parfaitement connu. Par conséquent, puisque l'environnement est exploré en ligne, une trajectoire joignant la configuration initiale à la configuration finale doit être calculée graduellement au cours du temps tant que la mission n'est pas achevée. Il est donc nécessaire de faire appel à une stratégie de planification sur un horizon glissant. Sur chaque horizon, une partie de la trajectoire est planifiée, en ligne, en résolvant un problème de commande optimale sous contraintes [Defoort et al., 2007d].

Afin de résoudre le problème de commande optimale, nous allons utiliser les propriétés de platitude que possèdent tous les robots mobiles autonomes constitués d'un solide unique en mouvement (robot de type unicycle, robot de type voiture, . . .). En effet, la platitude permet de simplifier considérablement les questions de commande optimale, souvent inextricables si l'on suit d'autres voies.

Un algorithme a été proposé dans [Milam, 2003] pour la génération hors ligne de la trajectoire optimale de systèmes non linéaires plats. Il repose sur la transformation du problème de commande optimale en un problème d'optimisation de paramètres grâce à la spécification des sorties plates par des fonctions splines. Cependant, cette technique n'est pas directement applicable en ligne. Il est nécessaire de la combiner avec une stratégie de planification sur un horizon glissant. Reprenant et adaptant l'algorithme proposé dans [Milam, 2003], nous allons élaborer un planificateur de trajectoire admissible sans collision pouvant être appliqué en ligne.

Après avoir exposé le problème de planification pour un robot mobile, l'algorithme hors ligne de génération de trajectoire, basé sur les concepts de platitude [Fliess et al., 1995], les fonctions splines [Boor, 1978] et la programmation non linéaire [Lawrance et al., 1997], est décrit. Puis, il est étendu à la planification en ligne. La généralité et l'efficacité de notre approche seront illustrées au travers d'exemples numériques. Cette stratégie sera la base de notre algorithme de génération de trajectoire pour la flottille de robots mobiles développé dans le prochain chapitre.

2.2 Description du problème

Considérons un robot mobile non holonome, identifié par l'indice i , dont la dynamique est régie par l'équation différentielle suivante :

$$\dot{q}_i = f_i(q_i, u_i) \quad (2.2.1)$$

où $q_i \in \mathbb{R}^n$ représente l'état du système, $u_i \in \mathbb{R}^m$ est l'entrée de commande et $f_i : \mathbb{R}^n \times \mathbb{R}^m \rightarrow \mathbb{R}^n$ est une application suffisamment différentiable.

2.2.1 Propriétés de platitude

La platitude est une propriété naturelle d'un bon nombre de systèmes dynamiques. Elle concerne autant les systèmes linéaires que non linéaires [Fliess et al., 1992, Fliess et al., 1995] et peut être étendue pour des systèmes à retard ou régis par des équations aux dérivées partielles (voir par exemple [Mounier et Rudolph, 1998, Martin et al., 2000]). Les systèmes plats forment une classe pour laquelle nous disposons d'une paramétrisation explicite de toutes leurs solutions. Par explicite, nous entendons ici sans intégration, uniquement à partir d'opérations de nature algébrique et d'un nombre fini de dérivations temporelles.

Définition 2.1 [Fliess et al., 1992, Fliess et al., 1995] *Considérons le système général*

$$\dot{q} = f(q, u)$$

où $q \in \mathbb{R}^n$ et $u \in \mathbb{R}^m$.

Ce système est différentiellement plat s'il existe $z = [z_1, \dots, z_m]^T \in \mathbb{R}^m$ (différentiellement indépendant

et dépendant des entrées, des dérivées des entrées jusqu'à un ordre $l \leq n$ et de l'état) appelé *sortie plate*, tel que l'état et la commande puissent être exprimés en fonction de la sortie plate et de ses dérivées. Ceci revient à dire qu'il existe trois fonctions régulières $\varphi_0 : \mathbb{R}^n \times (\mathbb{R}^m \times \dots \times \mathbb{R}^m) \rightarrow \mathbb{R}^m$, $\varphi_1 : (\mathbb{R}^m \times \dots \times \mathbb{R}^m) \rightarrow \mathbb{R}^n$ et $\varphi_2 : (\mathbb{R}^m \times \dots \times \mathbb{R}^m) \rightarrow \mathbb{R}^m$ telles que :

$$\begin{cases} z &= \varphi_0(q, u, \dot{u}, \dots, u^{(l)}) \\ q &= \varphi_1(z, \dot{z}, \dots, z^{(l-1)}) \\ u &= \varphi_2(z, \dot{z}, \dots, z^{(l)}) \end{cases} \quad (2.2.2)$$

L'existence d'un critère donnant les conditions pour décider si un système non linéaire est plat ou non est une question délicate. Néanmoins, pour certaines classes de systèmes non linéaires (systèmes linéarisables par bouclage statique, systèmes à une seule commande, systèmes affines en la commande de co-dimension 1, systèmes sans dérive), des résultats sont connus (voir par exemple [Martin, 1993] ou [Martin et Rouchon, 1995]). Le théorème fondamental suivant en est un exemple.

Théorème 2.1 [Martin et Rouchon, 1995] *Le système*

$$\dot{q} = \sum_{j=1}^{n-2} g_j(q) u_j$$

avec $q \in \mathbb{R}^n$ est plat dès qu'il est commandable.

Ainsi, de nombreux systèmes mécaniques sont plats. Revenons au cas des robots de type unicycle et de type voiture qui couvrent une grande classe de robots mobiles non holonomes. Comme il a été montré dans le chapitre introductif, le modèle cinématique du robot unicycle est :

$$\begin{cases} \dot{x}_i &= v_i \cos \theta_i \\ \dot{y}_i &= v_i \sin \theta_i \\ \dot{\theta}_i &= w_i \end{cases} \quad (2.2.3)$$

En prenant comme sortie $z_i = [x_i, y_i]^T$, le système (2.2.3) peut se réécrire sous la forme d'un système plat. En effet, on a :

$$\begin{cases} \theta_i &= \arctan\left(\frac{\dot{y}_i}{\dot{x}_i}\right) \\ v_i &= \sqrt{\dot{x}_i^2 + \dot{y}_i^2} \\ w_i &= \frac{\dot{x}_i \ddot{y}_i - \dot{y}_i \ddot{x}_i}{\dot{x}_i^2 + \dot{y}_i^2} \end{cases} \quad (2.2.4)$$

Similairement, le système :

$$\begin{cases} \dot{x}_i &= v_i \cos \theta_i \\ \dot{y}_i &= v_i \sin \theta_i \\ \dot{\theta}_i &= v_i \frac{\tan \psi_i}{L} \\ \dot{\psi}_i &= w_{\psi,i} \end{cases} \quad (2.2.5)$$

modélisant un robot de type voiture, est plat. En prenant comme sortie plate $z_i = [x_i, y_i]^T$, on obtient :

$$\left\{ \begin{array}{lcl} \theta_i & = & \arctan\left(\frac{\dot{y}_i}{\dot{x}_i}\right) \\ v_i & = & \sqrt{\dot{x}_i^2 + \dot{y}_i^2} \\ \psi_i & = & \arctan\left(L \frac{\dot{x}_i \ddot{y}_i - \dot{y}_i \ddot{x}_i}{(\dot{x}_i^2 + \dot{y}_i^2)^{3/2}}\right) \\ w_{\psi,i} & = & \dot{\psi}_i \end{array} \right.$$

Remarque 2.1 *Dans la suite de ce manuscrit, on ne considère que les systèmes plats, ce qui inclut tous les robots mobiles autonomes (constitué d'un solide unique en mouvement).*

2.2.2 Description du problème de planification

Le problème de planification consiste à calculer pour le robot d'indice i , la trajectoire admissible et sans collision, joignant la configuration initiale $q_i(t_{initial}) = q_{i,initial}$ à la configuration finale $q_i(t_{final}) = q_{i,final}$ (avec des commandes initiales $u_i(t_{initial}) = u_{i,initial}$ et finales $u_i(t_{final}) = u_{i,final}$ qui peuvent être fixées ou laissées libres) et optimisant un critère défini. La fonction critère peut être une mesure du temps, d'énergie ou un critère plus complexe tel que la visibilité. La tâche du robot est définie comme l'atteinte de la destination finale $q_i(t_{final})$.

Remarque 2.2 *Afin de simplifier les notations, on considère que la forme géométrique du robot est inscrite dans un cercle de centre le point (x_i, y_i) milieu de l'axe des roues motrices et de rayon ρ_i (voir figure 2.1). Les résultats peuvent être affinés en utilisant des formes plus complexes (ellipse, ...).*

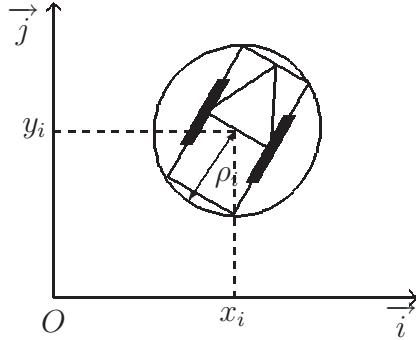


FIG. 2.1 – Forme géométrique du robot.

La prise en compte du modèle cinématique du robot dans le problème de planification est indispensable afin d'assurer que la trajectoire planifiée est admissible (i.e. respect des contraintes de non holonomie). Nous noterons $\mathcal{U}_i \subset \mathbb{R}^m$ l'ensemble des commandes admissibles du robot. Par conséquent :

$$u_i(t) \in \mathcal{U}_i, \quad \forall t \in [t_{initial}, t_{final}] \tag{2.2.6}$$

De plus, les obstacles présents dans l'environnement et supposés stationnaires apportent le volet géométrique du problème. Ainsi, l'espace des configurations \mathcal{Q}_i est divisé en deux :

- l'espace occupé $\mathcal{Q}_{i,occupé}$ caractérisant les régions où un risque de collision existe,
- l'espace libre $\mathcal{Q}_{i,libre}$ caractérisant les régions dans lesquelles le robot peut se déplacer en toute sécurité.

Néanmoins, il convient de noter que lorsque l'environnement est exploré en ligne, seule une zone autour du robot, correspondant à ses possibilités de détection et de localisation des obstacles avec précision, est prise en compte lors de la génération de trajectoire.

Définition 2.2 $\forall t_k \in [t_{initial}, t_{final}]$, on définit la **zone de détection** $\mathcal{D}_i(t_k) \subset \mathcal{Q}_i$ comme l'espace autour du robot d'indice i dans lequel l'environnement est parfaitement connu à l'instant t_k .

Définition 2.3 $\forall t_k \in [t_{initial}, t_{final}]$, on définit la **zone obstacle** $\mathcal{O}_i(t_k)$ par :

$$\mathcal{O}_i(t_k) = \mathcal{D}_i(t_k) \cap \mathcal{Q}_{i,occupé}$$

Notons que l'espace $\mathcal{O}_i(t_k)$ dépend du temps et évolue lorsque le robot se déplace et découvre de nouveaux obstacles. Les éléments de l'environnement non détectés ne sont pas pris en compte à l'instant t_k (voir figure 2.2-2.3).

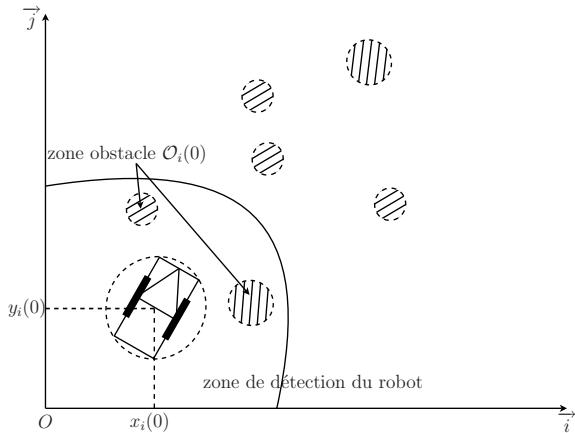


FIG. 2.2 – Zone obstacle à l'instant $t_k = 0s$.

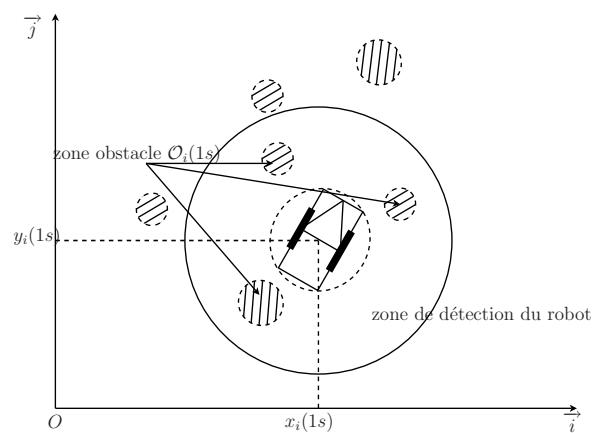


FIG. 2.3 – Zone obstacle à l'instant $t_k = 1s$.

Remarque 2.3 Comme il est fait couramment en pratique, les obstacles sont augmentés par les dimensions du robot de manière à ce que le véhicule puisse être considéré comme un point dans l'espace des configurations [Latombe, 1991].

Afin de modéliser simplement l'ensemble $\mathcal{O}_i(t_k)$ et de simplifier l'expression des contraintes d'évitement d'obstacles, on utilise un recouvrement de M_i disques¹. Par conséquent, l'ensemble $\mathcal{O}_i(t_k)$ se

¹Notons également que ce recouvrement de disques permet de modéliser les obstacles de forme quelconque ainsi que les zones inconnues (faces cachées, ...) assimilées à des régions de l'environnement où un risque de collision existe.

décompose de la façon suivante :

$$\bigcup_{m_i=1}^{M_i} O_{m_i} \approx \mathcal{O}_i(t_k) \quad (2.2.7)$$

Chaque disque O_{m_i} est défini par les coordonnées de son centre (X_{m_i}, Y_{m_i}) et son rayon r_{m_i} ($1 \leq m_i \leq M_i$).

Pour éviter les collisions, la distance entre le robot et les obstacles détectés O_{m_i} à l'instant t , i.e.

$$d_{i,O_{m_i}}(t) = \sqrt{(x_i(t) - X_{m_i})^2 + (y_i(t) - Y_{m_i})^2}$$

doit satisfaire :

$$d_{i,O_{m_i}}(t) \geq \rho_i + r_{m_i}, \quad \forall t \in [t_k, t_k + T_p], \quad \forall O_{m_i} \in \mathcal{O}_i(t_k) \quad (2.2.8)$$

où $T_p \in \mathbb{R}^+$ est l'horizon de planification.

2.3 Algorithme hors ligne de génération de trajectoire

2.3.1 Mise sous forme d'un problème de commande optimale

Dans le cadre d'une planification hors ligne, la carte de l'environnement est entièrement connue. L'horizon de planification $T_p \in \mathbb{R}^+$, qui correspond à la longueur de l'intervalle de temps sur lequel la fonction critère est évaluée, est :

$$T_p = t_{final} - t_{initial} \quad (2.3.1)$$

Le problème de planification de trajectoire peut être mis sous la forme du problème de commande optimale suivant : déterminer la trajectoire optimale $q_{i,ref}(t)$ et la commande associée $u_{i,ref}(t)$ dans l'intervalle de temps $[t_{initial}, t_{final}]$ qui minimise la fonction critère :

$$J = \int_{t_{initial}}^{t_{final}} L_i(q_i, u_i, t) dt \quad (2.3.2)$$

sous les contraintes : $\forall t \in [t_{initial}, t_{final}]$:

$$\dot{q}_i(t) = f_i(q_i(t), u_i(t)) \quad (2.3.3)$$

$$q_i(t_{initial}) = q_{i,initial} \quad (2.3.4)$$

$$q_i(t_{final}) = q_{i,final} \quad (2.3.5)$$

$$u_i(t_{initial}) = u_{i,initial} \quad (2.3.6)$$

$$u_i(t_{final}) = u_{i,final} \quad (2.3.7)$$

$$u_i(t) \in \mathcal{U}_i \quad (2.3.8)$$

$$d_{i,O_{m_i}}(t) \geq \rho_i + r_{m_i}, \quad \forall O_{m_i} \in \mathcal{Q}_{i,occupe} \quad (2.3.9)$$

Le terme $L_i(q_i, u_i, t)$ représente la fonction coût associée au robot.

Dans la littérature, des conditions nécessaires d'optimalité ont été introduites pour le problème d'optimisation sous contraintes (voir les ouvrages [Pontryagin et al., 1962, Bryson et Ho, 1975]). Cependant, les solutions analytiques du problème général d'optimisation sont difficiles à déterminer (voire impossibles). Dans ce mémoire, on utilisera une alternative qui consiste à utiliser des méthodes numériques directes afin de résoudre le problème.

2.3.2 Platitudo et planification de trajectoire

Nous allons voir que dans le cas des systèmes plats, le problème de commande optimale peut être résolu très simplement sans avoir à intégrer les équations du système.

Les conditions de platitudo reviennent à dire qu'il existe une sortie plate z_i telle que toutes les variables du système puissent s'exprimer en fonction de la sortie plate et d'un nombre fini de ses dérivées et que les équations différentielles du système sont identiquement vérifiées. Il en résulte que si l'on veut construire des trajectoires dont les conditions initiales et finales sont spécifiées, il suffit de calculer la trajectoire de la sortie plate correspondante. Ceci évite d'intégrer les équations différentielles du système (voir figure 2.4). Plus précisément, supposons que

$$\begin{cases} q_i &= \varphi_1(z_i, \dot{z}_i, \dots, z_i^{(l_i-1)}) \\ u_i &= \varphi_2(z_i, \dot{z}_i, \dots, z_i^{(l_i)}) \end{cases} \quad (2.3.10)$$

Puisque les valeurs initiales et finales de q_i et u_i sont données, la surjectivité de φ_1 et φ_2 permet de déterminer des valeurs initiales et finales de $z_i, \dot{z}_i, \dots, z_i^{(l_i)}$. Il suffit ensuite de trouver une trajectoire $z_i(t)$ au moins l_i fois dérivable qui satisfait ces conditions initiales et finales puisque q_i et u_i se déduisent de z_i et de ses dérivées jusqu'à l'ordre l_i en utilisant (2.3.10).

Une fois que les contraintes ainsi que la fonctionnelle à minimiser sont écrites en fonction de la sortie plate et de ses dérivées jusqu'à l'ordre l_i , la problème de commande optimale se réduit à trouver la trajectoire optimale des sorties plates minimisant :

$$J = \int_{t_{initial}}^{t_{final}} L_i(\varphi_1(z_i, \dot{z}_i, \dots, z_i^{(l_i-1)}), \varphi_2(z_i, \dot{z}_i, \dots, z_i^{(l_i)}), t) dt \quad (2.3.11)$$

sous les contraintes : $\forall t \in [t_{initial}, t_{final}]$:

$$\begin{cases} \varphi_1(z_i(t_{initial}), \dots, z_i^{(l_i-1)}(t_{initial})) &= q_{i,initial} \\ \varphi_1(z_i(t_{final}), \dots, z_i^{(l_i-1)}(t_{final})) &= q_{i,final} \\ \varphi_2(z_i(t_{initial}), \dots, z_i^{(l_i)}(t_{initial})) &= u_{i,initial} \\ \varphi_2(z_i(t_{final}), \dots, z_i^{(l_i)}(t_{final})) &= u_{i,final} \\ \varphi_2(z_i(t), \dots, z_i^{(l_i)}(t)) &\in \mathcal{U}_i \\ d_{i,O_{m_i}}(t) &\geq \rho_i + r_{m_i}, \quad \forall O_{m_i} \in \mathcal{Q}_{i,occupe} \end{cases} \quad (2.3.12)$$

Par construction, les trajectoires $q_i(t)$ et $u_i(t)$ obtenues en remplaçant z_i et ses dérivées par leur valeur en fonction du temps dans (2.3.10) satisfont identiquement, par définition des systèmes plats,

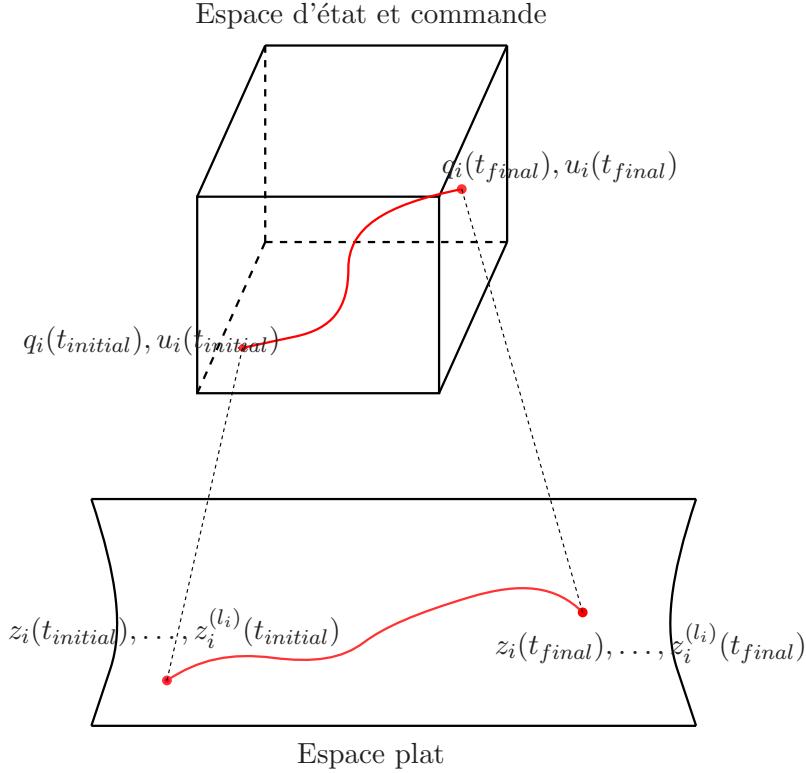


FIG. 2.4 – Platitude et planification de trajectoire.

les équations différentielles du système. En conséquence, le nombre de variables dans le problème de commande optimale est réduit. Ceci permet de diminuer le temps de calcul.

2.3.3 Spécification de la sortie plate par une fonction spline

Afin de transformer le problème de commande optimale en un problème de programmation non linéaire, il est nécessaire de paramétriser la sortie plate.

Choix des fonctions B-splines

De nombreuses courbes peuvent être utilisées pour spécifier la sortie plate (série de Fourier, polynômes, ...). En plus de représenter précisément une base de la solution du problème de génération de trajectoire avec un nombre raisonnable de paramètres, il est nécessaire de pouvoir facilement choisir le degré de continuité de la courbe (classe C^{l_i}), sans ajouter de nouvelles contraintes, puisque q_i et u_i se déduisent de z_i et de ses dérivées jusqu'à l'ordre l_i .

Les polynômes peuvent résoudre en partie ces problèmes puisqu'ils sont facilement évaluables et différentiables. Cependant, se pose le problème de l'utilisation d'un nombre élevé de paramètres si l'horizon de planification est grand. Un autre problème vient de la dépendance globale des paramètres. C'est-à-dire que si un ajustement est mauvais localement alors il sera mauvais pour l'ensemble de la courbe [Boor, 1978]. Ces difficultés entraînent un temps de calcul très important pour déterminer le

polynôme adéquat.

Le besoin ici est de pouvoir utiliser une famille très riche de courbes dépendant d'un nombre suffisant de paramètres pour pouvoir satisfaire les différentes contraintes. Il est nécessaire aussi de pouvoir estimer l'effet de chaque paramètre afin de trouver rapidement une courbe qui corresponde à la solution du problème de commande optimale. En fait, on souhaite que les fonctions de base soient à support compact, ce qui permet de limiter leur influence sur une petite zone. L'implémentation serait alors envisageable en ligne.

Une solution est d'utiliser des polynômes par morceaux qui ont comme avantage de présenter un support compact. En outre, il est possible de gérer la continuité au niveau des points de raccordement des segments polynomiaux de telle sorte que la courbe résultante, appelée fonction spline², soit de classe C^{l_i} . La courbe spline, spécifiant la sortie plate, est construite comme une combinaison linéaire de fonctions de base appelées B-splines. Une fonction B-spline correspond au raccordement de courbes de Bézier.

Base des B-splines

Soit une suite non décroissante de réels positifs $\{\text{nœud}_0 \leq \dots \leq \text{nœud}_M\}$, appelée suite de nœuds, qui contient l'ensemble des points de raccordement. Les B-splines d'ordre $d = 1$ sont définies par : $\forall j = \{0, \dots, M - 1\}$,

$$B_{j,1}(t) = \begin{cases} 1 & \text{si } \text{nœud}_j \leq t < \text{nœud}_{j+1} \\ 0 & \text{sinon} \end{cases} \quad (2.3.13)$$

et les B-splines d'ordre $d \geq 2$ ($d \in \mathbb{N}$) par la relation de récurrence : $\forall j = \{0, \dots, M - d\}$,

$$B_{j,d}(t) = \frac{t - \text{nœud}_j}{\text{nœud}_{j+d+1} - \text{nœud}_j} B_{j,d-1}(t) + \frac{\text{nœud}_{j+d} - t}{\text{nœud}_{j+d} - \text{nœud}_{j+1}} B_{j+1,d-1}(t) \quad (2.3.14)$$

Remarque 2.4 De manière générale, il faut noter que certains nœuds peuvent être confondus. Dans ce cas et par convention, on annule les $B_{j,d}$ qui présentent un dénominateur nul dans leur définition. On obtient ainsi des discontinuités dans le graphe des B-splines. Le nombre d'occurrence d'un point de raccordement est appelé multiplicité du point de raccordement.

Propriétés des B-splines

Avec les notations ci-dessus, on montre les propriétés suivantes :

1. $B_{j,d}(t) = 0$, $\forall t \notin [\text{nœud}_j, \text{nœud}_{j+d}]$. De plus $B_{j,d}(t) \geq 0$ sur $[\text{nœud}_j, \text{nœud}_{j+d}]$.
2. Les B-splines d'ordre d sont des fonctions polynomiales par morceaux de degré au plus $d - 1$.

Si les nœuds sont tous distincts, ce sont des fonctions de classe C^{d-2} . Lorsqu'un nœud est de multiplicité $1 \leq r \leq d - 1$, la B-spline n'est que $d - r - 1$ fois continûment dérivable en ce point.

²Pour une étude complète des fonctions splines, le lecteur peut se référer directement à l'ouvrage [Boor, 1978].

Si un nœud est de multiplicité d , il apparaît alors une discontinuité dans le graphe de la B-spline. A partir des relations de récurrence, on a :

$$\dot{B}_{j,d}(t) = (d-1) \left(\frac{B_{j,d-1}(t)}{\text{nœud}_{j+d-1} - \text{nœud}_j} - \frac{B_{j+1,d-1}(t)}{\text{nœud}_{j+d} - \text{nœud}_{j+1}} \right) \quad (2.3.15)$$

qui permet d'obtenir les dérivées successives des B-splines en tenant compte de la multiplicité des nœuds.

Exemple 2.1 On présente ici quelques exemples très simples de B-splines, représentés sur la figure 2.5, avec des nœuds entiers distincts (i.e. la suite de nœuds est $\{0, 1, 2, \dots\}$).

- La B-spline linéaire de degré 1 est composée de fonctions linéaires par morceaux :

$$B_{0,2}(t) = \begin{cases} t & \text{pour } 0 \leq t \leq 1 \\ 2-t & \text{pour } 1 \leq t \leq 2 \end{cases}$$

Elle est continue mais non dérivable.

- La B-spline quadratique de degré 2 est de classe C^1 :

$$B_{0,3}(t) = \begin{cases} \frac{t^2}{2} & \text{pour } 0 \leq t \leq 1 \\ \frac{-2t^2+6t-3}{2} & \text{pour } 1 \leq t \leq 2 \\ \frac{(3-t)^2}{2} & \text{pour } 2 \leq t \leq 3 \end{cases}$$

- La B-spline cubique est, quant à elle, de classe C^2 :

$$B_{0,4}(t) = \begin{cases} \frac{t^3}{6} & \text{pour } 0 \leq t \leq 1 \\ \frac{-3t^3+12t^2-12t+4}{6} & \text{pour } 1 \leq t \leq 2 \\ \frac{3t^3-24t^2+60t-44}{6} & \text{pour } 2 \leq t \leq 3 \\ \frac{(4-t)^3}{6} & \text{pour } 3 \leq t \leq 4 \end{cases}$$

Courbes spline

Définition 2.4 Etant donné $p + 1$ points C_0, \dots, C_p , appelés points de contrôle, une courbe spline d'ordre d est une combinaison linéaire des B-splines d'ordre d , i.e.

$$S_d(t) = \sum_{j=0}^p C_j B_{j,d}(t) \quad (2.3.16)$$

où p est défini par :

$$p = n_{knot}(d-1) - \sum_{k=1}^{n_{knot}-1} s_k$$

avec

- n_{knot} : le nombre d'intervalles $[\text{nœud}_j, \text{nœud}_{j+1}]$ ($j = 1, \dots, M-1$) de longueur non nulle,

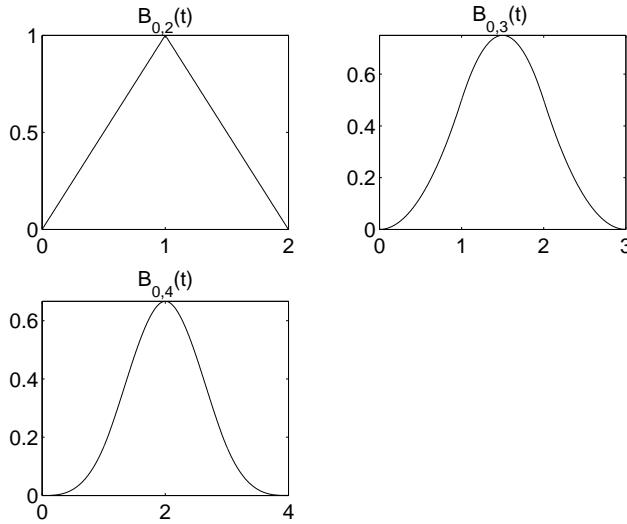


FIG. 2.5 – B-splines linéaire, quadratique et cubique

- s_k ($k = 1, \dots, n_{knot} - 1$) : le degré de continuité de chaque point de raccordement intérieur (i.e. dans l'intervalle ouvert $(\text{nœud}_0, \text{nœud}_M)$).

Remarque 2.5 Dans la plupart des applications, les points intérieurs de raccordement ont le même degré de continuité s , ce qui implique :

$$p = n_{knot}(d - 1 - s) + s.$$

Une propriété importante des fonctions splines concerne le polygone spline associé à la courbe S_d . On associe à S_d son polygone spline ou S-polygone qui est la ligne brisée joignant les points $\{C_0, \dots, C_p\}$. Le point $S_d(t)$ se trouve dans l'enveloppe convexe de son S-polygone (voir figure 2.6).

2.3.4 Transformation en un problème de programmation non linéaire

Dans la partie précédente, on a décrit la construction d'un espace de dimension finie utilisant les fonctions B-splines pour spécifier la trajectoire $z_i(t)$.

Soient un entier $d > l_i + 1$ ($d \in \mathbb{N}^*$) et $\{\delta_0 = t_{initial} < \delta_1 < \dots < \delta_{n_{knot}} = t_{final}\}$ une subdivision uniforme de $[t_{initial}, t_{final}]$ où n_{knot} est un entier non nul fixé. On construit alors à partir de cette subdivision la suite de nœuds permettant de définir une B-spline comme suit :

$$\{\text{nœud}_0 = \dots = \text{nœud}_{d-1} = \delta_0 < \text{nœud}_d = \delta_1 < \dots < \text{nœud}_{d-1+n_{knot}} = \dots = \text{nœud}_{2d-2+n_{knot}} = \delta_{n_{knot}}\} \quad (2.3.17)$$

La sortie plate z_i s'écrit alors comme une combinaison linéaire des fonctions B-splines d'ordre d :

$$z_i(t) = \sum_{j=0}^{d+n_{knot}-2} C_{j,i} B_{j,d}(t) \quad (2.3.18)$$

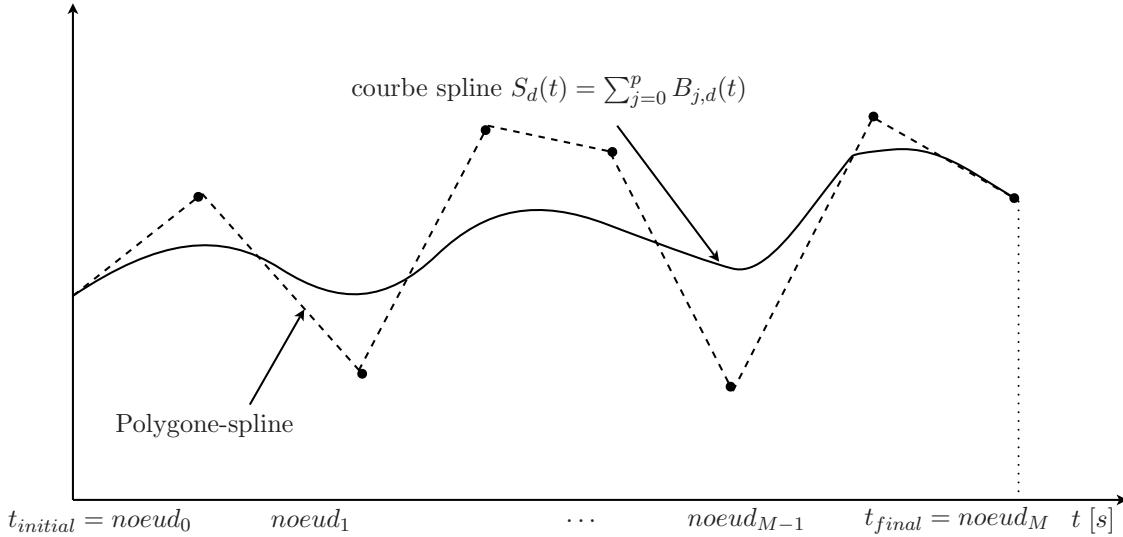


FIG. 2.6 – Exemple de fonction spline et de polygone spline

où $C_{j,i} \in \mathbb{R}^m$ désigne les vecteurs à optimiser contenant les points de contrôle, chacun de ces paramètres ayant un impact local.

Remarque 2.6 *L'ordre des B-splines d ainsi que la suite de nœuds sont choisis en fonction du nombre de variables que l'on veut optimiser, des contraintes aux bords et de l'ordre des dérivées l_i de la sortie plate intervenant dans l'expression du critère et des contraintes associées. Le problème principal réside dans le choix de l'entier n_{knot} afin de réaliser un bon compromis entre l'erreur de l'approximation et le temps de calcul.*

Après avoir exprimé les trajectoires de la sortie plate à l'aide des B-splines, le problème de commande optimale (2.3.11)-(2.3.12) peut facilement se transformer en un problème de programmation non linéaire. En effet, l'objectif est de résoudre le problème continu d'optimisation à l'aide d'un ordinateur qui est une machine discrète. La résolution numérique du problème d'optimisation nécessite une discrétisation du temps.

Soit

$$\{t_0 = t_{initial} < t_1 < \dots < t_{N_{ech}-1} = t_{final}\}$$

une subdivision uniforme³ de $[t_{initial}, t_{final}]$ où N_{ech} est un entier non nul fixé. Pour chaque instant t_k , $\forall k \in \{0, \dots, N_{ech} - 1\}$, le critère ainsi que les contraintes sont évalués. Le problème d'optimisation

³On peut utiliser des subdivisions à pas variables moyennant quelques modifications. En particulier, si les contraintes sont faibles sur l'horizon de planification, un large pas d'échantillonnage pourrait être utilisé afin de diminuer les temps de calcul.

continu se met sous la forme du problème de programmation non linéaire :

$$\min_{(C_{0,i}, \dots, C_{d+n_{\text{knot}}-2,i})} J = \sum_{k=0}^{N_{\text{ech}}-1} \mu_k L_i(\varphi_1(z_i(t_k), \dot{z}_i(t_k), \dots, z_i^{(l_i-1)}(t_k)), \varphi_2(z_i(t_k), \dot{z}_i(t_k), \dots, z_i^{(l_i)}(t_k)), t_k) \quad (2.3.19)$$

sous les contraintes $\forall k \in \{0, \dots, N_{\text{ech}} - 1\}$:

$$\left\{ \begin{array}{lcl} \varphi_1(z_i(t_{\text{initial}}), \dots, z_i^{(l_i-1)}(t_{\text{initial}})) & = & q_{i,\text{initial}} \\ \varphi_1(z_i(t_{\text{final}}), \dots, z_i^{(l_i-1)}(t_{\text{final}})) & = & q_{i,\text{final}} \\ \varphi_2(z_i(t_{\text{initial}}), \dots, z_i^{(l_i)}(t_{\text{initial}})) & = & u_{i,\text{initial}} \\ \varphi_2(z_i(t_{\text{final}}), \dots, z_i^{(l_i)}(t_{\text{final}})) & = & u_{i,\text{final}} \\ \varphi_2(z_i(t_k), \dots, z_i^{(l_i)}(t_k)) & \in & \mathcal{U}_i \\ d_{i,O_{m_i}}(t_k) & \geq & \rho_i + r_{m_i}, \quad \forall O_{m_i} \in \mathcal{Q}_{i,\text{occupe}} \end{array} \right. \quad (2.3.20)$$

Les poids μ_k dépendent de la méthode d'intégration choisie.

On a testé deux solveurs afin de résoudre le problème (2.3.19)-(2.3.20). Le premier, basé sur une pénalisation des contraintes [Joines et Houk, 1994], permet d'obtenir rapidement une solution mais les poids associés au viol des contraintes doivent être correctement ajustés. L'autre est basé sur les routines CFSQP (“Constrained Feasible Sequential Quadratic Programming”) développées à l'université du Maryland [Lawrance et al., 1997]. Ces routines utilisent un algorithme d'optimisation basé sur les méthodes de programmation séquentielle quadratique (en anglais “Sequential Quadratic Programming” (SQP)) avec un test de faisabilité. La programmation séquentielle quadratique est une technique itérative dans laquelle le critère est remplacé par une approximation quadratique et les contraintes par des approximations linéaires. Les routines CFSQP permettent de déterminer rapidement une solution vérifiant les contraintes (2.3.20), puis une solution quasi-optimale est calculée durant le temps alloué au calcul d'optimisation.

2.4 Algorithme en ligne de génération de trajectoire

Selon la distance à parcourir, le calcul d'une trajectoire complète de la configuration initiale à la configuration finale en une seule étape se révèle généralement impossible du fait de la complexité des calculs. En outre, l'environnement n'est que partiellement connu et doit être exploré au fur et à mesure que le robot effectue sa mission. Ainsi, la trajectoire doit être calculée graduellement au cours du temps. Ceci peut être réalisé par l'utilisation d'une stratégie basée sur un horizon glissant, dans laquelle des trajectoires sont calculées en résolvant des problèmes d'optimisation sur un horizon limité.

2.4.1 Principe de la planification sur un horizon glissant

L'algorithme proposé relaxe la contrainte d'atteinte de la configuration finale durant l'horizon de planification (i.e. équations (2.3.5) et (2.3.7)), permettant l'utilisation d'une stratégie de génération

de trajectoire sur un horizon glissant. On suppose, dans cette partie, que la commande finale est libre ou imposée à zéro.

Dans ce problème, on distingue deux types d'horizon (figure 2.7) :

- l'horizon de planification $T_p \in \mathbb{R}^+$ qui correspond à la longueur de l'intervalle de temps sur lequel le critère de performance est évalué.
- l'horizon de calcul $T_c \in \mathbb{R}^+$ ($T_c \leq T_p$) sur lequel le calcul d'une partie de la trajectoire est réalisé.

Les mises à jour du problème de commande optimale se font aux instants :

$$\tau_k = t_{initial} + kT_c, \quad k \in \mathbb{N} \quad (2.4.1)$$

Remarque 2.7 De la trajectoire et de la commande de référence calculées sur l'horizon T_p , seule la partie correspondante à l'horizon de calcul T_c sera prise en compte et stockée pour la suite.

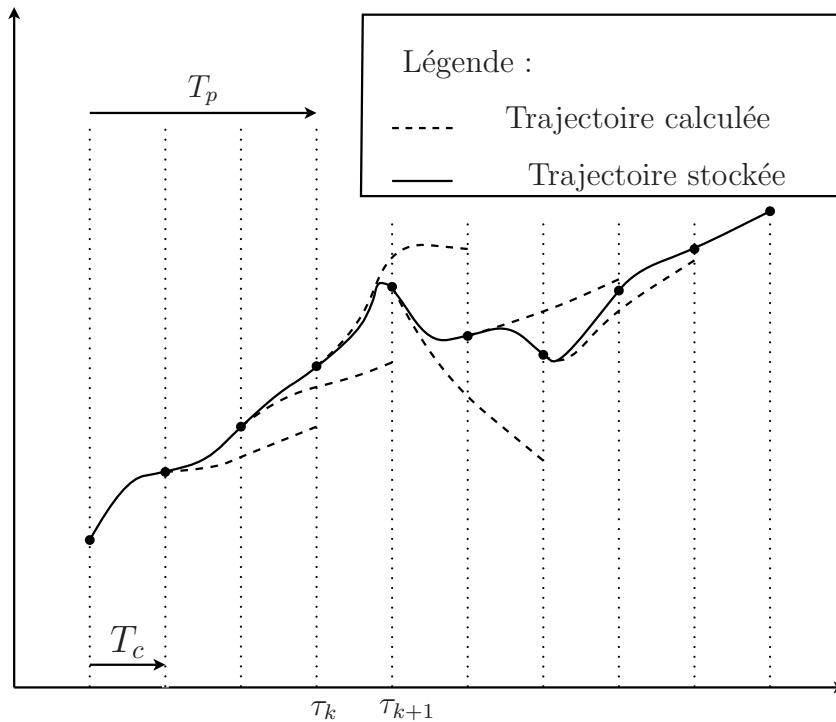


FIG. 2.7 – Horizons de calculs et de planification.

2.4.2 Mise en œuvre

La résolution du problème de planification de trajectoire sur un horizon glissant s'effectue en deux étapes :

- une phase d'initialisation avant que le robot ne se déplace,
- une phase de calculs itératifs pendant le déplacement du robot.

Phase d'initialisation

Avant qu'il ne se déplace, le robot calcule la trajectoire de référence commençant à l'instant $\tau_0 = t_{initial}$, notée $q_{i,ref}(t, \tau_0)$, et la commande correspondante, notée $u_{i,ref}(t, \tau_0)$, dans l'intervalle de temps $[\tau_0, \tau_0 + T_p]$ qui minimise le critère de performance choisi

$$J_{\tau_0} = \|q_i(\tau_0 + T_p, \tau_0) - q_{i,final}\|^2 + c \int_{\tau_0}^{\tau_0 + T_p} L_i(q_i(t, \tau_0), u_i(t, \tau_0), t) dt \quad (2.4.2)$$

avec $c \geq 0$ ⁴, sous les contraintes $\forall t \in [\tau_0, \tau_0 + T_p]$:

$$\left\{ \begin{array}{lcl} \dot{q}_i(t, \tau_0) & = & f_i(q_i(t, \tau_0), u_i(t, \tau_0)) \\ q_i(\tau_0, \tau_0) & = & q_{i,initial} \\ u_i(\tau_0, \tau_0) & = & u_{i,initial} \\ u_i(t, \tau_0) & \in & \mathcal{U}_i \\ d_{i,O_{m_i}}(t, \tau_0) & \geq & \rho_i + r_{m_i}, \quad \forall O_{m_i} \in \mathcal{O}_i(\tau_0) \end{array} \right. \quad (2.4.3)$$

Remarque 2.8 Sur chaque horizon de calcul, afin de caractériser toutes les variables (q_i, u_i, \dots), deux arguments sont utilisés. Le premier est l'argument conventionnel (le temps). Quant au second, il permet de mettre en lumière l'instant initial de l'intervalle de temps sur lequel la trajectoire et la commande sont optimisées.

Remarque 2.9 Le terme $\|q_i(\tau_0 + T_p, \tau_0) - q_{i,final}\|^2$, appelé partie terminale, privilégie dans le critère la prise en compte de l'état final et permet d'inciter le robot à atteindre la configuration finale $q_{i,final}$.

Seule la trajectoire et la commande optimale de référence sur l'intervalle de temps $t \in [\tau_0, \tau_1] = [t_{initial}, t_{initial} + T_c]$ sont stockées.

Phase de calculs itératifs

Pendant le déplacement du robot et jusqu'à ce qu'il ait atteint un voisinage de la configuration finale $q_{i,final}$, le problème de planification de trajectoire sur un horizon glissant revient à déterminer sur chaque intervalle de temps $[\tau_{k-1}, \tau_k]$ ($k > \mathbb{N}^*$), la trajectoire de référence commençant à l'instant τ_k , notée $q_{i,ref}(t, \tau_k)$, et la commande correspondante, notée $u_{i,ref}(t, \tau_k)$, dans l'intervalle de temps $[\tau_k, \tau_k + T_p]$ qui minimise le critère de performance :

$$J_{\tau_k} = \|q_i(\tau_k + T_p, \tau_k) - q_{i,final}\|^2 + c \int_{\tau_k}^{\tau_k + T_p} L_i(q_i(t, \tau_k), u_i(t, \tau_k), t) dt \quad (2.4.4)$$

⁴Notons que dans le cas d'une minimisation du temps de déplacement, le terme $\int_{\tau_0}^{\tau_0 + T_p} L_i(q_i(t, \tau_0), u_i(t, \tau_0), t) dt$ n'a pas de sens. Ainsi, le coefficient c est fixé à zéro. Par conséquent, le problème équivalent, dans le cadre d'une stratégie sur un horizon glissant, est la minimisation de la distance entre la position courante du robot et la position finale désirée.

avec $c \geq 0$, sous les contraintes $\forall t \in [\tau_k, \tau_k + T_p]$:

$$\dot{q}_i(t, \tau_k) = f_i(q_i(t, \tau_k), u_i(t, \tau_k)) \quad (2.4.5)$$

$$q_i(\tau_k, \tau_k) = q_{i,ref}(\tau_k, \tau_{k-1}) \quad (2.4.6)$$

$$u_i(\tau_k, \tau_k) = u_{i,ref}(\tau_k, \tau_{k-1}) \quad (2.4.7)$$

$$u_i(t, \tau_k) \in \mathcal{U}_i \quad (2.4.8)$$

$$d_{i,O_{m_i}}(t, \tau_k) \geq \rho_i + r_{m_i}, \quad \forall O_{m_i} \in \mathcal{O}_i(\tau_k) \quad (2.4.9)$$

Une fois ce problème résolu, la trajectoire et la commande optimale de référence sur l'intervalle de temps $t \in [\tau_k, \tau_{k+1}]$ ($k \in \mathbb{N}^*$) sont stockées.

Il est à noter que les contraintes (2.4.6)-(2.4.7) sur les conditions initiales font intervenir la configuration de référence $q_{i,ref}(\tau_k, \tau_{k-1})$ et la commande de référence $u_{i,ref}(\tau_k, \tau_{k-1})$ planifiées au cours de l'étape précédente. La figure 2.8 donne une représentation graphique du déroulement des calculs de planification.

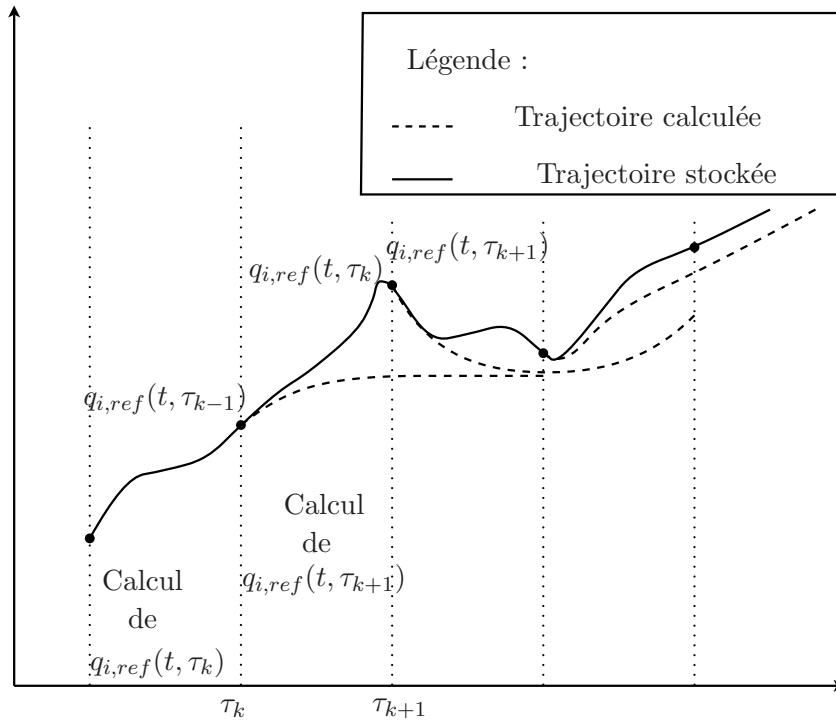


FIG. 2.8 – Mise en œuvre de la planification sur un horizon glissant

Remarque 2.10 Notons que le cadre classique de la commande à horizon glissant préconise d'appliquer uniquement la première valeur de la commande pour pouvoir faire face aux erreurs dans la modélisation ainsi qu'aux éventuelles perturbations. Ainsi, à chaque période d'échantillonnage, l'état et la commande courants du robot sont mesurés et le problème d'optimisation (2.4.2)-(2.4.3) est résolu.

Dans ce cas, les temps de calcul sont supposés négligeables. Cependant, en pratique, la puissance de calcul disponible est limitée, ce qui rend cette hypothèse fausse.

Ici, l'objectif n'est pas de faire face aux incertitudes et aux perturbations puisqu'à partir des trajectoires et des commandes de référence planifiées, une loi de commande robuste sera implémentée sur le robot pour éliminer l'impact de ces erreurs. Dans les chapitres suivants, plusieurs contrôleurs garantissant un suivi précis de la trajectoire planifiée seront développés.

Notre stratégie est, contrairement aux méthodes classiques de la commande à horizon glissant, de prendre en compte les temps de calcul.

Remarque 2.11 Les problèmes d'optimisation (2.4.2)-(2.4.3) et (2.4.4)-(2.4.9) sont résolus par la méthode décrite dans la Section 2.3 (i.e. utilisation des sorties plates, spécification de la sortie plate par une fonction spline, résolution du problème de programmation non linéaire).

Remarque 2.12 Dans le cas général, le choix des horizons T_p et T_c reste ouvert puisque l'optimalité d'un horizon est fortement liée à la complexité du problème. Le choix de l'horizon T_p influe sur la quantité d'informations fournies à l'algorithme et sur l'existence de solutions numériques au problème d'optimisation. Sa détermination doit prendre en considération les caractéristiques physiques du comportement, l'objectif à atteindre et les contraintes. Un compromis est à trouver entre une grande fenêtre de prédiction assurant la maîtrise sur un temps plus long et un horizon court garantissant des temps de calcul plus réalistes en vue d'une application en ligne. Augmenter la valeur de l'horizon et par conséquent augmenter le nombre de degrés de liberté, doit permettre a priori de résoudre des problèmes plus difficiles.

Quant à l'horizon de calcul T_c , il est nécessaire qu'il soit grand pour permettre la résolution du problème de commande optimale. Néanmoins, il doit être suffisamment inférieur à l'horizon de planification T_p pour éviter que la trajectoire générée durant l'horizon T_c ne s'approche trop près d'un obstacle inconnu situé juste en dehors du rayon de détection du robot. Ceci permet d'éliminer le risque de non solubilité des problèmes d'optimisation au cours des horizons suivants puisqu'il est possible que le robot n'ait pas assez de puissance pour éviter l'obstacle durant l'optimisation suivante. Ainsi, un compromis est à trouver entre une grande fenêtre de calculs permettant d'obtenir une résolution fine du problème de commande optimale et un horizon court garantissant une bonne réactivité, ce qui permet de faire face à des éléments imprévus⁵.

Remarque 2.13 Notons qu'il est envisageable de combiner l'algorithme de planification en ligne de trajectoire avec un algorithme d'évitement d'obstacle basé sur le concept de zone virtuelle déformable dont la formulation mathématique est présentée dans [Zapata et al., 2004, Gil-Pinto et al., 2006]. Cette approche permettrait l'implémentation de commandes d'évitement rapide d'obstacles.

⁵Il est possible de choisir un horizon T_c variable en construisant par exemple une grille de correspondance entre le temps nécessaire pour obtenir une solution sous-optimale acceptable et le nombre d'obstacles détectés.

Obstacle	Position abscisse (m)	Position ordonnée (m)	Rayon (m)
1	0.25	2.5	0.2
2	2.3	2.5	0.5
3	1.25	3	0.1
4	0.3	1	0.1
5	-0.5	1.5	0.3

TAB. 2.1 – Configuration des obstacles

2.5 Applications numériques sur un robot unicycle

Pour illustrer les deux algorithmes de génération de trajectoire hors ligne et en ligne, on effectue des tests dans un environnement complexe jonché d'obstacles. Le nombre, la position ainsi que la taille des obstacles sont générés de manière aléatoire. Dans les tests suivants, ils sont au nombre de cinq et sont décrits dans la Table 2.1. Le robot de type unicycle, dont la dynamique est donnée par l'équation (2.2.3), se situe à l'instant $t_{initial} = 0s$ à la configuration $q_{i,initial} = [0, 0, 90^\circ]^T$ à vitesse nulle et veut atteindre la configuration $q_{i,final} = [2, 5, 90^\circ]^T$ à une vitesse nulle le plus rapidement possible (i.e. minimisation du critère temporel $J = \int_{t_{initial}}^{t_{final}} dt$). La forme géométrique du robot est inscrite dans un cercle de rayon $\rho_i = 0.2m$. Les vitesses maximales linéaire et angulaire sont de $0.5m/s$ et de $5rad/s$ respectivement. Ainsi, l'ensemble des commandes admissibles du robot est

$$\mathcal{U}_i = \{(v_i(t), w_i(t)) \in \mathbb{R}^2 : |v_i(t)| \leq 0.5m/s, |w_i(t)| \leq 5rad/s\}$$

Les résultats suivants ont été obtenus en implémentant les programmes écrits en C sur un ordinateur Pentium IV 2.4 Ghz (192Mo de RAM). La routine d'optimisation utilisée est la routine CFSQP.

2.5.1 Exemple 1 : Connaissance complète de la carte

Lorsque la carte est connue totalement avant le déplacement du robot, une planification hors ligne est possible. Pour la résolution de ce problème, les paramètres donnés dans la Table 2.2 sont utilisés. Ainsi, avant le déplacement du robot, le problème de génération de trajectoire est résolu en utilisant la méthode décrite dans la Section 2.3 :

- sélection des sorties plates : x_i et y_i ,
- synthèse de la suite (2.3.17) de noeuds,
- construction des B-splines d'ordre d en utilisant la relation de récurrence (2.3.14),
- construction des dérivées première et seconde des B-splines d'ordre d en utilisant la relation de récurrence (2.3.15),

ordre des B-splines d	4
nombre d'intervalles de longueur non nulle de la suite de noeuds n_{knot}	15
nombre d'échantillons N_{ech} pour la discréétisation	100

TAB. 2.2 – Paramètres pour l'optimisation hors ligne de la trajectoire d'un robot

- paramétrisation de la sortie plate $[x_i, y_i]^T$ par une fonction spline, i.e.

$$[x_i(t), y_i(t)]^T = \sum_{j=0}^{d+n_{\text{knot}}-2} C_{j,i} B_{j,d}(t)$$

- détermination du temps optimal t_{final} et des paramètres de contrôle optimaux $C_{j,i}$ ($j = 0, \dots, d+n_{\text{knot}}-2$) en résolvant le problème de programmation non linéaire (2.3.19)-(2.3.20) par la routine d'optimisation CFSQP.

Etant donné que la planification s'effectue hors ligne, il est possible de laisser tourner l'algorithme pendant plusieurs minutes. Dans notre cas, le temps de calcul est de l'ordre de 4 minutes. L'algorithme donne un temps de parcours pour le robot de $t_{final} = 12.6s$. Les résultats sont donnés dans les figures 2.9-2.10. Afin d'augmenter la lisibilité des résultats de la figure 2.9, les obstacles sont augmentés du rayon du robot afin de pouvoir le considérer comme ponctuel (traits en pointillés). On peut remarquer que le robot évite les obstacles et que les commandes restent dans l'espace \mathcal{U}_i .

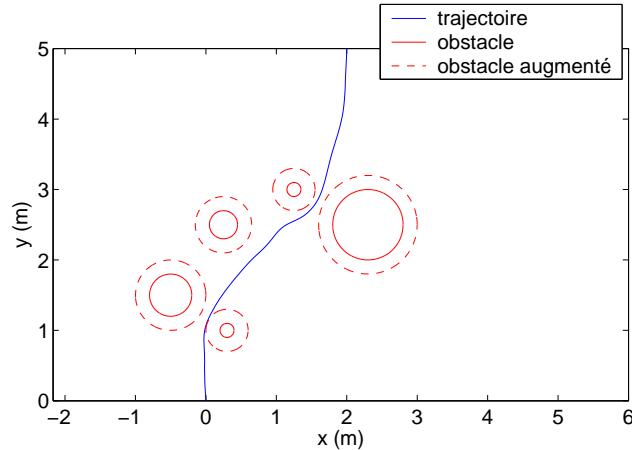


FIG. 2.9 – Trajectoire optimale d'un robot de type unicycle déterminée hors ligne

Bien que cette méthode donne des résultats quasi optimaux, la nécessité de connaître la carte complète et le temps de calcul élevé rendent sa mise en œuvre difficile. Ainsi, pour une application temps réel, cette méthode doit être combinée avec une stratégie de planification sur un horizon glissant.

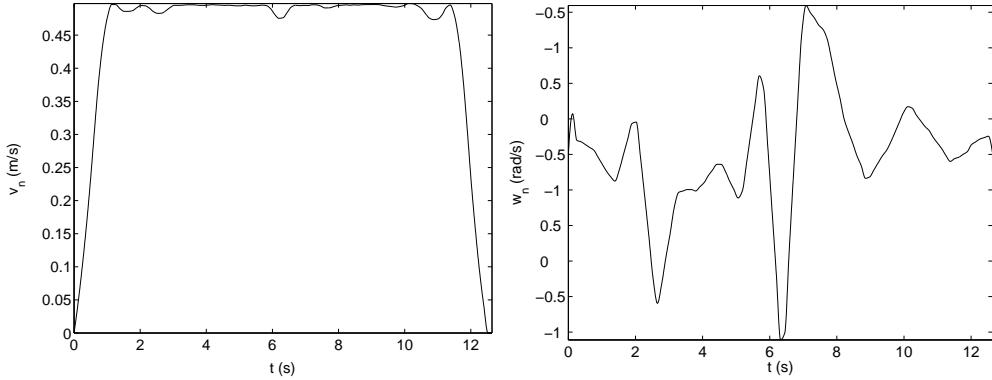


FIG. 2.10 – Commande optimale d'un robot de type unicycle déterminée hors ligne

horizon de planification T_p	2s
horizon de calculs T_c	1s
ordre des B-splines d	4
nombre d'intervalles de longueur non nulle de la suite de nœuds n_{knot}	6
nombre d'échantillons N_{ech} pour la discréétisation	20

TAB. 2.3 – Paramètres pour l'optimisation en ligne de la trajectoire d'un robot

2.5.2 Exemple 2 : Connaissance partielle de la carte

Nous considérons maintenant le scénario où l'environnement n'est pas complètement connu avant la mission mais seulement dans une zone autour du robot de rayon 2m (grandeur qui correspond à une bonne précision au niveau de la localisation des obstacles par les capteurs infra-rouges et la caméra du robot Pekee). Dans ce cas, la stratégie mise en place est une planification sur un horizon glissant décrite dans la section 2.4 dont les paramètres sont donnés dans la Table 2.3. Ainsi, on remplace le critère temporel précédent par le critère qui décrit la distance qui reste à parcourir pour atteindre l'objectif (i.e. $c = 0$). Les résultats sont donnés dans les figures 2.11-2.12. On peut remarquer que le robot évite les obstacles et que les commandes restent dans l'espace \mathcal{U}_i . Bien que le temps de parcours n'est plus minimal ($t_{final} = 15s$), cet algorithme peut être appliqué en ligne. En effet, le temps de résolution maximal de chaque problème d'optimisation (2.4.4)-(2.4.9) est d'environ 50ms et les obstacles sont pris en compte au fur et à mesure de leur découverte par le robot.

Remarque 2.14 Des vidéos donnant des résultats expérimentaux obtenus sur le robot Pekee pour différents environnements sont disponibles sur le site :

<http://syner.ec-lille.fr/robocoop/course/view.php?id=14>

Ces résultats illustrent l'efficacité de notre stratégie.

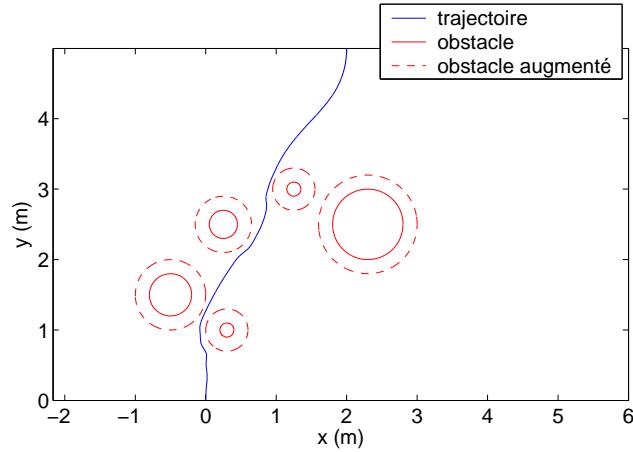


FIG. 2.11 – Trajectoire optimale d'un robot déterminée en ligne

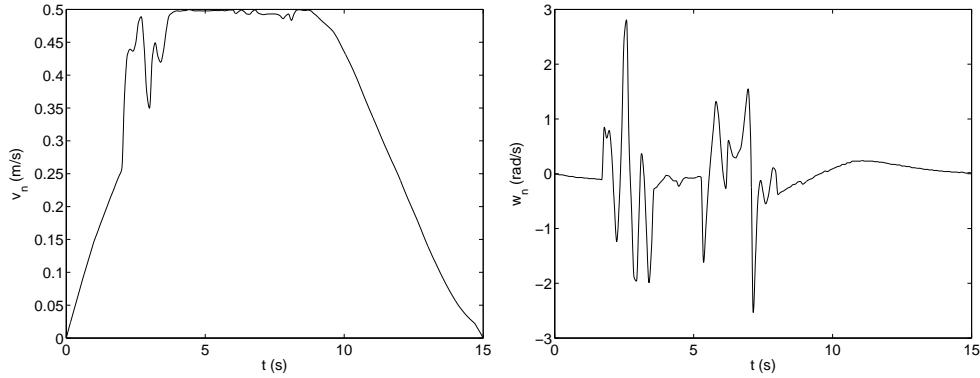


FIG. 2.12 – Commande optimale d'un robot déterminée en ligne

2.6 Conclusion

Ce chapitre traite du problème de planification de trajectoire admissible et sans collision pour un robot mobile non holonome dans un environnement inconnu. Après avoir abordé les concepts de platitude, les fonctions splines et la programmation non linéaire, un algorithme de génération de trajectoire hors ligne est construit pour un système plat dans un environnement connu. Puis, il est étendu de manière à pouvoir considérer un environnement inconnu, en utilisant une stratégie d'optimisation sur un horizon glissant. Des résultats numériques sur un robot de type unicycle permettent une comparaison des deux algorithmes hors ligne et en ligne, notamment en termes de temps de parcours et de temps de calcul.

Notre stratégie est suffisamment flexible pour pouvoir être adaptée au cadre multi-robots et sera la base de notre algorithme de génération de trajectoire pour la flottille de robots mobiles développé dans le prochain chapitre.

Chapitre 3

Planification de trajectoire : cadre multi-robots

3.1 Introduction

Dans le Chapitre 2, nous avons développé un algorithme de planification de trajectoire admissible et sans collision pour un robot mobile basé sur la platitude dans un environnement inconnu.

Dans ce chapitre, nous nous intéressons au problème de planification de trajectoire dans un cadre multi-robots. La présence de contraintes communes telles que l'évitement de collisions ou le maintien des liaisons de communication, permet une interaction entre robots. Ces interactions peuvent être mises à profit en introduisant des mécanismes de coordination afin de rendre cohérentes les actions des robots. Dans le cadre des processus de coordination, les robots échangent des informations notamment sur leurs intentions. Deux mécanismes de coordination peuvent être distingués selon la méthode de résolution des conflits via un superviseur (approche centralisée), ou individuellement par chaque robot (approche décentralisée).

La contribution de ce chapitre est de mettre en œuvre deux algorithmes de coordination afin de résoudre le problème de planification de trajectoire admissible et sans collision pour une flottille de robots mobiles évoluant dans un environnement inconnu.

Dans un premier temps, nous nous focaliserons sur une approche centralisée qui peut être vue comme une extension naturelle de l'algorithme de planification défini dans le cadre mono-robot. Bien que cette approche permette théoriquement d'obtenir les trajectoires optimales, elle présente de nombreux inconvénients. Le principal est le manque d'autonomie des véhicules relativement au superviseur. En effet, si le superviseur perd des informations ou s'il est détruit, aucune trajectoire ne peut être planifiée. En outre, toutes les informations transitent par celui-ci, ce qui implique des risques en termes de sécurité. Enfin, le nombre de calculs augmente très rapidement en fonction de la taille de la formation

et le problème devient rapidement insoluble pour une flottille de plus de cinq robots.

C'est pourquoi nous nous intéresserons, par la suite, à un algorithme décentralisé de planification en ligne, basé uniquement sur les informations locales disponibles. Le protocole permettra d'inclure la notion d'intention des robots qui peuvent provoquer un conflit (c'est-à-dire produire une collision ou perdre la liaison de communication), à partir de laquelle chaque robot élaborera sa propre trajectoire en prenant en considération les activités des autres robots. Non seulement cette implémentation distribuée augmentera l'autonomie des robots, mais aussi, réduira la complexité calculatoire par rapport à une implémentation centralisée.

Ces deux algorithmes seront testés et comparés avec d'autres approches décentralisées existantes ([Kuwata et al., 2006, Keviczky et al., 2006]) pour différents scénarii.

3.2 Description du problème dans le cadre multi-robots

Considérons une flottille de N_a robots mobiles non holonomes, identifiés par l'indice $i \in \{1, \dots, N_a\}$, dont le modèle cinématique est :

$$\dot{q}_i = f_i(q_i, u_i), \quad \forall i \in \{1, \dots, N_a\} \quad (3.2.1)$$

où $q_i \in \mathbb{R}^n$ représente l'état du $i^{\text{ème}}$ robot, $u_i \in \mathbb{R}^m$ est le vecteur de vitesses et $f_i : \mathbb{R}^n \times \mathbb{R}^m \rightarrow \mathbb{R}^n$ est une application suffisamment différentiable.

Le problème de planification consiste à calculer, de manière coopérative, pour les N_a robots, la trajectoire admissible et sans collision, joignant la configuration initiale $q_i(t_{initial}) = q_{i,initial}$ à la configuration finale $q_i(t_{final}) = q_{i,final}$ (avec des vitesses initiale $u_i(t_{initial}) = u_{i,initial}$ et finale $u_i(t_{final}) = u_{i,final}$ supposées nulles) et qui optimise un critère défini. En plus des contraintes individuelles qui ne font intervenir que le robot lui-même, définies dans le chapitre précédent (i.e. contraintes de non holonomie, contraintes (2.2.6) sur les vitesses admissibles et contraintes (2.2.8) d'évitement d'obstacles), les trajectoires planifiées devront respecter les contraintes, définies ci-après, qui couplent les états des robots.

D'une part, il est nécessaire de maintenir certaines liaisons de communication au cours du mouvement (par exemple pour un échange de stratégies, pour l'utilisation de capteurs décentralisés, pour maintenir la connexité de la formation, ...). Pour décrire ces contraintes de couplage, on peut définir le graphe de communication qui permettra de modéliser la structure topologique du réseau de communication entre les véhicules.

Définition 3.1 *Le graphe de communication* $(\mathcal{N}, \mathcal{A}, \mathcal{S})$ *est un graphe labellisé constitué :*

- *d'un ensemble fini $\mathcal{N} = \{1, \dots, N_a\}$ de N_a nœuds désignant les robots. Le $i^{\text{ème}}$ nœud représente le robot d'indice i auquel on associe le modèle (3.2.1).*

- d'un ensemble d'arcs $\mathcal{A} \subset \mathcal{N} \times \mathcal{N}$ caractérisant les liaisons de communication. La paire (i, p) appartient à \mathcal{A} lorsqu'il y a une liaison de communication entre les robots i et p .
- d'un ensemble de contraintes sur les arcs. Puisque la puissance disponible à bord des véhicules est limitée, la distance entre deux robots qui échangent des informations doit rester bornée.

Notons $d_{i,\text{com}} \in \mathbb{R}^+$ la portée de diffusion des informations du robot d'indice i . Cette caractéristique dépend de l'antenne d'émission et de la puissance embarquée à bord du robot (voir [Fraisse et al., 2005]). Pour chaque paire $(i, p) \in \mathcal{A}$, la distance inter-robots :

$$d_{ip}(t) = \sqrt{(x_i(t) - x_p(t))^2 + (y_i(t) - y_p(t))^2} \quad (3.2.2)$$

doit vérifier :

$$d_{ip}(t) \leq \min(d_{i,\text{com}}, d_{p,\text{com}}), \quad \forall t \geq 0 \quad (3.2.3)$$

Remarque 3.1 On se limite uniquement aux aspects géométriques et dynamiques et on ignore certains problèmes tels que les retards, les zones mortes ..., qui affectent également la qualité de communication.

Remarque 3.2 Le choix de la structure du graphe de communication peut également être considéré comme partie intégrante du processus d'optimisation. Dans ce mémoire, on ne se place pas dans un tel cas puisque le temps de calcul serait considérablement augmenté.

D'autre part, afin d'éviter les collisions inter-robots, les véhicules doivent rester à une distance de sécurité $d_{\text{sécurité}}$ les uns des autres au cours du temps¹.

Définition 3.2 On dit qu'il y a **éviter de collisions inter-robots** si la distance de séparation (3.2.2) pour chaque couple de véhicules $(i, p) \in \mathcal{N} \times \mathcal{N}$, $i \neq p$, est plus grande que la distance de sécurité $d_{\text{sécurité}} \geq \rho_i + \rho_p$, c'est-à-dire :

$$d_{ip}(t) > d_{\text{sécurité}}, \quad \forall t \geq 0 \quad (3.2.4)$$

Selon le nombre de robots et la distance à parcourir, le calcul des trajectoires complètes des configurations initiales aux configurations finales en une seule étape se révèle généralement impossible du fait de la complexité des calculs. En outre, comme dans le cas d'un seul robot, l'environnement n'est que partiellement connu et doit être découvert au fur et à mesure que les robots effectuent leur mission. Ainsi, en utilisant les notations du chapitre précédent, on applique une stratégie basée sur un horizon glissant, dans laquelle les trajectoires sont calculées en résolvant les problèmes d'optimisation relatifs aux phases d'initialisation et de calculs itératifs.

¹En général, la distance peut être différente suivant le couple de robots considéré. Dans ce mémoire, afin de simplifier les notations, nous considérons qu'elle est identique. Dans le cas contraire, il suffit de remplacer (3.2.4) par $d_{ip}(t) \geq d_{\text{sécurité}, ip}$ où $d_{\text{sécurité}, ip}$ sera à définir pour tous les couples $(i, p) \in \mathcal{N} \times \mathcal{N}$, $i \neq p$.

3.3 Architecture centralisée de planification

3.3.1 Mise en œuvre de l'algorithme centralisé

L'idée est d'étendre directement l'algorithme de génération en ligne de trajectoire, décrit dans le chapitre précédent, à une planification de plusieurs trajectoires à l'aide d'un superviseur (i.e. une unité indépendante ou un seul robot de la flottille). Par conséquent, deux étapes sont nécessaires (phase d'initialisation et phase de calculs itératifs).

Phase d'initialisation

Avant que les robots ne se déplacent, les configurations $q_{i,initial}$, les vitesses $u_{i,initial}$ et les zones obstacles $\mathcal{O}_i(t_{initial})$ de chaque véhicule ($i \in \mathcal{N}$) sont transmises au superviseur. Celui-ci calcule les trajectoires de référence commençant à l'instant $\tau_0 = t_{initial}$, notées $q_{i,ref}(t, \tau_0)$, et les commandes associées, notées $u_{i,ref}(t, \tau_0)$, dans l'intervalle de temps $[\tau_0, \tau_0 + T_p]$ qui minimise le critère de performance :

$$J_{\tau_0} = \sum_{i \in \mathcal{N}} \left(\|q_i(\tau_0 + T_p, \tau_0) - q_{i,final}\|^2 + c \int_{\tau_0}^{\tau_0 + T_p} L_i(q_i(t, \tau_0), u_i(t, \tau_0), t) dt \right) \quad (3.3.1)$$

avec $c \geq 0$, sous les contraintes : $\forall i \in \mathcal{N}, \forall t \in [\tau_0, \tau_0 + T_p]$:

$$\left\{ \begin{array}{lcl} \dot{q}_i(t, \tau_0) & = & f_i(q_i(t, \tau_0), u_i(t, \tau_0)) \\ q_i(\tau_0, \tau_0) & = & q_{i,initial} \\ u_i(\tau_0, \tau_0) & = & u_{i,initial} \\ u_i(t, \tau_0) & \in & \mathcal{U}_i \\ d_{i,O_{m_i}}(t, \tau_0) & \geq & \rho_i + r_{m_i}, \quad \forall O_{m_i} \in \mathcal{O}_i(t_{initial}) \\ d_{ip}(t, \tau_0) & \leq & \min(d_{i,com}, d_{p,com}), \quad \forall p \in \mathcal{N} \text{ tel que } (i, p) \in \mathcal{A} \\ d_{ip'}(t, \tau_0) & > & d_{\text{sécurité}}, \quad \forall p' \in \mathcal{N} \text{ tel que } p' \neq i \end{array} \right. \quad (3.3.2)$$

Une fois ce problème résolu, il transmet à chaque robot sa trajectoire et sa commande optimale de référence sur l'intervalle de temps $[t_{initial}, t_{initial} + T_c] = [\tau_0, \tau_1]$.

Phase de calculs itératifs

Pendant le déplacement des robots et jusqu'à ce qu'ils aient atteint un voisinage de leur configuration finale $q_{i,final}$, les véhicules transmettent au superviseur leur zone obstacle $\mathcal{O}_i(\tau_k)$. Puis, ce superviseur calcule, sur chaque intervalle de temps $[\tau_{k-1}, \tau_k = \tau_{k-1} + T_c]$ ($k > 0$), les trajectoires de référence commençant à l'instant τ_k , notées $q_{i,ref}(t, \tau_k)$, et les commandes associées, notées $u_{i,ref}(t, \tau_k)$, dans l'intervalle de temps $[\tau_k, \tau_k + T_p]$ qui minimise le critère de performance :

$$J_{\tau_k} = \sum_{i \in \mathcal{N}} \left(\|q_i(\tau_k + T_p, \tau_k) - q_{i,final}\|^2 + c \int_{\tau_k}^{\tau_k + T_p} L_i(q_i(t, \tau_k), u_i(t, \tau_k), t) dt \right) \quad (3.3.3)$$

avec $c \geq 0$, sous les contraintes $\forall i \in \mathcal{N}, \forall t \in [\tau_k, \tau_k + T_p]$:

$$\left\{ \begin{array}{lcl} \dot{q}_i(t, \tau_k) & = & f_i(q_i(t, \tau_k), u_i(t, \tau_k)) \\ q_i(\tau_k, \tau_k) & = & q_{i,ref}(\tau_k, \tau_{k-1}) \\ u_i(\tau_k, \tau_k) & = & u_{i,ref}(\tau_k, \tau_{k-1}) \\ u_i(t, \tau_k) & \in & \mathcal{U}_i \\ d_{i,O_{m_i}}(t, \tau_k) & \geq & \rho_i + r_{m_i}, \quad \forall O_{m_i} \in \mathcal{O}_i(\tau_k) \\ d_{ip}(t, \tau_k) & \leq & \min(d_{i,com}, d_{p,com}), \quad \forall p \in \mathcal{N} \text{ tel que } (i, p) \in \mathcal{A} \\ d_{ip'}(t, \tau_k) & > & d_{\text{sécurité}}, \quad \forall p' \in \mathcal{N} \text{ tel que } p' \neq i \end{array} \right. \quad (3.3.4)$$

où $q_{i,ref}(\tau_k, \tau_{k-1})$ et $u_{i,ref}(\tau_k, \tau_{k-1})$ sont les configurations et les commandes planifiées au cours de l'étape précédente.

Une fois ce problème résolu, la trajectoire et la commande optimale de référence sur l'intervalle de temps $t \in [\tau_k, \tau_{k+1}]$ ($k > 0$) sont transmises au robot correspondant.

Remarque 3.3 Les problèmes d'optimisation (3.3.1)-(3.3.2) et (3.3.3)-(3.3.4) sont résolus par la méthode décrite dans la Section 2.3 (i.e. utilisation des sorties plates de chaque robot de la flottille, spécification des sorties plates par une fonction spline, résolution du problème de programmation non linéaire). Cette stratégie permet de déterminer les vecteurs optimaux des points de contrôle pour les sorties plates de tous les robots en résolvant un problème d'optimisation de grande dimension (taille $mN_a(d + n_{knot} - 1)$).

3.3.2 Amélioration de l'algorithme centralisé

On propose ici un raffinement de l'algorithme centralisé afin de diminuer les temps de calcul et d'offrir ainsi une plus grande facilité de mise en œuvre. L'amélioration envisagée se situe au niveau de la résolution des problèmes d'optimisation (3.3.1)-(3.3.2) et (3.3.3)-(3.3.4). Plus précisément, nous souhaitons effectuer une initialisation rapide des vecteurs $C_{j,i} \in \mathbb{R}^m$ ($i \in \mathcal{N}, j \in 0, \dots, d + n_{knot} - 2$) contenant les points de contrôle pour les sorties plates de chaque robot, assez proche de leur valeur optimale². Une bonne initialisation permet effectivement d'arriver plus rapidement à la solution optimale du problème de minimisation.

L'amélioration de l'algorithme centralisé nécessite la vérification de l'hypothèse suivante :

Hypothèse 3.1 Chaque robot de la flottille présente les mêmes caractéristiques, c'est-à-dire le même rayon, le même ensemble des commandes admissibles et la même portée de diffusion des informations, i.e. $\forall i \in \mathcal{N}$:

$$\left\{ \begin{array}{lcl} \rho_i & = & \rho \\ \mathcal{U}_i & = & \mathcal{U} \\ d_{i,com} & = & d_{com} \end{array} \right.$$

²Dans la solution originelle du problème de minimisation, les vecteurs $C_{j,i}$ sont initialisés au vecteur nul.

L'idée de ce raffinement provient du fait que pour minimiser le temps de parcours, deux stratégies peuvent être envisagées pour l'évitement d'obstacles :

- l'obstacle est suffisamment petit pour que les robots l'évitent individuellement (figure 3.1),
- l'obstacle est trop grand et la flottille doit le contourner afin de préserver les communications (figure 3.2).

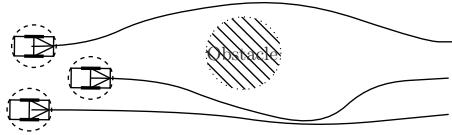


FIG. 3.1 – Stratégie d'évitement individuel d'obstacles.

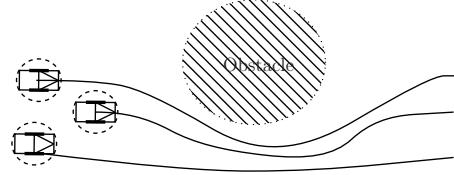


FIG. 3.2 – Stratégie d'évitement collectif d'obstacles.

Avant de résoudre le problème (3.3.1)-(3.3.2) ou les problèmes (3.3.3)-(3.3.4) permettant d'obtenir les trajectoires $q_{i,ref}(t, \tau_k)$ et les commandes $u_{i,ref}(t, \tau_k)$ optimales de référence, le superviseur planifie les trajectoires de la sortie plate associée au robot virtuel modélisant le centre de gravité de la flottille, dans une carte simplifiée de l'environnement. La procédure d'initialisation peut se décomposer comme suit :

- détermination de la configuration initiale du centre de gravité de la formation en faisant la moyenne des configurations initiales des différents robots, i.e. :

$$\begin{cases} \bar{q}(\tau_k, \tau_k) = \frac{1}{N_a} \sum_{i=1}^{N_a} q_i(\tau_k, \tau_k) \\ \bar{u}(\tau_k, \tau_k) = \frac{1}{N_a} \sum_{i=1}^{N_a} u_i(\tau_k, \tau_k) \end{cases}$$

- détermination de la carte simplifiée correspondant aux obstacles pour lesquels la stratégie d'évitement est collectif, i.e.

$$\left\{ O_m \in \bigcup_{i \in \mathcal{N}} \mathcal{O}_i(\tau_k) : 2(\rho + r_m) \geq d_{com} \right\}$$

Tous les obstacles n'appartenant pas à cet ensemble sont éliminés de la carte pour la phase d'initialisation.

- calcul de la trajectoire optimale du robot virtuel représentant le centre de gravité de la flottille dans la carte simplifiée en utilisant l'algorithme de planification en ligne décrit dans le chapitre précédent.

Les coefficients optimaux des vecteurs de points de contrôle de la sortie plate du robot virtuel sont stockés et servent comme initialisation du problème d'optimisation complet (3.3.1)-(3.3.2) ou (3.3.3)-(3.3.4).

Cette phase d'initialisation des vecteurs $C_{j,i} \in \mathbb{R}^m$, pendant laquelle la flottille est agrégée en un seul robot virtuel, correspond en fait à la décision prise sur la stratégie à adopter lors de l'évitement

d'obstacles (évitement individuel ou collectif). Elle permet de réduire les temps de calcul de l'ordre de 25% (voir le synoptique de l'algorithme centralisé de planification figure 3.3-3.8 pour les différentes étapes de la phase d'initialisation). Cependant, elle ne résout pas les problèmes au niveau du rôle central du superviseur.

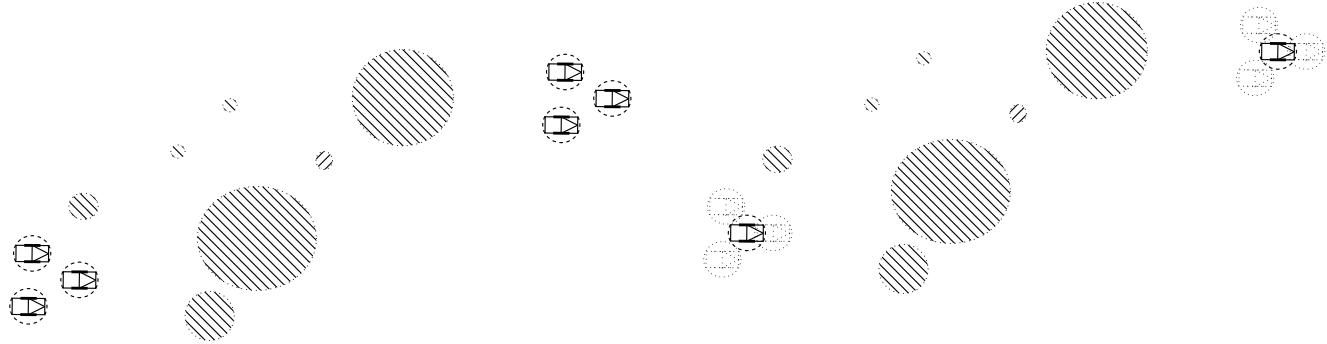


FIG. 3.3 – Carte initiale de l'environnement.

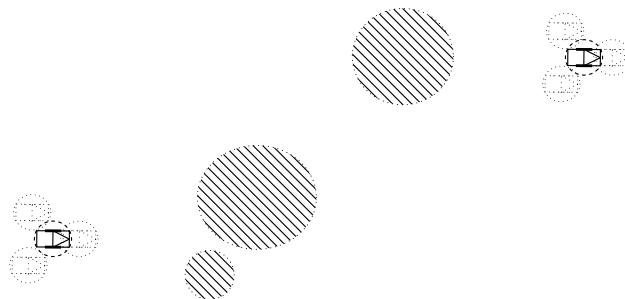


FIG. 3.5 – Détermination de la carte simplifiée.

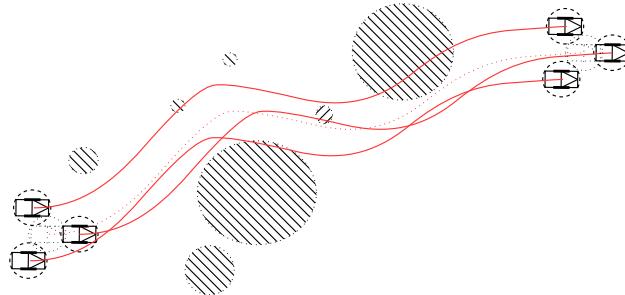


FIG. 3.7 – Initialisation des trajectoires des robots.

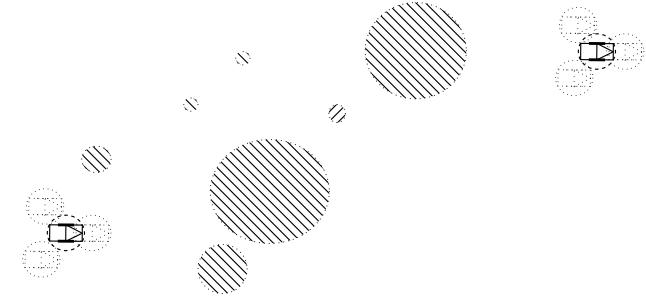


FIG. 3.4 – Détermination du centre de gravité de la flotteille.

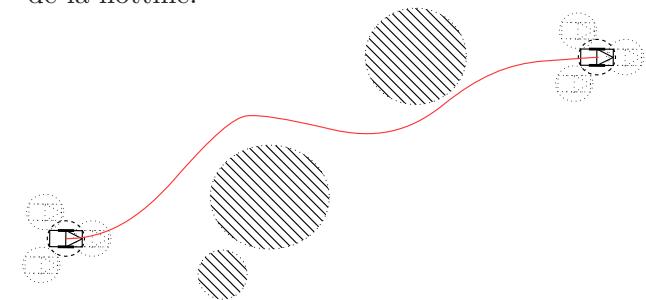


FIG. 3.6 – Calcul de la trajectoire optimale du robot virtuel.

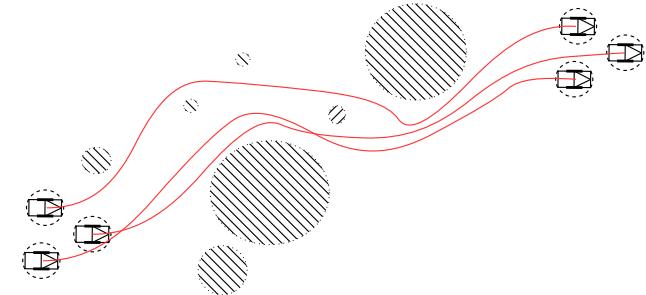


FIG. 3.8 – Calcul des trajectoires optimales des robots de la flotteille.

3.4 Architecture décentralisée de planification

Dans cette section, une implémentation décentralisée de planification en ligne de trajectoire permettant une coordination par ajustements mutuels (i.e. aucun véhicule n'a de contrôle sur les autres individus et le processus de décision est conjoint), est exposée. Chaque robot planifie uniquement sa propre trajectoire en utilisant des informations locales disponibles sur les robots risquant de provoquer un conflit. Après avoir défini les conflits entre les robots, l'algorithme décentralisé sera introduit.

3.4.1 Description des conflits

Introduisons les définitions suivantes.

Définition 3.3 A chaque instant τ_k , on définit la **zone d'accessibilité maximale** du robot d'indice i sur l'horizon de planification T_p , notée $\mathcal{R}_i(\tau_k) \subset \mathcal{Q}_i$, comme l'espace dans lequel le robot peut se trouver à l'instant $\tau_k + T_p$, i.e.

$$\mathcal{R}_i(\tau_k) = \{q_i(\tau_k + T_p) \in \mathcal{Q}_i : \forall t \in [\tau_k, \tau_k + T_p], \forall u_i(t) \in \mathcal{U}_i, q_i(t) \text{ est admissible}\} \quad (3.4.1)$$

Lorsque deux zones $\mathcal{R}_i(\tau_k)$ et $\mathcal{R}_p(\tau_k)$ ($(i, p) \in \mathcal{N} \times \mathcal{N}, i \neq p$) ne sont pas disjointes ou lorsque leur distance est trop faible, il est nécessaire de prendre en compte les contraintes qui couplent les états des robots i et p car il y a un risque de collision. De la même manière, lorsque la séparation de deux zones $\mathcal{R}_i(\tau_k)$ et $\mathcal{R}_p(\tau_k)$ ($(i, p) \in \mathcal{A}, i \neq p$) est trop importante, les robots risquent de perdre les liaisons de communication. Par conséquent, deux ensembles peuvent être définis.

Définition 3.4 L'ensemble $\mathcal{C}_{i,\text{collision}}(\tau_k) \subset \mathcal{N}$ des robots risquant de provoquer une collision avec le robot i sur l'horizon de planification T_p est :

$$\mathcal{C}_{i,\text{collision}}(\tau_k) = \left\{ p' \neq i, p' \in \mathcal{N} : \inf_{(q_i, q_{p'}) \in \mathcal{R}_i(\tau_k) \times \mathcal{R}_{p'}(\tau_k)} d_{ip'} \leq d_{\text{sécurité}} \right\} \quad (3.4.2)$$

Définition 3.5 L'ensemble $\mathcal{C}_{i,\text{com}}(\tau_k) \subset \mathcal{N}$ des robots risquant de provoquer une rupture de la liaison de communication avec le robot i sur l'horizon de planification T_p est :

$$\mathcal{C}_{i,\text{com}}(\tau_k) = \left\{ p \in \mathcal{N} : (i, p) \in \mathcal{A} \text{ et } \sup_{(q_i, q_p) \in \mathcal{R}_i(\tau_k) \times \mathcal{R}_p(\tau_k)} d_{ip} \geq \min(d_{i,\text{com}}, d_{p,\text{com}}) \right\} \quad (3.4.3)$$

Remarque 3.4 Dans le cas des robots de type unicycle ou de type voiture, il est possible de simplifier les calculs en approximant les zones $\mathcal{R}_i(\tau_k)$ ($i \in \mathcal{N}$) par des cercles de rayon $v_{i,\text{max}}T_p$ et de centre $[x_i(\tau_k), y_i(\tau_k)]^T$ où $v_{i,\text{max}}$ représente la vitesse linéaire maximale du véhicule. Ceci correspond à une stratégie du pire cas.

La définition (3.4.2) de l'ensemble $\mathcal{C}_{i,\text{collision}}(\tau_k)$ peut ainsi être simplifiée par :

$$\mathcal{C}_{i,\text{collision}}(\tau_k) = \{p' \neq i, p' \in \mathcal{N} : d_{ip'}(\tau_k) \leq d_{\text{sécurité}} + (v_{i,\text{max}} + v_{p',\text{max}})T_p\} \quad (3.4.4)$$

Similairement, la condition (3.4.3) est remplacée par :

$$\mathcal{C}_{i,com}(\tau_k) = \{p \in \mathcal{N} : (i, p) \in \mathcal{A} \text{ et } d_{ip}(\tau_k) + (v_{i,max} + v_{p,max})T_p \geq \min(d_{i,com}, d_{p,com})\} \quad (3.4.5)$$

Pour illustrer nos propos, un exemple est donné dans la figure 3.9. Dans cet exemple, à l'instant τ_k , le Robot 1 est en conflit avec le Robot 2 (risque de collision pendant l'horizon de planification) ainsi qu'avec le Robot 4 (risque de rupture de la liaison de communication). Le Robot 3 ne pose aucun problème et ne sera pas pris en compte dans le problème de planification du Robot 1.

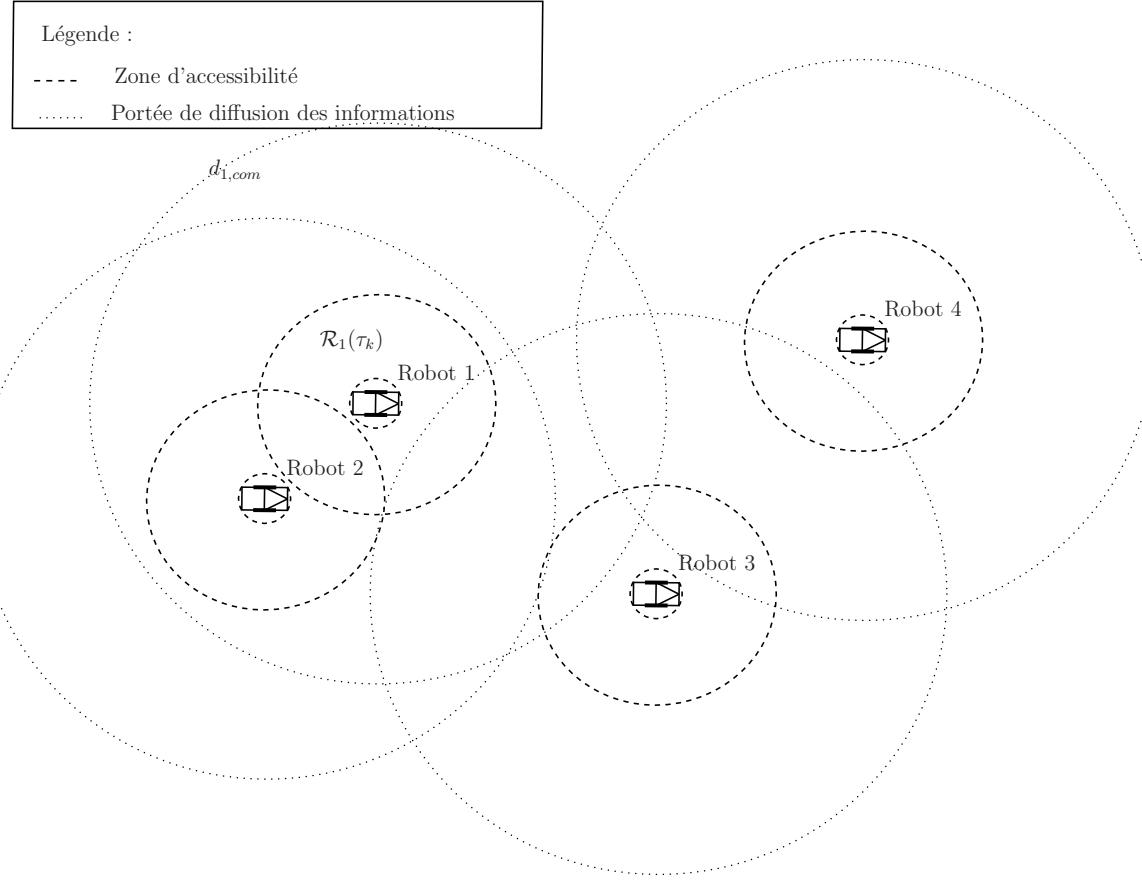


FIG. 3.9 – Description des zones de conflits possibles.

Définition 3.6 L'ensemble $\mathcal{C}_i(\tau_k) \subset \mathcal{N}$ risquant de provoquer un conflit avec le robot i sur l'horizon de planification T_p est l'union des ensembles $\mathcal{C}_{i,collision}(\tau_k)$ et $\mathcal{C}_{i,com}(\tau_k)$. C'est-à-dire,

$$\mathcal{C}_i(\tau_k) = \mathcal{C}_{i,collision}(\tau_k) \cup \mathcal{C}_{i,com}(\tau_k)$$

Remarque 3.5 Pour tout véhicule d'indice p n'appartenant pas à l'ensemble $\mathcal{C}_i(\tau_k)$, les contraintes d'évitement de collisions inter-robots et de maintien des communications, couplant les états des robots (i, p) sont automatiquement satisfaites durant l'intervalle de temps $[\tau_k, \tau_k + T_p]$. Par conséquent, à l'instant τ_k , le véhicule d'indice i doit uniquement connaître les intentions des robots appartenant à l'ensemble $\mathcal{C}_i(\tau_k)$.

3.4.2 Principe de l'algorithme décentralisé

Afin de générer de manière décentralisée des trajectoires admissibles et sans collision vérifiant également les contraintes de maintien des liaisons de communication, à chaque mise à jour :

$$\tau_k = t_{initial} + kT_c, \quad k \in \mathbb{N}^+$$

les robots planifient, individuellement, une nouvelle trajectoire. Chaque robot dispose donc d'un algorithme de planification en ligne de trajectoire. L'idée est de mettre en place un mécanisme de coordination par ajustements mutuels où certains robots s'accordent en vue de planifier leur trajectoire. Par conséquent, les robots doivent échanger leurs intentions.

Prenons le cas du robot d'indice i . Afin de générer sa trajectoire de référence $q_{i,ref}(t, \tau_k)$ admissible et sans collision, il doit connaître les intentions de tous les robots $p \in \mathcal{C}_i(\tau_k)$. Deux difficultés demeurent :

- la définition d'une trajectoire intuitive unique pour chaque robot,
- la cohérence entre la trajectoire intuitive et la trajectoire optimale de référence.

Afin d'éviter une architecture de type "meneur-suiveur" où une résolution séquentielle serait adoptée et d'éviter une dégradation des propriétés de décentralisation de l'algorithme, la solution envisagée est de considérer dans un premier temps le problème de planification de trajectoire en l'absence des contraintes qui couplent l'état des robots. Le résultat de ce problème correspondrait ainsi à la trajectoire intuitive du robot. Ceci implique une unicité au niveau de sa définition. En effet, cette trajectoire intuitive sera la même dans tous les problèmes de commande optimale dans lesquels elle intervient.

Par conséquent, pour chaque robot $i \in \mathcal{N}$, les phases de l'algorithme de planification en ligne de trajectoire (i.e. initialisation et calculs itératifs) sont subdivisées en deux grandes étapes :

- ★ étape 1 : détermination d'une trajectoire intuitive (trajectoire respectant uniquement les contraintes individuelles),
- ★ étape 2 : détermination de la trajectoire optimale de référence en fonction des informations échangées avec les véhicules appartenant à l'ensemble $\mathcal{C}_i(\tau_k)$.

Ainsi, trois types d'horizon sont utilisés :

- l'horizon d'intuition $T_d \in \mathbb{R}^+$ sur lequel la trajectoire intuitive du robot est calculée,
- l'horizon de planification $T_p \in \mathbb{R}^+$ ($T_p \leq T_d$) qui correspond à la longueur de l'intervalle de temps sur lequel le critère de performance ainsi que l'ensemble des contraintes sont évalués,
- l'horizon de calcul $T_c \in \mathbb{R}^+$ ($T_c \leq T_p$) sur lequel le calcul de la trajectoire est réalisé.

Pour chaque robot i , on distingue les trajectoires suivantes :

- $\hat{q}_i(t, \tau_k)$ est la trajectoire intuitive du robot i commençant à l'instant τ_k avec $t \in [\tau_k, \tau_k + T_d]$.
Cette trajectoire respecte uniquement les contraintes individuelles.

- $q_{i,ref}(t, \tau_k)$ est la trajectoire optimale planifiée du robot i commençant à l'instant τ_k avec $t \in [\tau_k, \tau_k + T_p]$ qui respecte l'ensemble des contraintes.

Les commandes correspondantes sont notées $\hat{u}_i(t, \tau_k)$ et $u_{i,ref}(t, \tau_k)$.

Pour un robot i ($i \in \mathcal{N}$), les deux étapes peuvent s'écrire sous la forme d'un problème de commande optimale.

A l'étape 1, on associe le problème $\widehat{P}_i(\tau_k)$ qui consiste en la détermination de la commande $\hat{u}_i(t, \tau_k)$ et de la trajectoire $\hat{q}_i(t, \tau_k)$, i.e.

Problème $\widehat{P}_i(\tau_k)$:

$$\min_{\hat{q}_i(t, \tau_k), \hat{u}_i(t, \tau_k)} \|\hat{q}_i(\tau_k + T_d, \tau_k) - q_{i,final}\|^2 + c \int_{\tau_k}^{\tau_k + T_d} L_i(\hat{q}_i(t, \tau_k), \hat{u}_i(t, \tau_k), t) dt$$

sous les contraintes $\forall t \in [\tau_k, \tau_k + T_d]$:

$$\left\{ \begin{array}{lcl} \dot{\hat{q}}_i(t, \tau_k) & = & f_i(\hat{q}_i(t, \tau_k), \hat{u}_i(t, \tau_k)) \\ \hat{q}_i(\tau_k, \tau_k) & = & q_{i,ref}(\tau_k, \tau_{k-1}) \\ \hat{u}_i(\tau_k, \tau_k) & = & u_{i,ref}(\tau_k, \tau_{k-1}) \\ \hat{u}_i(t, \tau_k) & \in & \mathcal{U}_i \\ \hat{d}_{i,O_{m_i}}(t, \tau_k) & \geq & \rho_i + r_{m_i}, \quad \forall O_{m_i} \in \mathcal{O}_i(\tau_k) \end{array} \right. \quad (3.4.6)$$

Le terme $\hat{d}_{i,O_{m_i}}(t, \tau_k)$ représente la distance entre la trajectoire intuitive $\hat{q}_i(t, \tau_k)$ et les obstacles détectés O_{m_i} , i.e. :

$$\hat{d}_{i,O_{m_i}}(t, \tau_k) = \sqrt{(\hat{x}_i(t, \tau_k) - X_{m_i})^2 + (\hat{y}_i(t, \tau_k) - Y_{m_i})^2}$$

Remarque 3.6 Au cours de la phase d'initialisation, c'est-à-dire avant le déplacement des robots, on note par convention :

$$\left\{ \begin{array}{lcl} \tau_{-1} & = & t_{initial} \\ q_{i,ref}(\tau_0, \tau_{-1}) & = & q_{i,initial} \\ u_{i,ref}(\tau_0, \tau_{-1}) & = & u_{i,initial} \end{array} \right.$$

A l'étape 2, on associe le problème $P_i^*(\tau_k)$ qui consiste en la détermination de la commande optimale $u_{i,ref}(t, \tau_k)$ et de la trajectoire optimale planifiée $q_{i,ref}(t, \tau_k)$, i.e. :

Problème $P_i^*(\tau_k)$:

$$\min_{q_i(t, \tau_k), u_i(t, \tau_k)} \|q_i(\tau_k + T_p, \tau_k) - q_{i,final}\|^2 + c \int_{\tau_k}^{\tau_k + T_p} L_i(q_i(t, \tau_k), u_i(t, \tau_k), t) dt$$

sous les contraintes $\forall t \in [\tau_k, \tau_k + T_p]$:

$$\dot{q}_i(t, \tau_k) = f_i(q_i(t, \tau_k), u_i(t, \tau_k)) \quad (3.4.7)$$

$$q_i(\tau_k, \tau_k) = q_{i,ref}(\tau_k, \tau_{k-1}) \quad (3.4.8)$$

$$u_i(\tau_k, \tau_k) = u_{i,ref}(\tau_k, \tau_{k-1}) \quad (3.4.9)$$

$$u_i(t, \tau_k) \in \mathcal{U}_i \quad (3.4.10)$$

$$d_{i,O_m i}(t, \tau_k) \geq \rho_i + r_{m_i}, \quad \forall O_m \in \mathcal{O}_i(\tau_k) \quad (3.4.11)$$

$$\hat{d}_{ip}(t, \tau_k) \leq \min(d_{i,com}, d_{p,com}) - \xi, \quad \forall p \in \mathcal{C}_{i,com}(\tau_k) \quad (3.4.12)$$

$$\hat{d}_{ip'}(t, \tau_k) > d_{\text{sécurité}} + \xi, \quad \forall p' \in \mathcal{C}_{i,collision}(\tau_k) \quad (3.4.13)$$

$$\hat{d}_{ii}(t, \tau_k) \leq \xi \quad (3.4.14)$$

où $\xi \in \mathbb{R}^+$ représente la déformation admissible entre la trajectoire intuitive et la trajectoire optimale planifiée. Les termes $\hat{d}_{ip}(t, \tau_k)$, $\hat{d}_{ip'}(t, \tau_k)$ et $\hat{d}_{ii}(t, \tau_k)$ représentent la distance entre la trajectoire du robot i à optimiser et la trajectoire intuitive du robot p , p' et i respectivement, c'est-à-dire :

$$\begin{cases} \hat{d}_{ip}(t, \tau_k) = \sqrt{(x_i(t, \tau_k) - \hat{x}_p(t, \tau_k))^2 + (y_i(t, \tau_k) - \hat{y}_p(t, \tau_k))^2} \\ \hat{d}_{ip'}(t, \tau_k) = \sqrt{(x_i(t, \tau_k) - \hat{x}_{p'}(t, \tau_k))^2 + (y_i(t, \tau_k) - \hat{y}_{p'}(t, \tau_k))^2} \\ \hat{d}_{ii}(t, \tau_k) = \sqrt{(x_i(t, \tau_k) - \hat{x}_i(t, \tau_k))^2 + (y_i(t, \tau_k) - \hat{y}_i(t, \tau_k))^2} \end{cases} \quad (3.4.15)$$

Notons qu'une contrainte supplémentaire, i.e. équation (3.4.14), est introduite dans le problème d'optimisation. Elle permet d'établir un degré de correspondance entre la trajectoire optimale de référence $q_{i,ref}(t, \tau_k)$ et la trajectoire intuitive $\hat{q}_i(t, \tau_k)$ connue par les robots en conflit avec le robot i . Ainsi, afin de satisfaire les contraintes qui couplent les états des robots en conflit, on autorise un écart d'au plus ξ par rapport à la trajectoire intuitive.

Dans le nouveau problème d'optimisation, seules les informations sur les véhicules en conflit avec le robot i sont utilisées. Ainsi, les ressources requises en communication sont nettement diminuées par rapport au problème initial. En outre, afin de garantir l'évitement de collisions inter-robots et de préserver les communications, l'incertitude ξ , due à l'écart entre la trajectoire inititive $\hat{q}_p(t, \tau_k)$ et la trajectoire optimale planifiée $q_{p,ref}(t, \tau_k)$, est prise en compte (voir les équations (3.4.12)-(3.4.13)).

Remarque 3.7 *Afin de pouvoir résoudre le problème $P_i^*(\tau_k)$, une communication bi-directionnelle entre le robot i et les robots p appartenant à l'ensemble $\mathcal{C}_i(\tau_k)$ doit être mise en place pour permettre l'échange des trajectoires intuitives.*

3.4.3 Mise en œuvre de l'algorithme

Le synoptique de l'algorithme décentralisé ([Defoort et al., 2007a, Defoort, 2007]) de planification pour chaque robot i ($i \in \mathcal{N}$) est donné dans la figure 3.10.

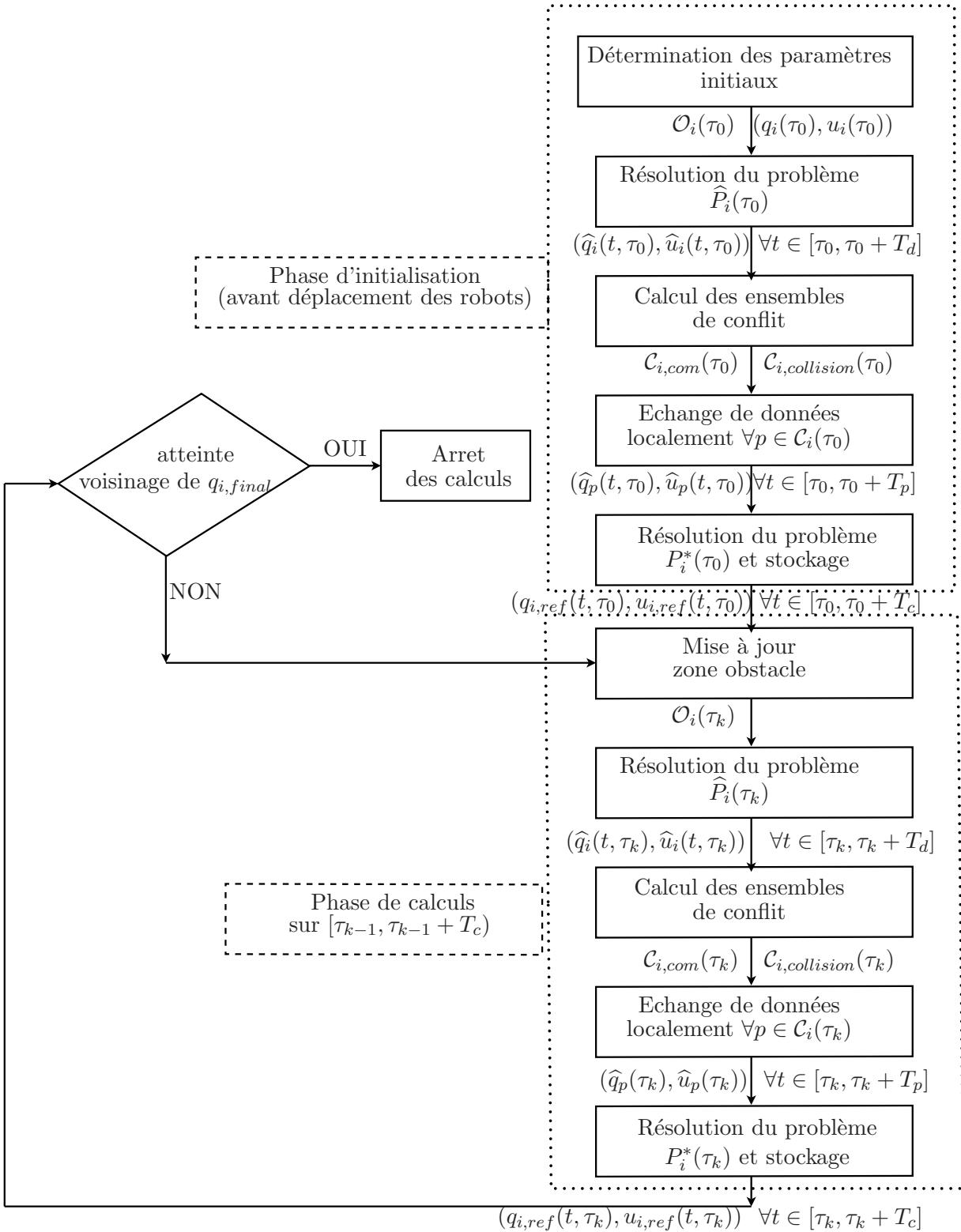


FIG. 3.10 – Synoptique de l'algorithme décentralisé de planification de trajectoire.

Remarque 3.8 Le choix des horizons T_c , T_p et T_d reste ouvert puisqu'ils sont corrélés. Cependant, quelques réflexions peuvent être émises à ce sujet. L'horizon de calcul T_c , choisi le plus petit possible afin de garantir une bonne réactivité face à des éléments imprévus, doit être suffisamment grand pour permettre la résolution des problèmes de commande optimale $\hat{P}_i(\tau_k)$, $P_i^*(\tau_k)$, les mises à jour et les transmissions d'informations. L'horizon T_p est supérieur à T_c pour éviter les blocages dus aux contraintes qui couplent les états des robots (évitement de collisions inter-robots, maintien des communications) dans les problèmes d'optimisation au cours des horizons suivants. Cependant, il doit être nettement inférieur à T_d afin d'avoir suffisamment d'autorité pour modifier la trajectoire intuitive. Enfin, l'horizon d'intuition T_d est lié à la portée des capteurs pour la localisation et la détection des obstacles.

Remarque 3.9 Rappelons pour mémoire que les problèmes d'optimisation $\hat{P}_i(t_k)$ et $P_i^*(t_k)$ ($k \in \mathbb{N}$) sont résolus par la méthode décrite dans la Section 2.3 (i.e. utilisation des sorties plates, spécification de la sortie plate par une fonction spline, résolution du problème de programmation non linéaire).

Les avantages de cet algorithme, mis en lumière dans les prochaines sections, sont nombreux :

- ★ mécanisme de coordination sûr (absence de superviseur et de robot meneur),
- ★ forte décentralisation des informations (chaque robot connaît uniquement sa trajectoire optimale et son objectif),
- ★ faible temps de calcul,
- ★ trajectoire quasi-optimale,
- ★ ressources requises au niveau des communications faibles (échange local d'informations).

3.5 Présentation d'autres algorithmes décentralisés

Afin de pouvoir évaluer précisément les performances de l'algorithme décentralisé de planification décrit auparavant, nous allons présenter d'autres approches aboutissant à la génération en ligne des trajectoires de la flottille de robots de manière plus ou moins décentralisée. Récemment, deux algorithmes, adaptables à notre problème de planification, ont été proposés dans [Keviczky et al., 2006, Kuwata et al., 2006]³.

Approche “faiblement décentralisée” de [Keviczky et al., 2006]

Nous n'allons pas ici entrer dans les détails mathématiques de l'algorithme de [Keviczky et al., 2006]. Nous allons plutôt exposer l'idée principale. L'algorithme proposé utilise un mécanisme de coordination par ajustements mutuels où certains robots s'accordent en vue de planifier leur trajectoire en

³A noter que ces deux algorithmes ont été proposés pour la résolution du problème de planification décentralisée de trajectoire pour des systèmes linéaires. Néanmoins, en utilisant la méthode de résolution du problème de commande optimale, développée dans la Section 2.3 (i.e. utilisation des sorties plates, spécification de la sortie plate par une fonction spline, résolution du problème de programmation non linéaire), il est possible de les étendre au cas non linéaire.

échangeant leurs intentions. Les différences avec notre algorithme se situent au niveau de ce qui est échangé entre les robots et au niveau du calcul de la trajectoire intuitive.

Dans le cas de l'algorithme de [Keviczky et al., 2006], les calculs de planification ne sont plus décomposés en deux étapes. En effet, sur chaque intervalle de temps $[\tau_{k-1}, \tau_k]$, seule la configuration $q_p(\tau_k, \tau_{k-1})$ est échangée entre les robots en conflit. Ceci a pour résultat une diminution de la quantité d'informations échangées. A partir des informations reçues par les véhicules en conflit avec le robot i ($q_p(\tau_k, \tau_{k-1}), \forall p \in \mathcal{C}_i(\tau_k)$), celui-ci planifie sa trajectoire optimale de référence $q_{i,\text{ref}}(t, \tau_k)$ et la trajectoire intuitive $\hat{q}_p(t, \tau_k)$ des robots $p \in \mathcal{C}_i(\tau_k)$.

Deux problèmes se posent en utilisant une telle stratégie. D'une part, l'absence de contrainte sur la correspondance entre la trajectoire intuitive et la trajectoire réellement planifiée implique que les trajectoires optimales planifiées localement ne respectent pas forcément les contraintes qui couplent les états des robots. En effet, que faire lorsque la trajectoire intuitive est très différente de la trajectoire optimale planifiée ? D'autre part, le calcul des trajectoires intuitives des robots p par le véhicule i augmente les temps de calcul et diminue le niveau de décentralisation car les objectifs $q_{p,\text{final}}, \forall p \in \mathcal{C}_i(\tau_k)$, doivent être connus du véhicule i .

Notons que dans le cas critique où tous les robots sont en conflit, cette approche est semblable à une approche centralisée.

Approche “meneur / suiveur” de [Kuwata et al., 2006]

L'idée n'est plus de mettre en place un mécanisme de coordination par ajustements mutuels, mais par “leadership”, où une relation hiérarchique existe entre les véhicules (certains robots exercent alors un contrôle sur d'autres).

Dans un premier temps, un ordre de résolution est déterminé afin d'établir les relations de priorité entre les robots de la flottille. Ainsi, sur chaque intervalle de temps $[\tau_{k-1}, \tau_k]$, le robot ayant la priorité la plus forte planifie sa trajectoire optimale sans tenir compte des autres robots. Une fois sa trajectoire planifiée, un autre robot, ayant la deuxième priorité la plus élevée, génère sa trajectoire optimale en prenant en compte les trajectoires des robots ayant une priorité plus élevée que la sienne. Et ainsi de suite. La résolution se fait donc de manière séquentielle. L'avantage principal de cette approche est sa facilité de mise en œuvre. Cependant, du fait de l'absence de retour d'informations du suiveur vers le meneur, la structure est mal adaptée à tous les problèmes sur les suiveurs. Enfin, lorsque le meneur est à puissance maximale, la marge de manœuvre du suiveur pour satisfaire l'ensemble des contraintes est très réduite.

3.6 Résultats numériques comparatifs

Pour illustrer les algorithmes de génération en ligne de trajectoire pour une flottille de robots mobiles, on effectue des tests pour différents scénarii. La forme géométrique de chaque robot i est

Nombre de robots N_a	2
horizon de planification T_p	2s
horizon de calculs T_c	0.5s
horizon d'intuition T_d	2s
déformation admissible ξ	0.25
ordre des B-splines d	4
nombre d'intervalles de longueur non nulle de la suite de nœuds n_{knot}	3
nombre d'échantillons N_{ech} pour la discréétisation	20

TAB. 3.1 – Paramètres pour l'optimisation en ligne de la trajectoire des robots

inscrite dans un cercle de rayon $\rho_i = 0.2m$. Les vitesses maximales linéaire et angulaire sont $v_{i,max} = 0.5m/s$ et $w_{i,max} = 5rad/s$, respectivement. L'ensemble des vitesses admissibles du robot est donc :

$$\mathcal{U}_i = \{(v_i(t), w_i(t)) \in \mathbb{R}^2 : |v_i(t)| \leq 0.5m/s, |w_i(t)| \leq 5rad/s\}$$

Les résultats suivants ont été obtenus en implémentant les programmes écrits en C sur un ordinateur Pentium IV 2.4 Ghz (192Mo de RAM).

3.6.1 Scénario 1 : 2 robots qui se croisent

Dans les tests suivants, deux robots de type unicycle, dont les dynamiques sont données par l'équation (2.2.3), se situent, respectivement, à l'instant $t_{initial} = 0s$ aux configurations $q_{1,initial} = [0, 0, 0^\circ]^T$ et $q_{2,initial} = [0, 5.1, 0^\circ]^T$ à vitesse nulle et veulent atteindre la configuration $q_{1,final} = [5, 5, 0^\circ]^T$ et $q_{2,final} = [5, 0, 0^\circ]^T$, respectivement, à vitesses nulles, le plus rapidement possible. Etant donné que l'environnement n'est pas complètement connu avant la mission mais seulement dans une zone autour des robots de rayon 1.5m, la stratégie mise en place est une planification sur un horizon glissant dont les paramètres sont donnés dans la Table 3.1.

Dans ce scénario, on ne considère pas de contrainte (3.2.3) sur la distance maximale inter-robots (c'est-à-dire, $d_{1,com} = d_{2,com} = +\infty$). Ainsi, au niveau des contraintes qui couplent les états des deux robots, seule la contrainte (3.2.4) d'évitement de collisions doit être vérifiée. On prend comme distance de sécurité inter-robots $d_{sécurité} = 0.4m$.

Pour les robots i ($i = 1, 2$), l'objectif est de déterminer sur chaque intervalle de temps $[\tau_{k-1}, \tau_k]$, la trajectoire admissible et sans collision minimisant la distance restante à parcourir. Il est à noter que ce problème est difficile à résoudre du fait de sa forte symétrie et permet de mettre en valeur les propriétés des différents algorithmes⁴.

⁴Afin de pouvoir comparer les résultats, tous les problèmes de commande optimale sont résolus par la méthode donnée Section 2.3. En outre, les paramètres d'optimisation sont les mêmes pour les différents algorithmes utilisés.

Approche centralisée développée dans la Section 3.3 et “faiblement décentralisée” de [Keviczky et al., 2006]

Un superviseur est utilisé pour mettre en place l’approche centralisée de l’algorithme de planification de trajectoire. A l’instant initial, les deux robots lui transmettent leur configuration initiale ainsi que leur objectif. Puis, le superviseur résout pendant chaque intervalle de temps $[\tau_{k-1}, \tau_k)$ un problème d’optimisation de taille $2N_a(d + n_{\text{knot}} - 1)$. Le temps maximum nécessaire au superviseur pour la résolution de ce problème lors du conflit entre les deux robots mobiles (i.e. le croisement) est de $172ms$. Les résultats issus de l’implémentation centralisée sont donnés dans la figure 3.11. Les vitesses linéaires, visualisées sur la figure 3.11(b) restent toujours inférieures à $0.5m/s$. Sur les figures 3.11(a)-(b), il est possible de voir que chaque robot essaie d’éviter l’autre en adaptant sa trajectoire et sa vitesse. Ainsi, lors du croisement, les deux robots ralentissent et modifient leur trajectoire. La figure 3.11(c) montre l’évolution de la distance entre les deux robots. Elle est toujours strictement supérieure à $0.4m$. Par conséquent, les robots ne sont jamais en collision. Le temps mis par les robots pour atteindre l’objectif est de l’ordre de $15.8s$ pour le robot 1 et $16.2s$ pour le robot 2.

L’utilisation d’un superviseur pour mettre en place l’algorithme “faiblement décentralisé” de [Keviczky et al., 2006] est inutile. Tant que les deux robots ne risquent pas d’entrer en collision durant l’horizon de planification, chacun planifie sa trajectoire sans tenir compte de l’autre en résolvant un problème d’optimisation de taille $2(d + n_{\text{knot}} - 1)$. Puis, lorsqu’un risque est encouru, ils échangent leur dernière configuration planifiée et mettent en place une approche centralisée au cours de laquelle ils calculent non seulement leur trajectoire mais aussi celle de l’autre robot (problème d’optimisation de taille $2N_a(d + n_{\text{knot}} - 1)$). Par conséquent, les résultats obtenus en implémentant l’approche “faiblement décentralisée” sont identiques à ceux de l’approche centralisée.

Approche “meneur / suiveur” de [Kuwata et al., 2006]

Un ordre de résolution est établi (robot 1 puis robot 2). Ici, contrairement au cas précédent, le robot meneur 1 ne cherche pas à éviter le robot suiveur 2. Il planifie, en premier lieu, sa trajectoire optimale en résolvant un problème d’optimisation de taille $2(d + n_{\text{knot}} - 1)$ sans tenir compte des contraintes d’évitement de collisions. Lorsque sa trajectoire optimale est calculée, il la communique au robot 2. Démarre alors la planification de trajectoire pour le robot 2 qui adapte sa trajectoire et sa vitesse de manière assez brusque de manière à pouvoir éviter le robot 1 sans perdre de temps. Le temps maximum nécessaire pour la résolution du problème de planification lors du conflit entre les deux robots mobiles est, ici, de $40ms$. Il est très faible étant donné la simplicité du problème. En effet, la génération de la trajectoire optimale du robot 1 est quasiment instantanée. Quant à celle du robot 2, elle est assez facile à déterminer. Les résultats issus de cet algorithme de planification sont donnés dans la figure 3.12. Les vitesses linéaires, visualisées sur la figure 3.12(b) restent toujours inférieures

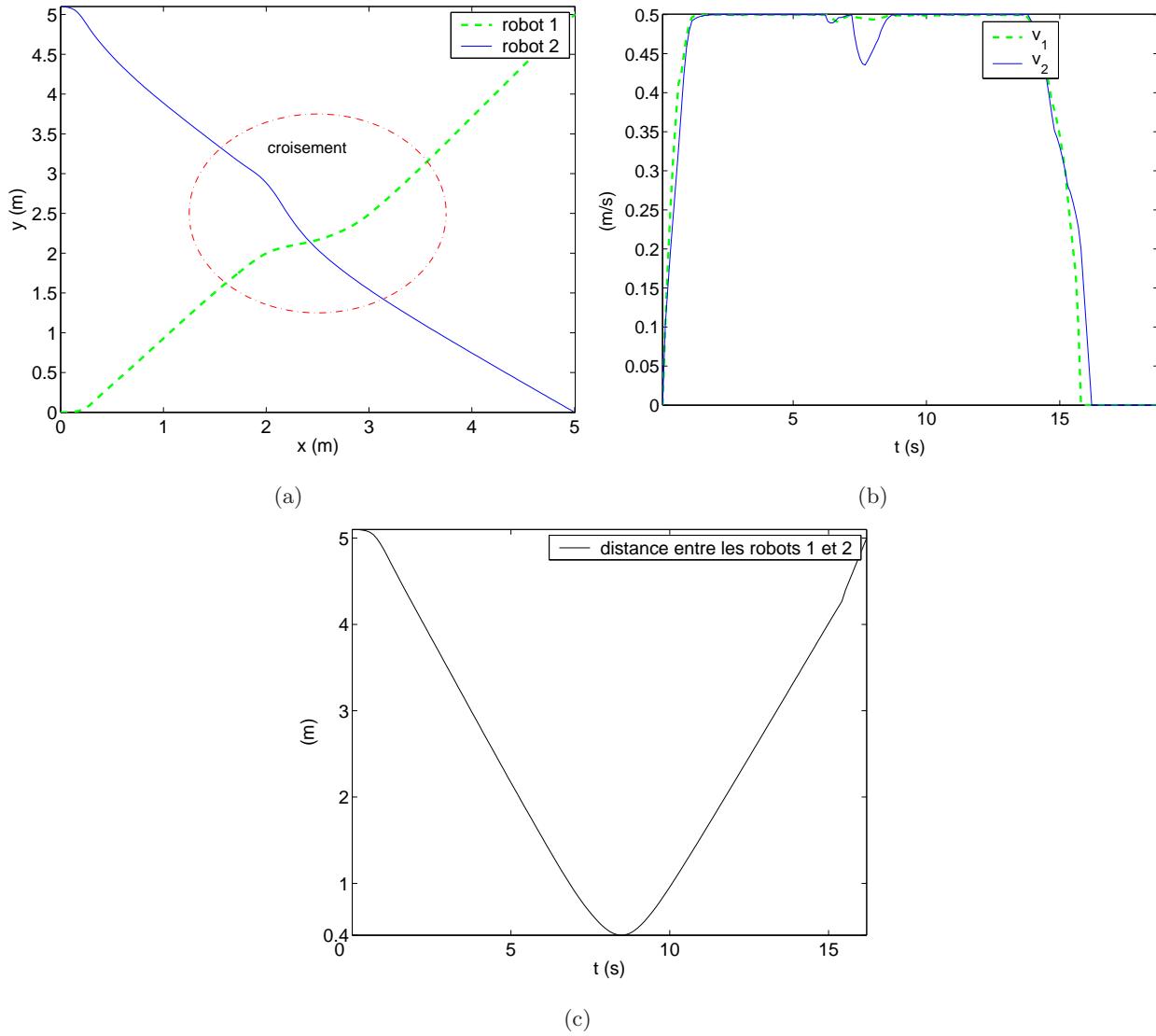


FIG. 3.11 – Approche centralisée, développée dans la Section 3.3, pour deux robots se croisant.

à 0.5m/s . Il est possible de voir que tous les efforts pour l'évitement de collisions sont réalisés par le robot 2 qui doit ralentir assez fortement pour éviter la collision. Ainsi, bien que la durée de parcours du robot soit très faible (15.7s), celle du robot 2 est supérieure (de l'ordre de 16.5s). La figure 3.12(c) montre l'évolution de la distance entre les deux robots qui reste toujours supérieure à 0.4m .

Approche décentralisée développée dans la section 3.4

Contrairement au cas précédent, aucun ordre de résolution n'est établi. Ici, la planification de trajectoire s'effectue en deux étapes. Dans un premier temps, chaque robot calcule sa trajectoire admissible intuitive. Une fois cette étape achevée (pour ce scénario, elle est quasiment instantanée), les deux robots échangent leur trajectoire intuitive si une collision peut se produire sur l'horizon de

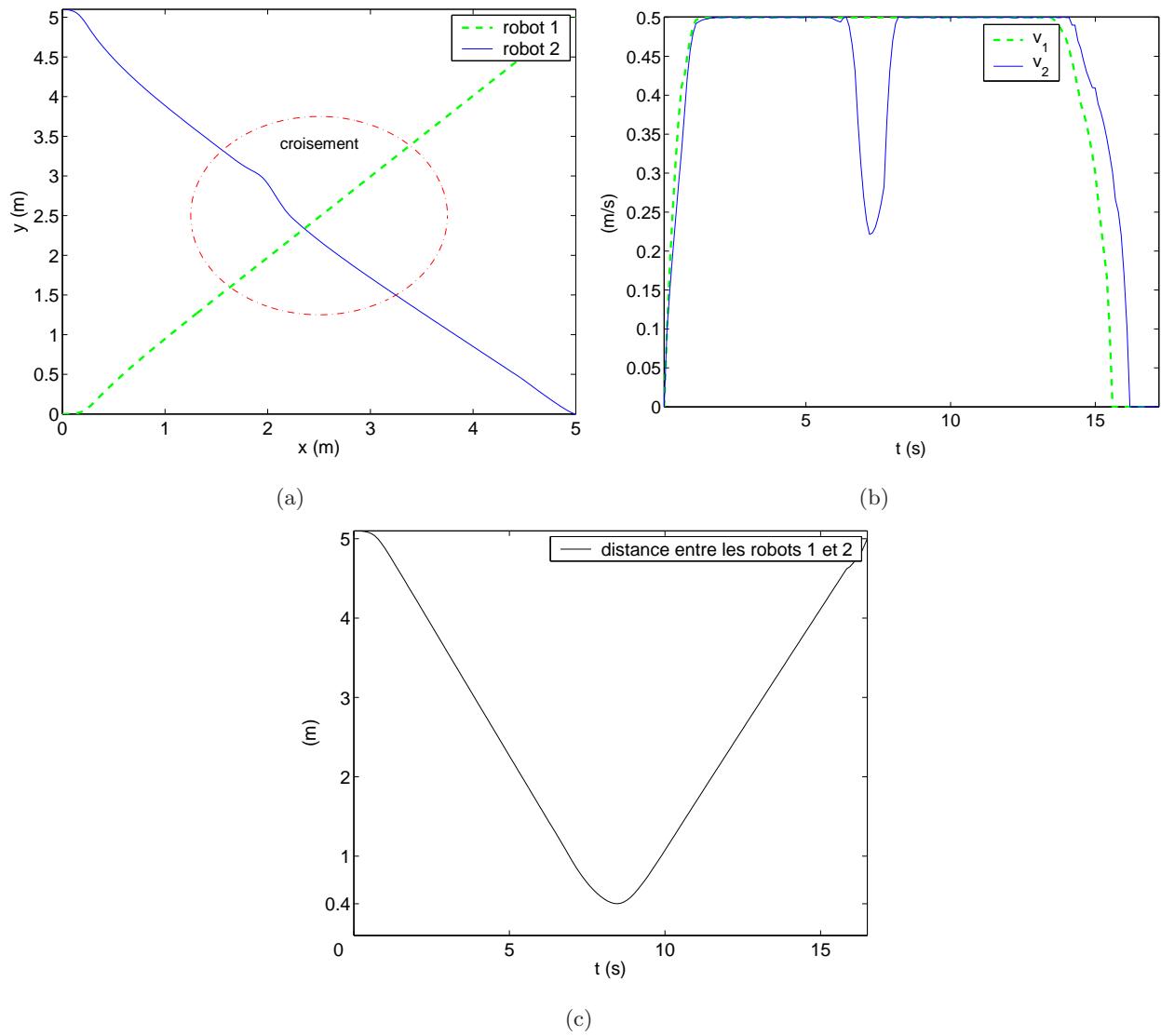


FIG. 3.12 – Approche de type “meneur / suiveur” pour deux robots se croisant.

planification. Puis, en fonction des informations échangées, chaque robot modifie sa trajectoire intuitive de manière à éviter les collisions. Le temps maximum nécessaire pour la résolution du problème de planification lors du conflit entre les deux robots mobiles est, ici, de 94ms. Elle est environ deux fois plus faible que lors de l’implémentation de l’algorithme centralisé ou “faiblement décentralisé”.

Remarque 3.10 Contrairement à l’approche “faiblement décentralisée” de [Keviczky et al., 2006], la contrainte (3.4.14) a été ajoutée de manière à assurer un degré de correspondance suffisant entre la trajectoire intuitive transmise à l’autre robot et la trajectoire optimale planifiée. Ceci permet de garantir l’évitement de collisions entre les deux robots.

Les résultats issus de cet algorithme de planification sont donnés dans la figure 3.13. Les vitesses linéaires, visualisées sur la figure 3.13(b) restent toujours inférieures à $0.5m/s$. Ici, les deux robots adaptent leur trajectoire et leur vitesse de manière à s'éviter. Le temps mis par les robots pour atteindre l'objectif est de l'ordre de $16.0s$ pour le robot 1 et $16.3s$ pour le robot 2. La figure 3.13(c) montre l'évolution de la distance entre les deux robots qui reste toujours supérieure à $0.4m$.

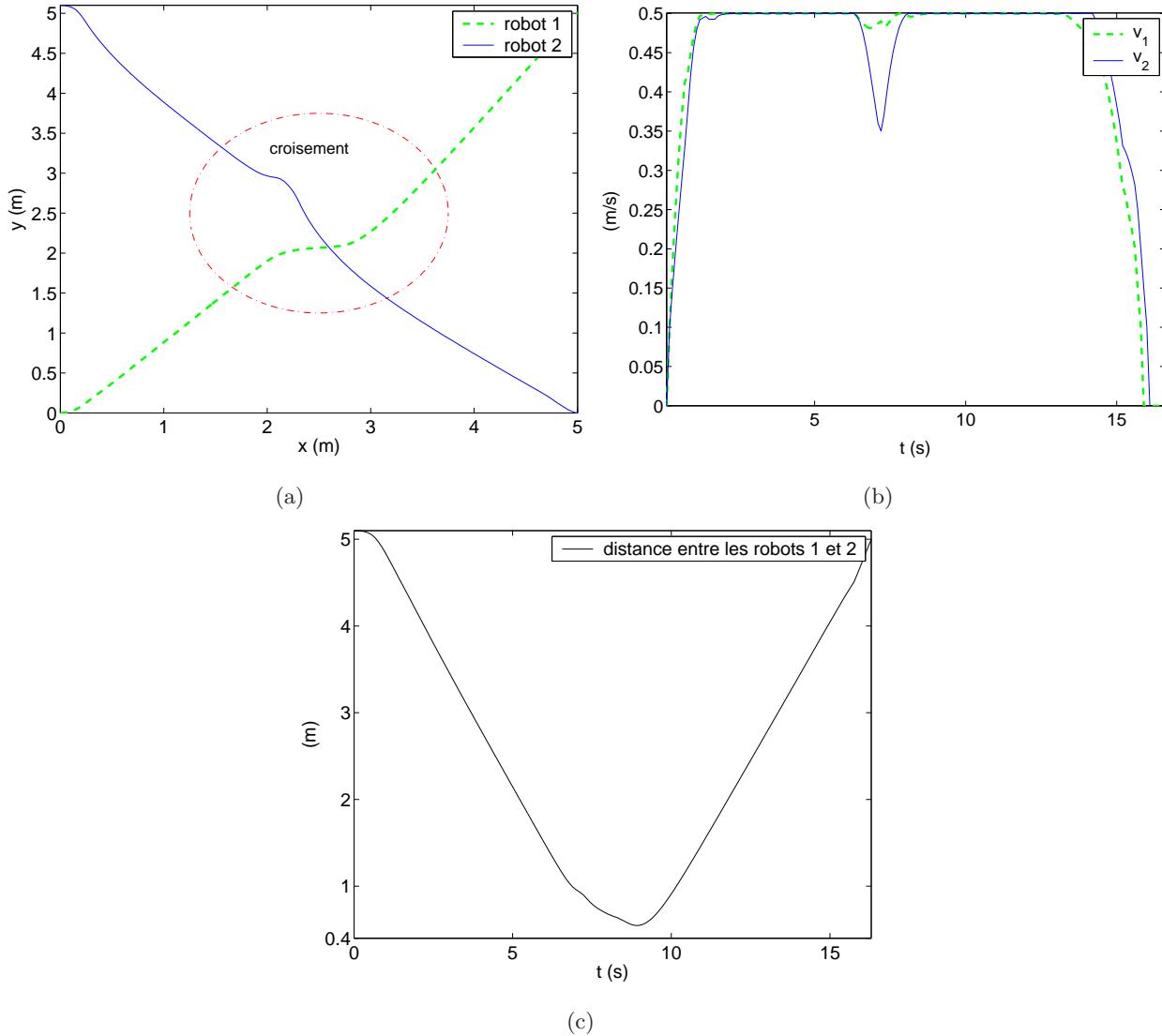


FIG. 3.13 – Approche (“fortement”) décentralisée, développée dans la section 3.4, pour deux robots se croisant.

Un récapitulatif des résultats obtenus en termes de temps de calcul pour construire la trajectoire optimale sur un horizon de planification et d’optimalité est présenté dans la Table 3.2.

Approche	Centralisée Section 3.3	Meneur / Suiveur décentralisée [Kuwata et al., 2006]	Faiblement décentralisée [Keviczky et al., 2006]	Fortement décentralisée [Defoort et al., 2007a]
Temps maxi résolution conflit	172ms	40ms	172ms	94ms
Durée atteinte objectif	16.2s	16.5s	16.2s	16.3s

TAB. 3.2 – Comparaisons des performances associées aux différents algorithmes de planification de la flottille de 2 robots

3.6.2 Scénario 2 : Flottille de 5 robots dans un environnement inconnu

On considère une formation de cinq robots mobiles de type unicycle dont les dynamiques sont données par l'équation (2.2.3). A l'instant $t_{initial} = 0s$, les robots ont une vitesse et une orientation nulles. Leur position initiale est donnée dans la table 3.3. La flottille de robots est ainsi en formation “ligne”. L'objectif est pour chaque robot d'atteindre les positions finales données dans la table 3.4 le plus rapidement possible. On peut remarquer qu'une reconfiguration de la forme de la flottille aura lieu (i.e. passage d'une forme “ligne” à une forme “triangulaire”). Au cours du déplacement, les robots doivent maintenir les liaisons de communication entre les paires (1,2), (1,3), (2,4) et (3,5) (voir le graphe de communication donné figure 3.14). Les cinq robots prennent également des décisions afin d'éviter les collisions entre-eux et avec les obstacles initialement inconnus. En effet, l'environnement n'est parfaitement connu que dans une zone de rayon 1.5m autour de chaque robot. La portée de diffusion des informations du robot i ($i = 1, \dots, 5$) est égale à $d_{i,com} = 2.5m$. La distance de sécurité inter-robots est $d_{sécurité} = 0.4m$.

Numéro du robot	Position en x	Position en y
1	0	0
2	0	2
3	0	-2
4	0	4
5	0	-4

TAB. 3.3 – Position initiale des robots “en ligne”

La stratégie mise en place pour les différents algorithmes est une planification sur un horizon glissant dont les paramètres sont donnés dans la Table 3.5. Etant donné que les robots doivent se croiser,

Numéro du robot	Position en x	Position en y
1	15	0
2	13.5	-1.5
3	13.5	1.5
4	12	-3
5	12	3

TAB. 3.4 – Position finale des robots “en triangle”

Nombre de robots N_a	5
horizon de planification T_p	2s
horizon de calcul T_c	0.5s
horizon d’intuition T_d	2.5s
déformation admissible ξ	0.25
ordre des B-splines d	4
nombre d’intervalles de longueur non nulle du vecteur noeud n_{knot}	3
nombre d’échantillons N_{ech} pour la discréttisation	25

TAB. 3.5 – Paramètres pour l’optimisation en ligne de la trajectoire des robots

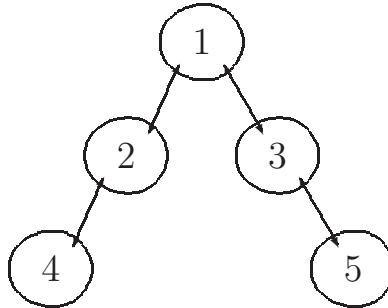


FIG. 3.14 – Graphe de communication de la flottille.

le nombre de conflits est assez important et les problèmes d’optimisation sont difficiles à résoudre. Ainsi, les avantages et surtout les limites des différents algorithmes pourront être mis en évidence.

Les résultats issus de l’algorithme de planification (“fortement”) décentralisé, proposé dans la Section 3.4, sont donnés dans la figure 3.15. Les cinq robots adaptent leur trajectoire et leur vitesse de manière à satisfaire l’ensemble des contraintes (i.e. contraintes de non holonomie, contraintes sur les vitesses admissibles, contraintes d’évitement d’obstacles, contraintes de non collisions inter-robots et contraintes de maintien des liaisons de communication). La coordination se fait par ajustements mutuels (aucun véhicule n’a de contrôle sur les autres). Ceci permet d’éviter une trop forte dépendance

vis-à-vis d'un seul robot ou d'un superviseur. Le temps mis par la flottille de robots pour atteindre l'objectif et se reconfigurer sous une forme triangulaire est de 36.5s. La figure 3.15(b) montre l'évolution des distances entre les robots qui maintiennent une liaison de communication. Ces distances restent toujours comprise entre 0.4m et 2.5m. Les autres distances inter-robots peuvent être visualisées sur la figure 3.15(c) et mettent en évidence l'absence de collisions entre les robots.

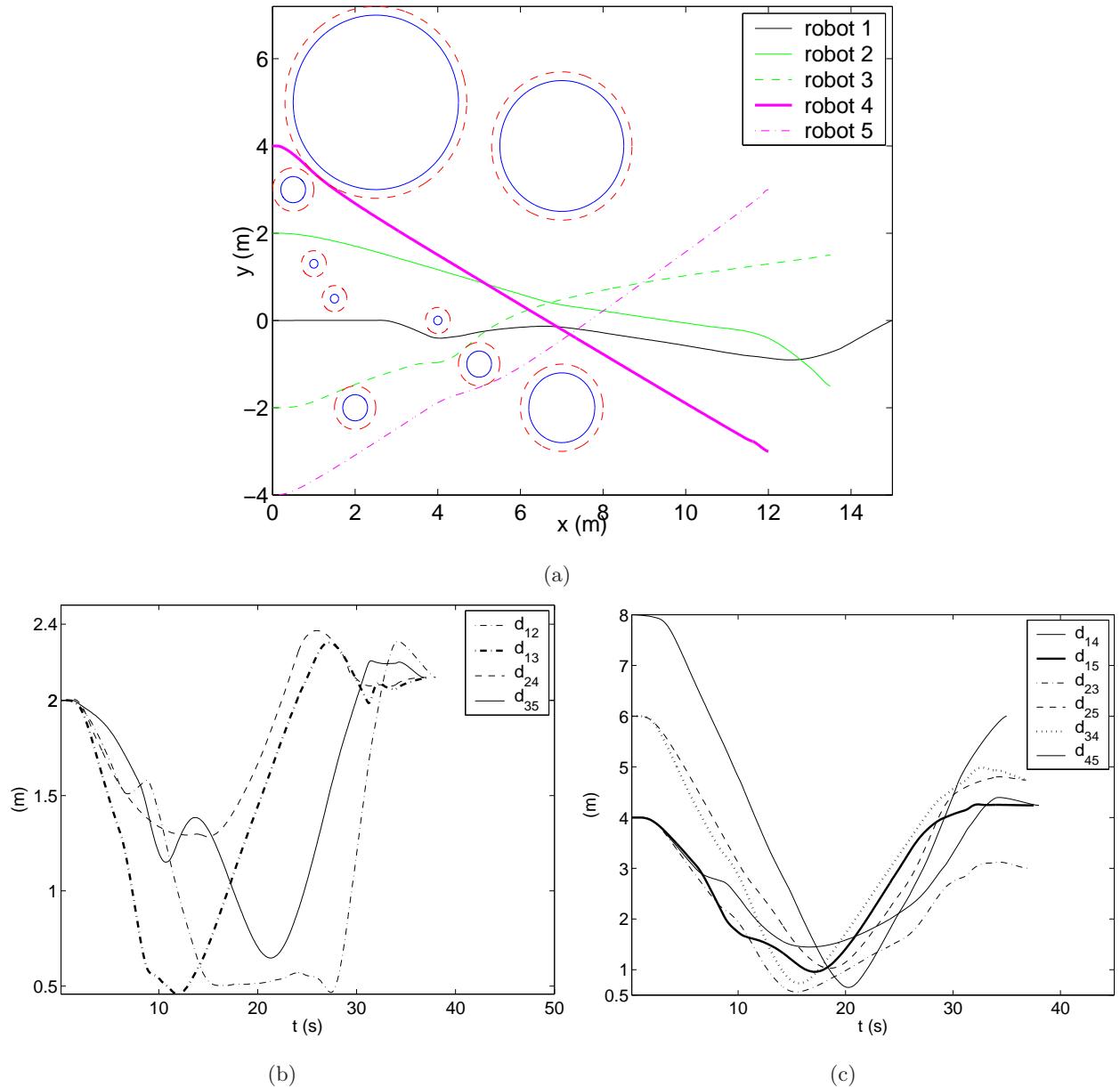


FIG. 3.15 – Approche (“fortement”) décentralisée, développée dans la Section 3.4, pour une flottille de cinq robots passant d'une forme “ligne” à une forme “triangulaire”.

Afin de mettre en lumière les avantages de l'architecture proposée, le même scénario est appliqué

aux autres approches. Les résultats associés à l'approche centralisée sont représentés sur la figure 3.16, ceux associés à l'approche “faiblement” décentralisée sur la figure 3.17 et enfin ceux associés à l'approche “meneur / suiveur” sur la figure 3.18. Sur ces figures, il est possible de voir que les contraintes de non collision inter-robots et de maintien des liaisons de communication sont satisfaites. Des comparaisons en termes de temps de calcul pour construire la trajectoire optimale sur un horizon de planification, de ressources en communication, de facilité de mise en œuvre et d’optimalité sont présentées dans la Table 3.6. L'algorithme centralisé offre les meilleures performances en termes d'optimalité au niveau de la durée du parcours. Cependant, puisque le temps maximum alloué à la résolution des conflits est supérieur à l'horizon de calculs T_c , cet algorithme n'est pas applicable en temps réel. Il en est de même pour l'algorithme “faiblement décentralisé”. Ceci s'explique par la taille élevée du problème d'optimisation à résoudre. Notons que pour l'algorithme centralisé, le temps de calcul des trajectoires optimales ne varie pas linéairement en fonction du nombre de véhicules mais plutôt de manière exponentielle. Quant à l'algorithme “faiblement décentralisé”, ce temps dépend du nombre maximum de robots en conflit durant un horizon de planification. Seule l'approche “fortement décentralisée” et l'approche “meneur / suiveur” sont applicables en temps réel. Cependant, en utilisant une stratégie “meneur / suiveur”, les performances sont dégradées du fait de l'absence de retour d'informations du suiveur vers le meneur. Ce manque de coopération implique une planification plus difficile pour les robots faiblement prioritaires. A la vue de ces résultats, on peut conclure que l'algorithme “fortement décentralisé”, développé dans la Section 3.4, offre le meilleur compromis entre optimalité et facilité de mise en œuvre pour ce scénario.

3.7 Conclusion

Dans ce chapitre, nous avons abordé le problème de planification de trajectoire admissible et sans collision pour une flottille de robots mobiles évoluant dans un environnement inconnu. Par rapport au cadre mono-robot, deux types de contraintes ont été ajoutés : les contraintes d'évitement de collisions inter-robots et les contraintes de maintien des liaisons de communication au cours du déplacement.

Afin de résoudre ce problème, deux algorithmes de coordination ont été proposés et validés en simulation. Le premier est basé sur une architecture centralisée et peut être vu comme une extension naturelle de l'algorithme de planification en ligne défini dans le cadre mono-robot. Dans ce cas, un superviseur génère les trajectoires pour tous les véhicules en résolvant un problème d'optimisation de grande dimension. Cependant, le manque d'autonomie des véhicules par rapport au superviseur, la centralisation des informations et les coûts en termes de temps de calcul rendent l'implémentation en ligne de cette stratégie délicate.

C'est pourquoi nous nous sommes intéressés, par la suite, à un algorithme décentralisé de planification en ligne sur un horizon glissant, basé uniquement sur les informations locales disponibles. L'idée était de décomposer en deux étapes la génération d'une trajectoire optimale satisfaisant l'ensemble

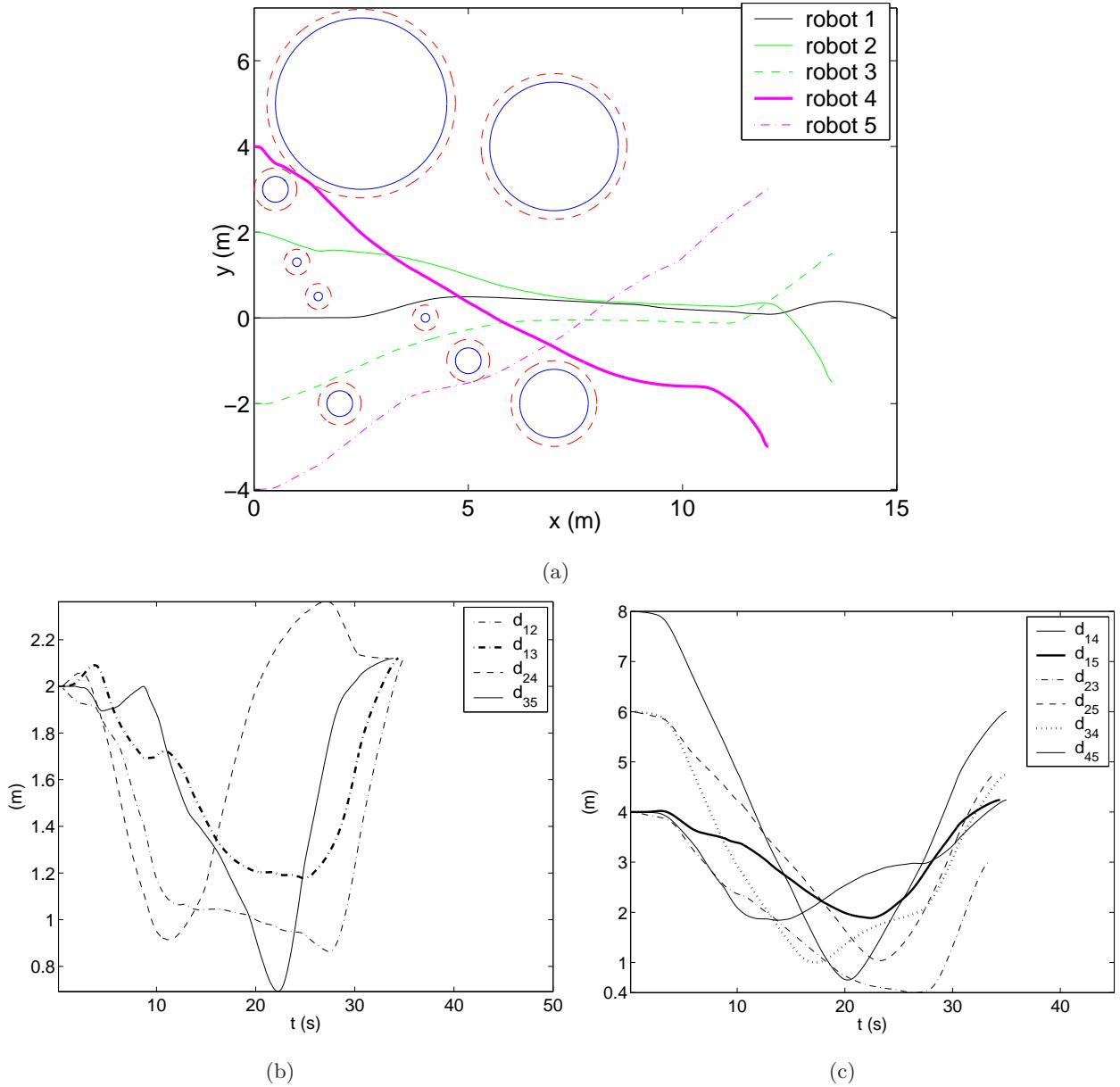


FIG. 3.16 – Approche centralisée, développée dans la Section 3.3, pour une flottille de cinq robots passant d’une forme “ligne” à une forme “triangulaire”.

des contraintes pour chaque robot. La première étape consistait à déterminer une trajectoire intuitive. Puis, en incluant les notions d’intention et d’engagement des robots qui peuvent provoquer un conflit, chaque véhicule élaborait sa trajectoire optimale de référence. Cette implémentation distribuée augmente non seulement l’autonomie des robots, mais elle réduit également la complexité calculatoire par rapport à une implémentation centralisée.

Des simulations pour différents scénarios ont montré l’efficacité de nos algorithmes et ont permis une comparaison avec d’autres méthodes existantes (approche “faiblement décentralisée” de

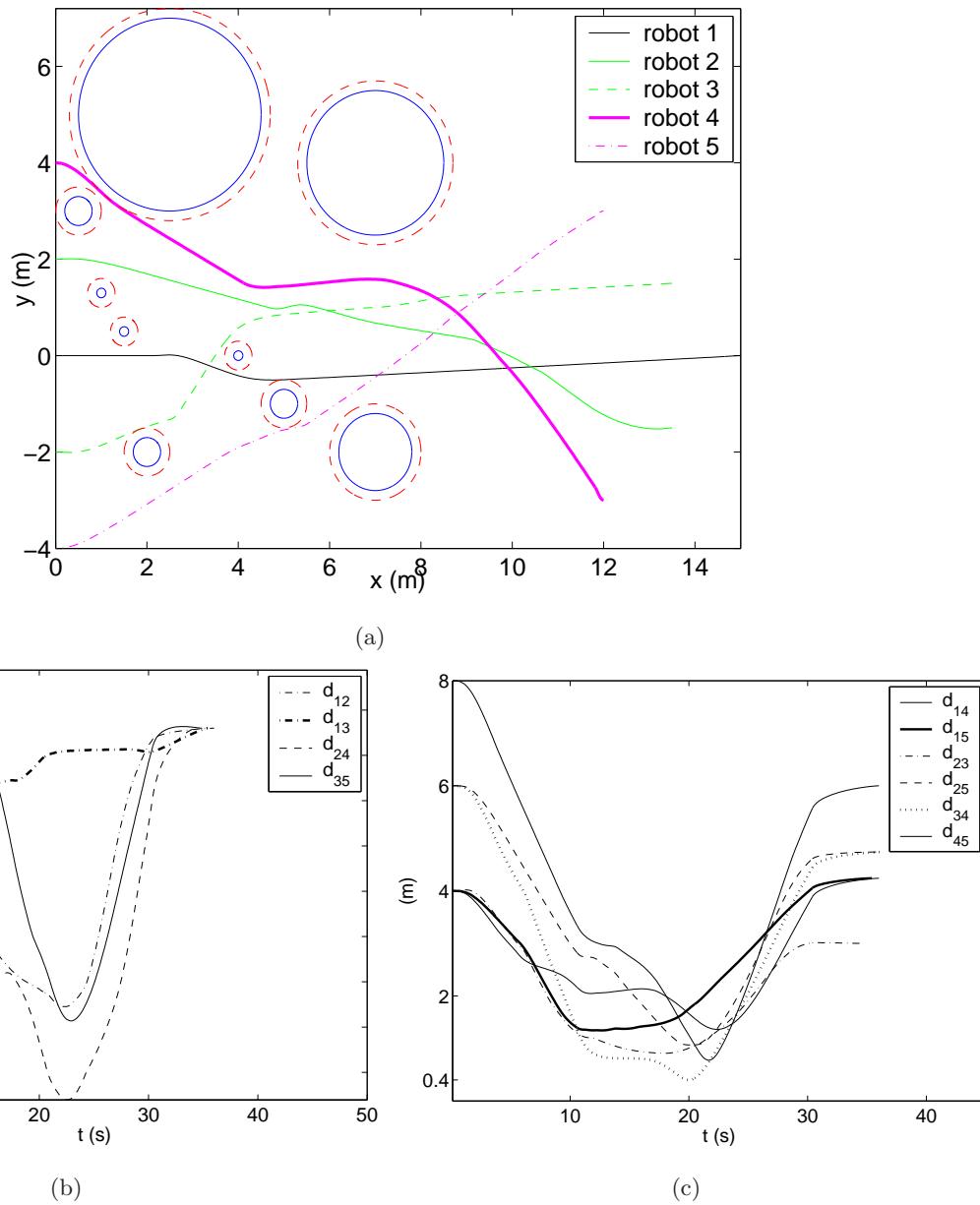


FIG. 3.17 – Approche “faiblement décentralisée” pour une flottille de cinq robots passant d’une forme “ligne” à une forme “triangulaire”.

[Keviczky et al., 2006], approche “meneur / suiveur” de [Kuwata et al., 2006]).

La suite de ce manuscrit est consacré à la poursuite des trajectoires planifiées.

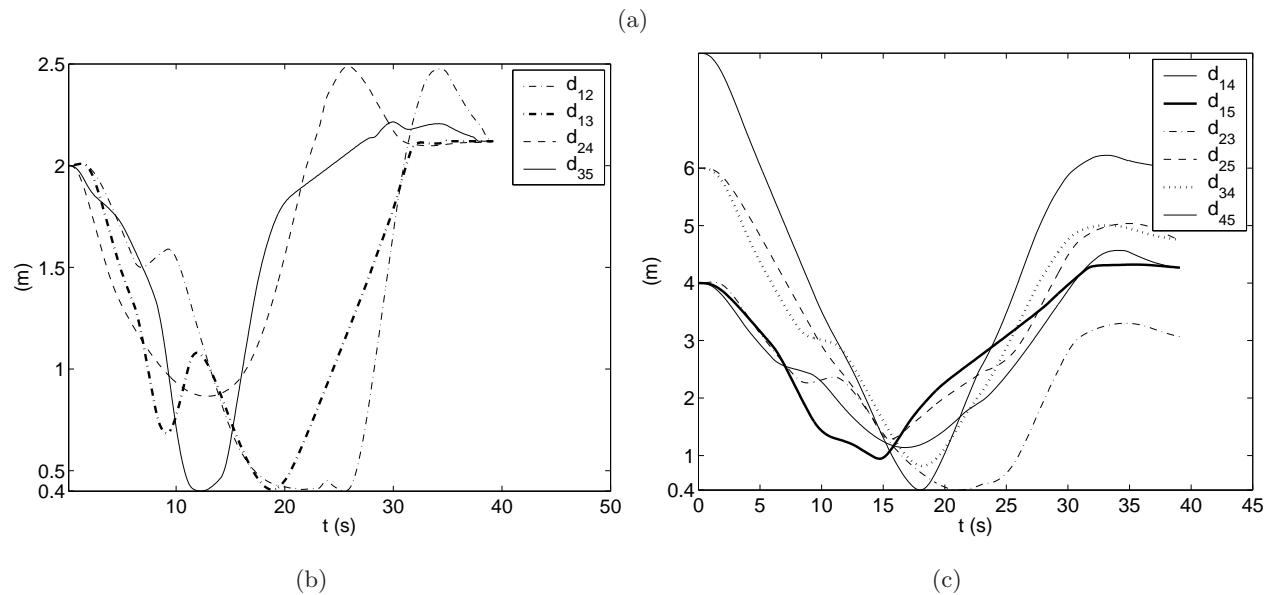
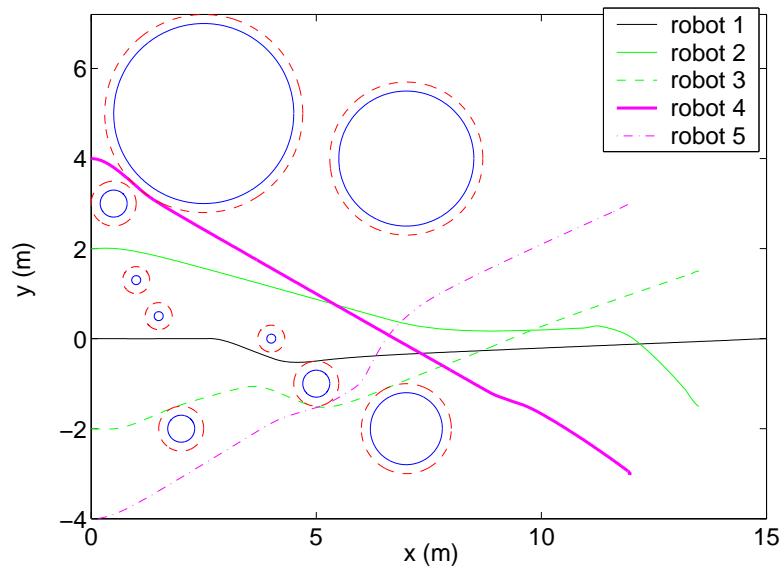


FIG. 3.18 – Approche “meneur / suiveur” pour une flottille de cinq robots passant d’une forme “ligne” à une forme “triangulaire”.

Approche	Centralisée Section 3.3	Meneur / Suiveur décentralisée [Kuwata et al., 2006]	Faiblement décentralisée [Keviczky et al., 2006]	Fortement décentralisée [Defoort et al., 2007a]
Temps maxi résolution conflit	2050ms	313ms	703ms	121ms
Echange Information	global	local	local	local
Mise en œuvre	-- (impossible quand $N_a \gg 1$)	++ (résolution séquentielle)	- (difficile si conflit entre de nombreux robots)	+
Durée atteinte objectif	35s	39s	36s	36.5s

TAB. 3.6 – Comparaisons des performances associées aux différents algorithmes de planification de la flottille de 5 robots

Deuxième partie

Poursuite de trajectoire

Chapitre 4

Commande par modes glissants

4.1 Introduction

Dans la première partie de ce mémoire, nous avons élaboré différents algorithmes de planification pour une flottille de véhicules, permettant de générer les trajectoires et les commandes de référence. Cependant, pour les robots réels, l'application directe de cette commande planifiée produit un écart systématique entre le résultat désiré et la réponse obtenue. Cette erreur peut entraîner des ruptures de communication voire des collisions ou tout simplement ne permet pas d'atteindre la configuration finale. Ainsi, il est nécessaire, à partir des trajectoires et des commandes planifiées, d'élaborer des commandes dites robustes capables de rendre le système moins sensible aux perturbations, aux simplifications et approximations dans la modélisation, aux variations des paramètres du modèle ainsi qu'aux incertitudes sur les variables physiques mesurées.

L'approche par correcteurs linéaires, du type PID, a vite montré ses limites. En effet, ceux-ci sont soumis à la loi de Bode qui veut que les effets d'amplitude et les effets de phase soient couplés et antagonistes. Les recherches se sont alors orientées vers des techniques non linéaires, telles que les méthodes adaptatives ou de stabilité absolue, mais également la technique des modes glissants. Les commandes par modes glissants (CMG) sont réalisées de manière à conduire et contraindre le système à rester dans le voisinage d'une surface de commutation. Il y a deux principaux avantages à une telle approche. Tout d'abord, le comportement dynamique résultant peut être déterminé par le choix d'une surface adéquate. Ensuite, la réponse du système en boucle fermée présente de bonnes propriétés de robustesse vis-à-vis des perturbations, ce qui fait de cette méthode une candidate sérieuse dans la perspective de l'élaboration de commandes robustes.

Cependant, la plupart des algorithmes de CMG d'ordre arbitraire souffrent de l'absence de conditions suffisantes de convergence constructives. La tâche de réglage des paramètres de la loi de commande s'avère alors difficile lorsqu'il s'agit d'atteindre des performances bien spécifiques pour le système bouclé. Ce handicap fait que ces techniques perdent de leur pertinence. Ainsi, la recherche d'algorithmes garantissant un régime glissant d'ordre quelconque et proposant un réglage simple des

paramètres de la loi de commande en vue d'atteindre les performances désirées, présente un réel intérêt.

Ce chapitre est consacré dans un premier temps à une présentation générale des concepts de base de la CMG d'ordre supérieur. Son principe, ses propriétés de robustesse ainsi que le phénomène de chattering et les solutions pour l'éliminer, y sont présentés.

Dans un second temps, un algorithme *original* de CMG d'ordre supérieur basé sur une extension dynamique du système ainsi que sur des propriétés d'homogénéité est proposé pour des systèmes non linéaires incertains. Cet algorithme permet de garantir l'établissement d'un régime glissant d'ordre quelconque avec un réglage simple des paramètres de la loi de commande en vue d'atteindre les performances désirées. Le lien entre l'algorithme proposé et le principe de commande par modes glissants avec action intégrale (CMGI), encore appelé *integral sliding mode*, introduit dans [Utkin et Shi, 1996], est mis en évidence lorsque l'état initial du système ainsi qu'un certain nombre de ses dérivées sont connus.

Un exemple académique, à savoir la commande d'un aéroglyisseur, ainsi que des tests expérimentaux pour la commande d'un moteur pas–à–pas illustreront l'algorithme de commande et ses performances.

4.2 Etat de l'art

Remarque 4.1 *Dans un souci de clarté et pour éviter d'alourdir les notations, les développements théoriques se feront sur la classe des systèmes non linéaires, continus, mono-entrée mono-sortie et affines en la commande. Tous les résultats exposés ci-après peuvent néanmoins être généralisés à des systèmes de la forme $\dot{x} = f(x, u)$ et au cas des systèmes multivariables.*

Les modes glissants pour les systèmes non linéaires ont été largement étudiés depuis leur introduction (voir les ouvrages [Utkin, 1992, Edwards et Spurgeon, 1998, Perruquetti et Barbot, 2002]). Dans cette section, une présentation générale et succincte de la CMG d'ordre un et d'ordre supérieur est donnée.

4.2.1 Commande par modes glissants d'ordre un

Généralités

Le principe de la CMG est, à l'aide d'une commande discontinue, de contraindre le système à atteindre une surface donnée pour ensuite y rester. La synthèse d'une CMG se déroule en deux temps :

- une surface est déterminée en fonction des objectifs de commande et des propriétés statiques et dynamiques désirées pour le système bouclé,
- une commande discontinue est synthétisée de manière à contraindre les trajectoires d'état du système à atteindre et, ensuite, à rester sur cette surface en dépit d'incertitudes, de variations de paramètres ...

Soit le système non linéaire, affine en la commande, défini par :

$$\dot{x} = f(x, t) + g(x, t)u + p(x, t) \quad (4.2.1)$$

où $x = [x_1, \dots, x_n]^T \in \mathcal{X}$ représente l'état du système avec \mathcal{X} un ensemble ouvert de \mathbb{R}^n et $u \in \mathcal{U}$ est l'entrée de commande qui est une fonction éventuellement discontinue, bornée, dépendante de l'état et du temps, avec \mathcal{U} un ouvert de \mathbb{R} . $f(x, t)$ et $g(x, t)$ sont des champs de vecteurs suffisamment différentiables. Le vecteur $p(x, t) \in \mathbb{R}^n$ représente les incertitudes et perturbations du système.

Soit $s(x, t) : \mathcal{X} \times \mathbb{R}^+ \rightarrow \mathbb{R}$ une fonction suffisamment différentiable et considérée comme une sortie fictive du système (4.2.1) telle que son annulation permette de satisfaire l'objectif de commande. La fonction $s(x, t)$ est appelée **variable de glissement**. L'ensemble

$$\mathcal{S} = \{x \in \mathcal{X} : s(x, t) = 0\} \quad (4.2.2)$$

représente alors une sous-variété de \mathcal{X} de dimension $(n - 1)$ appelée **surface de glissement**.

Définition 4.1 [Utkin, 1992] On dit qu'il existe un **régime glissant idéal** sur \mathcal{S} s'il existe un temps fini T_{eta} tel que la solution de (4.2.1) satisfasse $s(x, t) = 0$ pour tout $t \geq T_{eta}$.

Quand les trajectoires du système (4.2.1) évoluent sur la surface de glissement \mathcal{S} , sa dynamique est dite immergée dans l'état d'un système autonome de dimension $n - 1$. Ce système, appelé **système réduit**, a une dynamique déterminée par la surface de glissement. Une condition nécessaire pour l'établissement d'un régime glissant d'ordre un est que le système (4.2.1) soit de degré relatif égal à un par rapport à la variable de glissement [Utkin, 1992] (le **degré relatif** d'un système est le nombre minimum de fois qu'il faut dériver la sortie, par rapport au temps, pour faire apparaître l'entrée de manière explicite [Isidori, 1995]).

Une fois la surface de glissement choisie, la seconde étape consiste à construire une commande u de façon à ce que les trajectoires d'état du système soient amenées vers cette surface et soient ensuite maintenues dans un voisinage de celle-ci malgré la présence d'incertitudes et de perturbations sur le système. En d'autres termes, la commande doit rendre la surface de glissement localement attractive (i.e. au voisinage de la surface de glissement, les trajectoires du système de part et d'autre de la surface doivent tendre vers cette dernière). Une condition nécessaire et suffisante, appelée **condition d'attractivité**, pour qu'une variable de glissement $s(x, t)$ tends vers 0 est [Itksis, 1976] :

$$s\dot{s} < 0 \quad (4.2.3)$$

Cependant, l'inégalité (4.2.3) n'est pas suffisante pour assurer une convergence en temps fini vers la surface et donc, un fonctionnement en régime glissant. Elle est généralement remplacée par la condition suivante, appelée **condition de η -attractivité** :

$$s\dot{s} \leq -\eta|s|, \quad \eta > 0 \quad (4.2.4)$$

La méthode dite de la commande équivalente [Utkin, 1992] est un moyen de décrire le comportement du système lorsqu'il est restreint à la surface $\{s = 0\}$. Elle est obtenue grâce aux **conditions d'invariance** de la surface :

$$\begin{aligned} s &= 0 \\ \dot{s} &= \frac{\partial s}{\partial x} (f(x) + g(x)u^{eq}) = 0 \end{aligned} \quad (4.2.5)$$

u^{eq} , appelé **commande équivalente**, est associée au système nominal, i.e. sans incertitude, et est déterminée de façon unique par les conditions d'invariance (4.2.5), c'est-à-dire :

$$u^{eq} = - \left(\frac{\partial s}{\partial x} g(x) \right)^{-1} \frac{\partial s}{\partial x} f(x) \quad (4.2.6)$$

Cependant, cette commande ne force pas les trajectoires du système à atteindre la surface de glissement. Ainsi, la commande u est la somme de la commande équivalente et d'une composante discontinue assurant un régime glissant et l'insensibilité du système vis-à-vis des incertitudes et des perturbations, i.e.

$$u = u^{eq} - G \left(\frac{\partial s}{\partial x} g(x) \right)^{-1} \text{sign}(s) \quad (4.2.7)$$

où $G > 0$ est une constante positive et sign est la fonction signe classique.

Nous ne pouvons finir cette partie sans ajouter que les modes glissants présentent des propriétés de robustesse intéressantes vis-à-vis de certaines perturbations. Si celles-ci vérifient une condition, elles n'affectent pas le système quand il atteint le régime de glissement, comme l'indique le théorème suivant :

Théorème 4.1 [Utkin, 1992] *Un régime glissant sur \mathcal{S} , du système perturbé (4.2.1), est indépendant du signal de perturbation $p(x, t)$, si et seulement si, celui-ci est borné et vérifie*

$$p(x, t) \in \text{Vect}\{g(x)\} \quad (4.2.8)$$

*La condition (4.2.8) est appelée **condition de recouvrement** ou dans la dénomination internationale “matching condition” [Drazenovic, 1969].*

Remarque 4.2 *La commande u du système (4.2.1) étant une fonction discontinue en x , l'équation différentielle engendrée n'a de sens que dans la théorie des inclusions différentielles [Filippov, 1983], [Aubin et Cellina, 1984].*

Exemple illustratif

Afin d'illustrer les différentes étapes de synthèse d'une CMG d'ordre un et ses propriétés, considérons l'exemple d'un double intégrateur perturbé :

$$\begin{cases} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= u + p(x_1) \end{cases} \quad (4.2.9)$$

où $p(x_1) = a \sin(10x_1)$ est une perturbation bornée. L'objectif de commande est de contraindre les variables d'état (x_1, x_2) à atteindre l'origine. On définit la variable de glissement suivante (si l'on veut par exemple que le comportement du double intégrateur soit analogue à un premier ordre de constante de temps $\frac{1}{\lambda}$) :

$$s = \lambda x_1 + x_2, \quad \lambda > 0 \quad (4.2.10)$$

Après le choix de la variable de glissement, une commande discontinue est synthétisée de façon à rendre la surface $\mathcal{S} = \{x \in \mathcal{X} : s = 0\}$ invariante et attractive. En définissant u par :

$$u = -\lambda x_2 - G \operatorname{sign}(s), \quad G \geq a + \eta, \quad \eta > 0$$

un régime glissant prend place sur \mathcal{S} en un temps fini étant donné que

$$\dot{s} = s(\lambda x_2 + u + p) \leq -\eta |s|$$

La figure 4.1(a) montre les deux comportements successifs du système : tout d'abord, le système décrit dans le plan de phase, une trajectoire parabolique tant que la surface de glissement n'est pas atteinte (phase de convergence). Ensuite, il décrit un régime glissant en évoluant le long de \mathcal{S} (phase de glissement) jusqu'à l'origine du plan de phase. La figure 4.1(b) montre que le régime glissant prend place à partir de $T_{eta} = 2.2s$. A partir de cet instant, la commande commute à très haute fréquence et la dynamique du système bouclé se réduit à :

$$\dot{x}_1 = x_2 = -\lambda x_1$$

On peut remarquer qu'une fois le régime glissant établi sur \mathcal{S} , le système perturbé a un comportement dynamique identique à celui du double intégrateur sans perturbation en régime glissant. Le régime glissant jouit, ainsi, de la propriété d'insensibilité vis-à-vis des perturbations intervenant dans la même direction que u . Le système reste cependant sensible à de telles perturbations pendant le régime transitoire, i.e. avant que \mathcal{S} ne soit atteinte. Afin d'assurer l'insensibilité aux perturbations pendant toute la réponse du système, l'élimination de la phase de convergence vers \mathcal{S} est nécessaire [Utkin et Shi, 1996].

Le phénomène de réticence

Un régime glissant idéal requiert une commande pouvant commuter à une fréquence infinie. Évidemment, pour une utilisation pratique, seule une commutation à une fréquence finie est possible. Ainsi, durant le régime glissant, les discontinuités appliquées à la commande peuvent entraîner un phénomène de broutement, appelé réticence ou “chattering” en anglais. Celui-ci se caractérise par de fortes oscillations des trajectoires du système autour de la surface de glissement (figure 4.2). Les principales raisons à l'origine de ce phénomène sont les retards de commutation au niveau de la commande et la présence de dynamiques parasites en série avec les systèmes commandés [Young et al., 1999].

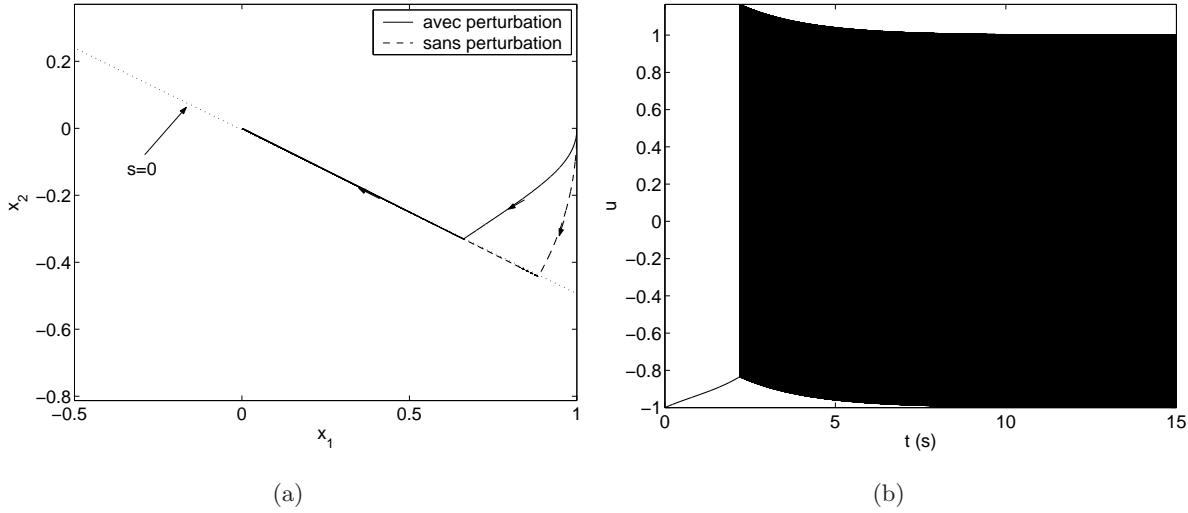


FIG. 4.1 – Exemple de CMG d’ordre un. (a) Trajectoire dans le plan de phase. (b) Commande en fonction du temps.

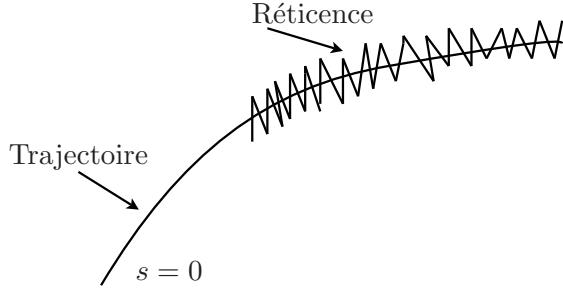


FIG. 4.2 – Le phénomène de réticence.

Ce phénomène constitue un désavantage non négligeable car, même s'il est possible de le filtrer en sortie du processus, il est susceptible d'exciter les modes à haute fréquence qui n'ont pas été pris en compte dans le modèle du système. Il peut être si pénalisant que l'utilisation d'une CMG d'ordre un peut, dans certaines applications, être à proscrire, vu que son utilisation peut dégrader les performances et même conduire à l'instabilité [Heck, 1991]. La réticence implique également d'importantes sollicitations mécaniques au niveau des actionneurs, pouvant provoquer leur usure rapide, ainsi que des pertes énergétiques non négligeables au niveau des circuits de puissance électrique.

De nombreuses études ont été effectuées dans le but de réduire ou d'éliminer ce phénomène. L'une d'entre elles consiste à remplacer la fonction signe par une approximation continue, de type grand gain, dans un voisinage de la surface [Slotine et Sastry, 1983], telle que la fonction saturation ou une fonction sigmoïde (par exemple $\tanh(\frac{x}{\epsilon})$, $\frac{2}{\pi} \arctan(\frac{x}{\epsilon})$, ...). Le régime glissant qui en résulte n'est plus confiné dans \mathcal{S} , mais dans un proche voisinage de celui-ci [Canudas-De-Wit et Perruquetti, 2002]. Bien

que cela permette d'atténuer le phénomène de réticence, la précision par rapport à l'objectif fixé, la robustesse de la commande et le temps de réponse s'en trouvent dépréciés. La technique des modes glissants d'ordre supérieur permet de passer outre ce phénomène indésirable et aussi de palier à la condition sur le degré relatif, rencontrée par la CMG d'ordre un.

4.2.2 Commande par modes glissants avec action intégrale (CMGI)

La CMGI, ou *integral sliding mode control* en anglais, introduite dans [Utkin et Shi, 1996], est un concept de loi de commande par modes glissants, qui possède, en plus des propriétés classiques, un nouvel avantage. Elle permet d'éliminer la phase de convergence vers la surface de glissement, phase durant laquelle le système est sensible aux perturbations et ainsi de garantir les propriétés de robustesse vis-à-vis des perturbations sur le modèle dès l'instant initial (voir [Cao et Xu, 2004, Fridman et al., 2005, Castanos et Fridman, 2006]).

Principe de base [Utkin et Shi, 1996]

L'idée de base de la CMGI consiste à déterminer la variable de glissement de telle sorte que les trajectoires d'état du système à l'instant initial $t = 0s$ soient déjà sur la surface de glissement. Ceci permet d'assurer un régime glissant sans phase de convergence et de garantir la robustesse de la commande pendant toute la réponse du système bouclé.

Reprendons le système non linéaire (4.2.1) et posons les hypothèses suivantes :

Hypothèse 4.1 *Le vecteur des perturbations $p(x, t) \in \mathbb{R}^n$ est borné par une fonction connue $a(x)$,*

$$\|p(x, t)\| \leq a(x)$$

et vérifie la condition de recouvrement (4.2.8).

Hypothèse 4.2 *Il existe une commande $u_{nom}(x)$ telle que l'origine du système (4.2.1) sans perturbation, i.e.*

$$\dot{x} = f(x, t) + g(x, t)u_{nom}(x) \quad (4.2.11)$$

soit asymptotiquement stable.

Afin de stabiliser le système incertain (4.2.1), la commande u est décomposée de la manière suivante :

$$u = u_{nom} + u_{disc} \quad (4.2.12)$$

u_{nom} rend l'origine du système (4.2.11) asymptotiquement stable. u_{disc} permet de rejeter l'effet des perturbations sur le système (4.2.1) pour tout $t \geq 0$.

Soit $s(x, t) \in \mathbb{R}$, la variable de glissement définie par :

$$s(x, t) = s_0(x) + s_{aux}(x, t) \quad (4.2.13)$$

Le terme $s_0(x)$ est une combinaison linéaire de l'état du système qui doit être choisi de telle sorte que la matrice $\frac{\partial s_0}{\partial x} g(x)$ soit non singulière. Le terme s_{aux} induit l'appellation intégrale et fournit un degré de liberté supplémentaire dans la construction de la variable de glissement.

Afin d'éliminer l'effet des perturbations, la commande équivalente de u_{disc} , notée u_{disc}^{eq} , qui décrit les trajectoires du système quand le régime glissant est obtenu, doit remplir la condition suivante :

$$g(x, t) u_{disc}^{eq} = -p(x, t) \quad (4.2.14)$$

De plus, en régime glissant, la dérivée de s le long des trajectoires du système, doit vérifier :

$$\begin{aligned} \dot{s} &= \dot{s}_0(x) + \dot{s}_{aux}(x, t) \\ &= \frac{\partial s_0}{\partial x} (f(x, t) + g(x, t)u_{nom} + g(x, t)u_{disc} + p(x, t)) + \dot{s}_{aux}(x, t) \\ &= 0 \end{aligned} \quad (4.2.15)$$

Les conditions (4.2.14) et (4.2.15) sont satisfaites lorsque :

$$\dot{s}_{aux}(x, t) = -\frac{\partial s_0}{\partial x} (f(x, t) + g(x, t)u_{nom}(x)) \quad (4.2.16)$$

$$s_{aux}(x, 0) = -s_0(x(0)) \quad (4.2.17)$$

La commande u_{disc} est construite de manière à forcer le régime glissant sur la surface $\{s = 0\}$ et s'écrit :

$$u_{disc} = -G(x) \operatorname{sign}(s) \quad (4.2.18)$$

où le gain $G(x)$ vérifie

$$G(x) > a(x)$$

Puisque la condition initiale $s_{aux}(x, 0)$ est déterminée à partir de la relation $s(x, 0) = 0$, le régime glissant est établi dès l'instant initial et la phase de convergence est éliminée. En régime glissant, étant donné que la relation (4.2.14) est vérifiée, la dynamique du système perturbé (4.2.1) sur la surface de glissement $\{s = 0\}$, n'est autre que la dynamique du système non perturbé (4.2.11). Par conséquent, l'origine du système (4.2.1) est asymptotiquement stable malgré la présence de perturbations bornées vérifiant la condition de recouvrement.

Remarque 4.3 Récemment, de nouveaux résultats, sur les propriétés de robustesse de la CMGI vis-à-vis d'une classe de perturbations plus large, ont été développés, notamment pour les perturbations "vanishing" dans [Cao et Xu, 2004] et les perturbations ne vérifiant pas la condition de recouvrement dans [Castanos et Fridman, 2006].

4.2.3 Commande par modes glissants d'ordre supérieur

Dans le milieu des années 80, une nouvelle technique de glissement, caractérisée par une commande discontinue agissant sur les dérivées d'ordre supérieur de la variable de glissement, a été proposée dans [Emelyanov et al., 1986].

Concepts de base

Considérons un système non linéaire incertain dont la dynamique est décrite par :

$$\dot{x} = f(x, t) + g(x, t)u \quad (4.2.19)$$

$$s = s(x, t) \quad (4.2.20)$$

où $x = [x_1, \dots, x_n]^T \in \mathcal{X} \subset \mathbb{R}^n$ représente l'état du système. La commande $u \in \mathcal{U} \subset \mathbb{R}$ est une fonction discontinue et bornée dépendante de l'état et du temps. f et g sont des champs de vecteurs suffisamment différentiables mais connus de façon incertaine.

Le problème posé est toujours de contraindre les trajectoires du système à évoluer sur la surface de glissement $s(x, t) = 0$, qui est ici une fonction à valeur réelle, suffisamment différentiable telle que ses $(\rho - 1)$ premières dérivées par rapport au temps ne soient fonctions que de l'état x (ce qui signifie qu'elles ne contiennent aucune discontinuité). Nous présentons ici brièvement la théorie des modes glissants d'ordre supérieur. Plus de précisions peuvent être trouvées dans [Emelyanov et al., 1986, Levant, 1993, Bartolini et al., 1999, Floquet, 2000, Fridman et Levant, 2002].

Définition 4.2 Soit le système non linéaire (4.2.19) et la variable de glissement $s(x, t)$ associée. L'ensemble de glissement d'ordre ρ par rapport à $s(x, t)$ est défini par :

$$\mathcal{S}_\rho = \left\{ x \in \mathcal{X} : s = \dot{s} = \dots = s^{(\rho-1)} = 0 \right\} \quad (4.2.21)$$

Remarque 4.4 Par abus de langage, l'ensemble \mathcal{S}_ρ est également appelé surface de glissement d'ordre ρ .

Définition 4.3 [Levant, 1993] Supposons que l'ensemble de glissement \mathcal{S}_ρ d'ordre ρ soit non vide et qu'il définisse localement un ensemble intégral au sens de Filippov. Alors, la dynamique satisfaisant (4.2.21) est appelée mode glissant d'ordre ρ par rapport à la variable de glissement s .

Définition 4.4 [Levant, 1993] On dit que la loi de commande u est un algorithme glissant idéal d'ordre ρ par rapport à \mathcal{S}_ρ si elle génère une solution au sens de Filippov sur \mathcal{S}_ρ , i.e.

$$s = \dot{s} = \dots = s^{(\rho-1)} = 0$$

Un régime glissant d'ordre un peut exister sur la surface $\mathcal{S} = \{x \in \mathcal{X} : s = 0\}$ si et seulement si le système (4.2.19) est de degré relatif un par rapport à la variable de glissement. La commande par modes glissants d'ordre supérieur permet de relâcher cette condition. Si le système est de degré relatif $\rho > 1$ par rapport à la variable de glissement, une CMG d'ordre ρ permettra d'obtenir une convergence en temps fini sur la surface en forçant les trajectoires d'état du système à être confinées au bout d'un temps fini dans l'ensemble de glissement d'ordre ρ .

Dans les définitions précédentes, il est supposé que l'ensemble de glissement d'ordre ρ est atteint exactement. Un tel régime glissant est qualifié d'idéal car les organes de commande doivent commuter à une fréquence infinie. Toutefois, ce n'est pas le cas en pratique du fait des imperfections de ces derniers. Par conséquent, le régime glissant ne prend place que dans un proche voisinage de la surface considérée. Ce comportement est qualifié de régime glissant réel. Il en résulte qu'un algorithme d'ordre ρ permettra, si la méthode d'intégration est à pas variable majoré par τ , d'obtenir la précision de convergence suivante [Fridman et Levant, 2002] :

$$|s| = O(\tau^\rho), |\dot{s}| = O(\tau^{\rho-1}), \dots, |s^{(\rho-1)}| = O(\tau) \quad (4.2.22)$$

Ainsi, obtenir une bonne précision de convergence d'un mode glissant requiert non seulement de maintenir la fonction contrainte à zéro, mais également ses dérivées successives. Ceci donne un argument supplémentaire aux modes glissants d'ordre supérieur. En effet, le développement précédent nous indique que pour un mode glissant classique, la précision de la convergence est de l'ordre de τ , alors qu'elle est de τ^ρ pour un mode glissant d'ordre ρ .

Générer un régime glissant asymptotiquement stable sur \mathcal{S}_ρ est possible en contraignant le système à glisser sur une surface auxiliaire qui est une simple combinaison linéaire des dérivées de la variable de glissement jusqu'à l'ordre $\rho-1$ [Emelyanov et al., 1986, Sira-Ramirez, 1993]. Cependant, les avantages de la commande par modes glissants (robustesse, précision) ne sont effectifs qu'une fois les trajectoires du système confinées dans l'ensemble de glissement \mathcal{S}_ρ , ce qui arrive lors d'une convergence en temps fini, mais pas lors d'une convergence asymptotique. Les algorithmes de commande par modes glissants d'ordre supérieur ne sont essentiellement connus que pour le cas $\rho = 2$. Certains travaux ont néanmoins permis de construire des commandes garantissant un régime glissant d'ordre quelconque en temps fini [Man et Yu, 1997, Levant, 2001, Levant, 2005]. Cependant, la nécessité de connaître avec précision les conditions initiales du système ou le réglage délicat des paramètres de la loi de commande leur confère généralement une difficulté de mise en œuvre pour les applications réelles.

Algorithmes par modes glissants d'ordre deux

Considérons le système non linéaire (4.2.19) avec $\rho = 1, 2$. Le but est de générer un régime glissant d'ordre deux par rapport à la variable de glissement s , c'est-à-dire de contraindre les trajectoires du

système à atteindre et se maintenir en temps fini dans l'ensemble de glissement \mathcal{S}_2 :

$$\mathcal{S}_2 = \{x \in \mathcal{X} : s = \dot{s} = 0\} \quad (4.2.23)$$

Ceci est réalisé par une commande agissant sur la dérivée seconde de la variable de glissement qui, de manière générale, peut s'écrire sous la forme :

$$\ddot{s} = \phi(x, t) + \varphi(x, t)v \quad (4.2.24)$$

avec

- $v = \dot{u}$ dans le cas où le système (4.2.19) est de degré relatif $\rho = 1$ par rapport à s ,
- $v = u$ dans le cas où le système (4.2.19) est de degré relatif $\rho = 2$ par rapport à s .

Afin de réaliser des algorithmes par modes glissants d'ordre deux, il est nécessaire de vérifier l'hypothèse de travail suivante pour valider l'atteignabilité de la surface de glissement et la bornitude de la variable \ddot{s} [Bartolini et al., 1999] :

Hypothèse 4.3 *Les fonctions incertaines $\varphi(x, t)$ et $\phi(x, t)$ sont bornées. Plus particulièrement, il existe quatre constantes positives S_0 , C_0 , K_m et K_M telles que, dans un voisinage $|s(x, t)| < S_0$, les inégalités suivantes soient vérifiées :*

$$|\phi(x, t)| < C_0 \quad \text{et} \quad 0 < K_m \leq \varphi(x, t) \leq K_M \quad (4.2.25)$$

Notons que cette hypothèse est relativement peu restrictive puisque si les fonctions ϕ et φ sont continues sur un compact où la fonction φ ne s'annule pas, elle est automatiquement vérifiée. On pourra donc se ramener à un compact afin d'appliquer les algorithmes suivants. En effet, il est facile de synthétiser une commande ralliant ce compact en temps fini si ce dernier n'est pas un ensemble singulier de l'équation différentielle.

Différents algorithmes menant au comportement désiré peuvent être trouvés dans la littérature. Nous en présenterons deux qui seront utilisés par la suite. Le premier est appelé algorithme de “twisting échantillonné” et le second, “algorithme du super-twisting” et sont dûs à [Fridman et Levant, 2002].

L'algorithme du **twisting échantillonné** est utilisé lorsque le degré relatif du système par rapport à s est $\rho = 2$. Son intérêt est que, non seulement, il ne requiert pas d'information sur la dérivée de la surface considérée, mais qu'il prend également en compte des contraintes d'ordre pratique telles que l'échantillonnage des mesures et de la loi de commande. La convergence en temps fini vers l'origine du plan de phase (s, \dot{s}) est due à la commutation de la commande entre deux valeurs, de telle manière que, à chaque commutation, les axes des abscisses et des ordonnées soient croisés de plus en plus près de l'origine (figure 4.3). La commande se présente de la manière suivante [Fridman et Levant, 2002] :

$$v = \begin{cases} -\lambda_m \operatorname{sign}(s), & \text{si } s\Delta_s \leq 0 \\ -\lambda_M \operatorname{sign}(s), & \text{si } s\Delta_s > 0 \end{cases} \quad (4.2.26)$$

avec

$$\Delta_s = \begin{cases} 0, & k = 0 \\ s(k\tau) - s((k-1)\tau), & k \geq 1 \end{cases} \quad (4.2.27)$$

où τ est la période d'échantillonnage et $k \in \mathbb{N}$. Les conditions suffisantes sur les gains λ_m et λ_M assurant la convergence en temps fini vers l'origine du plan de phase sont données par :

$$\begin{cases} \lambda_m > \frac{C_0}{K_m} \\ \lambda_M > 4\frac{K_m}{s_0} \\ K_m\lambda_M - C_0 > K_M\lambda_m + C_0 \end{cases} \quad (4.2.28)$$

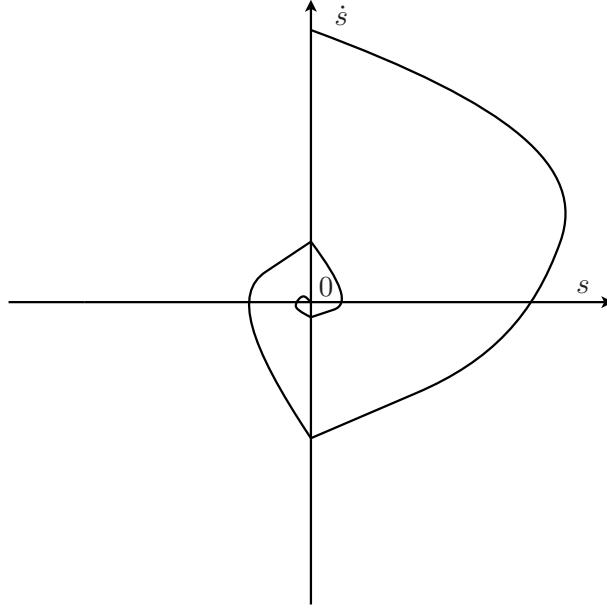


FIG. 4.3 – Plan de phase de l'algorithme du “twisting échantilloné”.

L'algorithme du *super-twisting* a été introduit pour des systèmes de degré relatif $\rho = 1$ par rapport à la variable de glissement. La loi de commande est constituée de deux termes continus qui, une fois encore ne dépendent pas de la dérivée de la variable de glissement. La discontinuité n'intervient ici que sur la première dérivée temporelle de l'entrée de commande u , ce qui permet d'éviter le phénomène de réticence. La commande s'écrit :

$$u = u_1 + u_2 \quad (4.2.29)$$

avec

$$\begin{cases} \dot{u}_1 = -\alpha \operatorname{sign}(s) \\ u_2 = -\lambda |s|^{\frac{1}{2}} \operatorname{sign}(s) \end{cases}$$

Les conditions suffisantes de convergence en temps fini sur l'ensemble de glissement (4.2.23) sont donnés par [Levant, 1993] :

$$\alpha > \frac{C_0}{K_m}, \quad \lambda > 0 \quad \text{et} \quad \lambda^2 \geq \frac{4C_0}{K_m^2} \frac{K_M\alpha + C_0}{K_m\alpha + C_0}$$

Algorithmes par modes glissants d'ordre quelconque

Contrairement aux modes glissants d'ordre deux, il existe très peu de résultats sur les algorithmes de commande par modes glissants d'ordre quelconque. Récemment, de nouveaux algorithmes apparaissent. L'algorithme de CMG proposé dans [Man et Yu, 1997] est, à notre connaissance, le premier travail proposant une solution à ce problème. Il est basé sur les propriétés de convergence en temps fini des équations différentielles non linéaires. Cependant, il souffre d'un problème de singularité autour de l'origine.

Plusieurs algorithmes proches de celui proposé dans [Man et Yu, 1997], ont été développés dans [Levant, 2001, Levant, 2003, Levant, 2005]. Bien qu'ils permettent de pallier au problème de singularité autour de l'origine, un problème se pose sur le choix des gains de commande. L'algorithme est peu constructif par manque de conditions nécessaires et/ou suffisantes relatives aux paramètres de la loi de commande, rendant le réglage de cette dernière empirique, souvent difficile et fastidieux.

Enfin, d'autres algorithmes de commande par modes glissants d'ordre supérieur ont été récemment introduits dans [Laghrouche et al., 2006, Laghrouche et al., 2007]. Le premier algorithme est basé, d'une part, sur la commande linéaire quadratique sur un horizon de temps fini avec contraintes sur l'état final et, d'autre part, sur la commande par modes glissants d'ordre un. Quant au second, il est basé sur la minimisation d'un critère quadratique et sur le concept de CMGI. Bien que ces algorithmes soient généraux, la connaissance précise et *a priori* des conditions initiales du système restreint considérablement l'applicabilité de cette approche.

Dans la section suivante, une approche *originale* de loi de commande par modes glissants d'ordre supérieur est proposée. Elle présente les caractéristiques suivantes :

- un régime glissant d'ordre quelconque en temps fini sans nécessiter la connaissance précise et *a priori* des conditions initiales du système,
- des conditions suffisantes de convergence,
- une méthode simple pour le réglage des paramètres de la loi de commande en vue d'atteindre les performances désirées.

4.3 Une solution originale pour la CMG d'ordre supérieur

4.3.1 Formulation du problème

Considérons le système non linéaire (4.2.19) ayant un degré relatif $\rho \in \mathbb{N}^+$ constant et connu par rapport à la variable de glissement $s(x, t)$. Le but est de générer un régime glissant d'ordre ρ par rapport à la variable de glissement $s(x, t)$, c'est-à-dire de contraindre les trajectoires du système à atteindre et se maintenir en temps fini dans l'ensemble de glissement \mathcal{S}_ρ défini par l'équation (4.2.21). L'objectif est donc de synthétiser une loi de commande u permettant la stabilisation en temps fini du

système incertain suivant :

$$\left\{ \begin{array}{lcl} \dot{z}_1 & = & z_2 \\ \dot{z}_2 & = & z_3 \\ \vdots & & \\ \dot{z}_{\rho-1} & = & z_\rho \\ \dot{z}_\rho & = & \phi(x, t) + \varphi(x, t)u \end{array} \right. \quad (4.3.1)$$

avec

$$\left\{ \begin{array}{lcl} z = [z_1, z_2, \dots, z_{\rho-1}, z_\rho]^T & = & [s, \dot{s}, \dots, s^{(\rho-2)}, s^{(\rho-1)}]^T \\ \phi(x, t) & = & L_f^\rho s(x, t) \\ \varphi(x, t) & = & L_g L_f^{\rho-1} s(x, t) \end{array} \right.$$

Les fonctions ϕ et φ peuvent généralement se décomposer en une partie nominale connue, notée $\bar{\phi}$ et $\bar{\varphi}$, et une autre, inconnue, relative aux incertitudes et aux perturbations, notée δ_ϕ et δ_φ , c'est-à-dire :

$$\left\{ \begin{array}{lcl} \phi & = & \bar{\phi} + \delta_\phi \\ \varphi & = & \bar{\varphi} + \delta_\varphi \end{array} \right.$$

Hypothèse 4.4 *On suppose que la partie nominale $\bar{\varphi}$ est inversible.*

En utilisant un retour d'état de la forme :

$$u = \bar{\varphi}^{-1} (w - \bar{\phi}) \quad (4.3.2)$$

où $w \in \mathbb{R}$ est la nouvelle commande, le système (4.3.1) devient :

$$\left\{ \begin{array}{lcl} \dot{z}_1 & = & z_2 \\ \dot{z}_2 & = & z_3 \\ \vdots & & \\ \dot{z}_{\rho-1} & = & z_\rho \\ \dot{z}_\rho & = & \vartheta(x, t) + (1 + \zeta(x, t)) w \end{array} \right. \quad (4.3.3)$$

où les fonctions ϑ et ζ définies par :

$$\left\{ \begin{array}{lcl} \vartheta & = & \delta_\phi - \delta_\varphi \bar{\varphi}^{-1} \bar{\phi} \\ \zeta & = & \delta_\varphi \bar{\varphi}^{-1} \end{array} \right.$$

regroupent les incertitudes du système. Pour la suite du travail, il est nécessaire de vérifier l'hypothèse suivante :

Hypothèse 4.5 *Les fonctions $\vartheta(x, t)$ et $\zeta(x, t)$ sont bornées. Plus particulièrement, il existe une fonction positive $a(x)$ et une constante positive $0 < b \leq 1$ telles que :*

$$\left\{ \begin{array}{l} |\vartheta(x, t)| \leq a(x) \\ |\zeta(x, t)| \leq 1 - b \end{array} \right. \quad (4.3.4)$$

Le système (4.3.3) peut être vu comme une chaîne perturbée de ρ intégrateurs. Pour le stabiliser à zéro en temps fini, il est possible de faire appel à des techniques de stabilisation en temps fini issues des propriétés d'homogénéité du système. Cependant, les méthodes existantes ne sont généralement pas constructives ou ne sont pas robustes vis-à-vis des perturbations. Après un rappel des principales commandes permettant de stabiliser en temps fini une chaîne non perturbée de ρ intégrateurs (c'est-à-dire avec $\vartheta(x, t) = 0$ et $\zeta(x, t) = 0$), une commande robuste stabilisant le système (4.3.3) sera synthétisée.

4.3.2 Rappel sur la stabilisation en temps fini d'une chaîne de ρ intégrateurs

Introduisons dans un premier temps, les concepts de stabilité en temps fini et d'homogénéité.

Définition 4.5 [Bacciotti et Rosier, 2005] Considérons le système de la forme :

$$\dot{x} = f(x) \quad (4.3.5)$$

où $x = [x_1, \dots, x_n]^T \in \mathcal{X} \subset \mathbb{R}^n$ représente l'état du système avec \mathcal{X} un ouvert de \mathbb{R}^n et $f = [f_1, \dots, f_n]^T : \mathcal{X} \rightarrow \mathbb{R}^n$ est continue. Le système (4.3.5) est (localement) **stable en temps fini** si :

- il est asymptotiquement stable,
- il existe un voisinage \mathcal{V} de l'origine tel que pour toute condition initiale $x_0 \in \mathcal{V} \setminus \{0\}$, il existe un temps d'établissement $T_{eta} > 0$ défini de manière à ce que toute solution $x(t)$ du système (4.3.5) vérifie :
 - $x(t) \in \mathcal{V} \setminus \{0\}$ pour tout $t \in [0, T_{eta}]$,
 - $\lim_{t \rightarrow T_{eta}} x(t) = 0$.

Enfin, lorsque $\mathcal{V} = \mathbb{R}^n$, l'origine est globalement stable en temps fini.

Définition 4.6 [Bacciotti et Rosier, 2005] Le système (4.3.5) est **homogène** de degré d ($d \in \mathbb{R}$) s'il existe $(r_1, \dots, r_n) \in (\mathbb{R}^+)^n$ tels que

$$f_i(\lambda^{r_1}x_1, \dots, \lambda^{r_n}x_n) = \lambda^{r_i+d}f_i(x_1, \dots, x_n), \quad \forall i = 1, \dots, n$$

pour tout $\lambda > 0$. $(\lambda^{r_1}x_1, \dots, \lambda^{r_n}x_n)$ est une famille de dilatation du système (4.3.5).

Théorème 4.2 [Bhat et Bernstein, 1997] Supposons que le système (4.3.5) soit homogène de degré $d < 0$, alors il est stable en temps fini si et seulement si il est asymptotiquement stable.

Considérons la chaîne non perturbée de ρ intégrateurs, découlant du système (4.3.3) :

$$\left\{ \begin{array}{lcl} \dot{z}_1 & = & z_2 \\ \dot{z}_2 & = & z_3 \\ & \vdots & \\ \dot{z}_{\rho-1} & = & z_\rho \\ \dot{z}_\rho & = & w \end{array} \right. \quad (4.3.6)$$

Il existe différentes approches pour parvenir à la stabilisation en temps fini du système (4.3.6). On peut citer, par exemple, celles qui consistent à déterminer une commande qui rende le système bouclé homogène de degré négatif, établissant ainsi une équivalence entre stabilisation asymptotique et stabilisation en temps fini. Cette approche est utilisée pour stabiliser le système (4.3.6) dans [Bhat et Bernstein, 1997], [Bhat et Bernstein, 1998] dans le cas $\rho = 2$ et dans [Hong, 2002], [Bhat et Bernstein, 2005] pour le cas général.

Cependant, dans [Hong, 2002], l'algorithme est peu constructif par manque de conditions nécessaires et/ou suffisantes relatives aux paramètres de la commande. Ceci rend son réglage difficile et fastidieux. Une autre technique développée dans [Praly, 1997], basée sur des fonctions homogènes, a aussi permis de résoudre le problème de stabilisation en temps fini du système (4.3.6). Cependant, cette algorithme nécessite une phase d'initialisation assez complexe. Plus de renseignements sur les techniques de stabilisation en temps fini peuvent être trouvés dans [Moulay, 2005].

Les seuls travaux, à notre connaissance, qui proposent une méthode constructive de synthèse d'une loi de commande garantissant une stabilisation en temps fini du système (4.3.6), sont développés dans [Bhat et Bernstein, 2005]. L'algorithme, basé sur une approche géométrique, est énoncé dans le théorème suivant.

Théorème 4.3 [Bhat et Bernstein, 2005] *Soient les constantes positives k_1, \dots, k_ρ telles que le polynôme $p^\rho + k_\rho p^{\rho-1} + \dots + k_2 p + k_1$ soit Hurwitz. Il existe $\epsilon \in (0, 1)$ tel que pour tout $\nu \in (1 - \epsilon, 1)$, le système (4.3.6) soit stabilisable en temps fini par le retour continu :*

$$w(z) = -k_1 \operatorname{sign}(z_1) |z_1|^{\nu_1} - \dots - k_\rho \operatorname{sign}(z_\rho) |z_\rho|^{\nu_\rho} \quad (4.3.7)$$

où ν_1, \dots, ν_ρ vérifient :

$$\nu_{i-1} = \frac{\nu_i \nu_{i+1}}{2\nu_{i+1} - \nu_i}, \quad i = 2, \dots, \rho, \quad (4.3.8)$$

avec $\nu_\rho = \nu$ et $\nu_{\rho+1} = 1$.

La démonstration de ce résultat est réalisée en utilisant une fonction dépendant continûment de ν . Nous renvoyons le lecteur à l'article [Bhat et Bernstein, 2005] pour de plus amples informations.

Exemple 4.1 En utilisant le théorème 4.3, on peut démontrer que le triple intégrateur non perturbé est stabilisable en temps fini par la commande :

$$w(z) = -\operatorname{sign}(z_1) |z_1|^{\frac{1}{2}} - 1.5 \operatorname{sign}(z_2) |z_2|^{\frac{3}{5}} - 1.5 \operatorname{sign}(z_3) |z_3|^{\frac{3}{4}} \quad (4.3.9)$$

obtenue en prenant $\rho = 3$, $k_1 = 1$, $k_2 = 1.5$, $k_3 = 1.5$ et $\nu = \frac{3}{4}$. La figure 4.4 donne la réponse temporelle du système commandé par $w(z)$. On peut remarquer que le système se stabilise en temps fini, avec T_{eta} de l'ordre de 20s. Néanmoins, lorsque le triple intégrateur est perturbé, la stabilisation

du système n'est plus garantie. Ainsi, en considérant le système :

$$\begin{cases} \dot{z}_1 = z_2 \\ \dot{z}_2 = z_3 \\ \dot{z}_3 = w + p(z) \end{cases} \quad (4.3.10)$$

où $p(z) = \sin(10z_1)$ est une perturbation bornée, la commande (4.3.9) n'assure plus la stabilisation du système. Les résultats associés sont représentés dans la figure 4.5.

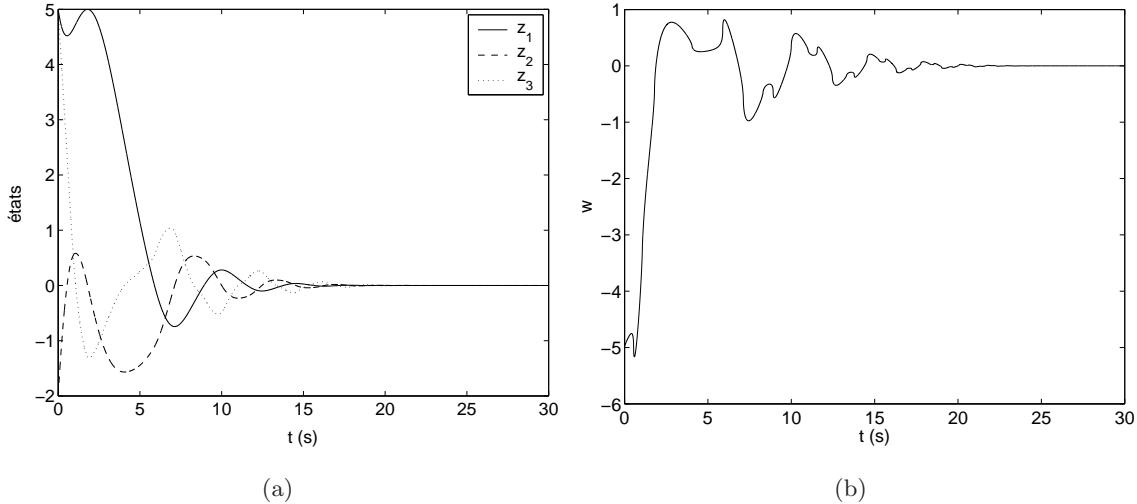


FIG. 4.4 – Stabilisation en temps fini du triple intégrateur sans perturbation. (a) Réponse temporelle du système. (b) Commande appliquée.

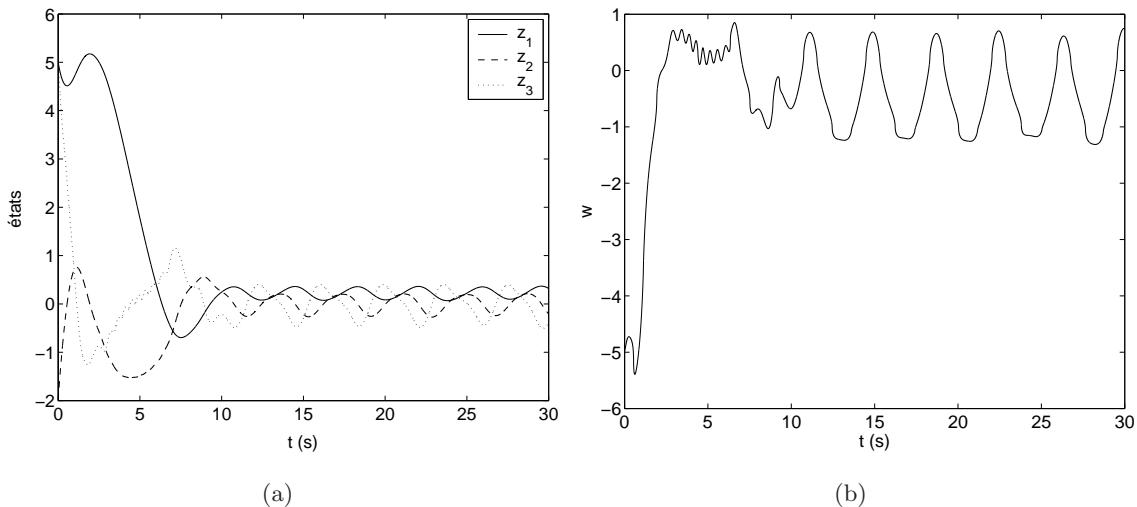


FIG. 4.5 – Réponse du triple intégrateur perturbé avec la commande temps fini [Bhat et Bernstein, 2005]. (a) Trajectoire $z(t)$. (b) Commande appliquée.

Ainsi, les méthodes existantes permettant la stabilisation en temps fini d'une chaîne de ρ intégrateurs ne sont généralement pas constructives ou ne sont pas robustes vis-à-vis des perturbations. L'idée est de généraliser les résultats de [Bhat et Bernstein, 2005] en synthétisant une commande robuste qui garantisse la stabilisation en temps fini du système (4.3.3).

4.3.3 Synthèse de la commande robuste en temps fini [Defoort et al., 2006b]

Idée de base

Notre algorithme tire son fondement du principe de la CMGI. En effet, dans un premier temps, on synthétise une loi de commande $w_{nom}(z)$ permettant de stabiliser en temps fini le système sans perturbation (4.3.6). Puis, une loi de commande discontinue $w_{disc}(z)$ est construite de manière à rejeter l'effet des incertitudes $\vartheta(x, t)$ et $\zeta(x, t)$ sur le système (4.3.3). Grâce à l'application de ces deux termes, les trajectoires d'état du système (4.3.3) sont stabilisées en $z = 0$ en temps fini. Ainsi, un régime glissant d'ordre ρ par rapport à la variable de glissement $s(x, t)$ prend place.

Néanmoins, la différence majeure se situe au niveau de la construction de la variable auxiliaire de glissement. En effet, la connaissance *a priori* de la condition initiale de la surface de glissement (voir équation (4.2.17)), indispensable à la synthèse de la CMGI, n'est plus nécessaire dans notre algorithme de commande robuste en temps fini. L'idée n'est plus d'éliminer la phase de convergence vers la surface de glissement mais plutôt de construire une loi de commande robuste en temps fini qui ne nécessite plus la connaissance précise et *a priori* des conditions initiales du système.

Afin de stabiliser en temps fini le système incertain (4.3.3), la loi de commande $w(z)$ est décomposée de la manière suivante :

$$\begin{cases} w(z) &= w_{nom}(z) + w_{disc}(z, z_{aux}) \\ \dot{z}_{aux} &= -w_{nom}(z) \end{cases} \quad (4.3.11)$$

La fonction auxiliaire $z_{aux}(t) \in \mathbb{R}$, qui engendre une nouvelle dynamique dans le système, sera utilisée, par la suite, pour la synthèse de la variable de glissement associée à la commande discontinue $w_{disc}(z, z_{aux})$. La commande nominale $w_{nom}(z)$ est décrite dans la section précédente (i.e. équation (4.3.7)).

Synthèse de la variable de glissement associée à la commande discontinue $w_{disc}(z, z_{aux})$

Soit $\sigma(z, t) \in \mathbb{R}$ la fonction définie par :

$$\sigma(z, t) = z_\rho(t) + z_{aux}(t) \quad (4.3.12)$$

La dérivée première de σ par rapport au temps est donnée par :

$$\begin{aligned} \dot{\sigma} &= \dot{z}_\rho + \dot{z}_{aux} \\ &= \vartheta(x, t) + (1 + \zeta(x, t))(w_{nom}(z) + w_{disc}(z, z_{aux})) - w_{nom}(z) \\ &= \vartheta(x, t) + (1 + \zeta(x, t))w_{disc}(z, z_{aux}) + \zeta(x, t)w_{nom}(z) \end{aligned}$$

Supposons qu'un régime glissant prenne place sur la surface $\{\sigma = 0\}$. Ainsi, lorsque ce régime glissant est établi, la commande équivalente de $w_{disc}(z, z_{aux})$, notée $w_{disc}^{eq}(z, z_{aux})$ et obtenue par la relation $\dot{\sigma} = 0$ (pour plus de détails sur la commande équivalente, on peut se référer à [Utkin, 1992]), est définie par l'équation suivante :

$$\vartheta(x, t) + (1 + \zeta(x, t)) w_{disc}^{eq}(z, z_{aux}) + \zeta(x, t) w_{nom}(z) = 0 \quad (4.3.13)$$

Par conséquent, la dynamique du système perturbé (4.3.3) sur la surface de glissement $\{\sigma = 0\}$, obtenue en remplaçant la commande w par la commande équivalente associée $w_{nom} + w_{disc}^{eq}$, est décrite par :

$$\left\{ \begin{array}{rcl} \dot{z}_1 & = & z_2 \\ \dot{z}_2 & = & z_3 \\ \vdots & & \\ \dot{z}_{\rho-1} & = & z_\rho \\ \dot{z}_\rho & = & w_{nom}(z) \end{array} \right. \quad (4.3.14)$$

qui n'est autre que la dynamique du système non perturbé (4.3.6) et ainsi qu'il a été vu dans la section précédente, elle converge vers zéro en un temps fini.

Synthèse de la commande discontinue $w_{disc}(z, z_{aux})$

Le rôle de la commande discontinue est de forcer le système à rentrer sur la surface $\{\sigma = 0\}$. Elle s'écrit de la manière suivante :

$$w_{disc}(z, z_{aux}) = -G(z) \operatorname{sign}(\sigma) \quad (4.3.15)$$

où le gain $G(z)$ est une fonction positive, définie par :

$$G(z) \geq \frac{(1-b)|w_{nom}(z)| + a(x) + \eta}{b}, \quad \eta > 0. \quad (4.3.16)$$

Théorème 4.4 [Defoort et al., 2006b, Defoort et al., 2007c] Soit le système non linéaire (4.2.19) de degré relatif ρ par rapport à la variable de glissement $s(x, t)$. Si les hypothèses 4.4 et 4.5 sont vérifiées, alors la commande :

$$u = \overline{\varphi}^{-1} (w_{nom}(z) + w_{disc}(z, z_{aux}) - \overline{\phi})$$

où $\dot{z}_{aux} = -w_{nom}(z)$, $w_{nom}(z)$ et $w_{disc}(z, z_{aux})$ sont données par les relations (4.3.7) et (4.3.12), (4.3.15), (4.3.16), respectivement, permet l'établissement d'un régime glissant d'ordre ρ par rapport à la variable de glissement $s(x, t)$.

Démonstration. Considérons la fonction candidate de Lyapunov :

$$V = \frac{1}{2} \sigma^2$$

La dérivée de V , le long des trajectoires du système (4.3.3), s'écrit :

$$\begin{aligned}\dot{V} &= \sigma (\vartheta(x, t) + (1 + \zeta(x, t)) w_{disc}(z, z_{aux}) + \zeta(x, t) w_{nom}(z)) \\ &= \sigma (-G(z) (1 + \zeta(x, t)) \text{sign}(\sigma) + \vartheta(x, t) + \zeta(x, t) w_{nom}(z)) \\ &= -G(z)|\sigma| - G(z)\zeta(x, t)|\sigma| + \vartheta(x, t)\sigma + \zeta(x, t)w_{nom}(z)\sigma \\ &\leq -G(z)|\sigma| + G(z)|\zeta(x, t)||\sigma| + |\vartheta(x, t)||\sigma| + |\zeta(x, t)||w_{nom}(z)||\sigma|\end{aligned}$$

Puisque l'hypothèse 4.5 est vérifiée, en utilisant la relation (4.3.16), on peut réécrire la dernière inégalité de la manière suivante :

$$\begin{aligned}\dot{V} &\leq -G(z)|\sigma| + G(z)(1 - b)|\sigma| + a(z)|\sigma| + (1 - b)|w_{nom}(z)||\sigma| \\ &\leq -G(z)|\sigma| + (1 - b)(G(z) + |w_{nom}(z)|)|\sigma| + a(x)|\sigma| \\ &\leq (-bG(z) + (1 - b)|w_{nom}(z)| + a(x))|\sigma| \\ &\leq -\eta|\sigma|\end{aligned}$$

Ainsi, les trajectoires d'état du système convergent vers la surface de glissement $\{\sigma = 0\}$ en temps fini et sont ensuite maintenues sur celle-ci malgré la présence d'incertitudes sur le système. Comme il a été décrit précédemment, lorsque ce régime glissant est établi, la dynamique du système perturbé (4.3.3) sur la surface de glissement $\{\sigma = 0\}$, n'est autre que la dynamique du système non perturbé (4.3.6). Etant donné que la loi de commande $w_{nom}(z)$ est élaborée en utilisant le théorème 4.3, la dynamique du système (4.3.6) converge vers zéro en temps fini. Par conséquent, un régime glissant d'ordre ρ par rapport à la variable de glissement $s(x, t)$ prend place avec un temps de convergence fini. ■

Remarque 4.5 *On peut noter que cet algorithme de commande par modes glissants d'ordre ρ par rapport à la variable de glissement $s(x, t)$ permet d'obtenir la précision de convergence $|s| = O(\tau^\rho)$ et de bonnes propriétés de robustesse vis-à-vis des perturbations.*

Exemple 4.2 Afin d'illustrer l'algorithme précédent, reprenons l'exemple 4.1 du triple intégrateur perturbé (4.3.10). En utilisant le théorème 4.4, on peut démontrer que le système (4.3.10) est stabilisable en temps fini par la commande :

$$\begin{aligned}w(z) &= w_{nom}(z) + w_{disc}(z, z_{aux}) \\ \dot{z}_{aux} &= -w_{nom}(z) \\ w_{nom}(z) &= -\text{sign}(z_1)|z_1|^{\frac{1}{2}} - 1.5 \text{sign}(z_2)|z_2|^{\frac{3}{5}} - 1.5 \text{sign}(z_3)|z_3|^{\frac{3}{4}} \\ w_{disc}(z, z_{aux}) &= -1.5 \left(\frac{2}{\pi} \arctan \left(\frac{z_3 + z_{aux}}{0.001} \right) \right)\end{aligned}$$

La partie continue $w_{nom}(z)$ est la même que dans l'exemple 4.1. La partie discontinue, choisie de manière à compenser la perturbation $p(z)$, force l'établissement d'un régime glissant sur la surface de glissement $\{z_3 + z_{aux} = 0\}$. Etant donné que pour cet exemple $a(z) = 1$ et $b = 1$, à partir de l'équation (4.3.16), on en déduit que le gain G doit vérifier $G > 1$ et est fixé à 1.5 ($\eta = 0.5$). Dans la simulation

donnée en figure 4.6, une version lissée de la fonction $\text{sign}(z_3 + z_{\text{aux}})$ a été utilisée afin de réduire la réticence : $\frac{2}{\pi} \arctan\left(\frac{z_3 + z_{\text{aux}}}{0.001}\right)$. Les résultats de simulation, montrant la convergence de z_1 , z_2 et z_3 vers zéro en temps fini avec un temps d'établissement $T_{\text{eta}} = 25\text{s}$, sont donnés dans la figure 4.6. On peut remarquer que la réponse du système (4.3.10) est celle voulue malgré les perturbations, ce qui n'a pas été le cas avec l'algorithme [Bhat et Bernstein, 1997] utilisé seul.

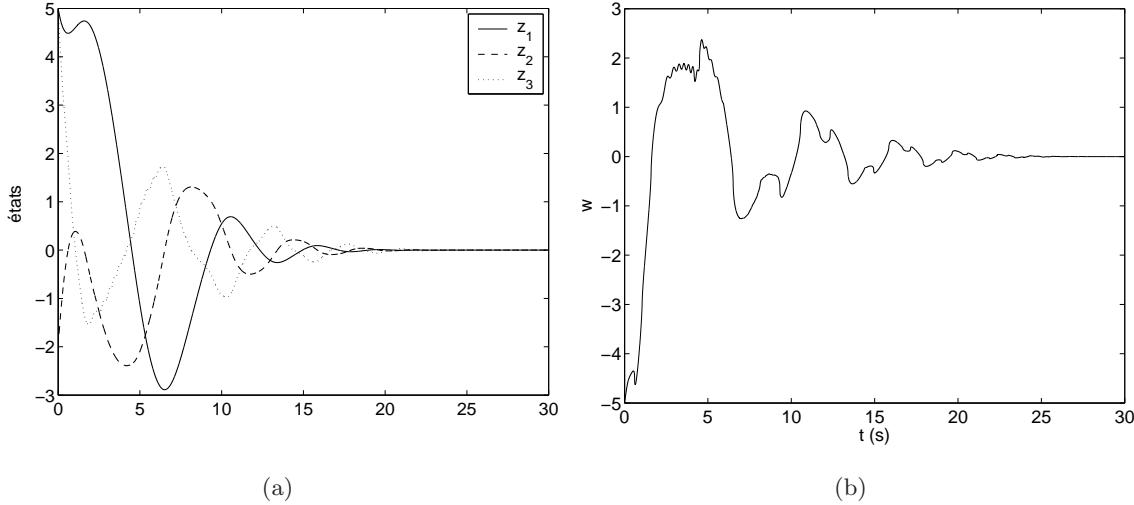


FIG. 4.6 – Stabilisation en temps fini du triple intégrateur perturbé. (a) Réponse temporelle du système. (b) Commande appliquée.

4.4 Applications de la CMG d'ordre supérieur

4.4.1 Commande d'un aéroglisseur [Defoort et al., 2006b]

Afin d'illustrer notre algorithme de CMG d'ordre supérieur, on considère l'exemple de la commande d'un aéroglisseur [Fantoni et al., 2000, Sira-Ramirez, 2002] (voir la figure 4.7).

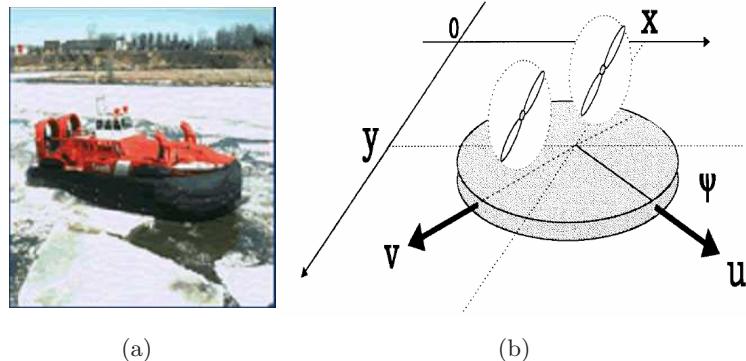


FIG. 4.7 – Aéroglisseur ou “hovercraft” en anglais.

Modèle de l'aéroglissoir et objectif de commande

Le comportement dynamique de ce système, sur la base d'hypothèses simplificatrices, est décrit par [Fantoni et al., 2000] :

$$\left\{ \begin{array}{l} \dot{x} = u \cos \psi - v \sin \psi \\ \dot{y} = u \sin \psi + v \cos \psi \\ \dot{\psi} = w \\ \dot{u} = vw + \tau_u \\ \dot{v} = -uw - \beta v + \delta_v(x) \\ \dot{w} = \tau_w + \delta_w(t) \end{array} \right. \quad (4.4.1)$$

où x , y et ψ représentent la position et l'orientation de l'aéroglissoir dans le référentiel terrestre, u la vitesse d'avance, v la vitesse de glissement et w la vitesse de lacet. Les commandes τ_u et τ_w sont respectivement la force et le couple appliqués au système. Des perturbations, affectant la dynamique sur la vitesse de lacet (vérifiant la condition de recouvrement) et affectant la dynamique sur la vitesse de glissement (ne vérifiant pas la condition de recouvrement), sont ajoutées. La perturbation $\delta_w(t)$ est un bruit blanc de moyenne nulle et de variance 0.2. Quant à la perturbation $\delta_v(x)$, elle sert à modéliser, par exemple, l'effet d'une vague sur l'aéroglissoir et est de la forme :

$$\delta_v(x) = 0.15 (\sin(10x) + 0.2 \cos(10x))$$

L'objectif de commande consiste à amener les trajectoires du système (4.4.1) à poursuivre en temps fini les trajectoires de référence $x_{ref}(t) = 10 \cos(\frac{\pi}{6}t)$ et $y_{ref}(t) = 10 \sin(\frac{\pi}{6}t)$.

Hypothèse 4.6 *La quantité $\beta u + \tau_u$ vérifie la relation :*

$$\beta u + \tau_u \geq C > 0 \quad (4.4.2)$$

Par conséquent, on se restreint au cas de la poursuite d'une trajectoire non stationnaire qui ne comporte pas de points d'arrêt. On peut se référer à [Sira-Ramirez, 2002] pour de plus amples détails sur la signification physique de cette hypothèse.

Commande par modes glissants d'ordre 4 – 4

En choisissant comme variables de glissement $s_1 = x - x_{ref}$ et $s_2 = y - y_{ref}$, le but de la commande est d'annuler en temps fini $s = [s_1, s_2]^T$. Etant donné que le degré relatif du système (4.4.1) par rapport à la variable de glissement s n'est pas défini (les dérivées secondes \ddot{x} et \ddot{y} font explicitement apparaître τ_u mais pas τ_w), une extension dynamique du système (4.4.1) est nécessaire avant de synthétiser la commande. Ainsi, une chaîne d'intégrateurs de dimension deux est ajoutée au niveau de l'entrée τ_u .

Une raison motivant l'utilisation d'une commande par modes glissants d'ordre supérieur réside dans les propriétés structurelles du système. En effet, le système étendu (système (4.4.1) et chaîne

d'intégrateurs de dimension deux au niveau de l'entrée τ_u) a un degré relatif égal à $[4, 4]^T$ par rapport à s . En dérivant 4 fois s_1 et s_2 , on obtient :

$$\begin{bmatrix} s_1^{(4)} \\ s_2^{(4)} \end{bmatrix} = (\bar{\varphi} + \delta_\varphi) \begin{bmatrix} \ddot{\tau}_u \\ \tau_w \end{bmatrix} + (\bar{\phi} + \delta_\phi) \quad (4.4.3)$$

avec

$$\begin{aligned} \bar{\varphi} &= \begin{bmatrix} \cos \psi & \cos \psi(\beta v) - \sin \psi(\beta u + \tau_u) \\ \sin \psi & \sin \psi(\beta v) + \cos \psi(\beta u + \tau_u) \end{bmatrix} \\ \delta_\varphi &= \begin{bmatrix} 0 & -\delta_v \cos \psi \\ 0 & -\delta_v \sin \psi \end{bmatrix} \\ \bar{\phi} &= \begin{bmatrix} -x_{ref}^{(4)} - w \cos \psi (w(2\beta u + \tau_u) + 2\beta^2 v) + \sin \psi (\beta^3 v + w(\beta^2 u - 2\dot{\tau}_u - \beta \tau_u) - 2w^2 \beta v) \\ -y_{ref}^{(4)} - w \sin \psi (w(2\beta u + \tau_u) + 2\beta^2 v) - \cos \psi (\beta^3 v + w(\beta^2 u - 2\dot{\tau}_u - \beta \tau_u) - 2w^2 \beta v) \end{bmatrix} \\ \delta_\phi &= \begin{bmatrix} \left(\dot{\delta}_v w^2 + \dot{\delta}_v \beta - \ddot{\delta}_v - \delta_w \tau_u - \delta_v \beta^2 - \delta_w \beta u \right) \sin \psi + \left(2\delta_v \beta w + \delta_w \beta v - 2\dot{\delta}_v w - \delta_v \delta_w \right) \cos \psi \\ \left(\delta_w \tau_u + \delta_v \beta^2 - \dot{\delta}_v \beta - \delta_v w^2 + \ddot{\delta}_v + \delta_w \beta u \right) \cos \psi + \left(2\delta_v \beta w - \delta_v \delta_w - 2\dot{\delta}_v w + \delta_w \beta v \right) \sin \psi \end{bmatrix} \end{aligned}$$

Les termes $\bar{\varphi}$ et $\bar{\phi}$ sont connus tandis que les termes δ_φ et δ_ϕ , relatifs aux perturbations δ_v et δ_w sont inconnus. Sous l'hypothèse 4.6, la matrice $\bar{\varphi}$ est inversible (car $\det \bar{\varphi} = \beta u + \tau_u$). Ainsi, en utilisant le retour d'état :

$$\begin{bmatrix} \ddot{\tau}_u \\ \tau_w \end{bmatrix} = \bar{\varphi}^{-1} (\tau - \bar{\phi}) \quad (4.4.4)$$

où $\tau \in \mathbb{R}^2$ est la nouvelle commande, le système (4.4.3) devient :

$$\begin{bmatrix} s_1^{(4)} \\ s_2^{(4)} \end{bmatrix} = [I_2 + \zeta] \tau + \vartheta \quad (4.4.5)$$

où ϑ et ζ , définies par :

$$\begin{cases} \vartheta &= \delta_\phi - \delta_\varphi \bar{\varphi}^{-1} \bar{\phi} \\ \zeta &= \delta_\varphi \bar{\varphi}^{-1} \end{cases}$$

regroupent les perturbations du système. Il est important de noter que $\|\zeta\| < \frac{2|\delta_v|}{C}$. Si $\frac{2|\delta_v|}{C} < 1$, il est possible d'utiliser les résultats présentés dans le théorème 4.4. Aussi, la loi de commande s'écrit :

$$\begin{aligned} \tau &= \tau_{nom} + \tau_{disc} \\ \dot{z}_{aux} &= -\tau_{nom} \\ \tau_{nom} &= -24 \left[\text{sign}(s_1) |s_1|^{\frac{3}{7}}, \text{sign}(s_2) |s_2|^{\frac{3}{7}} \right]^T - 50 \left[\text{sign}(\dot{s}_1) |\dot{s}_1|^{\frac{1}{2}}, \text{sign}(\dot{s}_2) |\dot{s}_2|^{\frac{1}{2}} \right]^T \\ &\quad - 35 \left[\text{sign}(\ddot{s}_1) |\ddot{s}_1|^{\frac{3}{5}}, \text{sign}(\ddot{s}_2) |\ddot{s}_2|^{\frac{3}{5}} \right]^T - 10 \left[\text{sign}(s_1^{(3)}) |s_1^{(3)}|^{\frac{3}{4}}, \text{sign}(s_2^{(3)}) |s_2^{(3)}|^{\frac{3}{4}} \right]^T \\ \tau_{disc} &= -G \text{sign}(s^{(3)} + z_{aux}) \end{aligned}$$

Le terme τ_{nom} est obtenu en prenant $\rho = 4$, $k_1 = 24$, $k_2 = 50$, $k_3 = 35$, $k_4 = 10$ et $\nu = \frac{3}{4}$ (voir théorème 4.3). Le gain G , devant vérifier la relation (4.3.16) étendue au cas multi-entrée, c'est-à-dire :

$$G \geq \frac{\frac{2|\delta_v|}{C} \|\tau_{nom}\| + \|\vartheta\| + \eta}{1 - \frac{2|\delta_v|}{C}}$$

avec $\eta > 0$, est ajusté de manière à obtenir une convergence en temps fini des erreurs de suivi. Dans les simulations suivantes, on a pris $G = 0.2\|\tau_{nom}\| + 11$.

Résultats de simulation

Le paramètre β est égal à 1.2. On peut ainsi noter que pour la trajectoire circulaire de référence choisie, la valeur de la quantité $\beta u + \tau_u = 10\frac{\pi}{6}\sqrt{\beta^2 + (\frac{\pi}{6})^2} \approx 6.85 \neq 0$. Les sorties x , y , ψ , u , v et w sont mesurables par des capteurs appropriés. On dispose donc de tout l'état pour réaliser notre algorithme de commande robuste d'ordre 4 – 4. La figure 4.8(a) représente l'évolution des erreurs de poursuite en x et en y et montre la convergence en temps fini ($T_{eta} = 7s$) de s_1 et s_2 vers zéro malgré le démarrage de l'aéroglissoeur assez loin des trajectoires de référence (voir la figure 4.8(b)). La figure 4.8(c) montre l'évolution des vitesses d'avance, de glissement et de lacet. On peut remarquer que la réponse du système est celle voulue malgré la présence de perturbations sur le modèle.

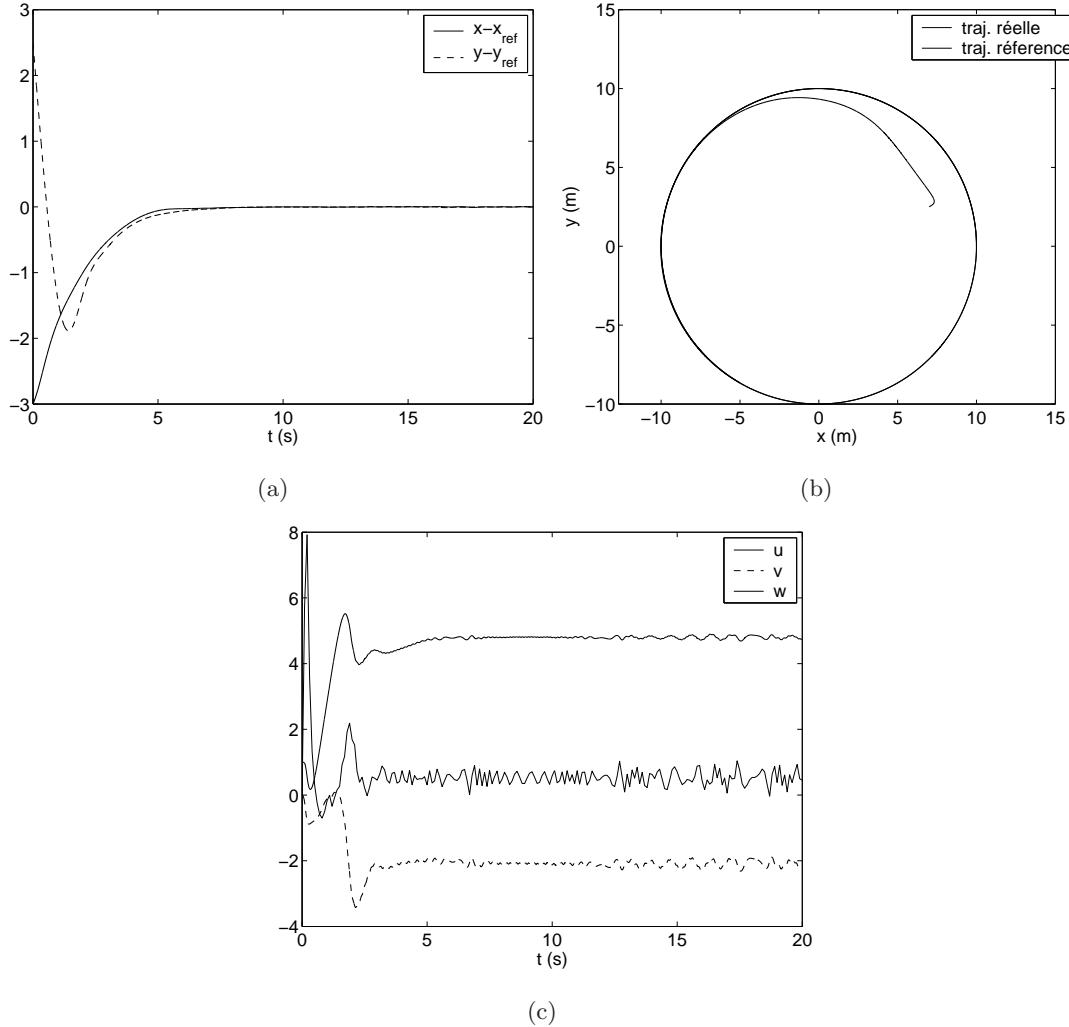


FIG. 4.8 – Stabilisation en temps fini des erreurs de suivi de trajectoire pour l'aéroglissoeur.

4.4.2 Commande d'un moteur pas-à-pas [Defoort et al., 2006d]

Introduction

Le moteur pas-à-pas est un actionneur électromécanique principalement utilisé pour le positionnement [Gieras et al., 2002]. Il est très fiable et requiert peu de maintenance puisque sans balai. Son aptitude à fournir de bonnes performances, sa petite taille ainsi que son faible coût, en font un actionneur apprécié dans de nombreuses applications telles que les imprimantes, les machines textiles, les machines-outils ou encore en robotique.

De nombreux travaux ont été développés pour la commande d'un moteur pas-à-pas. Parmi eux, on peut citer ceux basés sur les méthodes de :

- linéarisation entrée-sortie [Bodson et al., 1993, Zribi et Chiasson, 1993],
- perturbations singulières [Khalil et al., 1986],
- passivité associée à la platitude [Sira-Ramirez, 2000],
- commande backstepping [Xu et al., 1998],
- commande par modes glissants d'ordre 1 [Zribi et al., 2001, Lee et Shtessel, 1996],
- commande par modes glissants d'ordre 2 [Nollet et al., 2004, Nollet et al., 2006].

Ces méthodes présentent des facilités de mise en œuvre et de réglage mais pas toujours de bonnes qualités en termes de précision et de robustesse vis-à-vis d'un couple de charge ou de variations paramétriques (résistance, . . .).

L'objectif de cette partie est le développement et l'implantation sur le banc d'essai du LAGIS de l'Ecole Centrale de Lille d'une commande robuste multivariable par modes glissants d'ordre supérieur qui permette de garantir, sans nécessiter de phase d'initialisation et sans utiliser les données issues de la dynamo tachymétrique (car d'une part ces données sont fortement bruitées et d'autre part cela évite un capteur supplémentaire et coûteux) :

- une bonne qualité de l'asservissement : précision, rapidité, stabilité,
- de bonnes propriétés de robustesse vis-à-vis des variations paramétriques ainsi que du couple de charge,
- une facilité de mise en œuvre (facilité de réglage des paramètres de la loi de commande en vue d'atteindre les performances désirées).

Modèle du moteur pas-à-pas et objectif de commande

Le modèle d'état du moteur pas-à-pas, basé sur la transformation de Park et représenté dans le repère $d - q$, s'écrit de la manière suivante [Park, 1929] :

$$\begin{cases} \frac{di_d}{dt} = \left(\frac{1}{L} + \delta_1\right)v_d - \left(\frac{R}{L} + \delta_2\right)i_d + N\Omega i_q \\ \frac{di_q}{dt} = \left(\frac{1}{L} + \delta_1\right)v_q - \left(\frac{R}{L} + \delta_2\right)i_q - N\Omega i_d - \left(\frac{K}{J} + \delta_3\right)\Omega \\ \frac{d\Omega}{dt} = \left(\frac{K}{J} + \delta_4\right)i_q - \left(\frac{f_v}{J} + \delta_5\right)\Omega - \left(\frac{1}{J} + \delta_6\right)C_r \\ \frac{d\theta}{dt} = \Omega \end{cases} \quad (4.4.6)$$

où θ et Ω sont respectivement la position et la vitesse rotorique, i_d est le courant direct et i_q le courant en quadrature. N est le nombre de dents du rotor. J est le moment d'inertie, K est la constante de couple, f_v est le coefficient de frottements visqueux, R et L sont la résistance et l'inductance de chaque enroulement du moteur pas-à-pas. Les entrées v_d et v_q représentent respectivement la tension directe et la tension en quadrature.

Ici, deux types de perturbations sont pris en compte :

- le couple de charge C_r dont la valeur est inconnue,
- les incertitudes paramétriques (mauvaise connaissance ou variations, dues, par exemple, à l'échauffement) sur R , L , K , J et f_v . δ_i ($i = 1, \dots, 6$) représentent les différentes incertitudes paramétriques qui sont supposées avoir une dynamique d'évolution négligeable par rapport aux constantes de temps du système (i.e. $\dot{\delta}_i \approx 0$, $\forall i = 1, \dots, 6$), et être bornées.

Considérons dans un premier temps le modèle de référence du moteur pas-à-pas, sans couple de charge et sans incertitude, i.e.

$$\begin{cases} \frac{di_{d,ref}}{dt} = \frac{1}{L}v_{d,ref} - \frac{R}{L}i_{d,ref} + N\Omega i_{q,ref} \\ \frac{di_{q,ref}}{dt} = \frac{1}{L}v_{q,ref} - \frac{R}{L}i_{q,ref} - N\Omega i_{d,ref} - \frac{K}{L}\Omega_{ref} \\ \frac{d\Omega_{ref}}{dt} = \frac{K}{J}i_{q,ref} - \frac{f_v}{J}\Omega_{ref} \\ \frac{d\theta_{ref}}{dt} = \Omega_{ref} \end{cases} \quad (4.4.7)$$

En prenant comme sortie $[i_{d,ref}, \theta_{ref}]^T$, le système (4.4.7) est plat. En effet, on a :

$$\begin{cases} \Omega_{ref} = \frac{d\theta_{ref}}{dt} \\ i_{q,ref} = \frac{1}{K}(J\frac{d^2\theta_{ref}}{dt^2} + f_v\frac{d\theta_{ref}}{dt}) \\ v_{d,ref} = L\frac{di_{d,ref}}{dt} + Ri_{d,ref} - \frac{NL}{K}\frac{d\theta_{ref}}{dt}(J\frac{d^2\theta_{ref}}{dt^2} + f_v\frac{d\theta_{ref}}{dt}) \\ v_{q,ref} = \frac{JL}{K}\frac{d^3\theta_{ref}}{dt^3} + \frac{1}{K}(Lf_v + RJ)\frac{d^2\theta_{ref}}{dt^2} + \left(\frac{Rf_v}{K} + K + NLi_{d,ref}\right)\frac{d\theta_{ref}}{dt} \end{cases}$$

Le fait que le modèle du moteur soit un système plat est intéressant pour plusieurs raisons. La propriété de platitude facilite considérablement la planification de trajectoires. En imposant les trajectoires de référence θ_{ref} et $i_{d,ref}$, on obtient aisément les trajectoires de référence de l'ensemble des variables qui vérifient le système (4.4.7). D'autre part, un système plat est exactement linéarisable par bouclage dit

endogène, c'est-à-dire engendré par les variables du système et leurs dérivées.

L'objectif principal de commande est la poursuite d'une trajectoire de référence en position $\theta_{ref}(t)$ malgré la présence de perturbations. Un second objectif est ici ajouté. Il consiste à forcer le courant i_d à zéro de manière à minimiser les pertes Joules.

Commande par modes glissants d'ordre 2 – 3

En choisissant comme variables de glissement $s_1 = i_d - i_{d,ref}$ et $s_2 = \theta - \theta_{ref}$, le but de la commande est d'annuler en temps fini $s = [s_1, s_2]^T$. Puisque le système (4.4.6) admet un degré relatif égal à [1, 3] par rapport à s , en dérivant une fois s_1 et trois fois s_2 , on obtient :

$$\begin{cases} \dot{s}_1 &= \left(\frac{1}{L} + \delta_1\right)\bar{v}_d + \delta_{\phi_1}(e) + \bar{\phi}_1(e) \\ s_2^{(3)} &= \left(\frac{K}{JL} + \frac{\delta_4}{L} + \frac{K\delta_1}{J} + \delta_1\delta_4\right)\bar{v}_q + \delta_{\phi_2}(e) + \bar{\phi}_2(e) \end{cases} \quad (4.4.8)$$

avec

$$\begin{aligned} e &= [e_1, e_2, e_3, e_4]^T = [i_d - i_{d,ref}, i_q - i_{q,ref}, \Omega - \Omega_{ref}, \theta - \theta_{ref}]^T \\ \bar{v}_d &= v_d - v_{d,ref} \\ \bar{v}_q &= v_q - v_{q,ref} \\ \bar{\phi}_1(e) &= \frac{1}{L}(-Re_1 + NL(e_3e_2 + e_3i_{q,ref} + e_2\Omega_{ref})) \\ \bar{\phi}_2(e) &= -\frac{K}{JL}(Re_2 + NL(e_3e_1 + e_3i_{d,ref} + e_1\Omega_{ref}) + Ke_3) - \frac{f_v}{J^2}(Ke_2 - f_ve_3) \\ \delta_{\phi_1}(e) &= \delta_1v_{d,ref} - \delta_2(e_1 + i_{d,ref}) \\ \delta_{\phi_2}(e) &= -\delta_4\left(\frac{R}{L}e_2 + N(e_3e_1 + e_3i_{d,ref} + e_1\Omega_{ref}) + \frac{K}{L}e_3 - \frac{di_{q,ref}}{dt}\right) \\ &\quad -\delta_5\left(\frac{K}{J}e_2 - \frac{f_v}{J}e_3 - \frac{1}{J}Cr + \frac{d\Omega_{ref}}{dt}\right) - \left(\frac{f_v}{J} + \delta_5\right)(\delta_4(e_2 + i_{q,ref}) - \delta_5(e_3 + \Omega_{ref}) - \delta_6Cr) \\ &\quad -\left(\frac{K}{J} + \delta_4\right)(\delta_3(e_3 + \Omega_{ref}) + \delta_2(e_2 + i_{q,ref}) - \delta_1v_{q,ref}) - \delta_6\dot{C}_r + \frac{f_v}{J^2}Cr - \frac{1}{J}\dot{C}_r \end{aligned}$$

En utilisant le retour d'état :

$$\begin{cases} \bar{v}_d &= L(v - \bar{\phi}_1(e)) \\ \bar{v}_q &= \frac{JL}{K}(u - \bar{\phi}_2(e)) \end{cases}$$

où $[v, u]^T \in \mathbb{R}^2$ est la nouvelle commande, le système (4.4.8) devient :

$$\dot{s}_1 = (1 + \zeta_1)v + \vartheta_1(e) \quad (4.4.9)$$

$$s_2^{(3)} = (1 + \zeta_2)u + \vartheta_2(e) \quad (4.4.10)$$

où $\vartheta_1, \vartheta_2, \zeta_1$ et ζ_2 sont définies par :

$$\begin{cases} \zeta_1 &= L\delta_1 \\ \zeta_2 &= \frac{J}{K}\delta_4 + L\delta_1 + \frac{JL}{K}\delta_1\delta_4 \\ \vartheta_1(e) &= \delta_{\phi_1}(e) - \zeta_1\bar{\phi}_1(e) \\ \vartheta_2(e) &= \delta_{\phi_2}(e) - \zeta_2\bar{\phi}_2(e) \end{cases}$$

Par conséquent, il est possible de calculer indépendamment chacune des commandes v et u . Avant de les synthétiser, il est nécessaire de vérifier l'hypothèse suivante :

Hypothèse 4.7 Les fonctions incertaines $\dot{\vartheta}_1(e)$, $\vartheta_2(e)$, ζ_1 et ζ_2 sont bornées. Plus particulièrement,

- les incertitudes paramétriques δ_i sont bornées par des constantes $\delta_{i,max}$ positives connues telles que :

$$\begin{cases} |\delta_{1,max}| & < \frac{1}{L} \\ \left| \frac{J}{K} \delta_{4,max} + L \delta_{1,max} + \frac{JL}{K} \delta_{1,max} \delta_{4,max} \right| & < 1 - b, \quad 0 < b < 1 \end{cases}$$

- les trajectoires de référence $i_{q,ref}$, $i_{d,ref}$, θ_{ref} , Ω_{ref} , $i_{d,ref}$, $v_{d,ref}$, $v_{q,ref}$ ainsi que leur dérivée première par rapport au temps sont bornées par des constantes positives connues,
- le couple de charge ainsi que sa dérivée temporelle sont bornés.

Synthèse de la commande v

Puisque le système (4.4.6) a un degré relatif égal à un par rapport à s_1 , il est possible d'appliquer une commande par modes glissants d'ordre deux, afin d'éviter le phénomène de réticence, en utilisant par exemple l'algorithme du “super-twisting” [Levant, 1993] (voir Section 4.2.3). La loi de commande s'écrit [Levant, 1993] :

$$v = v_1 + v_2 \quad \text{avec} \quad \begin{cases} \dot{v}_1 = -\alpha \operatorname{sign}(s_1) \\ v_2 = -\lambda |s_1|^{\frac{1}{2}} \operatorname{sign}(s_1) \end{cases} \quad (4.4.11)$$

Les conditions suffisantes de convergence en temps fini sur l'ensemble de glissement $\{s_1 = \dot{s}_1 = 0\}$ sont données par :

$$\alpha > \frac{C_0}{1 - L\delta_{1,max}}, \quad \lambda > 0 \quad \text{et} \quad \lambda^2 \geq \frac{4C_0}{(1 - L\delta_{1,max})^2} \frac{(1 + L\delta_{1,max})\alpha + C_0}{(1 - L\delta_{1,max})\alpha + C_0}$$

où C_0 est une constante positive telle que dans un voisinage $|s_1(e, t)| < S_0$ ($S_0 > 0$), l'inégalité $|\dot{\vartheta}_1(e)| \leq C_0$ soit vérifiée.

L'application de la commande (4.4.11) permet la stabilisation en temps fini de l'erreur e_1 vers zéro en évitant le phénomène de réticence.

Synthèse de la commande u

Pour stabiliser en temps fini la variable de glissement s_2 , réécrivons le système (4.4.10) sous la forme :

$$\begin{cases} \dot{z}_1 = z_2 \\ \dot{z}_2 = z_3 \\ \dot{z}_3 = (1 + \zeta_2)u + \vartheta_2(e) \end{cases} \quad (4.4.12)$$

avec

$$z = [z_1, z_2, z_3]^T, \quad z_1 = e_4, \quad z_2 = \dot{e}_4, \quad z_3 = \ddot{e}_4$$

Puisque l'hypothèse 4.7 est vérifiée, il existe une fonction définie positive $a(z)$ et une constante positive $0 < b < 1$ telle que :

$$\begin{cases} |\vartheta_2(e)| \leq a(z) \\ |\zeta_2| \leq 1 - b \end{cases}$$

En utilisant les résultats présentés dans le théorème 4.4, la loi de commande u s'écrit :

$$\begin{cases} u &= u_{nom}(z) + u_{disc}(z, z_{aux}) \\ \dot{z}_{aux} &= -u_{nom}(z) \end{cases} \quad (4.4.13)$$

avec

$$u_{nom}(z) = -20 \operatorname{sign}(z_1) |z_1|^{\frac{1}{2}} - 10 \operatorname{sign}(z_2) |z_2|^{\frac{3}{5}} - 4 \operatorname{sign}(z_3) |z_3|^{\frac{3}{4}}$$

et

$$u_{disc}(z, z_{aux}) = -G(z) \operatorname{sign}(z_3 + z_{aux}) \quad (4.4.14)$$

Le gain $G(z)$ vérifie la relation (4.3.16), c'est-à-dire :

$$G(z) = \frac{(1-b)|u_{nom}(z)| + a(z) + \eta}{b}, \quad \eta > 0$$

Observateur de vitesse et d'accélération rotorique

Motivé par le fait que la mesure de la position effectuée par un codeur optique est bien plus fiable que la mesure bruitée de la vitesse fournie par une génératrice tachymétrique, il s'avère judicieux de n'utiliser que la mesure de position. De plus, cela permet de faire l'économie d'un capteur. Or, la loi de commande (4.4.13) nécessite l'information de la vitesse et de l'accélération (i.e. z_2 et z_3). Afin de n'utiliser que les mesures des courants et de la position, il est alors choisi d'implanter un observateur robuste basé sur l'algorithme de "twisting réel" [Floquet et al., 2003].

Le vecteur des estimées est $z_{est} = [z_{1,est}, z_{2,est}, z_{3,est}]^T = [\theta_{est} - \theta_{ref}, \Omega_{est} - \Omega_{ref}, a_{est} - \dot{\Omega}_{ref}]^T$ et les erreurs d'estimation sont $\varepsilon_\theta = \theta - \theta_{est}$ et $\varepsilon_\Omega = \Omega - \Omega_{est}$. L'observateur est défini comme suit :

$$\begin{cases} \dot{\Omega}_{est} = \frac{1}{J}(Ki_q - f_v\Omega_{est}) - \chi \\ \dot{\theta}_{est} = \Omega_{est} \end{cases} \quad (4.4.15)$$

où χ est une injection de sortie discontinue. La surface de glissement est $\{\varepsilon_\theta = 0\}$ et les dérivées successives sont :

$$\dot{\varepsilon}_\theta = \dot{\theta} - \dot{\theta}_{est} = \Omega - \Omega_{est} = \varepsilon_\Omega \quad (4.4.16)$$

$$\ddot{\varepsilon}_\theta = \dot{\Omega} - \dot{\Omega}_{est} = -\frac{f_v}{J}\varepsilon_\Omega + \delta_4 i_q - \delta_5 \Omega - \left(\frac{1}{J} + \delta_6\right) C_r + \chi \quad (4.4.17)$$

En appliquant l'algorithme du "twisting réel" [Floquet et al., 2003] :

$$\chi = \begin{cases} -\beta\varepsilon_\theta - \lambda_M \operatorname{sign}(\varepsilon_\theta) & \text{si } \varepsilon_\theta \Delta \varepsilon_\theta > 0 \\ -\beta\varepsilon_\theta - \lambda_m \operatorname{sign}(\varepsilon_\theta) & \text{si } \varepsilon_\theta \Delta \varepsilon_\theta \leq 0 \end{cases}$$

avec

$$\begin{cases} \lambda_m > |\delta_4 i_q - \delta_5 \Omega - (\frac{1}{J} + \delta_6) C_r|_{\max} \\ \lambda_M > \lambda_m + 2 |\delta_4 i_q - \delta_5 \Omega - (\frac{1}{J} + \delta_6) C_r|_{\max} \end{cases}$$

et

$$\Delta\varepsilon_\theta = \begin{cases} 0, & k = 0 \\ \varepsilon_\theta(k\tau) - \varepsilon_\theta((k-1)\tau), & k \geq 1 \end{cases}$$

où β est un scalaire positif et τ est la période d'échantillonnage, il peut être montré qu'un régime glissant d'ordre deux par rapport à ε_θ est établi en un temps fini (voir [Fridman et Levant, 2002]). Alors, Ω_{est} converge en temps fini vers Ω . En outre, lorsque le régime glissant est établi, la commande équivalente de χ , notée χ^{eq} et obtenue par la relation $\ddot{\varepsilon}_\theta = 0$, est définie par l'équation :

$$\chi^{eq} = -\delta_4 i_q + \delta_5 \Omega + \left(\frac{1}{J} + \delta_6 \right) C_r \quad (4.4.18)$$

Par conséquent, l'accélération du rotor $\dot{\Omega}$ sur la surface de glissement $\{\varepsilon_\theta = \dot{\varepsilon}_\theta = 0\}$, obtenue en remplaçant χ par la commande équivalente associée χ^{eq} , est décrite par :

$$\dot{\Omega} = \frac{1}{J}(Ki_q - f_v\Omega) = a_{est}$$

Après un temps fini, on obtient l'estimation en ligne de l'accélération a_{est} .

L'observateur proposé fournit donc une estimation rapide et précise de Ω et de $\dot{\Omega}$.

Résultats expérimentaux

Les expérimentations sont réalisées sur le banc d'essai développé au LAGIS (figure 4.9). Il se compose de :

- un moteur pas-à-pas de type Turbo Disc P850 dont les caractéristiques nominales, obtenues après identification, sont :

• Moment d'inertie :	$J = 5 \times 10^{-4} kg.m^2$
• Constante de couple :	$K = 0.4 Nm/A$
• Nombre de dents :	$N = 50$
• Résistance :	$R = 3\Omega$
• Inductance :	$L = 8mH$
• Coefficient de frottements visqueux :	$f_v = 1.8 \times 10^{-2} Nm.s/rad$

- un codeur optique absolu dont la précision est de $7.67 \cdot 10^{-4} rad$ qui mesure la position rotorique,
- un frein à poudre qui génère la charge,
- un ordinateur et une carte dSPACE 1104,
- une électronique de puissance dont la puissance maximale est $30V - 3A$,
- des transducteurs de courant d'une précision de $0.03A$ pour calculer les courants i_d et i_q .

Pour les expérimentations, la période d'échantillonnage est $\tau = 10^{-4}s$. Les différents paramètres de la loi de commande, choisis de façon à obtenir de très bonnes performances statiques et dynamiques, sont les suivants : $\alpha = 1$, $\lambda = 1000$ (pour la synthèse de v), $b = 0.9$, $a(z) = 1.5$, $\eta = 0.1$ (pour la synthèse de u), $\beta = 100$, $\lambda_m = 3500$ et $\lambda_M = 5000$ (pour la synthèse de l'observateur).

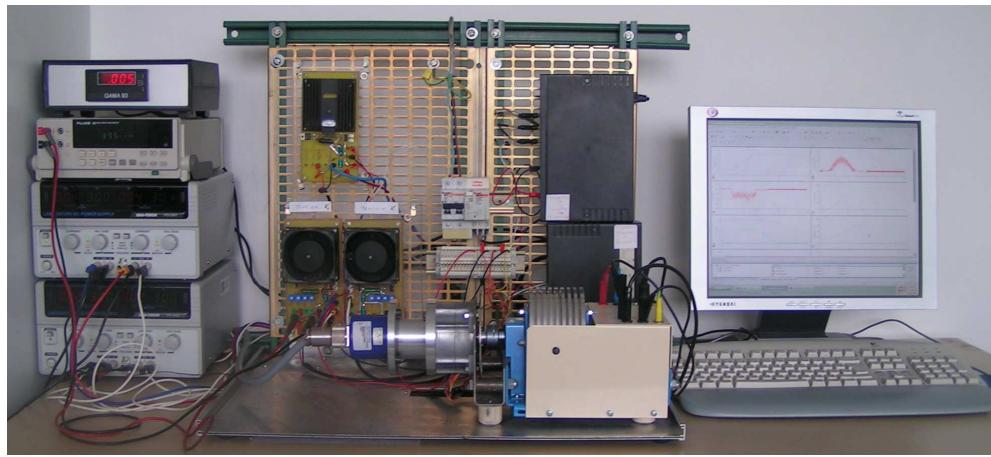


FIG. 4.9 – Plate-forme d'essais du moteur pas-à-pas.

L'objectif de commande est de déplacer le moteur pas-à-pas de $\theta_{initial} = 0\text{rad}$ à $\theta_{final} = 6\text{rad}$ en suivant une trajectoire de référence $\theta_{ref}(t)$. Afin de ne pas avoir de discontinuité et d'à-coups en vitesse et en accélération pour éviter des pics d'énergie électrique, la trajectoire de référence choisie est :

$$\theta_{ref}(t) = \theta_{initial} + (\theta_{final} - \theta_{initial})(10\Delta_t^3 - 15\Delta_t^4 + 6\Delta_t^5) \quad (4.4.19)$$

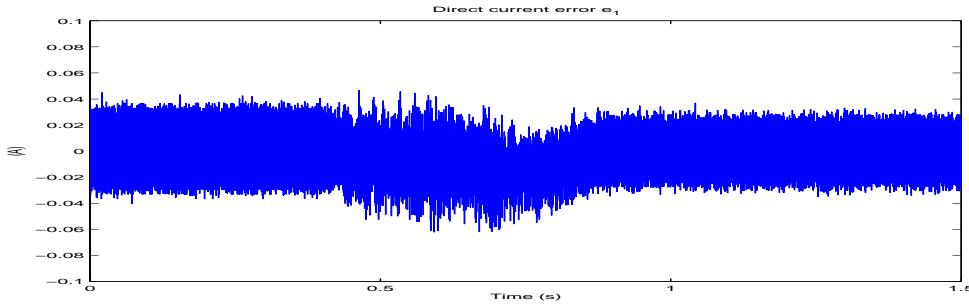
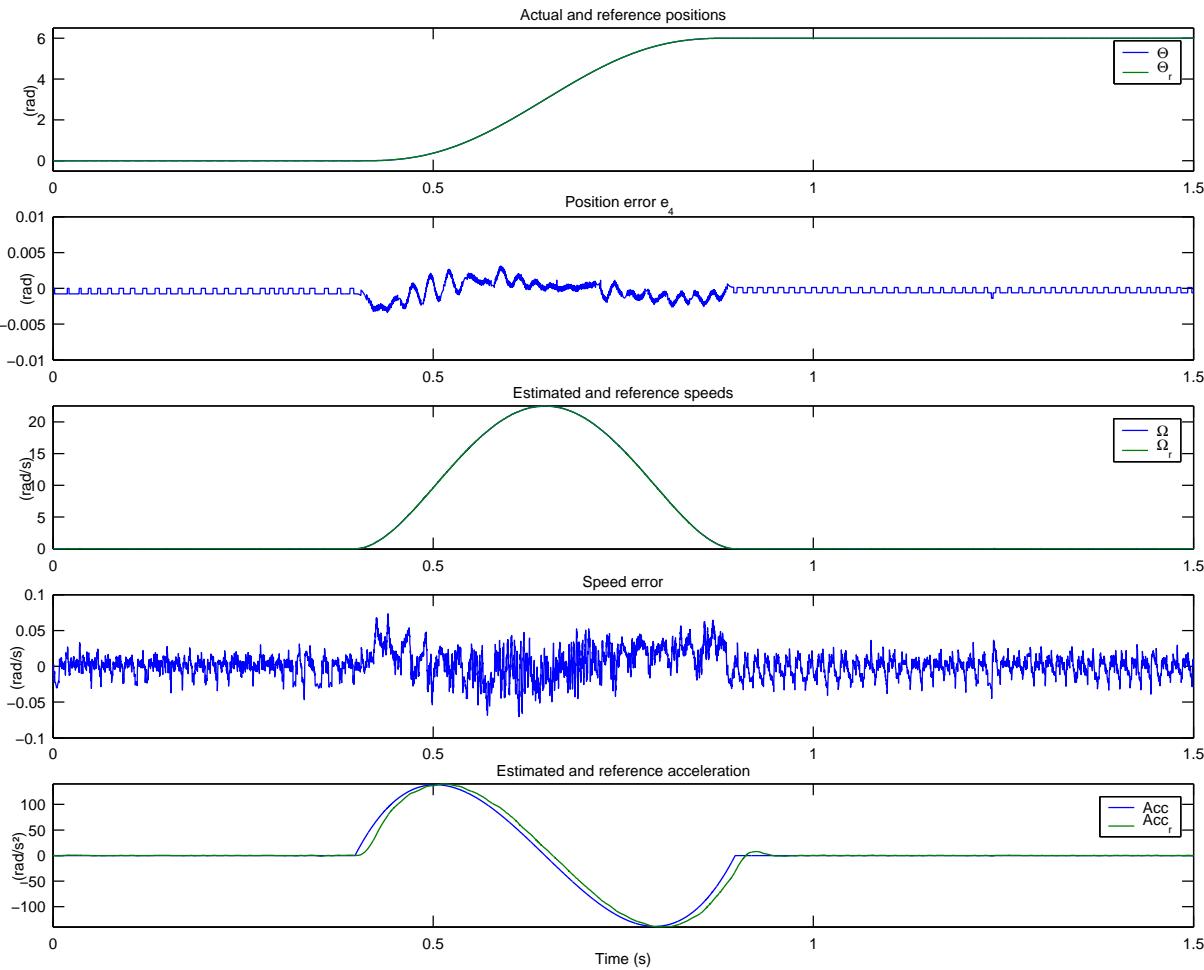
avec

$$\begin{aligned} \Delta_t &= \frac{(t-t_{initial})}{(t_{final}-t_{initial})}, & t_{initial} \text{ est l'instant initial et } t_{final} = t_{initial} + 0.5s \\ \theta_{ref}(t_{initial}) &= \theta_{initial}, & \theta_{ref}(t_{final}) = \theta_{final} \\ \dot{\theta}_{ref}(t_{initial}) &= 0, & \dot{\theta}_{ref}(t_{final}) = 0 \\ \ddot{\theta}_{ref}(t_{initial}) &= 0, & \ddot{\theta}_{ref}(t_{final}) = 0 \end{aligned}$$

Afin de minimiser les pertes Joules, le courant direct de référence est $i_{d,ref}(t) = 0$ [Bodson et al., 1993].

Les figures 4.10-4.11 présentent les résultats expérimentaux en l'absence de perturbations et d'incertitudes. La figure 4.10 montre que l'évolution de l'erreur en courant est de l'ordre de $0.03A$ qui est la précision des transducteurs de courant. Quant à la figure 4.11, elle montre que l'erreur de poursuite en position ne dépasse pas 4.10^{-3}rad . On peut également voir qu'en régime permanent les oscillations correspondent à la précision du codeur optique (7.10^{-4}rad). La position oscille entre deux valeurs 6.00013rad et 5.99936rad délivrées par codeur optique et encadrant la position permanente souhaitée de 6rad . L'observateur de vitesse donne de très bons résultats (erreur de 0.05rad/s). Sur la figure 4.11, on visualise une faible erreur d'estimation sur la vitesse et l'accélération.

Afin de tester les propriétés de robustesse de la loi de commande, des tests ont été effectués en considérant une variation de la résistance de 25% (simulant par exemple un échauffement) ainsi qu'en appliquant un couple de charge $C_r = 0.55\text{Nm}$ avec le frein à poudre. Les figures 4.12-4.14 présentent les résultats expérimentaux en présence du couple de charge et de variations paramétriques. La figure 4.12 montre que l'évolution de l'erreur en courant reste faible. La figure 4.13 montre que l'erreur de

FIG. 4.10 – Erreur en courant - $C_r = 0Nm$.FIG. 4.11 – Erreur en position angulaire du rotor - Observateur de vitesse et d'accélération rotorique - $C_r = 0Nm$.

poursuite en position ne dépasse pas $0.02rad$. Enfin, la figure 4.13 représente la tension et le courant appliqués sur l'une des phases du moteur.

Il convient de noter que ces résultats sont meilleurs que ceux obtenus sur la même plate-forme par une commande par modes glissants d'ordre deux basée sur le “twisting échantillonné” [Nollet et al., 2006] notamment au niveau de la consommation énergétique.

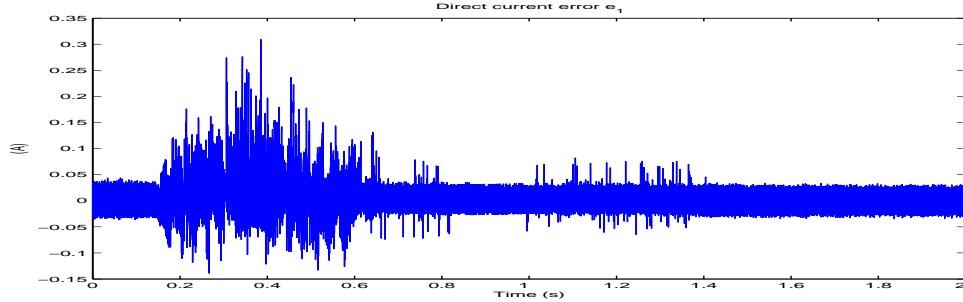


FIG. 4.12 – Erreur en courant - Incertitudes paramétriques - $C_r \neq 0$.

4.5 Conclusion

La première partie de ce chapitre a été consacrée à une présentation générale des concepts de base de la commande par modes glissants (conditions d'existence, phénomène de réticence, propriétés de robustesse, ...). Après avoir rappelé les principales raisons ayant motivé l'apparition de la commande par modes glissants d'ordre supérieur, nous avons présenté son principe. Cette approche contourne les principaux problèmes rencontrés avec la commande par modes glissants d'ordre un tout en préservant les propriétés de robustesse et de convergence en temps fini. Elle permet également d'augmenter la précision. Bien qu'il existe de nombreux algorithmes permettant de construire des commandes par modes glissants d'ordre deux, on en trouve peu capable de générer des modes glissants d'ordre quelconque.

La seconde partie de ce chapitre a été vouée à la construction d'un algorithme *original* de commande par modes glissants d'ordre quelconque qui ne nécessite pas de phase d'initialisation et qui garantisse un réglage simple et rapide des paramètres de la loi de commande. Cet algorithme, tirant son inspiration de la commande par modes glissants avec action intégrale [Utkin et Shi, 1996], consiste à ajouter un terme discontinu afin de rendre robuste vis-à-vis des perturbations une commande nominale basée sur des fonctions homogènes.

La troisième partie de ce chapitre a été destinée à la mise en application de notre algorithme de commande par modes glissants d'ordre supérieur pour la commande d'un aéroglisseur ainsi que d'un moteur pas-à-pas. Cette commande a été réglée de manière à être robuste vis-à-vis des perturbations de type vague pour l'aéroglisseur. Quant au moteur, elle a été réglée de manière à rejeter l'effet du couple de charge et des incertitudes paramétriques. Les différentes expérimentations sur un banc d'essais ont montré qu'elle conduit à d'excellents résultats, même lors de variations paramétriques

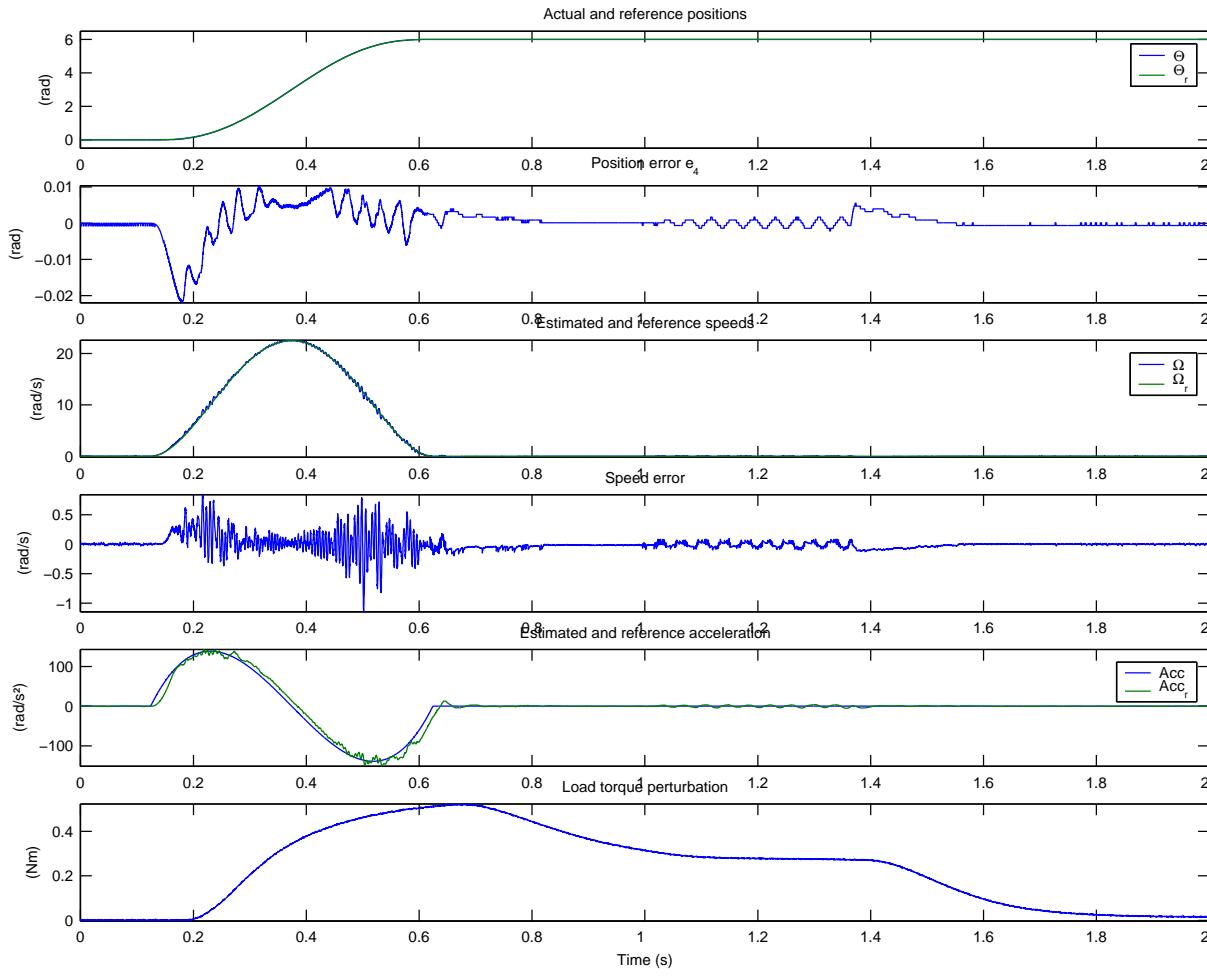


FIG. 4.13 – Erreur en position angulaire du rotor - Observateur de vitesse et d'accélération rotorique
- Incertitudes paramétriques - $C_r \neq 0$.

importantes.

Ceci achève la présentation des techniques robustes de commande par modes glissants sur lesquelles nous avons basé notre travail. Dans le chapitre suivant, deux autres commandes, facilement implémentables, tirant toujours leur inspiration du principe de la commande par modes glissants avec action intégrale, seront appliquées à des robots mobiles de type unicycle (sans et avec prise en compte des dynamiques des actionneurs), avec un objectif à la fois de stabilisation et de poursuite de trajectoire. Bien que l'objectif original du principe de la CMGI soit d'éliminer l'effet des perturbations dès l'instant initial [Utkin et Shi, 1996], deux autres finalités seront mises en évidence dans le chapitre suivant.

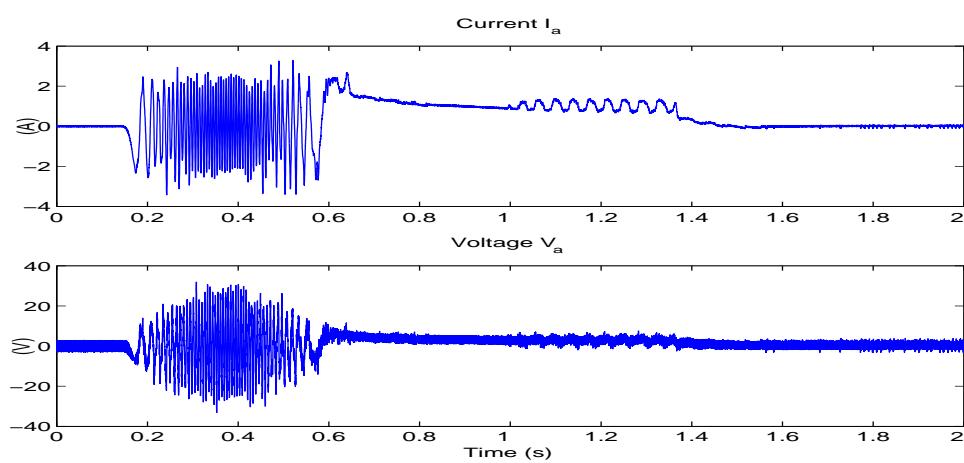


FIG. 4.14 – Tension et courant appliqués sur l'une des phases du moteur - Incertitudes paramétriques
- $C_r \neq 0$.

Chapitre 5

Stabilisation et suivi de trajectoire pour un robot

5.1 Introduction

Dans la première partie de ce mémoire, des algorithmes de planification permettant de générer les trajectoires et les commandes de référence exécutables par les robots mobiles, ont été développés. Ces algorithmes, applicables en temps réel, n'ont cependant pas été conçus dans le but de contrecarrer l'effet des imperfections (erreurs de modélisation et de mesure) et des perturbations. En effet, bien que les résultats obtenus en simulation sur un modèle sans perturbation soient satisfaisants, l'application directe des commandes de référence sur le robot Pekee (figure 5.1), dont les caractéristiques et le fonctionnement sont donnés dans l'Annexe A, s'est avérée décevante (voir la figure 5.2). Comme le montre la figure 5.2(b), l'erreur de poursuite $\sqrt{(x - x_{ref})^2 + (y - y_{ref})^2}$, entre la trajectoire de référence planifiée et la trajectoire réelle effectuée par le robot Pekee, diverge. Par conséquent, il est



FIG. 5.1 – Le robot Pekee.

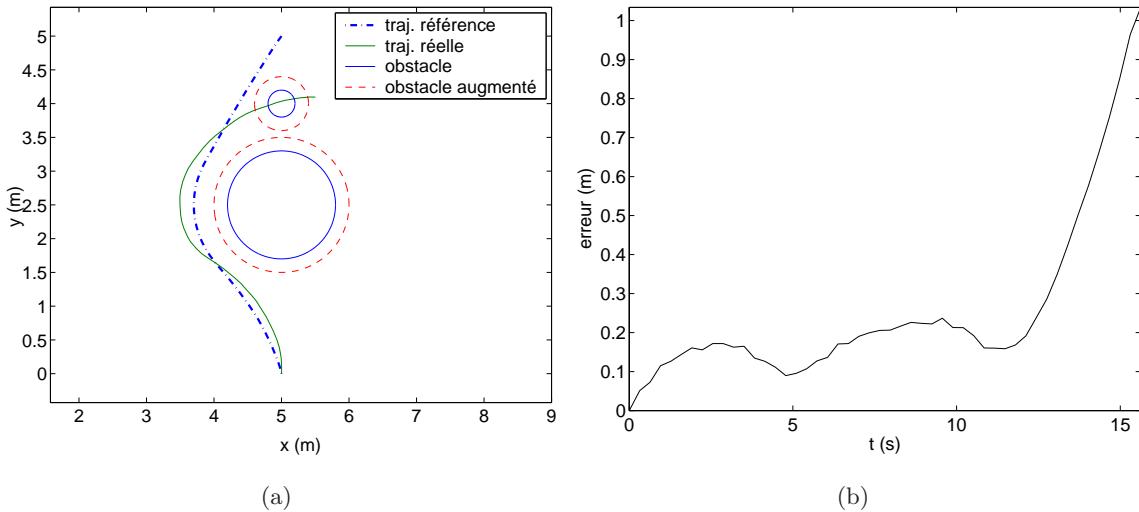


FIG. 5.2 – Application directe de la commande planifiée sur le robot Pekee. (a) Trajectoires planifiée et réelle. (b) Erreur de suivi.

indispensable de corriger les commandes planifiées en boucle ouverte en fonction de la position et de l’orientation du robot par rapport à la trajectoire planifiée.

De nombreuses méthodes pour la synthèse de commandes permettant de résoudre le problème de stabilisation asymptotique pour un système non holonome ont été développées. Cependant, l’étude de ces méthodes a mis en évidence un certain nombre de limitations (phénomène de réticence, singularités, convergence lente, problèmes de robustesse vis-à-vis de dynamiques non modélisées, …). Cela motive le développement d’une synthèse générique de commandes robustes évitant ces limitations.

Sur la base de méthodes originales, nous avons synthétisé et implémenté plusieurs lois de commande, garantissant la stabilisation et/ou le suivi de trajectoire, robustes par rapport aux incertitudes et aux perturbations en considérant dans un premier temps le modèle cinématique du robot [Defoort et al., 2007d] et dans un second temps son modèle dynamique [Defoort et al., 2005], [Defoort et al., 2006c], [Defoort et al., 2007b], [Defoort et al., 2007c]. Dans ce chapitre, deux commandes, tirant leur inspiration du principe de la commande par modes glissants avec action intégrale (CMGI), sont présentées.

La première consiste à ajouter un terme discontinu, définissant une structure variable de commande par modes glissants, à une loi de commande continue qui stabilise les erreurs de suivi de trajectoire en l’absence de perturbations [Defoort et al., 2007d, Defoort et al., 2005, Defoort et al., 2006c]. Cette méthodologie permet de mettre en lumière l’un des rôles de la CMGI qui consiste en l’amélioration des résultats obtenus à partir d’une commande nominale.

Dans la seconde méthode, le modèle du robot mobile doit être mis, par l’intermédiaire de changements de coordonnées locaux de l’espace d’état, sous une forme particulière, appelée **système de**

Heisenberg ou *intégrateur nonholonome* :

$$\begin{cases} \dot{Z} &= X_1 U_2 - X_2 U_1 \\ \dot{X}_1 &= U_1 \\ \dot{X}_2 &= U_2 \end{cases} \quad (5.1.1)$$

où $U = [U_1, U_2]^T \in \mathbb{R}^2$ est la commande. De nombreux travaux ont permis de stabiliser cette classe de systèmes non holonomes [Khennouf et de Wit, 1995, Bloch et Drakunov, 1996, Bloch et al., 2000, Marchand et Alamir, 2003, Prieur et Astolfi, 2003]. Cependant, une majorité d'entre-eux ne considère pas la présence de perturbations quelconques dans le modèle (5.1.1). Bien qu'une récente contribution dans [Drakunov et al., 2005] ait permis une stabilisation pratique du système (5.1.1) perturbé, de grands gains sont nécessaires au niveau de la construction de la surface de glissement. Ceci pose des problèmes au niveau de la détermination des gains de la commande. Dans notre travail, une autre formulation de la surface de glissement est introduite. Puis, deux commandes discontinues basées sur le principe de la CMGI et tenant compte des dynamiques des actionneurs, sont développées afin de garantir une stabilisation pratique des erreurs de suivi du robot malgré la présence de perturbations dans le modèle (5.1.1). La première loi est synthétisée de manière à rejeter l'effet des perturbations vérifiant la condition de recouvrement. Puis, en généralisant le concept de la CMGI, une nouvelle loi de commande par modes glissants d'ordre deux rend les trajectoires du système (5.1.1) insensibles vis-à-vis d'une plus large classe de perturbations. Cette méthodologie nous permet, enfin, de mettre en lumière un autre avantage de la CMGI. Par un choix approprié de la surface de glissement, il est possible d'éliminer les singularités dans la commande nominale, ce qui est difficilement réalisable avec une commande par modes glissants classiques du fait de la phase de convergence pendant laquelle les trajectoires du système sont sensibles vis-à-vis des perturbations.

Pour chacune de ces méthodes, nous présentons des résultats expérimentaux corroborant nos résultats théoriques.

Remarque 5.1 *Dans ce chapitre, puisque les commandes en boucle fermée sont construites de manière similaire pour chaque robot, l'indice i est omis afin de simplifier les notations. Nous appliquons, ici, au robot de type unicycle, l'approche de commande par modes glissants avec action intégrale afin de suivre la trajectoire de référence $q_{ref}(t) = [x_{ref}(t), y_{ref}(t), \theta_{ref}(t)]^T$. Cette trajectoire, planifiée par les algorithmes décrits dans la première partie, est admissible. C'est-à-dire qu'elle est solution du système d'équations différentielles :*

$$\begin{bmatrix} \dot{x}_{ref}(t) \\ \dot{y}_{ref}(t) \\ \dot{\theta}_{ref}(t) \end{bmatrix} = \begin{bmatrix} \cos \theta_{ref}(t) & 0 \\ \sin \theta_{ref}(t) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_{ref}(t) \\ w_{ref}(t) \end{bmatrix} \quad (5.1.2)$$

où $u_{ref}(t) = [v_{ref}(t), w_{ref}(t)]^T$ est la commande de référence planifiée.

5.2 Poursuite de trajectoires non stationnaires

5.2.1 Formulation du problème

Pour chaque robot, l'objectif est la synthèse d'une commande en boucle fermée $u(t)$ assurant le suivi de la trajectoire de référence $q_{ref}(t)$ malgré la présence d'imperfections (erreurs de modélisation et de mesure, variations paramétriques) et de perturbations. Ainsi, les équations cinématiques du robot unicycle réel peuvent s'écrire sous la forme :

$$\begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \dot{\theta}(t) \end{bmatrix} = \begin{bmatrix} \cos \theta(t) & 0 \\ \sin \theta(t) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v(t) \\ w(t) \end{bmatrix} + p(q, t) \quad (5.2.1)$$

où $q(t) = [x(t), y(t), \theta(t)]^T$ est l'état réel du robot, $u(t) = [v(t), w(t)]^T$ est la commande en boucle fermée et $p(q, t)$ représente les incertitudes et les perturbations.

Hypothèse 5.1 La fonction incertaine $p(q, t)$ vérifie la condition de recouvrement, i.e.

$$p(q, t) \in \text{vect} \left\{ [\cos \theta(t), \sin \theta(t), 0]^T, [0, 0, 1]^T \right\} \quad (5.2.2)$$

C'est-à-dire que l'on peut écrire $p(q, t) = \delta_v(q, t) [\cos \theta(t), \sin \theta(t), 0]^T + \delta_w(q, t) [0, 0, 1]^T$ avec $\delta_v(q, t) \in \mathbb{R}$ et $\delta_w(q, t) \in \mathbb{R}$.

Le problème de poursuite de trajectoire consiste à déterminer une commande permettant de stabiliser asymptotiquement l'erreur :

$$[e_x(t), e_y(t), e_\theta(t)]^T = [x(t) - x_{ref}(t), y(t) - y_{ref}(t), \theta(t) - \theta_{ref}(t)]^T \quad (5.2.3)$$

Pour résoudre ce problème, appliquons le changement de coordonnées suivant [Kanayama et al., 1990] :

$$\begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} = \begin{bmatrix} -\cos \theta & -\sin \theta & 0 \\ \sin \theta & -\cos \theta & 0 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} e_x \\ e_y \\ e_\theta \end{bmatrix}$$

qui préserve l'origine et donne comme dynamique :

$$\dot{e} = g_1(e, t) + g_2(e)(u + \delta(q, t)) \quad (5.2.4)$$

où

$$\begin{cases} e &= [e_1, e_2, e_3]^T \\ g_1(e, t) &= [v_{ref} \cos e_3, v_{ref} \sin e_3, w_{ref}]^T \\ g_2(e) &= \begin{bmatrix} -1 & e_2 \\ 0 & -e_1 \\ 0 & -1 \end{bmatrix} \\ \delta(q, t) &= [\delta_v(q, t), \delta_w(q, t)]^T \end{cases}$$

Dans le cadre de cette étude, deux hypothèses doivent être vérifiées.

Hypothèse 5.2 Les fonctions incertaines $\delta_v(q, t)$ et $\delta_w(q, t)$ sont bornées. Plus particulièrement, il existe deux fonctions positives connues $a_v(q)$ et $a_w(q)$ telles que :

$$\begin{cases} |\delta_v(q, t)| \leq a_v(q) \\ |\delta_w(q, t)| \leq a_w(q) \end{cases} \quad (5.2.5)$$

pour tout $t \geq 0$.

Hypothèse 5.3 Les vitesses de référence v_{ref} et w_{ref} sont uniformément continues et bornées. On suppose également que l'on se trouve dans le cadre du suivi de trajectoire non stationnaire qui ne présente pas de points d'arrêt. Ceci implique que, soit la vitesse linéaire $v_{ref}(t)$, soit la vitesse angulaire $w_{ref}(t)$, ne s'annule pas pour tout $t \geq 0$.

5.2.2 Synthèse de commande robuste stabilisante

Théorème 5.1 [Defoort et al., 2007d] Supposons que les hypothèses 5.1, 5.2 et 5.3 soient vérifiées. Le système (5.2.4) est globalement asymptotiquement stabilisable par la commande :

$$u = u_{nom}(e) + u_{disc}(e, e_{aux}) \quad (5.2.6)$$

où $u_{nom}(e)$ est définie par :

$$u_{nom}(e) = \begin{bmatrix} v_{nom} \\ w_{nom} \end{bmatrix} = \begin{bmatrix} v_{ref} \cos e_3 + \mu_3 \tanh e_1 \\ w_{ref} + \frac{\mu_1 v_{ref} e_2}{1+e_1^2+e_2^2} \frac{\sin e_3}{e_3} + \mu_2 \tanh e_3 \end{bmatrix} \quad (5.2.7)$$

$u_{disc}(e, e_{aux})$ par :

$$u_{disc}(e, e_{aux}) = \begin{bmatrix} -G_1(e) \operatorname{sign}(\sigma_1) \\ -G_2(e) \operatorname{sign}(-e_2 \sigma_1 + \sigma_2) \end{bmatrix} \quad (5.2.8)$$

et la variable de glissement $\sigma = [\sigma_1, \sigma_2]^T$ est donnée par les équations suivantes :

$$\begin{cases} \sigma &= \sigma_0(e) + e_{aux} \\ \sigma_0(e) &= [-e_1, -e_3]^T \\ \dot{e}_{aux} &= -\begin{bmatrix} -1 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix} (g_1(e, t) + g_2(e) u_{nom}(e)) \\ e_{aux}(0) &= [e_1(0), e_3(0)]^T \end{cases}$$

Les coefficients μ_i ($i = 1, 2, 3$) sont des constantes réelles positives quelconques et les gains $G_1(e)$, $G_2(e)$ vérifient les relations :

$$\begin{cases} G_1(e) \geq a_v(q) + \eta \\ G_2(e) \geq a_w(q) + \eta \end{cases} \quad (5.2.9)$$

avec $\eta > 0$.

Ceci implique que l'erreur de suivi $[e_x(t), e_y(t), e_\theta(t)]^T$ converge asymptotiquement vers zéro malgré les perturbations.

Démonstration. Considérons la fonction candidate de Lyapunov :

$$V = \frac{1}{2} \sigma^T \sigma$$

La dérivée de V suivant les trajectoires du système (5.2.4) s'écrit :

$$\begin{aligned}\dot{V} &= \sigma^T \left(\frac{\partial \sigma_0}{\partial e} \dot{e} + \dot{e}_{aux} \right) \\ &= \sigma^T \left(\begin{bmatrix} -1 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix} \dot{e} - \begin{bmatrix} -1 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix} (g_1(e, t) + g_2(e) u_{nom}(e)) \right) \\ &= \sigma^T \begin{bmatrix} -1 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix} g_2(e) (u_{disc}(e, e_{aux}) + \delta(q, t)) \\ &= \sigma^T \begin{bmatrix} 1 & -e_2 \\ 0 & 1 \end{bmatrix} (u_{disc}(e, e_{aux}) + \delta(q, t)) \\ &= [\sigma_1, -e_2 \sigma_1 + \sigma_2] (u_{disc}(e, e_{aux}) + \delta(q, t))\end{aligned}$$

Puisque l'hypothèse 5.2 est vérifiée, en utilisant les relations (5.2.9), on peut réécrire la dernière inégalité de la manière suivante :

$$\begin{aligned}\dot{V} &\leq -\eta (|\sigma_1| + |-e_2 \sigma_1 + \sigma_2|) \\ &\leq 0\end{aligned}$$

Ceci garantit l'attractivité de la surface de glissement $\{\sigma = 0\}$. Puisque la condition initiale $e_{aux}(0)$ est déterminée à partir de la relation $\sigma(0) = 0$, le régime glissant est établi dès l'instant initial et la phase de convergence est éliminée. La commande équivalente de u_{disc} qui décrit les trajectoires du système quand le régime glissant est obtenu, notée u_{disc}^{eq} et obtenue par la relation $\dot{\sigma} = 0$, est donnée par :

$$u_{disc}^{eq}(e, e_{aux}) = -\delta(q, t)$$

La dynamique du système en régime glissant, obtenue en remplaçant u_{disc} par la commande équivalente associée u_{disc}^{eq} est :

$$\dot{e} = g_1(e, t) + g_2(e) u_{nom} \quad (5.2.10)$$

Ainsi, étant donné que le système (5.2.10) est globalement asymptotiquement stable sous l'hypothèse 5.3 (voir [Jiang et al., 2001], proposition 2), le système (5.2.4) l'est également. ■

La loi de commande nominale (5.2.7) a été choisie du fait de sa propriété de bornitude. En effet,

$$\begin{cases} |v_{nom}| \leq |v_{ref}| + \mu_3 \\ |w_{nom}| \leq |w_{ref}| + \frac{\mu_1}{2} |v_{ref}| + \mu_2 \end{cases}$$

Ceci permet de prendre en compte les saturations des actionneurs.

Remarque 5.2 En pratique, afin de sélectionner les coefficients μ_i ($i = 1, 2, 3$), G_1 et G_2 , un compromis doit être réalisé entre optimalité, performance et robustesse vis-à-vis des perturbations.

Il est important de noter que l'utilisation de la commande nominale (5.2.7) seule ne donne généralement pas de très bons résultats en termes de précision au niveau du suivi de trajectoire (voir les résultats expérimentaux). C'est pourquoi le terme discontinu, permettant de rejeter une partie des perturbations, est ajouté de manière à améliorer sensiblement les résultats.

5.2.3 Résultats expérimentaux

Pour illustrer l'algorithme de commande donné dans le théorème 5.1, on effectue des tests sur le robot Pekee dans un environnement jonché d'obstacles. Le robot Pekee est un robot mobile de type unicycle constitué de trois roues : une roue folle pour permettre au robot d'être stable et deux roues motrices qui ne peuvent pas braquer et sont commandées par deux moteurs indépendants. Sa position est déterminée via deux odomètres (180 impulsions par tour de roue).

L'algorithme de planification de trajectoire, développé dans le chapitre 2, et le traitement d'images sont déportés sur un ordinateur distant (Pentium IV 2.4Ghz). Quant à l'algorithme de CMGI, il est directement implanté sur l'ordinateur embarqué équipé d'un microprocesseur Intel 486 fonctionnant sous système d'exploitation (OS) Linux temps réel. Le transfert des données entre le robot et le PC distant s'effectue par liaisons WiFi. Pour plus d'informations sur les caractéristiques du robot Pekee, le lecteur peut se référer à l'Annexe A.

Remarque 5.3 Ce robot est soumis à des perturbations et incertitudes non négligeables, dues notamment aux bruits de mesure, aux variations paramétriques (suivant l'état de la batterie), aux phénomènes de retard (sur la commande et sur la mesure), à la mauvaise calibration des moteurs, etc.

La période d'échantillonnage est $\tau = 0.1s$ (grandeur limitée par l'horloge principale du robot Pekee). Les expérimentations ont été réalisées avec les réglages suivants : $\mu_1 = 0.5$, $\mu_2 = 1$, $\mu_3 = 0.5$, $G_1 = 0.2$ et $G_2 = 0.2$.

Dans un premier temps, la commande appliquée au robot Pekee est la commande nominale (5.2.7) seule. La figure 5.3 montre les résultats associés. On peut remarquer que le robot suit la trajectoire de référence avec plus ou moins de précision. L'erreur de poursuite $\sqrt{(x - x_{ref})^2 + (y - y_{ref})^2}$ entre la trajectoire de référence planifiée et la trajectoire réellement effectuée par le robot Pekee reste inférieure à $0.11m$.

Afin d'améliorer les résultats précédents, on ajoute à la commande nominale (5.2.7) le terme discontinu (5.2.8). La figure 5.4 représente les résultats obtenus. Dans ce cas, le robot suit la trajectoire de référence avec une meilleure précision. En effet, l'erreur de poursuite $\sqrt{(x - x_{ref})^2 + (y - y_{ref})^2}$ est environ deux fois plus faible.

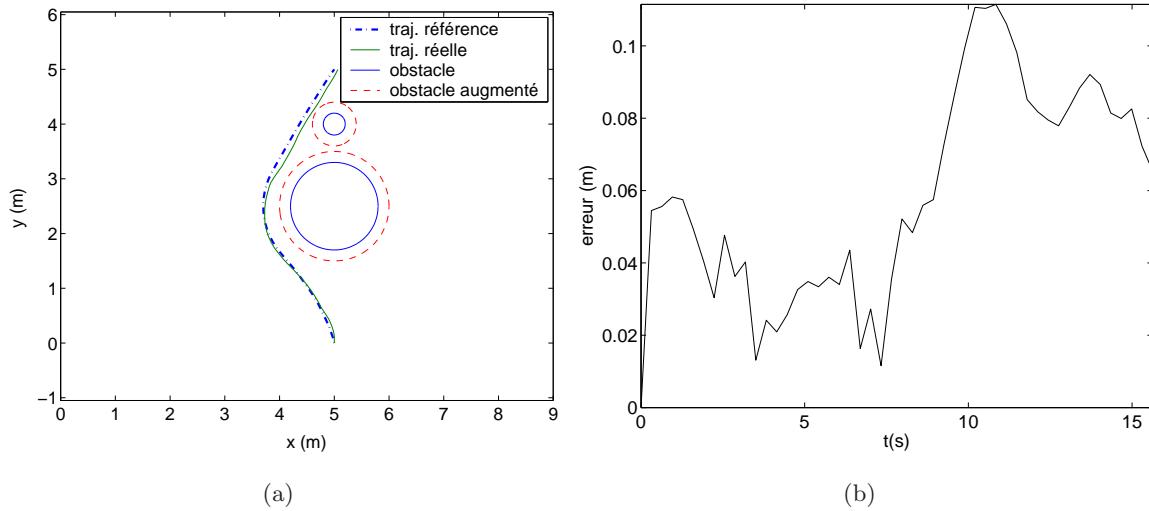


FIG. 5.3 – Application de la commande nominale (5.2.7) sur le robot Pekee. (a) Trajectoires planifiée et réelle. (b) Erreur de suivi.

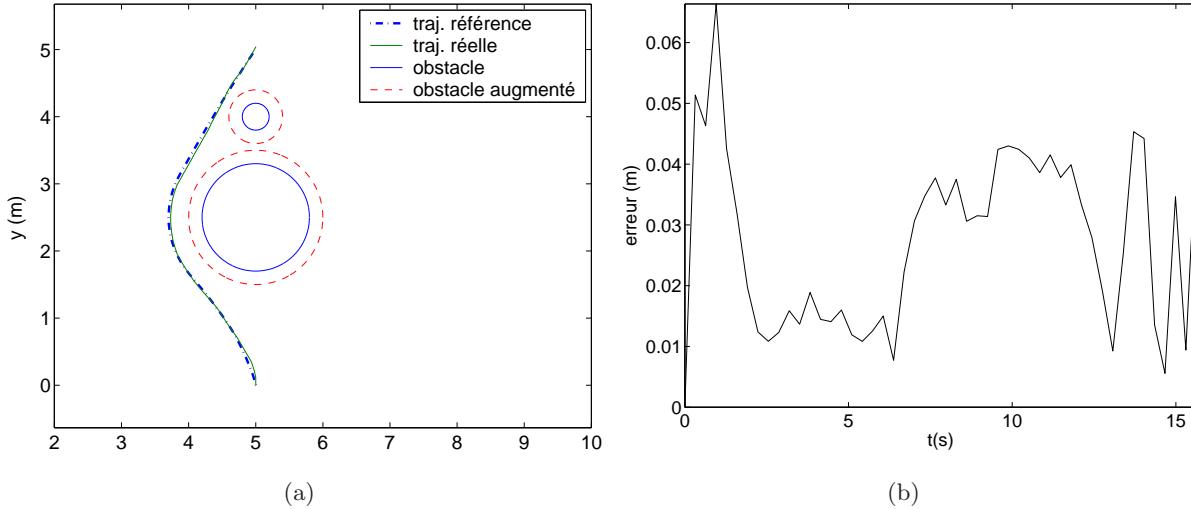


FIG. 5.4 – Application de la CMGI sur le robot Pekee. (a) Trajectoires planifiée et réelle. (b) Erreur de suivi.

Remarque 5.4 Des vidéos donnant les résultats expérimentaux obtenus sur le robot Pekee sont disponibles sur le site :

<http://syner.ec-lille.fr/robocoop/course/view.php?id=14>

Ces résultats expérimentaux mettent en lumière un des rôles de la CMGI qui consiste en l'amélioration des résultats obtenus à partir d'une commande nominale.

5.3 Approche unifiée pour la poursuite de trajectoire

Dans la partie précédente, nous avons élaboré une loi de commande robuste permettant de stabiliser sous certaines conditions les erreurs de suivi de trajectoire malgré les perturbations, pour le modèle cinématique du robot. Une extension de ces travaux pour le modèle dynamique du robot est donnée dans [Defoort et al., 2005, Defoort et al., 2006c]. Elle est basée sur l'utilisation d'extensions dynamiques et les propriétés de platitude du système. Cependant, une difficulté de mise en œuvre réside dans l'existence de singularités lorsque la trajectoire de référence possède des points d'arrêt et dans l'utilisation de différentiateurs d'ordre trois. Ceci rend l'implémentation en temps réel assez délicate du fait des limitations pratiques du robot Pekee.

Dans cette partie, nous proposons une autre méthode afin de synthétiser la commande pour des systèmes pouvant être mis, par l'intermédiaire de changements de coordonnées locaux de l'espace d'état, sous la forme du “système de Heisenberg” (ce qui est notamment le cas du modèle unicycle). Cette approche, basée sur un objectif de stabilisation pratique (i.e. stabilisation dans un voisinage d'un point), permet d'apporter une solution unifiée à de nombreux problèmes de poursuite de trajectoires (e.g. configuration fixe, trajectoires non-stationnaires) pour une très large classe de systèmes non linéaires incertains.

5.3.1 Motivations pour la stabilisation pratique

Notion de stabilité pratique

L'objectif de stabilisation pratique est évidemment moins fort que celui de stabilisation asymptotique. Mais, comme nous le verrons par la suite, il permet d'unifier l'étude d'un certain nombre de problèmes de stabilisation et correspond mieux, dans de nombreux cas, à ce qu'il est possible de réaliser de façon robuste pour les systèmes non holonomes.

Il existe de multiples définitions de la stabilité pratique dans la littérature actuelle. Si elles présentent toutes des différences qui peuvent être plus ou moins importantes, l'essence ou l'idée fondamentale reste la même : rendre la stabilité plus souple pour, entre autres, faciliter les applications industrielles. Nous cherchons à garantir que l'état se trouve à l'intérieur d'un domaine plus ou moins petit et connu, parfois sur un intervalle de temps fini.

La notion de stabilité pratique fut dans un premier temps développée dans [Lassale, 1961]. Elle est définie pour un système perturbé par rapport à un ensemble de conditions initiales et un ensemble de perturbations admissibles. L'intervalle de temps sur lequel la stabilité est étudiée est infini [Sontag, 1999]. D'autres auteurs ont abordé le problème de la stabilité pratique en temps fini [Weiss et Infante, 1967]. Dans [Grujic, 1973], l'auteur développe la notion de stabilité pratique sur un intervalle de temps fini pour des systèmes en régime forcé avec des perturbations exogènes non nulles. Des critères par méthodes de comparaison ont été obtenus dans [Perruquetti et al., 1995].

Nous utiliserons la définition suivante dont l'intérêt du point de vue applicatif est assez clair.

Définition 5.1 Considérons le système

$$\dot{x} = f(x) \quad (5.3.1)$$

où x est un vecteur de \mathbb{R}^n et $f : \mathbb{R}^n \rightarrow \mathbb{R}^n$ est une fonction continue.

L'origine du système (5.3.1) est **localement pratiquement stable** si pour un réel $\epsilon > 0$ donné, il existe deux réels positifs δ et T tels que pour toute condition initiale $\|x(0)\| \leq \delta$, $\|x(t)\| \leq \epsilon$, quel que soit $t \geq T$.

Enfin, l'origine du système (5.3.1) est **globalement pratiquement stable** si pour un réel $\epsilon > 0$ donné, il existe un réel positif T tel que pour toute condition initiale $x(0)$, $\|x(t)\| \leq \epsilon$, quel que soit $t \geq T$.

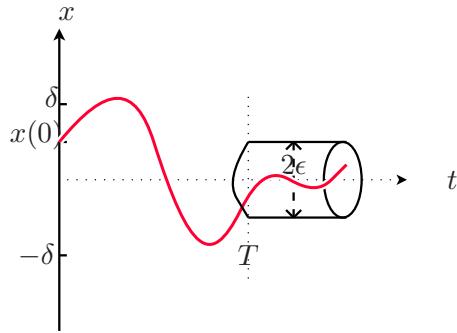


FIG. 5.5 – Illustration de la stabilité pratique.

Motivations

L'objectif de stabiliser asymptotiquement à l'aide d'un unique retour d'état continu non stationnaire, toutes les trajectoires admissibles du robot de type unicycle n'est pas réalisable [Lizarraga, 2004]. Ainsi, les résultats de stabilisation asymptotique de trajectoire existent seulement si les trajectoires de référence vérifient certaines conditions (voir par exemple l'hypothèse 5.3).

En outre, lorsque la trajectoire de référence est réduite à un point fixe, malgré l'existence de nombreuses méthodes pour la synthèse de commandes, il ne paraît pas possible de concilier l'exigence de convergence rapide avec celle de robustesse vis-à-vis d'incertitudes et de perturbations sur le modèle.

Ces arguments mettent en évidence l'intérêt, à la fois conceptuel et d'ordre pratique, de relâcher l'objectif de stabilisation asymptotique, soit parce qu'il n'est pas atteignable, soit encore parce qu'il ne l'est pas de façon efficace et robuste. L'approche basée sur la commande par modes glissants avec action intégrale (CMGI), permet d'apporter des solutions pour résoudre le problème de stabilisation pratique de trajectoire.

5.3.2 Mise sous forme du “système de Heisenberg”

Considérons le robot unicycle réel dont les équations cinématiques sont données par les équations différentielles (5.2.1).

Le problème de stabilisation pratique de trajectoire consiste à déterminer une commande permettant de stabiliser pratiquement l'erreur :

$$[e_x(t), e_y(t), e_\theta(t)]^T = [x(t) - x_{ref}(t), y(t) - y_{ref}(t), \theta(t) - \theta_{ref}(t)]^T$$

Pour résoudre ce problème, appliquons le difféomorphisme suivant [Dixon et al., 2000a] :

$$[Z, X]^T = M(\theta) \quad [e_x, e_y, e_\theta]^T \quad (5.3.2)$$

avec la matrice de transformation $M(\theta) \in \mathbb{R}^{3 \times 3}$ définie par :

$$M(\theta) = \begin{bmatrix} e_\theta \cos \theta - 2 \sin \theta & e_\theta \sin \theta + 2 \cos \theta & 0 \\ 0 & 0 & 1 \\ \cos \theta & \sin \theta & 0 \end{bmatrix} \quad (5.3.3)$$

où $Z \in \mathbb{R}$ et $X = [X_1, X_2]^T \in \mathbb{R}^2$ sont les variables auxiliaires d'erreur de suivi.

Remarque 5.5 La matrice $M(\theta)$ est non singulière. En effet, on a $\det(M(\theta)) = 2$.

Définissons également le retour d'état :

$$\begin{cases} U_1 &= w - w_{ref} \\ U_2 &= v - w(e_x \sin \theta - e_y \cos \theta) - v_{ref} \cos e_\theta \end{cases} \quad (5.3.4)$$

où $U = [U_1, U_2]^T$ est la nouvelle entrée.

Dans [Dixon et al., 2000a], il est montré qu'en utilisant le difféomorphisme (5.3.3) et le retour d'état (5.3.4), la dynamique de l'erreur de suivi est transformée en ce qui s'appelle le “système de Heisenberg” perturbé :

$$\begin{cases} \dot{Z} &= U^T J X + A(X)X + p_1 \\ \dot{X} &= U + p_2 \end{cases} \quad (5.3.5)$$

où

- J est une matrice constante antisymétrique (i.e. $J^T = -J$) définie comme :

$$J = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$$

- $A(X) \in \mathbb{R}^{1 \times 2}$ est la matrice :

$$A = \left[2v_{ref} \frac{\sin(X_1)}{X_1}, -2w_{ref} \right]$$

- le vecteur $[p_1, p_2^T]^T \in \mathbb{R}^3$ correspondant aux incertitudes et aux perturbations $p(q, t)$, est définie par :

$$[p_1, p_2^T]^T = M(\theta)p(q, t) + \frac{dM}{d\theta} [e_x, e_y, e_\theta]^T ([0, 0, 1] p(q, t))$$

Remarque 5.6 Dans ce qui suit, le terme $A(X)X$ est vu comme une perturbation sur le “système de Heisenberg” (5.1.1) donné dans l’introduction de ce chapitre. On peut noter qu’il est nul lorsque la trajectoire de référence est réduite à un point fixe.

Dans la majorité des travaux concernant la stabilisation du “système de Heisenberg” (par exemple [Bloch et Drakunov, 1996, Bloch et al., 2000, Marchand et Alamir, 2003, Prieur et Astolfi, 2003]), les commandes synthétisées sont discontinues et donnent lieu en pratique à des vitesses discontinues, ce qui pose généralement quelques problèmes. Ces problèmes peuvent évidemment être évités en ajoutant des intégrateurs dans la chaîne d’entrée de telle sorte que la partie discontinue de la commande n’agisse non plus directement sur les vitesses mais sur leurs dérivées. En outre, l’application d’une commande discontinue sur les parties électriques apparaît plus judicieuse puisque ces dernières sont généralement composées d’éléments discontinus par nature. Par conséquent, le problème devient la stabilisation du “système de Heisenberg étendu” :

$$\dot{Z} = U^T J X + \delta_Z(\Sigma, t) \quad (5.3.6)$$

$$\dot{X} = U + \delta_X(\Sigma, t) \quad (5.3.7)$$

$$\dot{U} = T + \delta_U(\Xi, t) \quad (5.3.8)$$

où $\Xi = [Z, X^T, Y^T]^T \in \mathbb{R}^5$ est le vecteur d’état, $T \in \mathbb{R}^2$ est la nouvelle commande et $\Sigma = [Z, X^T]^T \in \mathbb{R}^3$. Les termes $\delta_Z(\Sigma, t) = AX + p_1$, $\delta_X(\Sigma, t) = p_2$ et $\delta_U(\Xi, t)$ représentent les perturbations sur le modèle, les incertitudes paramétriques et éventuellement les dynamiques non modélisées. Ils sont supposés bornés par des fonctions connues et sont suffisamment différentiables de telle sorte que :

$$\begin{cases} |\delta_Z(\Sigma, t)| \leq a_Z(\Sigma), \|\delta_X(\Sigma, t)\| \leq a_X(\Sigma) \\ |\dot{\delta}_Z(\Sigma, t)| \leq \bar{a}_Z(\Xi), \|\dot{\delta}_X(\Sigma, t)\| \leq \bar{a}_X(\Xi) \\ \|\delta_U(\Xi, t)\| \leq a_U(\Xi) \end{cases} \quad (5.3.9)$$

où a_Z , a_X , a_U , \bar{a}_Z et \bar{a}_X sont des fonctions non négatives connues.

Remarque 5.7 Lorsque les termes $\delta_Z(\Sigma, t)$ et $\delta_X(\Sigma, t)$ sont différents du vecteur nul (ce qui est nécessairement le cas de la poursuite de trajectoire du fait de la présence du terme $A(X)X$ dans δ_Z), les perturbations ne vérifient pas la condition de recouvrement.

Remarque 5.8 Du fait de la présence des perturbations δ_Z , δ_X et δ_U dans le système (5.3.6)-(5.3.8), la classe de systèmes étudiée ici est différente de celle considérée dans [Prieur et Astolfi, 2003, Valtolina et Astolfi, 2003]. En effet, dans ces travaux, ainsi que dans [Marchand et Alamir, 2003,

Prieur et Trelat, 2005], les commandes discontinues sont construites de manière à stabiliser les systèmes mis sous forme chaînée. Or, généralement, le système (5.3.6)-(5.3.8) ne peut pas se mettre sous cette forme. Ainsi, la classe de système que nous proposons de stabiliser est plus large.

Le système (5.3.6)-(5.3.8) peut être vu comme l’interconnection de deux sous systèmes : une chaîne d’intégrateurs de dimension deux (équations (5.3.7)-(5.3.8)) avec présence de perturbations et un système scalaire dont la dynamique est entièrement décrite par le sous-système (5.3.6).

Remarque 5.9 *Le problème de stabilisation du “système de Heisenberg” (5.3.6)-(5.3.8) est très important puisque d’autres systèmes physiques tels que la machine asynchrone peuvent se mettre sous cette forme (voir [Defoort et al., 2007b]).*

5.3.3 Stabilisation pratique du “système de Heisenberg”

L’objectif de commande est la stabilisation pratique des variables Z et X du système (5.3.6)-(5.3.8) malgré la présence de perturbations. Il sera atteint en utilisant une CMGI. En effet, comme il le sera vu par la suite, la CMGI a une structure particulièrement bien adaptée à la stabilisation du “système de Heisenberg” et se révèle utile pour obtenir un bon compromis entre performance et robustesse.

Remarque 5.10 *On peut noter qu’il est essentiel de stabiliser la dynamique de Z plus rapidement que celle de X puisque dès que la dynamique de X est stabilisée vers zéro, il n’est plus possible d’agir sur celle de Z . Ainsi, le choix de la surface de glissement est décisif dans la synthèse de la CMGI.*

Dans un premier temps, une CMGI d’ordre un est construite de manière à stabiliser le “système de Heisenberg” lorsque les perturbations vérifient la condition de recouvrement. Puis, ces résultats sont généralisés de manière à couvrir une classe de systèmes plus large.

“Système de Heisenberg” avec perturbations vérifiant la condition de recouvrement

Dans un premier temps, on suppose que les perturbations vérifient la condition de recouvrement, c’est-à-dire que :

$$\delta_Z(\Sigma, t) = \delta_X(\Sigma, t) = 0$$

Puisque la dynamique de Z doit être stabilisée plus rapidement que celle de X , il apparaît intéressant de choisir la variable de glissement $s = [s_1, s_2]^T \in \mathbb{R}^2$:

$$\begin{cases} s_1(t) &= \dot{Z}(t) + m_1 Z(t) + m_2 Z_{aux}(t) \\ s_2(t) &= \dot{\psi}(t) + m_3 \psi(t) + m_4 \psi_{aux}(t) \end{cases} \quad (5.3.10)$$

avec

$$\psi(t) = \frac{1}{2} X(t)^T X(t) - \Theta(Z(t)) - \frac{1}{2} \epsilon \quad (5.3.11)$$

où $0 < \epsilon \ll 1$ et $\Theta : \mathbb{R} \rightarrow \mathbb{R}^+$ est une fonction définie positive de classe C^2 . Les constantes m_i ($i = 1, \dots, 4$) vérifient les relations :

$$\begin{cases} m_i > 0, & i = 1, \dots, 4 \\ m_3 \geq 2\sqrt{m_4} \end{cases} \quad (5.3.12)$$

Les équations (5.3.10) peuvent être réécrites, étant donné que $\delta_Z(\Sigma, t) = \delta_X(\Sigma, t) = 0$, sous la forme :

$$\begin{cases} s_1 = U^T J X + m_1 Z + m_2 Z_{aux} \\ s_2 = X^T U - \frac{\partial \Theta(Z)}{\partial Z} U^T J X + m_3 \psi + m_4 \psi_{aux} \end{cases} \quad (5.3.13)$$

Les fonctions Z_{aux} et ψ_{aux} introduisent l'action intégrale et fournissent un degré de liberté supplémentaire dans la construction de la variable de glissement. Leur dynamique est régie par les équations différentielles :

$$\begin{cases} \dot{Z}_{aux}(t) = Z(t) \\ \dot{\psi}_{aux}(t) = \psi(t) \end{cases} \quad (5.3.14)$$

Les conditions initiales $Z_{aux}(0)$ et $\psi_{aux}(0)$ sont déterminées à partir des relations $s_1(0) = 0$ et $s_2(0) = 0$. Par conséquent, afin d'éliminer la phase de convergence, les variables auxiliaires $Z_{aux}(0)$ et $\psi_{aux}(0)$ sont initialisées aux valeurs :

$$\begin{aligned} Z_{aux}(0) &= -\frac{1}{m_2} (U^T J X + m_1 Z)(0) \\ \psi_{aux}(0) &= -\frac{1}{m_4} \left(X^T U - \frac{\partial \Theta(Z)}{\partial Z} U^T J X + m_3 \psi \right)(0) \end{aligned} \quad (5.3.15)$$

On peut noter que le système (5.3.6)-(5.3.8) admet un degré relatif [1, 1] par rapport à s . En effet, en dérivant une fois s_1 et une fois s_2 le long des trajectoires du système, on obtient :

$$\dot{s} = \bar{\varphi}(\Xi) T + \bar{\phi}(\Xi) + \delta_\phi(\Xi) \quad (5.3.16)$$

où

$$\begin{aligned} \bar{\varphi}(\Xi) &= \begin{bmatrix} -X^T J \\ X^T + \frac{\partial \Theta(Z)}{\partial Z} X^T J \end{bmatrix} \\ \bar{\phi}(\Xi) &= \begin{bmatrix} m_1 U^T J X + m_2 Z \\ U^T U - \frac{\partial^2 \Theta(Z)}{\partial Z^2} (U^T J X)^2 - \frac{\partial \Theta(Z)}{\partial Z} + m_3 X^T U - m_3 \frac{\partial \Theta(Z)}{\partial Z} U^T J X + m_4 \psi \end{bmatrix} \\ \delta_\phi(\Xi) &= \begin{bmatrix} -X^T J \delta_U(\Xi) \\ X^T \left(\frac{\partial \Theta(Z)}{\partial Z} J \delta_U(\Xi) + \delta_U(\Xi) \right) \end{bmatrix} \end{aligned}$$

Ce système est fortement couplé par la matrice $\bar{\varphi}(\Xi) \in \mathbb{R}^{2 \times 2}$. Afin de répondre en partie à ce problème, appliquons le retour d'état :

$$T = \begin{bmatrix} J X & X - \frac{\partial \Theta(Z)}{\partial Z} J X \end{bmatrix} W \quad (5.3.17)$$

où $W \in \mathbb{R}^2$ est la nouvelle entrée. L'équation différentielle (5.3.16) est transformée en la forme suivante :

$$\dot{s} = N(\Xi)W + \bar{\phi}(\Xi) + \delta_\phi(\Xi) \quad (5.3.18)$$

où

$$\begin{aligned} N(\Xi) &= \begin{bmatrix} -X^T J \\ X^T + \frac{\partial \Theta(Z)}{\partial Z} X^T J \end{bmatrix} \begin{bmatrix} JX & X - \frac{\partial \Theta(Z)}{\partial Z} JX \end{bmatrix} \\ &= \begin{bmatrix} \|JX\|^2 & -\frac{\partial \Theta(Z)}{\partial Z} \|JX\|^2 \\ -\frac{\partial \Theta(Z)}{\partial Z} \|JX\|^2 & \|X\|^2 + \left(\frac{\partial \Theta(Z)}{\partial Z}\right)^2 \|JX\|^2 \end{bmatrix} \end{aligned} \quad (5.3.19)$$

Remarque 5.11 Initialement, la CMGI a été introduite pour rendre les trajectoires du système robustes vis-à-vis des perturbations dès l'instant initial. Ici, une autre application est mise en lumière : la possibilité d'éviter les singularités dans la commande nominale, ce qui est difficilement réalisable avec une commande classique par modes glissants du fait de la phase de convergence pendant laquelle les trajectoires du système sont sensibles vis-à-vis des perturbations.

Il est très important de noter que le déterminant de la matrice $N(\Xi)$ est égal à $\|JX\|^2 \|X\|^2$. Puisque la matrice J n'est pas singulière, la matrice de découplage $N(\Xi)$ est toujours inversible sauf sur la variété $X^T X = 0$. Etant donné que l'utilisation de la CMGI permet d'éliminer la phase de convergence vers la surface de glissement $\{s = 0\}$, la variété $X^T X = 0$ n'est jamais atteinte. Enfin, puisque $\|X(t)\| > 0$ ($\forall t \geq 0$), seule une stabilité pratique des trajectoires du système (5.3.6)-(5.3.8) est souhaitée.

Théorème 5.2 [Defoort et al., 2007b] Soit le système (5.3.6)-(5.3.8) avec $\delta_Z(\Sigma, t) = \delta_X(\Sigma, t) = 0$. Supposons qu'il existe une constante $\epsilon > 0$ telle que $\|X(0)\|^2 > \epsilon$, des constantes m_i ($i = 1, \dots, 4$) vérifiant les conditions (5.3.12) et une fonction $\Theta(Z)$ de classe C^2 définie positive telle que $\psi(0) = \frac{1}{2} (\|X(0)\|^2 - \epsilon) - \Theta(Z(0))$ soit positive, c'est-à-dire que :

$$\frac{\|X(0)\|^2 - \epsilon}{2} \geq \Theta(Z(0)) \quad (5.3.20)$$

Alors, le système (5.3.6)-(5.3.8) est pratiquement stable sous la loi de commande :

$$T = \begin{bmatrix} JX & X - \frac{\partial \Theta(Z)}{\partial Z} JX \end{bmatrix} (W_{nom}(\Xi) + W_{disc}(\Xi, Z_{aux}, \psi_{aux})) \quad (5.3.21)$$

où $W_{nom}(\Xi)$ est définie par :

$$W_{nom}(\Xi) = -N(\Xi)^{-1} \bar{\phi}(\Xi) \quad (5.3.22)$$

et $W_{disc}(\Xi, Z_{aux}, \psi_{aux})$ est la partie discontinue, construite de manière à forcer le régime glissant sur la surface $\{s = 0\}$:

$$W_{disc}(\Xi, Z_{aux}, \psi_{aux}) = -N(\Xi)^{-1} G(\Xi) [\text{sign}(s_1), \text{sign}(s_2)]^T \quad (5.3.23)$$

Le gain $G(\Xi)$ vérifie la relation

$$G(\Xi) > \|\delta_\phi(\Xi)\| + \eta \quad (5.3.24)$$

avec $\eta > 0$.

Démonstration. Considérons la fonction candidate de Lyapunov :

$$V = \frac{1}{2} s^T s$$

La dérivée de V suivant les trajectoires du système (5.3.6)-(5.3.8) s'écrit :

$$\begin{aligned}\dot{V} &= s^T (N(\Xi)W + \bar{\phi}(\Xi) + \delta_\phi(\Xi)) \\ &= s^T (-G(\Xi) [\text{sign}(s_1), \text{sign}(s_2)]^T + \delta_\phi(\Xi))\end{aligned}$$

En utilisant les relations (5.3.24), on en déduit que :

$$\begin{aligned}\dot{V} &\leq -\eta(|s_1| + |s_2|) \\ &\leq -\eta\sqrt{2}\sqrt{V}\end{aligned}$$

Ceci garantit l'attractivité et la stabilité en temps fini de la surface de glissement $\{s = 0\}$. En outre, puisque $s(0) = 0$, les trajectoires d'état du système évoluent sur la surface de glissement dès l'instant initial. Par conséquent, la phase de convergence est éliminée. La commande équivalente de W_{disc} , notée W_{disc}^{eq} et obtenue par la relation $\dot{s} = 0$ est donnée par :

$$W_{disc}^{eq}(\Xi, Z_{aux}, \psi_{aux}) = -\delta_\phi(\Xi)$$

Les dynamiques de Z et de ψ en régime glissant, obtenues en remplaçant W_{disc} par la commande équivalente associée W_{disc}^{eq} sont :

$$\begin{cases} \ddot{Z} = -m_1\dot{Z} - m_2Z \\ \ddot{\psi} = -m_3\dot{\psi} - m_4\psi \end{cases} \quad (5.3.25)$$

Puisque les constantes m_i ($i = 1, \dots, 4$) vérifient les conditions (5.3.12), les dynamiques du système (5.3.25) sont exponentiellement stables. En outre, les coefficients m_3 et m_4 ont été choisis de manière à rendre la variation de la fonction ψ monotone. Ainsi, en utilisant la condition (5.3.20), on obtient pour tout $t \geq 0$:

$$\psi(t) \geq 0$$

Par conséquent, les propriétés désirées suivantes sont établies :

- $\forall t \geq 0$, $\psi(t) \geq 0 \implies X^T X \geq 2\Theta(Z) + \epsilon \implies X^T X > 0$ (car $\Theta(Z)$ est définie positive). Les singularités de la matrice $N(\Xi)$ et donc de la commande T sont évitées (voir la remarque 5.11).
- Le système (5.3.6)-(5.3.8) est pratiquement stable. En effet, les dynamiques de Z et ψ sont exponentiellement stables. De plus, lorsque $Z = 0$ et $\psi = 0$, $\|X\|^2 = \epsilon$ puisque $\Theta(0) = 0$.

■

Remarque 5.12 La condition d'existence d'une constante $\epsilon > 0$ telle que $\|X(0)\|^2 > \epsilon$, n'est pas restrictive. En effet, lorsque cette hypothèse n'est pas vérifiée, une phase d'initialisation au cours de laquelle une commande adéquate est appliquée pendant un court laps de temps permet de s'éloigner de la variété $\|X\| = 0$. Ensuite, la loi de commande issue du théorème 5.2 est appliquée au système (5.3.6)-(5.3.8).

Exemple 5.1 Considérons l'exemple du “système de Heisenberg” (5.3.6)-(5.3.8) sans perturbation, c'est-à-dire $\delta_Z = \delta_X = \delta_U = 0$. Les paramètres de la variable de glissement (5.3.13), choisis de manière à vérifier les conditions (5.3.12) et (5.3.20) sont $m_1 = m_3 = 2$, $m_2 = m_4 = 1$, $\epsilon = 10^{-3}$ et $\Theta(Z) = \frac{1}{2}Z^2$ (fonction définie positive). La période d'échantillonnage est $\tau = 10^{-4}s$. Le gain $G = 5$ est ajusté de telle sorte que le système (5.3.6)-(5.3.8) soit pratiquement stable. La figure 5.6 donne la réponse temporelle du système commandé. On peut remarquer que l'état Z est asymptotiquement stable et que $\|X(t)\| \leq 1.5 \cdot 10^{-2}$ pour tout $\forall t \geq 13s$ (stabilisation pratique de X).

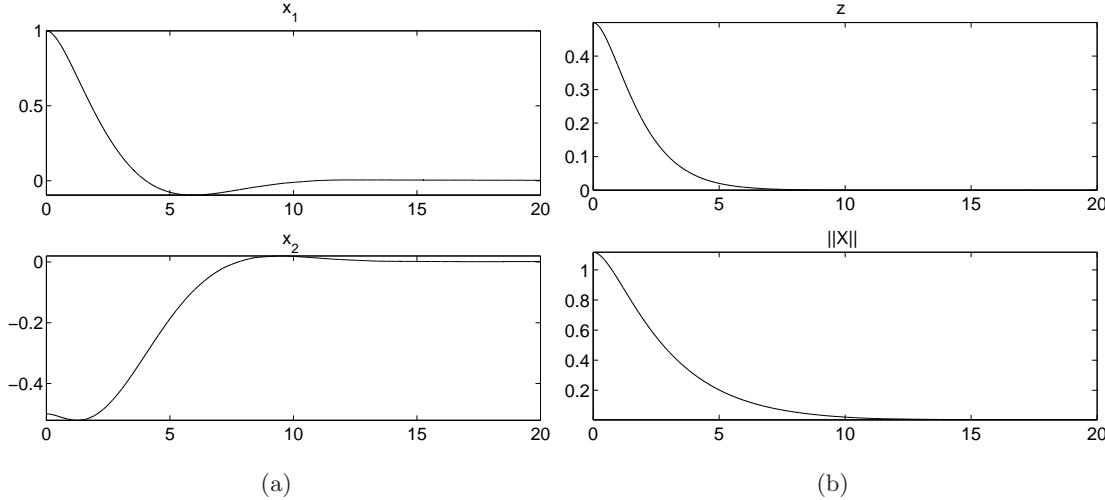


FIG. 5.6 – Stabilisation pratique du “système de Heisenberg” non perturbé.

“Système de Heisenberg” avec perturbations quelconques

Dorénavant, on ne suppose plus que les perturbations vérifient la condition de recouvrement. Par conséquent, les incertitudes sur les dynamiques de Z et de X ne permettent plus d'utiliser la variable de glissement (5.3.13). Afin de contrecarrer ce problème, on utilise une CMGI d'ordre deux.

Hypothèse 5.4 Les conditions initiales $\dot{Z}(0)$ et $\dot{X}(0)$ sont supposées connues.

Remarque 5.13 L'hypothèse 5.4 n'est pas réellement restrictive. En effet, il est possible avant d'appliquer la commande par modes glissants, d'obtenir les conditions initiales $\dot{Z}(0)$ et $\dot{X}(0)$ à l'aide d'un différentiateur robuste (voir [Levant, 1998]) ou par le biais de techniques algébriques (voir le projet ALIEN : Algèbre pour l'Identification et Estimation Numérique [INRIA, 2007]).

Soit la variable de glissement $\sigma = [\sigma_1, \sigma_2]^T \in \mathbb{R}^2$:

$$\begin{cases} \sigma_1(t) = Z(t) + m_1 Z_{aux,1}(t) + m_2 Z_{aux,2}(t) \\ \sigma_2(t) = \psi(t) + m_3 \psi_{aux,1}(t) + m_4 \psi_{aux,2}(t) \end{cases} \quad (5.3.26)$$

où la fonction $\psi(t)$ est définie par (5.3.11) et les constantes m_i ($i = 1, \dots, 4$) vérifient les conditions (5.3.12). Les dynamiques de $Z_{aux,i}(t)$ et $\psi_{aux,i}(t)$ ($i = 1, 2$) sont régies par les équations différentielles suivantes :

$$\begin{cases} \dot{Z}_{aux,1}(t) &= Z(t) \\ \dot{Z}_{aux,2}(t) &= Z_{aux,1}(t) \\ \dot{\psi}_{aux,1}(t) &= \psi(t) \\ \dot{\psi}_{aux,2}(t) &= \psi_{aux,1}(t) \end{cases} \quad (5.3.27)$$

Les conditions initiales $Z_{aux,i}(0)$ et $\psi_{aux,i}(0)$ ($i = 1, 2$) sont déterminées à partir des relations $\sigma_i(0) = 0$ et $\dot{\sigma}_i(0) = 0$ ($i = 1, 2$). Par conséquent, afin d'éliminer la phase de convergence, les variables auxiliaires sont initialisées aux valeurs :

$$\begin{cases} Z_{aux,1}(0) &= -\frac{1}{m_2}(\dot{Z} + m_1 Z)(0) \\ Z_{aux,2}(0) &= -\frac{1}{m_2}(Z + m_1 Z_{aux,1})(0) \\ \psi_{aux,1}(0) &= -\frac{1}{m_4} \left(X^T \dot{X} - \frac{\partial \Theta(Z)}{\partial Z} \dot{Z} + m_3 \psi \right)(0) \\ \psi_{aux,2}(0) &= -\frac{1}{m_4} (\psi + m_3 \psi_{aux,1})(0) \end{cases}$$

On peut noter que le système (5.3.6)-(5.3.8) admet un degré relatif [2, 2] par rapport à σ . En effet, en dérivant deux fois σ le long des trajectoires du système (5.3.6)-(5.3.8), on obtient :

$$\ddot{\sigma} = \bar{\varphi}(\Xi)T + \bar{\phi}(\Xi) + \tilde{\delta}_\phi(\Xi) \quad (5.3.28)$$

où

$$\tilde{\delta}_\phi = \begin{bmatrix} U^T J \delta_X - X^T J \delta_U + \dot{\delta}_Z + m_1 \delta_Z \\ \delta_X^T \left[2U + \delta_X + \frac{\partial \Theta(Z)}{\partial Z} JU \right] + X^T \left[\frac{\partial \Theta(Z)}{\partial Z} J \delta_U + \delta_U + \dot{\delta}_X \right] \\ - \frac{\partial^2 \Theta(Z)}{\partial Z^2} \delta_Z \left[2U^T JX + \delta_Z \right] - \frac{\partial \Theta(Z)}{\partial Z} \dot{\delta}_Z + m_3 X^T \delta_X - m_3 \frac{\partial \Theta(Z)}{\partial Z} \delta_Z \end{bmatrix}$$

Comme précédemment, appliquons le retour d'état :

$$T = \begin{bmatrix} JX & X - \frac{\partial \Theta(Z)}{\partial Z} JX \end{bmatrix} \widetilde{W} \quad (5.3.29)$$

où $\widetilde{W} \in \mathbb{R}^2$ est la nouvelle entrée. L'équation différentielle (5.3.28) devient :

$$\ddot{\sigma} = N(\Xi) \widetilde{W} + \bar{\phi}(\Xi) + \tilde{\delta}_\phi(\Xi)$$

Avant de synthétiser la commande, il est nécessaire de vérifier l'hypothèse suivante :

Hypothèse 5.5 Il existe deux constantes positives S_0 et C_0 telles que, dans un voisinage $\|\sigma(\Xi)\| < S_0$, l'inégalité suivante soit vérifiée :

$$\|\tilde{\delta}_\phi(\Xi, t)\| < C_0, \quad \forall t \geq 0$$

Notons que cette hypothèse est relativement peu restrictive. En effet, si $\tilde{\delta}_\phi(\Xi, t)$ est continue, alors en se restreignant à un compact, elle est automatiquement vérifiée.

Théorème 5.3 [Defoort et al., 2007b] Soit le système général (5.3.6)-(5.3.8). Supposons qu'il existe une constante $\epsilon > 0$ telle que $\|X(0)\|^2 > \epsilon$, des constantes m_i ($i = 1, \dots, 4$) vérifiant les conditions (5.3.12) et une fonction $\Theta(Z)$ de classe C^2 définie positive telle que $\psi(0) = \frac{1}{2}(\|X(0)\|^2 - \epsilon) - \Theta(z(0))$ soit positive.

Alors, le système (5.3.6)-(5.3.8) est pratiquement stable sous la loi de commande :

$$T = \begin{bmatrix} JX & X - \frac{\partial \Theta(Z)}{\partial Z} JX \end{bmatrix} \left(\widetilde{W}_{nom} + \widetilde{W}_{disc} \right) \quad (5.3.30)$$

où \widetilde{W}_{nom} est définie par :

$$\widetilde{W}_{nom} = -N(\Xi)^{-1} \overline{\phi}(\Xi) \quad (5.3.31)$$

et \widetilde{W}_{disc} par :

$$\widetilde{W}_{disc} = -N(\Xi)^{-1} \begin{bmatrix} \begin{cases} \lambda_m \operatorname{sign}(\sigma_1) & \text{si } \sigma_1 \Delta_{\sigma_1} \leq 0 \\ \lambda_M \operatorname{sign}(\sigma_1) & \text{si } \sigma_1 \Delta_{\sigma_1} > 0 \end{cases} \\ \begin{cases} \lambda_m \operatorname{sign}(\sigma_2) & \text{si } \sigma_2 \Delta_{\sigma_2} \leq 0 \\ \lambda_M \operatorname{sign}(\sigma_2) & \text{si } \sigma_2 \Delta_{\sigma_2} > 0 \end{cases} \end{bmatrix} \quad (5.3.32)$$

Δ_{σ_1} (respectivement Δ_{σ_2}) représente le signe de la dérivée de σ_1 (respectivement σ_2). Cette dérivée est obtenue à l'aide d'un différentiateur robuste d'ordre un¹ ([Levant, 1998]). Les gains λ_m et λ_M vérifient les relations :

$$\begin{cases} \lambda_m > C_0 \\ \lambda_M > \lambda_m + 2C_0 \end{cases} \quad (5.3.33)$$

La démonstration de ce théorème est largement inspirée de la démonstration du théorème 5.2. Ainsi, seules les principales lignes directrices sont données.

Démonstration. En utilisant les résultats issus de [Levant, 1993], il peut être montré que, sous la commande (5.3.30), les trajectoires du système (5.3.6)-(5.3.8) convergent en temps fini sur la surface de glissement $\{\sigma = \dot{\sigma} = 0\}$. En outre, puisque $\sigma(0) = \dot{\sigma}(0) = 0$, la phase de convergence est éliminée. La commande équivalente de \widetilde{W}_{disc} , notée $\widetilde{W}_{disc}^{eq}$ et obtenue par la relation $\ddot{\sigma} = 0$ est donnée par :

$$\widetilde{W}_{disc}^{eq} = -\delta_\phi(\Xi)$$

Les dynamiques de Z et de ψ en régime glissant, obtenues en remplaçant \widetilde{W}_{disc} par la commande équivalente associée $\widetilde{W}_{disc}^{eq}$ sont :

$$\begin{cases} \ddot{Z} = -m_1 \dot{Z} - m_2 Z \\ \ddot{\psi} = -m_3 \dot{\psi} - m_4 \psi \end{cases} \quad (5.3.34)$$

Par conséquent, en suivant l'idée de la démonstration du théorème 5.2, on obtient les propriétés suivantes :

¹Notons que seul le signe de la dérivée est utilisé dans la synthèse de la commande discontinue.

- $\forall t \geq 0, \|X(t)\| > 0,$
- le système (5.3.6)-(5.3.8) est pratiquement stable (convergence dans une boule centrée sur l'origine dont le rayon dépend de la constante ϵ).

Exemple 5.2 Reprenons l'exemple du “système de Heisenberg” (5.3.6)-(5.3.8) en considérant cette fois des perturbations. Dans les simulations, $\delta_Z = 0.1 \sin(2\pi t)$, $\delta_X = [0.1 \sin(2\pi t), 0.1 \sin(2\pi t)]^T$ et δ_U est un bruit blanc gaussien de moyenne nulle et de variance 0.1. Les paramètres de la variable de glissement (5.3.26), choisis de manière à vérifier les conditions (5.3.12) et (5.3.20) sont $m_1 = m_3 = 2$, $m_2 = m_4 = 1$, $\epsilon = 10^{-3}$ et $\Theta(Z) = \frac{1}{2}Z^2$ (fonction définie positive). La période d'échantillonnage est $\tau = 10^{-4}s$. Les gains $\lambda_m = 5$ et $\lambda_M = 20$ sont ajustés de telle sorte que le système (5.3.6)-(5.3.8) soit pratiquement stable. La figure 5.7 donne la réponse temporelle du système commandé. On peut remarquer que l'état Z est asymptotiquement stable et que $\|X(t)\| \leq 3 \cdot 10^{-2}$ pour tout $t \geq 13s$ (stabilisation pratique de X).

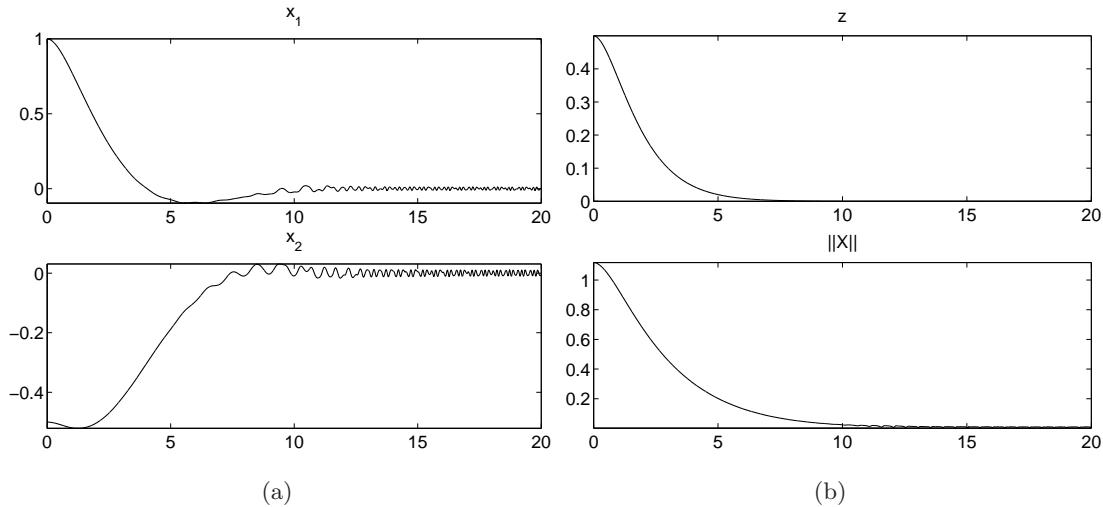


FIG. 5.7 – Stabilisation pratique du “système de Heisenberg” perturbé.

5.3.4 Résultats expérimentaux

Pour illustrer l'algorithme de commande par modes glissants d'ordre deux proposé dans le théorème 5.3, on effectue des tests sur le robot Pekee. La présence d'incertitudes et de perturbations sur le système (voir la remarque 5.3) ne permet pas d'obtenir des résultats satisfaisants en utilisant la CMGI d'ordre un définie dans le théorème 5.2.

En plus de très bonnes propriétés de robustesse, la loi de commande, écrite en C et implémentée directement sur l'ordinateur embarqué du robot, permet à la fois de résoudre le problème de poursuite de trajectoire réduite à un point fixe et le problème de poursuite de trajectoire non stationnaire.

L'acquisition des mesures et le calcul de la commande se font à une période d'échantillonage de $\tau = 0.1s$ (grandeur limitée par l'horloge principale du robot Pekee). Les expérimentations ont été réalisées avec les réglages suivants :

$$m_1 = 1, \quad m_2 = 1, \quad m_3 = 2, \quad m_4 = 1, \quad \Theta(Z) = \frac{1}{2}Z^2, \quad \epsilon = 0.1, \quad \lambda_m = 1, \quad \lambda_M = 5$$

Stabilisation d'une trajectoire réduite à un point fixe

La configuration initiale du robot Pekee est : $x(0) = 1m$, $y(0) = 1m$ et $\theta(0) = 0.1rad$. De plus, initialement, sa vitesse est nulle. L'objectif est la stabilisation pratique de ce robot à l'origine. Sa position et son orientation ainsi que les vitesses linéaire et angulaire appliquées sont représentées dans la figure 5.8(a) et (b) respectivement. Nous visualisons la trajectoire du système sur la figure 5.8(c). On peut remarquer que le robot se gare en $8.4s$ en effectuant des manœuvres de type créneau.

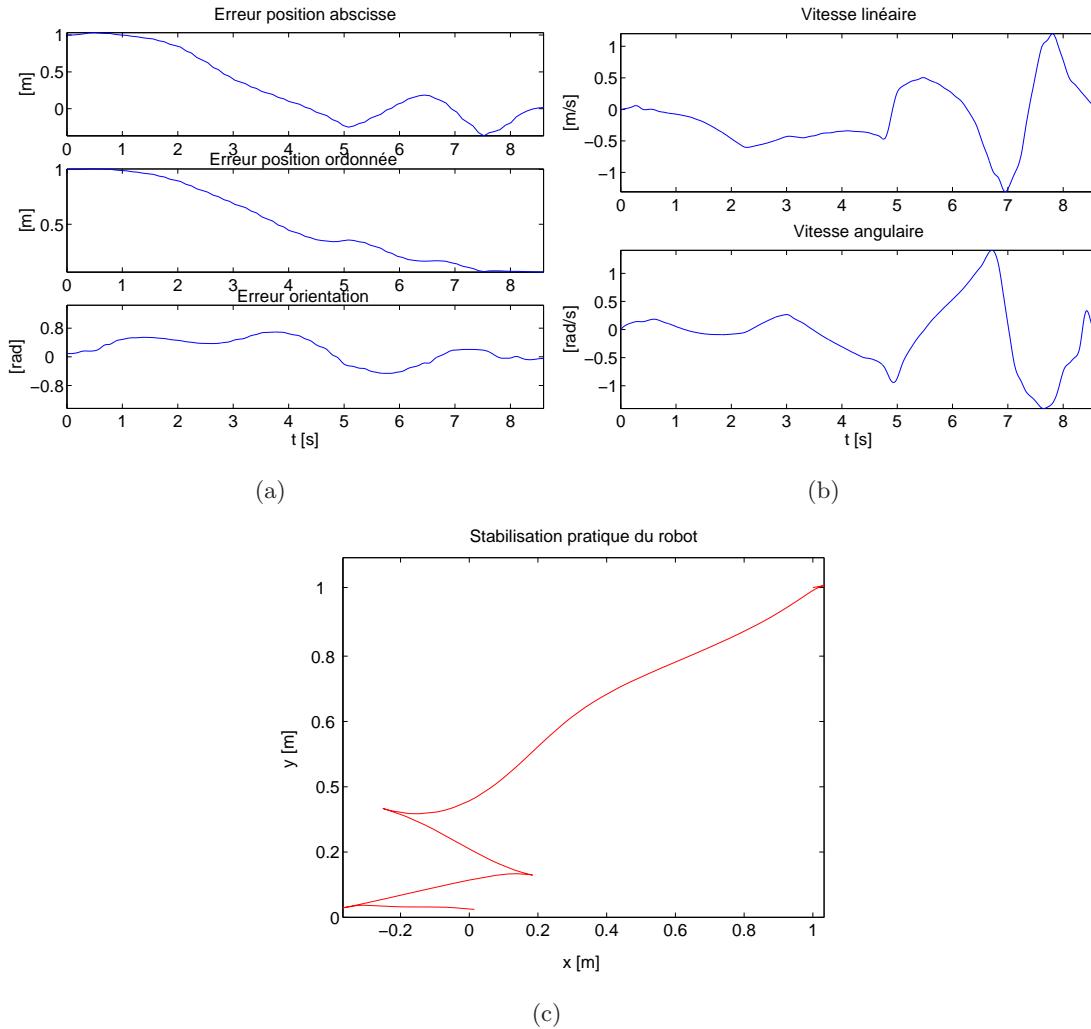


FIG. 5.8 – Stabilisation en une configuration fixe du robot Pekee.

Stabilisation d'une trajectoire non stationnaire

Dans cette expérimentation, les vitesses linéaire et angulaire de référence sont :

$$\begin{aligned} v_{ref} &= 0.3(1 - \exp(-t)) \\ w_{ref} &= 0.1 \sin(0.5t) \end{aligned}$$

Par conséquent, la trajectoire planifiée est une courbe représentée dans la figure 5.9(c). La configuration initiale du robot Pekee est $x(0) = 0.05$ m, $y(0) = 0$ m et $\theta(0) = 0.1$ rad et sa vitesse initiale est nulle. L'objectif est ici la stabilisation pratique des erreurs de suivi de trajectoire. La position et l'orientation du véhicule ainsi que les vitesses linéaire et angulaire appliquées sont représentées dans la figure 5.9(a) et (b) respectivement. Nous visualisons les trajectoires réelle et planifiée sur la figure 5.9(c). Nous observons bien que les erreurs de suivi de trajectoire sont pratiquement stables.

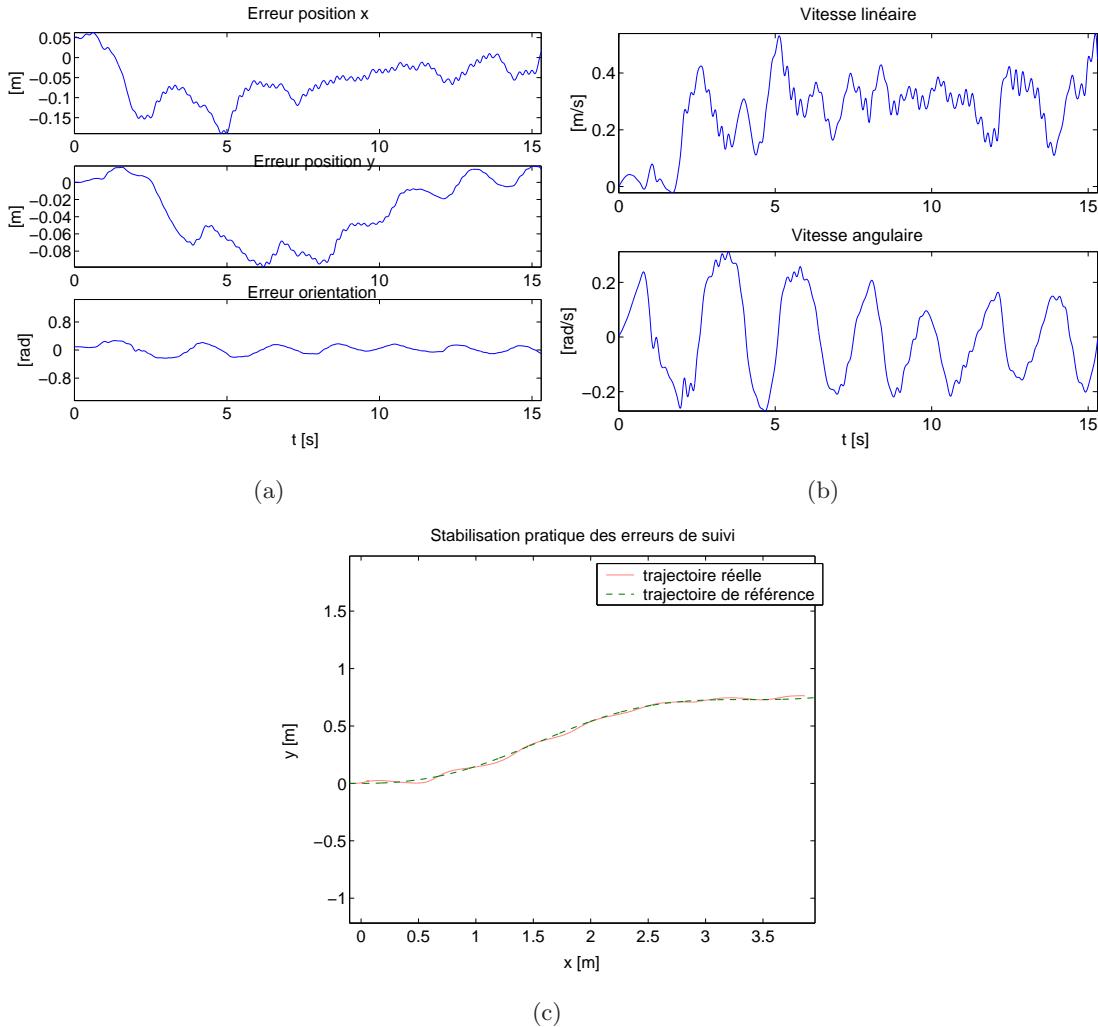


FIG. 5.9 – Stabilisation d'une trajectoire non stationnaire pour le robot Pekee.

Remarque 5.14 Les vidéos associées aux résultats expérimentaux obtenus sur le robot Pekee sont disponibles sur le site :

<http://syner.ec-lille.fr/robocoop/course/view.php?id=14>

5.4 Conclusion

Sur la base de méthodes originales, deux algorithmes de commande ont été synthétisés et implémentés afin de garantir la stabilisation et/ou le suivi de trajectoire malgré la présence d'incertitudes et de perturbations.

La première méthode consiste à ajouter un terme discontinu à une commande continue qui stabilise les erreurs de suivi de trajectoire en l'absence de perturbations (voir le théorème 5.1). Cette méthodologie permet de mettre en lumière un des rôles de la CMGI qui consiste en l'amélioration des résultats obtenus à partir d'une commande nominale.

En ce qui concerne la seconde méthode, afin de synthétiser la loi de commande, le modèle du robot mobile est mis sous la forme du “système de Heisenberg”. Puis, une formulation spécifique de la surface de glissement est introduite pour de tels systèmes. Enfin, après avoir généralisé le principe de la CMGI, une CMGI d'ordre deux est synthétisée. Il est montré qu'en relâchant l'objectif de stabilisation asymptotique en un objectif de stabilisation pratique, cette technique permet d'apporter une solution unifiée à de nombreux problèmes de poursuite de trajectoires (e.g. configuration fixe, trajectoires non stationnaires) en présence de perturbations ne vérifiant pas nécessairement la condition de recouvrement (voir le théorème 5.3). Cette méthodologie nous permet de mettre en lumière un autre avantage de la CMGI qui consiste, par un choix approprié de la surface de glissement, en l'élimination de singularités de la commande nominale.

Pour chacune de ces commandes, nous avons présenté des résultats expérimentaux corroborant nos résultats théoriques.

Il convient de noter que les différents algorithmes proposés dans ce chapitre nécessitent la connaissance de la configuration des robots mobiles par rapport à un repère fixe. Dans le chapitre suivant, on relâche cette contrainte en tirant avantage de la possibilité d'une localisation relative entre les robots. L'idée sera de développer, dans le cadre du suivi de trajectoire, un mécanisme décentralisé de coordination entre les véhicules.

Chapitre 6

Suivi coordonné de trajectoire pour la flottille

6.1 Introduction

En ce qui concerne la fonctionnalité de poursuite de trajectoire pour une flottille de robots mobiles, plusieurs stratégies peuvent être mises en place. La plus simple est d'étendre l'architecture de poursuite de trajectoire mono-robot dans un cadre multi-robots. Dans ce cas, l'algorithme de commande proposé dans le chapitre précédent est implanté sur chaque robot mobile. Ainsi, seule la fonctionnalité de planification intègre des mécanismes de coordination entre robots.

L'idée de ce chapitre est de développer, dans le cadre du suivi de trajectoire, un mécanisme décentralisé de coordination de type “meneur / suiveur”, où une relation hiérarchique existe entre les véhicules. Pour chaque véhicule, la règle de décision est de suivre le véhicule le plus proche de lui (meneur local). Cette architecture est liée à la présence de capteurs (caméra omni-directionnelle, télémètres, ...) et d'algorithme de vision permettant de déterminer les positions relatives entre robots. Par rapport aux stratégies non coopératives de poursuite de trajectoire, elle présente l'avantage de s'affranchir de la connaissance de la position des robots par rapport à un repère fixe. Deux conséquences directes peuvent être énumérées :

- ★ des commandes peuvent facilement être construites de manière à assurer l'évitement de collisions malgré la présence d'erreurs et de perturbations en utilisant les coordonnées relatives entre robots,
- ★ dans le cadre du maintien d'une forme géométrique fixe pour la flottille (i.e. positions relatives inter-robots constantes), seul les robots meneurs de la flottille planifient leur trajectoire.

Deux points très importants sont à noter :

- Les stratégies coopératives de poursuite de trajectoire ne peuvent être utilisées pour l'ensemble de la flottille. En effet, pour l'un au moins des robots, elle doit être non coopérative afin de

pouvoir localiser la flottille par rapport à un repère fixe.

- Une attention particulière doit être portée aux propriétés de robustesse des commandes stabilisantes. En effet, une erreur sur l'un des robots se répercute automatiquement sur les autres.

Après avoir formulé le problème de stabilisation des erreurs de suivi de trajectoire dans le cadre d'un mécanisme de coordination "meneur / suiveur" pour une flottille de robots mobiles de type unicycle, nous proposerons une solution basée sur le principe de modes glissants avec action intégrale, qui permet également de garantir l'évitement de collisions entre robots malgré la présence de perturbations sur le modèle.

En guise de conclusion, nous illustrerons l'efficacité de notre algorithme par le biais de résultats expérimentaux sur la plate-forme de robots Miabot.

6.2 Formulation du problème de commande collaborative

6.2.1 Positionnement "meneur / suiveur"

Considérons une flottille de N_a robots mobiles de type unicycle, identifiés par l'indice $i \in \mathcal{N}$, dont la dynamique réelle est régie par les équations différentielles suivantes :

$$\begin{cases} \dot{q}_i &= f(q_i)u_i + p_i(q_i, t) \\ \dot{u}_i &= T_i + \bar{p}_i(q_i, u_i, t) \end{cases} \quad (6.2.1)$$

où

- $q_i = [x_i, y_i, \theta_i]^T \in \mathbb{R}^3$ est la configuration réelle du robot,
- $u_i = [v_i, w_i]^T \in \mathbb{R}^2$ est le vecteur vitesse du robot,
- $T_i = [T_{1,i}, T_{2,i}]^T \in \mathbb{R}^2$ est l'entrée de commande (des intégrateurs ont été ajoutés dans la chaîne des entrées de telle sorte que la partie discontinue de la commande agisse non pas directement sur les vitesses, mais sur leurs dérivées¹),

$$– f(q_i) = \begin{bmatrix} \cos \theta_i & 0 \\ \sin \theta_i & 0 \\ 0 & 1 \end{bmatrix},$$

- $p_i : \mathbb{R}^3 \times \mathbb{R}^+ \rightarrow \mathbb{R}^3$ et $\bar{p}_i : \mathbb{R}^3 \times \mathbb{R}^2 \times \mathbb{R}^+ \rightarrow \mathbb{R}^2$ représentent les perturbations sur le modèle, les incertitudes paramétriques et éventuellement les dynamiques non modélisées.

La stratégie de commande décentralisée est basée sur le modèle "meneur / suiveur" (voir figure 6.1). On suppose que chaque robot dispose d'une caméra positionnée devant celui-ci. Au début de la mission l'un des véhicules est choisi arbitrairement comme meneur global de la flottille (robot noté avec l'indice m). Les autres véhicules sont les suiveurs. Par conséquent, la configuration du meneur de la flottille est définie par rapport au repère fixe. Quant aux positions des suiveurs, elles sont définies

¹Il est raisonnable de penser que l'utilisation d'un modèle plus précis de la dynamique des actionneurs conduirait à de meilleurs résultats.

par rapport à la position courante du voisin immédiatement supérieur (le plus proche du meneur global).

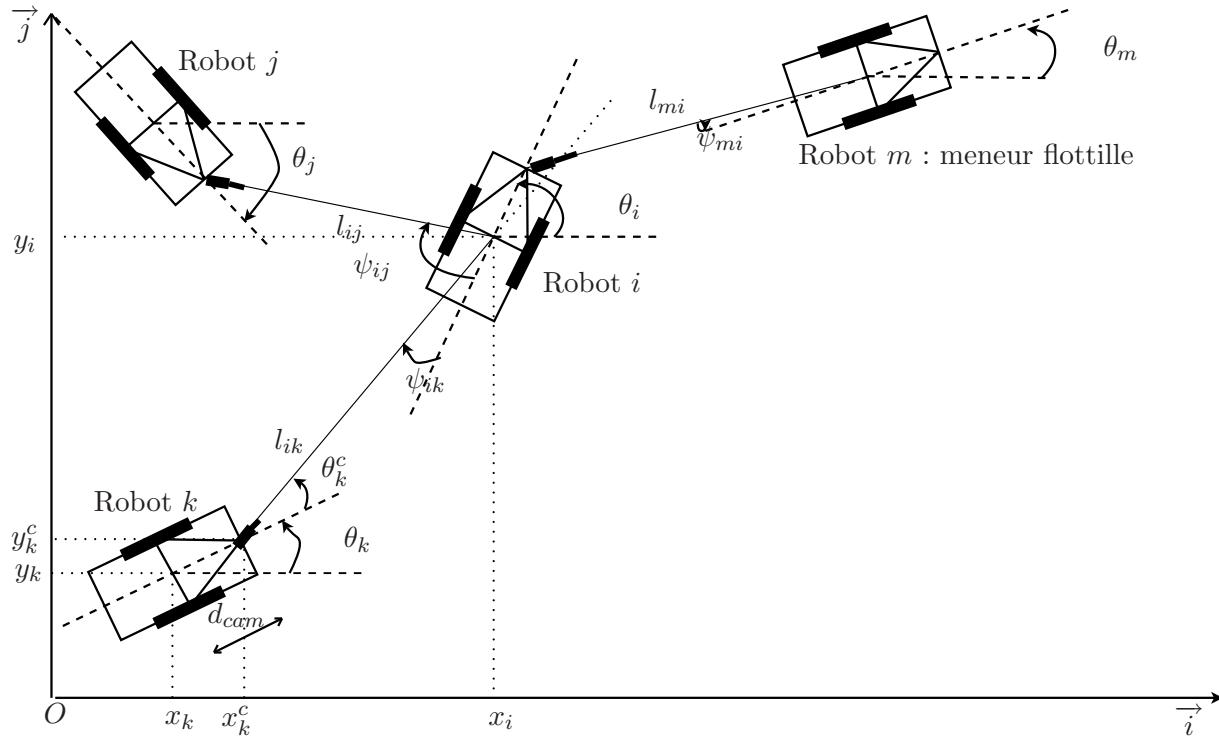


FIG. 6.1 – Exemple d'une stratégie de commande de type “meneur / suiveur”.

Soit la distance euclidienne $l_{ik}(t) \in \mathbb{R}^+$ et les angles $\psi_{ik}(t) \in (-\pi, \pi]$, $\theta_{ik}(t) \in (-\pi, \pi]$ ($\theta_{ik} = \theta_k^c - \psi_{ik}$) entre le robot meneur i et le robot suiveur k , représentés figure 6.1 et définis par les relations :

$$\begin{cases} x_i(t) - x_k^c(t) &= l_{ik}(t) \cos(\psi_{ik}(t) + \theta_i(t)) \\ y_i(t) - y_k^c(t) &= l_{ik}(t) \sin(\psi_{ik}(t) + \theta_i(t)) \\ \theta_{ik}(t) &= \theta_i(t) - \theta_k(t) \end{cases} \quad (6.2.2)$$

où les coordonnées de la caméra du robot k sont définies par :

$$\begin{cases} x_k^c(t) &= x_k(t) + d_{cam} \cos \theta_k(t) \\ y_k^c(t) &= y_k(t) + d_{cam} \sin \theta_k(t) \end{cases}$$

et $d_{cam} \in \mathbb{R}^+$ est la distance entre le centre de l'axe des roues motrices et la caméra du robot.

Remarque 6.1 Dans la suite de ce chapitre, on suppose que $d_{cam} > 0$. Néanmoins, il est possible de généraliser les résultats suivants dans le cas $d_{cam} = 0$ en utilisant comme nouvelle entrée de commande $[\dot{T}_{1,k}, T_{2,k}]^T$.

6.2.2 Objectif de commande

L'objectif général est de stabiliser les erreurs de suivi de trajectoire en utilisant les coordonnées relatives des différents robots. Ainsi, pour chaque couple $(i, k) \in \mathcal{N}^2$, $k \neq i$ de robots "meneur / suiveur", les positions relatives $h_{ik,des} = [l_{ik,des}, \psi_{ik,des}]^T$ désirées ainsi que leurs dérivées première $\dot{h}_{ik,des}$ et seconde $\ddot{h}_{ik,des}$ sont calculées en utilisant les trajectoires et les commandes de référence planifiées des robots i et k (cf. relations (6.2.2)).

Remarque 6.2 Pour le robot meneur global de la flottille, l'algorithme de poursuite de trajectoire est non coopératif (voir le chapitre précédent).

Le problème traité ici est donc, pour chaque robot suiveur $k \in \mathcal{N}$, $k \neq m$, de :

- * stabiliser asymptotiquement à l'origine l'erreur :

$$e_{1,k} = h_{ik} - h_{ik,des} \quad (6.2.3)$$

où $h_{ik} = [l_{ik}, \psi_{ik}]^T$ représente la configuration relative du robot k par rapport à son meneur i

- * maintenir les trajectoires $e_{1,k}(t)$, $t \geq 0$ dans l'ensemble non vide :

$$\Omega_k = \{e_{1,k}(t) \in \mathbb{R}^2 : |l_{ik}(t) - l_{ik,des}(t)| < \Lambda_k, |\psi_{ik}(t) - \psi_{ik,des}| < \Upsilon_k\} \quad (6.2.4)$$

où Λ_k et Υ_k sont des constantes strictement positives.

Remarque 6.3 Si la constante Λ_k vérifie la condition $\Lambda_k < l_{ik,des}(t) - (\rho_i + \rho_k) \forall t \geq 0$, alors l'évitement de collisions entre les robots i et k est garanti étant donné que $l_{ik}(t) > \rho_i + \rho_k (\forall t \geq 0)$.

Définissons l'erreur dans l'espace d'état :

$$e_k = [e_{1,k}^T, e_{2,k}^T]^T \quad (6.2.5)$$

où

$$e_{2,k}(t) = \dot{h}_{ik}(t) - \dot{h}_{ik,des}(t)$$

En dérivant les relations (6.2.2) et en utilisant les relations trigonométriques, on obtient :

$$\ddot{h}_{ik} = g_0(y_{ik})T_k + g_1(y_{ik}, u_k) + g_2(y_{ik})T_i + \pi_{ik}(y_{ik}, u_i, u_k) \quad (6.2.6)$$

avec

$$g_0(y_{ik}) = \begin{bmatrix} -\cos(\psi_{ik} + \theta_{ik}) & -d_{cam} \sin(\psi_{ik} + \theta_{ik}) \\ \frac{\sin(\psi_{ik} + \theta_{ik})}{l_{ik}} & -\frac{d_{cam} \cos(\psi_{ik} + \theta_{ik})}{l_{ik}} \end{bmatrix} \quad g_2(y_{ik}) = \begin{bmatrix} \cos(\psi_{ik}) & 0 \\ \frac{-\sin(\psi_{ik})}{l_{ik}} & -1 \end{bmatrix}$$

$$g_1(y_{ik}, u_k) = \begin{bmatrix} v_k \dot{\theta}_{ik} \sin(\psi_{ik} + \theta_{ik}) - d_{cam} w_k \dot{\theta}_{ik} \cos(\psi_{ik} + \theta_{ik}) + l_{ik} \dot{\psi}_{ik} (w_k + \dot{\psi}_{ik} + \dot{\theta}_{ik}) \\ \frac{1}{l_{ik}} (v_k \dot{\theta}_{ik} \cos(\psi_{ik} + \theta_{ik}) + d_{cam} w_k \dot{\theta}_{ik} \sin(\psi_{ik} + \theta_{ik}) - l_{ik} (w_k + 2\dot{\psi}_{ik} + \dot{\theta}_{ik})) \end{bmatrix}$$

et

$$\pi_{ik} = \begin{bmatrix} \pi_{1,ik} \\ \pi_{2,ik} \end{bmatrix} = \begin{bmatrix} \frac{d}{dt} \left(\frac{\partial l_{ik}}{\partial q_i} p_i + \frac{\partial l_{ik}}{\partial q_k} p_k \right) + \frac{\partial}{\partial q_i} \left(\frac{\partial l_{ik}}{\partial q_i} f(q_i) u_i + \frac{\partial l_{ik}}{\partial q_k} f(q_k) u_k \right) p_i \\ + \frac{\partial}{\partial q_k} \left(\frac{\partial l_{ik}}{\partial q_i} f(q_i) u_i + \frac{\partial l_{ik}}{\partial q_k} f(q_k) u_k \right) p_k + \frac{\partial l_{ik}}{\partial q_i} f(q_i) \bar{p}_i + \frac{\partial l_{ik}}{\partial q_k} f(q_k) \bar{p}_k \\ \frac{d}{dt} \left(\frac{\partial \psi_{ik}}{\partial q_i} p_i + \frac{\partial \psi_{ik}}{\partial q_k} p_k \right) + \frac{\partial}{\partial q_i} \left(\frac{\partial \psi_{ik}}{\partial q_i} f(q_i) u_i + \frac{\partial \psi_{ik}}{\partial q_k} f(q_k) u_k \right) p_i \\ + \frac{\partial}{\partial q_k} \left(\frac{\partial \psi_{ik}}{\partial q_i} f(q_i) u_i + \frac{\partial \psi_{ik}}{\partial q_k} f(q_k) u_k \right) p_k + \frac{\partial \psi_{ik}}{\partial q_i} f(q_i) \bar{p}_i + \frac{\partial \psi_{ik}}{\partial q_k} f(q_k) \bar{p}_k \end{bmatrix}$$

où $y_{ik} = [l_{ik}, \psi_{ik}, \theta_{ik}]^T$ est le vecteur de mesures représentant la configuration relative entre les robots i et k .

En utilisant le retour d'état suivant :

$$T_k = \begin{bmatrix} -\cos(\psi_{ik} + \theta_{ik}) & l_{ik} \sin(\psi_{ik} + \theta_{ik}) \\ -\frac{1}{d_{cam}} \sin(\psi_{ik} + \theta_{ik}) & -\frac{l_{ik}}{d_{cam}} \cos(\psi_{ik} + \theta_{ik}) \end{bmatrix} \bar{T}_k \quad (6.2.7)$$

où \bar{T}_k est la nouvelle commande, le système (6.2.6) se met sous la forme :

$$\ddot{h}_{ik} = \bar{T}_k + g_1(y_{ik}, u_k) + g_2(y_{ik}) T_i + \pi_{ik}(y_{ik}, u_i, u_k) \quad (6.2.8)$$

Remarque 6.4 Il est important de noter que le retour d'état (6.2.7) est bien défini, sauf en $l_{ik} = 0$ qui correspond au cas où les robots i et k sont en collision.

Par conséquent, la dynamique de l'erreur peut s'exprimer de la manière suivante :

$$\dot{e}_k = \begin{bmatrix} \dot{e}_{1,k} \\ \dot{e}_{2,k} \end{bmatrix} = \begin{bmatrix} e_{2,k} \\ \bar{T}_k + g_1(y_{ik}, u_k) + g_2(y_{ik}) T_i + \pi_{ik}(y_{ik}, u_i, u_k) - \ddot{h}_{ik,des} \end{bmatrix} \quad (6.2.9)$$

Hypothèse 6.1 On suppose qu'à l'instant initial, l'erreur $e_{1,k}(0)$ est dans l'ensemble Ω_k , c'est-à-dire :

$$e_{1,k}(0) \in \Omega_k \quad (6.2.10)$$

6.3 Algorithmes coordonnés de suivi de trajectoire

Dans cette section, on considère le problème de poursuite de trajectoire pour le robot k appartenant à la paire “meneur / suiveur” (i, k) . En fonction des informations transmises par le meneur i , différentes commandes peuvent être synthétisées. Ici, deux architectures sont considérées afin de stabiliser le système (6.2.9).

- La première, décrite par la figure 6.2, est basée sur le principe de CMGI d'ordre un et nécessite la connaissance des vitesses u_i ainsi que des accélérations T_i du robot meneur i , de la configuration relative h_{ik} du robot k par rapport au robot i et de sa dérivée \dot{h}_{ik} . En pratique, cet algorithme est délicat à mettre en œuvre du fait de la quantité d'informations échangées entre les robots.

- La deuxième est basée sur le principe de CMGI d'ordre deux. Son utilisation diminue considérablement les ressources en matière de communication. En effet, seule la connaissance de la configuration relative h_{ik} du robot k par rapport au robot i est nécessaire. Dans ce cas, comme le montre la figure 6.3, la vitesse et l'accélération du robot i sont considérées comme des perturbations. Ceci rend l'implémentation plus facile.

Ces deux stratégies supposent que l'hypothèse suivante soit vérifiée :

Hypothèse 6.2 *L'erreur de suivi de la trajectoire planifiée pour le robot i (i.e. $q_i(t) - q_{i,des}(t)$) converge asymptotiquement vers zéro.*

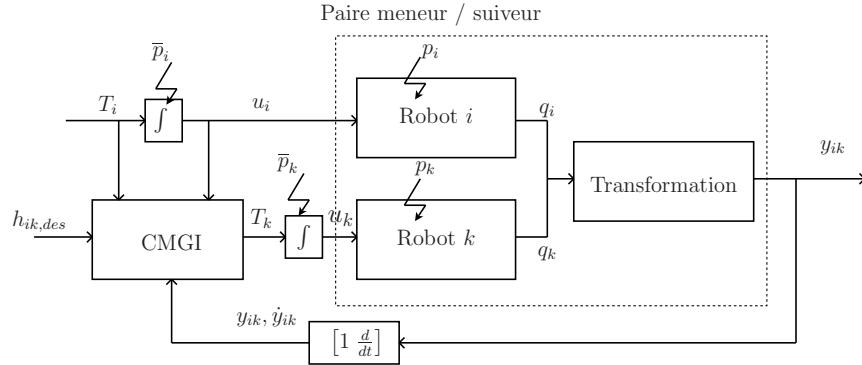


FIG. 6.2 – Architecture de CMGI d'ordre un pour la paire “meneur / suiveur”.

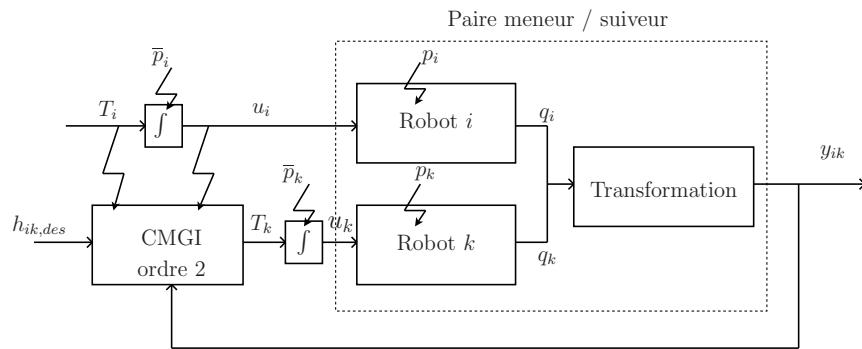


FIG. 6.3 – Architecture de CMGI d'ordre deux pour la paire “meneur / suiveur”.

6.3.1 CMGI d'ordre un

Afin d'indiquer les ressources matérielles du robot k indispensables pour la mise en place de la CMGI d'ordre 1, l'hypothèse suivante doit être vérifiée :

Hypothèse 6.3 *Le robot k dispose :*

- *d'un dispositif de localisation (caméra, télémètres, ...) permettant de calculer la configuration y_{ik} relative par rapport au robot i ainsi que sa dérivée \dot{y}_{ik} ,*

- de capteurs de vitesse afin de calculer sa vitesse u_k ,
- d'une antenne réceptrice afin de recevoir du robot i , sa vitesse u_i ainsi que son accélération T_i .

Remarque 6.5 Il est possible de calculer la dérivée \dot{y}_{ik} de la configuration relative par le biais d'un différentiateur robuste (pour plus de renseignements sur cette technique, on peut se référer à [Levant, 1998]) ou par le biais de techniques algébriques (voir [INRIA, 2007]).

Soit la variable de glissement $s_k = [s_{1,k}, s_{2,k}]^T \in \mathbb{R}^2$ suivante :

$$s_k(t) = e_{2,k}(t) + M_{1,k}e_{1,k}(t) + M_{2,k}e_{aux,k}(t) \quad (6.3.1)$$

avec $M_{1,k} = \begin{bmatrix} m_{1,k} & 0 \\ 0 & m_{3,k} \end{bmatrix}$ et $M_{2,k} = \begin{bmatrix} m_{2,k} & 0 \\ 0 & m_{4,k} \end{bmatrix}$. Les constantes $m_{1,k}$, $m_{2,k}$, $m_{3,k}$ et $m_{4,k}$, strictement positives, seront définies par la suite. La variable auxiliaire $e_{aux,k}(t) \in \mathbb{R}^2$ est régie par l'équation différentielle :

$$\dot{e}_{aux,k}(t) = e_{1,k}(t) \quad (6.3.2)$$

La condition initiale $e_{aux,k}(0)$ est déterminée à partir de la relation $s_k(0) = 0$. Par conséquent, afin d'éliminer la phase de convergence, la variable auxiliaire $e_{aux,k}$ est initialisée à la valeur :

$$e_{aux,k}(0) = -M_{2,k}^{-1}(e_{2,k}(0) + M_{1,k}e_{1,k}(0))$$

Théorème 6.1 [Defoort et al., 2006a] Supposons que les hypothèses 6.1, 6.2 et 6.3 soient vérifiées et que les constantes $m_{1,k}$, $m_{2,k}$, $m_{3,k}$, $m_{4,k}$, strictement positives, sont choisies de telle sorte que l'ensemble Ω_k soit positivement invariant pour le système non perturbé :

$$\dot{e}_k = \begin{bmatrix} 0_2 & I_2 \\ -M_{2,k} & -M_{1,k} \end{bmatrix} e_k \quad (6.3.3)$$

Alors, le système (6.2.9) est asymptotiquement stable et l'évitement de collisions entre les robots i et k est assuré depuis l'instant initial (i.e. $\forall t \geq 0$, $e_{1,k}(t) \in \Omega_k$) sous la commande :

$$\bar{T}_k = \ddot{h}_{ik,des} - g_1(y_{ik}, u_k) - g_2(y_{ik})T_i + \bar{T}_{k,nom} + \bar{T}_{k,disc} \quad (6.3.4)$$

avec

$$\begin{aligned} \bar{T}_{k,nom} &= -M_{1,k}e_{2,k} - M_{2,k}e_{1,k} \\ \bar{T}_{k,disc} &= -G_k(y_{ik}) \text{sign}(s_k) \end{aligned} \quad (6.3.5)$$

et où le gain $G_k(y_{ik})$ vérifie la relation :

$$G_k(y_{ik}) \geq \|\pi_{ik}\| + \eta \quad (6.3.6)$$

avec $\eta > 0$.

Remarque 6.6 Les coefficients $m_{1,k}$, $m_{2,k}$, $m_{3,k}$ et $m_{4,k}$ fixent naturellement les performances d'amortissement et de rapidité pour la stabilisation du système (6.2.9).

Démonstration. Considérons la fonction candidate de Lyapunov :

$$V = \frac{1}{2} s_k^T s_k$$

La dérivée de V suivant les trajectoires du système (6.2.9) s'écrit :

$$\begin{aligned}\dot{V} &= s_k^T (\bar{T}_{k,nom} + \bar{T}_{k,disc} + \pi_{ik}(y_{ik}, u_i, u_k) + M_{1,k} e_{2,k} + M_{2,k} e_{1,k}) \\ &= s_k^T (-G_k(y_{ik}) \operatorname{sign}(s_k) + \pi_{ik}(y_{ik}, u_i, u_k))\end{aligned}$$

En utilisant la relation (6.3.6), on en déduit que :

$$\begin{aligned}\dot{V} &\leq -\eta (|s_{1,k}| + |s_{2,k}|) \\ &\leq -\eta \sqrt{2} \sqrt{V}\end{aligned}$$

Ceci garantit l'attractivité et la stabilité en temps fini de la surface de glissement $\{s_k = 0\}$. En outre, puisque $s_k(0) = 0$, les trajectoires d'état du système évoluent sur la surface de glissement dès l'instant initial. Par conséquent, la phase de convergence est éliminée. La commande équivalente de $\bar{T}_{k,disc}$, notée $\bar{T}_{k,disc}^{eq}$ et obtenue par la relation $\dot{s}_k = 0$, est donnée par :

$$\bar{T}_{k,disc}^{eq} = -\pi_{ik}(y_{ik}, u_i, u_k)$$

Les dynamiques du système (6.2.9) en régime glissant sont donc régies par les équations différentielles :

$$\dot{e}_k = \begin{bmatrix} 0_2 & I_2 \\ -M_{2,k} & -M_{1,k} \end{bmatrix} e_k \quad (6.3.7)$$

Puisque les constantes $m_{1,k}$, $m_{2,k}$, $m_{3,k}$, $m_{4,k}$ sont choisies telle que l'ensemble Ω_k soit positivement invariant pour le système non perturbé (6.3.7), l'évitement de collisions entre les robots i et k est assuré depuis l'instant initial (i.e. $\forall t \geq 0$, $e_{1,k}(t) \in \Omega_k$). ■

6.3.2 CMGI d'ordre deux

Dans la partie précédente, une CMGI d'ordre un a permis la stabilisation des erreurs de suivi e_k malgré la présence de perturbations dans le modèle. Cependant, la nécessité de la connaissance de u_i , T_i et u_k restreint considérablement son application. Afin de diminuer la quantité d'informations échangées entre les robots, une CMGI d'ordre deux est implémentée par la suite. Cela permet d'éliminer certains capteurs notamment le capteur d'accélération et de construire une commande à partir uniquement des configurations y_{ik} relatives entre les robots.

Dans cette partie, l'hypothèse 6.3 indiquant les ressources matérielles requises pour le robot k est remplacée par l'hypothèse suivante :

Hypothèse 6.4 Le robot k dispose seulement d'un dispositif de localisation (caméra, télémètres, ...) permettant de calculer sa configuration relative par rapport au robot i , i.e. y_{ik} .

Etant donné que l'on veut s'affranchir de la connaissance de \dot{y}_{ik} , la variable de glissement (6.3.1) ne peut plus être choisie. Afin de pallier à ce problème, on utilise une CMGI d'ordre deux.

Hypothèse 6.5 La condition initiale $e_{2,k}(0)$ est supposée connue.

Remarque 6.7 L'hypothèse 6.5 n'est pas restrictive. En effet, deux cas de figure peuvent se présenter :

- les vitesses initiales des robots meneur et suiveur sont nulles (ce qui est généralement le cas en pratique). Par conséquent, on a $e_{2,k}(0) = 0$.
- les vitesses initiales des robots meneur et suiveur ne sont pas nulles. Dans ce cas, il est possible, avant d'appliquer la CMGI d'ordre deux, d'obtenir les conditions initiales $\dot{l}_{ik}(0)$ et $\dot{\psi}_{ik}(0)$ à l'aide d'un différentiateur robuste ou par le biais de techniques algébriques (voir [INRIA, 2007]).

Soit la variable de glissement $\bar{s}_k = [\bar{s}_{1,k}, \bar{s}_{2,k}]^T \in \mathbb{R}^2$ définie par :

$$\bar{s}_k(t) = e_{1,k}(t) + M_{1,k}e_{aux_1,k}(t) + M_{2,k}e_{aux_2,k}(t) \quad (6.3.8)$$

avec $M_{1,k} = \begin{bmatrix} m_{1,k} & 0 \\ 0 & m_{3,k} \end{bmatrix}$ et $M_{2,k} = \begin{bmatrix} m_{2,k} & 0 \\ 0 & m_{4,k} \end{bmatrix}$. Les constantes $m_{1,k}$, $m_{2,k}$, $m_{3,k}$ et $m_{4,k}$, strictement positives, sont définies de la même manière que dans la partie précédente. Les variables auxiliaires $e_{aux_1,k}(t) \in \mathbb{R}^2$ et $e_{aux_2,k}(t) \in \mathbb{R}^2$ sont régies par les équations différentielles :

$$\begin{cases} \dot{e}_{aux_1,k}(t) = e_{1,k}(t) \\ \dot{e}_{aux_2,k}(t) = e_{aux_1,k}(t) \end{cases} \quad (6.3.9)$$

Les conditions initiales $e_{aux_1,k}(0)$ et $e_{aux_2,k}(0)$ sont déterminées à partir des relations $\bar{s}_k(0) = 0$ et $\dot{\bar{s}}_k(0) = 0$. Par conséquent, afin d'éliminer la phase de convergence, elles sont initialisées aux valeurs :

$$\begin{cases} e_{aux_2,k}(0) = M_{2,k}^{-1}(-e_{2,k}(0) - M_{1,k}e_{1,k}(0)) \\ e_{aux_1,k}(0) = M_{2,k}^{-1}\left(-e_{1,k}(0) - M_{1,k}M_{2,k}^{-1}(-e_{2,k}(0) - M_{1,k}e_{1,k}(0))\right) \end{cases}$$

Avant de synthétiser la commande, il est nécessaire de vérifier l'hypothèse suivante :

Hypothèse 6.6 Il existe une constante positive C_k telle que, dans l'ensemble Ω_k , l'inégalité suivante soit vérifiée :

$$\|g_1(y_{ik}, u_k) + g_2(y_{ik})T_i + \pi_{ik}(y_{ik}, u_i, u_k)\| \leq C_k \quad (6.3.10)$$

Théorème 6.2 Supposons que les hypothèses 6.1, 6.2, 6.4, 6.5 et 6.6 soient vérifiées et que les constantes $m_{1,k}$, $m_{2,k}$, $m_{3,k}$, $m_{4,k}$, strictement positives, sont choisies de telle sorte que l'ensemble Ω_k est positivement invariant pour le système non perturbé :

$$\dot{e}_k = \begin{bmatrix} 0_2 & I_2 \\ -M_{2,k} & -M_{1,k} \end{bmatrix} e_k \quad (6.3.11)$$

Alors, le système (6.2.9) est asymptotiquement stable et l'évitement de collisions entre les robots i et k est assuré depuis l'instant initial (i.e. $\forall t \geq 0$, $e_{1,k} \in \Omega_k$) sous la commande :

$$\ddot{T}_k = \ddot{h}_{ik,des} + \bar{T}_{k,nom} + \bar{T}_{k,disc} \quad (6.3.12)$$

où

$$\bar{T}_{k,nom} = -M_{2,k}e_{1,k}$$

et

$$\bar{T}_{k,disc} = \begin{bmatrix} \begin{cases} -\lambda_{m,k} \operatorname{sign}(\bar{s}_{1,k}) & \text{si } \bar{s}_{1,k} \Delta_{\bar{s}_{1,k}} \leq 0 \\ -\lambda_{M,k} \operatorname{sign}(\bar{s}_{1,k}) & \text{si } \bar{s}_{1,k} \Delta_{\bar{s}_{1,k}} > 0 \end{cases} \\ \begin{cases} -\lambda_{m,k} \operatorname{sign}(\bar{s}_{2,k}) & \text{si } \bar{s}_{2,k} \Delta_{\bar{s}_{2,k}} \leq 0 \\ -\lambda_{M,k} \operatorname{sign}(\bar{s}_{2,k}) & \text{si } \bar{s}_{2,k} \Delta_{\bar{s}_{2,k}} > 0 \end{cases} \end{bmatrix}$$

$\Delta_{\bar{s}_{1,k}}$ (respectivement $\Delta_{\bar{s}_{2,k}}$) représente le signe de la dérivée de $\bar{s}_{1,k}$ (respectivement $\bar{s}_{2,k}$). Cette dérivée est obtenue à l'aide d'un différentiateur robuste d'ordre un ([Levant, 1998]). Les gains $\lambda_{M,k}$ et $\lambda_{m,k}$ vérifient les relations :

$$\lambda_{m,k} > C_k \quad , \quad \lambda_{M,k} > \lambda_{m,k} + 2C_k .$$

La démonstration de ce théorème est largement inspirée de la démonstration du théorème 6.1. Ainsi, seules les principales lignes directrices sont données.

Démonstration. En utilisant les résultats issus de [Levant, 1993], il peut être montré que, sous la commande (6.3.12), les trajectoires du système (6.2.9) convergent en temps fini sur la surface de glissement $\{\bar{s}_k = \dot{s}_k = 0\}$. En outre, puisque $\bar{s}_k(0) = \dot{s}_k(0) = 0$, la phase de convergence est éliminée. Les dynamiques du système (6.2.9) en régime glissant sont :

$$\dot{e}_k = \begin{bmatrix} 0_2 & I_2 \\ -M_{2,k} & -M_{1,k} \end{bmatrix} e_k \quad (6.3.13)$$

Par conséquent, on obtient les propriétés suivantes :

- * stabilisation asymptotique à l'origine de l'erreur $e_{1,k}$,
- * maintien des trajectoires $e_{1,k}(t)$, $\forall t \geq 0$ dans l'ensemble Ω_k .

6.4 Extension à une architecture à deux meneurs

Dans la section précédente, la règle de décision pour chaque véhicule était de suivre le robot le plus proche de lui immédiatement supérieur (meneur local). D'autres règles existent pour réaliser le mécanisme de coordination par "leadership" (voir par exemple [Tanner et al., 2004]), notamment celle qui consiste à suivre non plus un robot mais les deux robots, appelés meneurs locaux, les plus proches, immédiatement supérieurs.

Dans ce cas, l'objectif de commande devient, pour chaque robot suiveur $k \in \mathcal{N}$ de :

- ★ stabiliser asymptotiquement à l'origine l'erreur :

$$e_{1,k} = \bar{h}_{ik} - \bar{h}_{ik,des}$$

où $\bar{h}_{ik} = [l_{ik}, l_{jk}]^T$ représente la configuration relative du robot k par rapport à ses meneurs i et j et $\bar{h}_{ik,des} = [l_{ik,des}, l_{jk,des}]^T$ la configuration relative désirée (voir figure 6.4),

- ★ maintenir les trajectoires $e_{1,k}(t)$, $t \geq 0$ dans l'ensemble non vide :

$$\bar{\Omega}_k = \{e_{1,k}(t) \in \mathbb{R}^2 : |l_{ik}(t) - l_{ik,des}(t)| < \bar{\Lambda}_k, |l_{jk}(t) - l_{jk,des}(t)| < \bar{\Upsilon}_k\}$$

où $\bar{\Lambda}_k$ et $\bar{\Upsilon}_k$ sont des constantes strictement positives.

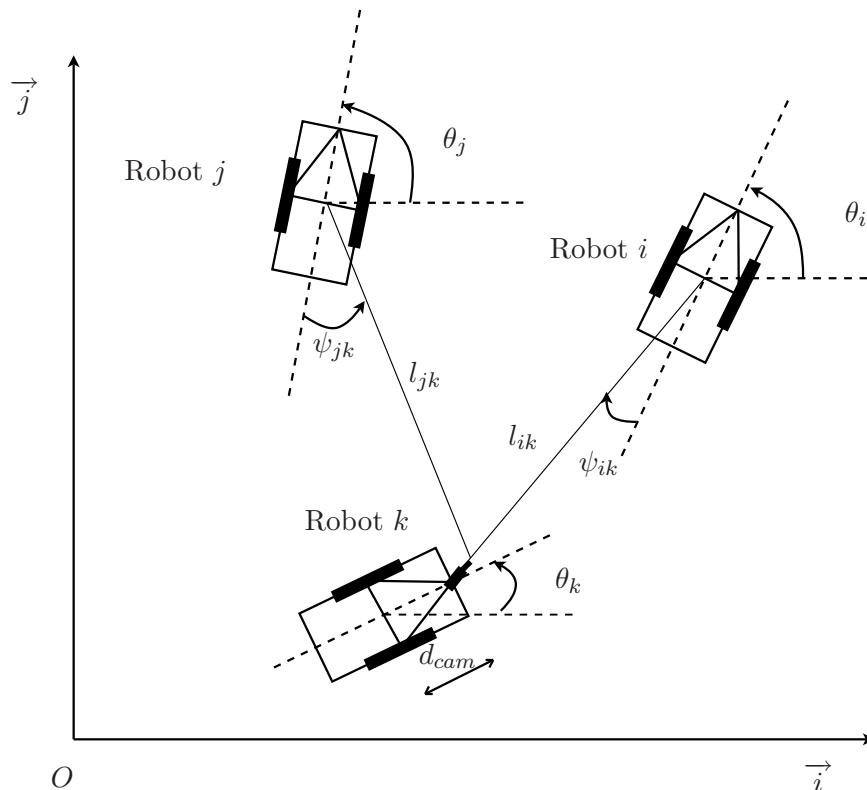


FIG. 6.4 – Représentation géométrique d'une triplet (i, j, k) avec (i, j) les meneurs et k le suiveur.

De la même manière que dans la partie précédente, il est possible de définir l'erreur dans l'espace d'état :

$$e_k = [e_{1,k}^T, e_{2,k}^T]^T$$

où

$$e_{2,k} = \dot{h}_{ik} - \dot{h}_{ik,des}$$

En utilisant le retour d'état suivant :

$$T_k = \frac{1}{\sin(\psi_{jk} + \theta_{jk} - \psi_{ik} - \theta_{ik})} \begin{bmatrix} -\sin(\psi_{jk} + \theta_{jk}) & \sin(\psi_{ik} + \theta_{ik}) \\ \frac{1}{d_{cam}} \cos(\psi_{jk} + \theta_{jk}) & -\frac{1}{d_{cam}} \cos(\psi_{ik} + \theta_{ik}) \end{bmatrix} \bar{T}_k \quad (6.4.1)$$

où \bar{T}_k est la nouvelle commande, on obtient le système :

$$\ddot{h}_{ik} = \bar{T}_k + \bar{g}_1(y_{ik}, y_{jk}, u_k) + \bar{g}_{21}(y_{ik})T_i + \bar{g}_{22}(y_{jk})T_j + \bar{\pi}_{ijk}(y_{ik}, y_{jk}, u_i, u_j, u_k)$$

avec

$$\begin{aligned}\bar{g}_1(y_{ik}, y_{jk}, u_k) &= \begin{bmatrix} v_k \dot{\theta}_{ik} \sin(\psi_{ik} + \theta_{ik}) - d_{cam} w_k \dot{\theta}_{ik} \cos(\psi_{ik} + \theta_{ik}) + l_{ik} \dot{\psi}_{ik} (w_k + \dot{\psi}_{ik} + \dot{\theta}_{ik}) \\ v_k \dot{\theta}_{jk} \sin(\psi_{jk} + \theta_{jk}) - d_{cam} w_k \dot{\theta}_{jk} \cos(\psi_{jk} + \theta_{jk}) + l_{jk} \dot{\psi}_{jk} (w_k + \dot{\psi}_{jk} + \dot{\theta}_{jk}) \end{bmatrix} \\ \bar{g}_{21}(y_{ik}) &= \begin{bmatrix} \cos(\psi_{ik}) & 0 \\ 0 & 0 \end{bmatrix} \quad \bar{g}_{22}(y_{jk}) = \begin{bmatrix} 0 & 0 \\ \cos(\psi_{jk}) & 0 \end{bmatrix} \quad \bar{\pi}_{ijk}(y_{ik}, y_{jk}, u_i, u_j, u_k) = \begin{bmatrix} \pi_{1,ik}(y_{ik}, u_i, u_k) \\ \pi_{1,jk}(y_{jk}, u_j, u_k) \end{bmatrix}\end{aligned}$$

où $y_{ik} = [l_{ik}, \psi_{ik}, \theta_{ik}]^T$ (respectivement $y_{jk} = [l_{jk}, \psi_{jk}, \theta_{jk}]^T$) est le vecteur de mesures représentant la configuration relative entre les robots i (respectivement j) et k .

Remarque 6.8 Le retour d'état (6.4.1) est bien défini, sauf en $\psi_{jk} + \theta_{jk} = \psi_{ik} + \theta_{ik}$ qui correspond au cas où les trois robots sont alignés. Par conséquent, à partir du dispositif de vision, il est impossible de détecter les deux robots meneurs puisque l'un des robots meneur obstrue le champ de vision du robot suiveur.

La dynamique de l'erreur peut s'exprimer de la manière suivante :

$$\dot{e}_k = \begin{bmatrix} e_{2,k} \\ \bar{T}_k + \bar{g}_1(y_{ik}, y_{jk}, u_k) + \bar{g}_{21}(y_{ik})T_i + \bar{g}_{22}(y_{jk})T_j + \bar{\pi}_{ijk}(y_{ik}, y_{jk}, u_i, u_j, u_k) - \ddot{h}_{ik,des} \end{bmatrix} \quad (6.4.2)$$

Etant donné les similitudes des équations (6.2.9) et (6.4.2) et de la formulation des problèmes de poursuite de trajectoire pour la paire (i, k) et pour le triplet (i, j, k) , les mêmes algorithmes de commande (voir les théorèmes 6.1 et 6.2) peuvent être utilisés pour le triplet “2 meneurs / suiveur”.

6.5 Résultats expérimentaux

Pour illustrer les algorithmes de commande proposés dans les théorèmes 6.1 et 6.2, on effectue des tests sur la plate-forme de trois robots Miabot.

6.5.1 Description de la plate-forme Miabot

Une vue globale de la plate-forme est donnée figure 6.5. Celle-ci est constituée de :

- trois robots Miabot. Le robot Miabot est un robot mobile de type unicycle constitué de deux roues motrices qui ne peuvent pas braquer et qui sont commandées par deux moteurs indépendants. Sa forme est un cube de 7cm de côté. Il est équipé d'un processeur Atmel ATMega64 (64Kb Flash) et d'un module de communication Bluetooth (11.5Kb/sec). Aucun capteur n'est embarqué sur le robot. Deux pastilles de couleur permettent de les localiser et les identifier.

- une aire de jeu de $3m \times 2m$,
- un système de vision centralisé. Une caméra vidéo est placée au dessus de l'aire de jeu et scrute l'ensemble du terrain.
- un ordinateur distant qui supervise l'ensemble du système. A partir des données issues de la caméra, il détermine la position et l'orientation des trois robots. Puis, il calcule les commandes à appliquer et les envoie ensuite au robot correspondant.



FIG. 6.5 – Plate-forme Miabot.



FIG. 6.6 – Robot Miabot.

Remarque 6.9 *Dans nos expérimentations, on reconstruit les coordonnées relatives entre les robots à partir de leurs coordonnées absolues en utilisant les relations (6.2.2) de manière à simuler une architecture décentralisée dans laquelle chaque robot disposerait d'une caméra devant lui ($d = 5cm$).*

6.5.2 Résultats expérimentaux

Afin de comparer les algorithmes de commande donnés dans les théorèmes 6.1 et 6.2, on considère un scénario assez simple avec un seul robot meneur et deux robots suiveurs. On se place dans le cadre du maintien d'une forme géométrique fixe pour la flottille (i.e. positions relatives inter-robots constantes). Cette forme est caractérisée par les grandeurs suivantes : $h_{12,des} = [20cm, \frac{\pi}{4}]^T$ et $h_{13,des} = [20cm, -\frac{\pi}{4}]^T$. La trajectoire de référence du robot meneur d'indice 1 est composé de deux lignes droites parcourues à la vitesse de $0.5m/s$ et d'un arc de cercle. Elle a été planifiée de manière à ne pas sortir des

limites du terrain. L'algorithme de suivi de trajectoire pour ce robot est celui défini dans le théorème 5.1.

Remarque 6.10 Notons qu'ici l'objectif est de présenter une application réelle basique qui illustre uniquement les performances et les limites des algorithmes développés dans les théorèmes 6.1 et 6.2. En effet, il serait plus robuste de générer les trajectoires optimales sous contraintes entre la position courante du suiveur et sa position désirée (approche développée dans le chapitre 3) tout en utilisant la CMGI pour le maintien robuste sur cette trajectoire.

Initialement, les robots sont positionnés aux configurations $q_1(0) = [0m, 0m, 0rad]^T$, $q_2(0) = [-0.3m, 0m, -0.04rad]^T$ et $q_3(0) = [-0.4m, 0m, 0.04rad]^T$ et ont des vitesses nulles. La période d'échantillonnage est $\tau = 0.06s$ (grandeur limitée par les caractéristiques de la caméra).

L'un des objectifs est donc de passer d'une formation de type "ligne" à une formation de type "triangulaire".

Scénario 1 : Connaissance de la vitesse du robot meneur

Lorsque l'on suppose que la vitesse du robot meneur est connue, il est possible d'appliquer le théorème 6.1. Les paramètres de la variable de glissement (6.3.1), choisis de manière à vérifier les hypothèses du théorème 6.1 sont $M_{1,2} = M_{1,3} = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$ et $M_{2,2} = M_{2,3} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$. Les gains de la CMGI d'ordre un des robots 2 et 3 sont $G_2 = G_3 = 3$. Les figures 6.7-6.8 donnent les résultats associés à l'algorithme 6.1. On peut remarquer que les erreurs de suivi convergent exponentiellement vers zéro malgré les perturbations, les incertitudes et les erreurs de mesure. Sur la figure 6.7, il est possible de voir le passage d'une formation "ligne" à une formation "triangulaire". Sur la figure 6.8, on observe que lorsque le robot meneur effectue le demi-tour, le robot 2 accélère (vitesse allant jusqu'à $1m/s$) alors que le robot 3 ralentit de manière à maintenir une formation "triangulaire". Il faut noter que la trajectoire planifiée du robot meneur a été réalisée en prenant en compte les limites physiques des robots afin d'éviter les saturations sur les vitesses des robots suiveurs.

Scénario 2 : Non connaissance de la vitesse du robot meneur

Lorsque la vitesse du robot meneur n'est pas connue, il n'est plus possible d'appliquer l'algorithme de commande donné dans le théorème 6.1. Par conséquent, afin de synthétiser la commande, nous devons utiliser le théorème 6.2. Les paramètres de la variable de glissement (6.3.8), choisis de manière à vérifier les hypothèses du théorème 6.2 sont $M_{1,2} = M_{1,3} = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$ et $M_{2,2} = M_{2,3} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$. Les gains de la CMGI d'ordre 2 des robots 2 et 3 sont $\lambda_{M,2} = \lambda_{M,3} = 12$ et $\lambda_{m,2} = \lambda_{m,3} = 4$. Les figures 6.9-6.10 donnent les résultats associés à l'algorithme 6.2. On peut remarquer que les erreurs de suivi convergent toujours vers zéro. Néanmoins, étant donné que l'on dispose de moins d'informations

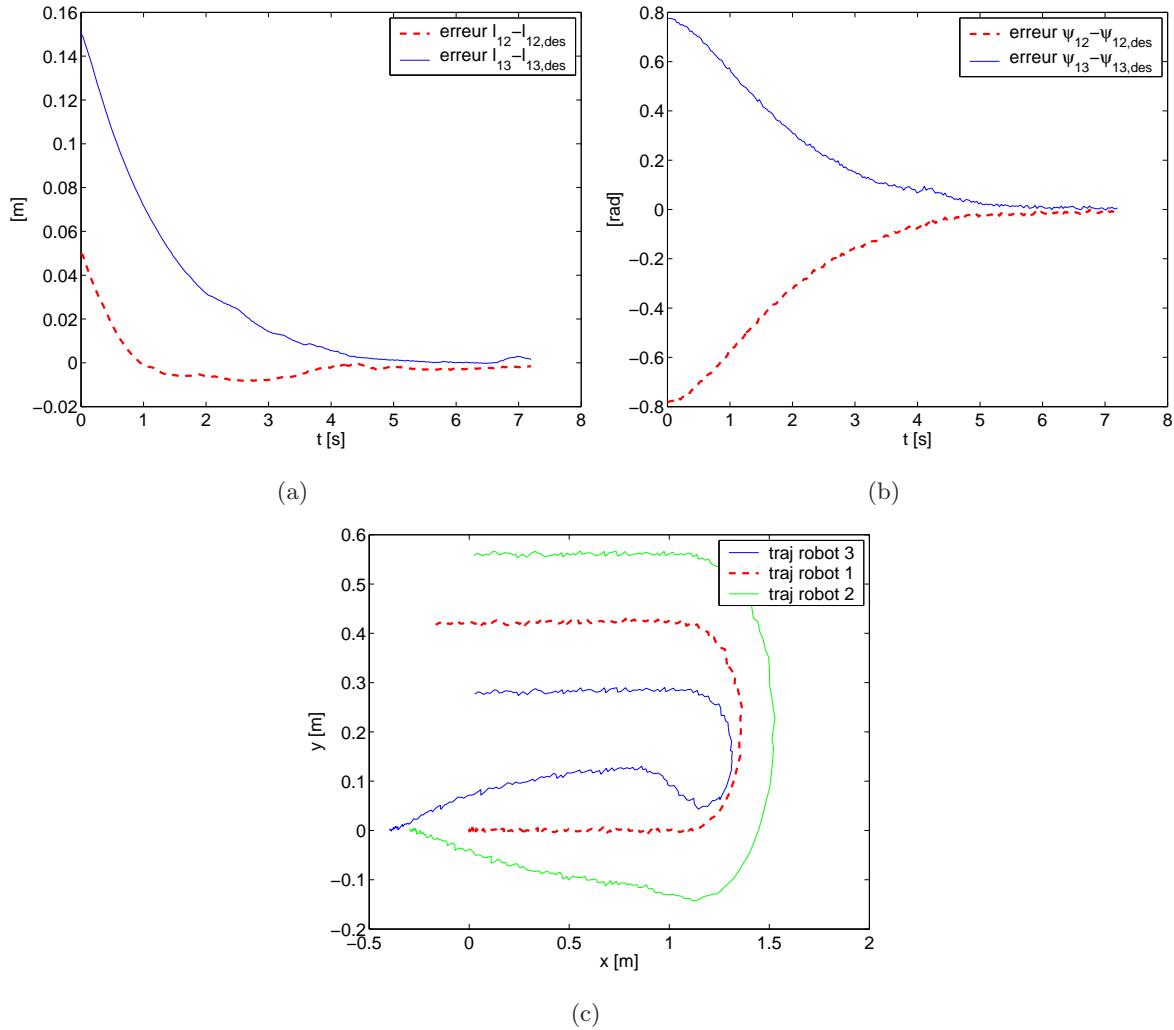


FIG. 6.7 – Evolution des erreurs de suivi de trajectoire en appliquant la CMGI d’ordre 1.

pour construire la commande et que la période d’échantillonnage τ est assez grande, les performances sont légèrement dégradées.

Remarque 6.11 *Les vidéos associées aux résultats expérimentaux obtenus sur la plate-forme Miabot sont disponibles sur le site :*

<http://syner.ec-lille.fr/robocoop/course/view.php?id=14>

6.6 Conclusion

Dans ce chapitre, nous avons développé, dans le cadre du suivi de trajectoire, des mécanismes décentralisés de coordination de type “meneur / suiveur”, basés sur la CMGI. Ils permettent non seulement de s’affranchir de la connaissance de la position de l’ensemble des robots par rapport à un

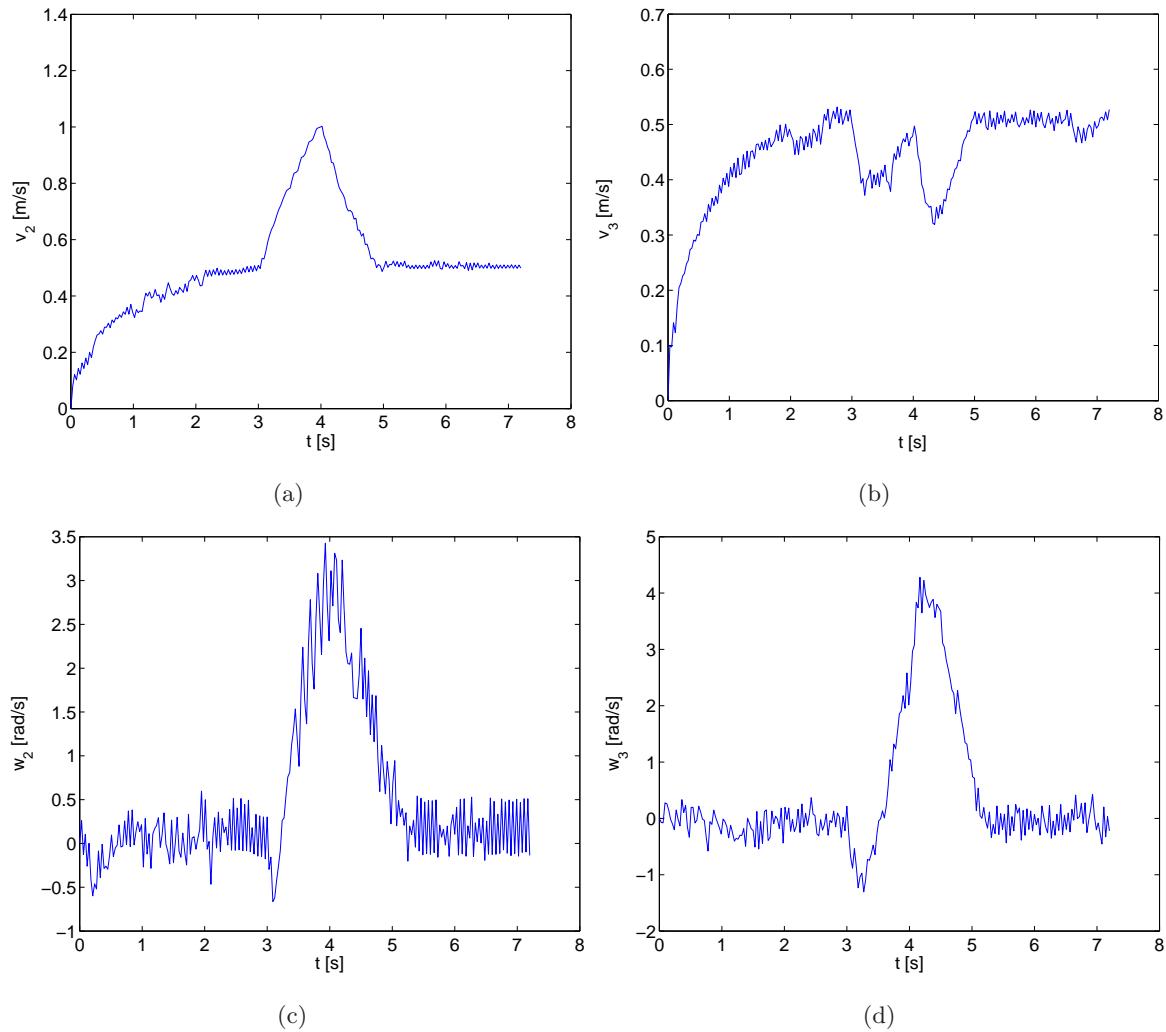


FIG. 6.8 – Vitesses linéaires et angulaires des robots 2 et 3 en appliquant la CMGI d’ordre 1.

repère fixe, mais aussi d’assurer l’évitement de collisions inter-robots malgré la présence de perturbations sur le modèle en utilisant les coordonnées relatives entre robots.

En plus des propriétés de robustesse vis-à-vis des perturbations, l’utilisation d’une CMGI d’ordre deux permet de s’affranchir de la connaissance des vitesses des robots meneurs.

Des résultats expérimentaux ont validé les commandes proposées et ont montré leur efficacité sur une flotte de trois robots mobiles.

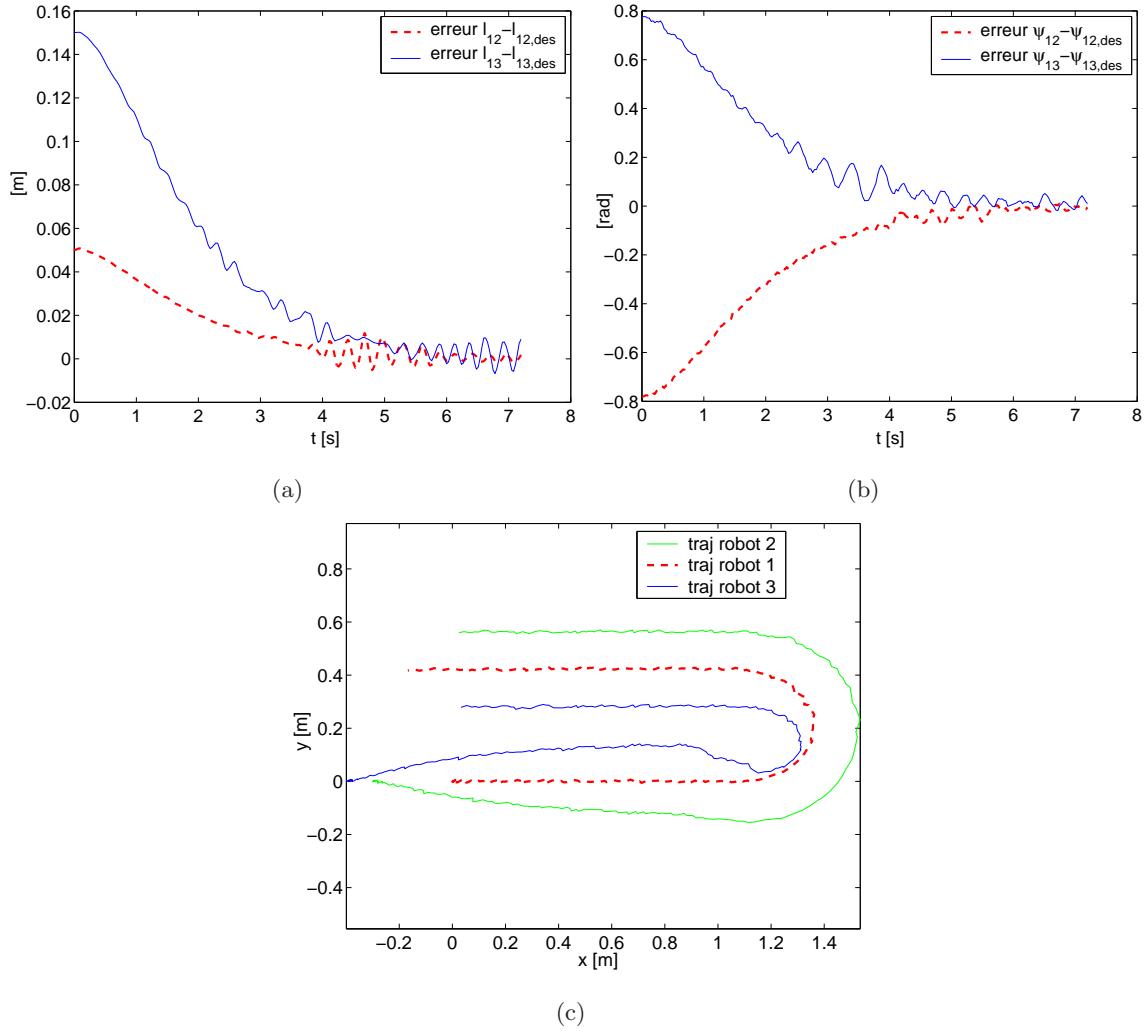


FIG. 6.9 – Evolution des erreurs de suivi de trajectoire en appliquant la CMGI d'ordre 2.

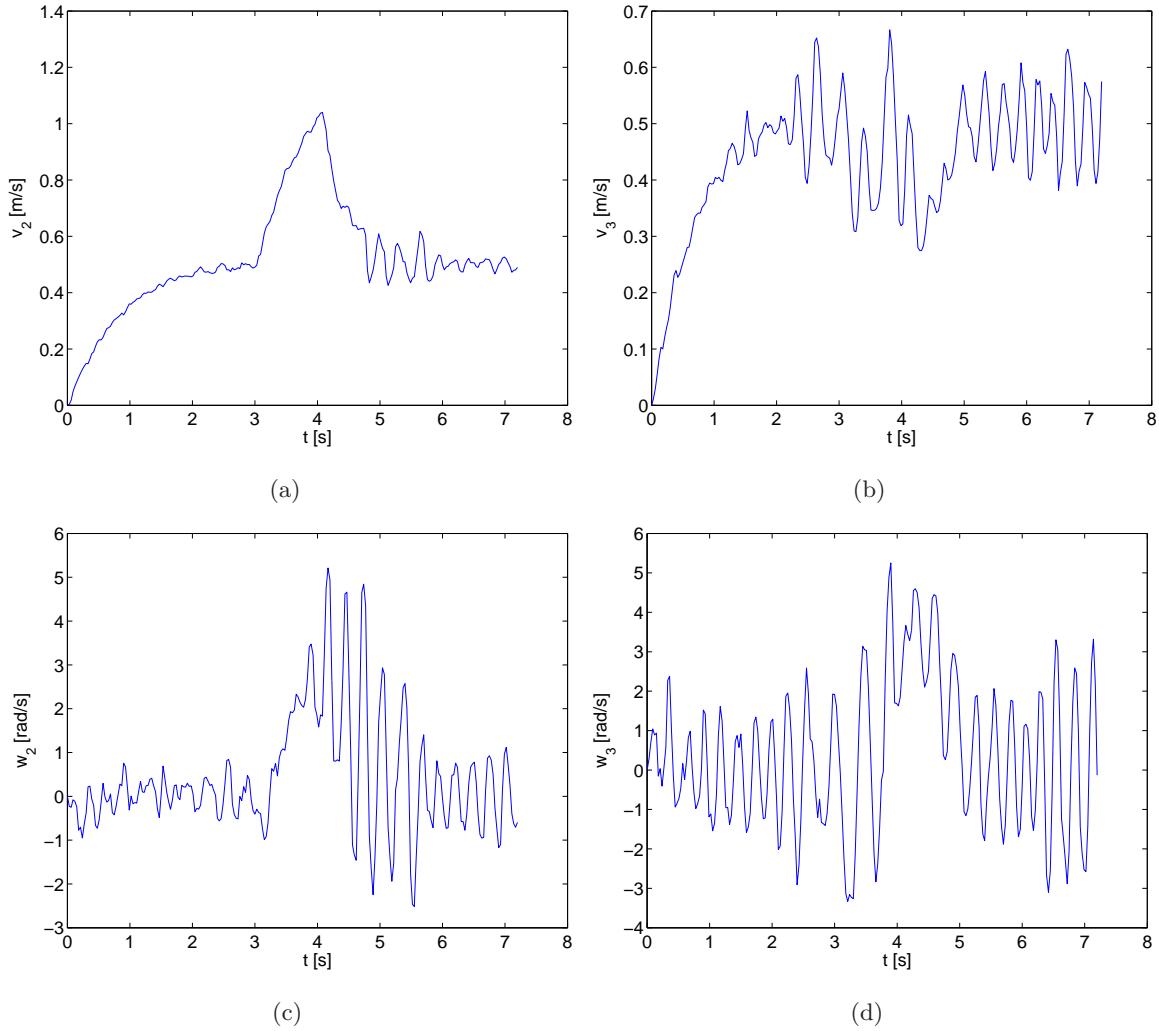


FIG. 6.10 – Vitesses linéaires et angulaires des robots 2 et 3 en appliquant la CMGI d'ordre 2.

Conclusion générale et perspectives

L'ensemble de ce mémoire est dévolu au problème de la navigation autonome des robots mobiles dans un cadre particulier. Ce cadre est défini par les caractéristiques des systèmes considérés et des applications envisagées, que nous pouvons résumer par la navigation pour une flottille de robots mobiles non holonomes.

Retour sur les contributions

Le problème que nous avons considéré est très ambitieux, puisqu'il s'agit de doter un système multi-robots à la fois d'une architecture de planification de trajectoire puissante et flexible et d'une architecture de poursuite de trajectoire performante et robuste.

- Dans le premier chapitre, nous avons présenté les bases théoriques relatives aux systèmes non holonomes. Nous avons notamment donné un rapide tour d'horizon sur les algorithmes répondant aux questions de la planification de trajectoire et de la poursuite de trajectoire dans les cadres mono et multi-robots et sur leurs inconvénients.
- Le chapitre 2 a été consacré au développement d'un algorithme de planification de trajectoire admissible pour un robot mobile suffisamment flexible pour pouvoir être étendu au cadre multi-robots. L'approche proposée basée sur la platitude, les fonctions splines et l'optimisation sous contraintes, a été validée expérimentalement sur le robot Pekee.
- Le chapitre 3 nous a plongé dans le vif du sujet. Dans ce chapitre, le problème de planification de plusieurs robots mobiles coopératifs a été traité. Nous y avons présenté deux mécanismes de coordination selon la méthode de résolution des conflits via un superviseur (approche centralisée) ou individuellement par chaque robot (approche décentralisée). Le premier est une extension directe du cas d'un seul robot. Le second est un algorithme décentralisé de planification en ligne, basé uniquement sur les informations locales disponibles. Il consiste à décomposer le problème de planification en deux étapes. D'abord, chaque robot construit une trajectoire intuitive. Puis, une fois les trajectoires intuitives échangées entre les robots en conflit, chaque véhicule ajuste

sa trajectoire de manière à éviter les collisions et les ruptures de communication. Ces deux algorithmes ont été testés et comparés à d'autres méthodes existantes. Nous avons ainsi mis en évidence les avantages et les limites des différentes techniques en nous appuyant sur différents critères :

- ◊ le temps de calcul,
- ◊ les ressources en communication,
- ◊ la mise en œuvre (nombre de paramètres à régler, facilité de réglage ...).

- Après s'être concentré sur le problème de planification de trajectoire, dans le chapitre 4, nous avons présenté les concepts de base de la commande par modes glissants. Nous y avons rappelé les définitions et les théorèmes usuels — notamment ceux qui concernent la commande par modes glissants avec action intégrale — qui sont utiles pour les chapitres suivants. Puis, nous avons proposé un algorithme original de commande par modes glissants d'ordre quelconque qui ne nécessite pas de phase d'initialisation et qui garantit un réglage simple et rapide des paramètres de la loi de commande. Cet algorithme consiste à ajouter un terme discontinu afin de rendre robuste vis-à-vis des perturbations une commande nominale basée sur des fonctions homogènes. Enfin, nous avons mis expérimentalement en évidence son efficacité à travers la commande d'un moteur pas-à-pas.
- Dans le chapitre 5, nous avons synthétisé et implémenté sur le robot Pekee deux algorithmes de commande qui garantissent la stabilisation et/ou le suivi de trajectoire malgré la présence d'incertitudes et de perturbations. La première méthode a permis de mettre en lumière un des rôles de la CMGI qui consiste en l'amélioration des résultats obtenus à partir d'une commande nominale. La seconde technique a mis en lumière un autre avantage de la CMGI : l'élimination de singularités dans la loi de commande nominale. En outre, il a été montré qu'en relâchant l'objectif de stabilisation asymptotique en un objectif de stabilisation pratique, cette technique permet d'apporter une solution unifiée à de nombreux problèmes de poursuite de trajectoires (e.g. configuration fixe, trajectoires non stationnaires) en présence de perturbations ne vérifiant pas nécessairement la condition de recouvrement.
- Dans le chapitre 6, nous avons développé dans le cadre du suivi de trajectoire un mécanisme décentralisé de coordination de type "meneur / suiveur" basé sur la CMGI. Il a permis non seulement de s'affranchir de la connaissance de la position de l'ensemble des robots par rapport à un repère fixe, mais aussi d'assurer l'évitement de collisions inter-robots malgré la présence de perturbations sur le modèle. Des résultats expérimentaux sur une flottille de trois robots Miabot ont montré son efficacité.

Problèmes ouverts

A l'issue de ce travail de thèse, plusieurs problèmes demeurent toutefois en suspens et d'autres méthodes restent à développer. Nous présentons ici ce qui nous semble être le cadre d'investigations où des avancées importantes sont tout à fait envisageables.

Les concepts théoriques introduits dans cette thèse peuvent conduire à plusieurs extensions ou applications futures. En effet, un très grand nombre de problèmes en ingénierie (robotique, systèmes électriques, problèmes de routage et de congestion, ...) fait appel aux techniques de commande robuste ainsi qu'aux techniques de planification du fait notamment de la possibilité et de la nécessité d'inclure des contraintes génériques et des objectifs de performance.

Dans ce mémoire, nous nous sommes intéressés au problème de planification de trajectoire admissible et sans collision. Nous avons donc privilégié le respect des contraintes par rapport à l'atteinte de l'objectif désiré. Il serait intéressant d'étudier les propriétés de convergence de l'algorithme de planification en ligne de trajectoire (dans le sens de convergence vers la configuration finale désirée en évitant les minima locaux). L'idée serait de donner des conditions suffisantes sur le choix des différents horizons (horizon de planification, horizon de calculs, horizon de détection). Ce problème est en grande partie lié à la modélisation de l'environnement et au choix du critère à optimiser. Il est résolu dans certains cas (par exemple, pour un système non linéaire avec un critère quadratique en l'absence de contraintes, des conditions de convergence existent [Jadbabaie et al., 2001]). Dans le cas général, ce problème reste ouvert et très difficile à résoudre.

Dans le chapitre 2, nous avons pu mettre en évidence l'intérêt de la transformation d'un problème de commande optimale sous la forme d'un problème variationnel à l'aide des propriétés de platitude du système. Dans ce mémoire, nous avons choisi de spécifier les sorties plates par des fonctions splines. Il serait intéressant de décomposer les sorties plates dans une autre base afin d'améliorer les performances de l'algorithme. Récemment, une spécification par une fonction NURBS ("Non-Uniform Rational B-splines"), qui est une représentation par fractions rationnelles par morceaux des sorties plates, a été proposée dans [Flores et Milam, 2006]. Son intérêt est de résoudre des problèmes complexes (passage étroit, ...). Le prix à payer est un nombre plus important de paramètres à régler.

En outre, l'utilisation des fonctions splines dans la spécification des sorties plates reste à approfondir notamment au niveau du placement des noeuds. L'objectif serait à terme de respecter une tolérance donnée sans pénaliser le temps de calcul. Des solutions ont été apportées pour les systèmes linéaires quadratiques [Auquiert et al., 2006]. Le problème général reste ouvert.

Au niveau du chapitre 4, il serait intéressant de développer des différentiateurs robustes, à convergence en temps fini, basés sur des conditions constructives de convergence et à réglage aisé. En effet, l'un des problèmes majeurs pour l'implémentation de notre algorithme de commande par modes glissants d'ordre supérieur est l'obligation de connaître des dérivées d'ordre supérieur de la variable de

glissement. Pour le moteur pas-à-pas, nous avons proposé un différentiateur robuste d'ordre trois. Il serait intéressant de généraliser ces résultats et démontrer la stabilité en boucle fermée de l'association commande / observateur.

Dans le chapitre 5, nous avons développé et implémenté des algorithmes permettant la stabilisation asymptotique ou pratique des erreurs de suivi de trajectoire. Il serait intéressant de synthétiser une loi de commande robuste garantissant la stabilisation en temps fini des erreurs de suivi. Ceci permettrait d'assurer l'évitement de collision avec les obstacles malgré la présence de perturbation et d'incertitudes dans le modèle.

Dans le chapitre 6, un point à approfondir est la reconstruction de la vitesse des robots meneurs afin d'améliorer la connaissance du véhicule sur son environnement. L'idée serait de mettre en place des observateurs, éventuellement basés sur des techniques algébriques, capables d'estimer la vitesse des autres robots. A partir de cette estimation, il est possible d'améliorer les algorithmes de commande par modes glissants avec action intégrale proposés. Le problème est de concevoir un observateur suffisamment performant et rapide pour pouvoir être appliqué en temps réel.

Enfin, au niveau applicatif, l'implémentation de l'algorithme décentralisé de planification de trajectoire sur une flotte de trois robots Pekee et de trois robots Miabot est en cours. A plus long terme, cette étude ouvre la porte à d'autres perspectives. L'une d'entre elles est l'application de l'algorithme décentralisé de planification de trajectoire dans le cadre d'un déplacement urbain autonome (voir le projet [GrayMatter, 2007]). De petites modifications de notre algorithme sont nécessaires afin de prendre en compte les obstacles mobiles. En effet, dans ce cas, les robots n'échangent plus d'informations sur leurs intentions. Par conséquent, il est nécessaire de mettre en place un observateur qui puisse prédire la trajectoire des véhicules risquant de provoquer une collision. A partir de cette trajectoire intuitive, on peut élaborer un algorithme similaire à celui proposé dans le chapitre 3.

Annexes

Annexe A

Le robot Pekee

A.1 Présentation du robot

Le robot Pekee est un robot construit par la société Wany Robotics. Il dispose de cinq emplacements pour des cartes filles (trois directement accessibles, les deux autres sont à l'intérieur) et d'un mécanisme de déplacement constitué de trois roues (deux roues motrices à l'avant commandées par deux moteurs indépendants et une roue folle à l'arrière pour permettre au robot d'être stable). Sa longueur est de 40cm , sa largeur de 25.5cm et son poids de 3.3kg . Ce robot est capable de mouvements et de comportements autonomes selon l'environnement qu'il identifie grâce à ses différents capteurs commandés par le microcontrôleur. Les capteurs intégrés comprennent : 15 télémètres infra-rouges (10 mesures par seconde), 2 odomètres (180 impulsions par tour de roue), 2 gyromètres (axes lacet et tangage), 2 capteurs de température interne et externe, un capteur de lumière et un détecteur de choc à variation de seuil. La caméra vidéo, quant à elle, est gérée par le PC embarqué.

Ce robot peut être commandé de deux façons différentes :

- soit par des programmes stockés dans la mémoire (Compact Flash) du PC embarqué : dans ce cas, le programme ne doit pas consommer trop de ressources (afin de ne pas poser de problème de dysfonctionnement au niveau du système d'exploitation embarqué). Ainsi, seul les algorithmes de suivi de trajectoire seront implantés dans cette mémoire.
- soit par un programme distant dans la mesure où la carte PC embarquée peut être vue comme un “pont” entre le microcontrôleur et le PC de bureau. Dans ce cas, nous avons à disposition une liaison WiFi permettant non seulement le transfert des données entre le robot et un PC distant mais également le transfert d'informations entre différents robots. Du fait de la faible puissance du PC embarqué, les algorithmes de planification de trajectoire et de vision sont déportés sur un PC de bureau (Pentium IV, 2.4GHz).

La figure A.1 illustre l'architecture de navigation pour le robot Pekee.

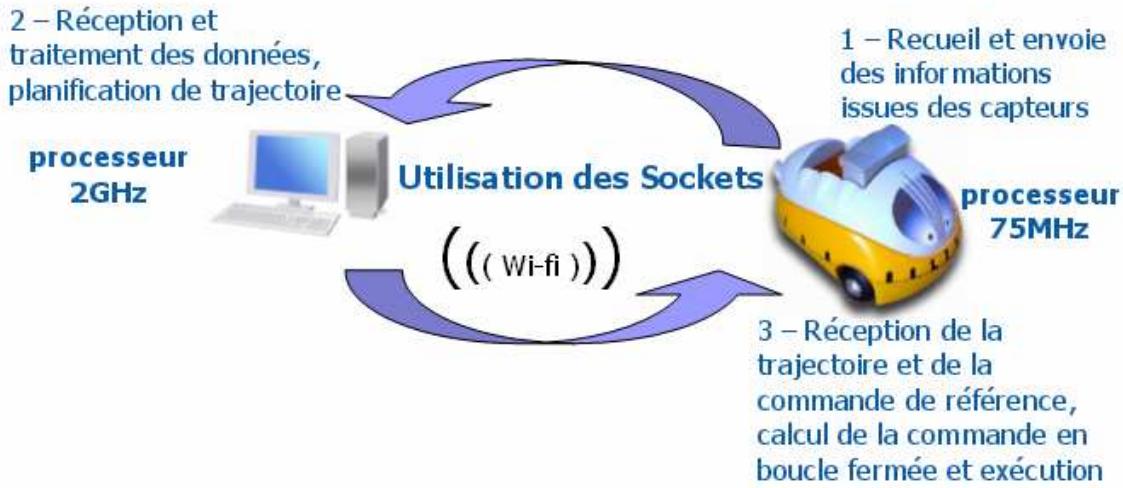


FIG. A.1 – Architecture de navigation pour le robot Pekee

A.2 Communication entre les différents modules du robot

D'une part, la communication entre les modules du robot est assurée par le PC embarqué, équipé d'un microprocesseur Intel 486, fonctionnant sous système d'exploitation (OS) Linux. D'autre part, elle est effectuée par un bus OPP : Ordinateur Parallèle et Polymorphique (protocole de communication, propriétaire à Wany Robotics). Il assure la gestion des communications entre cartes filles (PC embarqué, module Wifi,...) et microcontrôleur.

L'horloge principale permettant de cadencer cette communication est fixée à 63 ms : toute commande moteur (par exemple) envoyée entre deux tops d'horloge est ignorée. D'autres limitations d'interaction avec le robot sont liées aux capteurs infra-rouges qui peuvent introduire un retard de 400ms. En effet, on accorde seulement 30% de crédibilité à la mesure qui vient d'être faite. Il faut que celle-ci se confirme sur plusieurs échantillons pour devenir crédible : il s'agit d'un filtre à réponse impulsionale infinie (autrement dit une moyenne glissante) qui est implémenté sur la partie de la carte microcontrôleur du robot.

Nos expériences ont démontré que Windows98, initialement installé sur le PC embarqué du robot Pekee, est inadapté pour la mise en place des commandes pour le suivi de trajectoire. En effet, Windows n'est pas adapté à des applications en temps réel, et ceci pour une raison assez simple à comprendre : il nous est impossible de gérer les priorités d'exécution des tâches que nous lui demandons.

La solution envisagée fut de remplacer l'OS existant par celui de "Linux temps réel" sur le PC embarqué du robot Pekee.

A.3 Système d'exploitation temps réel

La gestion du temps constitue un problème majeur des systèmes d'exploitation car les OS modernes sont tous multitâches, mais, ils utilisent cependant le matériel basé sur des processeurs quant à eux qui ne le sont pas.

Par conséquent, l'OS doit partager le temps du processeur entre les différentes tâches. Cette notion de partage implique une gestion de passage d'une tâche à l'autre effectuée par un ensemble d'algorithmes appelés ordonnanceurs. Le rôle de l'ordonnanceur étant de partager le temps processeur entre les applications, ceci implique que le temps de réponse d'un système classique n'est pas garanti.

Le cas des systèmes temps réels est différent. Un système temps réel est une association logiciel-matériel où le logiciel permet, entre autres, une gestion adéquate des ressources matérielles en vue de remplir certaines tâches ou fonctions dans des limites temporelles bien précises.

A.4 Linux comme système temps réel

Linux n'est pas initialement un système temps réel. Le noyau Linux a été conçu dans le but d'en faire un système généraliste donc basé sur la notion de temps partagé et non de temps réel. Dans le but d'améliorer le comportement du noyau afin qu'il soit compatible avec les contraintes d'un système temps réel comme décrit précédemment, plusieurs solutions techniques sont disponibles. Celles-ci sont divisées en deux familles :

1. Le noyau temps réel auxiliaire.

Les promoteurs de cette technologie considèrent que le noyau Linux ne sera jamais véritablement temps réel et ajoute donc à ce dernier un véritable ordonnanceur temps réel à priorités fixes. Ce noyau auxiliaire traite directement les tâches temps réel et délègue les autres processus au noyau Linux, considéré comme la tâche de fond de plus faible priorité. Cette technique permet de mettre en place des systèmes temps réel "durs" (systèmes pour lesquels une gestion stricte du temps est nécessaire pour conserver l'intégrité du service rendu). Elle est utilisée par le système RTLinux et son cousin européen RTAI. Néanmoins, l'utilisation des systèmes temps réel "durs" peut poser quelques problèmes du point de vue de la compatibilité matérielle, car les pilotes de périphériques adaptés ne sont généralement pas fournis par le constructeur (en raison d'une trop faible diffusion). Pour la même raison, les interfaces de programmation d'applications et d'autres outils de développement sont souvent très sophistiqués ou simplement non disponibles.

2. Les patchs dits "préemptifs".

Ils permettent d'améliorer le comportement du noyau Linux en réduisant les temps de latence de ce dernier. Autrement dit, ces patchs corrigent la préemption du noyau, en réduisant le temps qui s'écoule entre le moment où une tâche est choisie pour être exécutée par l'ordonnanceur et le moment où elle est réellement exécutée. Ces modifications ne transforment pas Linux en noyau

temps réel “dur” mais permettent d’obtenir des résultats satisfaisants dans le cas de contraintes temps réel dites “molles” (systèmes acceptant des variations dans le traitement des données de l’ordre de la demi-seconde). Par contre, Linux avec patch temps réel conservera ses outils de développement classique. En effet, ces patchs constituent une solution intermédiaire entre un noyau standard, utilisant un ordonnanceur à temps partagé et un véritable noyau temps réel. En conclusion, l’utilisation d’un noyau modifié ne transforme pas linux en temps réel, mais permet d’améliorer le temps de latence dans les limites d’une utilisation acceptable pour une application donnée, en conservant une interface de programmation classique pour un utilisateur.

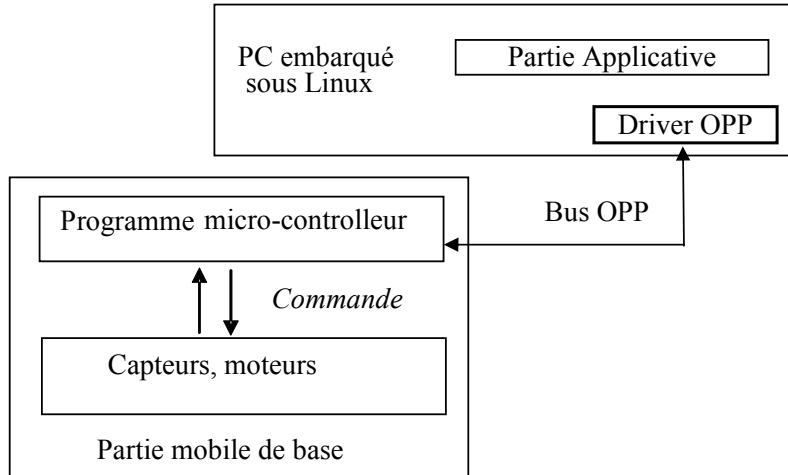


FIG. A.2 – Schéma fonctionnel du robot Pekee

A.5 Linux temps réel sur la carte compact-flash de Pekee

Les contraintes concernant le temps de communication avec Pekee par le bus OPP, ainsi que les inconvénients liés à l’OS temps réel “dur”, nous ont amenés à envisager l’utilisation d’un noyau avec un patch.

Pour le portage de Pekee sous Linux, nous avons fait le choix du noyau : linux-2.4.25 associé au patch de type “préemptif” (hrt : low-latency + preemptible). Il s’agit de la combinaison de deux patchs :

1. *preempt-kernel* permettant d’ajouter dans le noyau des points de préemption systématiques
2. *low-latency* tendant à réduire de longs intervalles de temps sans appel à l’ordonnanceur.

En installant ce nouvel OS sur la carte compact-flash de Pekee, nous avons mis en place un environnement de développement croisé (cross-compilateur) sur l’ordinateur hôte. Un cross-compilateur est comme un compilateur qui prend en entrée un code source et produit en sortie un binaire exécutable.

La seule différence, par rapport au compilateur classique, est que le binaire résultant (cross-compilé sur une machine-hôte) ne peut pas être exécuté sur la même machine, mais sur une machine-cible. Par exemple, sur une machine avec un microprocesseur Intel Pentium M 735, on peut cross-compiler des exécutables destinés à un microprocesseur 486 à 75Mhz de l'OS embarqué sur Pekee.

L'opération de construction d'un environnement croisé nous a permis de mettre en place, sur la machine hôte, un environnement de développement d'applications embarquées rendant possible l'utilisation d'un outil de développement évolué tel que "KDevelop".

Annexe B

Classification des véhicules à roues

La *posture* d'un véhicule est la position d'un repère lié à celui-ci, dans un repère lié à l'environnement. C'est un vecteur non-homogène, dont les deux premières composantes définissent la position du repère robot dans le repère fixe et la dernière l'orientation du repère robot par rapport au repère fixe, i.e. $\zeta = [x, y, \theta]^T$.

Les hypothèses utilisées pour l'écriture du modèle cinématique sont :

- le contact entre la roue et le sol est ponctuel,
- le sol est plan,
- les roues et le véhicule sont indéformables (le robot n'est pas équipé de suspension),
- l'axe de direction des roues est perpendiculaire au sol,
- l'axe de traction des roues est parallèle au sol.

B.1 Classification et description des roues

La configuration d'une roue et le nombre de paramètres nécessaires à sa description dépend du type de roue considérée. Nous introduisons ici les quatres types de roues principalement utilisés en robotique mobile :

- les *roues fixes* dont l'axe de rotation, parallèle au sol, passe par le centre de la roue,
- les *roues centrées orientables* dont l'axe d'orientation, perpendiculaire au sol, passe par le centre de la roue,
- les *roues décentrées orientables* dont l'axe d'orientation, perpendiculaire au sol, ne passe pas par le centre de la roue,
- les *roues suédoises* pour lesquelles la composante nulle de la vitesse de glissement du point de contact n'est pas dans le plan de la roue.

Ces différents types de roues sont présentés sur la figure B.1 alors que leur paramétrage respectif est schématisé sur la figure B.2.

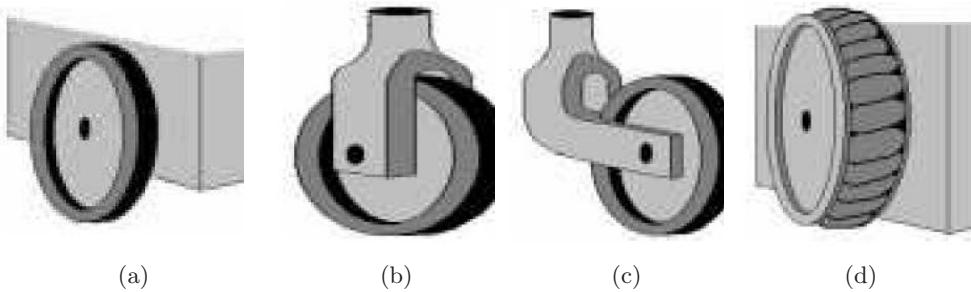


FIG. B.1 – Différents types de roues utilisées pour les plates-formes mobiles. (a) Roue fixe. (b) Roue orientable centrée. (c) Roue orientable décentrée. (d) Roue suédoise

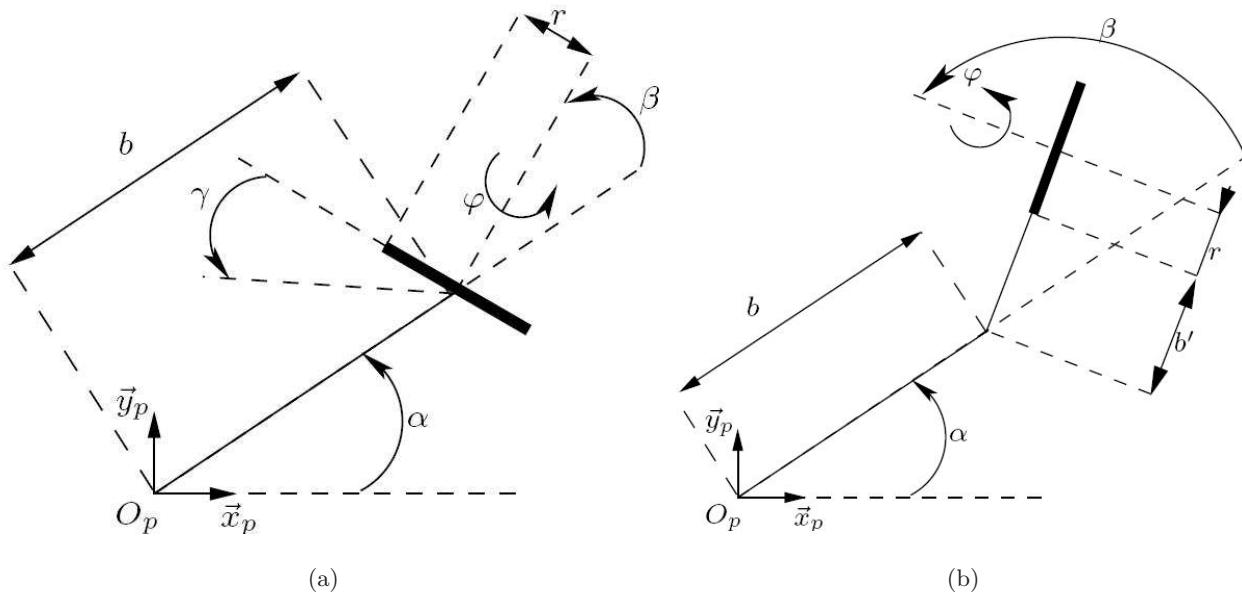


FIG. B.2 – Paramétrage pour la description de la configuration des différents types de roues. (a) Roue fixe, orientable centrée et suédoise. (b) Roue orientable décentrée.

Le tableau B.1 décrit le caractère constant ou variable des paramètres introduits dans la figure B.2 en fonction du type de roue.

L'hypothèse de liaisons parfaites induit trois hypothèses sur le contact entre la roue et le sol :

- la roue, comme le sol, sont indéformables,
- la surface de contact est assimilée à un point
- les roues roulent sans glisser sur le sol

Les contraintes de roulement parfait et sans glissement des roues sur le sol implique des relations différentielles non intégrables des niveaux de vitesses parmi les variables généralisées. Elles peuvent être exprimées sous la forme suivante :

Type de roue	(a)	(b)	(c)	(d)
r	constant	constant	constant	constant
b	constant	constant	constant	constant
φ	variable	variable	variable	variable
β	constant	variable	variable	constant
α	constant	constant	constant	constant
γ	0	0	0	constant
b'	0	0	0	constant

TAB. B.1 – Propriétés des paramètres de description des roues en fonction de leur type

- dans le plan vertical de la roue pour tout type de roue :

$$\begin{bmatrix} -\sin(\alpha + \beta + \gamma) & \cos(\alpha + \beta + \gamma) & b \cos(\beta + \gamma) \end{bmatrix} R(\theta) \dot{\zeta} + r \cos(\gamma) \dot{\varphi} = 0 \quad (\text{B.1.1})$$

- dans le plan orthogonal au plan vertical de la roue (excepté pour les roues suédoises) :

$$\begin{bmatrix} \cos(\alpha + \beta) & \sin(\alpha + \beta) & b' + b \sin \beta \end{bmatrix} R(\theta) \dot{\zeta} + b' \dot{\beta} = 0 \quad (\text{B.1.2})$$

$R(\theta)$ est une matrice orthogonale définissant le passage du repère fixe au repère lié au véhicule.

Elle s'écrit :

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

Remarque B.1 Il est intéressant de remarquer que dans le cas d'une roue suédoise telle que $\gamma = \frac{\pi}{2}$, l'équation (B.1.1) se ramène à une équation de la forme (B.1.2). Ce cas particulier est comparable au cas d'une roue conventionnelle pour laquelle la contrainte de roulement sans glissement ne s'exprimerait que dans le plan orthogonal à la roue. Le bénéfice du choix de ce type de roue est alors nul.

B.2 Classification des plates-formes mobiles

Considérons un robot mobile général équipé de N roues qui sont des quatre types présentés auparavant, i.e.

$$N = N_f + N_{oc} + N_{od} + N_s$$

avec :

- N_f : nombre de roues fixes,
- N_{oc} : nombre de roues orientables centrées,
- N_{od} : nombre de roues orientables décentrées,

- N_{sw} nombres de roues suédoises.

La configuration du robot est totalement décrite par les vecteurs suivants :

- la posture $\zeta = [x, y, \theta]^T$ qui définit la position et l'orientation du robot dans le plan,
- les coordonnées angulaires $\beta_o = [\beta_{oc}, \beta_{od}]^T$ avec β_{oc} désignant l'orientation des roues orientables centrées et β_{od} l'orientation des roues orientables décentrées,
- les coordonnées de rotation $\varphi = [\varphi_f, \varphi_{oc}, \varphi_{od}, \varphi_{sw}]^T$ caractérisant les angles de rotation des roues autour des axes horizontaux.

Le *vecteur des configurations* est $[\zeta, \beta_o^T, \varphi^T]^T$. L'ensemble des équations de contraintes d'un robot mobile s'écrit sous la forme matricielle générale :

$$\begin{aligned} J_1(\beta_{oc}, \beta_{od})R(\theta)\dot{\zeta} + J_2\dot{\varphi} &= 0 \\ C_1(\beta_{oc}, \beta_{od})R(\theta)\dot{\zeta} + C_2\dot{\beta}_{od} &= 0 \end{aligned} \quad (\text{B.2.1})$$

Les matrices J_1 , J_2 , C_1 et C_2 regroupent les paramètres géométriques de toutes les roues. Le système (B.2.1), réduit aux roues fixes et aux roues orientables centrées qui contraignent la plate-forme, s'écrit :

$$\begin{aligned} J_1^*(\beta_{oc})R(\theta)\dot{\zeta} + J_2^*\dot{\varphi} &= 0 \\ C_1^*(\beta_{oc})R(\theta)\dot{\zeta} &= 0 \end{aligned} \quad (\text{B.2.2})$$

Par conséquent, le vecteur $R(\theta)\dot{\zeta} = 0$ est contraint à appartenir au noyau de C_1^* . La dimension du noyau de C_1^* conditionne la mobilité du véhicule. Si le noyau est restreint au vecteur nul, tout mouvement du robot est impossible.

Le *degré de mobilité* du robot, noté δ_m est donc défini par :

$$\delta_m = \dim(\ker(C_1^*(\beta_{oc}))) = 3 - \dim(\text{Im}(C_1^*(\beta_{oc}))) = 3 - \text{rang}(C_1^*(\beta_{oc})) \quad (\text{B.2.3})$$

La non-holonomie des contraintes cinématiques impose des restrictions dans la mobilité du robot. La description de l'orientation des roues est très discutable. Parmi toutes les configurations possibles, seulement quelques unes permettent la mobilité du robot en satisfaisant le roulement pur et le non glissement. Pour plus de détails concernant ces restrictions, le lecteur peut se référer à l'article [Campion et al., 1996].

Le degré de directionnalité δ_g correspond au nombre de roues centrées orientables qu'il faut piloter indépendamment pour diriger le véhicule. Il est donné par :

$$\delta_s = \text{rang}(C_{1c}(\beta_{oc})) \quad (\text{B.2.4})$$

où C_{1c} concerne uniquement les roues orientables centrées.

Ces deux indices permettent d'établir une classification des véhicules en fonction de leur mobilité et du nombre de roues centrées indépendantes qu'il faut piloter pour les diriger. Cinq structures non

singulières existent dont les degrés de mobilité δ_m et de directionnalité δ_g vérifient les trois inéquations :

$$\begin{cases} 1 \leq \delta_m \leq 3 \\ 0 \leq \delta_g \leq 2 \\ 2 \leq \delta_m + \delta_g \leq 3 \end{cases} \quad (\text{B.2.5})$$

Elles sont désignées par la forme : robot mobile de type (δ_m, δ_g) . Suivant le critère de mobilité, deux classes importantes de robots mobiles apparaissent :

- les robots omni-directionnels dont la mobilité est totale dans le plan (i.e $\delta_m = 3$),
- les robots à mobilité réduite dont le degré de mobilité est inférieur à 3 (i.e. $\delta_m < 3$).

Le tableau B.2 récapitule les cinq types de robots mobiles qui correspondent aux paires (δ_m, δ_g) vérifiant les conditions (B.2.5).

Type	I	II	III	IV	V
δ_m	3	2	2	1	1
δ_g	0	0	1	1	2

TAB. B.2 – Types de robots mobiles.

Les caractéristiques principales de la conception de chaque type de robot mobile sont décrites brièvement et notre attention est portée sur des robots mobiles à trois roues. Une description complète peut être trouvée dans [Campion et al., 1996].

► Robot de type $(3, 0)$ (voir la figure B.3). Ces robots n'ont ni roue fixe ($N_f = 0$) ni roue orientable centrée ($N_{oc} = 0$). Le robot est dit “omni-directionnel” parce qu'il a une totale mobilité dans le plan (i.e. il peut bouger dans n'importe quelle direction sans aucune réorientation à chaque instant). Inversement, les quatre autres types de robots mobiles ont une mobilité réduite.

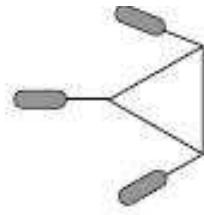


FIG. B.3 – Robot de type $(3, 0)$ avec trois roues orientables décentrées.

► Robot de type $(2, 0)$ (voir la figure B.4). Ces robots n'ont pas de roue orientable centrée. Mais, ils ont une ou plusieurs roues fixes montées sur le même axe. Les robots Hilare, Pekee, Miabot appartiennent à cette classe.

► Robot de type $(2, 1)$ (voir la figure B.5). Ces robots n'ont pas de roues fixes. Ils ont une roue orientable centrée et deux roues orientables décentrées.

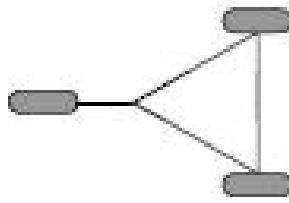


FIG. B.4 – Robot de type (2, 0) avec deux roues fixes sur le même axe et une roue orientable décentrée.

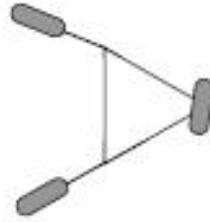


FIG. B.5 – Robot de type (2, 1) avec deux roues orientables décentrées et une roue orientable centrée.

► Robot type (1, 1) (voir la figure B.6). Ces robots ont une ou plusieurs roues fixes montées sur le même axe et aussi une ou plusieurs roues orientables centrées.

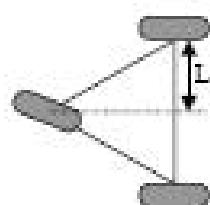


FIG. B.6 – Robot de type (1, 1) avec deux roues fixes sur le même axe et une roue orientable centrée.

► Robot type (1, 2) (voir la figure B.7). Ces robots n'ont pas de roue fixe. Ils ont au minimum deux roues orientables centrées.

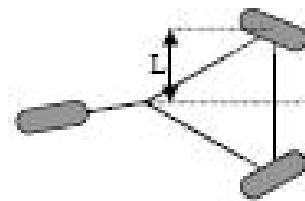


FIG. B.7 – Robot de type (1, 2) avec deux roues orientables centrées et une roue orientable décentrée.

Table des figures

1	Exemples d'application de commande coordonnée de robots mobiles (a) jeu d'équipe [Kim et al., 2004] (b) transport coopératif [Miyata et al., 2002].	14
2	A gauche, le robot Pekee. A droite, le robot Miabot.	16
3	Fonctionnalités de la navigation autonome. Nos contributions théoriques sont représentées en gras.	17
1.1	Description d'une roue.	22
1.2	Robot de type unicycle.	25
1.3	Robot de type voiture.	26
1.4	(a) Trajectoire planifiée. (b) Trajectoire réellement effectuée par le robot.	30
1.5	Suivi d'un véhicule de référence.	32
2.1	Forme géométrique du robot.	44
2.2	Zone obstacle à l'instant $t_k = 0s$	45
2.3	Zone obstacle à l'instant $t_k = 1s$	45
2.4	Platitude et planification de trajectoire.	48
2.5	B-splines linéaire, quadratique et cubique	51
2.6	Exemple de fonction spline et de polygone spline	52
2.7	Horizons de calculs et de planification.	54
2.8	Mise en œuvre de la planification sur un horizon glissant	56
2.9	Trajectoire optimale d'un robot de type unicycle déterminée hors ligne	59
2.10	Commande optimale d'un robot de type unicycle déterminée hors ligne	60
2.11	Trajectoire optimale d'un robot déterminée en ligne	61
2.12	Commande optimale d'un robot déterminée en ligne	61
3.1	Stratégie d'évitement individuel d'obstacles.	68
3.2	Stratégie d'évitement collectif d'obstacles.	68
3.3	Carte initiale de l'environnement.	69
3.4	Détermination du centre de gravité de la flottille.	69
3.5	Détermination de la carte simplifiée.	69

3.6	Calcul de la trajectoire optimale du robot virtuel.	69
3.7	Initialisation des trajectoires des robots.	69
3.8	Calcul des trajectoires optimales des robots de la flottille.	69
3.9	Description des zones de conflits possibles.	71
3.10	Synoptique de l'algorithme décentralisé de planification de trajectoire.	75
3.11	Approche centralisée, développée dans la Section 3.3, pour deux robots se croisant. . .	80
3.12	Approche de type “meneur / suiveur” pour deux robots se croisant.	81
3.13	Approche (“fortement”) décentralisée, développée dans la section 3.4, pour deux robots se croisant.	82
3.14	Graphe de communication de la flottille.	84
3.15	Approche (“fortement”) décentralisée, développée dans la Section 3.4, pour une flottille de cinq robots passant d'une forme “ligne” à une forme “triangulaire”.	85
3.16	Approche centralisée, développée dans la Section 3.3, pour une flottille de cinq robots passant d'une forme “ligne” à une forme “triangulaire”.	87
3.17	Approche “faiblement décentralisée” pour une flottille de cinq robots passant d'une forme “ligne” à une forme “triangulaire”.	88
3.18	Approche “meneur / suiveur” pour une flottille de cinq robots passant d'une forme “ligne” à une forme “triangulaire”.	89
4.1	Exemple de CMG d'ordre un. (a) Trajectoire dans le plan de phase. (b) Commande en fonction du temps.	98
4.2	Le phénomène de réticence.	98
4.3	Plan de phase de l'algorithme du “twisting échantillonné”.	104
4.4	Stabilisation en temps fini du triple intégrateur sans perturbation. (a) Réponse temporelle du système. (b) Commande appliquée.	109
4.5	Réponse du triple intégrateur perturbé avec la commande temps fini [Bhat et Bernstein, 2005]. (a) Trajectoire $z(t)$. (b) Commande appliquée.	109
4.6	Stabilisation en temps fini du triple intégrateur perturbé. (a) Réponse temporelle du système. (b) Commande appliquée.	113
4.7	Aéroglisseur ou “hovercraft” en anglais.	113
4.8	Stabilisation en temps fini des erreurs de suivi de trajectoire pour l'aéroglisseur.	116
4.9	Plate-forme d'essais du moteur pas-à-pas.	123
4.10	Erreur en courant - $C_r = 0Nm$	124
4.11	Erreur en position angulaire du rotor - Observateur de vitesse et d'accélération rotorique - $C_r = 0Nm$	124
4.12	Erreur en courant - Incertitudes paramétriques - $C_r \neq 0$	125

4.13	Erreur en position angulaire du rotor - Observateur de vitesse et d'accélération rotorique - Incertitudes paramétriques - $C_r \neq 0$	126
4.14	Tension et courant appliqués sur l'une des phases du moteur - Incertitudes paramétriques - $C_r \neq 0$	127
5.1	Le robot Pekee	129
5.2	Application directe de la commande planifiée sur le robot Pekee. (a) Trajectoires planifiée et réelle. (b) Erreur de suivi	130
5.3	Application de la commande nominale (5.2.7) sur le robot Pekee. (a) Trajectoires planifiée et réelle. (b) Erreur de suivi	136
5.4	Application de la CMGI sur le robot Pekee. (a) Trajectoires planifiée et réelle. (b) Erreur de suivi	136
5.5	Illustration de la stabilité pratique	138
5.6	Stabilisation pratique du "système de Heisenberg" non perturbé	145
5.7	Stabilisation pratique du "système de Heisenberg" perturbé	148
5.8	Stabilisation en une configuration fixe du robot Pekee	149
5.9	Stabilisation d'une trajectoire non stationnaire pour le robot Pekee	150
6.1	Exemple d'une stratégie de commande de type "meneur / suiveur"	155
6.2	Architecture de CMGI d'ordre un pour la paire "meneur / suiveur"	158
6.3	Architecture de CMGI d'ordre deux pour la paire "meneur / suiveur"	158
6.4	Représentation géométrique d'une triplet (i, j, k) avec (i, j) les meneurs et k le suiveur	163
6.5	Plate-forme Miabot	165
6.6	Robot Miabot	165
6.7	Evolution des erreurs de suivi de trajectoire en appliquant la CMGI d'ordre 1	167
6.8	Vitesses linéaires et angulaires des robots 2 et 3 en appliquant la CMGI d'ordre 1	168
6.9	Evolution des erreurs de suivi de trajectoire en appliquant la CMGI d'ordre 2	169
6.10	Vitesses linéaires et angulaires des robots 2 et 3 en appliquant la CMGI d'ordre 2	170
A.1	Architecture de navigation pour le robot Pekee	178
A.2	Schéma fonctionnel du robot Pekee	180
B.1	Différents types de roues utilisées pour les plates-formes mobiles. (a) Roue fixe. (b) Roue orientable centrée. (c) Roue orientable décentrée. (d) Roue suédoise	184
B.2	Paramétrage pour la description de la configuration des différents types de roues. (a) Roue fixe, orientable centrée et suédoise. (b) Roue orientable décentrée	184
B.3	Robot de type (3, 0) avec trois roues orientables décentrées	187
B.4	Robot de type (2, 0) avec deux roues fixes sur le même axe et une roue orientable décentrée	188

- B.5 Robot de type (2, 1) avec deux roues orientables décentrées et une roue orientable centrée.188
B.6 Robot de type (1, 1) avec deux roues fixes sur le même axe et une roue orientable centrée.188
B.7 Robot de type (1, 2) avec deux roues orientables centrées et une roue orientable décentrée.188

Liste des tableaux

2.1	Configuration des obstacles	58
2.2	Paramètres pour l'optimisation hors ligne de la trajectoire d'un robot	59
2.3	Paramètres pour l'optimisation en ligne de la trajectoire d'un robot	60
3.1	Paramètres pour l'optimisation en ligne de la trajectoire des robots	78
3.2	Comparaisons des performances associées aux différents algorithmes de planification de la flottille de 2 robots	83
3.3	Position initiale des robots “en ligne”	83
3.4	Position finale des robots “en triangle”	84
3.5	Paramètres pour l'optimisation en ligne de la trajectoire des robots	84
3.6	Comparaisons des performances associées aux différents algorithmes de planification de la flottille de 5 robots	90
B.1	Propriétés des paramètres de description des roues en fonction de leur type	185
B.2	Types de robots mobiles.	187

Index

- B-spline, 49
- chemin, 27
- chemin admissible, 27
- commande équivalente, 96
- condition d'attractivité, 95
- condition de η -attractivité, 95
- condition de Brockett, 31
- condition de recouvrement, 96
- conditions d'invariance, 96
- configuration, 27
- coordination par ajustements mutuels, 34
- coordination par leadership, 34
- courbe spline, 50
- degré relatif, 95
- ensemble de glissement, 101
- erreur de suivi, 31
- espace des configurations, 27
- evitement de collisions inter-robots, 65
- graphe de communication, 64
- homogénéité, 107
- planification de trajectoire, 27
- platitude, 42
- polygone spline, 51
- poursuite de trajectoire, 31
- processus centralisé, 34
- processus décentralisé, 35
- régime glissant idéal, 95
- robot de type unicycle, 24
- robot de type voiture, 26
- stabilité en temps fini, 107
- stabilité pratique, 138
- standardisation, 34
- super-twisting, 104
- surface de glissement, 95
- système de Heisenberg, 131
- système holonome, 22
- système non holonome, 22
- système réduit, 95
- trajectoire, 27
- trajectoire admissible, 27
- twisting échantillonné, 103
- variable de glissement, 95
- zone d'accessibilité, 70
- zone de détection, 45
- zone obstacle, 45

Bibliographie

- [Arkin, 1998] Arkin, R. (1998). *Behavior-Based Robotics*. MIT Press, Cambridge, MA.
- [Astolfi, 1996] Astolfi, A. (1996). Discontinuous control of nonholonomic systems. *System and Control Letters*, 27 :37–45.
- [Aubin et Cellina, 1984] Aubin, J. et Cellina, A. (1984). *Differential inclusions*. Springer-Verlag, Berlin.
- [Auquiert et al., 2006] Auquiert, P., Gibaru, O., et Perruquetti, W. (2006). Approximation par des b-splines de solutions optimales pour des problèmes lq : une estimation a posteriori de l'erreur. *e-sta*.
- [Bacciotti et Rosier, 2005] Bacciotti, A. et Rosier, L. (2005). *Liapunov Functions and stability in control theory*. Springer, Berlin.
- [Balch et Arkin, 1998] Balch, T. et Arkin, R. (1998). Behavior-based formation control for multirobot teams. *IEEE Transactions on Robotics and Automation*, 14 :926–939.
- [Barraquand et al., 1992] Barraquand, J., Langlois, B., et Latombe, J. C. (1992). Numerical potential field techniques for robot path planning. *IEEE Transactions on Systems, Man, and Cybernetics*, 22(2) :224–241.
- [Bartolini et al., 1999] Bartolini, G., Ferrara, A., Levant, A., et Usai, E. (1999). On second order sliding mode controllers. *Lecture notes in control and information sciences*.
- [Bhat et Bernstein, 1997] Bhat, S. et Bernstein, D. (1997). Finite-time stability of homogeneous systems. Dans *American Control Conference*.
- [Bhat et Bernstein, 1998] Bhat, S. et Bernstein, D. (1998). Continuous finite-time stabilization of the translational and rotational double integrator. *IEEE Transactions on Automatic Control*, 43(5) :678–682.
- [Bhat et Bernstein, 2005] Bhat, S. et Bernstein, D. (2005). Geometric homogeneity with applications to finite-time stability. *Mathematics of Control, Signals and Systems*, 17 :101–127.
- [Bloch et McClamroch, 1989] Bloch, A. et McClamroch, N. (1989). Control of mechanical systems with classical nonholonomic constraints. Dans *IEEE Conference on Decision and Control*, Tampa, USA.

- [Bloch et al., 1992] Bloch, A., Reyhanoglu, M., et McClamroch, N. (1992). Control and stabilization of nonholonomic dynamic systems. *IEEE Transactions on Automatic Control*, 37 :1746–1757.
- [Bloch et Drakunov, 1996] Bloch, A. M. et Drakunov, S. V. (1996). Stabilization and tracking in the nonholonomic integrator via sliding modes. *System and Control Letters*, 29 :91–99.
- [Bloch et al., 2000] Bloch, A. M., Drakunov, S. V., et Kinyon, M. K. (2000). Stabilization of nonholonomic systems using isospectral flows. *SIAM Journal on Control and Optimization*, 38 :855–874.
- [Bodson et al., 1993] Bodson, M., Chiasson, J., Novotnak, R., et Rekowski, R. (1993). High-performance non linear feedback control of a permanent magnet stepper motor. *IEEE Transactions on Control Systems Technology*, 1 :5–14.
- [Boor, 1978] Boor, C. D. (1978). *A Practical Guide to Splines*. Springer, New York.
- [Borenstein et Koren, 1991] Borenstein, J. et Koren, Y. (1991). The vector field histogram-fast obstacle avoidance for mobile robots. *IEEE Transactions on Robotics and Automation*, 7(3) :278–288.
- [Brock et Khatib, 1998] Brock, O. et Khatib, O. (1998). Mobile manipulation : collision free path modification and motion coordination. Dans *International Conference on Computational Engineering in Systems Applications*.
- [Brockett, 1983] Brockett, R. (1983). Asymptotic stability and feedback stabilization. Dans Brockett, R., Millman, R., et Sussmann, H., editors, *Differential geometric control theory*, pp. 181–195. Boston, MA : Birkhauser.
- [Bryson et Ho, 1975] Bryson, A. et Ho, Y. (1975). *Applied Optimal Control : Optimization, Estimation, and Control*. Hemisphere Publishing Corporation, New York.
- [Campion et al., 1996] Campion, G., d'Andrea Novel, B., et Bastin, G. (1996). Structural properties and classification of dynamic models of wheeled mobile robots. *IEEE Transactions on Robotics and Automation*, 12 :47–62.
- [Canudas-De-Wit et Perruquetti, 2002] Canudas-De-Wit, C. et Perruquetti, W. (2002). Smoothing strategies for high-gain control. Dans *IFAC Latin-American Conference on Automatic Control*, Mexico.
- [Canudas-De-Wit et al., 1996] Canudas-De-Wit, C., Siciliano, B., et Bastin, G. (1996). *Theory of Robot Control*. Springer-Verlag, Berlin.
- [Canudas-De-Wit et Sordalen, 1992] Canudas-De-Wit, C. et Sordalen, O. (1992). Exponential stabilization of mobile robots with nonholonomic constraints. *IEEE Transactions on Automatic Control*, 37 :1791–1797.
- [Cao et Xu, 2004] Cao, W. et Xu, J. (2004). Nonlinear integral-type sliding surface for both matched and unmatched uncertain systems. *IEEE Transactions on Automatic Control*, 49 :1355–1360.

- [Castanos et Fridman, 2006] Castanos, F. et Fridman, L. (2006). Analysis and design of integral sliding manifolds for systems with unmatched uncertainties. *IEEE Transactions on Automatic Control*, 5 :853–858.
- [Chazelle et Guibas, 1989] Chazelle, B. et Guibas, L. J. (1989). Visibility and intersection problems in plane geometry. *Discrete and Computational Geometry*, 4 :551–581.
- [Chen et Serrani, 2006] Chen, X. et Serrani, A. (2006). An internal model approach to autonomous leader/follower trailing for non-holonomic vehicles. *International Journal of Robust and Nonlinear Control*, 16 :641–670.
- [Choset, 1996] Choset, H. (1996). *Sensor Based Motion Planning : the Hierarchical Generalized Voronoi Graph*. Thèse de Doctorat, California Institute of Technology.
- [Chwa, 2004] Chwa, D. (2004). Sliding-mode tracking control of nonholonomic wheeled mobile robots in polar coordinates. *IEEE Transactions on Control Systems Technology*, 12 :637–644.
- [Coron, 1992] Coron, J. (1992). Global asymptotic stabilization for controllable systems without drift. *Mathematics of Control, Signals and Systems*, 5 :295–312.
- [Das et al., 2002] Das, A., Fierro, R., Kumar, V., Ostrowski, J., Spletzer, J., et Taylor, C. (2002). A vision based formation control framework. *IEEE Transactions on Robotics and Automation*, 18 :813–825.
- [Defoort, 2007] Defoort, M. (2007). Commande robuste décentralisée à horizon glissant d'une flottille de robots mobiles. Dans *JDMACS 07, Journées Doctorales du GDR MACS*, Reims, France.
- [Defoort et al., 2005] Defoort, M., Floquet, T., Kokosy, A., et Perruquetti, W. (2005). Tracking of a unicycle-type mobile robot using integral sliding mode control. Dans *International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, Barcelona, Spain.
- [Defoort et al., 2006a] Defoort, M., Floquet, T., Kokosy, A., et Perruquetti, W. (2006a). Commande coopérative d'une formation de robots mobiles. Dans *Conference Internat. Francophone d'Automatique (CIFA)*, Bordeaux, France.
- [Defoort et al., 2006b] Defoort, M., Floquet, T., Kokosy, A., et Perruquetti, W. (2006b). Finite-time control of a class of mimo nonlinear systems using high order integral sliding mode control. Dans *International Workshop on Variable Structure Systems (VSS)*, Alghero, Italy.
- [Defoort et al., 2006c] Defoort, M., Floquet, T., Kokosy, A., et Perruquetti, W. (2006c). Integral sliding mode control for trajectory tracking of a unicycle type mobile robot. *Integrated Computer Aided Engineering*, 13 :277–288.
- [Defoort et al., 2007a] Defoort, M., Floquet, T., Kokosy, A., et Perruquetti, W. (2007a). Decentralized robust control for multi-vehicle navigation. Dans *European Control Conference*, Greece.
- [Defoort et al., 2007b] Defoort, M., Floquet, T., Perruquetti, W., et Drakunov, S. V. (2007b). Integral sliding mode control of extended heisenberg system. *Soumis à International Journal of Control*.

- [Defoort et al., 2006d] Defoort, M., Nollet, F., Floquet, T., et Perruquetti, W. (2006d). Higher order sliding mode control of a stepper motor. Dans *IEEE International Conference on Decision and Control*, San Diego, USA.
- [Defoort et al., 2007c] Defoort, M., Palos, J., Floquet, T., Kokosy, A., et Perruquetti, W. (2007c). Practical stabilization and tracking of a wheeled mobile robot with integral sliding mode controller. Dans *IEEE International Conference on Decision and Control*.
- [Defoort et al., 2007d] Defoort, M., Palos, J., Kokosy, A., Floquet, T., Perruquetti, W., et Boulinguez, D. (2007d). Experimental motion planning and control for an autonomous nonholonomic mobile robot. Dans *IEEE International Conference on Robotics and Automation*, Roma, Italy.
- [Delaleau et da Silva, 1998] Delaleau, E. et da Silva, P. P. (1998). Filtrations in feedback systems : Part i - systems and feedbacks, part ii - input-output decoupling and disturbance decoupling. *Forum Math.*, 10 :259–276.
- [Desai et al., 1998] Desai, J., Ostrowski, J., et Kumar, V. (1998). Controlling formations of multiple mobile robots. Dans *IEEE International Conference on Robotics and Automation*, Leuven, Belgium.
- [Desai et al., 2001] Desai, J. P., Ostrowski, J. P., et Kumar, V. (2001). Modeling and control of formation of nonholonomic mobile robots. *IEEE Transactions on Robotics and Automation*, 17 :905–908.
- [Dimarogonas et al., 2003] Dimarogonas, D., Zavlanos, M., Loizou, S., et Kyriakopoulos, K. (2003). Decentralized motion control of multiple holonomic agents under input constraints. Dans *IEEE Conference on Decision and Control*.
- [Dixon et al., 2000a] Dixon, W. E., Dawson, D. M., Zhang, F., et Zergeroglu, E. (2000a). Global exponential tracking control of a mobile robot system via a pe condition. *IEEE Transactions on Systems, Man and Cybernetics Part B : Cybernetics*, 30 :1520–1540.
- [Dixon et al., 2000b] Dixon, W. E., Jiang, Z. P., et Dawson, D. M. (2000b). Global exponential setpoint control of wheeled mobile robots : a lyapunov approach. *Automatica*, 36 :1741–1746.
- [Drakunov et al., 2005] Drakunov, S. V., Floquet, T., et Perruquetti, W. (2005). Stabilization and tracking control for an extended heisenberg system with a drift. *Systems ans control Letters*, 54 :435–445.
- [Drazenovic, 1969] Drazenovic, B. (1969). The invariance conditions in variable structure systems. *Automatica*, 5(3) :287–295.
- [Dunbar et Murray, 2002] Dunbar, W. et Murray, R. (2002). Model predictive control of coordinated multi-vehicle formations. Dans *IEEE Conference on Decision and Control*.
- [Dunbar et Murray, 2006] Dunbar, W. et Murray, R. M. (2006). Distributed receding for multi-vehicle formation stabilization. *Automatica*, 42 :549–558.

- [Edwards et Spurgeon, 1998] Edwards, C. et Spurgeon, S. K. (1998). *Sliding Mode Control : Theory and Applications*. Taylor and Francis, London.
- [Emelyanov et al., 1986] Emelyanov, S. V., Korovin, S. V., et Levantovsky, L. V. (1986). Higher order sliding modes in the binary control system. *Soviet Physics*, 31 :291–293.
- [Fantoni et al., 2000] Fantoni, I., Lozano, R., Mazenc, F., et Pettersen, K. (2000). Stabilization of a nonlinear underactuated hovercraft. *International Journal of Robust and Nonlinear Control*, 10 :645–654.
- [Filippov, 1983] Filippov, A. (1983). *Differential Equations with Discontinuous Right Hand Sides. Mathematics and its Application*. Kluwer Ac. Pub.
- [Fliess et al., 1992] Fliess, M., Levine, J., Martin, P., et Rouchon, P. (1992). On differentially flat nonlinear systems. In M. Fliess(Ed.), *Nonlinear Control Systems Design*, Oxford, Pergamon Press, pp. 408–412.
- [Fliess et al., 1995] Fliess, M., Levine, J., Martin, P., et Rouchon, P. (1995). Flatness and defect of nonlinear systems : Introductory theory and examples. *International Journal of Control*, 61 :1327–1361.
- [Floquet, 2000] Floquet, T. (2000). *Contributions à la commande par modes glissants d'ordre supérieur*. Thèse de Doctorat, Ecole Centrale de Lille.
- [Floquet et al., 2003] Floquet, T., Barbot, J. P., et Perruquetti, W. (2003). Higher-order sliding mode stabilization for a class of nonholonomic perturbed systems. *Automatica*, 39 :1077–1083.
- [Flores et Milam, 2006] Flores, M. et Milam, M. (2006). Trajectory generation for differentially flat systems via nurbs basis functions with obstacle avoidance. Dans *American Control Conference*, Minneapolis, Minnesota USA.
- [Fox et al., 1997] Fox, D., Burgard, W., et Thrun, S. (1997). The dynamic window approach to collision avoidance. *IEEE Robotics and Automation Magazine*, 4(1) :23–33.
- [Fraisse et al., 2005] Fraisse, P., Gil-Pinto, A., Zapata, R., Perruquetti, W., et Divoux, T. (2005). Stratégie de commande collaborative pour des réseaux de robots. Dans *8^{ièmes} Journées Nationales du Réseau Doctoral de Microélectronique*, Guidel, France.
- [Fraisse et al., 2007] Fraisse, P., Lelevé, A., et Perruquetti, W. (sous presse, 2007). Robots en réseaux. Dans Richard, J.-P. et Divoux, T., editors, *Systèmes commandés en réseaux*. Hermès.
- [Fridman et Levant, 2002] Fridman, L. et Levant, A. (2002). Higher order sliding mode modes. Dans Perruquetti, W. et Barbot, J. P., editors, *Systems and Control Book Series*, Taylor and Francis, pp. 53–101. Boston, MA : Birkhauser.
- [Fridman et al., 2005] Fridman, L., Poznyak, A., et Bejarano, F. (2005). Decomposition of the min-max multi-model problem via integral sliding mode. *International Journal of Control*, 15 :559–574.

- [Gazi et Passino, 2004] Gazi, V. et Passino, K. (2004). Stability analysis of social foraging swarms. *IEEE Transactions on Systems, Man and Cybernetics*, 34 :539–557.
- [Gennaro et Jadbabaie, 2006] Gennaro, M. C. D. et Jadbabaie, A. (2006). Formation control for a cooperative multi-agent system using decentralized navigation functions. Dans *American Control Conference*.
- [Gieras et al., 2002] Gieras, J., Wing, M., et Jacek, F. (2002). *Permanent Magnet Motor Technology*. Electrical Engineering and Electronics, Marcel Dekker Inc.
- [Gil-Pinto et al., 2006] Gil-Pinto, A., Fraisse, P., et Zapata, R. (2006). A decentralized algorithm to adaptive trajectory planning for a group of nonholonomic mobile robots. Dans *IEEE International Conference on Intelligent Robots and Systems*, Beijing, China.
- [Godhavn et Egeland, 1997] Godhavn, J. et Egeland, O. (1997). A lyapunov approach to exponential stabilization of nonholonomic systems in power form. *IEEE Transactions on Automatic Control*, 42 :1028–1032.
- [GrayMatter, 2007] GrayMatter (2007). Urban challenge. Site internet : <http://www.darpa.mil/grandchallenge/index.asp>.
- [Grujic, 1973] Grujic, L. (1973). On practical stability. *International Journal of Control*, 17(4) :881–887.
- [Guo et Parker, 2002] Guo, Y. et Parker, L. E. (2002). A distributed and optimal motion planning approach for multiple mobile robots. Dans *IEEE Conference on Robotics and Automation*.
- [Heck, 1991] Heck, B. (1991). Sliding mode control for singular perturbed systems. *International Journal of Control*, 53 :985–1001.
- [Hespanha et al., 1999] Hespanha, J., Liberzon, D., et Morse, A. (1999). Logic-based switching control of a nonholonomic system with parametric modeling uncertainty. *Systems and Control Letters*, 38 :167–177.
- [Hong, 2002] Hong, Y. (2002). Finite-time stabilization and stabilizability of a class of controllable systems. *Systems and Control Letters*, 46(4) :231–236.
- [Hu et al., 2002] Hu, J., Prandini, M., et Sastry, S. (2002). Optimal coordinated maneuvers for three-dimensional aircraft conflict resolution. *AIAA Journal of Guidance, Control and Dynamics*, 25(5) :888–900.
- [Inalhan et al., 2002] Inalhan, G., Stipanovic, D., et Tomlin, C. (2002). Decentralized optimization, with application to multiple aircraft coordination. Dans *IEEE Conference on Decision and Control*, Orlando, USA.
- [INRIA, 2007] INRIA (2007). Projet ALIEN : Algèbre pour l'identification et estimation numérique. Site internet : <http://www.math-info.univ-paris5.fr/~mboup/ALIEN/index.html>.
- [Isidori, 1995] Isidori, A. (1995). *Nonlinear Control Systems*. Springer-Verlag, London.

- [Itkis, 1976] Itkis, U. (1976). *Control Systems of Variable Structure*. Wiley, New York.
- [Jacob, 1991] Jacob, G. (1991). Lyndon discretization and exact motion planning. Dans *European control conference*.
- [Jadbabaie et al., 2001] Jadbabaie, A., Yu, J., et Hauser, J. (2001). Unconstrained receding horizon control of nonlinear systems. *IEEE Transactions on Automatic Control*, 46 :776–783.
- [Jiang et al., 2001] Jiang, Z. P., Lefeber, E., et Nijmeijer, H. (2001). Saturated stabilization and tracking of a nonholonomic mobile robot. *System and Control Letters*, 42 :327–332.
- [Jiang et Nijmeijer, 1999] Jiang, Z. P. et Nijmeijer, H. (1999). A recursive technique for tracking control of nonholonomic systems in chained form. *IEEE Transactions on Automatic Control*, 44 :265–279.
- [Joines et Houk, 1994] Joines, J. et Houk, C. (1994). On the use of non stationnary penalty functions to solve nonlinear constrained optimization problems. Dans *IEEE International Conference on Evolutionnary Computation*, pp. 579–584.
- [Kanayama et al., 1990] Kanayama, Y., Kimura, Y., Miyazaki, F., et Noguchi, T. (1990). A stable tracking control method for an autonomous mobile robot. Dans *IEEE International Conference on Robotics and Automation*.
- [Kavraki et al., 1996] Kavraki, L., Lamiraux, F., et Holloman, C. (1996). Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12(4) :566–580.
- [Keviczky et al., 2006] Keviczky, T., Fregene, K., Borelli, F., Balas, G., et Godbole, D. (2006). Coordinated autonomous vehicle formations : decentralization, control synthesis and optimization. Dans *American Control Conference*, Minneapolis, Minnesota USA.
- [Khalil et al., 1986] Khalil, H., Kokotovic, P., et O'Reilly, J. (1986). *Singular perturbation methods in control : analysis and design*. Academic Press, Inc.
- [Khatib et al., 1997] Khatib, M., Jaouni, H., Chatila, R., et Laumond, J.-P. (1997). Dynamic path modification for car-like nonholonomic mobile robots. Dans *IEEE Conference on Robotics and Automation*.
- [Khatib, 1986] Khatib, O. (1986). Real-time obstacle avoidance for manipulators and mobile robots. *International Journal of Robotics Research*, 1(5) :90–98.
- [Khennouf et de Wit, 1995] Khennouf, H. et de Wit, C. C. (1995). On the construction of stabilizing discontinuous controllers for nonholonomic systems. Dans *IFAC Nonlinear Control Systems Design Symposium*.
- [Kim et al., 2004] Kim, J. H., Kim, D. H., Kim, Y. J., et Seow, K. T. (2004). *Soccer Robotics*. Springer-Verlag.

- [Kim et Khosla, 1992] Kim, J. O. et Khosla, P. K. (1992). Real-time obstacle avoidance using harmonic potential functions. *IEEE Transactions on Robotics and Automation*, 8(3) :338–349.
- [Kolmanovsky et McClamroch, 1995] Kolmanovsky, I. et McClamroch, N. H. (1995). Developments in nonholonomic control problems. *IEEE Control Systems Magazine*, 15 :20–36.
- [Kuwata et al., 2006] Kuwata, Y., Richards, A., Schouwenaraars, T., et How, J. P. (2006). Decentralized robust receding horizon control. Dans *American Control Conference*, Minneapolis, Minnesota USA.
- [Lafferriere et Sussmann, 1992] Lafferriere, G. et Sussmann, H. (1992). A differential geometric approach to motion planning. Dans adnd J. Canny, Z. L., editor, *Nonholonomic motion planning, International series in engineering and computer science*. Kluwer.
- [Laghrouche et al., 2007] Laghrouche, S., Plestan, F., et Glumineau, A. (2007). Higher order sliding mode control based on integral sliding mode. *Automatica*, 43 :531–537.
- [Laghrouche et al., 2006] Laghrouche, S., Smaoui, M., Plestan, F., et Brun, X. (2006). Higher order sliding mode control based on optimal approach of an electropneumatic actuator. *International Journal of Control*, 79 :119–131.
- [Lassale, 1961] Lassale, J. (1961). *Stability by Liapunov's direct method with applications*. Academic Press, New York.
- [Latombe, 1991] Latombe, J.-C. (1991). *Robot Motion Planning*. Kluwer Academic Publishers, Norwell, MA.
- [LaValle, 2006] LaValle, S. M. (2006). *Planning Algorithms*. Cambridge University Press, Cambridge.
- [Lawrance et al., 1997] Lawrance, C., Zhou, Z., et Tits, A. (1997). User's guide for cfsqp version 2.5. Rapport technique, Institute for Systems Research, University of Maryland, College Park.
- [Lawton et al., 2003] Lawton, J., Beard, R., et Young, B. (2003). A decentralized approach to formation maneuvers. *IEEE Transactions on Robotics and Automation*, 19 :933–941.
- [Lee et Shtessel, 1996] Lee, Y. et Shtessel, Y. B. (1996). Comparison of a feedback linearization controller and sliding mode controllers for a permanent magnet stepper motor. Dans *28th Southeastern Symposium on System Theory*.
- [Levant, 1993] Levant, A. (1993). Sliding order and sliding accuracy in sliding mode control. *International Journal of Control*, 58 :1247–1263.
- [Levant, 1998] Levant, A. (1998). Robust exact differentiation via sliding mode technique. *Automatica*, 34 :379–384.
- [Levant, 2001] Levant, A. (2001). Universal siso sliding-mode controllers with finite-time convergence. *IEEE Transactions on Automatic Control*, 49 :1447–1451.
- [Levant, 2003] Levant, A. (2003). Higher order sliding modes, differentiation and output feedback control. *International Journal of Control*, 76 :924–941.

- [Levant, 2005] Levant, A. (2005). Quasi-continuous high-order sliding-mode controllers. *IEEE Transactions on Automatic Control*, 50 :1812–1816.
- [Lindhe et al., 2005] Lindhe, M., Ogren, P., et Johansson, K. (2005). Flocking with obstacle avoidance : A new distributed coordination algorithm based on voronoi partitions. Dans *IEEE International Conference on Robotics and Automation*, Barcelona, Spain.
- [Lizarraga, 2004] Lizarraga, D. A. (2004). Obstructions to the existence of universal stabilizers for smooth control systems. *Mathematics of Control, Signals, and Systems*, 16 :255–277.
- [Loizou et Kyriakopoulos, 2002] Loizou, S. et Kyriakopoulos, K. (2002). Closed loop navigation for multiple holonomic vehicles. Dans *IEEE International Conference on Intelligent Robots and Systems*, Switzerland.
- [Lozano-Pérez, 1983] Lozano-Pérez, T. (1983). Spatial planning : A configuration space approach. *IEEE Transactions on Robotics and Automation*, 32 :108–120.
- [Man et Yu, 1997] Man, Z. et Yu, X. (1997). Terminal sliding mode control of mimo linear systems. *IEEE Transactions on Circuit and Systems*, 44 :1065–1070.
- [Marchand et Alamir, 2003] Marchand, N. et Alamir, M. (2003). Discontinuous exponential stabilization of chained form systems. *Automatica*, 39 :343–348.
- [Mariottini et al., 2005] Mariottini, G., Pappas, G., Prattichizzo, D., et Daniilidis, K. (2005). Vision based localization of leader-follower formations. Dans *IEEE International Conference on Decision and Control*, Spain.
- [Martin, 1993] Martin, P. (1993). A general sufficient conditions for flatness with m inputs and $m + 1$ states. Dans *IEEE International Conference on Decision and Control*, San Antonio, USA.
- [Martin et al., 2000] Martin, P., Laroche, B., et Rouchon, P. (2000). Motion planing for the heat equation. *International Journal of Robust and Nonlinear Control*, 10 :629–643.
- [Martin et Rouchon, 1995] Martin, P. et Rouchon, P. (1995). Any (controllable) driftless system with m inputs and $m + 2$ states is flat. Dans *IEEE International Conference on Decision and Control*, New Orlean, USA.
- [Milam, 2003] Milam, M. B. (2003). *Real time optimal trajectory generation for constrained dynamical systems*. Thèse de Doctorat, California Institute of Technology.
- [Mintzberg, 1979] Mintzberg, H. (1979). *The structuring of organizations*. Prentice-Hall, Englewoods Cliffs, N.J.
- [Miyata et al., 2002] Miyata, N., Ota, J., Arai, T., et Asama, H. (2002). Cooperative transport by multiple mobile robots in unknown static environments associated with real-time task assignment. *IEEE Transactions on Robotics and Automation*, 18 :769–780.
- [Moulay, 2005] Moulay, E. (2005). *Une contribution à l'étude de la stabilité en temps fini et de la stabilisation*. Thèse de Doctorat, Ecole Centrale de Lille.

- [Mounier et Rudolph, 1998] Mounier, H. et Rudolph, J. (1998). Flatness based control of nonlinear delay systems : A chemical reactor example. *International Journal of Control*, 71 :871–890.
- [Murray et Sastry, 1993] Murray, R. et Sastry, S. (1993). Nonholonomic motion planning : Steering using sinusoids. *IEEE Transactions on Automatic Control*, 38 :700–716.
- [Nijmeijer et Schaft, 1991] Nijmeijer, H. et Schaft, A. V. D. (1991). *Nonlinear Dynamical Control Systems*. Springer-Verlag.
- [Nollet et al., 2004] Nollet, F., Floquet, T., et Perruquetti, W. (2004). Observer-based second order sliding mode control for the stepper motor. Dans *International Worshop on Variable Struture System*.
- [Nollet et al., 2006] Nollet, F., Floquet, T., et Perruquetti, W. (2006). Lois de commande par modes glissants pour le moteur pas-à-pas. *Journal Européen des Systèmes Automatisés*, 40(8) :885–910.
- [Ogren, 2003] Ogren, P. (2003). *Formations and Obstacle Avoidance in Mobile Robot Control*. Thèse de Doctorat, Royal Institute of Technology.
- [Olfati-Saber et al., 2003] Olfati-Saber, R., Dunbar, W., et Murray, R. (2003). Cooperative control of multi-vehicle systems using cost graphs and optimization. Dans *American Control Conference*, Denver, USA.
- [Orqueda et Fierro, 2006] Orqueda, O. et Fierro, R. (2006). Robust vision based nonlinear formation control. Dans *American Control Conference*, Minneapolis, USA.
- [Park, 1929] Park, R. H. (1929). Two reaction theory of synchronous machines - generalized method of analysis- part i. *AIEE Trans*, 48 :716–727.
- [Perruquetti et Barbot, 2002] Perruquetti, W. et Barbot, J. P. (2002). *Sliding Mode Control in Engineering*. Ed. Marcel Dekker.
- [Perruquetti et al., 1995] Perruquetti, W., Richard, J.-P., Grujic, L., et Borne, P. (1995). On practical stability with settling time via vector norms. *International Journal of Control*, 62(1) :173–189.
- [Pomet, 1992] Pomet, J. B. (1992). Explicit design of time-varying stabilizing control laws for a class of controllable systems without drift. *Systems and Control Letters*, 18 :147–158.
- [Pontryagin et al., 1962] Pontryagin, L., Boltyanskii, V., Gamkrelidze, R., et Mishchenko, E. (1962). *The Mathematical Theory of Optimal Processes*. Wiley-Interscience.
- [Praly, 1997] Praly, L. (1997). Generalized weighted homogeneity and state depedendant time scale for linear controllable systems. Dans *IEEE International Conference on Decision and Control*, San Diego, California USA.
- [Prieur et Astolfi, 2003] Prieur, C. et Astolfi, A. (2003). Robust stabilization of chained systems via hybrid control. *IEEE Transactions on Automatic Control*, 48 :1768–1772.
- [Prieur et Trelat, 2005] Prieur, C. et Trelat, E. (2005). Robust optimal stabilization of the brockett integrator via a hybrid feedback. *Math. Control Signals Systems*, 17 :201–216.

- [Quinlan, 1994] Quinlan, S. (1994). *Real-Time Path Modification of Collision-Free Paths*. Thèse de Doctorat, Université de Stanford, Etats-Unis.
- [Rekleitis et al., 2000] Rekleitis, I. M., Dudek, G., et Milios, E. E. (2000). Multi-robot collaboration for robust exploration. Dans *IEEE International Conference on Robotics and Automation*, San Francisco, USA.
- [Rimon et Koditschek, 1992] Rimon, E. et Koditschek, D. (1992). Exact robot navigation using artificial potential functions. *IEEE Transactions on Robotics and Automation*, 8(5) :501–518.
- [Samson, 1990] Samson, C. (1990). Velocity and torque feedback control of a nonholonomic cart. Dans *International Workshop Nonlinear Adaptive Control : Issues in Robotics*, France.
- [Samson, 1995] Samson, C. (1995). Control of chained systems : Application to path following and time-varying point-stabilization of mobile robots. *IEEE Transactions on Automatic Control*, 40 :64–77.
- [Schouwenaars et al., 2001] Schouwenaars, T., Moor, B. D., Feron, E., et How, J. (2001). Mixed integer programming for multi-vehicle path planning. Dans *European Control Conference*.
- [Schwartz et Sharir, 1988] Schwartz, J. T. et Sharir, M. (1988). A survey of motion planning and related geometric algorithms. *Artificial Intelligence*, 37 :157–169.
- [Shin et McKay, 1986] Shin, K. et McKay, N. (1986). A dynamic programming approach to trajectory planning of robotic manipulators. *IEEE Transactions on Automatic Control*, 31 :491–500.
- [Sira-Ramirez, 1993] Sira-Ramirez, H. (1993). On the dynamical sliding mode control of nonlinear systems. *International Journal of Control*, 57(5) :1039–1061.
- [Sira-Ramirez, 2000] Sira-Ramirez, H. (2000). A passivity plus flatness controller for the permanent magnet stepper motor. *Asian Journal of Control*, 9 :1–9.
- [Sira-Ramirez, 2002] Sira-Ramirez, H. (2002). Dynamic second-order sliding mode control of the hovercraft vessel. *IEEE Transactions on Control Systems Technology*, 10(6) :860–865.
- [Slotine et Sastry, 1983] Slotine, J. et Sastry, S. (1983). Tracking control of nonlinear systems using sliding surfaces with application to robot manipulator. *International Journal of Control*, 38(2) :421–434.
- [Sontag, 1999] Sontag, E. (1999). Stability and stabilization : discontinuities and the effect of disturbances. Dans Clarke, F. et Stern, R., editors, *Nonlinear Analysis, Differential Equations, and Control*, pp. 551–598. Kluwer, Dordrecht.
- [Statheros et al., 2006] Statheros, T., Defoort, M., Khola, S., McDonald-Maier, K., Howells, W., Kokosy, A., Palos, J., Perruquetti, W., et Floquet, T. (2006). Automated control and guidance system (acos) : An overview. Dans *International Conference on Recent Advances in Soft Computing (RASC)*, Canterbury, UK.

- [Tanner et Kumar, 2005] Tanner, H. et Kumar, A. (2005). Formation stabilization of multiple agents using decentralized navigation functions. *Robotics : Science and Systems*, pp. 49–56.
- [Tanner et al., 2003] Tanner, H., Loizou, S., et Kyriakopoulos, K. (2003). Nonholonomic navigation and control of multiple mobile manipulators. *IEEE Transactions on Robotics and Automation*, 9 :777–790.
- [Tanner et al., 2004] Tanner, H., Pappas, G., et Kumar, V. (2004). Leader to formation stability. *IEEE Transactions on Robotics and Automation*, 20 :443–455.
- [Tomlin et al., 1998] Tomlin, C., Pappas, G. J., et Sastry, S. (1998). Conflict resolution for air traffic management : a study in multi-agent hybrid systems. *IEEE Transactions on Automatic Control*, 43 :509–521.
- [Utkin, 1992] Utkin, V. (1992). *Sliding Modes in Control and Optimization*. Springer-Verlag, Berlin, Germany.
- [Utkin et Shi, 1996] Utkin, V. I. et Shi, J. (1996). Integral sliding mode in systems operating under uncertainty conditions. Dans *IEEE International Conference on Decision and Control*, Kobe, Japan.
- [Valtolina et Astolfi, 2003] Valtolina, E. et Astolfi, A. (2003). Local robust regulation of chained systems. *Systems Control Letters*, 49 :231–238.
- [Vidal et al., 2004] Vidal, R., Shakernia, O., et Sastry, S. (2004). Following the flock. *IEEE Robotics and automation magazine*, 11 :14–20.
- [Weiss et Infante, 1967] Weiss, L. et Infante, E. (1967). Finite time stability under perturbing forces and on product spaces. *IEEE Transactions on Automatic Control*, AC-12(1) :54–59.
- [Xu et al., 1998] Xu, J., Lee, H., Lia, Q., et Wang, M. (1998). On adaptative robust backstepping control schemes suitable for pm synchronous motor. *International Journal of Control*, 70 :893–920.
- [Young et al., 1999] Young, K., Utkin, V., et Özgüner, O. (1999). A control engineer’s guide to sliding mode control. *IEEE Transactions on Control Systems technology*, 7 :328–342.
- [Zapata et al., 2004] Zapata, R., Cacitti, P., et Lépinay, P. (2004). Dvz-based collision avoidance control of nonholonomic mobile manipulators. *Journal Européen des Systèmes Automatisés*, 38(5) :559–588.
- [Zribi et Chiasson, 1993] Zribi, M. et Chiasson, J. (1993). Position control of a pm stepper motor by exact linearization. *IEEE Transactions on Automatic Control*, 36 :620–625.
- [Zribi et al., 2001] Zribi, M., Sira-Ramirez, H., et Ngai, A. (2001). Static and dynamic sliding mode control schemes for a permanent magnet stepper motor. *International Journal of control*, 74 :103–117.