REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR
ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE DE JIJEL
FACULTE DES SCIENCES ET DE LA TECHNOLOGIE
DEPARTEMENT D'ELECTRONIQUE

THÈSE PRÉSENTÉE POUR l'OBTENTION DU DIPLOME DE DOCTORAT EN SCIENCES EN ÉLECTRONIQUE

Par : Mohamed BOUKENS

Contribution à la commande des robots manipulateurs mobiles

Soutenue publiquement, le 01/ 07/ 2017

Président :	Mr. A. SOUKKOU	HDR	Université MSB Jijel.
Rapporteur :	Mr. A. BOUKABOU	Prof	Université MSB Jijel.
Co-rapporteur :	Mr. M. CHADLI	HDR	Université de Picardie Jules Verne, Amiens France.
Examinateurs :	Mr. S. LADACI	Prof	Ecole Nationale Polytechnique, Constantine.
	Mr. D. BOUDJEHEM	Prof	Université de Guelma.

Un petit cadeau pour ma femme

Dr. Ouarda Gueroui

Remerciements

Je tiens tout d'abord à adresser mes plus vifs remerciements et exprimer ma profonde gratitude au Directeur de cette Thèse Mr. Abdelkrim BOUKABOU, Professeur à l'Université de Jijel. Je vous remercie pour votre soutien scientifique, pour votre confiance, pour vos multiples conseils qui ont été pour moi une source d'enrichissement et d'encouragement, Ce travail n'aurait pu être mené à bout sans votre aide et votre attention permanente sur l'évolution de mes recherches tout au long de cette thèse.

J'exprime ma profonde reconnaissance au Co-Directeur de Thèse Mr. Mohammed CHADLI, Maître de Conférences HDR à l'Université de Picardie UPJV, France. Je vous remercie tout d'abord de m'avoir accueilli au sein de votre laboratoire Modélisation, Information et Systèmes (MIS), de l'Université de Picardie Jules Verne, à l'occasion de mon séjour de PNE. Je vous remercie pour votre souci constant de l'avancement de ma thèse, de m'avoir accordé votre confiance, vos conseils, et pour vos encouragements qui ont poussé en avant mes travaux de recherche et m'ont été très utiles pour mener à bien ce travail.

Mes remerciements vont ensuite aux membres du jury, qui m'ont fait l'honneur d'évaluer mon travail de thèse. Je tiens à exprimer mes sincères reconnaissances et mon profond respect à Mr. Ammar SOUKKOU, Maître de Conférences HDR à l'Université de Jijel, pour avoir accepté d'examiner ma thèse et pour m'avoir fait l'honneur de présider le jury de thèse. Je remercie également les membres du jury, Mr. Samir LADACI**,** Professeur à l'Ecole Nationale Polytechnique ENP de Constantine, et Mr. Djalil BOUDJEHEM, Professeur à l'Université de Guelma. Qui m'ont fait l'honneur de participer au jury de ma soutenance, et d'avoir pris le temps d'examiner mon travail, je vous remercie pour le temps que vous avez consacré à la lecture de mon manuscrit et à la rédaction de vos rapports éclairés.

Ce travail est particulièrement dédié à ma femme Dr. Ouarda Gueroui, qui n'a jamais cessé de me soutenir et me supporter pour que je puisse finir ce travail doctoral, et à qui je voudrais dédier ma thèse de doctorat.

Un grand merci à mon père et ma mère, à mon frère et mes sœurs, qui m'ont supporté tout au long de mes études et dont je serai indéfiniment redevable.

Finalement, je remercie toute personne qui m'a soutenu de près ou de loin tout au long de mon parcours.

ملخص

تركز هذه الأطروحة على جزء التحكم المتعلق بحركة الروبوت والهدف من ذلك هو اقتراح خوارزميات جديدة للمساهمة في مشكلة التحكم في حركة روبوت متحرك ذو ذراع. مستقل وعام لها مزايا مثيرة للاهتمام, بما في ذلك سهولة التنفيذ دون معرفة كلية بالنظام ولا يحتاج شخص خبير لكي يحدد له اعداداته ويشمل أساسا مستويين ، آلية تعمل على أساس نظام غامض وخوارزمية معدلة من مستعمرات النمل و يتمتع بسرعة حساب في الوقت الحقيقي ومرونة العمل مع قانون التحكم ومستوى آخر يتعلق بقانون التحكم عن طريق اختيار طريقة هجينة جديدة الذي تجمع بين جزء من التحكم الأمثل، التحكم العصبي المتكيف، والتحكم المتين المتكيف. في هذا المعنى، فإن الخوارزمية الجديدة تتجاوز دور الرقابة والتحسين. ويمكن أن ينظر إليها ليس فقط كقانون آخرللتحكم مع بعض المتانة لنظام ديناميكي غير معروف ولكن يمكن اعتبارها امتدادا كبير الذي يغطي الحالة الأكثر عامة لتطبيق قانون التحكم والاعداد

.كلمات البحث: الروبوت المتحرك ، امثل، قوية، التكيف، الشبكة العصبية، المنطق الضبابي، مستعمرات النمل

Résumé

Cette thèse s'intéresse à la partie commande de la navigation robotique. L'objectif est de proposer de nouveaux algorithmes pour contribuer au problème de commande des mouvements des robots manipulateurs mobiles. Ce travail trouve son intérêt dans la conception et l'exécution en temps réel d'un nouvel algorithme de contrôle intelligent, autonome et plus générique, possède des avantages intéressants dont la facilité d'implémentation avec la moindre connaissance sur le système à commander, et ne nécessite pas le savoir-faire d'un utilisateur qualifié pour l'implémentation et le paramétrage. Il comprend principalement deux niveaux; un mécanisme basé sur le système flou et un algorithme modifié des colonies de fourmis pour le réglage autonome des différents paramètres d'une loi de commande, et qui dispose une rapidité de calcul en temps réel et une souplesse de la mise en ligne avec la loi de commande ; et un autre niveau destiné à la loi de commande en choisissant une nouvelle approche hybride, qui combine une partie de la commande optimale, la commande neuronale adaptative, et la commande robuste adaptative. Dans ce sens, l'algorithme dépasse le simple rôle de commande et d'optimisation et peut être considérée non seulement comme une autre loi pour commander avec une certaine robustesse un système dynamique inconnu, mais comme une extension considérable qui couvre le cas le plus général de l'implémentation d'une loi de commande et leur paramétrage.

Mots clés : Robot manipulateur mobile, commande optimale, robuste, adaptative, réseau de neurones, logique floue, méthaheuristique, colonies de fourmis.

Abstract

This thesis focuses on the control part of robotic navigation. The objective is to propose new algorithms to contribute to the problem of motion control of mobile robot manipulators. This work is of interest in the design and real-time execution of a new intelligent, autonomous and more generic control algorithm, it has interesting advantages whose ease of implementation with the least knowledge about the system to be controlled, and does not require the know-how of a qualified user for implementation and parameterization. It mainly comprises two levels; a mechanism based on the fuzzy system and a modified algorithm of the ant colonies for the autonomous adjustment of the various parameters of a control law, and which provides a speed of computation in real time and a flexibility of executing in-line with the control law. And another level for the control law, by choosing a new hybrid approach, which combines a part of the optimal control, adaptive neural control, and robust adaptive control. In this sense, the algorithm goes beyond the simple role of control and optimization, and can be considered not only as another law to control with some robustness an unknown dynamic system but as a considerable extension which covers the most general case of the implementation of a control law and their parameterization.

Keywords: Mobile robot manipulator, optimal control, robust control, adaptive control, neural network, fuzzy logic, metaheuristic, ant colonies.

List of Publications

Journals:

- **Boukens.** M, Boukabou. A. and Chadli. M., Intelligent trajectory tracking controller for wheeled nonholonomic mobile robot manipulators: A real time ants colony optimization-based fuzzy approach, **Submitted in IEEE Transactions on Robotics, 2017.**
- **Boukens. M**, Boukabou. A. and Chadli. M., Robust adaptive neural network-based trajectory tracking control approach for nonholonomic electrically driven mobile robots. **Elsevier, Robotics and Autonomous Systems,** vol. 92, pp. 30–40**, 2017.**
- **Boukens. M** and Boukabou. A., Design of an intelligent optimal neural network-based tracking controller for nonholonomic mobile robot systems. **Elsevier, Neurocomputing,** vol. 226, pp. 46–57**, 2017.**

Proceedings of Peer-Reviewed International Conferences:

- **Boukens, M.**, Boukabou, A. and Belkheiri, M., Robust adaptive neural network based controller for nonholonomic mobile robot systems subject to unknown perturbations. In: **Proceedings of the IEEE** international conference on modeling, identification and control ICMIC, Algeria, pp. 774–779, DOI: 10.1109/ICMIC.2016.7804218, **2016.**
- **Boukens, M.** and Boukabou, A., Robust adaptive neural network-based control of robot manipulators subject to external disturbances. In: **Proceedings of the IEEE** International Conference on Industrial Instrumentation and Control, (ICIC), India, pp. 934 - 939, DOI: 10.1109/IIC.2015.7150878, **2015.**
- **Boukens, M.** and Boukabou, A., Neuro optimal controller for robot manipulators, Artificial Neural Networks and Machine Learning ICANN, Bulgaria**,** Lecture Notes in Computer Science, vol 8131. **Springer**, vol. 8131, pp. 503–510, **2013**.
- **Boukens, M.** and Boukabou, A., PD with fuzzy compensator control of robot manipulators: experimental study. In: **Proceedings of the IEEE** international conference on Systems and Control, Algeria, ,pp.973-978, DOI: 10.1109/ICoSC.2013.6750975, **2013.**
- **Boukens, M.** and Boukabou, A., Commande PD avec compensation neuronale: étude expérimentale en temps réel. 3rd International Conference on Systems and Processing Information, Guelma, Algeria, **2013.**

Sommaire

Liste des figures :

Tableaux :

Introduction générale.

I.1. Introduction

Depuis l'avènement de la technologie, le mot robot dans l'esprit du grand public construit une image différente, et plusieurs idées viennent étroitement liés à la science-fiction. La réalité des choses est un peu différente. Le domaine de la robotique telle qu'elle est connue aujourd'hui est une science interdisciplinaire comprenant de larges champs de recherches articulées autour d'un seul objectif.

Depuis toujours les chercheurs à s'interroger sur la nature de l'être vivant, afin de la copier intégralement ou partie par partie. Les enjeux sont philosophiques des sciences fictions par définition, et d'autre part industrielles par nécessité, en s'inspirant des mécanismes pour des réalisations techniques et en cherchant à automatiser tout processus dans la vie quotidienne. Cet objectif de l'automatisation flexible construit donc l'image du robot idéal et pousse la recherche vers la machine universelle dont l'homme rêve depuis toujours.

Dans cette optique et autour de ces pensées sont criées les techniques de l'intelligence artificielle. Ces approches sont devenues de plus en plus populaires dans de nombreuses disciplines. Sans aucun doute, les capacités mentales humaines d'apprentissage, de raisonnement, de mémorisation et de prédiction représentent la base de tout système intelligent. Malgré cela, nous sommes encore loin d'atteindre quelque chose de semblable à l'intelligence humaine.

En regardant les capacités de cerveau humain, l'intelligence fondamentale est d'assurer la survie dans la nature, et non d'effectuer des calculs précis. Les chercheurs veulent transférer certaines de ces capacités dans les algorithmes et les systèmes artificiels afin de leur permettre de survivre dans un environnement naturel. Ces recherches ne sont pas fermées autour de l'être humain mais tentent d'imiter l'intelligence trouvée dans toute la nature. Ce champ est un domaine en évolution rapide qui combine des méthodologies de diverses sources.

En effet, une question toujours évoquée et se pose dans le domaine de l'automatique ; qu'est-ce qu'un contrôle intelligent? La réponse à cette question peut devenir plutôt philosophique. Généralement, les techniques d'un contrôle intelligent offrent des alternatives aux approches conventionnelles en empruntant des idées à partir de systèmes biologiques intelligents ou en observant comment un système biologique fonctionne et en utilisant des démarches analogues dans la solution des problèmes de contrôle.

Dans ces dernières années, nous avons assisté au boom de la robotique, et les robots actuels sont dotés d'une intelligence, qui leurs donne une certaine autonomie qui va leurs permettre de se diffuser dans de nouveaux domaines. On parle aujourd'hui de robots mobiles d'exploration et d'intervention, de robots militaires, de robots médicaux, assistants ou domestiques et une infinité de systèmes robotiques apparait. Ces divers champs obligent ce système à avoir une grande capacité de contrôle, de traitement et une structure hiérarchique sophistiquée afin d'atteindre sa propre autonomie. Dans le but de réussir ce défi, il est indispensable de modifier nos techniques de conception conventionnelles, soient au niveau de traitement et de raisonnement ainsi qu'au niveau de la commande et l'exécution pour donner aux robots un degré d'autonomie en mutation continue.

L'espace dans lequel un robot mobile se déplace est souvent vaste partiellement ou totalement structuré. Le besoin de robot ayant une capacité hybride de déplacement et de

manipulation a conduit à la création de manipulateurs mobiles. Cette caractéristique de mobilité ouvre fortement les portes vers d'autres applications et plusieurs domaines peuvent bénéficier des apports des manipulateurs mobiles.

Cette famille de système est intrinsèquement non linéaire de par sa cinématique et ses caractéristiques dynamiques, et dans le principe d'un espace de travail soit étendu, on doit assurer non seulement la performance d'autonomie au niveau de traitement et de décision mais aussi, une commande qui présentent suffisamment de robustesse pour garantir cette performance.

En effet, notre travail présenté dans cette thèse vise à élaborer des lois de commande dans le but de garantir certain niveau de performances et plus précisément lors d'une application de suivi de trajectoire pour des missions de manipulation mobile. Les lois de commande développées permettent de prendre en compte un certain nombre de contraintes qui seront détaillées plus tard, et notre méthodologie repose sur l'intention de concevoir des solutions qui s'appliquent à une classe de systèmes robotiques plutôt qu'à une architecture particulière.

I.2. Contexte des manipulateurs mobiles:

Les manipulateurs mobiles sont apparus pour remplir des missions qui nécessitent à la fois des capacités de déplacement et de manipulation. Ces manipulateurs mobiles prennent des formes diverses bien adaptées aux missions dédiées à des tâches précises. Ces missions sont caractérisées par des objectifs de natures différentes ; explorations, interventions et manipulations, ainsi donc, des mouvements à effectuer et/ou des efforts à appliquer dans un espace de travail étendu et souvent peu structuré.

La maîtrise de mouvement d'un corps implique nécessairement sa localisation et son orientation dans l'espace. En effet, pour placer l'effecteur d'un bras manipulateur à une position arbitraire dans l'espace de travail, il faut disposer de trois degrés de liberté qui peuvent être des translations ou des rotations définissent ce que l'on appelle le porteur. Pour avoir une orientation quelconque de l'effecteur, il faut disposer en plus de trois autres degrés de liberté qui sont nécessairement des rotations constituent ce que l'on appelle le poignet d'un bras manipulateur.

Par ailleurs, un moyen de locomotion porte un manipulateur présente l'avantage d'avoir un espace de travail considérablement étendu du fait des degrés de mobilité apportés par sa plateforme mobile. En effet, le contrôle de mouvement d'un manipulateur mobile est intuitivement lié à ses caractéristiques cinématiques et dynamiques. Contrairement aux manipulateurs à bas fixe, le concept de la plateforme mobile est vaste et diffèrent par leurs caractéristiques structurelles ce qui donne une formulation différente de point de vue cinématique et du problème de contrôle pour chaque type.

Aussi, de nombreux robots ont été développés dans différentes centres de recherche œuvrant en milieux incertains, voire dangereux, en particulier dans le domaine de la défense et dans un cadre tout-terrain couvre également un intérêt majeur, et ce n'est pas notre but ici d'en faire une liste exhaustive. Parmi les manipulateurs mobiles existant actuellement nous pouvons citer trois grandes familles.

a. Les humanoïdes :

Un humanoïde est un robot dont l'apparence générale rappelle celle d'un corps humain et disposent des capacités de locomotion, de manipulation et d'interaction avec l'homme dans l'objectif de réaliser une machine à l'image de l'homme. Quand la marche sur tout terrain, si naturelle pour l'homme, représente un défi complexe à surmonter en robotique qui n'est pas encore réussi à reproduire à l'identique les mouvements autonomes de l'homme. La limite technique réside dans les problèmes d'adaptation du robot à son environnement. Comme le robot Atlas de Boston Dynamics et le terminateur russe FEDOR, l'un des robots marcheurs les plus avancés au monde.

Figure.1.1. Le robot Atlas de Boston Dynamics.

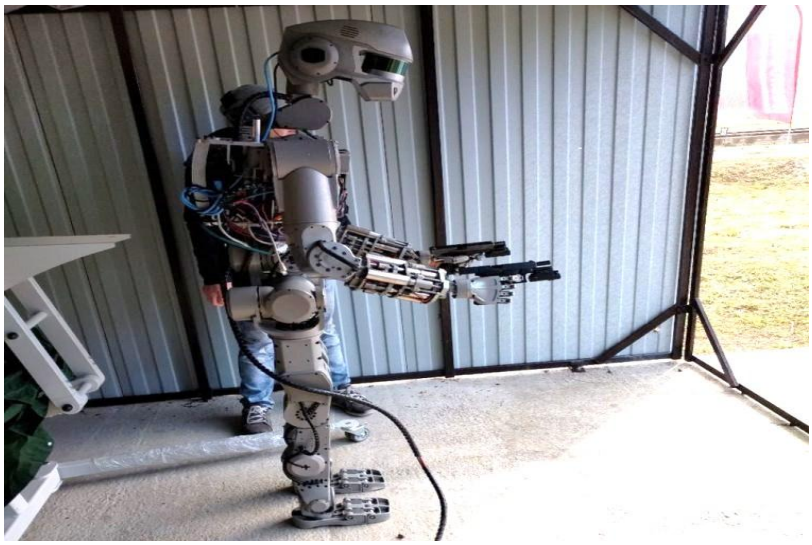


Figure.1.2. Le robot FEDOR.

b. Les manipulateurs mobiles sous-marins :

Souvent télé-opérés, ils permettent l'accès à des zones maritimes non accessibles aux plongeurs et fournissent des capacités de prélèvement, de manipulation, de mesure et d'acquisition de données adaptables aux missions envisagées. Les robots sous-marins autonomes, quant à eux, ne nécessitent pas l'intervention pour accomplir une mission. Nous pouvons citer, à titre d'exemple, H-ROV Ariane de l'Ifremer.

Figure.1.3. Robot sous-marin H-ROV Ariane de l'Ifremer.

c. Les manipulateurs mobiles à roues :

Les manipulateurs mobiles à roues prennent aujourd'hui des formes diverses, tant du point de vue de mode de locomotion que de leur taille ou encore de la fonction attribuée au bras manipulateur dans le système. Le nombre et les caractéristiques des roues et leur disposition confère à un robot son mode de mobilité propre, on rencontre principalement quatre types de robot roulants :

1. Robot omnidirectionnel :

Un type de plateforme mobile s'appelle aussi les robots mobiles holonomes. Ils ont une structure mécanique complexe qui leur permet de se déplacer dans toutes les directions indépendamment de son orientation et sans manœuvre ; marche avant-arrière. Une telle structure peut être réalisé par trois roues décentrées orientables, ou de formes sphériques pouvant tourner librement dans deux directions.

Figure.1.4. Robot Pepper avec une base omnidirectionnel.

Figure.1.5. Manipulateur mobile omnidirectionnel Kuka.

Un robot est dit holonome lorsque son nombre de degrés de libertés directement contrôlables est égal au nombre de degrés de mobilité dans son espace de travail. Le mouvement d'un corps sur un plan, dispose trois degrés de mobilité ; deux translations sur les axes X et Y et une rotation autour de l'axe Z. De point de vue cinématique et à partir d'une configuration initiale, un robot mobile holonome peut se déplacer en avant, latéralement et tourner sur lui-même simultanément et de manière indépendante. Cette caractéristique permet de commander indépendamment les trois variables d'état de la plateforme mobile ce qui nous permet d'avoir un asservissement découplé pour chaque variable.

Contrairement à ce type de robot mobile, on trouve le robot non-holonome qui dispose des roues ordinaires qui manque un degré de mobilité dans une direction latérale. Pour fixer les idées prenons l'exemple typique de la voiture pour se garer ; elle est obligée de faire une manœuvre, une marche avant puis une marche arrière. On trouve principalement trois types de robots mobiles non-holonomes.

2. **Robot uni-cycle :**

Il est actionné par deux roues indépendantes et dispose éventuellement un certain nombre de roues folles, qui leur permet d'être stables en statique et de ne pas basculer en mouvement. Une solution parfaite est de remplacer les roues par des chenilles lorsque le terrain possède des différences d'altitude considérables, et dont les modèles sont semblables au robot à roues du point de vue de la commande. Le nouveau robot de Boston Dynamics Handle peut être le plus sophistiqué au monde.

Figure.1.6. Plateforme Segway RMP200 avec manipulateur KUKA KR-5

Figure.1.7. Robot à chenilles, LAND Robot TALON IV.

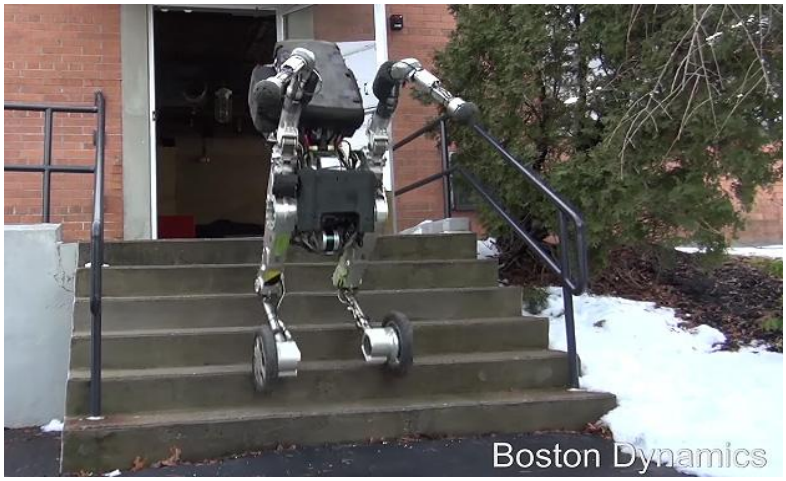


Figure.1.8. Robot Handle de Boston Dynamics.

3. **Robot tricycle** :

Constitué de deux roues dépendantes en vitesse d'un seul axe, et d'une roue centrée orientable qui produit une vitesse angulaire sur le plan. La vitesse linéaire est assurée par une traction des roues arrière.

Figure.1.9. Robot tricycle de National instruments

4. **Robot type voiture** :

Similaire au tricycle de point de vue cinématique, il diffère au niveau qui possède deux roues orientables dépendantes garantissant ainsi une meilleure stabilité.

Figure.1.10. Robot Black Max-QinetiQ

I.3. Description du système expérimental :

Avant de faire ici une brève description de la structure expérimentale, il est nécessaire ici de rappeler l'importance de l'expérimentation dans une discipline telle que la robotique.

Dans la structure générale d'un contrôleur de robot, on distingue un premier niveau de planificateur des tâches et d'un second niveau de calcul et de contrôle des mouvements envoyés par le planificateur. Ce contrôleur correspond tant à la commande qu'à la supervision du robot qui décrit une architecture matérielle et logicielle. Ainsi, Dans ce travail, nous n'abordons pas la problématique sur le planificateur mais, en revanche, nous nous sommes intéressés uniquement au contrôleur des mouvements (Figure.1.11), et plus précisément au cas d'un robot manipulateur mobile à roues.

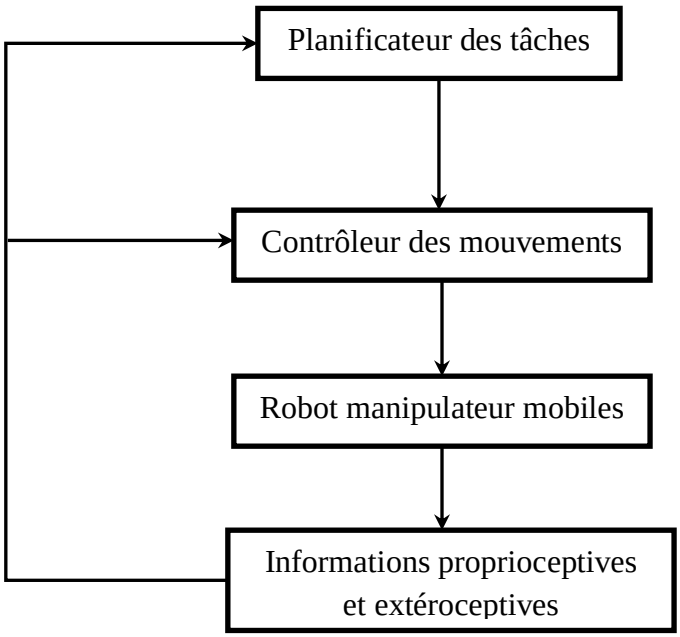


Figure.1.11. Structure fonctionnelle d'un contrôleur de robot.

À cause des nombreux avantages de simplification de la conception d'une commande dans la simulation, cette dernière est souvent choisie comme méthode d'analyse et validation qui permet d'avoir une idée générale sur l'efficacité de la commande. Toutefois, l'étude par simulation seule est contrebalancée par un défi fondamental qui représente la condition nécessaire pour qu'une simulation puisse être utile ; les résultats obtenus peuvent être reproduits dans la réalité ?

Les simulations ne peuvent pas refléter tous les phénomènes physiques car il est difficile, si ce n'est impossible, de les modéliser. Par ailleurs, des contraintes technologiques, dans la plupart des cas, ne sont pas prises en compte lors de la simulation comme les bruits de mesures et la résolution dues aux capteurs, le temps d'échantillonnage, les retards, les temps de traitement de données, saturations des actionneurs, frottements statiques, glissements, jeux mécaniques, les perturbations inconnus, etc. Si chacun de ces problèmes peut être résolu indépendamment, leur mise en simulation ajoute de complexité au problème posé. Or, la modélisation faite justement pour une simulation enlève l'aspect de cette complexité, de l'autre côté, une modélisation parfaite n'est qu'une illusion, ce qui affirme que le meilleur modèle du système réel est le système lui-même.

Ce travail de recherche a donc pensé d'aborder la vision d'une conception expérimentale dans le cadre de cette thèse, dans l'objectif de développer une architecture logicielle d'un contrôleur temps réel pour un manipulateur mobile, qui a guidé ou influencé le choix des approches de commande que nous allons présenter dans les chapitres suivants.

Afin de garantir le fonctionnement du système, le contrôleur doit accomplir une tache sous deux principales exigences indispensables ; la précision dans l'exécution et la rapidité de calcul en temps réel, En effet, le contrôleur de robot est un système temps réel qui a une réponse aux événements externes dans un temps typique. Les performances de contrôle de mouvement d'un robot ne dépendent pas seulement d'une loi de commande convenable, mais aussi d'un respect temporelles concernant son exécution pour éviter la discontinuité dans les mouvements. En fait, un calcul de la commande juste mais retardée donne une commande fausse, ce qui implique que la réponse d'une commande dépend également du temps auquel elle est produite.

La conception en robotique concerne à la fois les parties mécanique et électronique qui ne sont pas toujours le résultat d'une méthode très directe. Elles sont plutôt le résultat de nombreuses itérations de design. Ainsi, le prototype présenté dans la présente section est donc l'aboutissement d'un long processus.

Le manipulateur mobile (Figure.1.12) utilisé est composé de la plateforme à roues de type uni-cycle et du bras manipulateur série à trois liaisons pivotes. Pour fixer les idées, le poids de l'ensemble est d'environ 25 kg.

La plateforme de type uni-cycle est composé schématiquement de deux roues motrices indépendantes, et d'une roue folle non commandée à l'avant qui leur permet d'être stables en statique et de ne pas basculer en mouvement.

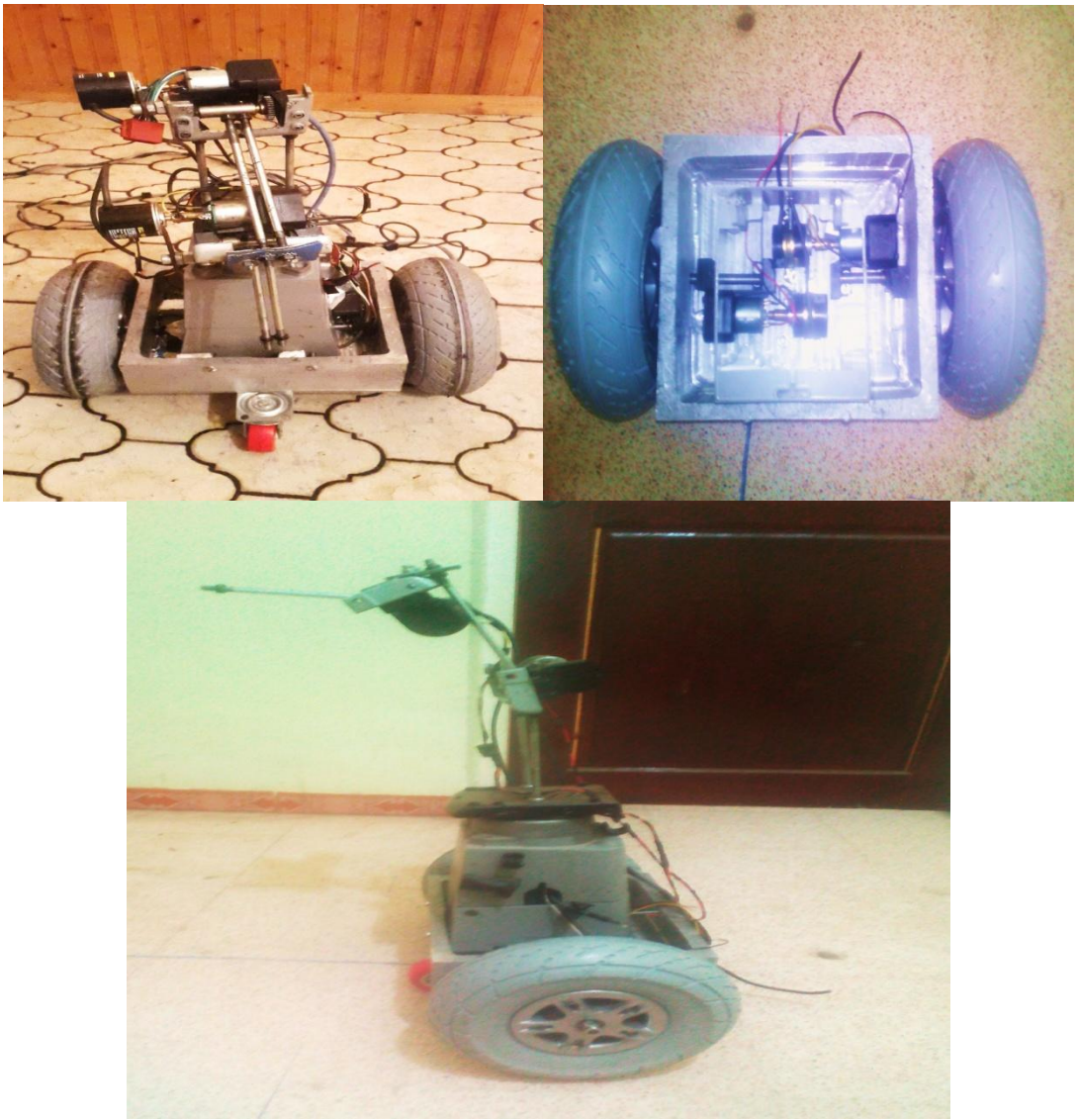


Figure.1.12. Robot manipulateur mobile uni-cycle.

Les mouvements sont réalisés grâce à des moteurs électriques à courant continu. Chacun des axes est équipé d'un capteur sur l'arbre de chaque moteur permettant de repérer sa position angulaire par rapport à la configuration d'origine.

Le système peut être résumé par le schéma-blocs présenté à la figure.1.13.

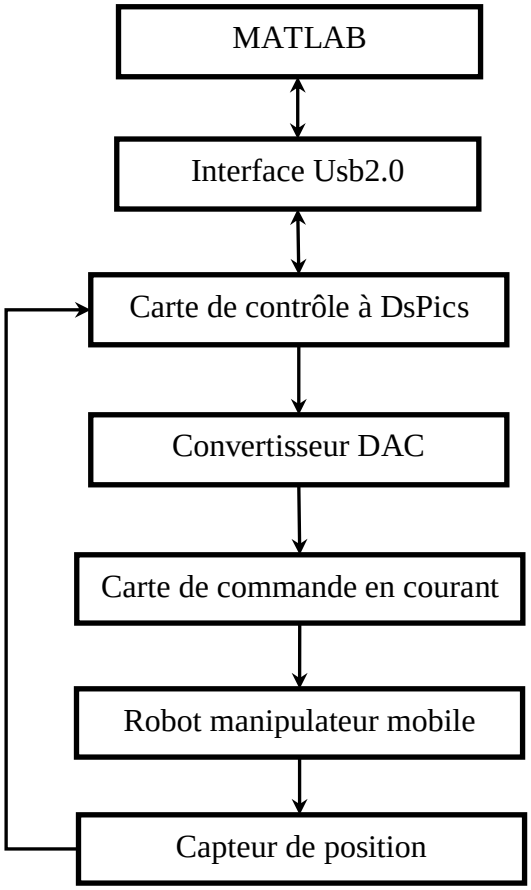


Figure.1.13. Schéma-blocs du système contrôle-commande.

L'architecture de la carte de contrôle est conçue sous forme parallèle sur plusieurs microcontrôleur de type dsPIC30f4011 (117Mhz), pour chaque actionneur un microcontrôleur permet de le commander et un microcontrôleur principal permet de gérer le protocole de communication entre eux. Chaque microcontrôleur est en mesure de traiter les données provenant des capteurs et de calculer la commande à envoyer au carte de puissance via des convertisseurs délivrant en analogique les signaux numériques codés sur 8 bits.

Dans les applications où la puissance est inférieure à quelques dizaines de watts, un petit moteur à courant continu à aimants permanents équipé d'un réducteur est souvent utilisé. C'est un moteur à excitation constante (aimant permanent), où la variable de commande est la tension appliquée à l'induit. En effet, les actionneurs sont des moteurs à courant continu et leur carte de commande présente un asservissement en courant, qui permet donc la commande en couple.

Le capteur sur chaque moteur est un encodeur en quadrature incrémental 1000 impulsions/tr qui permet de délivrer l'information sous forme numérique codée sur 2 bits en binaire et proportionnelle à la position et le sens de rotation de l'arbre du moteur.

Le logiciel utilisé comme interface d'acquisition est le MATLAB qui est un outil de calcul très répondu adapté pour les problèmes scientifiques. Avec, un ordinateur Dell portable Intel Core™2 duo à 2 GHz processeur, 2GB de mémoire vive et un système d'exploitation Vista édition familiale Basique a été utilisé dans notre expérimentation.

L'interface USB (full-speed12Mbits/sec) équipée d'un microcontrôleur PIC18f2550 (48Mhz), sert principalement d'interface entre le microcontrôleur principal et l'ordinateur. Elle est donc en mesure de recevoir au début de processus les données (par exemple l'ajustement des gains ou la mise à jour des paramètres) provenant du MATLAB et de les transmettre au microcontrôleur principal. D'autre part, elle est également en mesure de gérer l'acquisition de sorties au cours du processus (exemple ; position, vitesse, temps d'échantillonnage …etc.) et de les retransmettre à MATLAB pour les sauvegarder.

I.4. Problématiques et repères sur la commande des robots manipulateurs mobiles:

Afin de situer notre travail dans la littérature, nous proposons un rappel des études menées dans le cadre de la commande des robots manipulateurs mobiles selon un classement thématique. Intuitivement, il est impossible dans cette thèse, de faire une présentation exhaustive de différentes techniques, tant ce domaine recouvre de problématiques différentes.

Quand on parle de la navigation, relativement ou totalement autonome d'un manipulateur mobile en milieux partiellement connus voire complètement inconnus, cela réclame plusieurs problématiques qui doivent être adressées au niveau de contrôleur de robot ; des thématiques liées à la perception, la planification de tâche et la loi de commande. Cela signifie que le contrôleur doit être capable à la fois de percevoir correctement l'environnement, mais également de savoir comment réagir. Au niveau de planificateur de tâche, que le critère d'autonomie se présente dans la capacité de décision et de choisir des trajectoires pour accomplir correctement la tâche, même dans des situations inattendues et sans intervention humaine. D'autre part, au niveau de la commande, la capacité de générer un mouvement

associé à la trajectoire fournie par le planificateur est cruciale pour le succès de toute l'opération**.** Aussi, sans cette aptitude d'assurer l'exécution de la tâche, un robot ne sera pas en mesure d'accomplir les ordres fournis par le planificateur.

Dans la pratique pour des raisons de simplicité, on trouve souvent que les constructeurs des robots mobiles utilisent des lois de commandes qui sont basée seulement sur le modèle cinématique qui est plus simple que le modèle dynamique. Deux schémas se présentent quand on parle de la commande des moteurs d'un robot à partir des modèles cinématique et dynamique. Un moteur est commandé en vitesse lorsque la génération de consigne articulaire passe par le modèle cinématique seule où le modèle dynamique de robot est supposé négligeable devant la dynamique propre des moteurs.
En effet, les lois issues de ce principe apparaissent satisfaisantes pour certain application où les performances de haute précision ne sont pas nécessaires notamment en robotique mobile (une marge d'erreur de 1 cm peut être acceptable)
Un moteur est commandé en couple, lorsque la génération de consigne articulaire passe par le modèle cinématique et l'utilisation du modèle dynamique pour calculer le couple de consigne. En effet, ce principe apparait indispensable face à un problème de commande en position avec haute précision et notamment une commande en effort dont laquelle une force résistive doit être compensée avec précision.

Par conséquent, la première phase dans la synthèse d'une loi de commande basée sur la dynamique est la modélisation, qui consiste à rassembler les connaissances sur le comportement du système d'après une analyse théorique ou expérimentale des phénomènes physiques mis en jeu, ce qui nous conduit nécessairement à faire des hypothèses sur la structure de modèle qui peut suffisamment décrire ce comportement. Le meilleur modèle est celui qui conduit aux meilleures performances du système de commande, au sens du cahier des charges. À noter que particulièrement avec une loi de commande fondée sur le modèle dynamique d'un robot non holonome, les contraintes cinématique apparaissent de façon beaucoup plus explicites, et la commande ne peut plus être obtenue par application directe des techniques de commande utilisées pour les robots holonomes (exemple ; bras manipulateur).

Par ailleurs, dans la modélisation cinématique, l'obtention des paramètres géométriques ne pose pas vraiment de problème, par contre pour les paramètres dynamiques, il est nécessaire

de passer par une phase d'identification assez compliquée. D'autres problématiques de modélisations des robots mobiles à roues ont été formulées dans **[1]** où l'hypothèse de roulement sans glissements ne peut être conservée, et la détermination des efforts de glissement doit être traitée d'un point de vue dynamique.

Ainsi, construire un modèle qui décrive avec exactitude le comportement d'un manipulateur mobile à roues est une tâche difficile, comme ces systèmes mécatroniques sont un assemblage d'une variété d'éléments mécaniques, électriques, et pneumatiques dans le cas des suspensions par exemple.

À noter que notre objectif ici, n'est pas de faire une liste exhaustive de l'ensemble des méthodes de commande utilisées en robotique mais plutôt de présenter les grandes ligne sur les techniques qui sont basées sur l'analyse de modèle dynamique plutôt que le modèle cinématique seule.
Quant à la problématique associée à la commande des manipulateurs mobiles, outre les travaux liés à la modélisation des bras manipulateurs **[2-4],** et la modélisation des robots mobile **[1,5]**, à la planification de mouvements pour les bras manipulateurs **[6-7]**, et à la planification de mouvements pour les robots mobiles **[5,8]**, nous recommandons des ouvrages de synthèse pour la commande dynamique classique des robots manipulateurs **[10-11]** et des robots mobiles **[12-13]**.

Après une recherche rapide dans l'abondante littérature sur le sujet, nous pouvons constater que contrairement à l'automatique linéaire, en générale, l'automatique non linéaire ne donne pas de solutions universelles ni pour l'analyse ni pour la conception de la loi de commande.
En effet, à ce stade, la plupart des travaux dans la littérature proposent des démarches qui sont limitées à des classes bien particulières de systèmes ou une famille particulière tout en respectant quelques hypothèses et des suppositions imposées sur le système.

Il est naturel que la plupart des systèmes soient non linéaires et plus particulièrement les systèmes non holonomes. Il est donc important de pouvoir élaborer une commande non linéaire pour rendre compte le comportement global du système, non seulement autour de ses points de fonctionnement habituels comme le cas des systèmes linéarisés, mais également lors des passages d'un point de fonctionnement à un autre **[14]**. La complexité de telle loi de commande est proportionnelle à la complexité du système à contrôler.

La commande d'un tel système est donc un problème qui, pour être résolu de façon satisfaisante, doit prendre non seulement ses non linéarités en considération mais aussi des perturbations imprévues non négligeables doivent être compensées notamment lorsque par exemple la plateforme mobile remplit d'autres tâches qu'un simple déplacement sur un terrain bien structuré.
De point de vue robustesse, les paramètres mal connus, les dynamiques inconnues et négligées par une modélisation simplifiée et les perturbations externes peuvent influer sur la stabilité du système et provoquer une dégradation des performances, particulièrement en utilisant une loi de commande basée modèle. Dans ce sens, le problème de la commande des systèmes robotique a largement été abordé dans la littérature.

Au cours des dernières années, une variété de lois de commande dites classiques bien connues telles que des techniques de commande adaptatives, robustes et optimales ont été proposées pour le contrôle des systèmes robotiques.
La démarche commune dans le développement d'une loi de commande optimale pour un système non linéaire est d'assumer que la dynamique non linéaire est exactement connue.
Motivé par le désir d'éliminer l'exigence d'une connaissance exacte de la dynamique, R. Johansson **[15]** a développé l'un des premiers résultats pour illustrer l'interaction du contrôle adaptatif avec la commande optimale basée sur la solution de l'équation de Hamilton-Jacobi-Bellman (HJB) pour le contrôle du suivi des trajectoires des systèmes robotiques.

Plus précisément, Johansson a d'abord utilisé la linéarisation par compensation pour éliminer la dynamique non linéaire et calculer la commande optimale, ensuite, une loi de commande adaptative a été développée dans la présence d'une incertitude paramétrique dans le système. L'analyse dans **[15]** a indiqué que si l'erreur d'estimation de paramètre pouvait en quelque sorte converger à zéro, la commande converge vers la solution optimale.

Dawson et al. **[16]** a conçu une méthode de contrôle optimale modifiée en combinant à la fois la commande par couple calculé et la commande optimale où une autre solution de l'équation HJB a été proposée.
Un autre indice de performance a été utilisé dans **[17]** pour calculer une commande optimale qui permet de minimiser le temps de mouvement et augmenter les performances de suivi de trajectoire où des résultats expérimentaux ont été présentés. Le travail dans **[18]** présente une commande optimale pour une classe de robot à roues en minimisant la norme de la

commande à appliquer, en tenant compte des contraintes non-holonomes et du sous-actionnement.
Récemment Zhou and Wang **[19]** ont développé une commande optimale afin de minimiser que possible l'angle d'inclinaison des systèmes à pendules inversés (robot auto-équilibré) à roues avec retard sur l'entrée en minimisant un critère de performance quadratique.

Généralement, la difficulté essentielle après l'exigence d'un modèle dynamique pour le calcul d'une commande optimale réside dans la solution non explicite de l'équation HJB qui représente un problème non trivial notamment dans les applications temps réel.

En présence d'une dynamique non modélisée, des sources d'incertitude et encore des perturbations externes, les systèmes de contrôle robustes et adaptatifs sont les moyens les plus efficaces et les plus populaires pour gérer ce genres de problématique sur le modèle.
Ces approches ont suscité une attention croissante, pour la simple raison que dans la pratique, la plupart des systèmes robotiques ont des caractéristiques dynamiques inconnues a priori (Structure, paramètres) et/ou impliquent des incertitudes telles que les perturbations imprévues, par exemple la dynamique due à l'irrégularité du terrain dans certains cas d'applications, et il existe toujours une différence entre les modèles mathématiques utilisés pour la conception de la loi de commande basée modèle et le système réel.

Typiquement, un système de contrôle conçu avec succès devrait être toujours capable de maintenir la stabilité et certain niveau de performance. Par conséquent, le besoin d'un système de contrôle robuste, dispose une capacité d'adaptation et une convergence rapide pour les robots manipulateurs mobiles devient une nécessité notamment dans un environnement où les perturbations sont non négligeables. .

La stratégie de contrôle adaptatif directe a la capacité d'adaptation en temps réel. Cependant, ses performances de contrôle peuvent être affectées dans un cas réaliste par certaines incertitudes non structurées comme les perturbations externes. Une méthode de contrôle robuste peut être meilleure en rejetant les perturbations et en compensant les incertitudes non structurées.
Pour surmonter ces défauts et profiter à la fois les caractéristiques des méthodes de commande robustes et adaptatives. Une multitude de stratégies de contrôle adaptatif robustes sont proposées, et la problématique de la conception de contrôleurs adaptatifs robustes

valable pour les applications temps réel pour éliminer les effets de tout type de perturbations arbitraires est l'un des sujets difficiles.

En particulier, il y a eu des recherches liées au développement d'algorithmes classiques de contrôle robustes adaptatifs basés modèle pour les robots manipulateurs et les robots mobiles non holonomes. Dans la plupart de ces approches pour cette classe de systèmes de robots, une borne supérieure scalaire constante connue ou inconnue est généralement considérée pour les incertitudes du système, la dynamique inconnue et les perturbations externes.

Historiquement, Slotine et Li **[20]** ont proposé un algorithme robuste adaptatif, qui consiste en une commande PD linéaire et une compensation des incertitudes paramétriques d'un robot manipulateur.
Spong **[21]** a présenté une commande non linéaire robuste similaire à l'algorithme adaptatif développé par Slotine **[20]**. Ici, l'approche de Leitmann **[22]** a été utilisée pour concevoir la commande robuste dans laquelle, la borne supérieure de l'incertitude paramétrique doit être connue a priori. La même année, Dawson et al. **[23]** ont proposé une commande adaptative robuste avec un compensateur supplémentaire dispose une loi adaptative pour estimer la borne supérieure de la perturbation associée aux frottements dans le système.
Koo et Kim **[24]** présentent une loi robuste pour les robots manipulateurs avec incertitude paramétrique. La structure de contrôle de base est la même que celle de l'étude de Spong **[21]**, mais la loi de commande proposée ne nécessite aucune information préalable sur la limite supérieure de l'incertitude.
Man Zhihong **[25]** a proposé un schéma de contrôle robuste utilisant un compensateur à structure variable dans le système en boucle fermée pour un manipulateur avec une dynamique incertaine. Il a démontré que le système en boucle fermée est insensible aux incertitudes du système et à certaines perturbations externes. Mais, à cet effet, la limite supérieure de l'incertitude devrait être connue a priori. Le même auteur dans **[26]**, présente une loi de commande adaptative robuste par modes glissant pour les manipulateurs robotiques dans la présence d'une perturbation d'entrée bornée arbitraires. Il a démontré qu'un mécanisme adaptatif peut être introduit pour estimer la limite supérieure de la norme du vecteur de perturbation
Des années plus tard, dans **[27]**, aussi une approche adaptative robuste est conçue pour compenser les incertitudes issues des paramètres inconnus du robot. Cette loi dispose une fonction adaptative pour estimer la borne supérieure de l'incertitude paramétrique. La loi de

contrôle proposée a été élaborée en basant sur les approches de Spong **[21]** et Koo et Kim **[24]**.

Ainsi, S.Torres **[28]** présente une technique différente pour le contrôle des robots manipulateurs en présence d'incertitudes paramétriques ou des dynamiques inconnues. La méthode combine une loi d'adaptation avec la commande robuste bien connue proposé par Spong **[21]**. La nouvelle stratégie ajoute un schéma d'auto réglage afin de varier de manière adéquate la valeur du paramètre dans la partie robuste où, la règle classique de gradient descendante est utilisée pour l'adaptation.

En suivant la même stratégie, d'autres études de recherche dans **[29-35]** ont montré que la connaissance préalable de la borne supérieure des perturbations n'est pas requise dans la conception d'une loi de commande robuste.

Un point approprié de ces techniques de commande est qu'elles peuvent compenser relativement les effets des perturbations, à condition qu'une connaissance a priori du modèle de système soit disponible pour le calcul de cette commande. En pratique, cependant, la connaissance d'un modèle exact du robot mobile à roues non holonome est une tâche difficile, ce qui implique que cette hypothèse ne semble pas être pratique pour les applications réelles.

La commande des systèmes non linéaires utilisant des techniques intelligentes modernes, telles que les réseaux neuronaux et la logique floue a reçu une grande attention au cours des dernières années, et a connu un développement rapide conduisant à des performances considérables pour une large classe de systèmes non linéaires. La motivation derrière l'invention des techniques basées sur les systèmes neuronales et les systèmes flous pour la conception des contrôleurs n'était d'ailleurs pas toujours de mieux copier la réalité biologique et le comportement humain, mais plutôt pour les rendre plus faciles à programmer.

En effet, la conception des contrôleurs intelligents adaptés aux applications en temps réel pour les systèmes robotiques est l'une des défis les plus rencontrés dans le domaine de contrôle.

Les incertitudes, les dynamiques non modélisées et les différentes contraintes imposées sur la modélisation du processus peuvent être compensées par les contrôleurs flous avec l'obtention de meilleures performances **[36-41]**. Ces contrôleurs ont connu beaucoup de succès et devenus un sujet principal dans le domaine de la recherche des systèmes intelligents.

D'autre part, les réseaux de neurones (NN) constituent également des candidats potentiels pour ce genre de problématique de contrôle des systèmes robotiques. En effet, la capacité d'approximation des réseaux neuronaux est une propriété clé, qui permet d'approcher toute fonction non linéaire. C'est cette propriété qui motive leur utilisation pour la conception de systèmes de contrôle non linéaire, et habituellement utilisé pour l'identification du système ou le contrôle basé sur l'identification **[42-45]**. Ainsi, des efforts de recherche ont été menés sur le contrôle des mouvements de robots mobiles non holonomes utilisant des approches de commande basées sur le réseau de neurones.

À titre d'exemple, dans **[46]** une technique de commande neuronale adaptative robuste est appliquée sur un robot mobile non holonome caractérisée par des incertitudes paramétriques et non paramétriques. Park et al. **[47]** ont développé une commande par mode glissant et système neuronal adaptatif pour les robots mobiles à roues non holonomes avec des incertitudes de modèle et des perturbations externes.

D'autre part, des approches récentes considéraient le contrôle des systèmes non linéaires en combinant particulièrement la commande optimale et les réseaux neuronaux afin d'améliorer les performances.
Kim et al. **[48-49]** ont développé une approche basée sur le NN et la commande optimale pour les robots manipulateurs, dans laquelle la partie optimale a été appliquée pour minimiser un critère quadratique de sorte que le robot suit une trajectoire de référence. Le réseau de neurones a été utilisé pour compenser de façon adaptative les incertitudes, les dynamiques non modélisées et les perturbations externes.

Différemment, Cheng et al. **[50]** ont présenté une commande optimale pour les systèmes non holonomes en utilisant les NN pour résoudre approximativement l'équation HJB variable dans le temps. La même technique a été étendue plus tard dans **[51]** aux systèmes non linéaires affines.
Une solution présentée dans Ornelas-Tellez et al. **[52]** qui combinent la commande optimale inverse et un système neuronal ajusté en ligne avec le filtre de Kalman pour éviter de résoudre l'équation HJB.

Indépendamment des différentes approches présentées, ces lois de commande ont seulement montré des performance dans la plupart des cas par simulation en supposant que les couples appliqués sont directement injectés sur le système robotique sans tenir compte les dynamiques des actionneurs. Autrement dit, on considère que le sous-système électrique pour chaque actionneur fourni parfaitement le courant associé au couple de la commande calculée. En effet, une loi de commande qui traite l'effet de couplage entre le système mécanique et le système électrique représente un autre chalenge.
Le problème de contrôle des robots mobiles non holonomes impliquant des incertitudes sur les sous-systèmes électriques est encore un problème ouvert et a été étudié dans des travaux très récents **[53–59]**.

Finalement, pour clôturer la problématique de la commande, un autre point s'impose. Ajuster les paramètres d'une loi de commande non linéaire, représente l'un des défis le plus rencontrés dans la conception pratique des contrôleurs de mouvement pour les systèmes robotiques. En effet, trouver les paramètres optimaux d'une loi de commande n'est pas une chose aisée dans la pratique.

Dans ce sens, plusieurs travaux ont essayé d'améliorer la performance d'une loi de commande en utilisant l'intelligence artificielle qui essaye de simuler le raisonnement humain ou la nature. Outre l'utilisation directe de la logique floue et le réseau neurones dans la boucle de commande principale, les travaux de recherche ont également montré un aspect diffèrent de l'exploitation d'un système flou ou neuronal comme un processus auxiliaire pour ajuster les paramètres d'une loi de commande donnée **[60-61].** D'un autre côté, d'autres algorithmes d'optimisation aussi de l'intelligence artificielle permettent de traiter le problème de réglage des paramètres ; sont les méta-heuristiques **[62-68]**.

Particulièrement, dans la littérature, le problème d'optimisation populaire pour les systèmes robotiques est la planification de trajectoire pour la navigation des robots mobiles. De nombreuses techniques et plusieurs stratégies ont été employées, à savoir, les algorithmes génétiques (AG) **[69]**, des Essaims de Particules (PSO) **[70-71]**, la colonie des abeilles **[72],** la colonie de fourmis **[73]**, etc.

Indépendamment de l'application et sans perdre de généralité, ces méthodes peuvent être classées en deux approches d'optimisation ; à savoir, optimisation hors ligne avec exploration globale et l'optimisation en ligne avec exploration locale pour atteindre une performance

optimale où, dans ce dernier cas, ils sont qualifiés pour les implémentations en ligne si le temps de calcul est assez faible.

En général, deux notions d'efficacité sont rencontrées en méta-heuristique ; la vitesse de convergence et la précision. La vitesse est souvent mesurée avec le nombre d'itérations nécessaires pour obtenir la solution souhaitée. La précision se réfère à la différence entre l'optimum trouvé par la méta-heuristique et l'optimum souhaité du point de vue de la qualité de la solution ou du cahier de charge.

Les algorithmes des méta-heuristiques présentés dans la littérature ont obtenu de bons résultats dans le cas d'une optimisation en ligne avec une exploration locale ou d'une optimisation hors ligne avec une exploration globale. Toutefois, le cas d'une optimisation en ligne et en temps réel avec une exploration globale n'a pas été traité.

En dernière analyse, l'utilisation des méta-heuristiques peut sembler relativement simple, mais la mise en œuvre de la théorie dans la pratique est souvent difficile et n'est pas un problème trivial.

D'une part, dans la méthode basée sur l'optimisation hors ligne, il est nécessaire d'avoir un modèle nominal qui peut représenter la dynamique du système pendant la simulation. Cependant, il est très difficile, voire impossible, de trouver de tels modèles pour des systèmes complexes ou hybrides. Or, il faut toujours garder à l'esprit que nous avons imposé dans cette thèse l'hypothèse que nous ne disposons pas de modèle du système.

D'autre part, dans la méthode basée sur l'optimisation en ligne, les algorithmes de méta-heuristique dans la littérature reposent tous sur le même principe. Dans un espace de recherche bien limité (valeurs min et max), et sur la base d'une solution initiale bien définie à l'avance et considérée comme un point de départ, rechercher une solution dans le voisinage et accepter cette solution si elle améliore la solution actuelle. Par conséquent, le premier problème imposé dans la conception est que, les limites min et max de l'espace de recherche et les valeurs initiales des paramètres à optimiser sont d'abord déterminées souvent par la méthode expérimentale Essais-erreurs, afin d'éviter le danger d'endommager le système réel et obtenir une évolution rapide de l'algorithme vers les meilleures solutions dans cet espace de recherche.

En pratique, seul le savoir-faire et l'expérience de l'utilisateur permettent de pallier ce défi, et quoi qu'il en soit, le résultat reste sensible en premier lieu à l'espace de recherche choisi et notamment aux paramètres initiaux ou aux conditions initiales.

Enfin, tous les algorithmes méta-heuristiques quant à eux, ont des paramètres dont le réglage n'est pas nécessairement trivial et la qualité des solutions trouvées par les méta-heuristiques dépend de leur paramétrage.

I.5. Contributions de la thèse :

Avant d'aborder nos travaux d'étude en détails dans les prochains chapitres, les contributions de la thèse à la commande des robots manipulateurs mobiles sont rapportées dans cette section. Chacune des études a suivi une présentation méthodique de mise en œuvre et de la validation lors de simulations et d'essais expérimentaux dans le cadre de suivi de trajectoire.

La stratégie adoptée dans notre travail repose sur la démarche de garder à l'esprit que la synthèse de la loi de commande doit répondre aux exigences suivantes :

- la loi de commande permet d'assurer une bonne précision de suivi de trajectoire avec un minimum d'énergie possible,
- la loi de commande doit être indépendante du modèle dynamique ou non basée modèle,
- la loi de commande autorise une adaptation en temps réel continue face à d'éventuels changements des paramètres du système,
- la loi doit être robuste et insensible aux perturbations externes imprévues,
- la loi permet de tenir compte du couplage intrinsèques entre la dynamique mécanique et électrique et des contraintes non holonomes,
- avoir un mécanisme autonome en temps réel pour le réglage de différents paramètres de la loi de commande, la rapidité de convergence, la capacité de calcul en temps réel et ne nécessite pas un effort d'un expert pour leur paramétrage,
- avoir une souplesse de s'étendre sans modification majeure pour d'autres systèmes non linéaires,

En partant de cette constatation, nous avons créé une structure de contrôle efficace qui peut faire face à un certain niveau de complexité. Notre architecture de contrôle proposée est inspirée des différentes techniques proposées dans la littérature. En effet, nous avons adopté une approche hybride intégrant une partie de la commande optimale, la commande neuronale, la commande robuste adaptative, un système flou et les algorithmes de méta-heuristique.

En premier lieu, nous avons commencé par exploiter l'approche de la commande optimale, plus particulièrement celle basée sur la résolution d'un problème quadratique de l'erreur en question et de l'énergie. Une solution de l'équation HJB a été proposée où la synthèse de cette loi de commande est plus simple au sens qu'elle ne nécessite pas de résoudre un système d'équations implicites.

D'autre part, il est clair que pour implémenter une loi de commande optimale, il est nécessaire de pouvoir s'appuyer sur un modèle représentatif du système. Or nous avons fait ici l'hypothèse que nous ne disposons pas de modèle du système.
Les caractéristiques de l'approximation et de l'apprentissage des réseaux de neurones, nous donnent la solution pour éviter la phase de modélisation et l'identification classique. Toutefois, souvent dans la littérature, les systèmes de réseaux de neurones proposés doivent être utilisés dans deux phases successives avec un processus d'apprentissage qui ne s'adapte pas à la contrainte imposée de l'adaptation en temps réel continue. Pour pallier ce problème, il faut nécessairement procéder à une estimation rapide, voire en temps réel, de la dynamique non modélisée. En effet, les incertitudes, les dynamiques non modélisées et les différentes contraintes imposées sur le modèle peuvent être compensées par un réseau de neurones adaptatif dans lequel ces paramètres seraient ajustés en permanence pendant l'utilisation du système.
Par ailleurs, l'erreur d'approximation de réseau de neurones et les différentes perturbations externes non négligeables peuvent être compensées par une nouvelle loi de commande adaptative robuste.
Indépendamment des nombreuses approches robustes des systèmes robotiques présentées dans la littérature, les résultats de recherche existants ont seulement considéré un terme scalaire pour estimer la borne supérieure des perturbations, ce qui est certainement insuffisant pour éliminer simultanément les effets de chaque perturbation arbitraire pour chaque sous-système.

Ainsi, une estimation inadéquate des perturbations d'entrée peut entraîner des performances indésirables (vibration dans les mouvements du robot) du système dans la pratique et notamment au niveau de la saturation des actionneurs. Or une valeur optimale d'un simple scalaire peut ne pas être facilement obtenue en raison de la complexité de la structure des incertitudes, et notamment lorsque le système nécessite une grande précision avec un minimum d'énergie.

Pour faire face à cette lacune, nous avons proposé une nouvelle loi adaptative de la borne supérieure sous forme matricielle, de sorte que chaque perturbation locale pour chaque sous-système est supposée découplée et bornée localement. De cette façon, dans l'objectif d'optimiser les performances de la commande robustes, nous utilisons une loi adaptative pour estimer les bornes supérieures des perturbations de chaque sous-système à la fois.
Ainsi, l'effet de couplage entre la dynamique mécanique et électrique a été traité de la même manière. À cet effet, nous avons proposé une nouvelle loi de commande qui tient compte de la dynamique électrique pour les manipulateurs mobiles actionnés par des moteurs électriques.

Finalement, l'ajustement systématique des paramètres des lois de commande pour des systèmes complexes constitue un problème souvent délicat lors de l'élaboration d'une structure de commande. En effet, notre travail entre ainsi dans ce cadre pour répondre à cette problématique, une des contributions de cette thèse est le développement d'un algorithme autonome basé sur la logique floue et la méta-heuristique pour le réglage des paramètres d'une loi de commande en temps réel, en prenant en compte ainsi la contrainte sur la saturation des actionneurs.

Cet algorithme qui met en évidence la possibilité d'hybridation entre un système flou et la méta-heuristique est inspiré du principe de l'algorithme de colonie de fourmi. Ce dernier a été modifié et développé pour une implémentation en temps réel, et il dispose une exploration continue dans l'espace de recherche globale et un temps de convergence rapide qui s'adapte aux contraintes du calcul en temps réel. La population des fourmis commence la recherche à partir de l'origine où les paramètres de la loi de commande sont définis comme zéros, de cette façon, nous n'avons pas besoin de conditions initiales appropriées. Quant au paramétrage de l'algorithme, son choix dépends de la fonction objectif à minimisée, de sort que tous les paramètres sont fonction des erreurs en question (erreur de position et de vitesse, articulaires

ou opérationnelles) et les valeurs de la saturation des moteurs, ainsi donc, ne demande pas un effort de savoir-faire par l'utilisateur.

Bien que l'algorithme qui sera présenté plus tard, est applicable quelle que soit la structure de la loi de commande. Ainsi, les lois de commande développées dans cette thèse peuvent également être utilisées pour d'autres systèmes non linaires, de façon plus ou moins directe selon le système considéré.

Sur le plan pratique avec l'optique d'une mise en œuvre expérimentale, par rapport à notre contribution qui impose la contrainte temps réel et à l'issu des résultats présentés dans cette thèse, un robot manipulateur mobile à deux roues a été spécialement conçu durant les années de thèse pour la validation temps réel des approches de commande proposées que nous allons présenter dans les prochaines chapitres.

Outre cette introduction qui fait office à la fois de motivation et de présentation générale de la problématique et nos contributions, le reste de cette thèse se compose de quatre chapitres organisés comme suit :

En premier lieu, le chapitre 2 rappelle certaines généralités sur la modélisation des manipulateurs mobiles. Ainsi, une modélisation cinématique et dynamique seront présentées.
Afin de répondre à la problématique, le chapitre 3 développe les deux lois de commande proposées, à savoir, la commande neuro optimale robuste adaptative et la commande neuro robuste adaptative pour les robots électriques.
Ainsi, dans le chapitre 4, nous allons présenter notre nouvelle approche intelligente pour le réglage des paramètres des lois de commande qui est basée sur une hybridation entre le système flou et les colonies de fourmis.
Enfin, nous terminerons ce travail par une conclusion reprenant les principaux résultats présentés.

Chapitre II.

Modélisation des manipulateurs mobiles.

II.1. Introduction

Un système complexe tel que le manipulateur mobile peut être considéré comme un générateur de mouvements et d'efforts dans l'espace est en effet conçu pour remplir des missions qui nécessitent des capacités de locomotion et de manipulation. Tandis que ce système n'est pas directement télé-opéré et doit posséder une certaine autonomie d'action, leur contrôleur automatique doit connaître et maîtriser son mouvement et ses efforts puisque l'opérateur humain n'est pas directement dans la boucle d'asservissement. Ainsi, il est nécessaire de passer par une modélisation mathématique qui représente les relations entre les entrées et les sorties du processus à commander par des équations. En d'autres termes, quand on modélise un système, on est capable de prédire son comportement lorsqu'on le soumit à des entrées connues.

Avant de procéder à la synthèse d'une loi de commande, la modélisation est une étape clé qui nous donne une vue claire sur le comportement du système et ainsi donc un premier pas vers l'analyse et la commande.

En termes de modélisation, nous nous attachons à présenter les relations entre les vitesses du système qui représentent un modèle cinématique et les relations entre les vitesses, les accélérations et les forces du système qui représentent le modèle dynamique. Ce dernier, s'il n'est pas forcément utilisé directement dans la loi de commande, est nécessaire à la compréhension de son comportement et leurs données peuvent être utilisées indirectement à des fins de synthèse des lois de commande, ou encore pour la simulation, la détection d'anomalies de fonctionnement et le diagnostic de pannes, etc.,

II.2. Modèle cinématique :

On considère le cas d'un manipulateur mobile constitué d'une plateforme mobile uni-cycle à roues équipée d'un bras manipulateur.

Le premier point dans la construction d'un modèle consiste à déterminer un ensemble de variables permettant de repérer la configuration de l'effecteur du manipulateur dans l'espace. Ces variables sont appelées coordonnées généralisées. Une étude cinématique traite le mouvement des corps en regardant la géométrie ou les repères associés au cours du mouvement, mais sans tenir compte des forces qui le génèrent et indépendamment des détails de conception mécanique et des limites des actionneurs.

a. Modèle d'un bras manipulateur :

Un robot manipulateur est constitué par l'assemblage de corps articulés. La liaison cinématique entre deux corps fait limiter les degrés de mobilité de chaque corps. Dans l'assemblage de bras manipulateur, on peut imaginer diverses combinaisons de translations (la liaison prismatique) et de rotations (la liaison pivote) définies une chaine cinématique.

Nous considérons la structure articulée à n corps rigides représentée indépendamment du type de la base mobile sur la figure 2.1. Cette structure est holonome se compose d'une chaine cinématique montées en série. Ainsi, elle est référencée dans un système de coordonnées orthonormé R_0 situé à son origine et constitue un système dont les entrées correspondent aux coordonnées généralisées$\mathbf{q_i(t)}$, i =1,.., n et dont les sorties correspondent aux positions et orientations de tous les repères $\mathbf{R_i}$ des corps successifs $\mathbf{S_i}$ référencés dans $\mathbf{R_0}$. Ces sorties sont assimilées à la posture **x(t)** de l'effecteur E référencée dans $\mathbf{R_0}$.

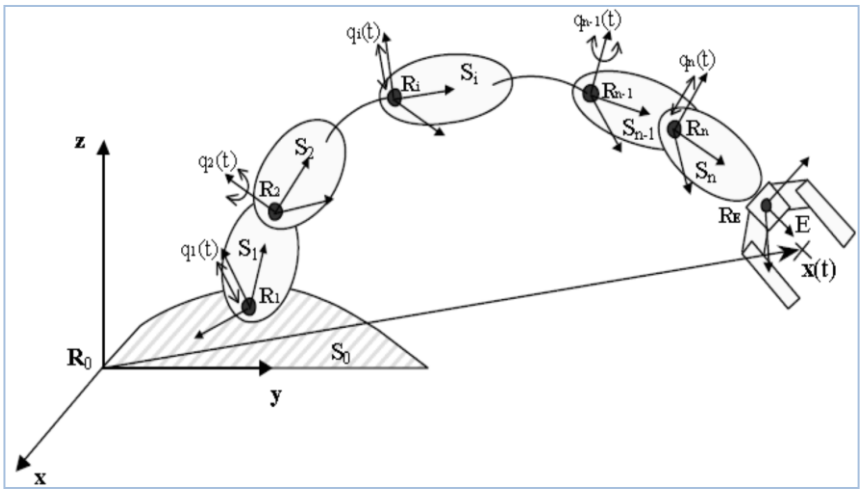


Figure.2.1. Système mécanique articulé sériel.

En effet, la variable X(t) appelé **vecteur des coordonnées opérationnelle**, qui caractérise la position et l'orientation de l'effecteur du robot dans l'espace, est déterminée par le vecteur des coordonnées articulaires généralisées **q(t).**

Le modèle géométrique direct d'un robot manipulateur est la fonction **X** qui permet d'exprimer la situation de l'organe terminal du robot manipulateur (position et orientation) en fonction de sa configuration **q**. Cette fonction est définie par **[2]**:

$$\begin{aligned} &\mathbb{R}^n \rightarrow \mathbb{R}^m \\ &q \quad \rightarrow X(q). \end{aligned} \tag{2.1}$$

La méthode systématique qui permet de calculer la fonction X(q) est celle de Denavit-Hartenberg modifiée **[2]** qui facilite la description géométrique du manipulateur, cette dernière nous permet d'aboutir au modèle cinématique du robot et offre une souplesse dans le calcul du modèle dynamique.

La représentation de DH (ou MDH) consiste en l'utilisation d'une transformation homogène T qui représente le système de coordonnées de chaque liaison par rapport au système de coordonnées de la liaison précédente. À travers une séquence de transformations successives d'un repère relativement à un repère précédant, on peut ramener le système de coordonnées attaché à l'élément terminal $\{R_n\}$ au système de coordonnées attaché à la base $\{R_0\}$ **[2]**

$$ {}^0T_n = {}^0T_1(q_1)\ {}^1T_2(q_2) \dots {}^{n-2}T_{n-1}(q_{n-1})^{n-1}T_n(q_n) = \begin{pmatrix} & {}^0R_n & & {}^0P_n \\ 0 & 0 & 0 & 1 \end{pmatrix}, \quad (2.2) $$

où ${}^0R_n$ Matrice de rotation et ${}^0P_n$ vecteur de position cartésienne.

Une application réciproque qui permet de calculer les variables articulaires q directement contrôlable à partir d'une situation désirée $X(q)$ s'appelle le modèle inverse principalement traité au niveau de planificateur de tâche

$$ \begin{aligned} &\mathbb{R}^m \to \mathbb{R}^n \\ &X \quad \to q(t). \end{aligned} \quad (2.3) $$

Une étude qui en dérive directement de modèle géométrique, dite étude cinématique ou différentielle, elle s'intéresse plus précisément au comportement du robot en vitesse.
La vitesse opérationnelle de l'effecteur est formalisée par la dérivé directe de modèle géométrique ce qui permet d'établir une relation entre le vecteur $\dot{X}$ en position et orientation, et le vecteur $\dot{q}$ des vitesses articulaires. Le modèle cinématique peut être par sous forme matricielle **[2]**

$$ \dot{X} = \begin{pmatrix} V \\ \omega \end{pmatrix} = J(q)\dot{q}\ , \quad (2.4) $$

avec $J\ (q) \in \mathbb{R}^{6\times n}$, appelée la Jacobéenne du manipulateur.

Cette relation signifie que lorsque les actionneurs déplacent les différents corps du manipulateur à des vitesses $\dot{q}$ constantes dans l'espace articulaire, la vitesse instantanée vue dans l'espace opérationnelle dépend de la configuration articulaire q du manipulateur.
Le modèle cinématique inverse est nécessaire quand on définit la vitesse de l'effecteur dans le repère de la base pour en déduire les vitesses articulaires directement asservies. Le problème revient donc à inverser la matrice Jacobéenne J.

b. Modèle d'une plateforme à roues uni-cycle :

On se place dans le cadre d'un mouvement plan contrairement au bras manipulateur, une plateforme uni-cycle à roue évoluant sur un plan horizontal, peut être considérée comme un objet rigide se déplaçant avec deux degrés de liberté. Ainsi, l'état du système est complètement caractérisé par la connaissance de par la position et l'orientation d'un repère mobile situé généralement au centre de l'axe des roues motrices G relativement à un repère fixe R (O,$\vec{x}$,$\vec{y}$,$\vec{z}$) (Fig.2.2).

On peut donc définir le vecteur d'état (x, y,θ), où x et y sont respectivement l'abscisse et l'ordonnée du point G et l'orientation est repérée par l'angle θ autour de l'axe verticale $\vec{z}$.

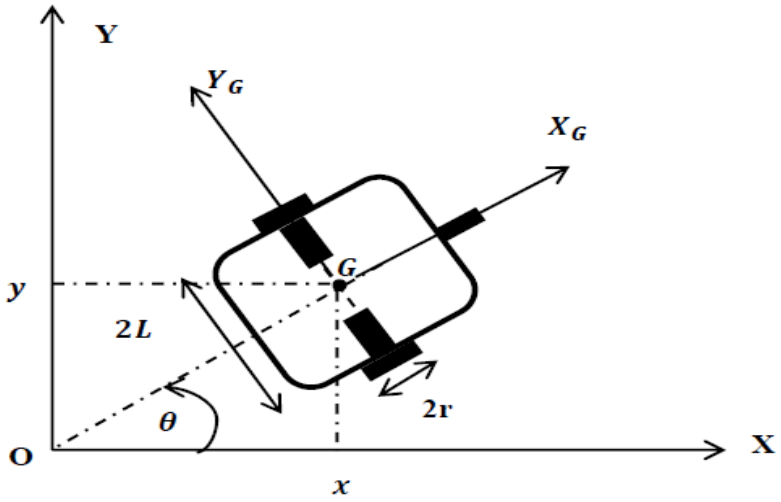


Figure.2.2. Robot mobile uni-cycle.

Les caractéristiques d'une roue et le nombre de paramètres nécessaires à sa description est dépendant du type de roue considéré.

Afin de simplifier la procédure, des hypothèses doit être prisent en considération ; la plateforme mobile est considérée comme un corps rigide évoluant dans un plan horizontal et équilibré en statique; les roues sont supposées indéformable et de rayon ***r*** constant, chaque contact roue/sol est réduit à un point ; les roues roulent sans glisser et sans pivoter sur le sol.

Pour faire l'étude cinématique nous modéliserons la roue comme indiqué sur la figure et on s'intéresse ici au contact ponctuel à l'instant **t**, dans un référentiel fixe ***R*** entre la roue et le sol.

Considérons une roue de rayon **r** roulant sur un support fixe horizontal. On appelle **P** le point coïncidant à l'instant **t** avec le point **S** de contact entre la roue et le sol, dont le centre de masse est noté **C.**

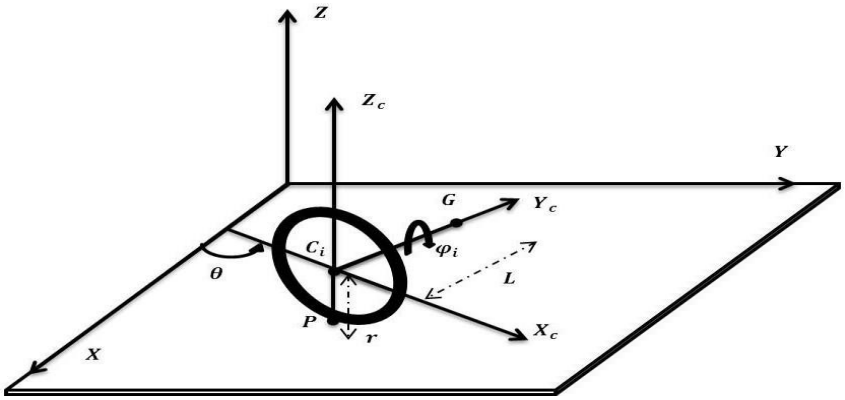


Figure.2.3. Représentation d'une roue dans le plan cartésien.

La relation de Varignon qui donne la composition des vitesses linéaires de point P et C_i appartenant à la roue supposée rigide s'écrit :

$$\vec{V}_P = \vec{V}_{C_i} + \vec{\omega} \Lambda \overrightarrow{C_i P}, \tag{2.5}$$

où $\vec{\omega}$ est le vecteur rotation instantanée de la roue dans le repère R.

On note $\dot{\varphi}_i$ la vitesse angulaire de la roue droite ($i = d$) ou gauche ($i = g$) qui permet de paramétrer la rotation propre de la roue dans son référentiel $\boldsymbol{R_C}$, qui est en rotation d'angle θ par rapport à $\boldsymbol{R}$.

D'après la loi de composition des vitesses de rotation

$$\begin{aligned} \vec{\omega}_{roue/R} &= \vec{\omega}_{roue/R^*} + \vec{\omega}_{R^*/R}, \\ &= -\dot{\varphi}_i \vec{\boldsymbol{y}}_{\boldsymbol{R_c}} + \dot{\theta}\vec{\boldsymbol{z}}, \\ &= -\dot{\varphi}_i(-\sin(\theta)\,\vec{\boldsymbol{x}} + \cos(\theta)\vec{\boldsymbol{y}}) + \dot{\theta}\vec{\boldsymbol{z}}\,. \end{aligned} \tag{2.6}$$

Sachant que le roulement est sans glissement ($\vec{\boldsymbol{V}}_{\boldsymbol{P}} = \boldsymbol{0}$), on aura

$$\begin{aligned} \dot{x}_{c,i}\vec{\boldsymbol{x}} + \dot{y}_{c,i}\vec{\boldsymbol{y}} + \left(-\dot{\varphi}_i(-sin(\theta)\,\vec{\boldsymbol{x}} + cos(\theta)\vec{\boldsymbol{y}}) + \dot{\theta}\vec{\boldsymbol{z}}\right) \wedge (r\vec{\boldsymbol{z}}) =, \\ \left(\dot{x}_{c,i} - \boldsymbol{r}\dot{\varphi}_i cos(\theta)\right)\vec{\boldsymbol{x}} + \left(\dot{y}_{c,i} - r\dot{\varphi}_i\, sin(\theta)\right)\vec{\boldsymbol{y}} = 0. \end{aligned} \tag{2.7}$$

L'équation (2.7) nous donne le système de contraintes cinématiques suivant :

$$\begin{cases} \dot{x}_{c,i} - r\dot{\varphi}_i \cos(\theta) = 0, \\ \dot{y}_{c,i} - r\dot{\varphi}_i\, sin(\theta) = 0. \end{cases} \tag{2.8}$$

Le modèle (2.8) peut être transformé pour faire apparaître les composantes des vitesses

$$\begin{cases} \dot{x}_{c,i}\, sin(\theta) - \dot{y}_{c,i}\cos(\theta) = 0, \\ \dot{x}_{c,i} co\, s(\theta) + \dot{y}_{c,i}\, sin(\theta) = r\dot{\varphi}_i. \end{cases} \tag{2.9}$$

Ces deux équations sont des contraintes de type non holonome, signifiant que l'on ne peut pas les intégrer de façon à ne faire apparaître que les coordonnées généralisées.

Ainsi, l'application de la relation de Varignon sur le centre G de la plateforme mobile et le centre de chaque roue droite et gauche, qui sont tous sur le même axe nous donne le système d'équation suivant

$$\vec{V}_{P,d} = \vec{V}_G + \vec{\omega} \Lambda \overrightarrow{GP_d}\,, \tag{2.10}$$

$$\vec{V}_{P,g} = \vec{V}_G + \vec{\omega} \Lambda \overrightarrow{GP_g}\,, \tag{2.11}$$

$$\dot{x}\vec{\boldsymbol{x}} + \dot{y}\vec{\boldsymbol{y}} + \left(-\dot{\varphi}_d(-sin(\theta)\,\vec{\boldsymbol{x}} + cos(\theta)\vec{\boldsymbol{y}}) + \dot{\theta}\vec{\boldsymbol{z}}\right) \wedge \left(-L\left((-sin(\theta)\,\vec{\boldsymbol{x}} + co\, s(\theta)\,\vec{\boldsymbol{y}})\right) + r\vec{\boldsymbol{z}}\right) = 0 \tag{2.12}$$

$$\dot{x}\vec{\boldsymbol{x}} + \dot{y}\vec{\boldsymbol{y}} + \left(-\dot{\varphi}_g(-sin(\theta)\,\vec{\boldsymbol{x}} + cos(\theta)\vec{\boldsymbol{y}}) + \dot{\theta}\vec{\boldsymbol{z}}\right) \wedge \left(L\left((-sin(\theta)\,\vec{\boldsymbol{x}} + co\, s(\theta)\,\vec{\boldsymbol{y}})\right) + r\vec{\boldsymbol{z}}\right) = 0 \tag{2.13}$$

Ce qui donne le système de contraintes cinématiques suivant :

$$\begin{cases} \dot{x} - r\dot{\varphi}_d \cos(\theta) + L\dot{\theta}\cos(\theta) = 0, \\ \dot{y} - r\dot{\varphi}_d\, sin(\theta) + L\dot{\theta}\cos(\theta) = 0, \end{cases} \tag{2.14}$$

$$\begin{cases} \dot{x} - r\dot{\varphi}_g \cos(\theta) - L\dot{\theta}\cos(\theta) = 0, \\ \dot{y} - r\dot{\varphi}_g\, sin(\theta) - L\dot{\theta}\cos(\theta) = 0. \end{cases} \tag{2.15}$$

De simple transformation nous donne les contraintes cinématique pour la plateforme mobile

$$\begin{cases} \dot{x}\, sin(\theta) - \dot{y}\cos(\theta) = 0, \\ \dot{x} co\, s(\theta) + \dot{y}\, sin(\theta) = r\dot{\varphi}_d - L\dot{\theta}, \\ \dot{x}\cos(\theta) + \dot{y}\, sin(\theta) = r\dot{\varphi}_g + L\dot{\theta}. \end{cases} \tag{2.16}$$

D'autre part, le système d'équation suivante permet de calculer la vitesse linéaire et angulaire de la plateforme uni-cycle

$$\vec{V}_{C,d} = \vec{V}_G + \vec{\omega} \Lambda \overrightarrow{GC_d}, \tag{2.17}$$

$$\vec{V}_{C,g} = \vec{V}_G + \vec{\omega} \Lambda \overrightarrow{GC_g}. \tag{2.18}$$

En utilisant les deux équations (2.17) et (2.18), on aura

$$\vec{V}_{C,d} + \vec{V}_{C,g} = 2\vec{V}_G, \tag{2.19}$$

$$\vec{V}_{C,d} - \vec{V}_{C,g} = \vec{\omega}\Lambda\overrightarrow{GC_d} - \vec{\omega}\Lambda\overrightarrow{GC_g} = \vec{\omega}\Lambda\overrightarrow{C_gC_d} = 2L\dot{\theta}\, sin(\theta)\, \vec{\boldsymbol{y}} + 2L\dot{\theta}cos(\theta)\vec{\boldsymbol{x}}. \tag{2.20}$$

En remplaçant les équations (2.14) et (2.15) dans (2.19) et (2.20), la vitesse linéaire et la vitesse angulaire de la plateforme mobile devient

$$V_G = \sqrt{\dot{x}^2 + \dot{y}^2} = \frac{r\dot{\varphi}_d + r\dot{\varphi}_g}{2} = \frac{V_d + V_g}{2}, \tag{2.21}$$

$$\dot{\theta} = \frac{r\dot{\varphi}_d - r\dot{\varphi}_g}{2L} = \frac{V_d - V_g}{2L}. \tag{2.22}$$

Nous considérons le vecteur des coordonnées généralisées $\left[\dot{x}, \dot{y}, \dot{\theta}\right]^T$, à partir des équations (2.14) et (2.15), le modèle cinématique en fonction des vitesses angulaires des roues motrices gauche et droite s'écrit :

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} \frac{r}{2}\cos(\theta) & \frac{r}{2}\cos(\theta) \\ \frac{r}{2}\sin(\theta) & \frac{r}{2}\sin(\theta) \\ \frac{r}{2L} & -\frac{r}{2L} \end{pmatrix} \begin{pmatrix} \dot{\varphi}_d \\ \dot{\varphi}_g \end{pmatrix} = S(\theta)\dot{q}_p. \tag{2.23}$$

En utilisant ce modèle, la méthode de localisation par l'odométrie est certainement la méthode la plus couramment adoptée. Le principe consiste à déduire la position et l'orientation de la plateforme mobile, de façon incrémentale, à partir de la vitesse et les paramètres des roues motrices. Ces paramètres sont généralement difficiles à obtenir de façon précise et, dépendent ainsi de la nature du sol et la déformation des roues (mal gonflée ou déformations dues à la charge).

Cette technique exploite les capteurs de position pour chaque roue motrice pour calculer la distance parcourue par la roue. Cependant, dès que la distance parcourue augmente, des erreurs d'origines diverses s'accumulent. Elles résultent de l'imprécision introduite par les capteurs sur la mesure des déplacements angulaires des roues et d'une modélisation approximative de la localisation du robot mobile.

Par ailleurs, la commande cinématique pour la plateforme mobile adoptée dans cette thèse repose sur les travaux de Kanayama *et al* **[94]**. Supposons que la plateforme mobile doit

suivre une trajectoire de référence avec une position et une orientation définie par $[x_r(t), y_r(t), \theta_r(t)]^T$. On définit alors une erreur cinématique comme suit

$$\begin{pmatrix} e_x \\ e_y \\ e_\theta \end{pmatrix} = \begin{pmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_r - x \\ y_r - y \\ \theta_r - \theta \end{pmatrix}. \tag{2.24}$$

Ainsi, la vitesse virtuelle désirée qui garantit que l'erreur cinématique soit uniformément asymptotique stable est donnée par

$$\begin{pmatrix} \dot{x}_d \\ \dot{y}_d \\ \dot{\theta}_d \end{pmatrix} = \begin{pmatrix} \cos(\theta) & 0 \\ \sin(\theta) & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} k_x e_x + V_r \cos(e_\theta) \\ \dot{\theta}_r + k_y e_y V_r + k_\theta V_r cos(e_\theta) \end{pmatrix} \tag{2.25}$$

avec $V_r = \sqrt{\dot{x}_r^2 + \dot{y}_r^2}$ et k_x, k_y, k_θ sont des constante positives.

II.3. Modèle dynamique d'un manipulateur mobile à roues :

Considérant que les équations de la cinématique décrivent le mouvement sans tenir compte des forces qui produisent le mouvement, les équations dynamiques explicitement décrivent la relation entre les forces et les coordonnés cinématique du système ; vitesses et accélérations généralisées. Ce modèle peut prendre des formes différentes selon le formalisme de calcule utilisé. L'un des formalismes explicites les plus employés est la méthode de Lagrange qui permet de décrire la dynamique du système sous la forme d'équations simples non récursives et prendre en compte d'éventuelles contraintes non holonomes

La modélisation des bras manipulateurs peut être trouvée dans la plupart des ouvrages de robotique générale de base, citons par exemple **[2-4].**

La modélisation dynamique de plateformes mobiles évoquée dans la littérature est généralement présentée sous des formes simplifiées et le problème est souvent traité autour d'un exemple particulier de plateforme mobile. Une démarche peut être recommandée dans **[74]** permet de mieux concevoir le formalisme de Lagrange pour modéliser des systèmes non holonomes de type plateformes à roues

L'étude de la dynamique d'un manipulateur sur une plateforme uni-cycle dans **[75]** montre de point de vue dynamique, que pratiquement seulement les effets de couplage dynamique de la

plateforme sur le manipulateur influencent les performances du système quand il évolue de manière synchronisé.

Nous présentons dans cette section le modèle d'un manipulateur mobile à roue en utilisant le formalisme de Lagrange. Ce modèle est écrit sous une forme explicite directement exploitable pour la commande dynamique et il permet la prise en compte d'éventuels efforts d'interaction.

Dans le cadre de cette présentation, une modélisation détaillée n'est pas nécessaire puisque l'on se concentrera principalement de point de vue commande, sur les systèmes appartenant à cette classe de manipulateurs mobiles à roues dont la dynamique est supposée inconnue

Dans la mécanique de Lagrange, le résultat de calcule des équations dynamiques peut s'écrire sous la forme suivante :

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) + \tau_d(t) = J^T(q)\lambda + D(q)\tau(t), \qquad (2.26)$$

avec la matrice $M(q)$ peut s'écrire sous la forme :

$$M(q) = \begin{pmatrix} M_p(q) + M_{bp}(q) & M_{b/p}(q) \\ M_{p/b}(q) & M_b(q) \end{pmatrix}, \qquad (2.27)$$

où

- $M_p(q)$ la matrice d'inertie de la plateforme seule,
- $M_{bp}(q)$ une matrice permettant de tenir compte de la présence de la charge du manipulateur sur la plateforme,
- $M_{b/p}(q)$ permet de tenir compte des efforts d'accélération du bras manipulateur agissant sur la plateforme,
- $M_b(q)$ la matrice d'inertie du bras manipulateur seul où la base est considérée immobile,
- $M_{p/b}(q)$ une matrice dont les termes sont liés aux efforts d'accélération de la plateforme agissant sur le bras manipulateur,

$$C(q,\dot{q}) = \begin{pmatrix} C_p(q,\dot{q}) & C_{b/p}(q,\dot{q}) \\ C_{p/b}(q,\dot{q}) & C_b(q,\dot{q}) \end{pmatrix}, \qquad (2.28)$$

avec

- $C_p(q,\dot{q})$ la matrice de Coriolis et centrifuge de la plateforme seul,
- $C_b(q,\dot{q})$ la matrice Coriolis et centrifuge de manipulateur seule,
- $C_{p/b}(q,\dot{q})$ la matrice d'efforts de la plateforme agissant sur le bras manipulateur,
- $C_{b/p}(q,\dot{q})$ la matrice d'efforts du bras manipulateur agissant sur la plateforme,

$G(q)$ vecteur des forces de gravité associé au manipulateur,

$\tau_d(t)$ vecteur regroupe tout dynamique non modélisée inconnue ou négligée (ex ; frottements secs, dynamique des réducteurs…etc.) et éventuelles perturbations externes,

$J^T(q)\lambda$ vecteur des forces corresponds à la prise en compte des contraintes non holonome à avec $J^T(q)$ la matrice des contraintes cinématique et λ multiplicateurs de Lagrange,

$D(q)$ matrice de transformation,

$\tau(t)$ vecteur regroupant l'ensemble des couples engendrés par les actionneurs.

Théoriquement, il est toujours préférable d'établir un modèle de connaissance fondé sur des lois basique de la physique (lois de la mécanique, de l'électromagnétisme…etc.). Cependant, il arrive généralement que le système soit trop complexe, ou que les phénomènes qui le régissent soient mal connus. À cause d'incertitudes, le modèle mathématique ne possède pas un comportement semblable que possible à celui du système réel.

Néanmoins, si on arrive à caractériser ces incertitudes, il est possible de compléter le modèle nominal par une partie supplémentaire incertaine sous forme structurelle ou non. Les incertitudes structurelles sont généralement dues à la dynamique interne inconnue ou négligé, des erreurs d'approximation et de simplifications nécessaires afin d'obtenir un modèle exploitable mais reste caractérisé toujours par son domaine de validité. D'autre part, les incertitudes non structurelles représentent des dynamiques externes, par exemple : bruit de mesures, perturbations externes, etc. En général leurs dynamiques sont inconnues et on ne dispose d'aucune information sur leurs façons d'agir sur le système, sauf qu'elles sont bornées en normes comme l'énergie de tout système physique est limitée.

II.4. Points sur le mouvement et les contraintes non holonomes :

La configuration d'un robot est normalement décrite par un certain nombre de variables. Pour les robots mobiles, ce sont généralement la position et l'orientation du robot. Pour les robots manipulateurs, ces variables sont les positions des différentes articulations du bras du robot. Un mouvement pour un robot peut donc être considéré comme un chemin dans l'espace de configuration. Un tel chemin devrait rester dans le sous-espace des configurations dans lesquelles il n'y a pas de collision entre le robot et les obstacles, ce qu'on appelle l'espace libre. En effet, le problème de la planification des mouvements cherche à déterminer un tel chemin à travers l'espace libre de manière efficace.

La planification des mouvements peut être divisée en deux classes ; lorsque tous les degrés de liberté peuvent être modifiés de manière indépendante (comme dans un bras complètement actionné), nous parlons de planification de mouvements holonome ; une autre instance du problème de la planification des mouvements doit être envisagée en présence de contraintes cinématiques.

Ainsi, lorsque les degrés de liberté d'un système de robot ne sont pas indépendants ; comme par exemple une voiture qui ne peut pas tourner autour de son axe sans changer sa position, nous parlons de planification de mouvement non-holonome. Dans ce cas, tout chemin dans l'espace de configuration libre ne correspond pas nécessairement à un chemin réalisable, ainsi donc de point de vue cinématique, la planification du mouvement non-holonome s'avère être beaucoup plus difficile que la planification des mouvements holonome, ce qui représente un problème fondamentale pour la plupart des types de robots mobiles. C'est essentiellement la raison pour laquelle les techniques purement cinématique développées dans la planification de mouvement pour les systèmes holonomes ne s'appliquent pas directement à celles qui ne sont pas holonomes.

Un espace de travail d'un manipulateur mobile est considérablement étendu du fait des degrés de mobilité apportés par sa plateforme mobile. Toutefois, les degrés supplémentaires fournis par cette plateforme au manipulateur causent une redondance du système par rapport à une tâche de manipulation réalisable par le manipulateur.
Fréquemment, la planification de la tâche de tels systèmes est obtenue en exécutant deux phases séparées en traitant deux sous-systèmes complètement découplés ; une phase de déplacement de la plateforme mobile afin d'amener le bras manipulateur dans la zone objective où seuls les degrés de mobilité de la plateforme sont utilisés, et par la suite une phase de manipulation ou la réalisation d'une tâche où seuls les degrés de mobilité du bras manipulateur sont utilisés où la plateforme est immobile. Cependant, certaines applications nécessitent une synchronisation entre le mouvement de la plateforme mobile et le mouvement du bras manipulateur embarqué. Ainsi, une tâche de manipulation mobile peut être définie plus précisément comme l'exécution d'une trajectoire opérationnelle comprenant des phases de déplacement et de manipulation synchronisés ou non,

La tâche opérationnelle qui soit planifiée instantanée ou non, peut être caractérisée par le type de grandeur physique mise en jeu dans le mouvement.

De point de vue commande cinématique tout en considérant les contraintes non-holonomes auxquelles la base mobile est soumise, on peut classifier la génération de mouvement en trois catégories.

a- **Mouvement point à point:** le robot doit atteindre une configuration objective désirée à partir d'une configuration initiale donnée. En utilisant une terminologie de point de vue contrôle, le déplacement point à point est un problème de stabilisation autour d'un point d'équilibre dans l'espace d'état du robot.

b- **Suivi de chemin:** le robot doit atteindre et suivre un chemin géométrique dans l'espace cartésien à partir d'une configuration initiale donnée où la vitesse du robot ne dépend pas de ce chemin. Dans ce contexte, le contrôleur de mouvement reçoit une description géométrique du chemin cartésien où l'évolution du paramètre du temps est généralement libre et par conséquence, la vitesse peut être choisie arbitrairement. Un exemple typique est la conduite sur route lorsqu'on cherche à maintenir une distance constante par rapport à une ligne marquée sur le sol.

c- **Suivi de trajectoire :** ce problème se distingue du précédent par le fait que le chemin géométrique à suivre est définie par une loi temporelle où le profil de la vitesse est un paramètre à respecter. Dans certains problèmes, l'espace de travail peut être dynamique et l'emplacement des obstacles peut changer dans le temps, ce qui rend la dimension temporelle un paramètre important dans le mouvement du robot. Le problème peut pareillement être interprété comme la poursuite d'une voiture à une autre de référence

Au niveau de la loi de commande, les systèmes non-holonomes sont caractérisés par des équations de contraintes disposant les dérivées temporelles des variables de configuration du système. Ces équations ne sont pas intégrables, et apparaissent généralement lorsque le nombre des variables directement contrôlable soit inférieure au nombre des variables de configuration du système ; par exemple, un robot de type voiture est commandé avec deux variables (vitesses linéaires et angulaires) alors qu'il se déplace dans un espace de configuration tridimensionnel.

Par ailleurs, les contraintes introduisent deux types de difficultés dans la résolution des équations dynamique et au niveau de l'exécution du mouvement. Les coordonnées ne sont plus indépendantes puisqu'elles sont liées par des équations de contraintes, de sorte que les équations du mouvement ne sont pas indépendantes ainsi donc, l'existence d'une convergence indépendante de l'erreur stabilisée n'est plus garantie pour les systèmes non-holonomes. D'autre part, les forces de contraintes font partie des inconnues et doivent être obtenues à partir de la résolution des équations ou éliminées de l'équation dynamique.

Théoriquement on rencontre trois scénarios possibles ; 1) les contraintes sont holonomes, et il est facile de les utiliser pour éliminer les coordonnées généralisées surplus ; 2) les contraintes sont holonomes; cependant, l'élimination des coordonnées généralisées surplus conduit à des expressions très compliquées ; 3) les contraintes sont non-holonomes, et nous ne pouvons donc pas éliminer les coordonnées géométriques surplus, car les contraintes ne sont pas intégrables.

II.5. Conclusion

Dans ce chapitre, dans un premier temps, nous avons développé le modèle cinématique qui permet la fonctionnalité de changeur de coordonnées, afin de pouvoir commander le robot via les coordonnées articulaires associées à une tâche opérationnelle. Ensuite, nous avons présenté le modèle dynamique générale d'un robot manipulateur mobile représenté par le formalisme de Lagrange.

Les différentes modèles et les principes de modélisation des robots dans ce chapitre sont utilisés d'une façon ou une autre dans les contrôleurs des robots modernes, et par ailleurs, engendrés par les systèmes de conception et de fabrication assistés par ordinateur (CFAO) pour simuler le comportement des robots.

Finalement, nous avons posé des points sur le planificateur de tâche et les difficultés rencontrées dans la résolution des équations dynamique et au niveau de l'exécution du mouvement pour des robots avec des contraintes cinématiques non-holonomes.

Chapitre III.

Loi de commande neuro optimale robuste adaptative des robots manipulateurs mobiles.

III.1. Introduction :

La commande basique d'un système dynamique signifie le forcer à se comporter d'une manière désirée dans un cadre structuré. En d'autres termes, une loi est dite de commande si elle permet d'amener l'état d'un système à un certain état final en respectant certains critères. L'objectif ultime de l'ingénierie de contrôle est de mettre en place un contrôleur qui pourrait fonctionner avec une indépendance croissante vis-à-vis des interventions humaines dans un environnement non structuré et incertain. Un tel système de contrôle peut être qualifié autonome ou intelligent. Par ailleurs, la recherche d'un état de fonctionnement dit optimal est parmi les défis majeurs dans divers domaines.

Pour aborder précisément la notion de système de contrôle optimal, nous devons utiliser le langage mathématique. Si on définit un espace dit des systèmes contrôlables, on associe pour chaque système un ensemble des lois de commande appartient à un espace parallèle dit l'espace des lois de commande admissibles. Ainsi, le problème de la commande optimale est de découvrir analytiquement une loi de commande dans l'espace parallèle des commandes pour un système donné dans l'espace des systèmes contrôlables, qui est modélisé par un ensemble d'équations mathématique de sorte qu'un certain critère d'optimalité soit abouti. Autrement dit, parmi toutes les lois de commande associées à un système donné, on détermine mathématiquement celle qui minimise un certain critère de notre choix.

Historiquement la théorie de la commande optimale est liée aux principes variationnels de la mécanique classique. Couramment, le célèbre problème d'optimisation de la courbe brachistochrone de Johann Bernoulli est considéré comme l'origine du calcul des variations. Bernoulli et, plus tard, Euler et Lagrange, ont contribué de manière consistante au développement du calcul des variations. Au dix-neuvième siècle, Hamilton, Weierstrass et Jacobi ont développé davantage cette théorie **[76]**.

Le célèbre principe du maximum de Pontryagin, formulé par L. S. Pontryagin en 1956, qui donne une condition nécessaire d'optimalité représente le point clé de la théorie de la commande optimale. Par la suite cette théorie a été renforcée par la méthode de programmation dynamique de Bellman qui fournit une condition suffisante d'optimalité et la découverte des liens entre la solution d'une commande optimale et la solution fondée sur la théorie de stabilité de Lyapunov **[77]**.

Nous présentons dans la suite la formulation de problème classique de commande optimale, dont on déduit les conditions d'optimalité. L'indice de performance principal a adopté tout au long de notre travail est l'indice quadratique au cas des systèmes sans contraintes. Les démarches présentées dans ce chapitre ne se limitent pas nécessairement à la robotique.

III.2. Formulation classique de la commande optimale :

III.2.1. Cas général d'un système non linéaire:

Nous considérons la famille de problèmes d'optimisation suivante :

$$min_{u\in U}(J = \int_{t_0}^{t_f} L(x,u,t)\, dt), \tag{3.1}$$

où U représente l'espace des commandes admissibles pour une famille des systèmes dont l'évolution dynamique représenté par une équation d'état le plus souvent explicitée sous la forme

$$\dot{x} = f(x,u,t). \tag{3.2}$$

Le problème de la commande optimale dans ce cas consiste alors de calculer explicitement la commande u minimisant J. Le principe d'optimalité de Bellman s'énonce de la façon suivante :

Une commande optimale possède la propriété en principe que, quelques soient l'état initial et l'instant initial t, la commande restant à prendre doit aussi constituer une commande optimale.

Mathématiquement, il découle de ce principe d'optimalité l'équation suivante :

$$V(x,t) = min_{u[t,t_f]}\left\{J = \int_t^{t_f} L(\tau,x,u)d\tau\right\}. \tag{3.3}$$

Ainsi, l'équation s'écrit avec un changement de variable comme suit

$$\begin{aligned} V(x,t) &= min_{u[t,t_f]}\left\{\int_t^{t+\Delta t} L(x,u,\tau)d\tau + \int_{t+\Delta t}^{t_f} L(x,u,\tau)d\tau\right\}, \\ &= min_{u[t,t_f]}\left\{\int_t^{t+\Delta t} L(x,u,\tau)\, d\tau + V(x(t+\Delta t), t+\Delta t)\right\} \end{aligned} \tag{3.4}$$

Par ailleurs, une approximation de Taylor au premier ordre en Δt de l'intégrale nous donne :

$$\int_t^{t+\Delta t} L(x,u,\tau)\, d\tau = L(x,u,t)\, \Delta t + O(\Delta t). \tag{3.5}$$

De la même manière le développement de Taylor au premier ordre de $V(x(t+\Delta t), t+\Delta t)$ s'écrit :

$$V(x(t+\Delta t), t+\Delta t) = V(x(t),t) + \frac{\partial V}{\partial t}\Delta t + \left(\frac{\partial V}{\partial x}\right)^T \dot{x}\Delta t + O(\Delta t). \tag{3.6}$$

Finalement en remplaçant ces deux approximations dans (3.4), on obtient :

$$V(x,t) = min_u \left\{ L(x,u,t)\,\Delta t + V(x,t) + \frac{\partial V}{\partial t}\Delta t + \left(\frac{\partial V}{\partial x}\right)^T \dot{x}\Delta t + O(\Delta t) \right\}. \quad (3.7)$$

En simplifiant par $V(x,t)$ et on calcule la limite de l'équation lorsque Δt tend vers 0, on aura l'équation de Hamilton-Jacobi-Bellman :

$$-\frac{\partial V}{\partial t} = min_u \left\{ L(x,u,t) + \left(\frac{\partial V}{\partial x}\right)^T \dot{x} \right\}. \quad (3.8)$$

L'équation peut encore s'écrit sous la forme compacte :

$$min_u \{ L(x,u,t) + \dot{V}(t,x) \} = 0. \quad (3.9)$$

Il est facile de démontrer que la résolution de l'équation (3.9) nous donne une seule solution u^* dite optimale. À cet effet, nous considérons deux commandes ; u^*(optimale) et u (quelconque), dans l'espace des commandes associées au système, générant respectivement les trajectoires x^* et x. À partir de l'équation (3.9), on obtient :

$$\begin{cases} L(x,u,t) + \dot{V}(x,t) \geq 0, \\ L(x^*,u^*,t) + \dot{V}(x^*,t) = 0. \end{cases} \quad (3.10)$$

Avec un simple intégrale de ces deux expressions de t_0 à t_f, on peut recalculer le critère à minimiser J :

$$\begin{cases} \int_{t_0}^{t_f} L(x,u,t)\,dt + V(x_f,t_f) - V(x_0,t_0) \geq 0, \\ \int_{t_0}^{t_f} L(x^*,u^*,t)\,dt + V(x_f^*,t_f) - V(x_0,t_0) = 0. \end{cases} \quad (3.11)$$

En éliminant $V(x_0,t_0)$, on conclut finalement que la commande u^* est la seule qui donne une solution minimale :

$$\int_{t_0}^{t_f} L(x^*,u^*,t)\,dt + V(x_f^*,t_f) \leq \int_{t_0}^{t_f} L(x,u,t)\,dt + V(x_f,t_f). \quad (3.12)$$

III.2.2. Cas d'un système affine quadratique :

Nous considérons le cas particulier de la famille des systèmes affines-quadratiques en u

$$\dot{x} = f(x,t) + g(x,t)u\ . \tag{3.13}$$

Dans la régulation autour d'un point d'équilibre, le problème peut être formalisé comme la minimisation d'un critère quadratique représentant une pondération de l'énergie et de l'état :

$$min_{u\in U}\left\{J = \int_{t_0}^{t_f} x^T Q x + u^T R u\ \ dt\right\}, \tag{3.14}$$

où $Q > 0$ $R > 0$.

Il est possible de calculer explicitement la commande optimale en utilisant la condition nécessaire de stationnarité par rapport à u:

$$\frac{\partial}{\partial u}\left(l(t,x,u^*) + \dot{V}\right) = 0. \tag{3.15}$$

La résolution de l'équation (3.15) conduit à la forme explicite suivante :

$$u^* = -\frac{1}{2}R(t,x)^{-1}g(t,x)^T\ (\frac{\partial V}{\partial x})\ . \tag{3.16}$$

Ainsi, l'équation de Hamilton-Jacobi-Bellman vérifie la relation :

$$-\frac{\partial V}{\partial t} = min_u\left\{x^T Q x + u^T R u\ \ + \left(\frac{\partial V}{\partial x}\right)^T \dot{x}\right\}. \tag{3.17}$$

Pour résoudre cette équation, il est convenable d'admettre que la valeur optimale du coût (3.3) est une fonction quadratique de l'état :

$$V(t,x) = x^T K(t) x, \tag{3.18}$$

où $K = K^T \geq 0$ et $K(t_f) = 0$.

On peut alors calculer la commande optimale en utilisant (3.16)

$$u^* = -\frac{1}{2}R(t,x)^{-1}g(t,x)^T\ K(t)\ x\ . \tag{3.19}$$

En remplaçant (3.18)-(19) dans (3.17), on aura:

$$-x^T\dot{K}(t)x = x^T[Q - Kg(t,x)R^{-1}g(t,x)^T K]x + 2Kf(t,x). \tag{3.20}$$

Dans le cas particulier où $f(x,t)$ s'écrit par rapport à l'état x ; $f(t,x) = Ax$ on aura finalement l'équation différentielle de Riccati:

$$-\dot{K}(t) = KA + A^T K - Kg(t,x)R^{-1}g(t,x)^T K + Q. \quad (3.21)$$

Donc, la solution (3.19) s'obtient à partir de la résolution de l'équation différentielle (3.21) avec conditions terminales, ce qui nécessite le calcul de K(t) sur tout l'intervalle $[t_0, t_f]$ avant d'appliquer la commande.

En règle générale, si on est capable de résoudre l'équation HJB dans laquelle le problème se réduit à résoudre des équations aux dérivées partielles du premier ordre, alors on est capable de calculer analytiquement la commande optimale associée. D'autre part, la démarche dans le développement de la loi de commande optimale pour un système non linéaire est d'assumer que les dynamiques non linéaires $f(t,x)$ et $g(t,x)$ sont exactement connues. Généralement, la difficulté essentielle après l'exigence d'un modèle dynamique précise pour le calcul d'une commande optimale réside dans la solution indirecte de l'équation HJB qui représente un problème non trivial notamment dans les applications temps réel.

III.3. Réseau de neurones pour l'approximation :

III.3.1. Introduction :

Nous allons faire ici une brève description des réseaux de neurones. De même nous ne cherchons pas à dresser une liste des différents modèle de réseaux neuronales rencontrés dans la littérature, mais nous nous intéressons plutôt à celle que nous avons mise en œuvre dans cette thèse pour la conception d'une loi de commande pour les robots manipulateurs mobiles.

Quelle que soit la nature d'un système dynamique, la connaissance de son comportement est la première question clé afin de maîtriser ou d'optimiser leur fonctionnement. La mise en place d'une représentation qui possède un comportement semblable que possible à celui du système communément appelée modèle, généralement sous forme d'une relation analytique définie par une structure et des paramètres.

Pour un système mécatronique comme les robots, il est possible d'établir ce modèle à partir de connaissances de la physique (lois de la mécanique, électrique, hydraulique…etc.).

Toutefois, dès que le système soit complexe et hybride, ou les phénomènes qui le régissent sont inconnus, il arrive fréquemment que la détermination d'un tel modèle suffisamment précis devient une tâche difficile, voire parfois impossible, ou encore ce modèle soit trop compliqué pour être exploité.

Par ailleurs, on trouve des techniques d'identification des systèmes reposent souvent sur une hypothèse de l'existence d'une structure déterministe (modèle d'hypothèse), et l'évaluation du système par une analyse expérimentale. Le domaine de validité d'un tel modèle reste souvent lié au domaine de l'expérience utilisée et ce n'est pas facile de le généraliser. Quoi qu'il en soit, cette démarche seule représente un problème et un défi difficile.

Grâce aux résultats concrets obtenus au cours des dernières années, des familles de fonctions, constituent des approximateurs universels très avantageuses pour pallier la phase de la modélisation et l'identification des systèmes dynamiques dans la conception des lois de commande. Inspiré de système nerveux ; les réseaux de neurones, de raisonnement humain ; les systèmes flous, issues du traitement de signal ; les ondelettes, entrent dans cette famille. La caractéristique d'approximation permet en fait sous certaines conditions la représentation fonctionnelle paramétrée de tout système avec une précision arbitraire, si sa structure est convenablement choisie.

Les réseaux de neurones artificiels occupent aujourd'hui une place dominante dans plusieurs domaines de l'ingénierie ; commande, identification, classification, reconnaissance de forme, optimisation…etc. En raison de leur capacité d'approximation, de parcimonie (petit nombre de fonctions pondérées) et d'apprentissage en apprenant directement à partir des données. Ces caractéristiques ont motivé une recherche approfondie et des développements dans les réseaux de neurones artificiels ces dernières années, et une récession récente dans le domaine des réseaux neuronaux en raison de l'introduction de nouvelles topologies de réseau et d'algorithmes d'apprentissage.

L'évolution de cette technique remonte d'une part aux travaux de Werbos qui a conçu un mécanisme d'apprentissage économique en temps de calcul, appelé algorithme de rétro-propagation (Back-Propagation) **[78]**, et la démonstration de ses propriétés d'approximation universel par la suite **[79]**. Les travaux de **[80]** considérés comme l'une des premières applications dans le domaine de la modélisation non linéaire des systèmes

III.3.2. Définitions :

a. Le neurone formel :

Un neurone formel est une fonction mathématique pondérée représente la brique de base dans un réseau de neurones, caractérisée par une fonction d'activation $\boldsymbol{\sigma}$ décrivant l'état de sa sortie **s** en fonction de ses entrées $\boldsymbol{x_i}$. La sortie est transmise aux neurones auxquels ce neurone est connecté à son tour dans le réseau (Figure 3.1).

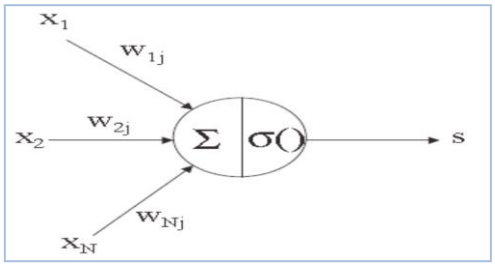


Figure.3.1. Un neurone formel.

avec w_i *: poids d'interconnexion.*

b. Le réseau de neurones:

Un ensemble de neurones connectés entre eux et organisés en réseau. Le comportement d'un réseau de neurones est déterminé par trois éléments fondamentaux ; les neurones ou fonctions d'activation, le modèle de connexion des neurones et l'algorithme d'apprentissage ou d'adaptation. La façon dont les neurones d'un réseau neuronal sont interconnectés détermine sa structure.

Les structures les plus utilisées pour l'identification et la commande sont ; les réseaux feedforward à une seule couche, les réseaux Multicouches MLP (Multi-Layer Perceptrons), les réseaux de base radiale RBF, et les réseaux neuronaux dynamiques ou récurrents.

Dans le cadre de cette thèse, nous présentons le modèle de réseau que nous utiliserons plus tard dans la conception des lois de commande. Le réseau est de type MLP possédant une seule couche cachée avec une fonction d'activation de sortie linéaire (Figure3.2). La fonction réalisée par un tel réseau est définie par

$$y_i = \sum_{j=1}^{N_2} w_{ij}\left[\sigma\left(\sum_{k=1}^{N_1} v_{jk}x_k\right)\right] \quad ; i = 1,..,N_3, \tag{3.22}$$

où w_{ij} sont les poids d'interconnexion entre les neurones dans la couche cachée et les sortie, σ la fonction d'activation réalisée par chaque neurone, v_{jk} sont les poids d'interconnexion entre les entrées et la couche cachée, $x = [x_1\ x_2\ ...\ x_{N_1}]^T \in \mathbb{R}^{N_1}$ le vecteur d'entrée, N_2 Le nombre de neurones dans la couche cachée.

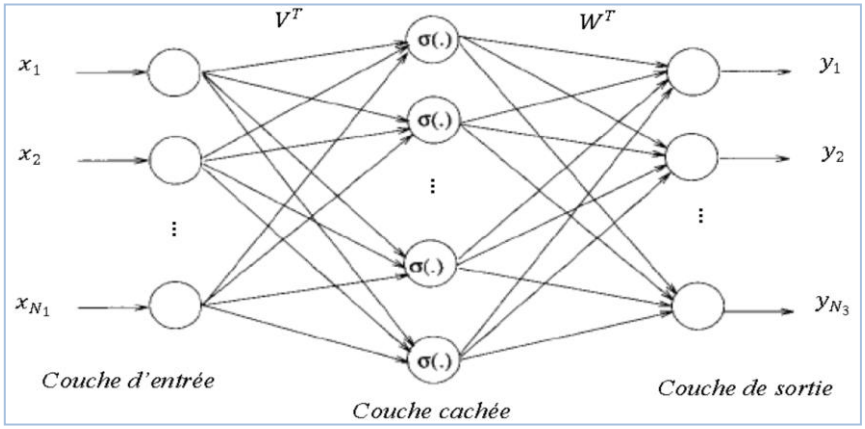


Figure.3.2. La structure MLP à trois couches.

Ou sous forme matricielle

$$Y = W^T\sigma(V^T x), \tag{4.23}$$

avec $\begin{cases} V^T \in \mathbb{R}^{N_2 \times N_1}\ ,\ x \in \mathbb{R}^{N_1}, \\ W^T \in \mathbb{R}^{N_3 \times N_2}\ ,\ \sigma \in \mathbb{R}^{N_2}, \\ Y \in \mathbb{R}^{N_3}\ . \end{cases}$

c. La propriété d'approximation :

La propriété d'approximation dépend de la fonction d'activation choisie. L'intérêt de la recherche sur les capacités du perceptron multicouches pour approximer une fonction continue arbitraire a d'abord été mentionnée dans **[81]** et par la suite dans **[82].**

C'est Cybenko qui a d'abord démontré qu'une seule couche cachée est suffisante pour approximer uniformément toute fonction continue avec une fonction d'activation de type Sigmoïdale **[83-84].** La propriété d'approximation a été prouvée également pour les fonctions de type tangent hyperbolique dans **[82].** De la même façon le réseau RBF de fonctions gaussiennes possèdent des propriétés équivalentes qui ont été présentées dans **[85].** Toutefois, généralement les paramètres auxiliaires dans la fonction gaussienne tels que la position des

centres des RBF compliquent leur intérêt dans la conception d'une loi de commande et l'utilisation de réseau MLP peut être plus adéquat pour notre application.

À partie de la théorie d'approximation des fonctions, un réseau de neurones pouvait implémenter n'importe quelle fonction continue.

Théorème : **[84,86]**

soit $S \subset \mathbb{R}^{N_1}$, $\forall x \in S, \forall f(x) \in C^{N_3}(S), \exists \varepsilon_n > 0$, $\exists\, \sigma, \exists\, N_2 \in \mathbb{N}$, $\exists\, W, V$ tel que

$$f(x) = W^T \sigma(V^T x) + \varepsilon(x) \quad \text{tel que} \quad \sup\|\varepsilon(x)\| < \varepsilon_n \,, \tag{3.24}$$

où $\varepsilon(x)$ représente l'erreur d'approximation.

Ainsi, ce théorème d'approximation peut être énoncé de la manière suivante ; toute fonction bornée régulière et dérivable peut être approximée avec une précision arbitraire, dans un domaine fini de l'espace de ses variables, par un réseau de neurones avec une couche de neurones cachés en nombre suffisant, possédant tous la même fonction d'activation, et un neurone de sortie linéaire.

L'erreur d'approximation du réseau, notée $\varepsilon(x)$, est une quantité critique qui représente l'écart minimum possible entre la fonction non linéaire inconnue f(x) et la fonction de sortie de l'approximateur.

Le nombre de neurones dans une couche cachée est le paramètre de conception le plus important par rapport aux capacités d'approximation d'un réseau neuronal. Rappelons que le nombre de variables d'entrée et le nombre de neurones de sortie sont en général déterminés par la nature du système à approximer. En général, augmenter le nombre de neurones dans une couche cachée désigné par N_2, réduit l'erreur d'approximation et peut être rendu arbitrairement petit sur une région compacte mais différent de zéro dans le cas pratique. Toutefois, on cherche toujours le nombre minimal de neurones pour obtenir une performance désirée.

Il existe plusieurs démarches pour déterminer le nombre de neurones dans une couche cachée; la méthode ascendante qui consiste à augmenter itérativement le nombre de neurones dans le réseau (Cascade-Correlation) ; la méthode descendante qui consiste à éliminer des neurones dans le réseau pendant ou après l'apprentissage (Weight Elimination).

L’estimation des poids du réseau de neurones nécessite un algorithme d’adaptation ou d’apprentissage. Cet algorithme est un processus auto organisation, dans le sens où le réseau peut former sa propre représentation en fonction de l'information reçue permettant de modifier de manière plus ou moins automatique ses paramètres. Une caractéristique d'apprentissage ou d’adaptation permet aux réseaux de neurones d'identifier un processus inconnu avec peu de connaissances a priori sur le processus.

L'intelligence du réseau neuronal réside essentiellement dans la capacité de généralisation qui mesure la robustesse du réseau s'il est capable de se comporter comme le système approximé pour des entrées non utilisées dans l’apprentissage. Une règle d'apprentissage correctement conçue devrait garantir la convergence relative globale du réseau pour une application donnée.

Selon l’algorithme utilisé pour l’estimation des poids du réseau, le réseau soit adaptatif ou non. Par exemple, dans l'apprentissage supervisé dans la reproduction d’une commande existante **[87],** les paramètres du réseau sont estimés pour rendre la sortie du réseau aussi proche de celle du superviseur. Ce type de technique emploie un ensemble des données limité durant l’apprentissage, et par conséquent, le réseau ne sera pas adaptatif tant que la phase d’apprentissage et la phase d’utilisation de réseau sont distinctes. Or, il est difficile d’ajouter une expérience supplémentaire sans initialiser l’apprentissage, ce qui implique que le réseau est non modifiable pendant son utilisation.

Pour un réseau adaptatif, les paramètres d'un réseau de neurones doivent être estimés en temps réel et en permanence pendant l'utilisation du système, ce qui convient par exemple pour des systèmes ayant des caractéristiques variant dans le temps. Si le système neuronal à base d’apprentissage hors ligne mémorise ces paramètres pour un point de fonctionnement donné, alors un système neuronal adaptatif doit les découvrir instantanément chaque fois que les points de fonctionnement se changent. Cependant, la stabilité du réseau doit être garantie et une vitesse d'adaptation relativement élevée est essentielle pour les applications en temps réel.

Une autre caractéristique liée à l’architecture parallèle des réseaux de neurones souvent citée dans la littérature comme un point fort, et qui lui confère une grande rapidité de calcul. Ceci est vrai seulement lorsque l’implémentation du réseau est réellement sur des multiprocesseurs parallèles comme les FPGA.

III.4. Description des systèmes de robots manipulateurs mobiles non-holonomes :

L'évolution dynamique du robot manipulateur mobile est souvent formulée avec les équations générales obtenues à partir de la mécanique Lagrangienne. Généralement la dynamique des robots manipulateurs mobiles avec des contraintes non holonomes peut être exprimée comme

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) + \tau_d(t) = J^T(q)\lambda + D(q)\tau(t), \quad (3.25)$$

$$J(q)\dot{q} = 0, \quad (3.26)$$

où les coordonnées généralisées $q(t) \in \mathbb{R}^n$ avec les vitesses associées $\dot{q}(t)$ et les accélérations $\ddot{q}(t)$ sont contrôlées par les couples appliqués $\tau(t) \in \mathbb{R}^p$. Le moment d'inertie généralisé $M(q) \in \mathbb{R}^{n\times n}$, les forces Centrifuge et Coriolis $C(q,\dot{q})\dot{q} \in \mathbb{R}^n$, les forces gravitationnelles $G(q) \in \mathbb{R}^n$, la matrice de transformation d'entrée $D(q) \in \mathbb{R}^{n\times p}$, et la matrice de contraintes cinématiques $J(q) \in \mathbb{R}^{m\times n}$ toutes varient selon la configuration du robot. La variable $\lambda \in \mathbb{R}^m$ représente le vecteur multiplicateur de Lagrange, et $\tau_d(t) \in \mathbb{R}^n$ regroupe les dynamiques inconnues et les perturbations externes.

Nous supposons que les positions $q(t)$ et les vitesses $\dot{q}(t)$ sont mesurables. De plus, nous supposons que les couples $\tau(t)$ sont disponibles comme entrées de commande et que les matrices $M(q)$, $C(q,\dot{q})$ et $J(q)$ sont structurées satisfont les propriétés suivantes:

P1: La matrice $M(q)$ est symétrique et définie positive ; $M(q) = M^T(q) > 0$.

P2. La matrice $\dot{M}(q) - 2C(q,\dot{q})$ est skew-symétrique ; $x^T\big(\dot{M}(q) - 2C(q,\dot{q})\big)x = 0, \forall x \in \mathbb{R}^n$.

P3 La matrice $J(q)$ est de rang plein.

Dans la conception classique de contrôle optimal, la démarche consiste à calculer analytiquement la loi de commande pour un système de dynamique supposé parfaitement connu, de telle sorte qu'un certain critère d'optimalité soit réduit au minimum. Ce dernier comprend un coût fonctionnel, qui est fonction des variables d'état et de contrôle et, en général, il est nécessaire de résoudre les équations HJB hors ligne. Par contre, la contrainte imposée dans notre étude réclame que la loi de commande doit être calculée en ligne et il faut toujours respecter l'hypothèse imposé que nous ne disposons pas de modèle du système.

En effet, la commande adaptative robuste et le réseau de neurones adaptatif sont utilisés pour améliorer les performances face à une dynamique non modélisée et à une perturbation externe, en ajoutant une loi de commande non linéaire à la loi optimale linéaire avec une solution explicite de HJB calculée en ligne.

Les inconnues du problème sont le vecteur q et le vecteur multiplicateurs de Lagrange λ. Les équations (3.25) et (3.26), déterminent complètement la dynamique du système. Nous avons donc un système à n équations et $n+m$ inconnues. Si le but n'est pas de calculer les efforts de réactions dues aux liaisons cinématiques considérées comme dépendantes, il est possible de simplifier cette relation en éliminant les multiplicateurs de Lagrange.

Posons $q = \left[q_p^T, q_a^T\right]^T$, selon la propriété P3, l'équation (3.26) peut être divisée en deux parties linéairement indépendantes comme suit

$$J_1(q)\dot{q}_p + J_2(q)\dot{q}_a = 0, \tag{3.27}$$

où $q_p \in \mathbb{R}^m$ est considéré comme une variable passive avec $J_1(q) \in \mathbb{R}^{m\times m}$ est une matrice non singulière, et $q_a \in \mathbb{R}^p$ est considéré comme une variable active avec $J_2(q) \in \mathbb{R}^{m\times p}$.

Ainsi, l'équation (3.27) peut être écrite comme suit

$$\dot{q}_p = -J_1^{-1}(q)J_2(q)\dot{q}_a = J_{12}(q)\dot{q}_a. \tag{3.28}$$

Posons une matrice $S(q) = \left[J_{12}^T(q), I_p\right]^T \in \mathbb{R}^{n\times p}$ avec I_p matrice d'identité

En remplaçant $\dot{q} = S(q)\dot{q}_a$ et $\ddot{q} = S(q)\ddot{q}_a + \dot{S}(q)\dot{q}_a$ dans l'équation (3.25) et (3.26), on obtient

$$M(q)S(q)\ddot{q}_a + \left(C(q,\dot{q})S(q) + M(q)\dot{S}(q)\right)\dot{q}_a + G(q) + \tau_d(t) = J^T(q)\lambda + D(q)\tau(t), \tag{3.29}$$

et

$$J(q)S(q) = S^T(q)J^T(q) = 0. \tag{3.30}$$

Ainsi, l'élimination des multiplicateurs de Lagrange consiste à multiplier à gauche chaque terme de (3.29) par $S^T(q)$. En utilisant le résultat précédemment établi $S^T(q)J^T(q) = 0$, nous pouvons obtenir la forme dynamique modifiée :

$$\bar{M}(q)\ddot{q}_a + \bar{C}(q,\dot{q})\dot{q}_a + \bar{G}(q) + \bar{\tau}_d(t) = \bar{D}(q)\tau(t), \tag{3.31}$$

avec $\bar{M}(q) = S^T(q)M(q)S(q)$, $\bar{C}(q,\dot{q}) = S^T(q)\left(C(q,\dot{q})S(q) + M(q)\dot{S}(q)\right)$, $\bar{G}(q) = S^T(q)G(q)$, $\bar{\tau}_d(t) = S^T(q)\tau_d(t)$ et $\bar{D}(q) = S^T(q)D(q)$.

Cette relation représente le modèle dynamique réduit d'un système mécanique. Il comporte m équations et autant d'inconnues. Ces inconnues sont les coordonnées généralisées $q_a \in \mathbb{R}^m$du système pouvant évoluer indépendamment. L'évolution dynamique des multiplicateurs de Lagrange non conservées dans le système peut être reconstruite à partir de l'évolution de ces coordonnées généralisées q_a.

Le modèle dynamique réduit à une équation tout à fait exploitable pour la commande en couple d'un manipulateur mobile. Il permet de calculer directement les couples moteurs nécessaires à une trajectoire donnée du vecteur d'état q_a. Nous verrons plus tard qu'il est possible de résoudre explicitement le problème de la commande optimale d'un système dynamique écrit sous la forme de (3.31), où la solution HJB n'est pas fonction du modèle dynamique.

Etant donné une trajectoire désirée $q_{a,d}(t) \in \mathbb{R}^m$, l'erreur de suivi de la trajectoire est définie par $e(t) = q_{a,d}(t) - q_a(t)$ et l'erreur filtrée associée est définie par $r(t) = \dot{e}(t) + \Lambda e(t)$ tandis que $\Lambda = \Lambda^{\mathrm{T}} \in \mathbb{R}^{p\times p}$ est une matrice définie positive. Ainsi, les équations dynamiques par rapport aux erreurs $e\,(t)$ et $r\,(t)$ peuvent être obtenues comme

$$\dot{e}(t) = r(t) - \Lambda e(t), \tag{3.32}$$

$$\bar{M}(q)\dot{r}(t) = -\bar{C}(q,\dot{q})r(t) + H(r_e) - \bar{D}(q)\tau(t), \tag{3.33}$$

avec $r_e = \left[q_a^T(t), \dot{q}_a^T(t), q_{a,d}^T(t), \dot{q}_{a,d}^T(t), \ddot{q}_{a,d}^T(t)\right]^T$, et la fonction non linéaire $H(r_e)$ regroupe toute la dynamique inconnue du système telle que

$$H(r_e) = \bar{M}(q)\left(\ddot{q}_{a,d} + \Lambda\dot{e}(t)\right) + \bar{C}(q,\dot{q})\left(\dot{q}_{a,d} + \Lambda e(t)\right) + \bar{G}(q) + \bar{\tau}_d(t). \tag{3.34}$$

On définit l'erreur augmenté par $\tilde{z}^T(t) = [e^T(t), r^T(t)]^T$. Maintenant, par convention, on réécrit (3.33) dans la forme de l'espace d'état, nous obtenons la dynamique de représentation de l'état de l'erreur augmentée complète suivante

$$\dot{\tilde{z}}^T(t) = \begin{pmatrix}\dot{e}(t)\\ \dot{r}(t)\end{pmatrix} = \begin{pmatrix}-\Lambda & I_{p\times p}\\ 0_{p\times p} & -\bar{M}^{-1}(q)\bar{C}(q,\dot{q})\end{pmatrix}\begin{pmatrix}e(t)\\ r(t)\end{pmatrix} + \begin{pmatrix}0_{p\times p}\\ \bar{M}^{-1}(q)\end{pmatrix}\left(H(r_e) - \bar{D}(q)\tau(t)\right). \tag{3.35}$$

Ou encore avec une notation plus courte

$$\dot{\tilde{z}}^T(t) = A(q,\dot{q})\tilde{z}(t) + B(q)u(t), \tag{3.36}$$

où les forces généralisées $\tau(t)$ sont utilisées pour l'attribution de la loi de commande et la variable $u(t)$ représente la commande auxiliaire.

III.5. Conception de la commande optimale :

Pour obtenir la loi de commande optimale pour le robot manipulateur mobile, le problème de contrôle est formulé comme un problème d'optimisation quadratique. Ensuite, l'objectif de contrôle optimal est d'assurer que le robot mobile non-holonome suit la trajectoire désirée $q_{a,d}(t)$ tout en minimisant simultanément un indice de performance quadratique donné.
Selon la technique de commande optimale, un indice de performance quadratique est défini comme

$$J = \int_{t_0}^{\infty} L(\tilde{x},u)\, dt\ , \tag{3.37}$$

avec le Lagrangien

$$L(\tilde{x},u) = \tfrac{1}{2}\tilde{x}^T(t)Q\tilde{x}(t) + \tfrac{1}{2}u(t)^T Ru(t)\ , \tag{3.38}$$

où les matrices Q et R sont symétriques définies positives et $\tilde{\mathrm{x}} = [e^T(t), \dot{e}^T(t)]^T$.

La condition nécessaire et suffisante pour minimiser (3.37) associée au système (3.36) est de choisir la fonction $V(\tilde{z},t)$ satisfait l'équation HJB suivante

$$\frac{\partial V(\tilde{z}(t),t)}{\partial t} + \min_u H\left(\tilde{z},u,\frac{\partial V(\tilde{z}(t),t)}{\partial t}\right) = 0, \tag{3.39}$$

où le Hamiltonien d'optimisation H est défini comme

$$H\left(\tilde{z},u,\frac{\partial V(\tilde{z}(t),t)}{\partial t}\right) = \frac{\partial V(\tilde{z}(t),t)}{\partial \tilde{z}}\dot{\tilde{z}} + L(\tilde{x},u). \tag{3.40}$$

En outre, la fonction $V(\tilde{z}(t),t)$ satisfait l'équation différentielle partielle suivante

$$-\frac{\partial V(\tilde{z}(t),t)}{\partial t} = \frac{\partial V(\tilde{z}(t),t)}{\partial \tilde{z}}\dot{\tilde{z}} + L(\tilde{x},u^*). \tag{3.41}$$

Le commande optimale pour laquelle le Hamiltonien H soit minimal est obtenue à partir de la dérivée partielle par rapport à $u(t)$. Ainsi, le minimum $\min_u H$ est garanti pour la commande $u = u^*$ telle que

$$H^* = \min_u H = \min\left(\frac{\partial V(\tilde{z}(t),t)}{\partial \tilde{z}}\dot{\tilde{z}} + L(\tilde{x},u)\right). \quad (3.42)$$

Lemme 1 : *La fonction $V(\tilde{z}(t),t)$ suivante satisfait l'équation HJB associée au système (3.36)*

$$V(\tilde{z}(t),t) = \frac{1}{2}\tilde{z}^T(t)\begin{pmatrix} K & 0 \\ 0 & \bar{M}(q) \end{pmatrix}\tilde{z}(t) = \frac{1}{2}\tilde{z}^T(t)P(q)\tilde{z}(t), \quad (3.43)$$

pour $K = K^T > 0$ et $\hat{\Lambda} = \begin{pmatrix} I_{p\times p} & 0_{p\times p} \\ \Lambda & I_{p\times p} \end{pmatrix}$ solutions de l'équation matricielle algébrique suivante

$$\begin{pmatrix} 0 & K \\ K & 0 \end{pmatrix} - \hat{\Lambda}^T\begin{pmatrix} 0 & 0 \\ 0 & R^{-1} \end{pmatrix}\hat{\Lambda} + Q = 0_{2p\times 2p}\,. \quad (3.44)$$

La commande optimale $u(t) = u^*(t)$ qui minimise $V(\tilde{z}(t),t)$ est une solution explicite directe et non basée modèle définie par

$$u^*(t) = -R^{-1}\big(\dot{e}(t) + \Lambda e(t)\big). \quad (3.45)$$

Preuve : La commande optimale $u(t) = u^*(t)$ pour que le Hamiltonien H soit minimal est calculée à partir de la dérivée partielle de H (3.39) par rapport à $u(t)$

$$\frac{\partial}{\partial u}H\left(\tilde{z},u,\frac{\partial V}{\partial t}\right) = 0. \quad (3.46)$$

Ainsi, on obtient

$$\frac{\partial}{\partial u}\left(\frac{\partial V(\tilde{z}(t),t)}{\partial \tilde{z}}\dot{\tilde{z}} + L(\tilde{x},u)\right) = 0, \quad (3.47)$$

où la dérivée partielle de la fonction $V(\tilde{z}(t),t)$ par rapport à la variable d'état donne $\frac{\partial V}{\partial \tilde{z}} = \tilde{z}^T(t)P$ et le Lagrangien est donné par (3.38). Donc l'équation (3.47) devient

$$\frac{\partial}{\partial u}\left(\tilde{z}^T(t)P(A\tilde{z} + Bu) + \frac{1}{2}\tilde{x}^T(t)Q\tilde{x}(t) + \frac{1}{2}u(t)^T Ru(t)\right) = \tilde{z}^T(t)PB + u(t)^T R\,. \quad (3.48)$$

Par conséquent, $u^*(t)$ qui minimise (3.48) soit

$$u^*(t) = -R^{-1}B^T P\tilde{z}(t) = -R^{-1}\big(\dot{e}(t) + \Lambda e(t)\big). \quad (3.49)$$

D'autre part, la condition nécessaire et suffisante pour l'optimalité de la commande $u^*(t)$ est de choisir la fonction $V(\tilde{z}(t), t)$ satisfait l'équation (3.50) telle que

$$\frac{\partial V}{\partial t} + H^* = 0. \quad (3.50)$$

La dérivée partielle par rapport à t de $V(\tilde{z}(t), t)$ est $\frac{\partial V(\tilde{z}(t),t)}{\partial t} = \frac{1}{2}\tilde{z}^T(t)\dot{P}\tilde{z}(t)$ et le Hamiltonien optimal est donné par

$$\begin{aligned} H^* &= \frac{\partial V}{\partial \tilde{z}}\dot{\tilde{z}} + L(\tilde{x}, u^*), \\ &= \tilde{z}^T(t)P(A\tilde{z} + Bu) + \frac{1}{2}\tilde{x}^T(t)Q\tilde{x}(t) + \frac{1}{2}u^*(t)^T R u^*(t). \end{aligned} \quad (3.51)$$

En remplaçant $u^*(t)$ (3.49) dans (3.51) et on recalcule (3.50), on aura

$$\tilde{x}^T\left[\hat{\Lambda}^T\left(PA + \frac{1}{2}\dot{P} - \frac{1}{2}P^T BR^{-1}B^T P\right)\hat{\Lambda} + \frac{1}{2}Q\right]\tilde{x} = 0, \quad (3.52)$$

avec $\tilde{z}(t) = \hat{\Lambda}\tilde{x}(t)$.

Comme $PA = \frac{1}{2}(PA + A^T P)$, on obtient

$$\tilde{x}^T\big[\hat{\Lambda}^T\big(PA + A^T P + \dot{P} - P^T BR^{-1}B^T P\big)\hat{\Lambda} + Q\big]\tilde{x} = 0. \quad (3.53)$$

D'autre part, on a

$$\tilde{z}^T\big(PA + A^T P + \dot{P}\big)\tilde{z} = \tilde{z}^T\begin{pmatrix} -2K\Lambda & 2K \\ 0 & \dot{\bar{M}} - 2\bar{C} \end{pmatrix}\tilde{z}. \quad (3.54)$$

On prend en compte de la propriété P2, l'équation (3.54) est réduit à

$$\tilde{x}^T\hat{\Lambda}^T\big(PA + A^T P + \dot{P}\big)\tilde{x}\hat{\Lambda} = \tilde{x}^T\begin{pmatrix} 0 & 2K \\ 0 & 0 \end{pmatrix}\tilde{x} = \tilde{x}^T\begin{pmatrix} 0 & K \\ K & 0 \end{pmatrix}\tilde{x}. \quad (3.55)$$

En remplaçant (3.55) dans (3.52) et le fait que $P^T BR^{-1}B^T P = \begin{pmatrix} 0 & 0 \\ 0 & R^{-1} \end{pmatrix}$, finalement l'équation $V(\tilde{z}(t), t)$ est une solution de HJB pour $u = u^*$ avec les matrices K et $\hat{\Lambda}$ représentent une solution de l'équation matricielle algébrique suivante

$$\begin{pmatrix} 0 & \mathrm{K} \\ \mathrm{K} & 0 \end{pmatrix} - \hat{\Lambda}^{\mathrm{T}}\begin{pmatrix} 0 & 0 \\ 0 & \mathrm{R}^{-1} \end{pmatrix}\hat{\Lambda} + \mathrm{Q} = 0_{2\mathrm{p}\times 2\mathrm{p}}\,. \quad (3.56)$$

Les matrices Q et K sont choisies comme suit

$$\mathrm{Q} = \mathrm{Q}^{\mathrm{T}} = \begin{pmatrix} K\Lambda & 0 \\ 0 & R^{-1} \end{pmatrix} > 0, \quad K = K^T = R^{-1}\Lambda > 0. \tag{3.57}$$

Notons que la fonction quadratique $V(\tilde{z}(t), t)$ (3.43) est une fonction de Lyapunov pour le système en boucle fermée par la commande u^* (3.49). La dérivée $\dot{V}(\tilde{z}(t), t)$ est définie négative ainsi que l'erreur $\tilde{z}(t)$ est globale uniforme asymptotique stable. À partir de de la solution de l'équation HJB (3.41), on aura

$$\dot{V}(\tilde{z}(t), t) = -\frac{1}{2}(\tilde{x}^T(t)Q\tilde{x}(t) + u^*(t)^T R u^*(t)). \tag{3.58}$$

En remplaçant (3.49) et (3.56) dans (3.58), et le fait que $\tilde{x}(t) = \hat{\Lambda}^{-1}\tilde{z}(t)$, on obtient

$$\begin{aligned} \dot{V}(\tilde{z}(t), t) &= -\frac{1}{2}\tilde{z}^T(t)\left(\hat{\Lambda}^{-T}Q\hat{\Lambda}^{-1} + P^T B R^{-1} B^T P\right)\tilde{z}(t) \\ &= -\frac{1}{2}\tilde{z}^T(t)\begin{pmatrix} 2K\Lambda & -K \\ -K & 2R^{-1} \end{pmatrix}\tilde{z}(t) < 0, \quad \forall t > 0, \quad \|\tilde{z}\| \neq 0. \end{aligned} \tag{3.59}$$

III.6. Conception de la commande neuro optimale robuste adaptative :

En respectant toujours l'hypothèse imposée sur la connaissance du modèle du système. La conception des fonctions non linaires inconnues considérées dans la loi de commande optimale, est basés sur le principe de l'approximation ou la compensation par un réseau de neurones adaptatif.

Etant donné un vecteur d'entrée $x = \left[x_1, x_2 \dots x_{N_1}\right]^T \in \mathbb{R}^{N_1}$, l'équation d'un réseau de neurones à trois couches peut être représentée sous forme matricielle. La relation entrées-sorties est donnée par

$$O = W_1^T \psi(W_2^T I), \tag{3.60}$$

avec $O \in \mathbb{R}^{N_3}$ et $I \in \mathbb{R}^{N_1}$ sont les entrées et les sorties de réseau de neurones, respectivement. $W_1^T \in \mathbb{R}^{N_3 \times N_2}$ sont les poids d'interconnexion entre les neurones dans la couche cachée et les sortie, $W_2^T \in \mathbb{R}^{N_2 \times N_1}$ sont les poids d'interconnexion entre les entrées et la couche cachée, ψ est la fonction d'activation réalisée par chaque neurone dans la couche cachée et N_2 est le nombre de neurones dans la couche cachée.

L'avantage intéressant dans les réseaux de neurones adaptatif, est qu'ils n'ont pas besoin d'un modèle du système pour la conception d'un algorithme d'adaptation, de sorte que la dynamique du robot est compensée en temps réel par la reconstruction d'une fonction mise dans la loi de commande qui minimise l'erreur de suivi de trajectoire et non pour identifier le modèle réel du système. Dans une partie de la théorie de l'approximation universelle, un réseau neuronal pourrait implémenter toute fonction à condition qu'elle soit continue.

Ainsi, avec un réseau neuronal à trois couches (3.60), la fonction non linéaire $H(r_e)$ (3.34) peut être supposée représentée par le système neuronal comme suit

$$H(r_e) = W_1^T \psi(W_2^T r_e) + \varepsilon(r_e) + \bar{\tau}_d(t, q). \tag{3.61}$$

Donc la fonction estimée $\widehat{H}(r_e)$ de $H(r_e)$ s'écrit

$$\widehat{H}(r_e) = \widehat{W}_1^T \psi\left(\widehat{W}_2^T r_e\right), \tag{3.62}$$

avec $\widehat{W}_1^T$ et $\widehat{W}_2^T$ les poids estimés de W_1^T et W_2^T, respectivement. La fonction tangente hyperbolique est choisie comme fonction d'activation $\psi = \tanh(.)$.

Dans la suite, une loi de commande neuro optimale robuste est proposée afin d'assurer les performances de suivi de trajectoire dans la présence de l'erreur d'approximation de réseau de neurones et les différentes perturbations externes. De cette façon, la connaissance de la dynamique du robot peut être surmontée par le système neuronal ce qui permet d'éviter la phase de modélisation classique des systèmes dynamiques.

La loi de commande est donnée par

$$\bar{D}(q)\tau(t) = -u^*(t) + \widehat{H}(r_e) + u_R(t), \tag{3.63}$$

avec $u^*(t)$ est la commande optimale donnée par (3.49), $\widehat{H}(r_e)$ est le système d'approximation neuronal adaptatif. L'approche de contrôle robuste est de supposer a priori qu'il y aura des incertitudes ou perturabtions, le terme auxilaire $u_R(t)$ repésente la commande robuste adaptative qui permet de compenser l'erreur d'approximation de réseau de neurones et éventuelles perturbations externes.

Le terme erreur d'approximation se réfère aux différences entre le système neuronal et le système réel. Il est important de noter que l'erreur d'approximation et les perturbations

physique externes du système ont exactement le même effet où l'erreur de reconstruction du réseau de neurones peut être vu comme une perturbation logicielle.

Le problème de la commande neuronale se pose comme suit ; pour la loi de commande définie par (3.63), déterminer la loi d'adaptation des différents poids du réseau de neurones $\widehat{W}_1^T$ et $\widehat{W}_2^T$ pour estimer en ligne la dynamique non linéaire représentée dans (3.61) et reconstruire une fonction qui minimise l'erreur de suivi de trajectoire. Nous déterminerons plus tard les lois adaptatives du réseau de neurones à partir de l'analyse de la stabilité de Lyapunov.

En soustrayant l'équation (3.62) de (3.61), l'erreur d'approximation $\tilde{H}(r_e)$ est définie par

$$\tilde{H}(r_e) = H(r_e) - \hat{H}(r_e) = W_1^T \psi(W_2^T r_e) - \widehat{W}_1^T \psi\left(\widehat{W}_2^T r_e\right) + \varepsilon(r_e) + \bar{\tau}_d(t,q). \quad (3.64)$$

Le développement en série de Taylor de la fonction d'activation pour r_e donné s'écrit

$$\psi(W_2^T r_e) = \psi\left(\widehat{W}_2^T r_e\right) + \psi'\left(\widehat{W}_2^T r_e\right)\ (W_2^T - \widehat{W}_2^T) r_e + O^2, \quad (3.65)$$

où ψ' est la jaconbienne et $O(H)^2$ est le terme inconnu d'ordre deux supposé borné.

On définie l'erreur de l'estimation des poids $\widetilde{W}_1^T = W_1^T - \widehat{W}_1^T$ et $\widetilde{W}_2^T = W_2^T - \widehat{W}_2^T$, en remplaçant (3.65) dans (3.64), on obtient

$$\begin{aligned}
\tilde{H}(r_e) &= W_1^T \psi(W_2^T r_e) - \widehat{W}_1^T \psi\left(\widehat{W}_2^T r_e\right) + \varepsilon(r_e) + \bar{\tau}_d(t,q), \\
&= \left(\widetilde{W}_1^T + \widehat{W}_1^T\right)\left[\psi\left(\widehat{W}_2^T r_e\right) + \psi'\left(\widehat{W}_2^T r_e\right) \widetilde{W}_2^T r_e + O^2\right] - \widehat{W}_1^T \psi\left(\widehat{W}_2^T r_e\right) + \varepsilon(r_e) + \bar{\tau}_d(t,q) \\
&= \widetilde{W}_1^T \left[\psi\left(\widehat{W}_2^T r_e\right) - \psi'\left(\widehat{W}_2^T r_e\right) \widehat{W}_2^T r_e\right] + \widehat{W}_1^T \psi'\left(\widehat{W}_2^T r_e\right) \widetilde{W}_2^T r_e + \Psi, \quad (3.66)
\end{aligned}$$

avec $\Psi = \widetilde{W}_1^T \psi'\left(\widehat{W}_2^T r_e\right) W_2^T r_e + W_1^T O^2 + \varepsilon(r_e) + \bar{\tau}_d(t,q)$ est la perturbation globale.

Notant que les perturbations réelles qui affectent les performances de suivi de trajectoire d'un robot manipulateur mobile sont de sources multiples, interne ou externe sont traitées de la même manière par la loi de commande. La norme de la perturbation Ψ est supposée bornée par uns constante scalaire positive inconnue nommée la borne supérieure inconnue; $\|\Psi\| \leq \mu$. Par conséquent, dans cette hypothèse, la seule exigence imposée est que la perturbation doit être bornée, mais la valeur numérique de cette limite supérieure n'est pas nécessairement

connue. En effet, cette limite supérieure μ est estimée à l'aide d'un algorithme adaptatif qui sera conçu plus tard.

En remplaçant (3.49) et (3.63) dans (3.36) et en utilisant (3.66), la dynamique complète en boucle fermée de l'erreur peut être représentée par

$$\begin{aligned}\dot{\tilde{z}}(t) &= A(q)\tilde{z}(t) + B(q)\left[H(r_e) - \widehat{H}(r_e) + u^*(t) - u_R\right],\\ &= A(q)\tilde{z}(t) + B(q)\left[\widetilde{H}(r_e) + u^*(t) - u_R\right],\\ &= A(q)\tilde{z}(t) + B\left[\widetilde{W}_1^T\left[\psi\left(\widehat{W}_2^T r_e\right) - \psi'^{\left(\widehat{W}_2^T r_e\right)}\widehat{W}_2^T r_e\right]\right],\\ &\qquad +B\left[\widehat{W}_1^T\psi'\left(\widehat{W}_2^T r_e\right)\widetilde{W}_2^T r_e + \Psi + u^*(t) - u_R\right],\\ &= (A - BR^{-1}B^T P)\tilde{z}(t) \; + B[\widetilde{W}_1^T\left(\psi\left(\widehat{W}_2^T r_e\right) - \psi'\left(\widehat{W}_2^T r_e\right)\widehat{W}_2^T r_e\right),\\ &\qquad + \widehat{W}_1^T\psi'\left(\widehat{W}_2^T r_e\right)\widetilde{W}_2^T r_e + \Psi - u_R]. \qquad (3.67)\end{aligned}$$

Dans la conception des lois de commandes pour des systèmes non linéaires, nous étudions souvent la stabilité d'un système en boucle fermée en examinant le comportement d'une fonction candidate de Lyapunov.

Théoriquement, l'objectif de la commande est atteint lorsque la fonction de Lyapunov associée au système en boucle fermée tend vers zéro sous certain théorème d'analyse. Quand l'incertitude ou une perturbation d'un système tend à empêcher la convergence de la fonction de Lyapunov vers zéro, nous ajouterons simplement un terme de stabilisation auxiliaire à la loi de commande afin de compenser cet effet, et améliorer la robustesse de la loi de commande dans la présence des éventuelles perturbations. En effet, la loi de commande associée au système est dite robuste de point de vue stabilité si la stabilité est assurée malgré la présence de la perturbation, que représente une condition nécessaire mais n'est pas suffisante pour certains systèmes. D'autre part, si la loi de commande peut garantir des performances données malgré la présence d'une perturbation, alors on parle de la robustesse de point de vue performance.

Théorème : *Considérons le système robotique représenté par (3.25) et (3.26) avec un modèle dynamique inconnue et des perturbations externes. En utilisant la loi de commande définie par (3.63) dans laquelle, la loi optimale est donnée par (3.49), le système neuronal est conçu comme (3.62) et la commande robuste adaptative est donnée par (3.68) dans lesquelles, les algorithmes d'adaptation pour le système neuronal sont donnés par (3.69) et (3.70), et*

l'algorithme d'adaptation pour le terme robuste est donné par (3.71). Donc, l'erreur de suivi de trajectoire $\|\tilde{z}(t)\|$ *est asymptotique stable.*

$$u_R = \text{sign}(\tilde{z}^T(t)PB)^T\hat{\mu}(t), \tag{3.68}$$

$$\dot{\widehat{W}}_1(t) = \sigma_{W_1}\left(\psi\left(\widehat{W}_2^T r_e\right) - \psi'\left(\widehat{W}_2^T r_e\right)\widehat{W}_2^T r_e\right)\tilde{z}^T PB, \tag{3.69}$$

$$\dot{\widehat{W}}_2(t) = \sigma_{W_2}\left(r_e\tilde{z}^T PB\widehat{W}_1^T\psi'\left(\widehat{W}_2^T r_e\right)\right), \tag{3.70}$$

$$\dot{\hat{\mu}}(t) = \sigma_\mu \text{sign}(\tilde{z}^T PB)B^T P\tilde{z}, \tag{3.71}$$

avec σ_{W_1}, σ_{W_2} *et* σ_μ*sont des constantes positives, et* sign(.) *est la fonction signe.*

Preuve : On définit la fonction candidate de Lyapunov suivante

$$V\left(\tilde{z}, \widetilde{W}_i, \tilde{\mu}, t\right) = \frac{1}{2}\tilde{z}^T(t)P(q)\tilde{z}(t) + \sum_{i=1}^{2}\frac{1}{2\sigma_{W_i}}tr(\widetilde{W}_i^T\widetilde{W}_i) + \frac{1}{2\sigma_\mu}\tilde{\mu}^2\,, \tag{3.72}$$

où $\widetilde{W}_i = W_i - \widehat{W}_i$ et $\tilde{\mu} = \mu - \hat{\mu}$ sont les erreurs d'estimation et $tr(.)$ l'opérateur $trac$ d'une matrice.

La dérivée de la fonction de Lyapunov s'écrit

$$\begin{aligned}\dot{V}\left(\tilde{z}, \widetilde{W}_i, \tilde{\mu}, t\right) &= \tilde{z}^T P(q)\dot{\tilde{z}} + \frac{1}{2}\tilde{z}^T\dot{P}(q)\tilde{z} + \sum_{i=1}^{2}\frac{1}{\sigma_{W_i}}tr(\widetilde{W}_i^T\dot{\widetilde{W}}_i) + \frac{1}{\sigma_\mu}\tilde{\mu}\dot{\tilde{\mu}}\,,\\ &= \tilde{z}^T\left(PA - PBR^{-1}B^TP + \frac{1}{2}\dot{P}\right)\tilde{z}\\ &\quad + \tilde{z}^T PB\left[\widetilde{W}_1^T\left(\psi\left(\widehat{W}_2^T r_e\right) - \psi'\left(\widehat{W}_2^T r_e\right)\widehat{W}_2^T r_e\right) + \widehat{W}_1^T\psi'\left(\widehat{W}_2^T r_e\right)\widehat{W}_2^T r_e + \Psi\right.\\ &\quad \left. - u_R(t)\right]\\ &\quad - \sum_{i=1}^{2}\frac{1}{\sigma_{W_i}}tr(\widetilde{W}_i^T\dot{\widehat{W}}_i) - \frac{1}{\sigma_\mu}\tilde{\mu}\dot{\hat{\mu}}\,.\end{aligned} \tag{3.73}$$

À partir de (3.59), on a $PA - PBR^{-1}B^TP + \frac{1}{2}\dot{P} == -\frac{1}{2}\begin{pmatrix}2K\Lambda & -K\\ -K & 2R^{-1}\end{pmatrix} = -F$. Comme $x^Ty = tr(yx^T)$, la fonction (3.73) devient

$$\begin{aligned}\dot{V}\left(\tilde{z}, \widetilde{W}_i, \tilde{\mu}, t\right) &= -\tilde{z}^T F\tilde{z} + tr\left\{\widetilde{W}_1^T\left[\left(\psi\left(\widehat{W}_2^T r_e\right) - \psi'\left(\widehat{W}_2^T r_e\right)\widehat{W}_2^T r_e\right)\tilde{z}^T PB - \frac{1}{\sigma_{W_1}}\dot{\widehat{W}}_1\right]\right\},\\ &\quad + tr\left\{\widetilde{W}_2^T\left[r_e\tilde{z}^T PB\widehat{W}_1^T\psi'\left(\widehat{W}_2^T r_e\right) - \frac{1}{\sigma_{W_2}}\dot{\widehat{W}}_2\right]\right\} + \tilde{z}^T PB(\Psi - u_R(t)) - \frac{1}{\sigma_\mu}\tilde{\mu}\dot{\tilde{\mu}}\,,\end{aligned} \tag{3.74}$$

En remplaçant les algorithmes d'adaptation du système neuronal (3.69) et (3.70) dans (3.74), on aura

$$\dot{V}\left(\tilde{z}, \widetilde{W}_i, \tilde{\mu}, t\right) = -\tilde{z}^T F\tilde{z} + \tilde{z}^T PB[\Psi - u_R(t)] - \frac{1}{\sigma_\mu}\tilde{\mu}\dot{\hat{\mu}}\,. \tag{3.75}$$

Maintenant, on remplace le terme robuste (3.68) et son algorithme d'adaptation (3.71) dans (3.75), on aura ainsi

$$\begin{aligned}\dot{V}\left(\tilde{z}, \widetilde{W}_i, \tilde{\mu}, t\right) &= -\tilde{z}^T F\tilde{z} + \tilde{z}^T PB[\Psi - \text{sign}(\tilde{z}^T PB)^T\hat{\mu}] - \tilde{\mu}sign(\tilde{z}^T PB)B^T P\tilde{z}, \\ &= -\tilde{z}^T F\tilde{z} + \tilde{z}^T PB[\Psi - \text{sign}(\tilde{z}^T PB)^T\mu].\end{aligned} \tag{3.76}$$

Comme $\|\Psi\| \leq \mu$, et en utilisant l'inégalité de Cauchy-Schwarz, on a

$$\begin{aligned}\tilde{z}^T PB[\Psi - \text{sign}(\tilde{z}^T PB)^T\mu] &\leq \|\tilde{z}^T PB\|\,(\|\Psi\| - \mu), \\ &\leq 0.\end{aligned} \tag{3.77}$$

Finalement, la dérivée de la fonction de Lyapunov satisfait

$$\dot{V}\left(\tilde{z}, \widetilde{W}_i, \tilde{\mu}, t\right) \leq -\lambda_{min}\{F\}\|\tilde{z}\|^2 \leq 0. \tag{3.78}$$

D'après le théorème de stabilité de Lyapunov **[88]**, le Lemme de Barbalat indique que $\dot{V}\left(\tilde{z}, \widetilde{W}_i, \tilde{\mu}, t\right)$ converge vers zéro lorsque t tend vers l'infinie. Ainsi, l'erreur de suivi de trajectoire $\|\tilde{z}\|^2$ est globalement asymptotique stable.

Remarque : Le terme robuste $u_R(t)$ (3.68) est une loi auxiliaire dans la commande qui permet de compenser les différentes perturbations et de garantir la robustesse. Le terme scalaire $\hat{\mu}(t)$ dans (3.68) est conçu afin d'estimer la borne supérieur de la perturbation globale dans le système, qui est certainement insuffisant pour éliminer l'effet des perturbations arbitraires présentées dans chaque sous-système simultanément. En effet, une estimation inappropriée de la borne supérieure pour l'un de sous-systèmes peut causer une dégradation de performance pour le système globale dans la pratique notamment la saturation de la commande et des vibrations dans le mouvement. D'autre part, une valeur optimale de la borne supérieure qui permet de compenser au même temps chaque perturbation peut ne pas être facilement obtenue par l'algorithme d'adaptation (3.71) en raison de la complexité de la structure des perturbations.

Par conséquent, une démarche différente est introduite pour estimer la limite supérieure, de sorte que pour chaque sous-système de robot, sa perturbation locale sera supposée bornée par une limite supérieure dans laquelle l'estimation sera traitée localement. De cette façon, nous

utilisons plusieurs bornes adaptatives pour estimer la limite supérieure pour chaque sous-système indépendamment. En effet, la loi de commande proposée conduit à une meilleure estimation et garantit une erreur minimale dans la présence des perturbations multiples dans chaque sous-système. Le schéma fonctionnel de la commande est présenté sur la figure 3.3, elle est composée de la loi de commande optimale comprend un système neuronal adaptatif ainsi qu'une commande robuste adaptative

Supposition 1: chaque élément du vecteur de la perturbation $\Psi = [\Psi_1, \Psi_2, \dots, \Psi_p]^T$ est borné par une constante positive inconnue notée la limite supérieure locale inconnue; $\Psi_i \leq \omega_i$, et où, la norme du vecteur Ψ est bornée par le minimum entre $a_i\omega_i$ qui représente la limite supérieure globale inconnue ; $\|\Psi\| \leq \lambda_{min}(W_3)$ où la matrice $W_3 = \text{diag}[a_i\omega_i] \in \mathbb{R}^{p\times p}$ avec a_i certain constante positive inconnue.

La simple condition imposée ici est que les perturbations locales doivent être bornées, mais il n'est pas nécessaire de connaître la valeur de chaque limite supérieure locale. Ainsi, les multiples limites supérieures sont estimées par la loi d'adaptation proposée dans le prochain théorème.

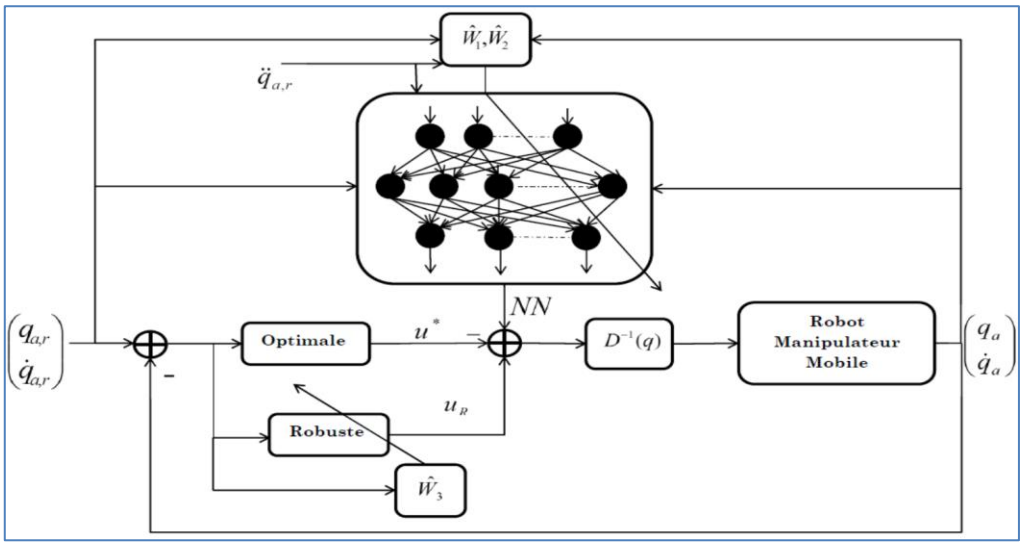


Figure.3.3. Schéma bloc de la loi de commande proposée.

Théorème : *Considérons le système robotique représenté par (3.25) et (3.26), avec des dynamiques non modélisée et des perturbations externes. Si l'entrée de commande est conçue comme (3.63) pour laquelle, la commande robuste est donnée par (3.79) et l'algorithme d'estimation adaptatif des limites supérieures inconnues est conçu comme (3.80). Donc, l'erreur de suivi de trajectoire* $\|\tilde{z}(t)\|$ *est globalement asymptotique stable*

$$u_R(t) = \widehat{W}_3^T \text{sign}\big(B^T P\tilde{z}(t)\big), \tag{3.79}$$

$$\dot{\widehat{W}}_3(t) = \sigma_{W_3} \text{diag}\Big[\big(\text{sign}\big(B^T P\tilde{z}(t)\big)\tilde{z}^T(t)PB\big)_{ii}\Big], \tag{3.80}$$

avec $\widehat{W}_3 = diag[\widehat{\omega}_i] \in \mathbb{R}^{p\times p}$ *la matrice estimée de* W_3 *et* σ_{W_3} *constante positive,*

Preuve : On considère la fonction candidate de Lyapunov suivante

$$V\big(\tilde{z}, \widetilde{W}_i, t\big) = \frac{1}{2}\tilde{z}^T(t)P(q)\tilde{z}(t) + \sum_{i=1}^{3} \frac{1}{2\sigma_{W_i}} tr\big(\widetilde{W}_i^T \widetilde{W}_i\big)\,, \tag{3.81}$$

où $\widetilde{W}_i = W_i - \widehat{W}_i$ sont les erreurs d'estimation de réseau de neurones et la commande robuste.

En calculant la dérivée de la fonction de Lyapunov, on aura

$$\dot{V}\big(\tilde{z}, \widetilde{W}_i, t\big) = \tilde{z}^T P(q)\dot{\tilde{z}} + \frac{1}{2}\tilde{z}^T \dot{P}(q)\tilde{z} + \sum_{i=1}^{3} \frac{1}{\sigma_{W_i}} tr\big(\widetilde{W}_i^T \dot{\widetilde{W}}_i\big)\,,$$

$$= -\tilde{z}^T F\tilde{z} + \tilde{z}^T PB[\widetilde{W}_1^T\big(\psi\big(\widehat{W}_2^T r_e\big) - \psi'\big(\widehat{W}_2^T r_e\big)\widehat{W}_2^T r_e\big),$$

$$+\widehat{W}_1^T \psi'\big(\widehat{W}_2^T r_e\big)\widetilde{W}_2^T r_e + \Psi - u_R(t)] - \sum_{i=1}^{3} \frac{1}{\sigma_{W_i}} tr\big(\widetilde{W}_i^T \dot{\widehat{W}}_i\big). \tag{3.82}$$

Comme $x^T y = tr(yx^T)$ et en remplaçant les lois d'adaptation neuronales (3.69) et (3.70), la dérivée de Lyapunov (3.82) devient

$$\dot{V}\big(\tilde{z}, \widetilde{W}_i, t\big) = -\tilde{z}^T F\tilde{z} + \tilde{z}^T PB[\Psi - u_R(t)] - \frac{1}{\sigma_{W_3}} tr\big(\widetilde{W}_3^T \dot{\widehat{W}}_3\big). \tag{3.83}$$

Finalement, en remplaçant la loi robuste modifiée (3.79) et son algorithme d'adaptation (3.80), et nous utilisons la relation suivante

$$\tilde{z}^T PB\widehat{W}_3^T \text{sign}\big(B^T P\tilde{z}(t)\big) = tr\big(\widehat{W}_3^T \text{sign}\big(B^T P\tilde{z}(t)\big)\tilde{z}^T PB\big),$$

$$= tr\Big\{\widehat{W}_3^T \text{diag}\big[\text{sign}\big(B^T P\tilde{z}(t)\big)\tilde{z}^T PB\big]_{ii}\Big\}. \tag{3.84}$$

On obtient

$$\dot{V}\big(\tilde{z}, \widetilde{W}_i, t\big) = -\tilde{z}^T F\tilde{z} + \tilde{z}^T PB\big[\Psi - W_3^T sign\big(B^T P\tilde{z}(t)\big)\big]. \tag{3.85}$$

En utilisant l'inégalité de Cauchy-Schwarz et la supposition 1, on aura

$$\tilde{z}^T PB[\Psi - W_3^T sign(B^T P\tilde{z})] \leq \|\tilde{z}^T PB\|\,(\|\Psi\| - \lambda_{min}(W_3)\,) \leq 0, \tag{3.86}$$

Donc, la dérviée de la fonction de Lyapunov (3.85) satisfait

$$\dot{V}\left(\tilde{z}, \widetilde{W}_i, t\right) \leq -\lambda_{min}\{F\}\|\tilde{z}\|^2 \leq 0. \tag{3.87}$$

D'après le Lemme de Barbalat, l'erreur $\|\tilde{z}\|$ est globalement asymptotique stable.

Remarque : Mathématiquement et d'après la théorie de Lyapunov, le terme robuste garantie la stabilité dans la présence des perturbations bornées. Cependant, la loi d'adaptation $\dot{\widehat{W}}_3(t)$ (3.80) est calculée par l'integrale d'une fonction positive $\forall t \geq 0$, ce qui signifie que la fonction estimée $\dot{\widehat{W}}_3(t)$ est une fonction monotones croissante, ainsi donc, elle atteint une valeur constante si et seulement si l'erreur filtrée $r(t)$ a une valeur de zéro, ce qui est pratiquement impossible d'avoir une erreur nulle. D'autre part, la vitesse et la résolution de l'adaptation de la fonction $\widehat{W}_3(t)$ est affectées par le gain σ_{W_3}. à cet effet, les valeurs éstimées de $\widehat{W}_3(t)$ peuvent dépasser les bornes réelles des perturbations ce qui provoque des saturations de la commande. Pour resoudre ce problème, nous proposons de modifer la loi d'adaptation oû la démonstartion de la stabilté est présentée dans la partie suivante.

La loi d'adaptation modifiée de la commande robuste est donnée par

$$\dot{\widehat{W}}_3(t) = \sigma_{W_3}\left(-\alpha\|\tilde{z}^T(t)PB\|\widehat{W}_3(t) + \text{diag}\left[\left(\text{sign}\left(B^T P\tilde{z}(t)\right)\tilde{z}^T(t)PB\right)_{ii}\right]\right), \tag{3.88}$$

avec α est une constante positive.

Preuve : Nous consédirons la même démarche précidante. En remplaçant la commande robuste (3.79) et la loi d'adaptation modiée (3.88) dans l'équation (3.83), on obtient

$$\dot{V}\left(\tilde{z}, \widetilde{W}_i, t\right) = -\tilde{z}^T F\tilde{z} + \alpha\|\tilde{z}^T(t)PB\|tr\left(\widetilde{W}_3^T\widehat{W}_3\right). \tag{3.89}$$

D'une part, on a

$$tr\left(\widetilde{W}_3^T\widehat{W}_3\right) = tr\left(\widetilde{W}_3^T W_3\right) - \left\|\widetilde{W}_3^T\right\|_F^2 \leq \left\|\widetilde{W}_3^T\right\|_F \left(W_{sup} - \left\|\widetilde{W}_3^T\right\|_F\right), \tag{3.90}$$

avec $\|.\|_F$ est la norme Frobenius et $\|W_3^T\|_F \leq W_{sup}$.

D'autre part, on a $\|\tilde{z}^T(t)PB\| = \|r(t)\|$ et $\|\tilde{z}(t)\|^2 = \|r(t)\|^2 + \|e(t)\|^2$, donc la dérivée de la fonction de Lyapunov (3.89) satisfait

$$\dot{V}(\tilde{z}, \widetilde{W}_i, t) \leq -\lambda_{min}\{F\}\|e\|^2 - \|r(t)\|\left[\lambda_{min}\{F\}\|r(t)\| + \alpha\left\|\widetilde{W}_3^T\right\|_F\left(\left\|\widetilde{W}_3^T\right\|_F - W_{sup}\right)\right].$$

(3.91)

Une condition suffisante pour que la dérivée de Lyapunov devient négative est que le terme entre crochet soit positif.

Ainsi

$$\dot{V}(\tilde{z}, \widetilde{W}_i, t) \leq -\lambda_{min}\{F\}\|e\|^2 - \|r(t)\|\left[\lambda_{min}\{F\}\|r(t)\| + \alpha\left(\left\|\widetilde{W}_3^T\right\|_F - \frac{W_{sup}}{2}\right)^2 - \alpha\frac{W_{sup}^2}{4}\right]$$

$$\leq 0, \qquad (3.92)$$

quand la condition suivante soit satisfaite

$$\|r(t)\| \geq \alpha\frac{W_{sup}^2}{4\lambda_{min}\{F\}} \quad \text{ou} \quad \left\|\widetilde{W}_3^T\right\|_F \geq W_{sup}. \qquad (3.93)$$

D'après la théorie standard de Lyapunov **[88]**, l'erreur $\|r(t)\|$, $\|e(t)\|$ et le signal de commande sont uniformement bornés (UUB), ainsi, l'erreur de suivi de trajectoire converge vers la région autour de zéro définie par (3.93).

Remarque : La région attractive associée à l'erreur de suivi de trajectoire dans (3.93) peut être minimisée avec le réglage des paramètres α et $\lambda_{min}\{F\}$. Ainsi, la loi d'adaptation pour le système neuronal qui est fondée sur Lyapunov permet l'estimation en temps réel des poids du réseau de neurones. Le nombre de neurones dans la couche cachée N_2 doit être choisi de façon à avoir une bonne performance. En général, augmenter le nombre N_2 réduit l'erreur d'approximation, mais on cherche toujours un nombre minimal de neurones pour obtenir une performance désirée. La meilleur façon est la méthode ascendante qui consiste à augmenter itérativement le nombre de neurones dans le réseau. En outre, Le terme robuste est consédiré comme une commande auxiliaire pour garantir la robustesse dans la présence de l'erreur d'approximation et des éventuelles perturbations.

III.7. Résultats de simulations :

Dans cette section, nous présenterons nos études pour valider les résultats théoriques obtenus en considérant le problème du contrôle de mouvement dans un cadre de suivi de trajectoire de deux systèmes de robots mobiles non-holonomes, à savoir, un manipulateur mobile uni-cycle et un robot auto-équilibré à deux roues.

III.7.1. Exemple 1 :

Examinons le problème de contrôle de mouvement du robot avec des perturabations externes, La figure.3.4 montre un modèle typique d'un manipulateur de robot mobile non holonome, qui consiste en une base mobile de type uni-cycle et un manipulateur à deux degré de liberté monté sur le centre de base du robot. La base est une plateforme avec deux roues motrices montées sur le même axe ainsi qu'une roue libre avant.

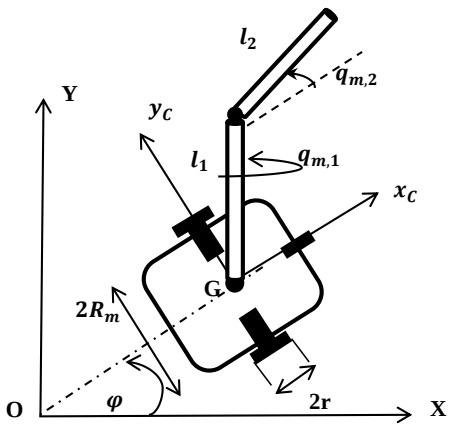


Figure.3.4. Robot manipulateur mobile uni-cycle.

Le manipulateur mobile peut être considéré comme une combinaison de deux sous-systèmes, c'est-à-dire le sous-système de la base mobile non-holonome et le sous-système de manipulation holonome. Par conséquent, les coordonnées généralisées du manipulateur mobile peuvent être séparées en deux ensembles $q = [q_b, q_m]^T$, avec $q_b = [x, y, \varphi]^T$ représente la position et l'orientation de la base mobile et $q_m = [q_{m1}, q_{m2}]^T$ représente les positions articulaires du bras manipulateur.

Les équations dynamiques du manipulateur mobile sont exprimées à l'aide du formalisme de Lagrange comme suit :

$$\begin{pmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{pmatrix}\begin{pmatrix} \ddot{q}_b \\ \ddot{q}_m \end{pmatrix} + \begin{pmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{pmatrix}\begin{pmatrix} \dot{q}_b \\ \dot{q}_m \end{pmatrix} + \begin{pmatrix} G_1 \\ G_2 \end{pmatrix} + \begin{pmatrix} \tau_{d1} \\ \tau_{d2} \end{pmatrix},$$

$$= \begin{pmatrix} J^T(q_b)\lambda \\ 0 \end{pmatrix} + \begin{pmatrix} D(q_b)\tau_b \\ \tau_m \end{pmatrix}, \qquad (3.94)$$

avec

$$M_{11} = \begin{pmatrix} m_{bm} + \frac{2}{r^2} I_w \sin^2(\varphi) & -\frac{2}{r^2} I_w \sin(\varphi)\cos(\varphi) & 0 \\ -\frac{2}{r^2} I_w \sin(\varphi)\cos(\varphi) & m_{bm} + \frac{2}{r^2} I_w \cos^2(\varphi) & 0 \\ 0 & 0 & m_a \end{pmatrix},$$

$$M_{12} = M_{21}^T = \begin{pmatrix} 0 & 0 & I_1 + I_2 \\ 0 & 0 & 0 \end{pmatrix}^T, M_{22} = \begin{pmatrix} I_1 + I_2 & 0 \\ 0 & I_2 \end{pmatrix},$$

$$C_{11} = \dot{\varphi}\begin{pmatrix} \frac{2}{r^2} I_w \sin(\varphi)\cos(\varphi) & -\frac{2}{r^2} I_w \cos(2\varphi) & 0 \\ -\frac{2}{r^2} I_w \cos(2\varphi) & -\frac{2}{r^2} I_w \sin(\varphi)\cos(\varphi) & 0 \\ 0 & 0 & 0 \end{pmatrix},$$

$$C_{12} = C_{21}^T \cong \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}^T, C_{22} \cong \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}, G_1 = [0,0,0]^T, G_2 = [0, 2m_2 g l_2 \sin(q_{m2})]^T,$$

$$D_b = \frac{1}{r}\begin{pmatrix} \cos(\varphi) & \cos(\varphi) \\ \sin(\varphi) & \sin(\varphi) \\ R_m & -R_m \end{pmatrix}, \text{ où } m_{bm} = m_b + m_{m1} + m_{m2} \text{ et } m_a = I_w + I_1 + I_2 + 2I_w \frac{R^2}{r^2}.$$

Le mouvement du robot mobile est soumis à des contraintes non holonomes données par

$$\dot{y}cos(\varphi) - \dot{x}sin(\varphi) = 0\,, \qquad (3.95)$$

$$\dot{x}cos(\varphi) + \dot{y}sin(\varphi) + R_m\dot{\varphi} - r\dot{\theta}_R = 0, \qquad (3.96)$$

$$\dot{x}cos(\varphi) + \dot{y}sin(\varphi) - R_m\dot{\varphi} - r\dot{\theta}_L = 0, \qquad (3.97)$$

où θ_R, θ_L sont les angles de rotation des roues motrices.

Afin de déterminer la matrice de transformation S(q), on choisit la variable active du système $q_a = [\theta_R, \theta_L, q_{m1}, q_{m2}]^T$ ce qui s'écrit en fonction des coordonnées généralisées par la relation suivante

$$\dot{q} = \begin{pmatrix} \hat{S}(q) & 0_{3\times2} \\ 0_{2\times2} & I_{2\times2} \end{pmatrix} \dot{q}_a = S(q)\dot{q}_a, \tag{3.98}$$

où $\hat{S}(q) = \begin{pmatrix} \frac{r}{2}\cos(\varphi) & \frac{r}{2}\cos(\varphi) \\ \frac{r}{2}\sin(\varphi) & \frac{r}{2}\sin(\varphi) \\ \frac{r}{2R_m} & -\frac{r}{2R_m} \end{pmatrix}$.

Les paramètres physique du robot sont donnés par $l_1 = 1.1\ [m]$, $l_2 = 0.8\ [m]$, $r = 0.1\ [m]$ et $R_m = 0.15\ [m]$. $m_b = 10\ [Kg]$, $m_1 = 2\ [Kg]$, $m_2 = 1\ [Kg]$, $I_w = 0.5\ [Kgm^2]$, $I_0 = 0.5\ [Kgm^2]$, $I_1 = I_2 = 0.05\ [Kgm^2]$.

Les perturbations dans les paramètres et les perturbations externes sont choisies comme suit ;

$m_b = 10 + 2\sin(\frac{2\pi}{5}t)\ [Kg], m_1 = 2 + \sin(\frac{2\pi}{5}t)[Kg], m_2 = 1 + 0.5\sin(\frac{2\pi}{5}t)[Kg]$

$I_0 = 0.5 + 0.1\sin(\frac{2\pi}{5}t)\ [Kgm^2], I_1 = I_2 = 0.05 + 0.01sin(\frac{2\pi}{5}t)\ [Kgm^2],$

$\bar{\tau}_{d1} = [0.4\exp(-1.5t + 1.5),0]^T$ et $\bar{\tau}_{d2} = [\cos(2\pi t)\ ,0]^T$ pour $0 \le t < 2$

$\bar{\tau}_{d1} = [0,\ \ 0.1\sin(t) + 0.2\exp(-t + 5)]^T$ et $\bar{\tau}_{d2} = [0, -\cos(2\pi t)]^T$ pour $2 \le t < 5$.

L'objectif de la loi de commande est d'assurer la convergence du robot vers la trajectoire de référence en présence des perturbations non négligeables. La plateforme mobile non-holonome doit suivre une trajectoire de référence linéaire spécifiée dans l'espace cartésien définie par $x_r(t) = 0.2t$ et $y_r(t) = 0.2t$ dans le plan $\{X, Y\}$.

La trajectoire de référence pour le manipulateur est décrite par un polynôme quintique avec une continuité de vitesse et d'accélération

$$q_{mr,1,2}(t) = \begin{cases} \frac{\pi}{2}\left[10\left(\frac{t}{2.5}\right)^3 - 15\left(\frac{t}{2.5}\right)^4 + 6\left(\frac{t}{2.5}\right)^5\right]; 0 \le t \le 2.5, \\ \frac{\pi}{2} - \frac{\pi}{2}\left[10\left(\frac{t-2.5}{2.5}\right)^3 - 15\left(\frac{t-2.5}{2.5}\right)^4 + 6\left(\frac{t-2.5}{2.5}\right)^5\right]; 2.5 \le t \le 5, \end{cases} \tag{3.99}$$

où la configuration initiale du manipulateur mobile est $q_1(0) = \left[0, 0, \frac{\pi}{4}, 0,0\right]^T$.

La loi de commande a été implémentée en simulation selon les deux approches de la commande robuste ; commande robuste avec un paramètre adaptatif scalaire et la commande robuste avec multi-paramètres adaptatifs. Le réseau dispose 13 neurones dans la couche cachée où $r_e = \left[e^T, r^T, \dot{q}_{a,r}^T, \varphi\right]^T \in \mathbb{R}^{13\times1}$. Les valeurs des poids du réseau et la commande

robuste sont initialisés comme zéros (10^{-3}), ainsi que le paramétrage de la loi de commande est déterminé par la méthode Essais-erreur comme suit ;

$R^{-1} = \text{diag}[0.5,0.5,0.5,0.5]$, $\Lambda = diag[2,2,50,50]$, $\sigma_{W1} = 5$, $\sigma_{W2} = 2$, $\sigma_{W3} = 2$, $\alpha = 0.2$.

$k_x = 20$, $k_y = 30$, $k_\varphi = 15$.

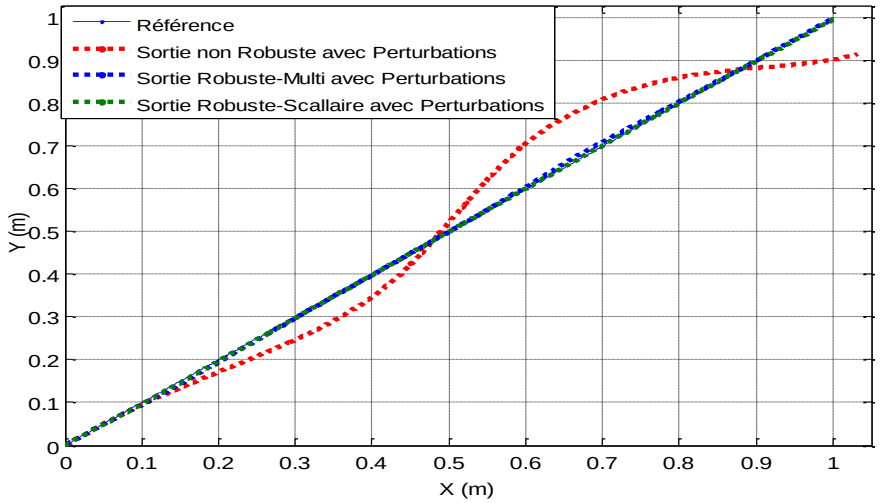
Dans un premier temps, nous appliquons la loi de commande sans le terme robuste sur le système soumis aux perturbations paramétriques et externes, et par la suite, nous ajoutons la commande robuste adaptative scalaire et la commande robuste adaptative multi-paramètres, respectivement, à la loi de commande dans les mêmes conditions pour examiner évidemment leurs comportements à compenser les perturbations dans le système.

Notons que la loi de commande proposée compense la dynamique du système en ligne en utilisant le système neuronal, avec aucune modélisation préalable n'est requise. Pour valider le fonctionnement de cette loi de commande, des simulations de suivi de trajectoire, dont les résultats sont présentés sur la figure.3.5 (a-b), ainsi que les poids du réseau de neurones et les bornes estimées de la commande robuste sont présentées sur les figures.3.5 (c-f).

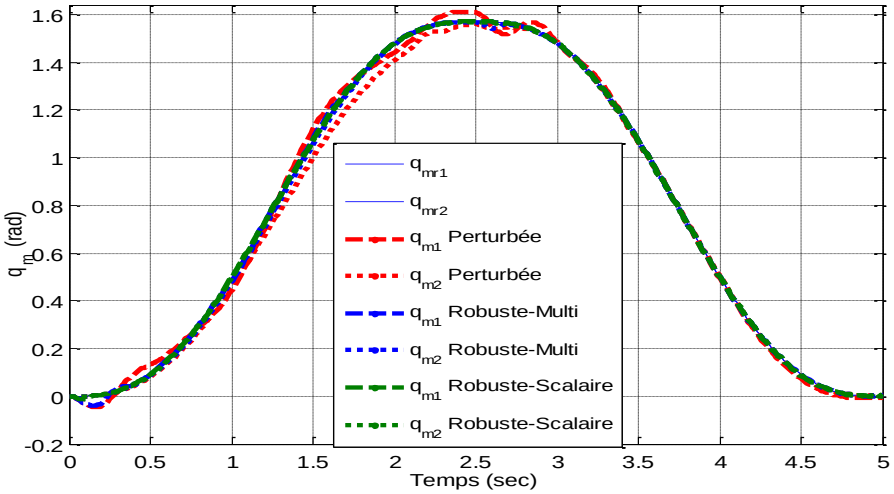
Afin d'examiner d'une part la présence et l'absence du terme robuste et d'autre part le comportement de chaque terme à traiter les perturbations dans le système, les trois courbes en pointillé, présentées sur la même figure.3.5 (a-b); tracé rouge, en utilisant la commande sans tenir compte du terme robuste, le tracé bleu, en tenant compte du terme robuste à paramètres-multiple (multi-bornes supérieures), alors le tracé vert en tenant compte le terme robuste à paramètre adaptatif scalaire.

On peut, dans un premier temps, remarquer clairement sur cette figure que la perturbation occasionnée sur le système est bien non négligeable tout en respectant les saturations des actionneurs (figure3.5.g) où la force perturbatrice est dans les limites d'énergie du système, ainsi que la perturbation impose une dégradation de l'erreur de suivi de trajectoire remarquable. Ensuite, avec l'intégration du terme robuste adaptatif multi-paramètres, afin de rattraper le retard en suivi de trajectoire, la loi robuste adapte leur paramètres (figure.3.5.e) pour générer une commande qui compense l'effet de la perturbation présentée sur chaque sous-système différemment, dans laquelle chaque paramètre converge vers une valeur différente dans l'objectif de traiter localement la perturbation et minimiser l'erreur associée à chaque sous-système.

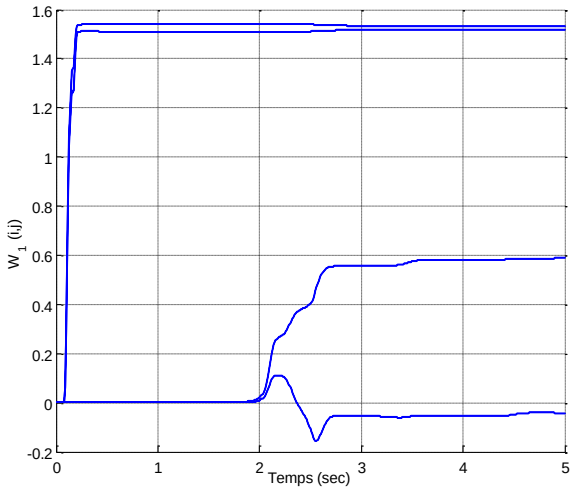
De la même façon, la commande robuste adaptative à paramètre scalaire (figure3.5.f) tente à trouver la valeur optimale globale qui compense simultanément les différentes perturbations, ainsi donc celle qui minimise toutes les erreurs de suivi de trajectoire (la somme d'erreurs) à la fois. En effet, avec la difficulté de trouver dans un temps critique durant la trajectoire la valeur de ce scalaire, la commande résultante provoque une saturation remarquable en comparaison avec la commande robuste adaptative avec multi-paramètres (figure3.5.h-i) qui traite chaque perturbation localement, ce qui résulte une commande avec un minimum d'énergie suffisante pour compenser les perturbations et minimiser ainsi toutes les erreurs du système simultanément.



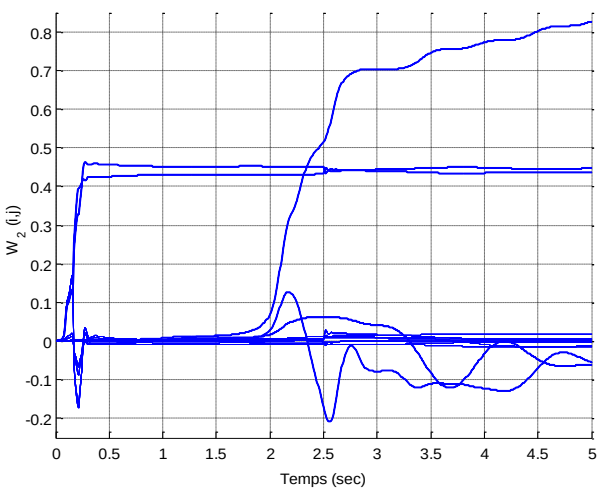
(a)



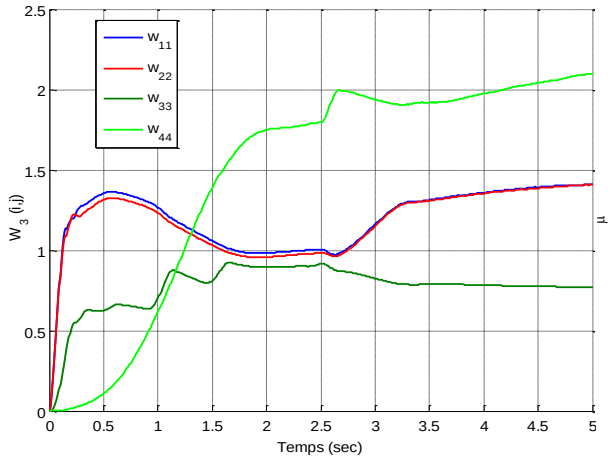
(b)


W1 (i,j)
Temps (sec)

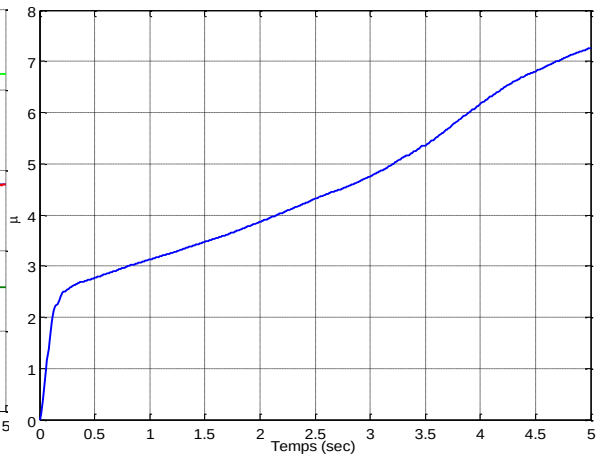
(c)


W2 (i,j)
Temps (sec)

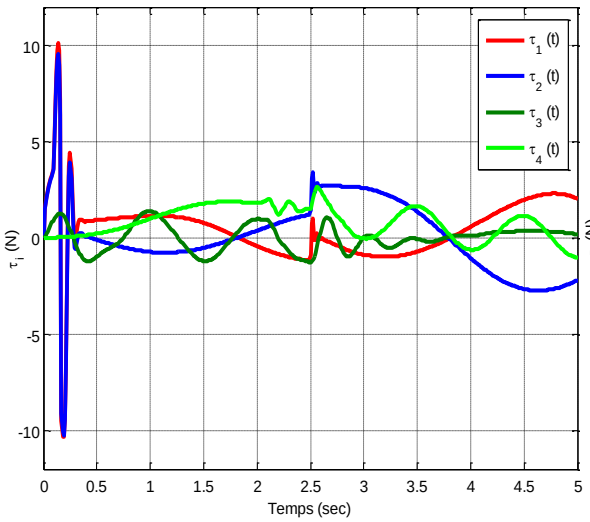
(d)


w11
w22
w33
w44
W3 (i,j)
Temps (sec)

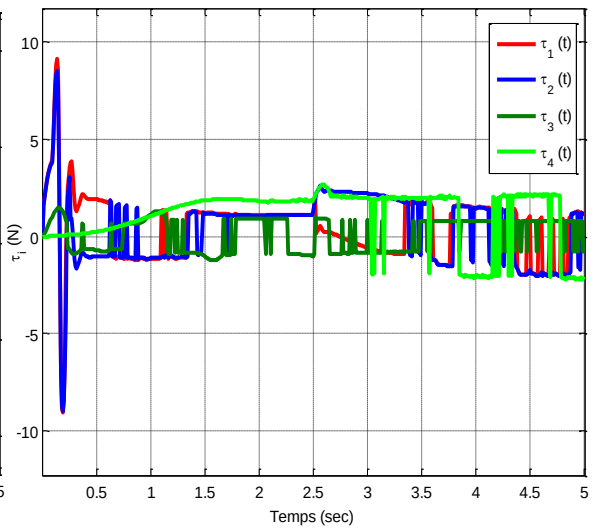
(e)


μ
Temps (sec)

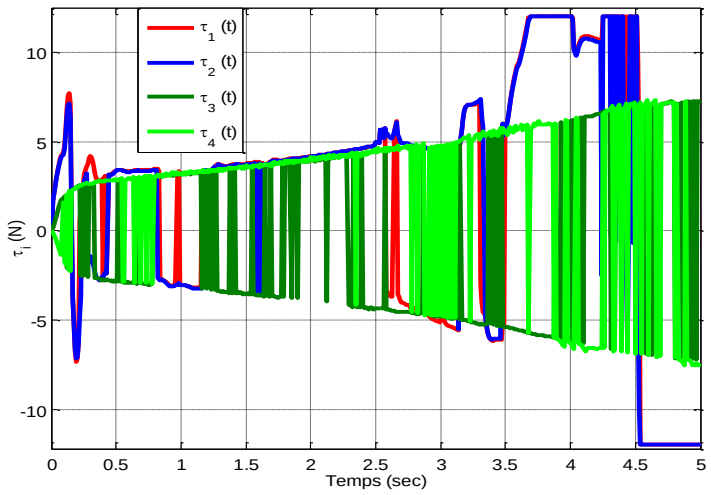
(f)


τ1 (t)
τ2 (t)
τ3 (t)
τ4 (t)
τi (N)
Temps (sec)

(g)


τ1 (t)
τ2 (t)
τ3 (t)
τ4 (t)
τi (N)
Temps (sec)

(h)



(i)

Figure.3.5 (a, b) la trajectoire de sortie et de référence pour la plateforme mobile et le bras manipulateur, respectivement. (c, d) les poids adaptatifs du réseau de neurones. (e) les paramètres adaptatifs de la commande robuste multi-paramètres. (f) le paramètre adaptatif pour la commande robuste à paramètre scalaire. (g) le signal de la loi de commande non robuste. (h) le signal de la loi de commande avec le terme robuste multi-paramètres. (i) le signal de la loi de commande avec le terme robuste à paramètre scalaire.

En comparant les résultats, la chose que l'on peut constater est que la prise en compte de terme robuste adaptative à paramètres multiples dans la loi de commande est essentielle pour assurer non seulement un bon suivi de la trajectoire mais aussi une énergie minimale notamment dans la présence d'une perturbation non négligeable. En effet, avec cette commande robuste, l'erreur de suivi de trajectoire reste dans une marge autour de zéro malgré la présence de la perturbation en comparant avec le résultat obtenu de la loi de commande sans le terme robuste. Les résultats satisfaisants pour le suivi de trajectoire obtenus, montrent clairement que la commande proposée a pu justifier sa robustesse à surmonter les incertitudes paramétriques variables dans le temps, les dynamiques inconnues et tout en tenant compte une énergie minimale.

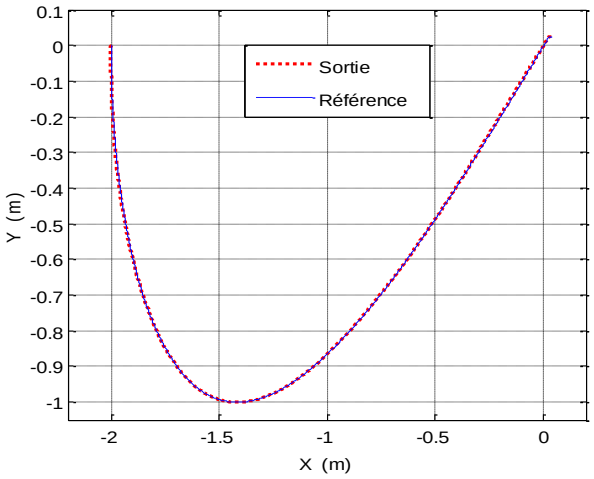
III.7.2. Exemple 2 :

Les résultats des essais présentés dans cette partie sont issus d'expérimentations réalisées avec le robot mobile uni-cycle présenté dans la chapitre 1.

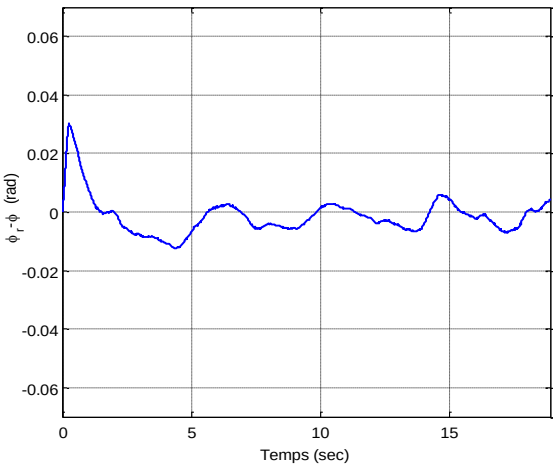
Figure.3.6 Robot mobile uni-cycle.

Dans l'objectif de valider et justifier le fonctionnement de la loi de commande proposée, nous avons implémenté et exécuté la loi de commande en temps réel sur le robot expérimental où son modèle dynamique est totalement inconnu. À cet effet, le système neuronal est appliqué afin de compenser cette dynamique inconnue conjointement avec la commande robuste, ce qui permet ainsi de compenser l'erreur d'approximation du système neuronal et d'autres perturbations imprévues durant la trajectoire et sans négliger la dynamique des actionneurs.

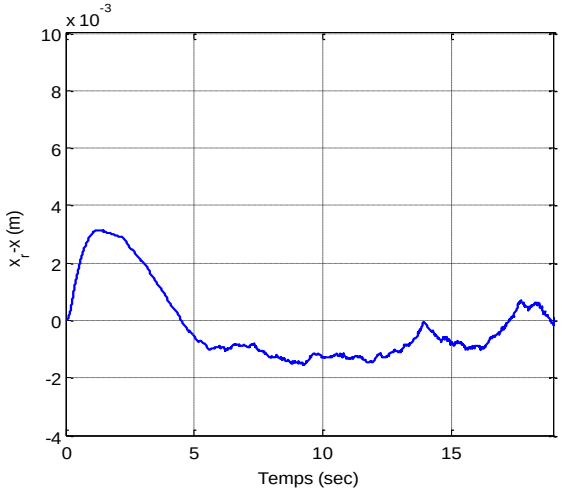
La trajectoire de référence à suivre par la plateforme mobile non-holonome est de type circulaire qui est similaire au lemniscat de Bernoulli spécifiée dans l'espace cartésien définie par ; $x_r(t) = -2cos\left(\frac{2\pi}{57.2}t\right)$ *et* $y_r(t) = -sin\left(\frac{2\pi}{37.68}t\right)$ dans le plan $\{X, Y\}$, où la configuration initiale du robot est fixée à $q_1(0) = \left[-2, 0, -\frac{\pi}{2}\right]^T$. Nous notons que tous les gains de la loi de commande sont obtenus par la méthode empirique Essais-erreur après plusieurs tentative de réglage afin d'obtenir les meilleures performances de suivi de trajectoire.
La trajectoire réellement suivie par le robot mobile lors des essais expérimentaux ainsi que les erreurs de suivi de trajectoire sont représentées sur la figure 3.7 (a-d). Les déférents poids adaptatifs du réseau de neurones et les paramètres estimés de la commande robuste sont présentés sur la figure 3.7 (e-g). Tout comme l'étude par simulation, des résultats similitudes sont obtenus. Évidemment après une période transitoire, une convergence stable autour des valeurs bornées spécifiques pour chaque sous-système garantissant un suivi de trajectoire acceptable.


Sortie
Référence
Y (m)
X (m)

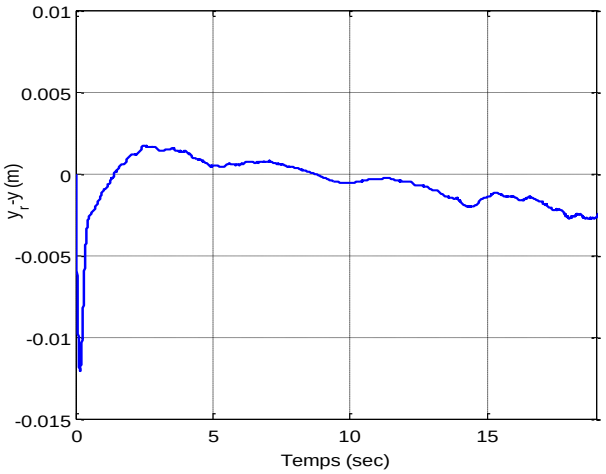
(a)


$\phi_r - \phi$ (rad)
Temps (sec)

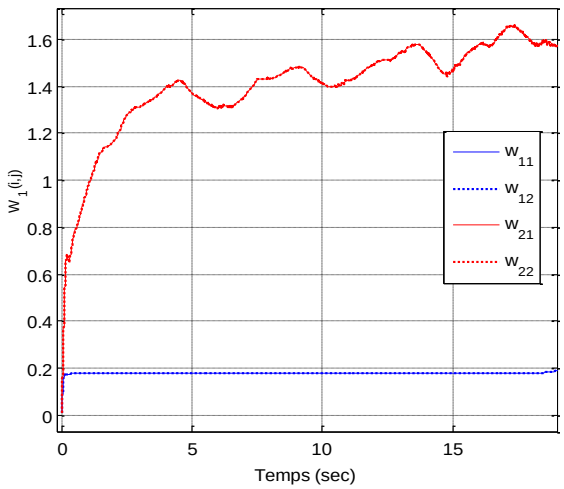
(b)


x 10^-3
x_r-x (m)
Temps (sec)

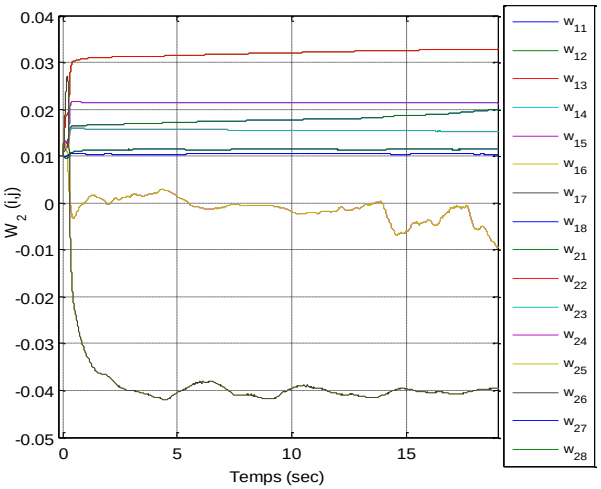
(c)


y_r-y (m)
Temps (sec)

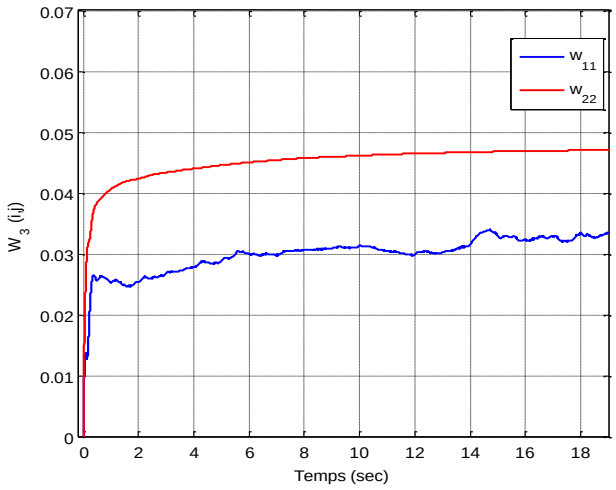
(d)


w_{11}
w_{12}
w_{21}
w_{22}
w_1(i,j)
Temps (sec)

(e)


w_2 (i,j)
Temps (sec)

(f)



(g)

Figure.3.7 (a) la trajectoire réelle et de référence dans le plan $\{X, Y\}$. (b,c,d) erreurs de suivi sur l'orientation φ et sur les positions x, y, respectivement. (e,f) les poids adaptatifs du réseau de neurones. (g) les bornes supérieures adaptatives de la commande robuste.

III.8. Loi de commande neuro robuste adaptative d'un robot manipulateur mobile avec des actionneurs électriques :

L'étude que nous considérons dans cette partie est la version étendue à la dynamique électrique pour les systèmes robotiques. En effet, dans la plupart des travaux dans la littérature traitent la commande des robots manipulateurs mobiles sous l'hypothèse que le sous-système électrique pour chaque actionneur fourni parfaitement le courant associé au couple de la commande calculée. Ainsi, les performances de la commande au niveau de la dynamique mécanique restent forcément liées, d'une part à la fidélité aux performances au niveau de la commande électrique, et d'autre part aux saturations des actionneurs du robot. Par conséquent, la loi de commande d'un robot doit traiter l'effet de couplage entre le système mécanique et le système électrique.

III.8.1. Description du système :

Généralement, les équations dynamiques d'un robot manipulateur mobile non holonome motorisé par des acionneurs électriques à courant continu sont éxprimées comme suit

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) + d_1(t) = J^T(q)\lambda + D(q)\tau(t), \quad (3.100)$$

$$L\frac{di}{dt} + f(i,\dot{q}) + d_2(t) = v(t), \quad (3.101)$$

$$\tau(t) = K_T i(t), \quad (3.102)$$

$$J(q)\dot{q} = 0, \quad (3.103)$$

où $q(t), \dot{q}(t) \in \mathbb{R}^n$ sont les coordonnées généralisées qui représentent les positions et les vitesses du robot manipulateur mobile. $M(q) \in \mathbb{R}^{n\times n}$ le moment d'inertie généralisé, $C(q,\dot{q})\dot{q} \in \mathbb{R}^n$ les forces Centrifuge et Coriolis, $G(q) \in \mathbb{R}^n$ les forces gravitationnelles, $D(q) \in \mathbb{R}^{n\times p}$ la matrice de transformation d'entrée, $\tau(t) \in \mathbb{R}^p$ les couples électromécaniques appliqués, $J(q) \in \mathbb{R}^{m\times n}$ la matrice de contraintes cinématiques, $\lambda \in \mathbb{R}^m$ est le vecteur multiplicateur de Lagrange, $L = diag[L_i] \in \mathbb{R}^{m\times m}$ est les inductances de l'induit des moteurs, $i(t) \in \mathbb{R}^p$ est les courants d'induit des moteurs à courant continu, $f(i,\dot{q}) \in \mathbb{R}^p$ repésente les forces contre-électromotrices et les résistances de l'induit de chaque moteur, $v(t) \in \mathbb{R}^p$ les tensions d'alimentation des moteurs, $K_T = diag[K_{Ti}] \in \mathbb{R}^{p\times p}$ est la matrice des constantes contre-électromotrices, $d_1(t) \in \mathbb{R}^n$ et $d_2(t) \in \mathbb{R}^p$ regroupe les dynamiques mécaniques et électriques inconnues et les perturbations externes.

En utilisant le résultat précédemment établi pour l'élimination des multiplicateurs de Lagrange et l'équation (3.86), nous pouvons obtenir la forme réduite de la dynamique mécanique:

$$\bar{M}(q)\ddot{q}_a + \bar{C}(q,\dot{q})\dot{q}_a + \bar{G}(q) = \bar{D}(q)i(t) - \bar{d}_1(t,q), \quad (3.104)$$

avec $\bar{M}(q) = S^T(q)M(q)S(q)$, $\bar{C}(q,\dot{q}) = S^T(q)\left(C(q,\dot{q})S(q) + M(q)\dot{S}(q)\right)$, $\bar{G}(q) = S^T(q)G(q)$, $\bar{d}_1(t,q) = S^T(q)d_1(t)$ et $\bar{D}(q) = S^T(q)D(q)K_T$.

À partir de cette nouvelle modélisation, les variables caractérisant l'état de notre système sont maintenant les variables actives $q_a \in \mathbb{R}^p$ et le courant d'induit $i(t) \in \mathbb{R}^p$

Afin de mettre en œuvre une loi de commande effective pour les applications pratiques, il est plus approprié de faire les hypothèses suivantes :

Hypothèse 1 : l'inductance L et les constantes contre-électromotrices K_T sont exprimée respectivement comme $L = L_0 + \Delta L$ et $K_T = K_0 + \Delta K$ où L_0 , K_0 sont les valeurs nominales supposées connues, ΔL, ΔK repésentent des petites perturabtions autour des valeurs nominales L_0 et K_0, respectivement.

Hypothèse 2 : Les matrices $\bar{M}(q)$, $\bar{C}(q,\dot{q})$, $\bar{G}(q)$ et $f(i,\dot{q})$ sont supposées complètement inconnues, et la matrice de transformation cinématique est exprimée par $\bar{D}(q) = \bar{D}_0(q) + \Delta D$ où ΔD repésente les incertitudes sur les paramètres cinématique de la plateforme mobile non holonome.

Pour une trajectoire désirée pour le robot manipulateur mobile non holonome $q_{a,d}(t) \in \mathbb{R}^m$, l'erreur de position est définie par $e_1(t) = q_{a,d}(t) - q_a(t)$ et l'erreur filtrée associée est définie par $e_2(t) = \dot{e}(t) + \Lambda e(t)$ où $\Lambda = \Lambda^{\mathrm{T}} \in \mathbb{R}^{p\times p}$ est une matrice définie positive. Ainsi, les équations dynamiques mécaniques fonction de l'erreur $e_1(t)$ et $e_2(t)$ s'écritent

$$\bar{M}(q)\dot{e}_2(t) = -\bar{C}(q,\dot{q})e_2(t) - \bar{D}(q)i(t) + H_1(q_e) + \bar{d}_1(t,q), \qquad (3.105)$$

avec la fonction non linéaire $H_1(q_e) = \bar{M}(q)\left(\ddot{q}_{a,d} + \Lambda\dot{e}_1\right) + \bar{C}(q,\dot{q})\left(\dot{q}_{a,d} + \Lambda e_1\right) + \bar{G}(q)$ regroupe toute la dynamique inconnue du système et $q_e = \left[q_a^T, \dot{q}_a^T, q_{a,d}^T, \dot{q}_{a,d}^T, \ddot{q}_{a,d}^T\right]^T$.

D'autre part, on définit l'erreur $e_3(t) = i_d(t) - i(t)$ où $i_d(t)$ est le courant désiré associé à la trajectoire désirée du robot manipulateur mobile. D'après l'équation eléctrique (3.101), la dérivée de $e_3(t)$ peut être calculeé comme suit

$$L\dot{e}_3(t) = L\frac{di_d}{dt} + f(i,\dot{q}) - v(t) + d_2(t),$$

$$= L\frac{di_d}{dt} + H_2(q_f) - v(t) + d_2(t), \qquad (3.106)$$

avec la fonction $H_2(q_f)$ regroupe tout la dynamique électrique inconnue et $q_f = [i^T, \dot{q}^T]^T$.

III.8.2. Conception de la loi de commande :

Dans cette section, nous examinerons le problème de contrôle d'un robot manipulateur mobile à commande électrique décrit par les équations (3.100-3.103) sous les hypothèses imposées sur les modèles mécanique et électrique dans lesquelle les differentes perturbations affectant le système sont supposées bornées. Notre approche est basée sur le système neuronal pour approximer les fonctions incconnues $H_1(q_e)$ et $H_2(q_f)$, et par la suite nous développons une

nouvelle loi de commande qui garantie la robustesse face aux erreurs d'approximation et éventuelles perturabtions imprévues. Le schéma fonctionnel de la loi de commande proposée est représenté sur la figure.3.8.

Nous consédirons le modèle neuronal (3.55) pour approximer les fonctions $H_1(q_e)$ et $H_2(q_f)$ comme suit

$$H_1(q_e) = W_1^T \psi_e + \varepsilon_1(q_e), \tag{3.107}$$

$$H_2(q_f) = W_3^T \psi_f + \varepsilon_2(q_f), \tag{3.108}$$

avec $\psi_e = \psi_e(W_2^T q_e)$ et $\psi_f = \psi_f(W_3^T q_f)$.

En remplaçant l'equation (3.107) dans (3.105) et l'équation (3.108) dans (3.106), on obtient

$$\bar{M}(q)\dot{e}_2(t) = -\bar{C}(q,\dot{q})e_2(t) - \bar{D}(q)i(t) + W_1^T\psi_e + \varepsilon_1 + \bar{d}_1, \tag{3.109}$$

$$L\dot{e}_3(t) = L\frac{di_d}{dt} + W_3^T\psi_f + \varepsilon_2 - v(t) + d_2. \tag{3.110}$$

On définie $\widehat{H}_1(q_e) = \widehat{W}_1^T\hat{\psi}_e$ et $\widehat{H}_2(q_f) = \widehat{W}_3^T\hat{\psi}_f$ comme les estimations des fonction $H_1(q_e)$ et $H_2(q_f)$, respectivement. Ainsi, en basant sur le système neuronal, on définit le courant désiré et la tension de commande comme suit

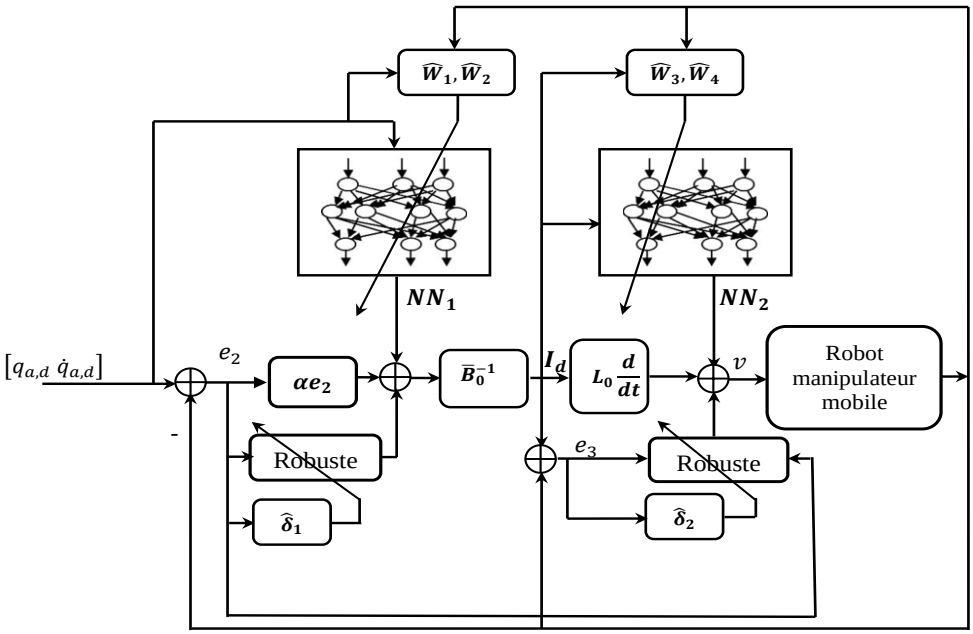


Figure.3.8. Bloc diagramme de la loi de commande proposée.

$$\bar{D}_0(q)i_d(t) = \widehat{W}_1^T\hat{\psi}_e + \hat{\delta}_1^T\frac{e_2}{\|e_2\|} + \alpha e_2, \tag{3.111}$$

$$v(t) = \widehat{W}_3^T\hat{\psi}_f + \hat{\delta}_2^T\frac{e_3}{\|e_3\|} + \beta e_3 + \bar{D}_0^T(q)e_2 + L_0\frac{di_d}{dt}, \tag{3.112}$$

avec $\alpha = \alpha^T > 0,\ \beta = \beta^T > 0,\ \hat{\delta}_1^T = \text{diag}[\hat{\delta}_{1,i}] \in \mathbb{R}^{p\times p}$ et $\hat{\delta}_2^T = \text{diag}[\hat{\delta}_{2,i}] \in \mathbb{R}^{p\times p}$. Les matrices $\hat{\delta}_1^T$ et $\hat{\delta}_2^T$ seront conçues afin de surmonter l'effet de la perturbation et garantir la robustesse. En utilisant les lois de commande (3.111) et (3.112), le système en boucle fermée s'écrit

$$\begin{aligned}\bar{M}(q)\dot{e}_2(t) &= -\bar{C}(q,\dot{q})e_2(t) - (\bar{D}_0 + \Delta D)(i_d - e_3) + W_1^T\psi_e + \varepsilon_1 + \bar{d}_1,\\ &= -\bar{C}(q,\dot{q})e_2(t) + \bar{D}_0e_3 - \bar{D}_0i_d + W_1^T\psi_e - \Delta Di(t) + \varepsilon_1 + \bar{d}_1,\\ &= -\left[\bar{C}(q,\dot{q}) + \frac{\hat{\delta}_1^T}{\|e_2\|} + \alpha\right]e_2(t) + \bar{D}_0e_3 - \widehat{W}_1^T\hat{\psi}_e + W_1^T\psi_e - \Delta Di(t) + \varepsilon_1 + \bar{d}_1,\end{aligned} \tag{3.113}$$

et

$$\begin{aligned}L\dot{e}_3(t) &= (L_0 + \Delta L)\frac{di_d}{dt} + W_3^T\psi_f - (\widehat{W}_3^T\hat{\psi}_f + \hat{\delta}_2^T\frac{e_3}{\|e_3\|} + \beta e_3 + \bar{D}_0^T(q)e_2 + L_0\frac{di_d}{dt}) + d_2 + \varepsilon_2,\\ &= -\bar{D}_0^T(q)e_2 - \left(\beta + \hat{\delta}_2^T\frac{1}{\|e_3\|}\right)e_3 + W_3^T\psi_f - \widehat{W}_3^T\hat{\psi}_f + \Delta L\frac{di_d}{dt} + d_2 + \varepsilon_2.\end{aligned} \tag{3.114}$$

D'après (3.66), on a

$$H(q_x) - \hat{H}(q_x) = \widetilde{W}_i^T\left[\hat{\psi}_x - \hat{\psi}'_x\,\widehat{W}_{i+1}^Tq_x\right] + \widehat{W}_i^T\hat{\psi}'_x\widetilde{W}_{i+1}^Tq_x + \omega_x, \tag{3.115}$$

avec $\omega_x = \widetilde{W}_i^T\hat{\psi}'_x\,W_{i+1}^Tq_x + W_i^TO^2 + \varepsilon(q_x)$.

En remplaçant l'équation (3.115) dans (3.113) et (3.114) avec $q_x = q_e, q_f$, on obtient

$$\bar{M}(q)\dot{e}_2 = -\left[\bar{C}(q,\dot{q}) + \frac{\hat{\delta}_1^T}{\|e_2\|} + \alpha\right]e_2 + \bar{D}_0e_3 + \widetilde{W}_1^T\left[\hat{\psi}_e - \hat{\psi}'_e\,\widehat{W}_2^Tq_e\right] + \widehat{W}_1^T\hat{\psi}'_e\widetilde{W}_2^Tq_e + \Psi_e, \tag{3.116}$$

et

$$L\dot{e}_3 = -\bar{D}_0^T(q)e_2 - \left(\beta + \hat{\delta}_2^T\frac{1}{\|e_3\|}\right)e_3 + \widetilde{W}_3^T\left[\hat{\psi}_f - \hat{\psi}'_f\,\widehat{W}_4^Tq_f\right] + \widehat{W}_3^T\hat{\psi}'_f\widetilde{W}_4^Tq_f + \Psi_f, \tag{3.117}$$

avec $\Psi_e = \omega_e - \Delta D i(t) + \varepsilon_1 + \bar{d}_1$ et $\Psi_f = \omega_f + \Delta L \frac{di_d}{dt} + d_2 + \varepsilon_2$ sont les perturbations globales.

Hypothèse 3: chaque élément dans les vecteurs des perturbations Ψ_e et Ψ_f est supposé borné par une constante positive inconnue appelée la borne supérieure locale inconnue ; $\Psi_{e,i} \leq \omega_{e,i}$ et $\Psi_{f,i} \leq \omega_{f,i}$ respectivement, avec la norme du vecteur Ψ_e et Ψ_f est bornée par le minimum de $a_{e,i}\omega_{e,i}$ et $a_{e,i}\omega_{f,i}$, respectivement, qui représentent les bornes supérieures globales inconnues ; $\|\Psi_e\| \leq \lambda_{min}(\delta_1)$ et $\|\Psi_f\| \leq \lambda_{min}(\delta_2)$, où la matrice $\delta_1 = \text{diag}[a_{e,i}\omega_{e,i}] \in \mathbb{R}^{p\times p}$ et $\delta_2 = \text{diag}[a_{f,i}\omega_{f,i}] \in \mathbb{R}^{p\times p}$ avec $a_{e,i}$ et $a_{f,i}$ certaines constantes positives inconnues

Théorème : *Considérons le système dynamique du robot représenté par (3.100-3.103). Si les lois de commande sont conçues comme (3.111-3.112), dans lesquelles les lois d'adaptation sont conçues comme (3.118-3.123), donc la bornitude (UUB) des erreurs en question peut être garantie. De plus, l'erreur de suivi de trajectoire peut être suffisamment petite en ajustant les paramètres α, β, γ_1 et γ_2*

$$\dot{\widehat{W}}_1(t) = \eta_{W_1}\left(\hat{\psi}_e - \hat{\psi}'_e \widehat{W}_2^T q_e\right) e_2^T, \tag{3.118}$$

$$\dot{\widehat{W}}_2(t) = \eta_{W_2} q_e e_2^T \widehat{W}_1^T \hat{\psi}'_e, \tag{3.119}$$

$$\dot{\widehat{W}}_3(t) = \eta_{W_3}\left(\hat{\psi}_f - \hat{\psi}'_f \widehat{W}_4^T q_f\right) e_3^T, \tag{3.120}$$

$$\dot{\widehat{W}}_4(t) = \eta_{W_4} q_f e_3^T \widehat{W}_3^T \hat{\psi}'_f, \tag{3.121}$$

$$\dot{\hat{\delta}}_1(t) = \eta_{\delta_1}\left(\text{diag}[e_{2.i}^2]\frac{1}{\|e_2\|} - \gamma_1 \|e_2\| \hat{\delta}_1(t)\right), \tag{3.122}$$

$$\dot{\hat{\delta}}_2(t) = \eta_{\delta_2}\left(\text{diag}[e_{3.i}^2]\frac{1}{\|e_3\|} - \gamma_2 \|e_3\| \hat{\delta}_2(t)\right), \tag{3.123}$$

avec η_{W_1}, η_{W_2}, η_{W_3}, η_{W_4}, η_{δ_1}et η_{δ_2} sont des constantes positives et $e_{2,i}$, $e_{3,i}$ sont les élements des vecteurs e_2 et e_3, respectivement.

Preuve : Nous consédirons la fonction candidate de Lyapunov suivante

$$V\left(e_i, \widetilde{W}_i, \tilde{\delta}_i, t\right) = \frac{1}{2} e_2^T \bar{M}(q) e_2 + \frac{1}{2} e_3^T L e_3 + \sum_{i=1}^{4} \frac{1}{2\eta_{W_i}} tr(\widetilde{W}_i^T \widetilde{W}_i) + \sum_{i=1}^{2} \frac{1}{2\eta_{\delta_i}} tr(\tilde{\delta}_i^T \tilde{\delta}_i) ,$$

(3.124)

où $\widetilde{W}_i = W_i - \widehat{W}_i$ et $\tilde{\delta}_i = \delta_i - \hat{\delta}_i$ sont les erreurs d'estimation.

Nous calculons la dérivée de Lyapunov, on trouve

$$\dot{V} = e_2^T \bar{M}(q) \dot{e}_2 + \frac{1}{2} e_2^T \dot{\bar{M}}(q) e_2 + e_3^T L \dot{e}_3 + \sum_{i=1}^{4} \frac{1}{\eta_{W_i}} tr(\widetilde{W}_i^T \dot{\widetilde{W}}_i) + \sum_{i=1}^{2} \frac{1}{\eta_{\delta_i}} tr(\tilde{\delta}_i^T \dot{\tilde{\delta}}_i) . \quad (3.125)$$

En remplaçant les équations (3.116,3.117) dans (3.125), on aura

$$\dot{V} = e_2^T \left(- \left[\bar{C}(q, \dot{q}) + \frac{\hat{\delta}_1^T}{\|e_2\|} + \alpha \right] e_2 + \bar{D}_0 e_3 + \widetilde{W}_1^T \left[\hat{\psi}_e - \hat{\psi}'_e \widehat{W}_2^T q_e \right] + \widehat{W}_1^T \hat{\psi}'_e \widetilde{W}_2^T q_e + \Psi_e \right)$$

$$+ e_3^T \left(- \bar{D}_0^T(q) e_2 - \left[\beta + \hat{\delta}_2^T \frac{1}{\|e_3\|} \right] e_3 + \widetilde{W}_3^T \left[\hat{\psi}_f - \hat{\psi}'_f \widehat{W}_4^T q_f \right] + \widehat{W}_3^T \hat{\psi}'_f \widetilde{W}_4^T q_f + \Psi_f \right)$$

$$+ \frac{1}{2} e_2^T \dot{\bar{M}}(q) e_2 + \sum_{i=1}^{4} \frac{1}{\eta_{W_i}} tr(\widetilde{W}_i^T \dot{\widetilde{W}}_i) + \sum_{i=1}^{2} \frac{1}{\eta_{\delta_i}} tr(\tilde{\delta}_i^T \dot{\tilde{\delta}}_i). \quad (3.126)$$

On prend en compte la propriété P2, le fait que $x^T y = tr(yx^T)$ et $e^T \tilde{\delta}^T e = tr\left(\tilde{\delta}^T e e^T\right) = tr(\tilde{\delta}^T \text{diag}[e_i^2])$ et en remplaçant les équations (3.118-123), on aura

$$\dot{V} = -e_2^T \alpha e_2 - e_3^T \beta e_3 + e_2^T \left(\Psi_e - \frac{\delta_1^T}{\|e_2\|} e_2 \right) + + e_3^T \left(\Psi_f - \frac{\delta_2^T}{\|e_3\|} e_3 \right),$$

$$+ \gamma_1 \|e_2\| tr\left(\tilde{\delta}_1^T \hat{\delta}_1\right) + \gamma_2 \|e_3\| tr\left(\tilde{\delta}_2^T \hat{\delta}_2\right). \quad (3.127)$$

En utilisant l'inégalité de Cauchy-Schwarz et l'hypothèse 3, on aura

$$e^T \left(\Psi - \frac{\delta^T}{\|e\|} e \right) \leq \|e\| \left(\|\Psi\| - \lambda_{min}(\delta^T) \right) \leq 0. \quad (3.128)$$

D'une part, on a

$$tr\left(\tilde{\delta}_i^T \hat{\delta}_i\right) = tr\left(\tilde{\delta}_i^T \delta_i\right) - \left\|\tilde{\delta}_i^T\right\|_F^2 \leq \left\|\tilde{\delta}_i^T\right\|_F \left(\delta_{sup,i} - \left\|\tilde{\delta}_i^T\right\|_F \right), \quad (3.129)$$

avec $\|.\|_F$ est la norme Frobenius et $\|\delta_i\|_F \leq \delta_{sup,i} \ \ (i = 1,2)$.

Ainsi, la dérivée de la fonction de Lyapunov satisfait

$$\dot{V} \leq -\lambda_{min}\{\alpha\} \|e_2\|^2 - \lambda_{min}\{\beta\} \|e_3\|^2 + \gamma_1 \|e_2\| \left\|\tilde{\delta}_1^T\right\|_F \left(\delta_{sup,1} - \left\|\tilde{\delta}_1^T\right\|_F \right),$$

$$+\gamma_2\|e_3\|\left\|\tilde{\delta}_2^T\right\|_F\left(\delta_{sup,2}-\left\|\tilde{\delta}_2^T\right\|_F\right),$$

$$\leq -\|e_2\|\left[\lambda_{min}\{\alpha\}\|e_2\|+\gamma_1\left\|\tilde{\delta}_1^T\right\|_F\left(\left\|\tilde{\delta}_1^T\right\|_F-\delta_{sup,1}\right)\right],$$

$$-\|e_3\|\left[\lambda_{min}\{\beta\}\|e_3\|+\gamma_2\left\|\tilde{\delta}_2^T\right\|_F\left(\left\|\tilde{\delta}_2^T\right\|_F-\delta_{sup,2}\right)\right],$$

$$\leq -\|e_2\|\left[\lambda_{min}\{\alpha\}\|e_2\|+\gamma_1\left(\left\|\tilde{\delta}_1^T\right\|_F-\frac{\delta_{sup,1}}{2}\right)^2-\gamma_1\frac{\delta_{sup,1}^2}{4}\right],$$

$$-\|e_3\|\left[\lambda_{min}\{\beta\}\|e_3\|+\gamma_2\left(\left\|\tilde{\delta}_2^T\right\|_F-\frac{\delta_{sup,2}}{2}\right)^2-\gamma_2\frac{\delta_{sup,2}^2}{4}\right], \quad (3.130)$$

Donc, la condition suffisante suivante garantit que la dérivée de Lyapunov soit négative

$$\|e_2\|\geq\gamma_1\frac{\delta_{sup,1}^2}{4\,\lambda_{min}\{\alpha\}} \quad \text{ou} \quad \left\|\tilde{\delta}_1^T\right\|_F\geq\delta_{sup,1}, \quad (3.131)$$

et
$$\|e_3\|\geq\gamma_2\frac{\delta_{sup,2}^2}{4\,\lambda_{min}\{\beta\}} \quad \text{ou} \quad \left\|\tilde{\delta}_2^T\right\|_F\geq\delta_{sup,2}. \quad (3.132)$$

D'après la théorie standard de Lyapunov **[88]**, l'erreur $\|e_2\|$, $\|e_3\|$ et le signal de commande sont uniformement bornées (UUB), ainsi, l'erreur de suivi de trajectoire converge toujours vers une région peut être minimisée avec le réglage des paramètres α, β, γ_1 et γ_2.

Remarque : Les termes robustes dans la loi de commande peut présenter un problème quand la norme $\|e_i\|$ $(i=2,3)$ s'approche de zéro. à cet effet, nous proposons la modification suivante

$$\bar{D}_0(q)i_d(t)=\widehat{W}_1^T\hat{\psi}_e+\hat{\delta}_1^T\frac{e_2}{\|e_2\|+\varepsilon_2}+\alpha e_2, \quad (3.133)$$

$$v(t)=\widehat{W}_3^T\hat{\psi}_f+\hat{\delta}_2^T\frac{e_3}{\|e_3\|+\varepsilon_3}+\beta e_3+\bar{D}_0^T(q)e_2+L_0\frac{di_d}{dt}, \quad (3.134)$$

$$Si\ \|e_2\|>\varepsilon_2 \quad \dot{\hat{\delta}}_1(t)=\eta_{\delta_1}\left(\text{diag}[e_{2.i}^2]\frac{1}{\|e_2\|}-\gamma_1\|e_2\|\hat{\delta}_1(t)\right),sinon\ \dot{\hat{\delta}}_1(t)=0, \quad (3.135)$$

$$Si\ \|e_3\|>\varepsilon_3 \quad \dot{\hat{\delta}}_2(t)=\eta_{\delta_2}\left(\text{diag}[e_{3.i}^2]\frac{1}{\|e_3\|}-\gamma_2\|e_3\|\hat{\delta}_2(t)\right)sinon\ \dot{\hat{\delta}}_2(t)=0, \quad (3.136)$$

avec ε_2 et ε_3 constantes positives représentent chaqu'un la région d'erreur qui est consédirée acceptable associée à $\|e_2\|$ et $\|e_3\|$, respectivement.

III.9. Simulations :

Afin de monter le fonctionnement de la loi de commande proposée, nous considérons dans cette partie le problème de contôle de mouvement dans un cadre de suivi de trajectoire. Par soucis de simplicité, le robot exemple retenu pour la simulation est d'un robot mobile non-holonome de type uni-cycle. Le principe de fonctionnement de la loi de commande se généralise facilement en fait à d'autres robots commandés par des actionneurs électriques. Ce robot mobile typique est une plateforme avec deux roues motrices à courant continu montées sur le même axe ainsi qu'une roue libre avant.

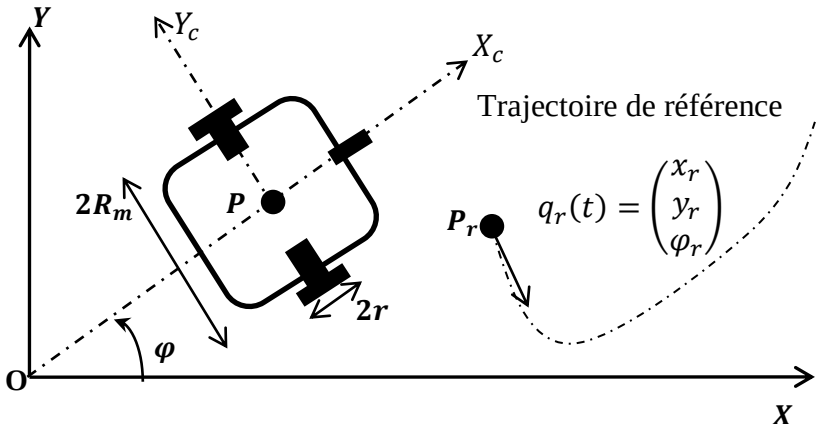


Figure.3.9. Robot mobile uni-cycle.

Les équations dynamiques de ce système sont exprimées à l'aide du formalisme de Lagrange comme suit :

$$m\ddot{x} = \frac{1}{r}(\tau_R + \tau_L)\cos(\varphi) - \lambda \sin(\varphi) + d_1, \quad (3.137)$$

$$m\ddot{y} = \frac{1}{r}(\tau_R + \tau_L)\sin(\varphi) + \lambda \cos(\varphi) + d_2, \quad (3.138)$$

$$I_w\ddot{\varphi} = \frac{R_m}{r}(\tau_R - \tau_L) + d_3, \quad (3.139)$$

$$v = L\frac{di}{dt} + RI + K\dot{q}_a + d_4, \quad (3.140)$$

$$\tau = [\tau_R, \tau_L]^T = N\tau_{moteurs}, \quad (3.141)$$

où y, x sont les coordonnées cartésiennes sur le plan {X, Y} et φ est l'orientation du robot mobile, m est la masse totale de la plateforme, I_w est son moment d'inertie, $2R_m$ est la distance entre les roues motrices, r est le diamètre de la roue motrice supposé identique pour les deux roues, τ_R, τ_L sont des entrées de couple agissant sur les roues motrices droite et

gauche, respectivement, λ est le multiplicateur inconnu de Lagrange et τ_{di} $(i = 1,2,3)$ sont les perturabtions externes, $K \in \mathbb{R}^{2\times2}$ sont les constantes c-f-e-m, $I \in \mathbb{R}^2$ est le courant d'induit, la résistance $R \in \mathbb{R}^{2\times2}$, $K\dot{q}_a \in \mathbb{R}^2$ est la force contre électromotrice des moteurs, $v \in \mathbb{R}^2$ est la tension de commande des moteurs et $N \in \mathbb{R}^{2\times2}$ est le rapport de réduction des motoréducteurs.

Le mouvement du robot mobile est soumis à des contraintes non holonomes données par

$$\dot{y}cos(\varphi) - \dot{x}sin(\varphi) = 0, \tag{3.142}$$

$$\dot{x}cos(\varphi) + \dot{y}sin(\varphi) + R_m\dot{\varphi} - r\dot{\theta}_R = 0, \tag{3.143}$$

$$\dot{x}cos(\varphi) + \dot{y}sin(\varphi) - R_m\dot{\varphi} - r\dot{\theta}_L = 0. \tag{3.144}$$

Les paramètres physique du robot sont donnés par $m = 10 + 2\sin(\frac{2\pi}{5}t)$ $[Kg]$, $I_w = 0.5 + 0.1\cos(\frac{2\pi}{5}t)$ $[Kgm^2]$, Le rapport de réduction $N = \text{diag}[50,50]$, $r = 0.12$ $[m]$ et $R_m = 0.15$ $[m]$, $R = \text{diag}[5,5]$ $[\Omega]$,

$K = \text{diag}[0.1,0.1] + \text{diag}\left[0.01\sin\left(\frac{2\pi}{5}t\right), 0.01\cos\left(\frac{2\pi}{5}t\right)\right]$ $[Nm/A]$,

et $L = \text{diag}[0.02,0.02] + \text{diag}\left[0.001\sin\left(\frac{2\pi}{5}t\right), 0.001\cos\left(\frac{2\pi}{5}t\right)\right]$ $[H]$.

La robustesse de la loi de commande proposée face aux perturbations externes a été simulée avec les perturabtions suivantes ; $d_1 = 0.08\exp(-1.5t + 1.5)$, $d_2 = -5d_1, d_3 = 2d_1$ pour $0 \le t < 3$ et $d_1 = 0.1\exp(-t + 5) + 0.5\sin(t)$, $d_2 = -5d_1, d_3 = 2d_1$ pour $t \ge 3$, $d_4 = \left[0.2\sin\left(\frac{2\pi}{5}t\right), -0.2\,cos\left(\frac{2\pi}{5}t\right)\right]$.

Dans la présence des perturbations externes non négligeables, la loi de commande doit assurer la convergence du robot vers la trajectoire de référence qui est définie dans le plan cartésien $\{X, Y\}$ dans un premier exemple par $q_r(t) = \left[0.2t, 0.2t, \frac{\pi}{4}\right]^T$ avec $q_r(0) = \left[0, 0, \frac{\pi}{4}\right]^T$ et dans le deuxième exemple par $x_r = -2\cos\left(\frac{2\pi}{75.2}t\right)$ *et* $y_r(t) = -\sin\left(\frac{2\pi}{37.68}t\right)$ dans le plan $\{X, Y\}$, où la position initiale du robot est fixée à $q_1(0) = \left[-2, 0, -\frac{\pi}{2}\right]^T$.

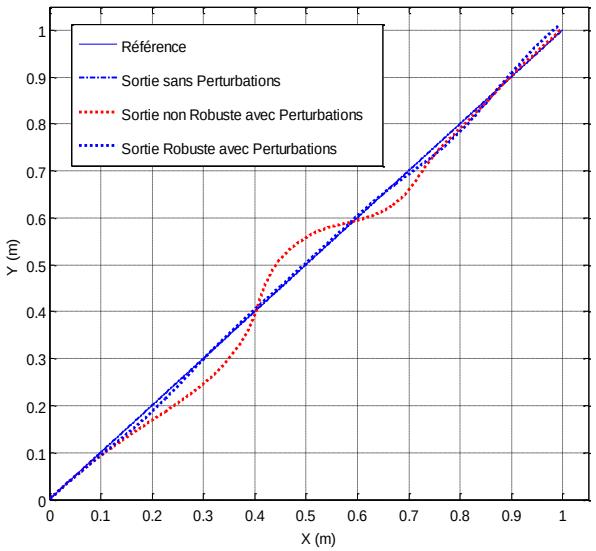
La loi de commande a été implémentée avec un réseau à 7 neurones dans la couche cachée où $q_e = [e_1^T, e_2^T, \dot{q}_a^T, \varphi]^T \in \mathbb{R}^{7\times1}$ pour le premier système neuronal, et 4 neurones pour le deuxième système neuronal avec $q_f = [I_m^T, \dot{q}_m^T]^T \in \mathbb{R}^{4\times1}$. Les poids du réseaux sont

initialisés comme $W_i(0) = 0.01$ $(i = 1,2,3,4)$, ainsi que les paramètres adaptatifs de la commande robuste sont initialisés par zéros. Des performances convergentes de la loi de commande sont atteintes avec un paramétrage qui a été fixé après plusieurs tentatives par la méthode Essais-erreur comme suit ; $\alpha = \text{diag}[2,2]$, $\Lambda = \text{diag}[0.2,0.2]$, $\beta = \text{diag}[40,40]$, $\eta_{W1} = 1$, $\eta_{W2} = 0.5$, $\eta_{W3} = 20$, $\eta_{W4} = 1$, $\eta_{\delta 1} = 5$, $\eta_{\delta 2} = 50$, $\gamma_1 = 0.05$ et $\gamma_2 = 0.5$. $k_x = 20$, $k_y = 30$, $k_\varphi = 15$.

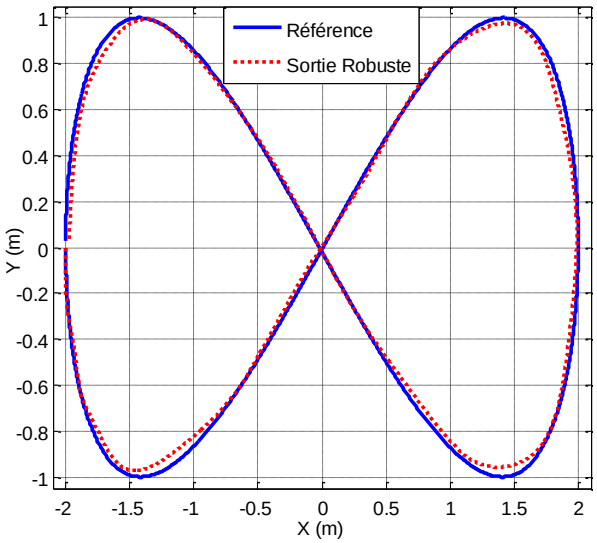
La figure.3.10 (a) montre trois cas de sorties obtenues de suivi de trajectoire par rapport à la trajectoire de référence, et ainsi que les signaux de commande appliqués pour chaque cas (figure.3.10 (c-e)), en prenant en compte les saturations sur les actionneurs qui ont été fixées par $sat = 24$ $[V]$.

Sur la figure.3.10 (a), on peut remarquer l'influence des perturbations sur le système dans l'absence de terme robuste, tout en comparant avec la sortie obtenue dans le cas nominal sans perturbations. Malgré que la force perturbatrice reste accessible et ne sorte pas des limites d'énergie du système (figure3.10 (d)), le système neuronal adaptatif seul n'a pas pu compenser l'absence de la commande robuste dans la présence des perturbations occasionnées (figure.3.10 (f-i)) en raison que le réseau de neurones prend de temps pour s'adapter avec le changement dans le système provoqué par les perturbations.

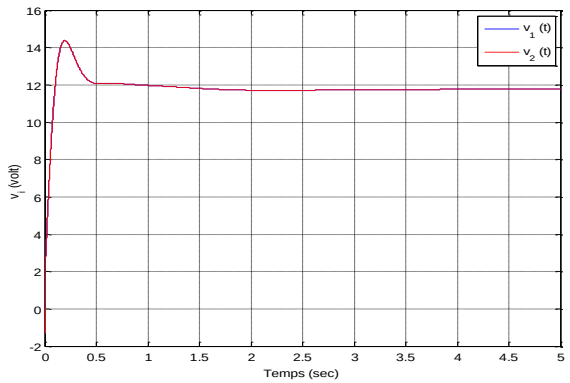
Par contre, dans la présence du terme robuste (sortie robuste avec perturbation sur la figure.3.10 (a)), afin de rattraper rapidement le retard en suivi de trajectoire, les différents paramètres associés à la commande robuste adaptative convergent vers des valeurs assurant la compensation locale des différentes perturbations présentées sur les sous-systèmes (figure.3.10 (j)). Le terme robuste joue le rôle d'une commande auxiliaire qui traite et compense à la fois les perturbations externes et l'erreur d'approximation du système neuronal face à une dynamique mécanique et électrique inconnues. La loi d'adaptation de la commande robuste serte à ajuster les paramètres participant à la minimisation de l'erreur de suivi de la trajectoire de façon en ligne ce qui garantir un résultat plus amélioré avec une énergie minimale possible.


Référence
Sortie sans Perturbations
Sortie non Robuste avec Perturbations
Sortie Robuste avec Perturbations
Y (m)
X (m)

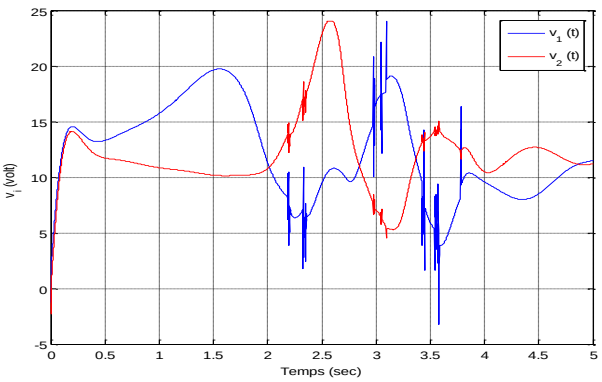
(a)


Référence
Sortie Robuste
Y (m)
X (m)

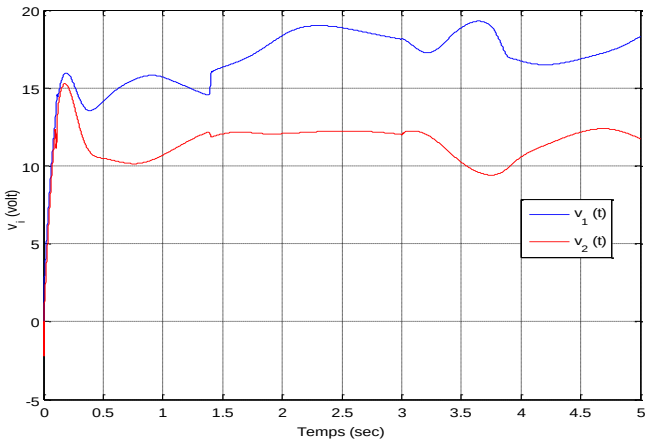
(b)


v1 (t)
v2 (t)
vi (volt)
Temps (sec)

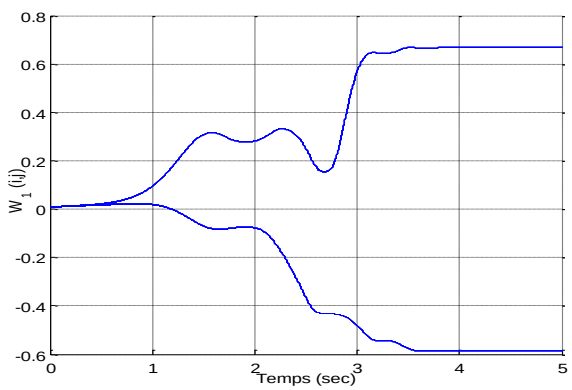
(c)


v1 (t)
v2 (t)
vi (volt)
Temps (sec)

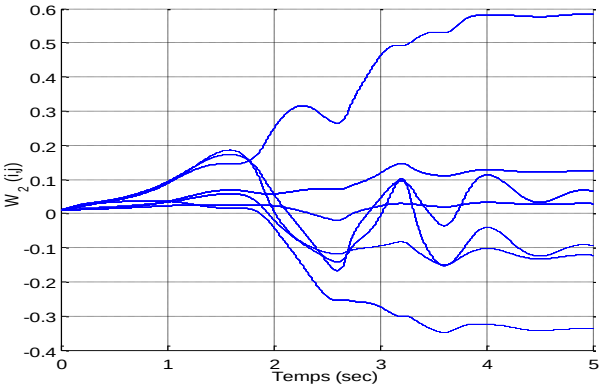
(d)


v1 (t)
v2 (t)
vi (volt)
Temps (sec)

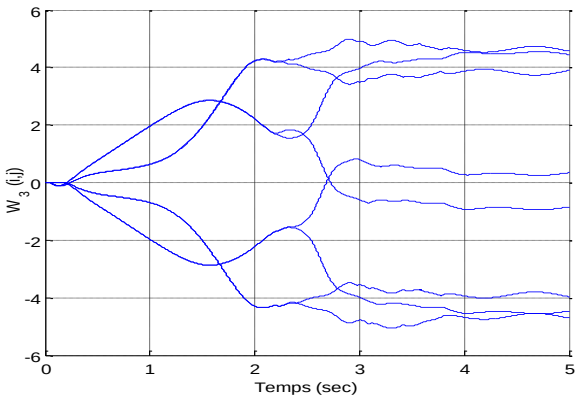
(e)



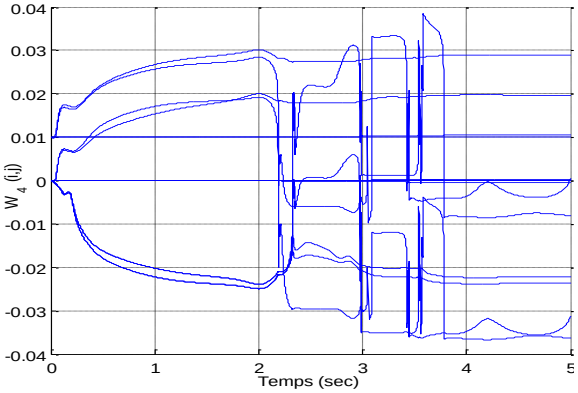
(f)



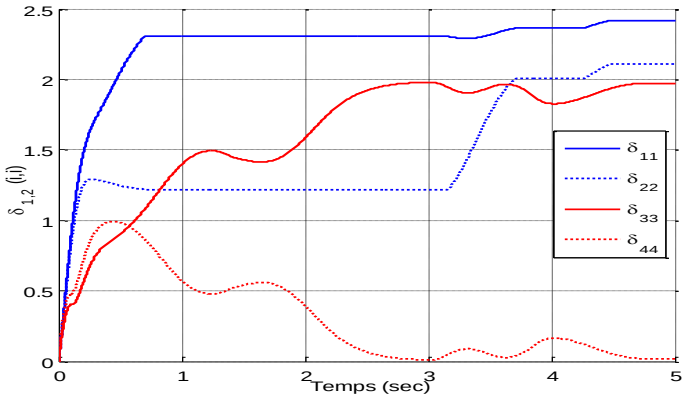
(g)



(h)



(i)



(j)

Figure.3.10. (a,b) trajectoire de sortie et de référence 1 et 2 dans la plan $\{X, Y\}$. (c,d,e) le signale de commande dans l'absence des perturbations, la présence des perturbations et l'absence de terme robuste et la présence des perturbations et de terme robuste, respectivement, pour la trajectoire de référence 1. (f,g,h,i) les poids adaptatifs des réseaux de neurones pour la trajectoire de référence 1. (j) les bornes supérieures adaptatives de la commande robuste pour la trajectoire de référence 1.

III.10. Conclusion :

Dans le cadre de ce chapitre, nous avons répondu aux contraintes imposées sur la loi de commande dans notre problématique citée dans le chapitre 1. La première loi de commande proposée combine une loi optimale, un système neuronal adaptatif et une commande robuste adaptative multi-paramètres. La deuxième loi de commande prend en compte la dynamique électrique du robot. La loi optimale est conçue d'une manière explicite sans passer par la résolution des équations du Riccati tout en minimisant un indice de performance. Le réseau de neurones adaptatif est sans doute un système d'approximation pour rapprocher en ligne le comportement de la dynamique du robot. Ensuite, la commande robuste est considérée comme une commande auxiliaire pour compenser les perturbations et l'erreur d'approximation du système neuronal. Notons que, la loi de commande proposée apprend et découvre de manière en ligne la dynamique inconnue du système; de sorte qu'aucune information préalable n'est requise sur le système de point de vue modélisation. En effet, la loi de commande est caractérisée par une réponse rapide, des performances de suivi de trajectoire satisfaisantes, un minimum d'énergie, et une robustesse face à des incertitudes paramétriques variant dans le temps et aux perturbations externes, par conséquent, pourraient être facilement adoptées pour d'autres système robotique.

Pratiquement, il n'y a pas de procédure claire et systématique pour guider le choix des gains. L'ajustement des valeurs numériques pour les paramètres de contrôle peut être se faire en trois étapes ; tout d'abord, nous ajustons chaque paramètre proportionnel à l'erreur en question par la méthode Essais-erreur pour obtenir une erreur de suivi de trajectoire raisonnable mais pas encore acceptable. Deuxièmement, nous ajustons la partie du système neuronal; où la taille de la couche cachée devrait être sélectionnée par la méthode ascendante pour satisfaire les performances du système. En outre, en sélectionnant les poids du réseau initiaux comme zéro, le système neuronal sort de la boucle de commande et ne laisse que les autres termes stabilisent le système jusqu'à ce que le réseau commence à s'adapter. Enfin, le terme robuste est ajusté comme une commande auxiliaire pour minimiser davantage l'erreur de suivi de trajectoire et garantir la robustesse face à une perturbation externe.

Toutefois, un grand nombre d'essais et plusieurs tentatives de paramétrage ont été faits par la méthode Essais-erreur afin de trouver la bonne combinaison entre les gains de la loi de commande, ce qui représente le problème majeur dans l'implémentation d'une loi de commande à plusieurs gains.

Chapitre IV.
Algorithme d'optimisation des paramètres des lois de commande par les colonies de fourmis et le système flou.

IV.1. Introduction

Notre dernière contribution dans cette thèse, réside dans le développement d'une nouvelle approche intelligente pour le réglage des paramètres des lois de commande. La méthode la plus souvent utilisée est le réglage empirique par Essais-erreur. Toutefois, il n'y a pas de procédure claire et systématique pour guider le choix des gains d'une loi de commande compliquée.

Par ailleurs, en examinant les différentes techniques d'optimisation par méta-heuristique citées dans notre problématique au chapitre 1, et généralement dans la littérature qui sont conçues spécialement pour l'optimisation des paramètres d'une loi de commande donnée, nous constatons que leur paramétrage, le nombre d'itérations et le temps de traitement nécessaire pour atteindre la convergence des paramètres à optimiser est souvent l'un des défis majeur le plus rencontrés dans l'implémentation en temps réel.

En effet, adapter un algorithme d'optimisation par méta-heuristique pour un fonctionnement en temps réel n'est pas une chose aisée, ce qui nous a amené à développer une nouvelle stratégie qui présente comparativement plusieurs avantages. Notre technique repose sur une hybridation entre le principe de l'optimisation approchée par les colonies de fourmis et la logique floue. L'intérêt majeur de cette approche réside dans le fait que la synthèse de l'algorithme est plus générique, plus facile à mettre en œuvre et ne nécessite pas le savoir-faire d'un utilisateur qualifié pour leur paramétrage. Cet algorithme présenté dans ce chapitre, mette en évidence la solution clé du réglage autonome et en temps réel des paramètres d'une loi de commande compliquée, et c'est à ce niveau que se situe le défi que beaucoup de méthodes de réglage ne sont pas arrivées à surmonter. Nous souhaitons à travers cette contribution répondre à cette question en développant une nouvelle approche de méta-heuristique basée sur les colonies de fourmis et qui intègre un système flou dans le but de mieux orienter la recherche vers une meilleure solution dans l'espace de recherche.

IV.2. Généralités sur le système flou :

Après sa naissonce en 1963 par son fondateur ou son père criateur Lotfi Zadeh, qui a posé la théorie des sous-ensembles flous et la notion de variables linguistiques, la logique floue est devenue de plus en plus populaires dans le domaine de l'ingénérie. Cette technique a été introduite dans le but d'approcher le raisonnement humain et transférer certaines de ces capacités à l'aide d'une représentation des connaissances facilement interprétables.

La résolution par logique floue a été abordée historiquement en 1975 par le professeur Mamdani, qui a développé une stratégie de contrôle pour la conduite d'un moteur à vapeur. Parallèlement, la véritable application industrielle de la logique floue en 1980 par la société danoise Smidth, qui a réalisé le contrôle par logique floue d'un four à ciment. À la fin des années 1980, la logique floue connaît son véritable départ au Japon, notamment grâce aux travaux de Sugeno. Dans la même époque, à Sendai au Japon, un contrôleur de freinage avec une bonne précision d'arrêt d'un métro a été réalisé avec un fonctionnement plus confortable. La conception de ce contrôleur avait besoin d'une année de simulation et ajustements de paramètres avant le début de son exploitation commercial. **[89]**. Et depuis, les applications de la logique floue ont vécu une véritable explosion, et il suffit de consulter l'abondante littérature dans le web sur le sujet pour s'en convaincre.

La démarche principale dans la conception des systèmes flous consiste à implémenter un savoir-faire humain sous la forme d'un algorithme qui fait le lien entre une modélisation symbolique et une modélisation numérique. En cela, elle peut être classée dans le domaine de l'intelligence artificielle.

Le formalisme mathématique fourni par la logique floue, présente l'avantage d'utiliser des règles simples permettant de traduire facilement le raisonnement d'un expert de prise de décisions pour répondre à un problème spécifique. En effet, l'humain peut définir une solution de commande de façon linguistique avec un minimum de connaissance sur le processus. La logique floue permet de traduire cette solution en un ensemble de règles de la forme « Si Observation Alors Décision ».

Le principe de la logique floue permet d'étendre la logique de Boole classique de tout ou rien sous une forme algébrique. En effet, la variable booléenne qui signifie qu'une proposition soit vraie ou fausse et ne peut prendre que l'une des deux valeurs 0 ou 1, est généralement inconvenable à la plupart des phénomènes, Ainsi, l'idée était de remplacer l'appartenance d'une variable boolienne à un ensemble classique par un degré d'appartenance pouvant prendre toutes les valeurs comprises entre 0 et 1. On parle donc d'une logique à plusieurs niveaux.

A. Ensemble flou :

La définition d'un ensemble flou s'adapte à une représentation des connaissances imprécises exprimées soit par un langage humain, ou sont obtenues avec des capteurs de mesure et d'observation imprécises.
La représentation à plusieurs niveaux ou le caractère graduel des ensembles flous ressemble à l'idée que, plus on se rapproche à un ensemble, plus l'appartenance à cet ensemble est forte.

1. Fonction d'appartenance :

Soit Ω un ensemble, un sous-ensemble flou $A \subset \Omega$ défini sur l'univers X est caractérisé par une fonction d'appartenance $\mu_A \in [0,1]$ qui représente le degré d'appartenance d'une variable $x \in X$ à l'ensemble A (figure.4.1). Autrement dit, un élément peut appartenir simultanément aux plusieurs sous-ensembles de manière graduelle.

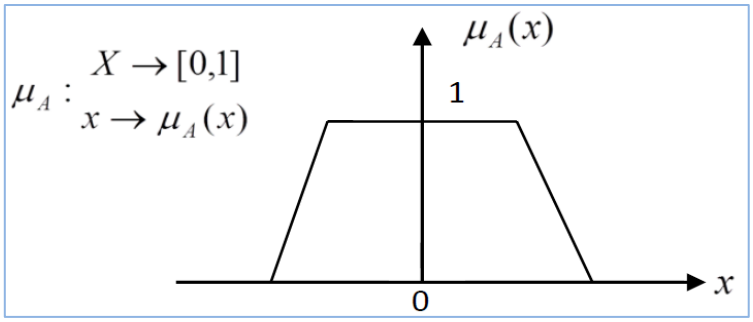


Figure.4.1. Fonction d'appartenance trapézoïdale.

Un sous-ensemble flou $A \subset \Omega$ peut être représenté comme suit :

$$A = \{(x, \mu_A(x)); x \in X, \mu_A(x) \in [0\ 1]\}. \quad (4.1)$$

L'idée de la logique floue est de traduire une valeur numérique en prédicat (appelée prémisse ou condition) dont la validité est floue. Il est ainsi possible de modéliser un processus ou une expertise en prenant en compte l'imprécision du modèle ou de l'expert.

Si une donnée n'est pas connue précisément, elle peut être exprimée par un intervalle de confiance précis. Cet intervalle est un ensemble de valeurs possibles pour la donnée.

Pour fixer l'idée nous prenons l'exemple suivant : supposons que x une variable mesurée (Figure.4.2) tel que x=4.5 ; en langage flou, elle peut être appartient à la zone positive avec $\mu(x) = \mu_p$ et simultanément à la zone zéro positive avec $\mu(x) = \mu_z$, autrement dit, la décision de sortie va prendre en compte que $x = 4.5 \pm \Delta x$ tel que Δx représente l'erreur de mesure.

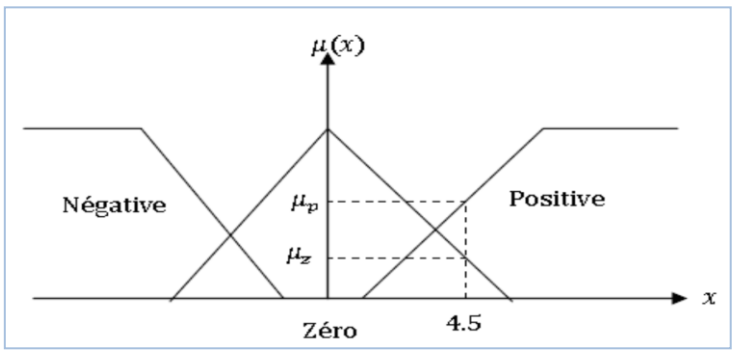


Figure.4.2. Degré d'appartenance d'une variable numérique

2. Opérations sur les ensembles flous :

Nous trouvons plusieurs opérations et notions complémentaires dans la théorie des sous-ensembles flous. Nous rappelons brièvement quelques opérations sur les ensembles flous :

- La disjonction floue, «OU» ou union :

La fonction d'appartenance $\mu_{A\cup B}$ est définie, pour tout $u \in U$, par :

Mamdani : $\mu_{A\cup B}(x) = \max\{\mu_A(x), \mu_B(x)\}$, (4.2)

Sugeno : $\mu_{A\cup B}(u) = \mu_A(u) + \mu_B(u) - \mu_A(u)\cdot\mu_B(u)$. (4.3)

- La conjonction floue, «ET» ou intersection :

La fonction d'appartenance $\mu_{A\cap B}$ est définie, pour tout $u \in U$, par :

Mamdani : $\mu_{A\cap B}(u) = \min\{\mu_A(u), \mu_B(u)\}$, (4.4)

Sugeno : $\mu_{A\cap B}(u) = \mu_A(u)\cdot\mu_B(u)$. (4.5)

3. Variable linguistique :

Une variable linguistique est caractérisée par un quintuple (x, T, X, R, M) dans lequel :

- x est le nom de la variable,
- T est l'ensemble des noms des valeurs linguistiques de x,
- R est une règle sémantique pour la génération des termes des valeurs de x,
- M est une règle syntaxique qui associe à chaque valeur son sens.

Un exemple représentatif typique (Figure.4.3) de la variable linguistique « Température » avec trois termes linguistique Basse, Moyenne et Elevée

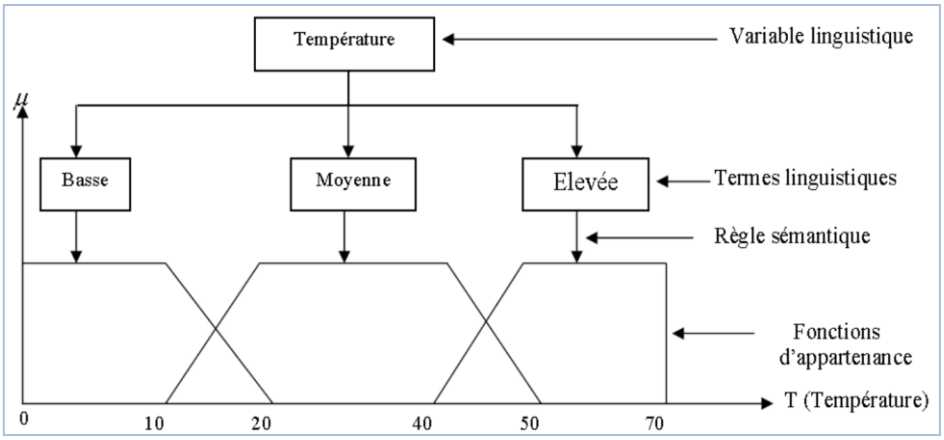


Figure.4.3. Exemple d'une variable linguistique.

B. Structure d'un système flou :

En général, les systèmes flous s'appuient sur une représentation de la connaissance sous forme de règles « Si…Alors » qui permettent de représenter les relations entre les variables d'entrée et de sortie dont l'expression générique est de la forme : **Si** antécédent **Alors** conséquent

L'antécédent ou prémisse représente une description linguistique qui indique les conditions de validité du phénomène représenté. Le conséquent ou conclusion représente le comportement associé aux conditions de validité décrites par l'antécédent. Selon la structure particulière de la proposition conséquente, on peut distinguer deux types de modèles flous:

1- Modèle de Mamdani :

Dans un modèle de type Mamdani, l'antécédent et le conséquent sont tous des propositions floues qui utilisent des variables linguistiques. L'avantage majeur de ces modèles est que les prémisses et les conclusions des règles sont de nature entièrement linguistique. Le jeu de règles de type Mamdani est de la forme $Si\ x_1 est\ A_1\ et\ x_2 est\ B_1\ Alors\ \ y_i est\ C_1$

2- Modèle de Takagi-Sugeno :

Comme celui de Mamdani, un modèle de type Takagi-Sugeno construit à partir d'une base de règles "Si…Alors…", dans laquelle la prémisse est exprimée linguistiquement, mais la conclusion utilise des variables numériques plutôt que des variables linguistiques. Le

conséquent peut s'exprimer de manière plus générale par une fonction ou d'une équation différentielle dépendant des variables associées à l'antécédent. Le jeu de règles du type Takagi-Sugeno est de la forme:

$$Si\ x_1 est\ A_1\ et\ x_2 est\ B_1\ Alors\ \ y_i = f_i(x). \qquad (4.6)$$

La synthèse d'un système flou passe par les étapes successives suivantes :

1. La partition en sous-ensembles flous des différents variables misent en jeu, et la transformation de ces variables en variables floues ; La Fuzzification,

2. Déterminer une base de règle qui caractérise le fonctionnement désiré du système,

3. Évaluation de règles floues dans un mécanisme d'inférence,

4. Extraire la sortie numérique de la fonction d'appartenance de sortie établie par le mécanisme d'inférence ; la défuzzification.

Le choix de la méthode de fuzzification, l'inférence et la défuzzification permet d'établir différentes classes de systèmes flous.

a. Interface de Fuzzification :

Comme le mécanisme d'inférence traite des fonctions d'appartenances à partir des grandeurs linguistiques, l'opérateur de fuzzification consiste à transformer les variables d'entrée en variables linguistiques représentées par des ensembles flous. Il s'agit d'attribuer à chaque variable numérique un degré d'appartenance à un ensemble flou. Si une entrée $x = 4.5$, (Figure.4.4), après fuzzification cette valeur appartient à Zéro avec un degré μ_Z et appartient aussi à Positive avec un degré μ_P.

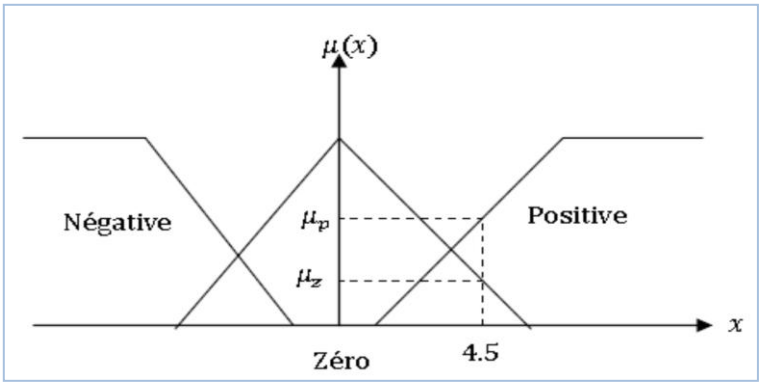


Figure.4.4. Exemple de fuzzification.

b. Base de règles floues et mécanisme d'inférence

La notion d'une fonction de transfert entre les variables d'entrée et celles de sortie, est présentée dans le système flou par des règles de décision pour formaliser le raisonnement humain. En effet, la base de règles floues n'apporte pas de solution, sauf s'il existe une expertise ou un savoir-faire. Une règle floue est du type : si « prédicat » alors « conclusion ».

Le mécanisme d'inférence quant à lui, permet de calculer le sous-ensemble flou relatif à la sortie du système, à partir de la base de règles et l'entrée fuzzifiée. On trouve plusieurs possibilités pour réaliser l'opérateur d'inférence, les plus utilisées sont ; opérateur d'inférence max-min et max-prod. Le tableau.4.1 résume ces deux principales méthodes

Tableau 4.1. Exemple de méthodes d'inférence.

Max-min	Max-prod
ET ⇔ min	*ET* ⇔ min
OU ⇔ max	*OU* ⇔ max
Alors ⇔ min	*Alors* ⇔ *prod*

Pourquoi prend-on le maximum des minimums ? Kaufmann un des fondateurs de la théorie des ensembles flous et de la logique floue, écrit : « En prenant le minimum, on est certain de prendre la situation la plus pessimiste. En prenant le maximum de ces minimums, on prend la meilleure valeur étant donné les informations disponibles » **[89]**. Dans la conclusion, en prenant le minimum ou le produit des conditions, on considère que les conditions sont satisfaites au degré le plus faible. Si une des conditions n'est plutôt pas satisfaite, on ne donne donc aucune ou faible certitude à la conclusion.

L'exemple suivant, figure.4.5-4.6 permet de montrer la démarche de l'opérateur d'inférence max-min et max-prod. Supposons que $x_1 = 0.4\, et\, x_2 = -0.5$ et que le mécanisme d'inférence est décrit par les deux règles suivantes:

Si (x_1 est P **ET** x_2 est Z), **Alors** y est Z ; **Ou Si (**x_1 est Z **OU** x_2 est N), **Alors** y est N.

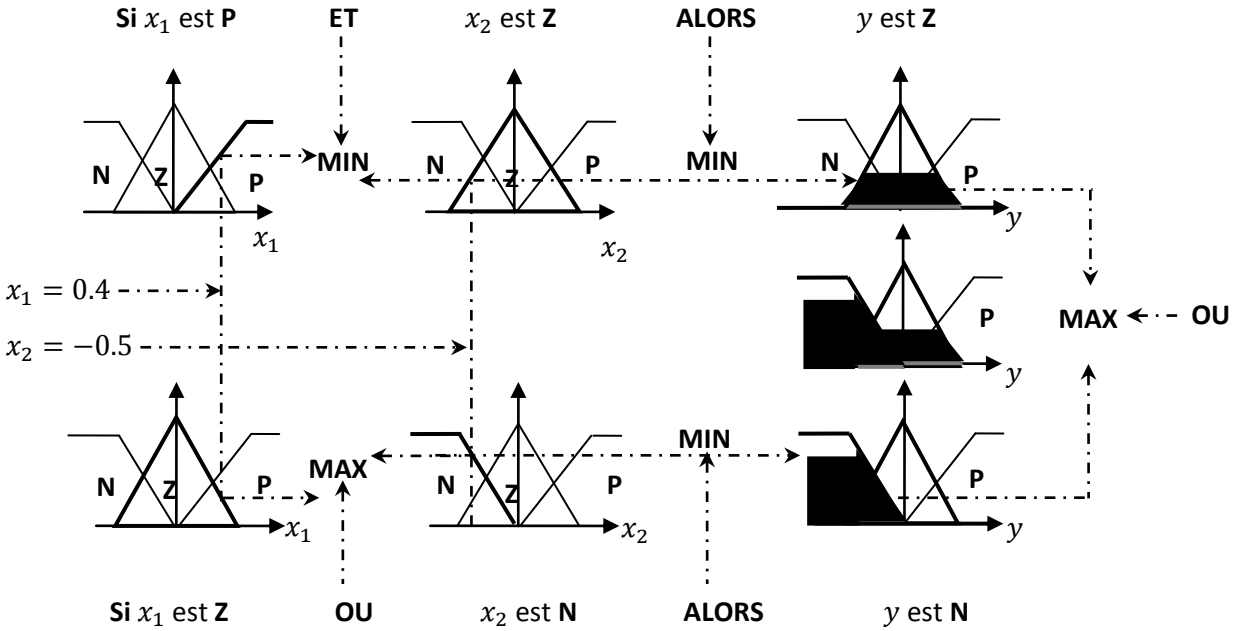


Figure.4.5. Opérateur d'inférence Max-Min.

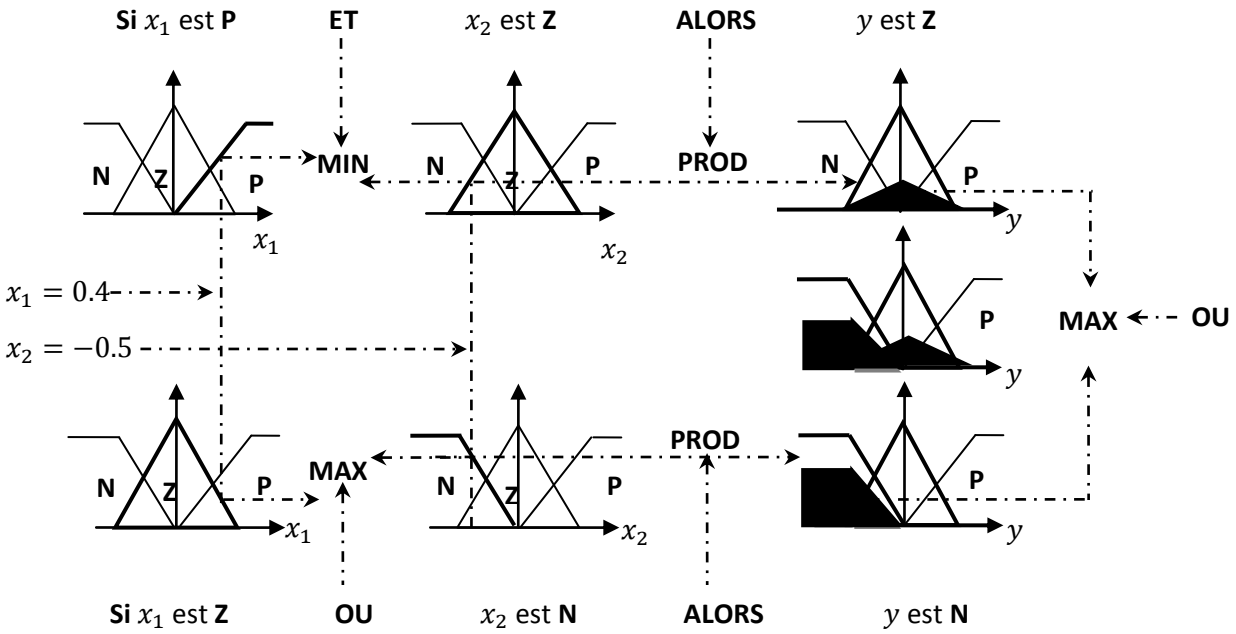


Figure.4.6. Opérateur d'inférence Max-Prod.

c. Interface de défuzzification :

L'opérateur de défuzzification représente la transformation inverse de l'opérateur de fuzzification. Comme le mécanisme d'inférence fournit l'ensemble flou décrivant la conclusion, une sortie se présente sous la forme d'une grandeur numérique via la défuzzification. Dans le cas de type Mamdani, une valeur numérique doit être obtenue. La méthode la plus communément utilisée est le centre de gravité de l'union des sous-ensembles flous. Il s'agit de calculer la position du centre de la surface résultante formée par l'union, l'abscisse de ce centre représente finalement la sortie du système flou. Lorsque les fonctions d'appartenance pour la variable de sortie de formes trapézoïdales et triangulaires, le centre de gravité peut être calculé par :

$$y = \frac{\sum \mu_i(y)\, p_i}{\sum \mu_i(y)}\,, \tag{4.7}$$

où p_i représente le centre de gravité de la surface associée au sous ensemble i de la sortie y.

IV.3. Généralité sur les algorithmes de métaheuristiques :

IV.3. 1. Introduction :

Après avoir donné quelques définitions préalables et placé les métaheuristiques dans leur contexte, nous rappellerons les énoncés de quelques algorithmes, en analysant leur mode de fonctionnement respectif ainsi que les modalités de leur utilisation.

En mathématiques, l'optimisation recouvre toutes les méthodes qui permettent de déterminer l'optimum d'une fonction, avec ou sans contraintes. Dans les applications de l'ingénierie, les méta-heuristiques occupent une place très importante pour résoudre des problèmes d'optimisation difficiles. Le terme méta-heuristique désigne aujourd'hui un très grand nombre de modèles algorithmiques.

Par abus de langage, la complexité d'un algorithme signifie sa complexité en temps. En effet, la théorie de la complexité s'intéresse à l'étude de la difficulté des problèmes en informatique. On dit qu'un problème est facile s'il existe un algorithme de complexité polynomiale (soit proportionnel à N^n ,où N est le nombre de paramètres inconnus du problème) le résolvant ; on

dit qu'un problème est difficile s'il n'existe pas d'algorithme de complexité polynomiale le résolvant.

À titre exemple, dans un problème combinatoire, l'optimisation consiste à trouver la meilleure solution entre un nombre fini de choix, ou encore, minimiser une fonction sur un ensemble fini de ces variables. Pour les non spécialistes dans le domaine, ceci parait facile avec une recherche exhaustive par énumération explicite, mais le problème devient infaisable et provoque l'explosion combinatoire dès que la taille du problème est grande, et il n'est pas alors faisable d'examiner toutes les solutions. Parmi les exemples typiques dans l'optimisation combinatoire est le problème du voyageur de commerce ou celui des tournées de véhicules, qui consiste à calculer les meilleurs parcours d'un ensemble de véhicules de livraison dans plusieurs destinations à partir d'un entrepôt. Pour fixer l'idée, dans le cas du problème du voyageur de commerce, par exemple, l'espace de recherche est de l'ordre $\frac{(n-1)}{2}!$, où *n* est le nombre de villes à visiter. Avec seulement 50 villes, il faudra évaluer $3.04.10^{62}$ trajets.

Par ailleurs, pour les problèmes d'optimisation continue, on trouve des méthodes exactes, dites déterministes, qui permettent de résoudre certains types de problèmes, sachant que la fonction objectif ou à minimiser est convexe, continue ou encore dérivable et intuitivement a un modèle connue.

En effet, si les méthodes de résolution exactes permettent d'obtenir une solution optimale garantie, dans certaines situations, on peut cependant accepter une solution sans garantie d'optimalité, mais avec un temps de calcul raisonnable. Dès que les approches traditionnelles échouent, il devient intuitivement de se tourner vers des techniques de résolution approchée.

En partant de cette constatation, des techniques développées appelées heuristiques, qui donnent des solutions proches de l'optimum pour les problèmes d'optimisation difficiles. Au début, la plupart sont conçues précisément pour un type de problème donné. D'autres, au contraire, désormais appelées "méta-heuristiques", qui s'adaptent aux différents types de problèmes combinatoires et continus.

IV.3.2. Classification des métaheuristiques :

Dans cette section, nous allons essayer de classifier et présenter la stratégie de recherche de quelques méta-heuristiques le plus populaires, dans leur version de base avec une description des algorithmes associés, et notamment les techniques que nous avons rencontrés pendant nos recherches qui exposent le champ d'application qui nous intéresse dans cette thèse.

La naissance des méta-heuristiques a conduit à une avancée importante pour la résolution pratique de nombreux problèmes dans le domaine de l'ingénierie, ce qui a créé par la suite une inspiration pour le développement même de ces approches.
Parmi les méta-heuristiques, on peut différencier deux classes **[90]**, à savoir ; les méthodes à base de voisinage ou encore de la recherche locale, qui consiste partant d'une seule solution initiale, à faire évoluer et améliorer cette solution dans l'espace de voisinage. La deuxième classe appelée les approches distribuées qui manipulent en parallèle une population de solutions. Une autre manière de classifier les méta-heuristiques consiste à séparer celles qui sont inspirées de la biologie ou l'être vivant, de celles qui ne le sont pas.

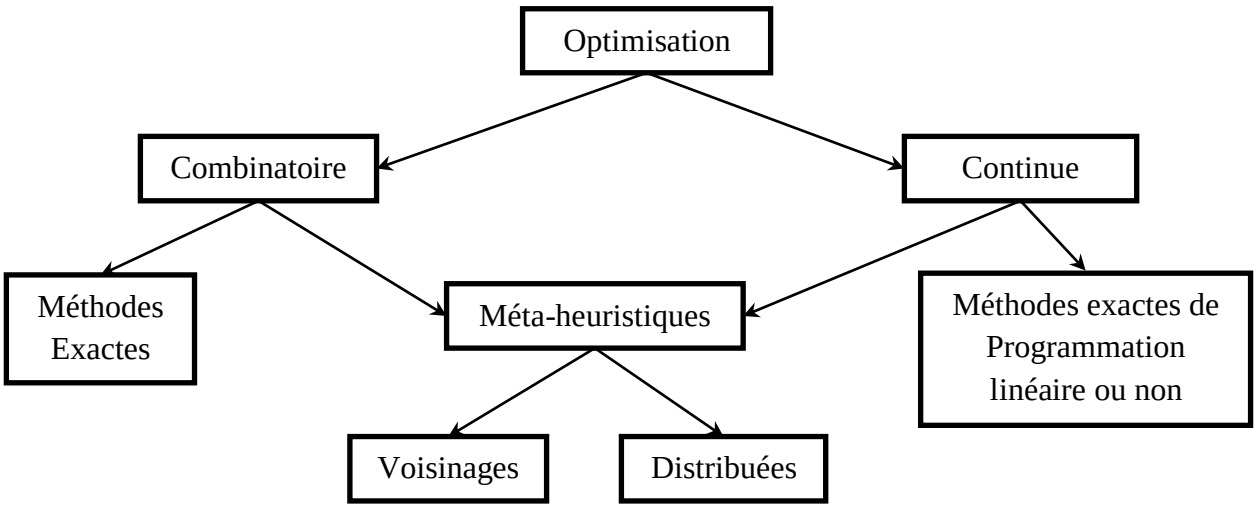


Figure.4.7. Classification générale des méthodes d'optimisation.

En règle générale, les méta-heuristiques doivent respecter un équilibre entre l'intensification de la recherche et la diversification de celle-ci, qui joue un rôle principal dans leurs réussites face à un problème d'optimisation donné. D'une part, en s'appuyant sur les solutions déjà trouvées, l'intensification s'intéresse à des solutions de plus grande qualité. D'autre part, la diversification met en place une stratégie qui permet d'ignorer quelques solutions meilleures

afin d'explorer un plus grand espace de solutions et d'échapper à des minima locaux. En effet, ne pas respecter cet équilibre par l'algorithme conduit à une convergence trop rapide vers un minimum local ; à cause de manque de diversification suffisante, ou à une exploration trop longue ; à cause de manque d'intensification suffisante.

A. Méthodes de recherche locale :

Méta-heuristiques à base de voisinages s'appuient toutes sur un même principe. à partir d'une solution initiale bien déterminée à l'avance et qui est considérée comme un point de départ, et peut être obtenue de façon exacte, ou par tirage aléatoire, la recherche consiste à passer d'une solution à une autre voisine par des déplacements successifs pour réaliser une trajectoire, un parcours progressif dans l'espace des solutions, et l'ensemble résultant est appelé le voisinage. Dans cette catégorie, on trouve ; la méthode de Descente, le recuit simulé, la recherche Tabou.

En règle générale, la recherche locale s'arrête quand la meilleure solution localement est trouvée. C'est le cas des méthodes de descente qui s'articulent toutes autour d'une démarche simple. Partir d'une solution existante, chercher une solution dans le voisinage et accepter cette solution si elle améliore la solution actuelle. En effet, l'équilibre souhaité entre intensification et diversification n'existe donc plus comme cette méthode se base sur une amélioration progressive de la solution et donc souvent reste bloquée dans un minimum local rencontré durant la recherche. Une solution très simple de diversifier la recherche consiste à exécuter l'algorithme en prenant un autre point de départ à savoir le Multi-start descente ce qui simule une recherche en parallèle.

Le cas des méthodes de recuit simulé, représente un cas plus étendu de la méthode de Descente, la technique de recherche peut s'échapper d'un minimum local par le fait de créer des mouvements pour lesquels la nouvelle solution retenue sera de qualité moindre que la précédente.

Exemple : Recuit Simulé (Simulated Annealing) :

Cette technique a été introduite en 1983 par Kirkpatrick *et al* **[90]** et peut être considérée comme la première méta-heuristique. Le principe s'inspire d'un processus de métallurgie qui consiste à améliorer la qualité d'un métal solide. En partant d'une température élevée où le métal est dans l'état liquide, on refroidit progressivement le métal à la recherche d'un meilleur équilibre thermodynamique. En, effet, passer d'un état du métal à un autre simule le passage d'une solution à une autre voisine.

Pour simuler cette évolution, la méthode du recuit simulé se base sur l'algorithme de Métropolis, à partir d'une configuration initiale, on provoque un changement itératif sur la température T, alors si cette nouvelle configuration a pour effet de diminuer l'énergie *E* du système, elle est acceptée, sinon, elle est acceptée avec la probabilité $e^{\Delta E/T}$.

Procédure recuit simulé :

Solution initiale s, Poser T ← T0

Répéter :

Choisir aléatoirement s ∈ N(s) //Voisinage,

Générer un nombre réel aléatoire r dans [0,1],

Si r $< e^{\frac{f(s)-f(s')}{T}}$ *alors Poser s ← s',*

Mettre à jour T,

Jusqu'à ce que le critère imposé soit satisfait,

Fin procédure.

Une interprétation de son fonctionnement est la suivante

- Si $f(s') < f(s)$ alors $e^{\frac{f(s)-f(s')}{T}}$ >1, donc r est toujours inférieur à cette valeur, et on accepte la solution *s'*, une meilleure solution est donc toujours acceptée, ce qui paraît logique,
- Si $f(s') > f(s)$ et *T* est très grand, alors $e^{\frac{f(s)-f(s')}{T}} \cong 1$, et il y a de fortes chances d'accepter *s'*, bien que la solution *s'* soit plus mauvaise que *s* ,
- Si $f(s') > f(s)$ et *T* est très petit, alors $e^{\frac{f(s)-f(s')}{T}} \cong 0$, et on va donc probablement refuser *s'*

De point de vue paramétrage, le choix de la fonction de température et sa valeur initiale est déterminant et dépend de la qualité de la solution de départ. Le paramètre *T* peut variée linéairement à chaque itération, mais on peut aussi envisager une décroissance par paliers, où T reste constant, tant qu'un certain critère n'a pas été rempli. Une fois cette condition atteinte, on varie légèrement la température et on recommence une nouvelle recherche à ce nouveau palier. En principe, l'algorithme est stoppé lorsque la température atteint la valeur nulle ou bien lorsque la solution trouvée est considérée acceptable.

B. Méthodes de recherche distribuées:

Contrairement aux approches locales, les méta-heuristiques à base de population consistent à évoluer un ensemble de solutions en parallèle, et possible de combiner ces solutions entre elles pour en former de nouvelles en essayant d'hériter des bonnes caractéristiques des solutions précédentes. Le processus est répété jusqu'à ce qu'un critère d'arrêt soit satisfait. En effet, l'évolution de plusieurs solutions permet d'améliorer l'exploration de l'espace de recherche. Dans cette famille, on cite par exemple : les algorithmes génétiques, les essaims particulaires, les colonies de fourmis,...etc.

1. Exemple de l'algorithme génétique:

Les Algorithmes Génétiques (AGs) s'inspirent de la capacité des populations d'organismes vivants à s'adapter à leur environnement à l'aide de mécanismes d'héritage génétique. Autrement dit, ces algorithmes représentent une version artificielle de la théorie de l'évolution de Charles Darwin. Cette technique est devenue populaire après la publication en 1989 de Genetic Algorithms in search, Optimization and Machine Learning par D.E. Goldberg **[90].** Cette approche a connu par la suite de nombreuses transformations selon la variété des problèmes.

Le même principe est partagé généralement dans les algorithmes génétiques. À partir d'une population d'individus initiale générée aléatoirement ou représente des solutions possibles comme un point de départ, évoluent itérativement parallèlement en évaluant leur performance relative. Sur la base de ces performances, on cherche à créer une nouvelle population en simulant les deux mécanismes qui conduisent l'évolution des êtres vivants, à savoir ; la

sélection qui détermine quels individus d'une population survivent, et la reproduction qui assure l'héritage et la combinaison des survivants.
La nouvelle population formée remplace la population précédente et le processus recommence à nouveau. Chaque individu dans une population possède une certaine qualité de solution de l'objectif visé. D'une génération à la suivante, la qualité de la population doit globalement s'améliorer.

En langage algorithmique, chaque individu d'une population est dénommé chromosome, et chaque chromosome est composé des variables élémentaires appelées gènes qui peut prendre différentes valeurs dans un codage donnée. La fonction à optimiser est souvent appelée fonction d'évaluation ou fitness.
L'évolution d'une génération à une autre se fait en deux étapes; la reproduction et le remplacement Figure.4.8. La reproduction consiste à appliquer les fameux opérateurs génétiques; croisement et mutation, sur les individus de la population en question. Les individus prenant part à la reproduction sont précédemment sélectionnés, en respectant le principe suivant; plus un individu est compétent, plus sa probabilité de participation à la reproduction est élevée. L'opérateur de croisement ou couplage permet de produire deux descendants à partir de deux parents choisis. L'opérateur de mutation, permet de créer un nouvel individu à partir d'un seul individu en changeant certaines caractéristiques d'une solution ce qui donne de la diversité à la population. L'étape suivante dans l'évolution dite de remplacement, consiste à choisir entre les membres de la nouvelle génération et la population courante au sens de la fonction objective associée à cette génération, ceux qui peuvent poursuivre l'évolution de l'algorithme.

Comme le facteur de diversification est assuré dans la mutation, ce qui signifie la modification aléatoire d'une partie de leur génotype, permet d'éviter de tomber dans un minimum local. L'intensification est assuré à la fois par les opérateurs de sélection et de croisement. En effet, la sélection est faite en favorisant les meilleurs individus qui forcent intuitivement la convergence rapide de la population. D'un autre côté, l'opérateur de croisement ou couplage est dessiné assurer l'héritage des meilleures caractéristiques aux enfants.

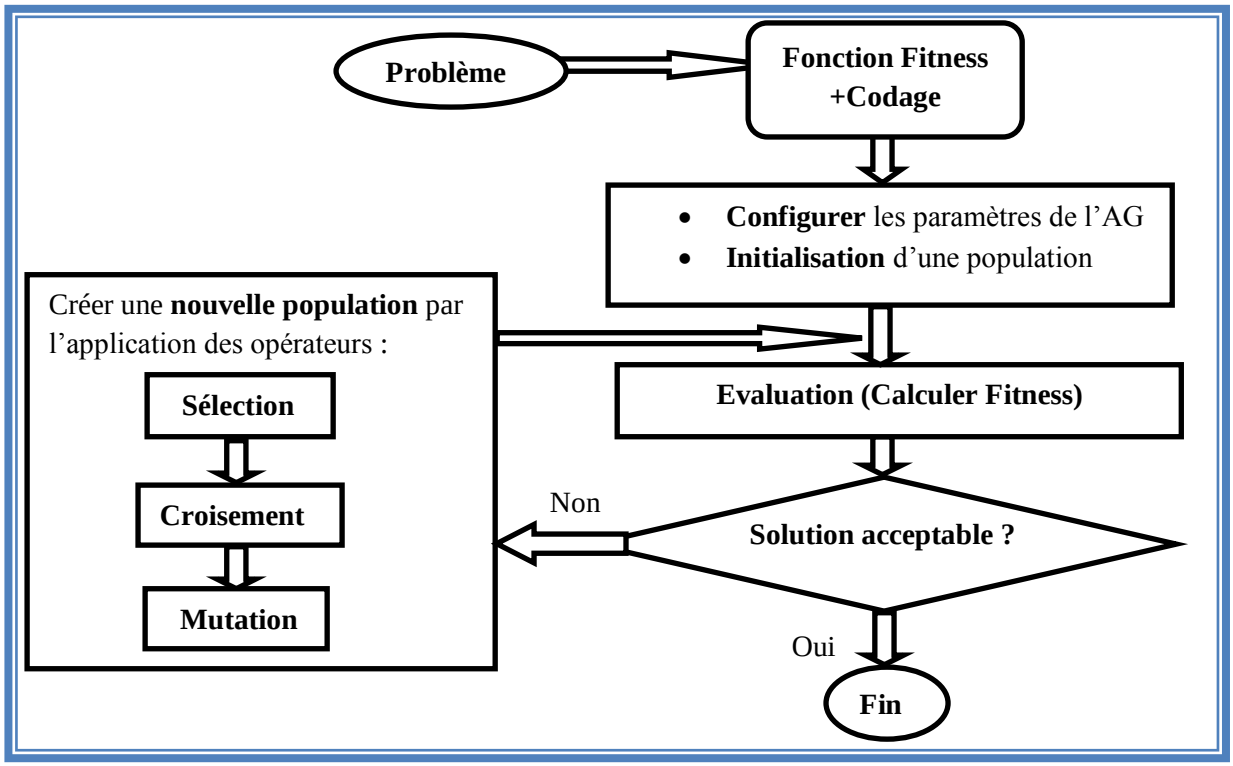


Figure.4.8. Organigramme de base de l'algorithme génétique.

2. Les algorithmes de colonies de fourmis :

Les algorithmes à base de colonies de fourmis ont été introduits par Dorigo. Le premier système introduit par Dorigo est appelé Ant System **[91].** Cette approche essaye de simuler la capacité collective de résoudre certains problèmes observée chez une colonie de fourmis malgré un membre est individuellement très limité. Tout comme le principe de la méthode des essaims particulaires (PSO particle swarm optimization) **[92]**, inspirée de la dynamique des populations qui s'organisent en foules oiseaux, poissons…etc.

La fourmi est un des insectes qui vit toujours dans une collectivité bien organisée. Ces insectes sociaux constituent des colonies appelées fourmilières, de quelques dizaines à plusieurs millions d'individus qui se sont adaptées à presque tous les milieux terrestres et souterrains. Les fourmis ouvrières se regroupent selon le travail partagé qu'elles auront à un stade de leur vie ; s'occuper de la reine, la construction et la défense du nid, la recherche de nourriture,…etc. Les fourmis ont une communication entre individus et se reconnaissent entre eux grâce à leur odeur captée avec leurs antennes. L'utilisation principale de cette odeur réside dans le repérage des chemins pour guider les fourmis vers des sources de nourriture

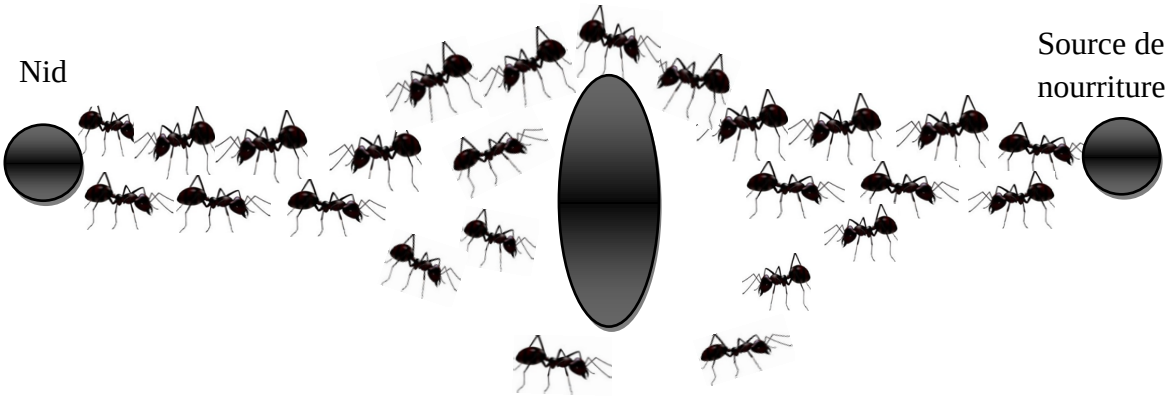
[91]. Chez la plupart des espèces de fourmis, la colonie dispose une capacité d'organisation sociale complexe qui permet d'accomplir des tâches difficiles rencontrés dans la nature, On parle alors d'intelligence collective guidée en raison que ce comportement n'est pas programmé chez les individus.

Les entomologistes ont analysé la collaboration qui s'établit entre les fourmis pour aller chercher de la nourriture à l'extérieur de la fourmilière. Il est remarquable que les fourmis suivent généralement le même chemin, et que ce chemin est le plus court possible. En 1980, Jean-Louis Deneubourg a montré expérimentalement qu'une colonie de fourmis faces à de deux chemins de longueurs différentes vers une source de nourriture, plus souvent choisissait le plus court chemin. Ainsi, Il décrit ce phénomène ; « … un « éclaireur », qui découvre par hasard une source de nourriture, rentre au nid en traçant une piste chimique. Cette piste stimule les ouvrières à sortir du nid et les guide jusqu'à la source de nourriture. Après s'y être alimentées, les fourmis ainsi recrutées rentrent au nid en renforçant à leur tour la piste chimique. Cette communication attire vers la source de nourriture une population de plus en plus nombreuse. Un individu qui découvre une source de nourriture y attire en quelques minutes *n* congénères (par exemple 5) ; chacun de ceux-ci y attirent à leur tour *n* congénères (25), et ainsi de suite. » *(sur wikipedia)*.

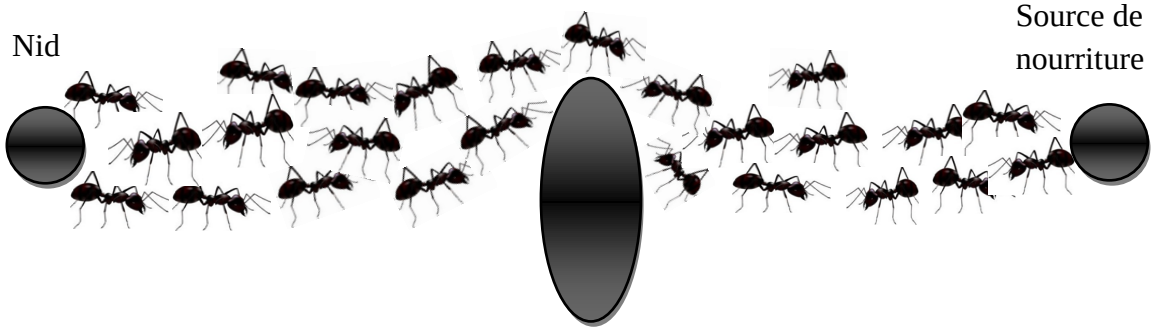
En partant de cette constatation, en s'inspirant de comportement des fourmis dans le domaine de l'informatique et la robotique afin de crier des systèmes distribués. Le premier algorithme est construit par Dirigo **[91]** de manière tout à fait analogue pour la résolution du célèbre problème du voyageur de commerce. Dans le problème de la navigation des robots, en 1991, l'ingénieur américain James McLurkin est le premier qui a élaboré l'idée de robots fourmis **[93].**

Ainsi, l'idée clé dans un système à fourmis est la phéromone. Lorsque plusieurs possibilités de chemin s'offrent à un élément de la colonie, ce dernier choisit avec une plus grande probabilité les directions marquées par de plus fortes concentrations de phéromone, plus concentré signifie que le chemin sera choisi plus souvent. En effet, comme la phéromone étant une odeur, elle s'évapore avec le temps. Donc, si peu de fourmis empruntent un chemin, il est possible que ne soit plus attractif au bout d'un moment, il en est de même si des fourmis exploratrices empruntent un chemin plus long. Par contre, si le chemin est fortement emprunté, chaque nouvelle fourmi qui passe redépose un peu de phéromone et renforce ainsi

la trace, qui donne alors à ce chemin une plus grande probabilité d'être emprunté et qui forme ainsi des routes à fourmis dans la nature.



Une fourmi choisit de tourner à gauche ou à droite avec la même probabilité, et la phéromone est déposée avec forte concentration sur le chemin plus court



Toutes les fourmis finalement choisissent le chemin le plus court

Figure.4.9. Une colonie de fourmis retrouve le plus court chemin face à un obstacle.

L'algorithme de base qui est conçu selon ce comportement était destiné à résoudre le problème du voyageur de commerce, et c'est sous cette forme que nous allons le présenter.
Le principe consiste à lancer des fourmis, et à les laisser élaborer pas à pas la solution, en allant d'une ville à l'autre. C'est donc un algorithme qui repose sur la construction progressive de solutions. Afin de ne pas revenir sur ses pas, une fourmi mémorise une liste qui contient la liste des villes déjà visitées, et puis au bout d'un moment les chemins qui conduisent à une bonne solution deviendront les plus empruntées jusqu'à l'obtention d'une meilleure solution.

Procédure Ants System :

Initialiser : $\tau_{ij}(1) = \tau_0$

Répéter :

Pour chaque fourmi *i*=1 à *k* faire ;

Choisir une ville initiale au hasard ;

Trajet(*i*) ;

Fin Pour ;

Mettre à jour phéromones ()

Sauvegarder la meilleure tournée : cost(t)=min (Trajet) ;

Jusqu'à ce que le critère de terminaison soit satisfait. ;

Solution=min (cost) ;

Fin procédure.

Dans l'étape *Trajet*(i), chaque fourmi construit une route en choisissant les villes selon une règle de transition aléatoire. Si $p_{ij}^k(t)$ est la probabilité qu'à l'itération *t* la fourmi *k* choisisse d'aller de la ville *i* à la ville *j*, alors est définie par :

$$p_{ij}^k(t) = \frac{(\tau_{ij}(t))^\alpha \cdot (\eta_{ij})^\beta}{\sum_{l \in N_i^k} (\tau_{il}(t))^\alpha \cdot (\eta_{il})^\beta} \qquad j \in N_i^k, \qquad (4.8)$$

$$= 0 \qquad\qquad j \notin N_i^k,$$

où N_i^k l'ensemble des villes que la fourmi k, placée sur la ville i, n'a pas encore visité à l'instant t

$\tau_{ij}(t)$ le taux de phéromone sur la route *i-j* à l'instant *t*, η_{ij} est l'inverse de la distance séparant les villes *i* et *j*,

α et β sont deux paramètres contrôlant respectivement l'influence du taux de phéromone sur le trajet *i-j*, et l'influence de la distance sur le trajet *i-j*.

En d'autres termes, plus il y a de phéromone sur le trajet entre deux villes, plus la probabilité est grande que la fourmi va choisir ce trajet. Mais ceci est contrebalancé par la longueur du trajet. Ainsi, les paramètres α et β permettent de régler cet équilibre.

Dans l'étape *Mettre à Jour Pheromones( )*. Lorsque toutes les fourmis ont construit leurs solutions, cette fonction modifie la quantité de phéromone sur les routes en fonction des

trajets effectivement empruntés par les fourmis, la quantité dépend de la qualité de la solution trouvée selon la formule :

$$\Delta\tau_{ij}^{k}(t) = \frac{Q}{L^{k}(t)} \quad \text{si le trajet } (i,j) \text{ est dans la tournée de la fourmi } k,$$

$$= 0 \quad \text{sinon,} \tag{4.9}$$

où Q est une constante et $L^{k}(t)$ est la longueur totale de la tournée de la fourmi k.

Un mécanisme d'évaporation des pistes de phéromone est introduit afin d'éviter la convergence rapide des fourmis vers un trajet qui se forment trop vite, et ne tombent trop rapidement dans un optimum local.

$$\tau_{ij}(t+1) = (1-\rho)\cdot\tau_{ij}(t) + \Delta\tau_{ij}(t)\ , \tag{4.10}$$

avec $\Delta\tau_{ij}(t) = \sum_{x=1}^{k}\Delta\tau_{ij}^{x}(t)$, k le nombre de fourmis et $0 < \rho \leq 1$.

Le choix d'un nouvel trajet se fait aléatoirement, c'est donc un facteur de diversification. L'intensification se retrouve dans l'opérateur de mis à jours de la quantité de phéromone qui concentre plus ou moins la phéromone associée à une solution.

En dernière analyse, les métaheuristiques constituent une famille de méthodes approchées adaptables à un grand nombre de problèmes d'optimisation, et généralement sans changements profonds de l'algorithme employé. De point de vue principe de fonctionnement, chacune explore et exploite l'espace de recherche selon une démarche propre. Effectivement, certaines méta-heuristiques présentent un organigramme d'être simple à mettre en œuvre, comme le recuit simulé; d'autres sont plutôt adaptées à certaine problème spécifique, comme le système de colonies de fourmis qui s'adapte avec les problèmes qui sont sous forme de graphe.

L'utilisation de ces algorithmes peut paraître relativement simple, mais passer de la théorie à la pratique est souvent difficile. En effet, pratiquement et quel que soit la méthode utilisée, on peut voir apparaître des difficultés et l'implémentation réelle pose des problèmes inévitables.

D'une part, les méta-heuristiques disposent de paramétrage n'est pas nécessairement trivial. La qualité des solutions trouvées par les méta-heuristiques dépend de leur paramétrage et de

l'équilibre entre un balayage de tout l'espace des solutions et une exploration locale intensifiée. Enfin, obtenir de bonnes performances passe obligatoirement par une étape d'initialisation. En pratique, seul le savoir-faire et l'expérience de l'utilisateur permet de gérer ces problèmes.

En règle générale, deux notions d'efficacités contradictoires rencontrées dans les méta-heuristiques à savoir ; la vitesse de convergence et la précision. La vitesse est souvent mesurée en nombre d'itérations nécessaire afin d'atteindre la solution désirée. La précision se rapporte à l'écart entre l'optimum trouvé par la méta-heuristique et l'optimum désiré du point de vue qualité de la solution. Généralement, un choix doit être fait quant au critère d'arrêt adéquat. Un nombre d'évaluation est souvent utilisé, mais on peut également choisir d'atteindre une valeur désirée de la fonction objective. Malheureusement souvent, un algorithme rapide est peu précis, et inversement.

IV.4. Conception de l'algorithme de réglage autonome AFA:

Dans cette section, nous détaillons l'algorithme AFA (Ants Fuzzy Algorithme) proposé dans un formalisme d'optimisation et de réglage autonome des paramètres d'une loi de commande donnée. Comme nous avons déjà mentionné, on s'est inspiré de concept de base de l'algorithme de colonies de fourmis et en ajoutant de nouveaux composants de système flou à cet algorithme, on peut alors construire un système hybride pour le développement d'une nouvelle technique intégrée en ligne dans la loi de commande. Il est évident que cette simple modification dans l'algorithme de base entraîne de profonds changements dans le comportement de l'algorithme lui-même.
En effet, le problème de réglage des paramètres est formulé comme une tâche d'optimisation en temps réel pour garantir que le système de robot mobile non-holonome suit la trajectoire désirée en accordant la minimisation d'une fonction objectif.

Nous considérons la loi de commande neuro optimale robuste suivante :

$$\bar{D}(q)\tau(t) = R^{-1}r + \hat{W}_1^T\psi\left(\hat{W}_2^T r_e\right) + \hat{W}_3^T sign(r(t)), \tag{4.11}$$

$$r(t) = \dot{e} + \Lambda e, \tag{4.12}$$

$$\dot{\hat{W}}_1(t) = \sigma_{W_1}\left(\psi\left(\hat{W}_2^T r_e\right) - \psi'\left(\hat{W}_2^T r_e\right)\hat{W}_2^T r_e\right)r^T, \tag{4.13}$$

$$\dot{\widehat{W}}_2(t) = \sigma_{W_2}\left(r_e r^T \widehat{W}_1^T \psi'\left(\widehat{W}_2^T r_e\right)\right), \quad (4.14)$$

$$\dot{\widehat{W}}_3(t) = \sigma_{W_3}\left(-\alpha\|r^T(t)\|\widehat{W}_3(t) + \mathrm{diag}\left[\left(\mathrm{sign}\left(r(t)\right)r^T(t)\right)_{ii}\right]\right). \quad (4.15)$$

Dans la conception de l'algorithme AFA, nous considérons l'erreur généralisée du système $E = [E_1^T(t),\ E_2^T(t)]^T$, avec $E_1(t) = [r^T(t), e^T(t)]^T$ l'erreur dans l'espace articulaire et $E_2(t) = \left[e_x(t), e_y(t), e_\varphi(t)\right]^T$ l'erreur dans l'espace opérationnelle.

Ainsi, nous associons pour chaque paramètre de la loi de commande l'erreur correspondant ; pour les paramètres $K_1 = \left[\sigma_{W_1}, \sigma_{W_2}, \sigma_{W_3}, \alpha, R^{-1}, \Lambda\right]^T$, l'erreur proportionnelle associée est $E_1(t)$, et pour les paramètres $K_2 = \left[k_x, k_y, k_\varphi\right]^T$(2.25), l'erreur proportionnelle associée est $E_2(t)$. Ainsi, chaque paramètre K_i sera estimé en temps réel pour minimiser son erreur en question.

Pour résoudre le problème, la logique floue et l'algorithme modifié de colonies de fourmis sont introduits dans la loi de commande de façon en ligne. En effet, les différents paramètres seront ajuster en temps réel ce qui permet d'éliminer la phase de réglage par la méthode empirique d'Essais-erreur, et d'assurer la propriété de convergence et ainsi donc, une meilleure précision de suivi de trajectoire sera obtenue.

L'idée d'utiliser le système flou est que l'information linguistique provenant de la décision d'un expert humain peut être incorporée de manière systématique dans le mécanisme d'inférence flou.

L'algorithme de fourmis utilise la recherche en groupe fonctionnant en parallèle et supervisé par un système flou dans chaque processus itératif temporisé en temps réel.

Initialement, la population des fourmis commence à partir de l'origine où les paramètres de la loi de commande sont initialisés par des zéros, de cette façon, nous n'avons pas besoin de conditions initiales appropriées. Ensuite, progressivement avec une exploration rapide de l'espace de recherche, les fourmis convergent vers la solution optimale, du sens celle qui minimise l'erreur de suivi de trajectoire. Le système flou est considéré comme un mécanisme auxiliaire qui augmente l'espace de recherche et change la zone explorée par les fourmis chaque itération pour maintenir la diversité.

Ce mécanisme intelligent simplifie la mise en œuvre du contrôle en temps réel et améliore les performances de la loi de commande, où le robot pourra s'adapter aux conditions changeantes et indésirables en temps réel pour toute trajectoire désirée et sans effort d'un expert dans la phase de la conception.

IV.4.1. Le système flou :

Un système flou est composé de quatre composants principaux ; la fuzzification, la base des règles, le mécanisme d'inférence et la défuzzification. Les ensembles flous sont caractérisés par leurs fonctions d'appartenance. Le moteur d'inférence flou effectue une décision à partir d'ensembles flous dans l'espace d'entrée à des ensembles flous dans l'espace de sortie en basant sur des règles floues.
Il existe de nombreux choix et configurations pour chaque opérateur dans un système flou. Par conséquent, un système flou particulier pourrait être une combinaison de ces configurations.

Habituellement, les entrées du système flou sont associées aux conditions, et les sorties sont associées à la conséquence. Les règles suivantes sont considérées dans la conception de système flou où la base consiste en une collection de règles Si-Alors de type Mamdani dans lesquelles les entrées-sorties sont représentés par des variables linguistiques.

$$R^r(y_j): Si\ x_1 est\ I_1^1\ et\ x_2 est\ I_2^4\ et\ \dots$$
$$Or\ Si\ x_1 est\ I_1^2\ et\ x_2 est\ I_2^5\ et\ \dots$$
$$Or\ \dots$$
$$Alors\ y_j est\ O_j^r\ ,$$

où $x = [x_1, x_2, \dots, x_h]^T$ et $y = [y_1, y_2, \dots, y_l]^T$ sont les vecteurs entrées-sorties du système flou, respectivement. I_i^r $(i = 1 \dots h)$ et O_j^r $(j = 1 \dots l)$ sont des variables linguistiques d'ensembles flous dans les sous-espaces I_i et O_j, décrits par leurs fonctions d'appartenance $\mu_{I_i^r}(x_i)$ et $\mu_{O_j^r}(y_j)$ $(r = 1,2, \dots R)$ avec R est le nombre des règles floues.

En utilisant la base des règles linguistique Tab.4.2, à chaque période de supervision T, le système flou reçoit l'intégrale de la norme de l'erreur $\int_T \|E_M(j,t)\|$ associée à chaque

paramètre et la norme de la commande appliquée $\|\bar{\tau}(t)\|$ comme variables d'entrée. La sortie du système flou est le nouvel espace de recherche.

Tableau 4.2. Règles de système flou

$\int_T \|E(j,t)\|$ / $\|\bar{\tau}(t)\|$	Z	P
Z	Z	P
P	Z	N

Nous associons à chaque sortie $y(j,t)$ une entrée $x(j,t)$ telle que

$$y(j,t) = \left[y_{kx}, y_{ky}, y_{k\varphi}, y_{\Lambda}, y_{R^{-1}}, y_{\sigma_{W_1}}, y_{\sigma_{W_2}}, y_{\sigma_{W_3}}, y_{\alpha}\right]^T, \quad (4.16)$$

$$x(1,t) = \left[\int_T \|e_x(\tau)\|, \|\bar{\tau}(t)\|\right]^T, \quad (4.17)$$

$$x(2,t) = \left[\int_T \|e_y(\tau)\|, \|\bar{\tau}(t)\|\right]^T, \quad (4.18)$$

$$x(3,t) = \left[\int_T \|e_\varphi(\tau)\|, \|\bar{\tau}(t)\|\right]^T, \quad (4.19)$$

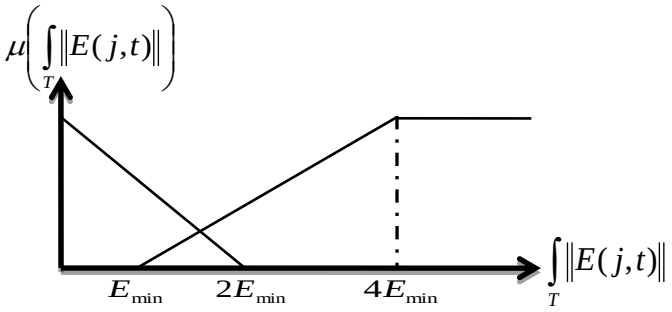
$$x(4,t) = \left[\int_T \|e(\tau)\|, \|\bar{\tau}(t)\|\right]^T, \quad (4.20)$$

$$x(5,t) = x(6,t) = x(7,t) = x(8,t) = x(9,t) = \left[\int_T \|r(\tau)\|, \|\bar{\tau}(t)\|\right]^T. \quad (4.21)$$

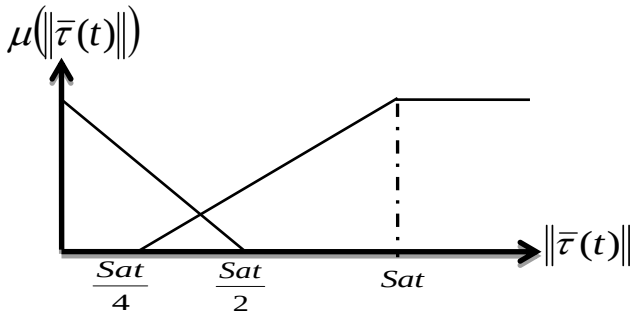
Le nombre de sous-ensembles pour chaque variable d'entrée ne sont pas forcément égaux. En général, le nombre de règles dépend essentiellement de la précision souhaitée.

Deux variables linguistiques pour les entrées sont définies comme (Z, P), et trois variables (N, Z, P) pour les sorties du système flou, où N=Négative, Z=Zéro et P=Positive.

Les fonctions d'appartenance pour les variables d'entrées et de sorties sont choisies de type triangulaires pour la variable linguistique Z et trapézoïdales pour N et P (figure.4.10), avec $E_{\min}$ est l'erreur min acceptable et Sat est la saturation de la commande



(a)



(b)

Figure.4.10. Fonctions d'appartenance pour les variables d'entrées.

En utilisant la fuzzification linguistique selon la figure 4.10, le mécanisme d'inférence Max-min selon le Tab.4.2 et la défuzzification par centre de gravité, la sortie du système flou peut être calculée par la relation suivante

$$y(j,t) = \frac{\mu_j^P(x_j) - \mu_j^N(x_j)}{\mu_j^P(x_j) + \mu_j^N(x_j) + \mu_j^Z(x_j)}, \tag{4.22}$$

avec $x_j = [\|E_M(j,t)\|, \|\bar{\tau}(t)\|]^T$.

IV.4.2. L'algorithme de colonies de fourmis modifié

Cet algorithme s'inspire de la nature et de son organisation. Son principe repose sur le comportement particulier des fourmis quand elles quittent leur fourmilière pour explorer leur environnement à la recherche de nourriture, elles finissent de crier des chemins attractifs qui se révèlent souvent être le plus court de passer de la fourmilière à une source de nourriture intéressante. Lorsqu'une fourmi se déplace d'une source de nourriture vers le nid, elle dépose relativement de la phéromone sur le sol en proportion de la qualité de la source trouvée. Cette

substance attire les autres fourmis de la colonie. Ainsi, lorsque plusieurs possibilités sont disponibles pour une fourmi de la colonie, elle se dirige avec une grande probabilité vers les directions marquées par des concentrations plus élevées de phéromone. De cette façon, les sources les plus exploitables seront favorisées par la colonie. Ce comportement constitue la base de la coopération conduisant à la résolution de notre problème en question.

Dans ce qui suit, l'algorithme est conçu en fonction de cette approche qui permet de trouver les valeurs optimales des paramètres de la loi de commande en temps réel. En incorporant le système flou, qui peut être considéré comme un superviseur afin d'augmenter l'espace de recherche et changer la zone explorée par les fourmis à chaque période T. Le mécanisme flou ne permet pas aux fourmis d'effectuer des explorations inutiles. Il permet en fonction de la fonction objective de plus ou moins diversifier et de plus ou moins intensifier la recherche.

Nous associons pour chaque vecteur des paramètres de la loi de commande K_1 et K_2 une colonie de fourmis M. Nous définissons les fourmis de la colonie M par $A_1, A_2, \dots, A_N$, où N est la taille de la population dans la colonie $M = 1,2$. L'algorithme AFA pour une colonie de fourmis est décrit dans le pseudocode suivant.

Procédure AFA
Étape 1: Générer la population initiale autour de l'origine,
Étape 2: Tant que $\int_T E_M(j,\tau) > E_{min}(j)$ ***faire***
Si $(t > T + kT)$ ***donc***
k = k + 1;
Pour *chaque paramètre* ***faire***
Updat-Zone = Système-Flou( );
Fin Pour ;
Fin Si ;
Étape 3: Pour *chaque fourmi* A_k ***faire***
Commande( );
Updat-phéromone( );
Fin Pour ;
Étape 4: *Mouvement( );*
Fin Tant que ;
Fin procédure.

Les étapes de base de l'algorithme sont les suivantes :

Étape 1: Dans l'initialisation, la population des fourmis partagent une zone autour de l'origine où les paramètres de la loi de commande sont définis comme des zéros ;(Ex ; l'ordre de 10^{-4}).

Étape 2: Après chaque période de supervision T pendant la trajectoire, nous calculons pour chaque paramètre $K(j,t)$ la nouvelle zone à explorer par la fourmi A_1, selon la formule suivante :

$$K_{A_1}(j,t) = K_{A_1}(j,t-T) + m_j y(j,t) f(j,t), \tag{4.23}$$

où m_j est le nombre du processus flou répété sur chaque paramètre, $y(j,t)$ est la sortie du système flou et $f(j,t)$ est la pondération de $y(j,t)$ qui représente la fonction proportionnelle à chaque paramètre dans la loi de commande telle que :

$$f_{k_x}(t) = \int_T \|e_x(t)\|, \tag{4.24}$$

$$f_{k_y}(t) = \int_T \|e_y(t)\|, \tag{4.25}$$

$$f_{k_\varphi}(t) = \int_T \|\sin(e_\varphi(t))\|, \tag{4.26}$$

$$f_\Lambda(t) = \int_T \|e(t)\|, \tag{4.27}$$

$$f_{R^{-1}}(t) = \int_T \|r(t)\|, \tag{4.28}$$

$$f_{\sigma_{W_1}}(t) = \int_T \|(\psi(\widehat{W}_2^T r_e) - \psi'(\widehat{W}_2^T r_e)\widehat{W}_2^T r_e) r^T\|, \tag{4.29}$$

$$f_{\sigma_{W_2}}(t) = \int_T \|r_e r^T \widehat{W}_1^T \psi'(\widehat{W}_2^T r_e)\|, \tag{4.30}$$

$$f_{\sigma_{W_3}}(t) = \int_T \left\|-\alpha\|r^T(t)\|\widehat{W}_3(t) + \text{diag}\left[\left(\text{sign}(r(t))r^T(t)\right)_{ii}\right]\right\|, \tag{4.31}$$

$$f_\alpha(t) = \int_T \left\|\|r^T(t)\|\widehat{W}_3(t)\right\|. \tag{4.32}$$

Ensuite, la nouvelle zone sera partagée sur la population des fourmis comme suit :

$$A_k: \quad K_{A_k}(j,t) = \frac{N-k+1}{N}\left(K_{A_1}(j,t) - K_{A_N}(j,t-T)\right) + K_{A_N}(j,t-T)(k = 2,3,\dots,N). \tag{4.33}$$

Étape 3:

1) dans la fonction *Commande( );* pour chaque fourmi dans une colonie, on utilise ses paramètres dans la loi de commande pendant une période dT qui représente le temps de réponse du système commandé (les actionneurs).
2) *Updat-phéromone( );* pour chaque position d'une fourmi, on associe une variable $p_k(t)$ appelée la phéromone artificielle. L'intensité de la phéromone est

proportionnelle à l'efficacité de cette position estimée par la fourmi A_k, variée avec une vitesse exponentielle calculée comme suit

$$p_k(t) = \exp\left(-a \int_{dT} \|E_M(t)\|\right), \quad (4.34)$$

avec a est une constante positive.

Étape 4: *Mouvement(*); Le déplacement de la fourmi A_k est basé sur la quantité de phéromone $p_k(t)$ dans sa position actuel et ainsi les phéromones $p_i(t)$ $(i = 1,2, \dots, N, i \neq k)$ associées aux autres fourmis dans la colonie M. Chaque position d'une fourmi A_i dans la colonie est considérée comme une position attractive pour la fourmi A_k. En effet, le déplacement admissible pour la fourmi A_k vers la fourmi $A_i (i = 1 \dots N, i \neq k)$ est calculé selon la règle de décision probabiliste suivante

$$P_k = \frac{p_k(t)}{\sum_i p_i(t)} \ (k = 1,2, \dots, N). \quad (4.35)$$

Ainsi, pour chaque fourmi $A_k (k = 1,2, .., N)$ son déplacement est calculé par le code suivant :

Si $P_k \leq P_r$ *(*$P_r = \text{rand} \in]0,1]$ *est une variable aléatoire)*

$K_{A_k}(j,t) = K_{A_k}(j, t - NdT) + L\frac{P_r}{P_k}\cos(\text{rand } 2\pi)$ *;// cos(rand* 2π*) un sens aléatoire*

Sinon

Si $p_i(t) > s$ *//(s : un seuil sert à éliminer la mauvaise solution ainsi d'éviter de tomber dans un minimum local)*

Si $p_k(t) < p_i(t)$ *//(Pour chaque fourmi dans la colonie, on calcule le déplacement //de la fourmi* A_k *vers la fourmi* A_i, $(i = 1 \dots N, i \neq k)$*)*

$D = K_{A_i}(j, t - NdT) - K_{A_k}(j, t - Ndt)$;

$K_{A_k}(j,t) = K_{A_k}(j, t - NdT) + L\frac{P_i}{P_k}\text{sign}(D)$ *;(la fourmi* A_k *se dirige vers* A_i*)*

Fin-Si ;

Fin-Si ;

Si $K_{A_k}(j,t) = K_{A_k}(j, t - NdT)$ *// S'il n'y a pas de mouvement, un déplacement local est //réalisé en fonction de l'erreur en question afin de l'améliorer*

Si $p_k(t) < s$

$K_{A_k}(j,t) = K_{A_k}(j, t - NdT) + L\|E_M(j,t)\|$;

Fin-Si ;

Fin-Si ;

Fin-Si ;

Le principe consiste à poser des fourmis autour de l'origine, et à les laisser découvrir graduellement les solutions et se déplacer d'une position à une autre dans la zone de recherche. C'est donc un algorithme qui repose sur la construction progressive de solutions.
La fourmi se déplace avec une probabilité proportionnelle à la phéromone dans une région, mais son déplacement est contrebalancé par la probabilité aléatoire. En atteignant une nouvelle position, la fourmi exploite les valeurs des paramètres associées à cette position en tant que paramètres de la loi de commande, et met à jour la phéromone à cette position. En fait, la diminution de l'intensité de la phéromone dépend de l'inefficacité de cette position (fonction de la qualité de l'erreur), de sorte que la phéromone associée à cette position s'évapore rapidement si les autres fourmis n'atteignent pas cette position. Ainsi, l'intensité de la phéromone dans une région peut augmenter en raison que les fourmis déposent de phéromone dans cette région ce qui favorise la convergence vers cette région. Par ailleurs, le déplacement aléatoire de temps en temps, met en œuvre une forme d'équilibre afin d'éviter une convergence trop rapide de l'algorithme vers une région sous-optimale, et favorise ainsi l'exploration de nouvelles zones dans l'espace de recherche. En effet, le défaut de maintenir cet équilibre conduit à une convergence trop rapide vers les minimums locaux (manque de diversification) ou à une exploration trop longue (manque d'intensification). Les paramètres responsables à cet équilibre sont a, s et P_r.

L'avantage de l'algorithme réside dans l'exploration globale et rapide de l'espace de recherche de façon en ligne avec la loi de commande, ce qui s'adapte avec une application en temps réel.
Finalement, le paramétrage de l'algorithme est défini par un nombre de paramètres simples à régler par l'utilisateur, à savoir, $M, N, E_{j,min}, a, Sat, L, s, dT$ et T.

M : le nombre de colonies de fourmis dépend de nombre de vecteur de paramètres de la loi de commande ; dans l'exemple de notre loi de commande, nous avons deux type de paramètres en relation avec les erreurs dans la boucle de commande ; l'erreur dans l'espace articulaire et l'erreur dans l'espace opérationnelle, donc deux colonies doit être utilisées dans l'algorithme,

N : Le nombre de fourmis ou la taille de la population dans une colonie. Ainsi, un nombre élevé favorise une meilleure exploration de l'espace de recherche, par contre, sa rapidité de convergence sera lente.

$E_{j,min}$: L'erreur minimale acceptable pour une application donnée ; par exemple pour une plateforme mobile, l'erreur acceptable sur le plan $\{X, Y\}$ peut être choisie comme $E_{x,y,min} = 1\ cm$,

a : Constante qui définit la vitesse exponentielle fonction de l'erreur en question, le choix de la valeur de a est imposé par la quantité de phéromone associée à l'erreur minimale acceptable,

Sat: La saturation des actionneurs,

$L(j,t)$: La résolution ou le pas de déplacement, peut être choisi comme la valeur moyenne pour chaque paramètre $L(j,t) = 1\% \ \frac{1}{N}\sum_i K_{A_i}(j,t)$, (4.36)

s : Un seuil sert à éliminer la mauvaise solution ainsi d'éviter de converger vers un minimum local,

dT : Cette période représente le temps de réponse des actionneurs à déterminer par l'expertise,

T : Dans chaque période de supervision T pendant la trajectoire, le système flou intervient.

Avec les paramètres s, a et P_r, l'intensification et la diversification sont garanties dans l'algorithme. D'une part, l'intensification permet de rechercher des solutions de meilleure qualité basées sur des solutions déjà trouvées, et d'autre part, la diversification met en place la stratégie pour explorer davantage l'espace de solutions et échapper aux minimums locaux. Ainsi, Le mécanisme flou ne permet pas aux fourmis d'effectuer des explorations inutiles. Il permet en fonction de la fonction à minimiser de plus ou moins diversifier et de plus ou moins intensifier la recherche en favorisant des zones de recherche attractives.

IV.5. Simulations et résultats expérimentaux :

IV.5.1. Simulations :

Le système exemple retenu pour la simulation est un robot mobile à deux roues comme le montre la Figure.4.11 Le système est soumis à des contraintes non-holonomes et des perturbations externes. Nous nous intéressons au robot mobile de type uni-cycle correspondant à notre application dans l'objectif de mieux comparer l'étude par simulation et

l'étude expérimentale présentée dans la section suivante afin de valider ainsi le caractère temps réel de l'algorithme AFA.

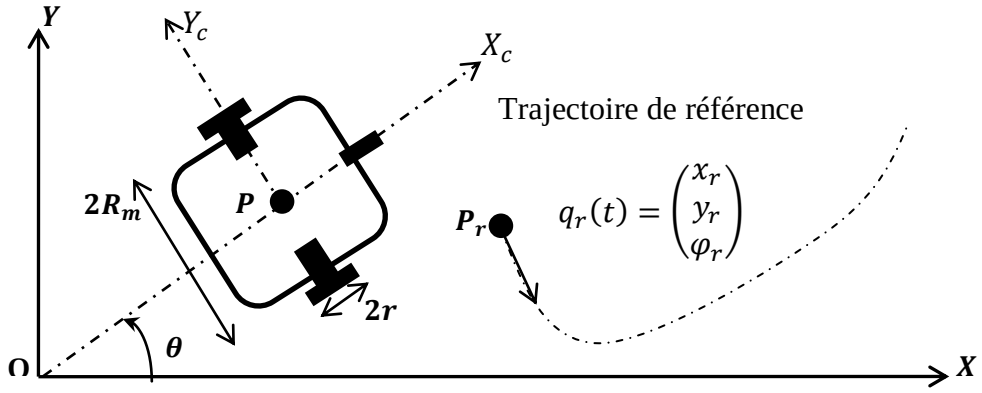


Figure.4.11 Robot mobile uni-cycle non holonome.

Les équations qui décrivent le mouvement du système sont définies par

$$m\ddot{x} = \frac{1}{r}(\tau_R + \tau_L)\cos(\theta) - \lambda \sin(\theta) + \tau_{d1} \quad (4.37)$$

$$m\ddot{y} = \frac{1}{r}(\tau_R + \tau_L)\sin(\theta) + \lambda \cos(\theta) + \tau_{d2} \quad (4.38)$$

$$I_w\ddot{\varphi} = \frac{R_m}{r}(\tau_R - \tau_L) + \tau_{d2} \quad (4.39)$$

$$v = RI + K\dot{q}_a \quad (4.40)$$

$$\tau = [\tau_R, \tau_L]^T = N\tau_{moteurs} \quad (4.41)$$

$$\dot{y}cos(\theta) - \dot{x}sin(\theta) = 0 \quad (4.42)$$

où x et y, θ représentent la position et l'orientation cartésienne du robot relativement à un repère fixe R (O,$\vec{x}$,$\vec{y}$,$\vec{z}$), m est la masse de la plateforme, I_w est le moment d'inertie, τ_R et τ_L sont les couples de commande générés par les motoréducteurs droite et gauche, respectivement, $R_m = 0.15\ [m], r_m = 0.12\ [m]$, τ_{di} $(i = 1,2,3)$ sont les perturbations externes, $K = \text{diag}(0.04) \in \mathbb{R}^{2\times2}$ sont les constantes c-f-e-m, $I \in \mathbb{R}^2$ est le courant d'induit, la résistance $R = \text{diag}(1)[\Omega] \in \mathbb{R}^{2\times2}$, $K\dot{q}_a \in \mathbb{R}^2$ est la force contre électromotrice, $v \in \mathbb{R}^2$ est la tension de commande et $N = \text{diag}(50) \in \mathbb{R}^{2\times2}$ est le rapport de réduction.

Les valeurs nominales et perturbées des paramètres du système sont choisies comme suit ; $m = 10 + 2\sin\left(\frac{\pi}{5}t\right)\ [Kg]$, $I_w = 0.5 + 0.1\cos\left(\frac{\pi}{5}t\right)\ [Kgm^2]$ et les perturbations externes $\tau_{d1} = \exp\big(-(2t-5)\big)\,\dot{x}(t)$, $\tau_{d2} = \exp\big(-(2t-5)\big)\,\dot{y}(t)$, $\tau_{d3} = 0.1\exp\big(-(2t-5)\big)\,\dot{\theta}(t)$

pour $0 \leq t < 5$ et $\tau_{d1} = 40 \sin\left(\frac{2\pi}{5}t\right)\dot{x}(t)$,$\tau_{d2} = 40 \cos\left(\frac{2\pi}{5}t\right)\dot{y}(t)$, $\tau_{d3} = 4\,sin\left(\frac{2\pi}{5}t\right)\dot{\theta}(t)$ pour $5 \leq t$.

La loi de commande a été implémentée avec un système neuronal à 8 neurones dans la couche cachée où $r_e = [e^T, r^T, \dot{q}_a^T, \theta, \dot{\theta}] \in \mathbb{R}^{8\times1}$. Les valeurs des poids du réseaux sont initialisés comme zéros et le paramétrage de l'algorithme AFA est choisi comme suit ; $M = 2$, $N = 5$, $a_1 = 200$, $a_2 = 400$, $s = 0.5$, $dT = 0.1$ sec, $T = 1$ sec, $Sat = 24$ V, $e_{x,min} = 0.03$, $e_{y,min} = 0.03$, $e_{\theta,min} = 0.05$, $e_{min} = \frac{0.5}{4}$ et $r_{min} = 0.2 + \Lambda e_{min}$.

Pour observer et comparer les résultats de la simulation plus facilement, nous choisissons deux types de trajectoires de référence pour les simulations: l'une est une trajectoire en ligne droite et l'autre est une trajectoire circulaire, en raison que ce deux types composent la plupart des trajectoires reconnues dans les applications réelles des robots mobiles. La première trajectoire de référence est définie par $q_r(t) = \left[0.2t, 0.2t, \frac{\pi}{4}\right]^T$ avec des différentes configurations initiales; $q_1(0) = \left[0, 0.5, -\frac{\pi}{4}\right]^T$, $q_2(0) = \left[-0.5, -0.5, \frac{\pi}{2}\right]^T$, $q_3(0) = \left[-0.5, 1, -\frac{\pi}{6}\right]^T$ et $q_4(0) = \left[0.5, -1, \frac{\pi}{3}\right]^T$.

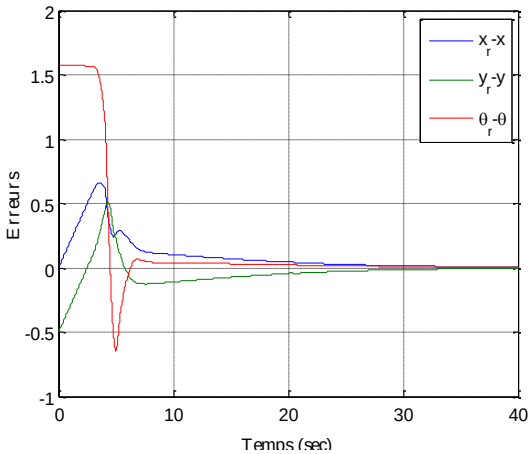
La trajectoire de référence circulaire est définie par $x_r(t) = -2\cos\left(\frac{2\pi}{18.8}t\right)$ et $y_r(t) = -2\sin\left(\frac{2\pi}{9.42}t\right)$ dans le plan $\{X, Y\}$ comme une forme de huit qui est similaire aux lemniscat de Bernoulli où la configuration initiale réelle du robot mobile est $q_1(0) = \left[-2, 0, -\frac{\pi}{2}\right]^T$. La trajectoire de référence est appliquée périodiquement jusqu'à ce que le processus commence à apprendre et que la performance convergente soit atteinte. De plus, afin de mieux vérifier l'efficacité et la supériorité de la stratégie de contrôle proposée avec différentes trajectoires de référence, les paramètres AFA sont identiques à ceux des simulations de la trajectoire en ligne droite.

D'autre part, indépendamment des différentes approches dans la littérature qui traitent le problème de réglage des paramètres d'une loi de commande de façon en ligne, à base de méta-heuristique ou non, les valeurs initiales des paramètres, ou de la population initiale dans une approche méta-heuristique sont d'abord déterminées habituellement par la méthode d'Essais-erreur empirique, afin d'obtenir une évolution plus rapide vers les meilleures valeurs de réglage et éviter ainsi d'endommager le système avec des faux paramètres dans les essais expérimentaux.

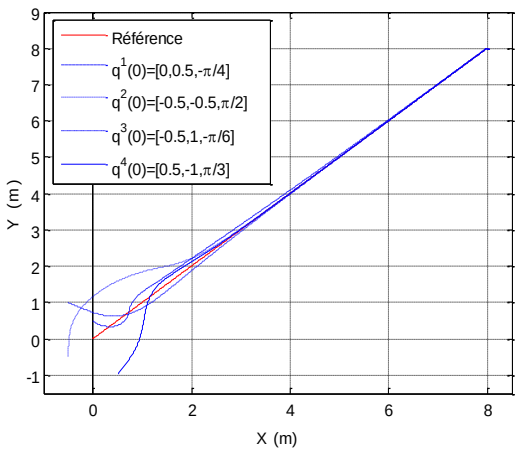
Par contre, notre contrôleur intelligent proposé découvre pour la première fois le système dynamique en temps réel, dans lequel, l'algorithme AFA est introduit dans le contrôleur de façon en ligne pour une exploration globale de l'espace de recherche, ce qui permet d'ajuster les paramètres de la loi de commande avec la moindre connaissance sur le système, où les valeurs initiales des paramètres du contrôleur sont définies comme des zéros. Par conséquent, le résultat ne sera pas sensible aux paramètres initiaux et aucune information préalable et des conditions initiales ne sont requises, où le paramétrage de l'algorithme AFA est très simple à définir par l'utilisateur.

Les résultats de suivi de trajectoire sont affichés sur les figures.4.12-13 (a-b), et les sorties de l'algorithme AFA sont présentées sur les figures.4.12-13 (e-f). Le délai de démarrage du robot observé à l'instant $t = 2$ sec, est dû particulièrement à l'initialisation des différents paramètres de la loi de commande autour de zéro. Graduellement de l'origine, avec une exploration rapide de l'espace de solution, les fourmis convergent vers des solutions différentes et trouvent des valeurs optimales relativement à la trajectoire désirée, ce qui montre le caractère adaptatif de l'algorithme face à un changement de la trajectoire.

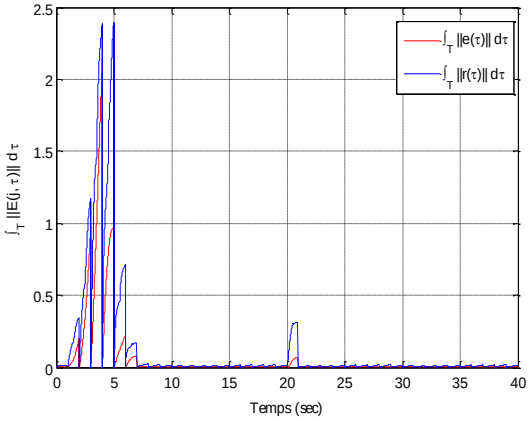
Évidemment, d'après les figures, la sortie réelle (Fig.4.12,4.13-(b)) converge rapidement vers la trajectoire de référence et un suivi satisfaisant et des performances convergentes sont atteintes après seulement t = 10 sec et t = 20 sec environ, pour la trajectoire linéaire et la trajectoire circulaire, respectivement, et où les effets dus à la dynamique inconnue et aux perturbations externes ont été compensés, tout en comparant avec les déférentes technique dans la littérature qui traitent la même problématique. Il est clairement que la vitesse de convergence de l'algorithme est incomparable avec la méthode Essais-erreur qui demande beaucoup de temps pour trouver des gains satisfaisants; des fois une journée complète.



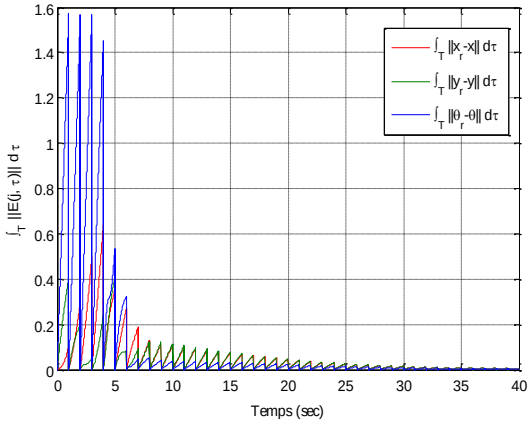
(a)



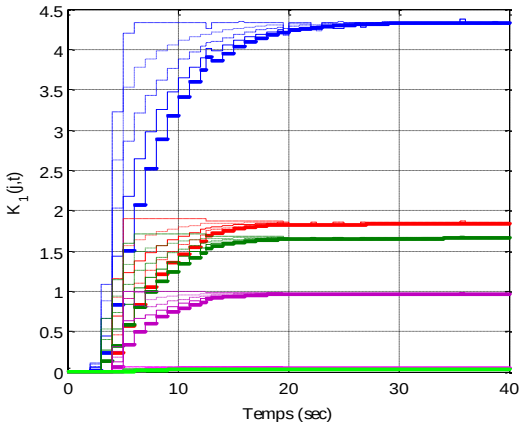
(b)



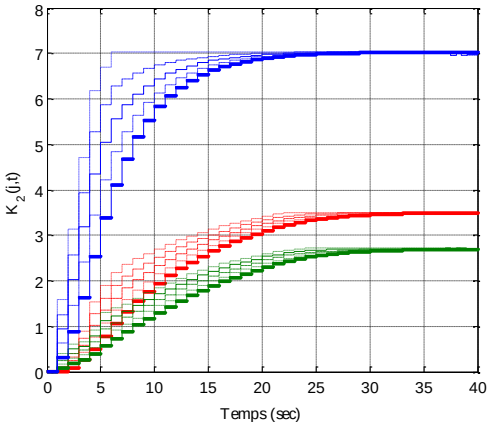
(c)



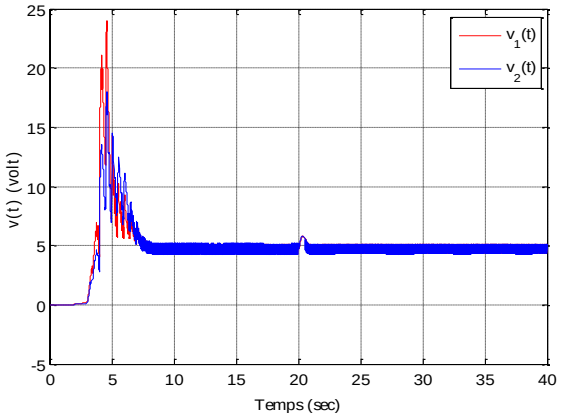
(d)



(e)

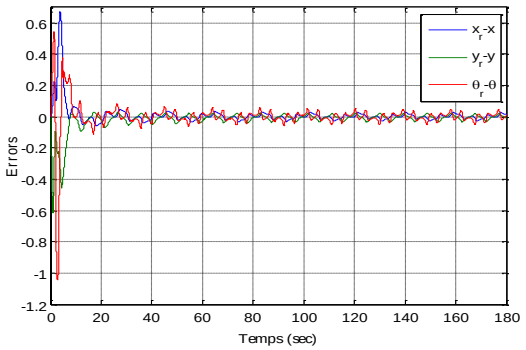


(f)

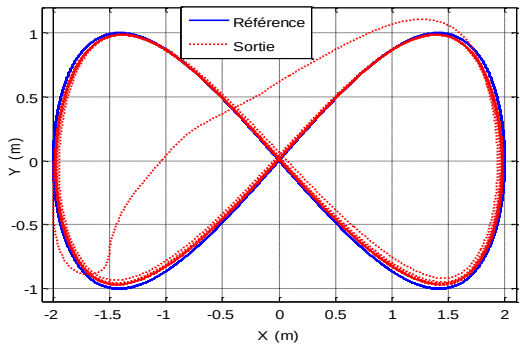


(g)

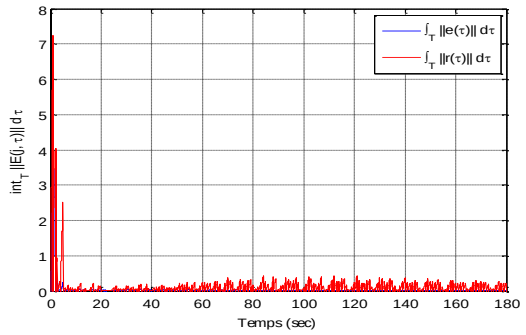
Figure.4.12. (a) L'erreur de suivi de trajectoire avec $q_1(0) = \left[0, 0.5, -\frac{\pi}{4}\right]^T$.(b) La trajectoire de sortie avec des configurations initiales multiples. (c, d) La variation des entrées E_1 et E_2. (e, f) Les paramètres K_1 et K_2 de l'algorithme AFA. (g) Le signal de commande appliqué.



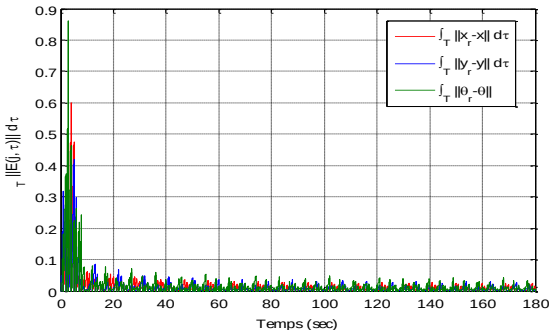
(a)



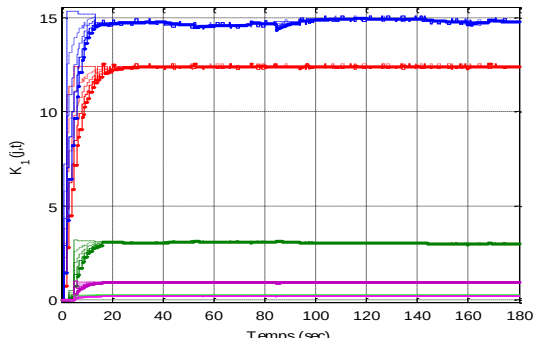
(b)



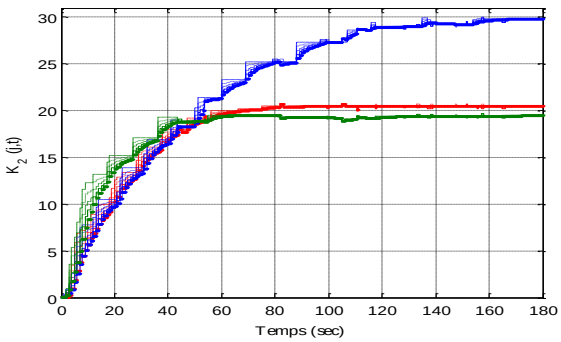
(c)



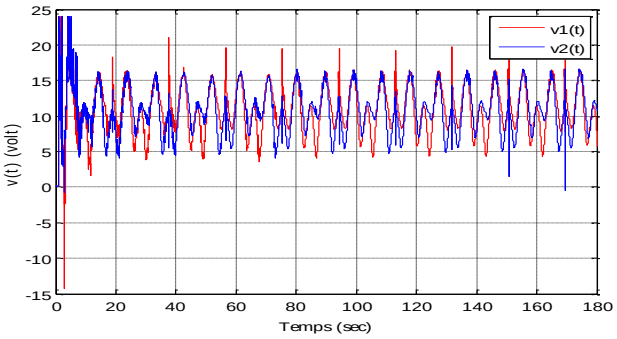
(d)



(e)



(f)



(g)

Figure.4.13. (a) L'erreur de suivi de trajectoire. (b) La trajectoire de sortie dans le plan {X,Y}. (c, d) La variation des entrées E_1 et E_2. (e, f) Les paramètres K_1 et K_2 de l'algorithme AFA. (g) Le signal de commande appliqué.

Par ailleurs, pour vérifier davantage l'efficacité de l'algorithme proposé, nous présentons quelques comparaisons de simulation où la performance et les propriétés de notre algorithme AFA sont comparées avec une approche hors ligne basée sur un algorithme génétique multi-objectif NSGA-II proposé dans **[62],** qui traite le réglage des paramètres de la loi de commande PID appliquée à un robot manipulateur (Figure.4.14).

En effet, nous avons considéré le même système robotique, et la même trajectoire de référence à suivre par le manipulateur. La trajectoire désirée est décrit par un polynôme d'interpolation cubique comme le montre la Figure.4.15 (b). En utilisant notre algorithme AFA, toujours avec le même principe, la trajectoire de référence est appliquée quatre fois de manière périodique jusqu'à ce que le processus commence à apprendre et que la performance imposée soit atteinte.

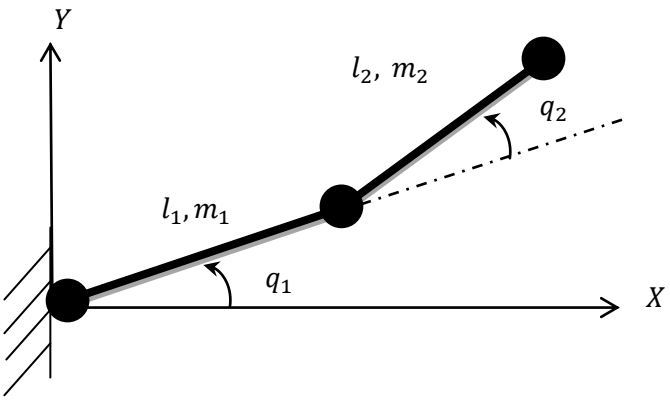
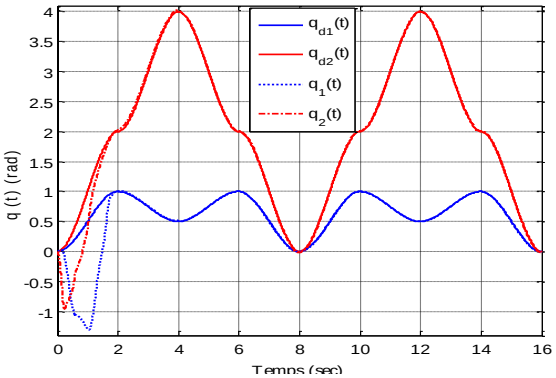


Figure. 4.14. Bras manipulateur à deux degrés de liberté.

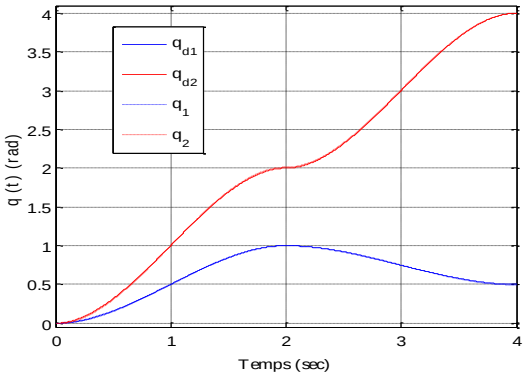
La loi de commande PID peut être présenté par $\tau = K_d r(t) + K_i \int e(t)dt$ où $r(t) = \dot{e}(t) + \Lambda e(t)$. Alors les paramètres à régler sont les gains K_d, Λ et K_i qui minimisent à la fois la fonction objective représentée par les erreurs de position $f_1 = \sum_{t=0}^{4}(|e_1(t)| + |e_2(t)|)$ des deux articulations du robot, et la variation des couples de commande $f_2 = \sum_{t=0}^{4}(|\Delta\tau_1(t)| + |\Delta\tau_2(t)|)$. Le paramétrage de l'algorithme AFA est choisi comme suit ; $M = 1$, $N = 5$, $a_1 = 200$, $s = 0.5$, $dT = 0.1$ sec, $T = 0.2$ sec, $Sat = 5$ V, $e_{min} = \frac{0.01}{2}$, $r_{min} = 1$ et $\left(\int_{t=0}^{4} e(t)dt\right)_{min} = \int_{t=0}^{4} e_{min}dt$.

La figure.4.15 (a-e) montre le suivi de trajectoire par le robot manipulateur par rapport à la trajectoire de référence, l'erreur de suivi de chaque variable et les couples appliqués, respectivement. Les résultats de l'algorithme AFA sont présentés à la figure.4.15 (f).

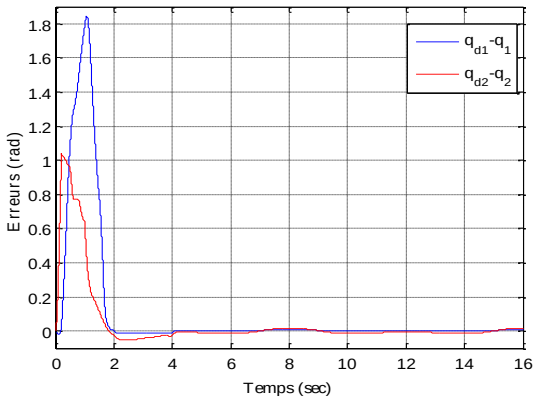
Évidemment, d'après les figures, en utilisant l'algorithme de réglage proposé, la sortie du système converge vers l'état désiré de manière en ligne. On peut constater qu'après seulement 8 sec et avec une population de cinq fourmis, les performances convergentes sont atteintes tout en comparant avec l'approche hors ligne NSGA-II. En effet, l'algorithme NSGA-II adopte une taille de population de 40 individus et le cycle évolutif est arrêté après 200 générations ce qui simule une durée de recherche de 200×4 sec $= 800$ sec. D'autre part, le signal de commande dans **[62]** a montré plus d'oscillations et d'ondulations au début $t \in [0, 1.5 \text{ sec}]$, et de plus, les fonctions objectives à minimisées dans le cas de l'algorithme AFA proposé sont $f_1 = 13.5$ et $f_2 = 9.8$ en appliquant les valeurs finales, par contre dans le cas de l'algorithme NSGA-II, $f_1 = 15.54$ et $f_2 = 13.44$, ce qui montre clairement la supériorité de l'algorithme AFA proposé. Les figures.4.15 (b, e) montrent les résultats de suivi de trajectoire et la commande appliquée en utilisant les valeurs finales de l'algorithme AFA.



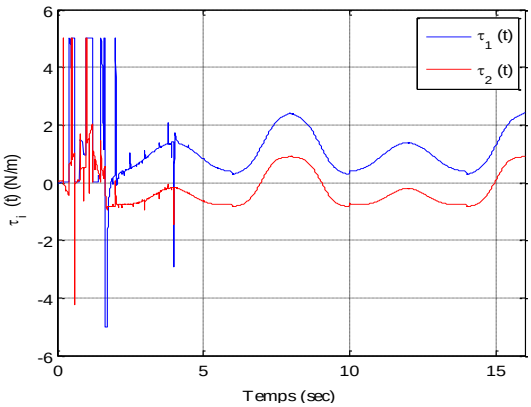
(a)



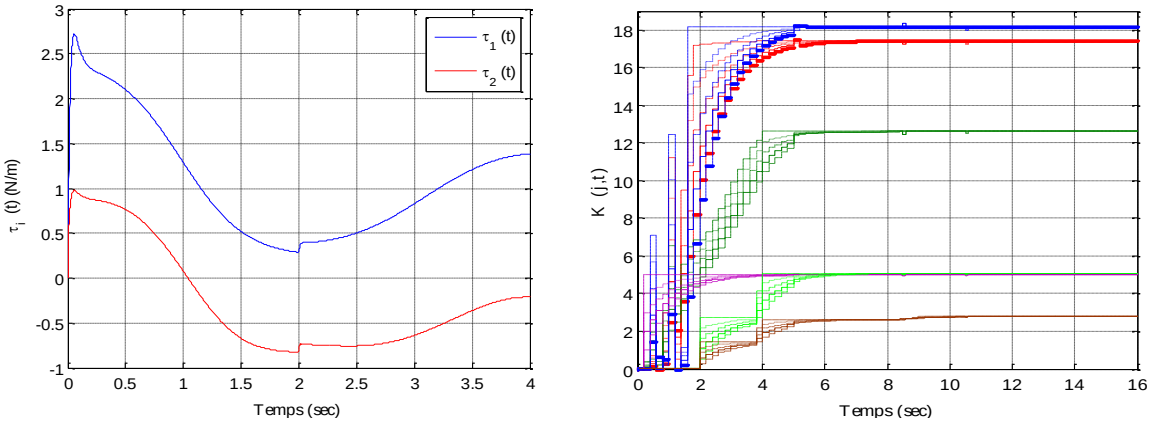
(b)



(c)



(d)



(e) (f)

Figure.4.15. (a) la trajectoire de sortie périodique. (b) La trajectoire de sortie avec les paramètres finals de l'algorithme AFA. (c) L'erreur de suivi de la trajectoire périodique. (d) Le signal de commande appliqué pour la trajectoire périodique. (e) Le signal de commande appliqué avec les paramètres finals de l'algorithme AFA. (f) Les paramètres K de la commande PID de l'algorithme AFA.

IV.5.2. Résultats expérimentaux :

Dans cette section, des résultats expérimentaux sont présentés afin de valider et démontrer la performance de l'algorithme proposé dans un problème de suivi d'une trajectoire donnée. L'objectif de cette étude expérimentale trouve son intérêt dans la démonstration de l'aspect temps réel de l'algorithme AFA qui est implémenté en ligne avec la loi de commande neuro optimale robuste.

Le manipulateur mobile expérimental présenté au chapitre 1, est composé d'une plateforme de type uni-cycle porteuse d'un manipulateur à deux articulations. La plateforme mobile non-holonome doit suivre une trajectoire de référence circulaire spécifiée dans l'espace cartésien définie par ; $x_r(t) = -0.75\cos\left(\frac{2\pi}{27}t\right)$ et $y_r(t) = -0.75\sin\left(\frac{2\pi}{27}t\right)$ dans le plan $\{X, Y\}$, où la configuration initial du robot est $q_1(0) = \left[-0.75, 0, -\frac{\pi}{2}\right]^T$.

Tout comme le cas de la simulation, la trajectoire de référence est appliquée périodiquement jusqu'à ce que la performance convergente soit atteinte, et puis une trajectoire sous la forme d'une ligne droite est considérée à $t \geq 60.75$ sec, avec $x_r(t) = -0.75\frac{2\pi}{27}(t - 60.75)$ et $y_r(t) = -0.75$ dans le plan $\{X, Y\}$. D'autre part, pour le manipulateur, nous avons

sélectionné une trajectoire de référence pour les deux articulations, qui est décrite par un polynôme quintique avec une continuité de la vitesse et de l'accélération afin d'assurer un mouvement souple au démarrage et à l'arrêt.

$$q_r(t) = \begin{cases} \frac{\pi}{2}\left[10\left(\frac{t}{1.5}\right)^3 - 15\left(\frac{t}{1.5}\right)^4 + 6\left(\frac{t}{1.5}\right)^5\right]; 0 \le t \le 1.5, \\ \frac{\pi}{2} - \frac{\pi}{2}\left[10\left(\frac{t-1.5}{1.5}\right)^3 - 15\left(\frac{t-1.5}{1.5}\right)^4 + 6\left(\frac{t-1.5}{1.5}\right)^5\right]; 1.5 \le t \le 3. \end{cases} \quad (4.43)$$

Les déférents paramètres adaptatifs du réseau de neurones et les bornes supérieures estimées de la commande robuste sont montrées sur la figure 4.16. Une convergence stable après certainement une période transitoire, autour des valeurs bornées spécifiques pour chaque sous-système garantissant le suivi de trajectoire. Les figures.4.17-18 (a-b) montrent la trajectoire réelle du robot mobile et le bras manipulateur, respectivement, où les sorties de l'algorithme AFA pour le robot mobile et le bras manipulateur sont illustrées à la Fig.4.17(e-f) et Fig.4.18(d), respectivement. Comme le cas des simulations, un délai de démarrage du robot est observé à t=1 sec environ, en raison que les fourmis sont initialement commencent de l'origine autour de zéro.

Progressivement, la performance convergente en temps réel est obtenue après seulement 10 sec pour la plateforme mobile et 4 sec pour le manipulateur, avec des résultats de suivi de trajectoire satisfaisants pour les deux sous-systèmes. Le mécanisme flou ne permet pas aux fourmis d'effectuer des explorations inutiles. Il permet en fonction des résultats observés, de plus ou moins diversifier et de plus ou moins intensifier la recherche. De plus, la variation des paramètres remarquée à l'instant $t = 60.75$ sur la figure.4.17 (f) montre ainsi le caractère adaptatif de l'algorithme face à un changement dans la trajectoire désirée qui a été remarquée aussi dans les simulations précédentes.

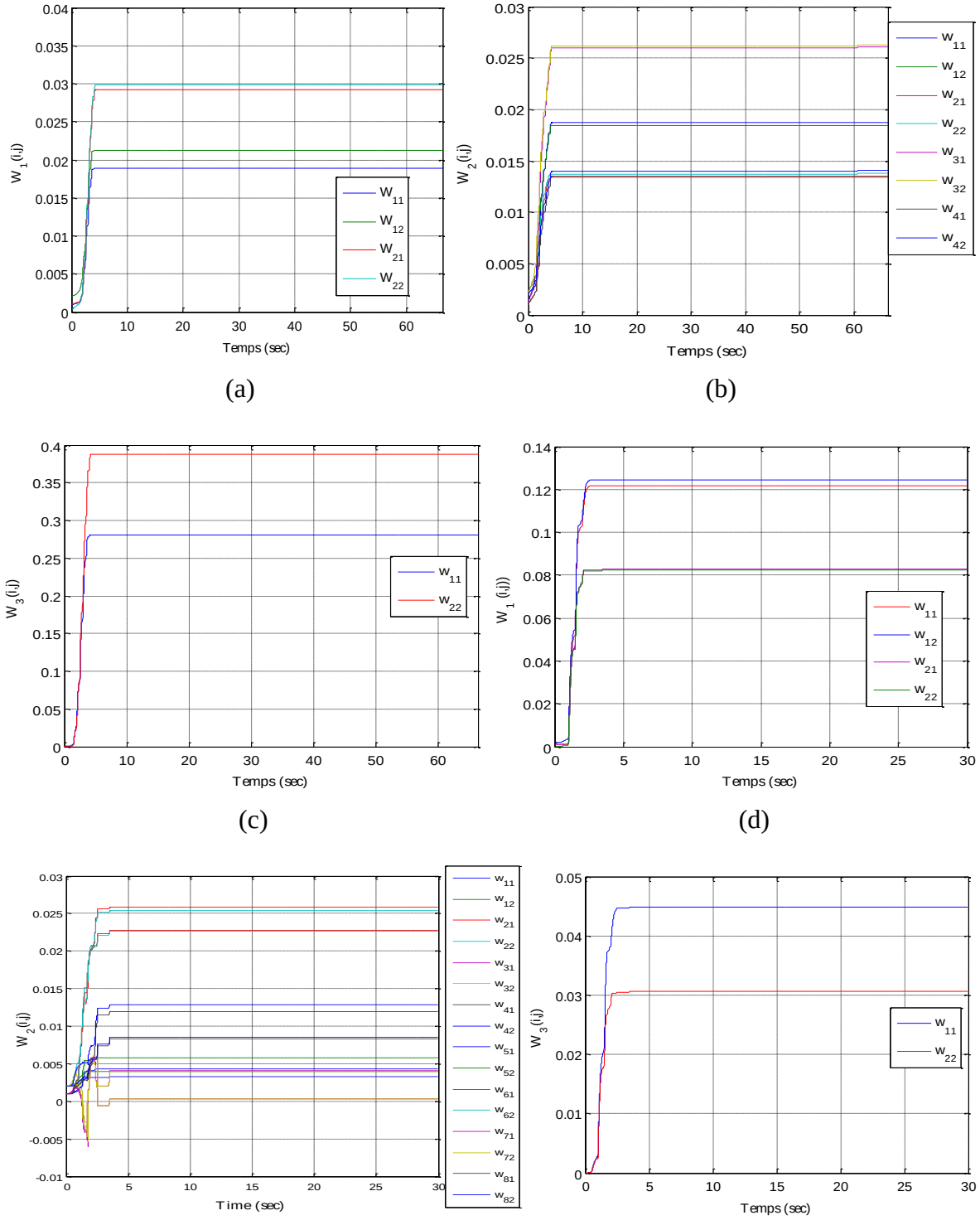


Figure.4.16. (a, b) Les poids du réseau de neurones W_1 et W_2 pour la plateforme mobile. (c) Les paramètres estimés W_3 de la commande robuste pour la plateforme mobile. (d, e) Les poids du réseau de neurones W_1 et W_2 pour le manipulateur. (f) Les paramètres estimés W_3 de la commande robuste pour le manipulateur.

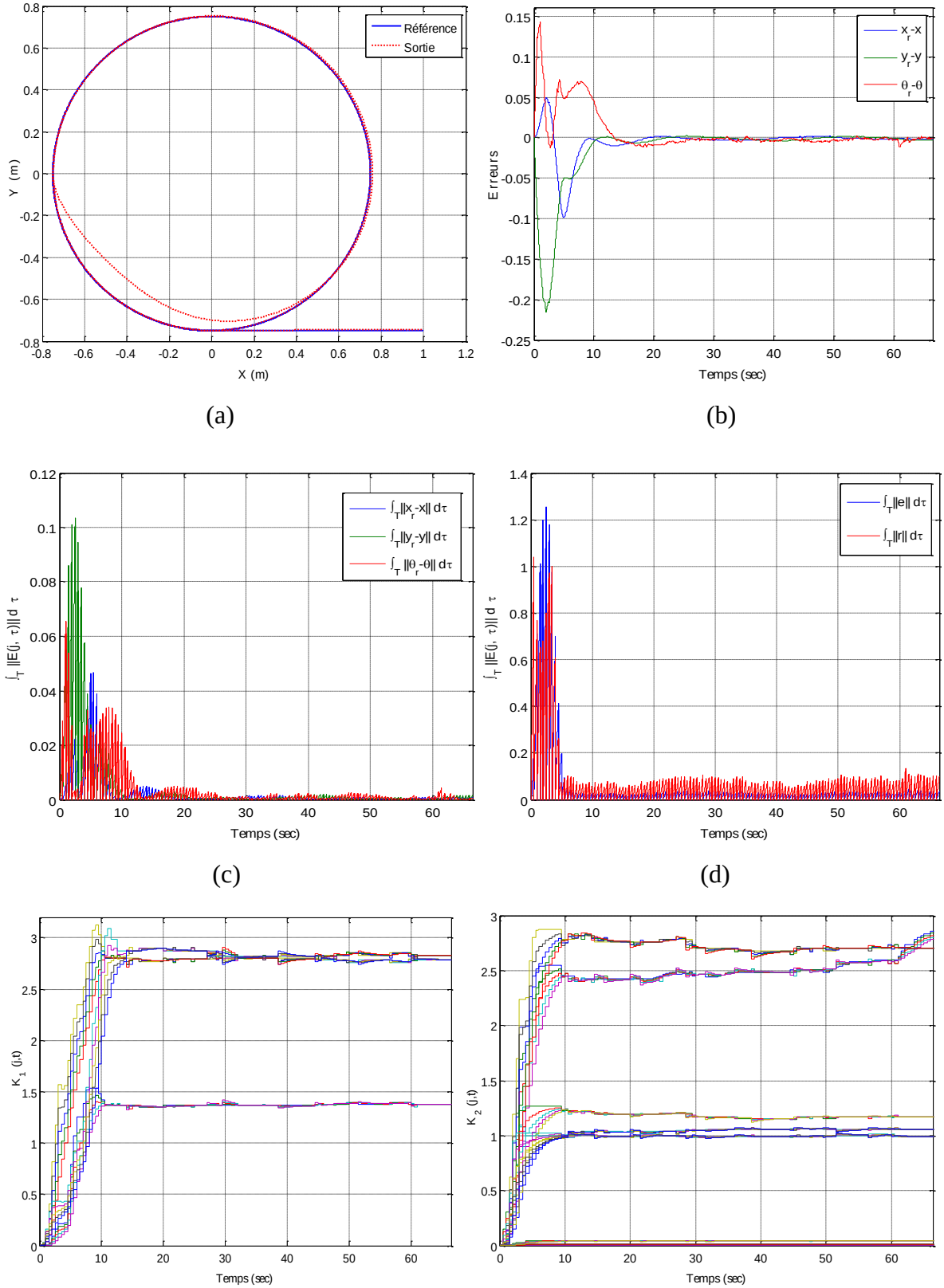
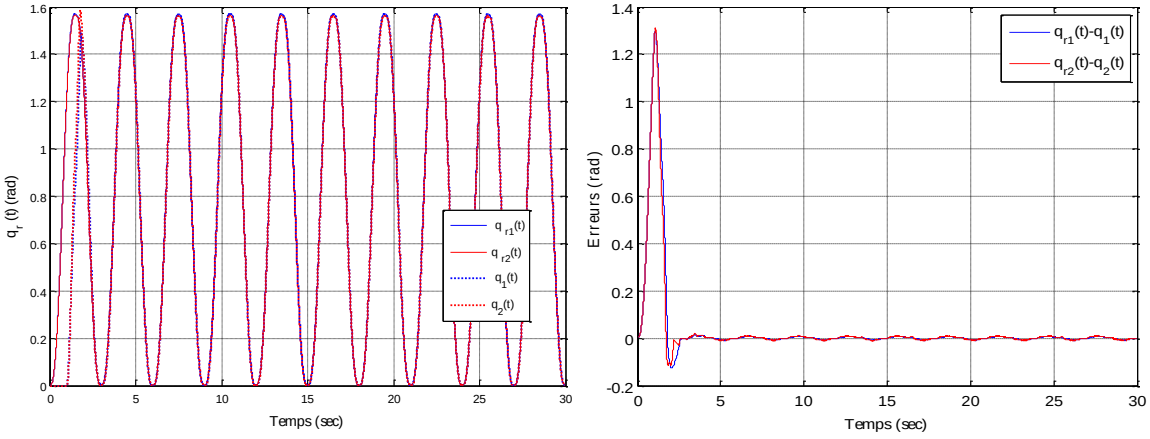
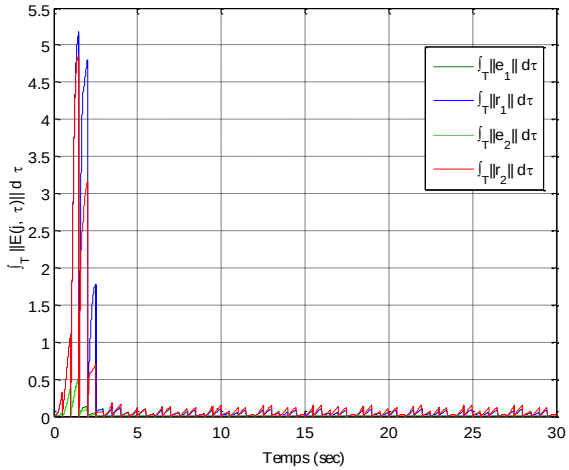


Figure.4.17. (a) la trajectoire de sortie périodique du robot mobile. (b) L'erreur de suivi de trajectoire du robot mobile. (c, d) La variation des entrées E_1 et E_2 pour le robot mobile. (e, f) Les paramètres K_1 et K_2 de l'algorithme AFA pour le robot mobile.

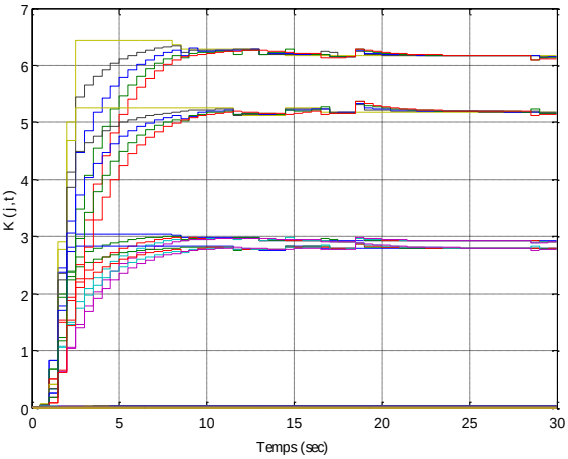


(a)

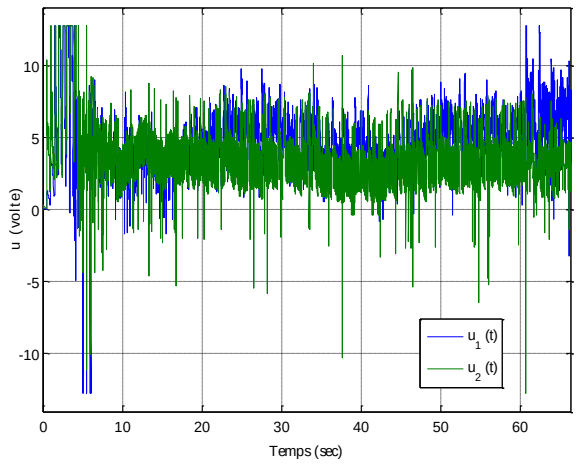
(b)



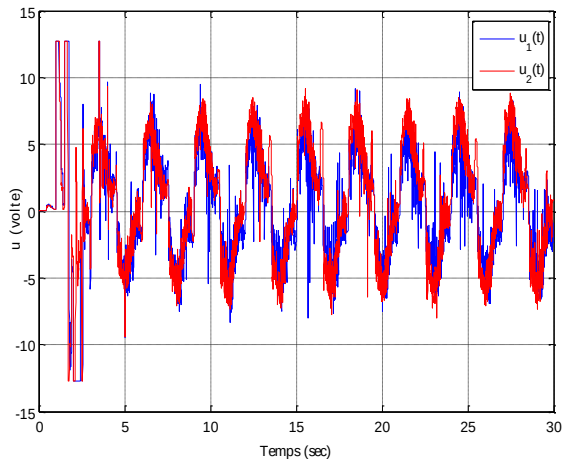
(c)



(d)



(e)

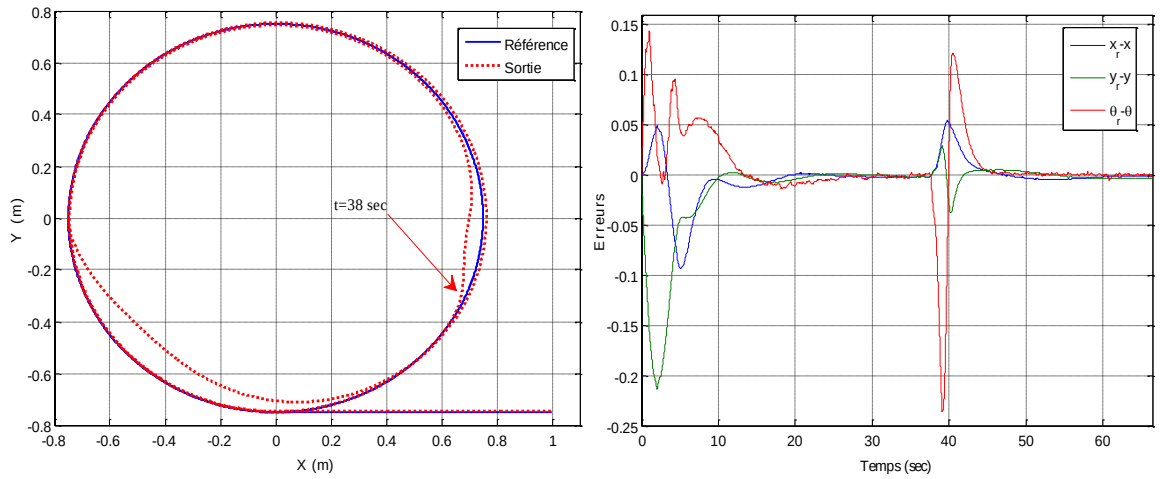


(f)

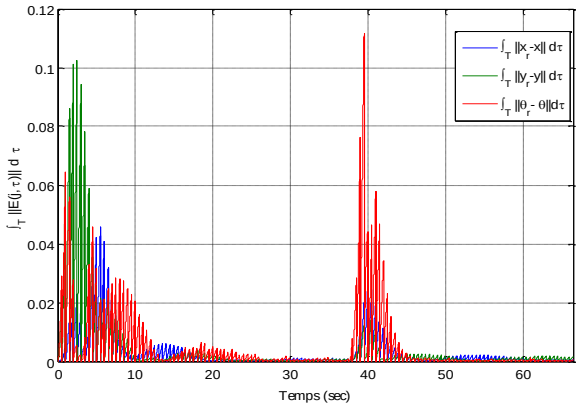
Figure.4.18. (a) la trajectoire de sortie périodique du robot manipulateur. (b) L'erreur de suivi de trajectoire du robot manipulateur. (c) La variation de l'entrée E_1 pour le robot manipulateur. (d) Les paramètres K_1 de l'algorithme AFA pour le robot manipulateur. (e,f) Les signaux de commande pour le robot mobile et le manipulateur, respectivement.

Ensuite, l'objectif d'une seconde expérience est de montrer la capacité de la loi de commande à compenser une perturbation non négligeable appliquée sur un seul sous-système à la fois, et dans des instants différentes durant la trajectoire, et ainsi, d'étudier l'efficacité et la robustesse du contrôleur proposé et le fonctionnement de la loi robuste adaptative conjointement avec l'algorithme de réglage AFA. Donc, à l'instant $t = 38\,\text{sec}$ durant la trajectoire réelle du robot, une force de freinage transitoire a été appliquée sur la roue gauche de la plateforme mobile pendant environ deux secondes, ce qui représente une perturbation externe qui simule une variation brusque ou un obstacle imprévue sur le sol. De la même façon, à l'instant $t = 17\,\text{sec}$ un freinage externe transitoire a été appliqué sur la première articulation du manipulateur.

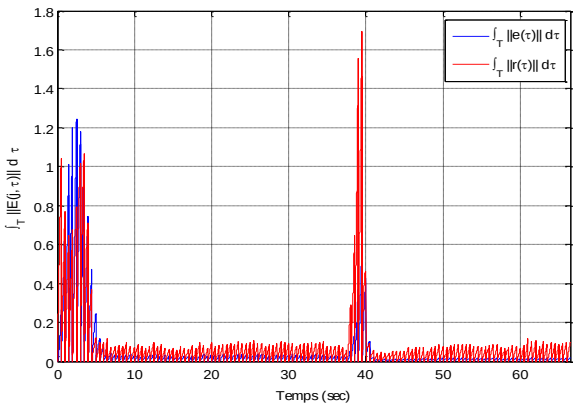
Les Figues. 4.19-20 représentent le comportement réel du système perturbé et les sorties de l'algorithme AFA pour le robot mobile et le bras manipulateur, respectivement. Nous pouvons remarquer clairement sur les figures, qu'après une certaine période transitoire due à la perturbation externe, afin de rattraper le retard en suivi de trajectoire, les différents paramètres associés à la loi de commande neuro robuste adaptative et à l'algorithme de réglage AFA convergent vers des valeurs assurant la compensation de la perturbation sur le système en trois secondes environs. De plus, le comportement du contrôleur face à un changement imprévu dans le système prouve aussi l'aspect adaptatif de l'algorithme AFA, et en particulier montre ainsi l'estimation multiple de la commande robuste adaptative, où chaque borne supérieure locale reste seulement sensible à sa perturbation locale (Figures 4.19-g, 4.20-e).



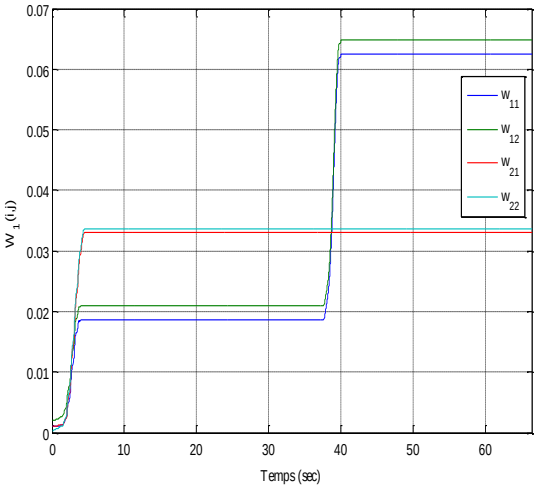
(a) (b)


Temps (sec)

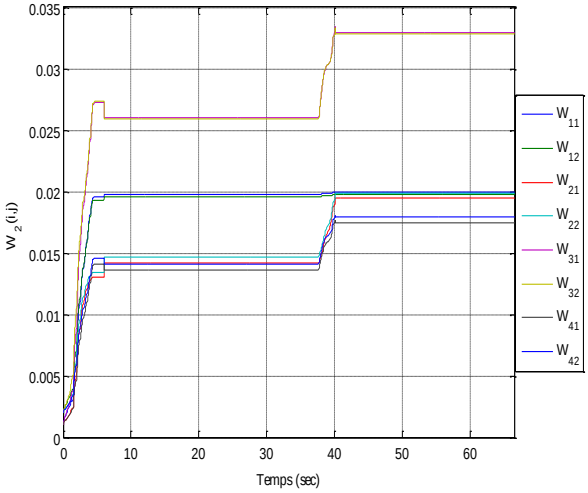
(c)


Temps (sec)

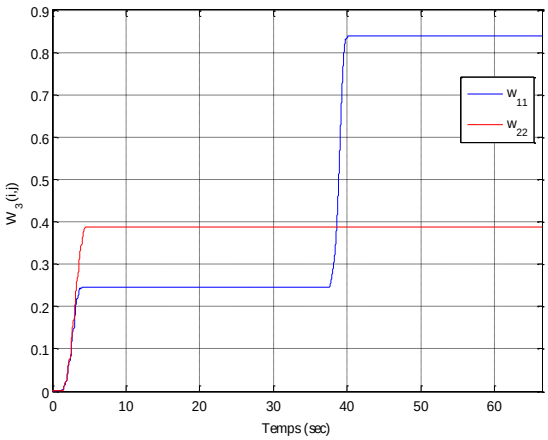
(d)


W11
W12
W21
W22
Temps (sec)

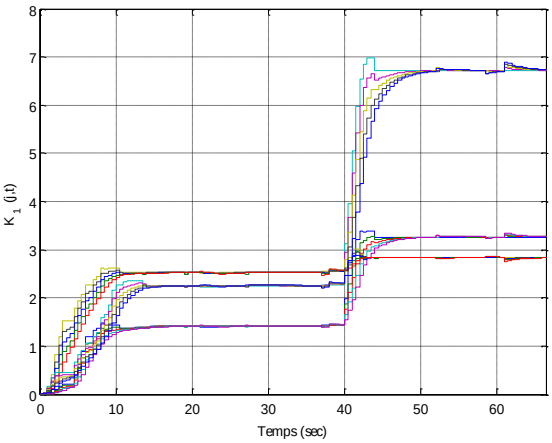
(e)


W11
W12
W21
W22
W31
W32
W41
W42
Temps (sec)

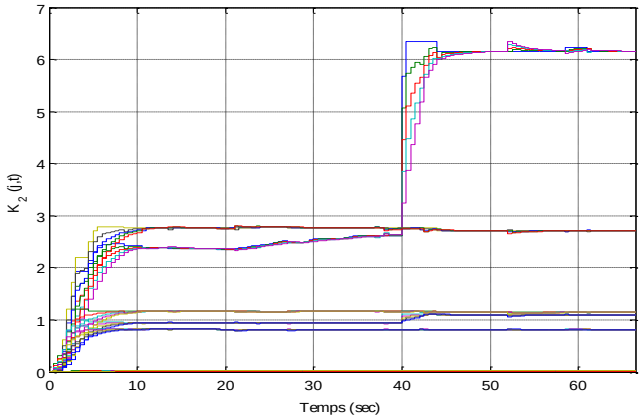
(f)


W11
W22
Temps (sec)

(g)

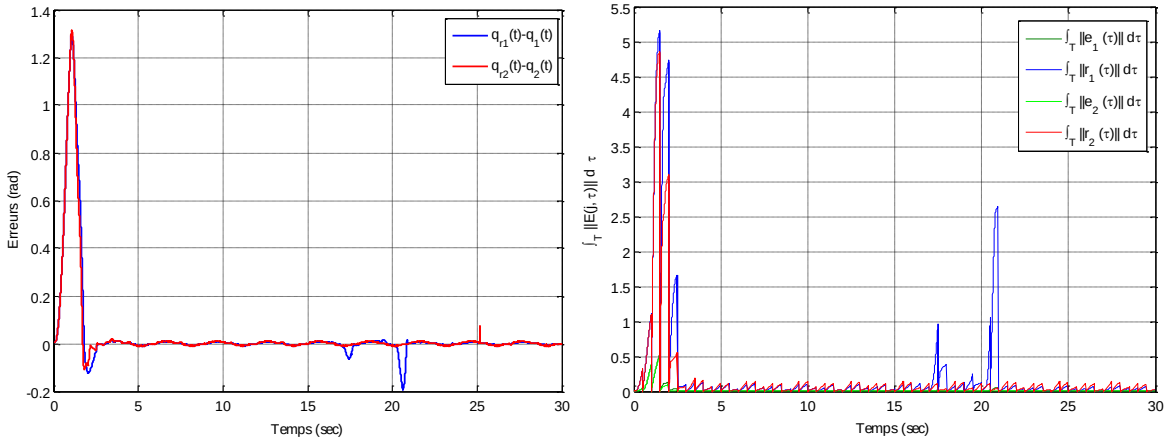

Temps (sec)

(h)

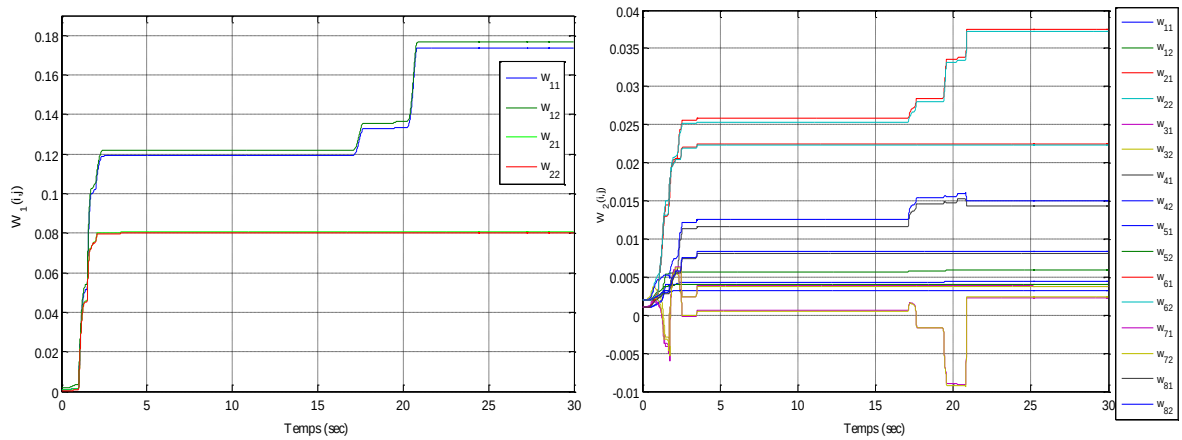


(i)

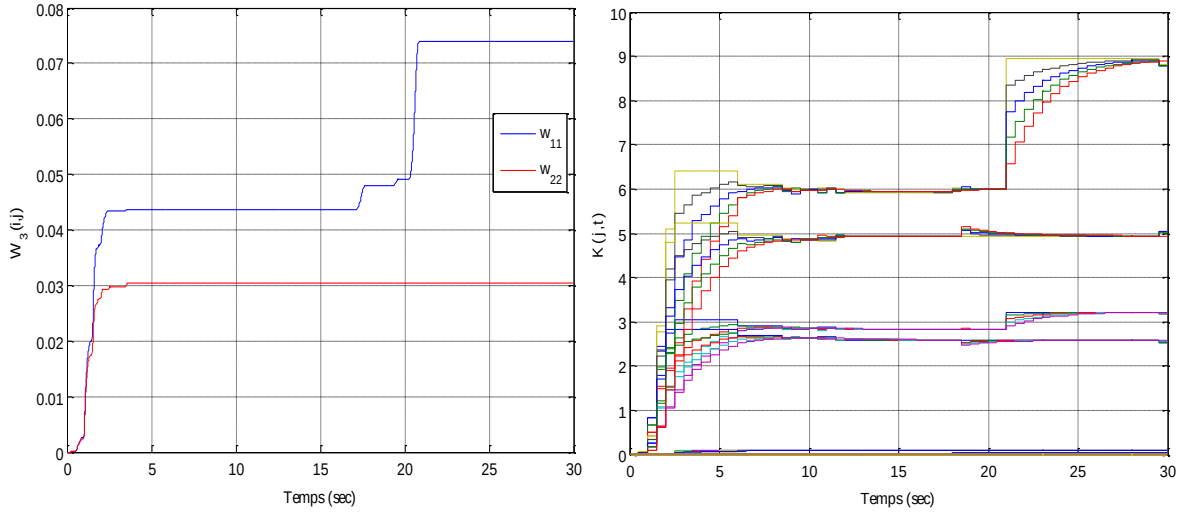
Figure.4.19. (a) la trajectoire de sortie du robot mobile avec perturbation externe. (b) L'erreur de suivi de trajectoire avec perturbation externe. (c, d) La variation des entrées E_1 et E_2 pour le robot mobile. (e, f, g) Les poids du réseau de neurones et les bornes estimées de la commande robuste, respectivement, pour le robot mobile. (h, i) Les paramètres K_1 et K_2 de l'algorithme AFA pour le robot mobile.



(a) (b)



(c) (d)



(e) (f)

Figure.4.20. (a) L'erreur de suivi de trajectoire avec perturbation externe sur le bras manipulateur. (b) La variation de l'entrée E_1 pour le bras manipulateur. (c, d, e) Les poids du réseau de neurones et les bornes estimées de la commande robuste, respectivement, pour le bras manipulateur. (f) Les paramètres K_1 de l'algorithme AFA pour le bras manipulateur.

Ainsi, l'efficacité du système de contrôle proposé est vérifiée et illustrée par ces résultats expérimentaux. Les résultats expérimentaux et ceux de simulation ont beaucoup de similitudes, notamment le comportement de l'algorithme AFA dont l'allure qui confirme le caractère adaptatif en ligne et la convergence rapide en assurant une précision dans le cadre de la commande de suivi de trajectoire

De toute évidence, le contrôleur proposé conjointement avec l'algorithme d'optimisation garantie une stabilité améliorée et une précision de suivi de la trajectoire temporelle du robot, y compris la dynamique non modélisée, les perturbations externes, et sans négliger la dynamique des actionneurs, où la saturation est également prise en compte dans l'algorithme AFA.

IV.6. Conclusion :

Nous avons repondu dans le cadre de ce chapitre, à la dernière contrainte imposée sur la loi de commande dans notre problématique citée dans le chapitre 1, à savoir, la conception d'un algorithme de réglage autonome des différents paramètres d'une loi de commande, et qui

dispose la capacité de calcul en temps réel et une souplesse de la mise en ligne avec la loi de commande. Par soucis de simplicité, l'algorithme a été développé principalement pour la loi de commande neuro optimale robuste adaptative présentée dans le chapitre 4. Évidemment, la conception de l'algorithme se généralise facilement en fait à n'importe quelles lois de commande sans modifications profondes, tout comme le cas de la commande PID dans la partie simulations.

Prenons l'exemple de la commande PID adaptative connue dans la littérature, où son algorithme d'adaptation permet de fournir une approche pour l'ajustement des paramètres du régulateur en temps réel afin de maintenir des performances désirées lorsque la dynamique linéaire du procédé change dans le temps. Toutefois, la loi d'adaptation de ce mode de contrôle a besoin également d'un ajustement de ces différents gains qui influent directement sur l'efficacité et le mode de fonctionnement de l'algorithme d'adaptation. Ainsi, un algorithme ordinaire d'ajustement des paramètres nécessite un ajustement lui aussi.

Afin de montrer le fonctionnement de l'algorithme proposé, nous avons proposé une validation par simulations mais aussi expérimentale sur un robot manipulateur mobile non holonome réel. Pour diverses raisons, nous n'avons pas eu l'occasion de tester d'autres algorithmes de méta-heuristiques. Dans les différents cas des résultats issus des simulations ou d'essais réels, l'algorithme de réglage autonome a pu justifier son fonctionnement en ligne avec la loi de commande en question et notamment son caractère temps réel et adaptatif. De plus, l'algorithme proposé améliore non seulement les performances de la loi de commande, mais aussi réduit le degré de difficulté dans le réglage d'une loi de commande en temps réel, et élimine ainsi l'utilisation de la méthode empirique Essais-erreur.

Finalement, nous pouvons dire que ce type d'algorithme peut être appelé une boîte noire intelligente dans un sens qui dépasse le simple rôle d'une méthode d'optimisation ordinaire.

Conclusion générale et Perspectives

Les travaux réalisés dans le cadre de cette thèse répondent à la volonté de développer un algorithme de contrôle intelligent pour la commande des robots manipulateurs mobiles en investiguant les déférentes techniques proposées dans la littérature. Ainsi, nous avons adopté une approche hybride intégrant une partie de l'automatique classique et avancée afin de créer une structure de contrôle efficace qui peut faire face à un certain niveau de complexité.
Notre logique de recherche repose sur la démarche de garder à l'esprit que le système de contrôle doit respecter des contraintes imposées lors de la mise en œuvre de l'architecture de contrôle. Chaque étude présentée dans cette thèse a suivi une représentation systématique et des validations lors de simulations et d'essais expérimentaux.

Dans le premier chapitre de cette thèse, nous avons d'abord abordé les études menées dans le cadre de la commande des robots manipulateurs mobiles selon un classement thématique, afin de situer notre travail dans la littérature. L'objectif étant, d'une part d'examiner l'état de l'art et d'autre part de s'inspirer des approches traitées dans la littérature afin de proposer une nouvelle architecture la plus appropriée pour notre problématique. Cette famille de système dynamiques est apparu pour des missions qui demandent à la fois des capacités de déplacement et de manipulation, et la problématique sur la commande des robots mobiles s'ouvre aussi vers de nombreux autres systèmes dynamiques, tels que ; les véhicules, les drones, les engins sous-marins, etc.

L'objectif de cette thèse trouve son intérêt dans la conception et l'exécution en temps réel d'un algorithme de commande autonome plus générique, possède des avantages intéressants dont la facilité d'implémentation avec la moindre connaissance sur le système à commander, et ne nécessite pas le savoir-faire d'un utilisateur qualifié pour leur implémentation et leur paramétrage, où il suffit d'avoir une simple vision externe du comportement du système (nombre de variables d'état, le temps de réponse des sous-systèmes, l'erreur en question acceptable pour une application donnée…etc.).

Dans le troisième et le quatrième chapitre, nous avons présenté notre architecture de contrôle proposée qui atteint l'objectif imposé. Elle comprend principalement deux niveaux; un

mécanisme basé sur le système flou et un algorithme modifié de métaheuristique pour le réglage autonome des différents paramètres d'une loi de commande, et qui dispose la capacité de calcul en temps réel et une souplesse de la mise en ligne avec la loi de commande ; le deuxième niveau dédié à la loi de commande en choisissant une approche hybride qui combine une partie de la commande optimale, la commande neuronale adaptative, et la commande robuste adaptative.

Ainsi, notre stratégie peut être considérée non seulement comme une autre loi pour commander avec une certaine robustesse un système dynamique inconnu, mais comme une extension considérable qui couvre le cas le plus général de l'implémentation d'une loi de commande et leur paramétrage.

L'algorithme de commande repose sur une stratégie donnant structurellement une priorité importante à la phase d'ajustement de leurs paramètres de manière autonome. Dans ce sens, l'algorithme dépasse le simple rôle de commande et d'optimisation, en prenant en compte la contrainte de l'inconnu sur le système, la rapidité en temps réel, l'aspect autonome robuste et adaptatif de l'algorithme, et la contrainte sur la saturation des actionneurs. Ainsi, un autre intérêt majeur de notre approche réside dans le fait que la synthèse de l'algorithme s'étend sans modification majeure de façon plus ou moins directe selon le système considéré.

Par ailleurs, afin que notre étude dans cette thèse ait un sens, il était primordiale de réaliser des résultats expérimentaux sur un robot réel, et il ne fallait pas de rester au stade de la simulation tant que les contraintes imposées s'articulent sur l'aspect temps réel de l'algorithme de contrôle. C'est pour cette raison que la condition d'avoir un robot réel avec une carte de contrôle convenable a été prise en considération durant nos travaux de recherche.

Notre algorithme a été conçu et validé par des résultats expérimentaux et des simulations. Le système étudié qui correspond à ce de notre application temps réel est un choix que nous faisons non pas pour des raisons de limitation théorique de nos approches, et la plupart des résultats se généralisent en fait à d'autres systèmes robotiques. Afin de s'approcher des conditions expérimentales, le robot exemple retenu pour la simulation est ainsi un manipulateur mobile de type uni-cycle.

De ce fait, nous avons tenté de présenter les avantages et les capacités offertes par cet algorithme de contrôle dans le problème de suivi d'une trajectoire de référence réalisable lorsque le robot est soumis à une perturbation externe. Dans les différents études issus des simulations ou d'essais réels, l'algorithme de réglage autonome a pu justifier son fonctionnement en ligne avec la loi de commande et notamment son caractère temps réel et adaptatif. De plus, l'algorithme proposé améliore non seulement les performances de la loi de commande, mais aussi réduit le degré de difficulté dans le réglage d'une loi de commande en temps réel, et élimine ainsi l'utilisation de la méthode empirique Essais-erreur.

Perspectives :

En fin, nos travaux de thèse ont permis d'ouvrir le champ à des perspectives variées que nous pourrions mener dans le futur sur la thématique de recherche abordé. L'axe de recherche qui peut être envisagé pour la continuité de nos travaux est la navigation des robots manipulateurs mobiles aériens. Le défi que pose ce type de robot s'articule sur le fait d'ajouter un bras manipulateur à un robot aérien pose des problèmes avant tout sur la stabilité et la robustesse. En vol, le système de contrôle doit s'adapter en temps réel et anticiper l'impact que le mouvement du bras va avoir sur la dynamique de l'ensemble notamment dans le cas d'un contact physique avec l'environnement.

En effet, nous pouvons penser ainsi, à deux sous-axes de recherches qui apparaissent à l'issue de cette vision, à savoir, le planificateur des tâches et le contrôle par retour extéroceptifs. Le premier peut donc recouvrir le niveau décisionnel. D'autre part, les capteurs extéroceptifs capables de fournir des informations, soit sur l'environnement du robot, soit sur la situation de l'effecteur dans cet environnement. Percevoir l'environnement est l'une des facultés qui caractérisent l'homme et bien d'autres organismes vivants. Pour augmenter l'autonomie du robot en environnement structuré ou non structuré, des recherches est donc nécessaires sur les capteurs et les traitements associés.

L'objectif ultime est de mettre en place un contrôleur qui pourrait fonctionner avec une autonomie intelligente en temps réel, ce qui nous permettra de soulever et de résoudre d'autres problématiques théoriques et expérimentales, ainsi d'éprouver nos algorithmes de contrôle où le robot manipulateur aérien sera l'épreuve idéale.

BIBLIOGRAPHIE

[1] B. Thuilot, Contribution à la modélisation et à la commande de robots mobiles à roues. Thèse en mathématique et automatique, Ecole Nationale Supérieure des Mines de Paris, 1995.

[2] W. Khalil, Modeling, identification and control of robots. Butterworth-Heinemann, 2004.

[3] R. Murray, Z. Li et S. Sastry, A Mathematical Introduction to Robotic Manipulation. CRC Press, 1994.

[4] S. Cubero, Industrial Robotics Theory, Modelling and Control, pro literatur Verlag Robert Mayer-Scholz, 2007.

[5] R. Siegwart et I. Nourbakhsh, Introduction to Autonomous Mobile Robots, The MIT Press Cambridge, Massachusetts London, England, 2004.

[6] J.C. Latombe, Robot Motion Planning, Springer Science & Business Media, 1991.

[7] R. Brooks, Planning Collision- Free Motions for Pick-and-Place Operations, Robotics Researches, MIT press, pp. 5-38, 1983.

[8] S. Jalel, Optimisation de la navigation robotique, Thèse de l'Institut National Polytechnique de Toulouse (INP Toulouse), 2016.

[9] R. Kelly, V. Santibáñez et A. Loría, Control of robot manipulators in joint space, Springer-Verlag London Limited, 2005.

[10] W. Khalil, Commande des robots manipulateurs, Lavoisier, 2002.

[11] N. Munro et F. Lewis, Robot Manipulator Control Theory and Practice, Marcel Dekker, Inc.2nd, 2004.

[12] S. Tzafestas, Introduction to Mobile Robot Control, Elsevier Science, 2013.

[13] A. Meystel, Autonomous Mobile Robots: Vehicles with Cognitive Control, World Scientific, 1991.

[14] M Chadli et P. Borne, Multimodèles en automatique: outils avancés d'analyse et de synthèse. Hermès science publ.-Lavoisier. ISBN : 978-2-7462-3825-1. 2012.

[15] R. Johansson, Quadratic optimization of motion coordination and control, IEEE Trans. Autom. Control, vol. 35(11), pp.1197–1208, 1990.

[16] D. Dawson, M. Grabbe et F. Lewis, Optimal control of a modified computed-torque controller for a robot manipulator, Int. J. Robot. Autom., vol. 6 (2), pp. 161–165, 1991.

[17] Z. Shiller, H. Chang et V. Wong, The practical implementation of time-optimal control for robotic manipulators, Robot. Comput. Integr., Manuf., vol. 12 (1), pp. 29–39, 1996.

[18] I. Hussein et A. Bloch, Optimal control of underactuated nonholonomic mechanical systems, IEEE Trans. Autom. Control, vol. 53 (3), pp. 668–682, 2008.

[19] Y. Zhou, Z. Wang, Motion controller design of wheeled inverted pendulum with an input delay via optimal control theory, Optim. Theory Appl., vol. 168 (2), pp. 625–645, 2016.

[20] J.J. Slotine et W. Li, On the adaptive control of robotic manipulators, Internat. J. Robotics Res., vol. 6(3), pp. 49-59, 1987.

[21] M. Spong, On the robust control of robot manipulators, IEEE Trans. Automat. Control, vol. 37, pp. 1782-1786, 1992.

[22] G. Leitmann, On the efficiency of nonlinear control in uncertain linear system, J. Dyn. Measm. Control, vol. 102, pp. 95–102, 1981.

[23] D. Dawson, Z. Qu, et F. Lewis, Hybrid adaptive robust control for a robot manipulator, Int. J. Adaptive Control Signal Process, vol. 6, pp. 537-545. 1992.

[24] K. Koo et J.H. Kim, Robust control of robot manipulators with parametric uncertainty, IEEE Trans.Automat. Control, vol. 39(6), pp. 1230-1233, 1994.

[25] M. Zhihong et M. Palaniswami, Robust tracking Control for Rigid Robotic Manipulators, IEEE Trans.Automat. Control, vol. 39(1), 1994.

[26] M. Zhihong et D. Habibi, A robust adaptive sliding-mode control for rigid robotic manipulators with arbitrary bounded input disturbances, Journal of Intelligent and Robotic Systems, vol. 17, pp. 371-386, 1996.

[27] R. Burkan et I. Uzmay, Variable upper bounding approach for adaptive-robust control in robot control, Journal of Intelligent and Robotic Systems, vol. 37, pp. 427-442, 2003.

[28] S. Torres, J.A. Mendez, L. Acosta et V.M. Becerra, On improving the performance in robust controllers for robot manipulators with parametric disturbances, Control Engineering Practice, vol. 15, pp. 557-566, 2007.

[29] M. Zeinali et L. Notash, Adaptive sliding mode control with uncertainty estimator for robot manipulators, Mechanism and Machine Theory, col. 45, pp. 80-90, 2010.

[30] K. Shojaei et A.M. Shahri, Adaptive robust time varying control of uncertain nonholonomic robotic systems, IET Control Theory Appl., vol. 6 (1), pp. 90-102, 2012.

[31] K. Shojaei et A.M. Shahri, Output feedback tracking control of uncertain nonholonomic wheeled mobile robots: a dynamic surface control approach. IET Control Theory Appl., vol. 6(2), pp. 216-28, 2012.

[32] M. Boukattaya, M. Jallouli et T. Damak, On trajectory tracking control for nonholonomic mobile manipulators with dynamic uncertainties and external torque disturbances, Robotics and Autonomous Systems, vol. 60, pp. 1640-1647, 2012.

[33] Z. J. Yang, Y. Fukushima et P. Qin, Decentralized adaptive robust control of robot manipulators using disturbance observers, IEEE Transactions on Control Systems Technology, vol. 20(5), pp. 1357-1365, 2012.

[34] J. Peng, J. Yu et J. Wang, Robust adaptive tracking control for nonholonomic mobile manipulator with uncertainties. ISA Transactions, vol. 53(4), pp. 1035-1043, 2014.

[35] L. Xin, Q. Wang, J. She et Y. Li, Robust adaptive tracking control of wheeled mobile robot, Robotics and Autonomous Systems, vol. 78, pp.36-48, 2016.

[36] M. Chadli et P. Borne, Multiple Models Approach in Automation: Takagi-Sugeno Fuzzy Systems, Wiley-ISTE, ISBN: 978-1-84821-412-5. 2012.

[37] B. Kosko, Fuzzy System as Universal Approximators, IEEE trans. Computers, vol. 43(11), pp. 1329-1333, 1994.

[38] Z. Song, J. Yi, D. Zhao et X. Li, A computed torque controller for uncertain robotic manipulator systems: Fuzzy approach, Fuzzy Sets and Systems, vol. 154, pp. 208-226, 2005.

[39] B. Yoo et W. Ham, Adaptive control of robot manipulators using fuzzy compensator, IEEE trans. Fuzzy Systems, vol. 8, pp. 186-199, 2000.

[40] W. Wang et H. Lin, Fuzzy control design for the trajectory tracking on uncertain nonlinear systems, IEEE Trans. Fuzzy Syst., vol. 7 (1), pp. 53-62, 1999.

[41] C. Tseng, B. Chen et H. Uang, Fuzzy tracking control design for nonlinear dynamic systems via T-S fuzzy model, IEEE Trans. Fuzzy Syst., vol. 9 (3), pp. 381-392, 2001.

[42] S. Haykin, Neural Networks .A Comprehensive Foundation, Macmillan, New York, 1994.

[43] S. Spieckermann, S. Dull, S. Udluft, A. Hentschel et T. Runkler, Exploiting similarity in system identification tasks with recurrent neural networks, Neurocomputing, vol. 169 (2), pp. 343-349, 2015.

[44] A. Tutunji, Parametric system identification using neural networks, Applied Soft Computing, vol. 47, pp. 251-261, 2016.

[45] M. Hector, R. Ugalde, J. Carmona, Juan Reyes-Reyes, V. Alvarado et J. Mantilla, Computational cost improvement of neural network models in black box nonlinear system identification, Neurocomputing, vol. 166 (20), pp. 96-108, 2015.

[46] W. Dong et K.D. Kuhnert, Robust adaptive control of nonholonomic mobile robot with parameter and no parameter uncertainties, IEEE Trans. Robot., vol. 21, pp. 261-266, 2005.

[47] B.S. Park, S.J. Yoo, J.B. Park et Y.H. Choi, Adaptive neural sliding mode control of nonholonomic wheeled mobile robots with model uncertainty, IEEE Trans. Control Syst. Technol., vol. 17, pp. 207-214, 2009.

[48] Y. Kim, F. Lewis et D. Dawson, Intelligent optimal control of robotic manipulators using neural networks, Automatica, vol. 36 (9), pp. 1355-1364, 2000.

[49] Y. Kim et F. Lewis, Optimal design of CMAC neural-network controller for robot manipulators, IEEE Trans. Syst. Man Cybern., vol. 30 (1), pp. 22-31, 2000.

[50] T. Cheng, F. Lewis et M. Abu-Khalaf, A neural network solution for fixed-final time optimal control of nonlinear systems, Automatica, vol. 43 (3), pp. 482–490, 2007.

[51] T. Cheng, F. Lewis et M. Abu-Khalaf, Fixed-final-time-constrained optimal control of nonlinear systems using neural network HJB approach, IEEE Trans. Neural Netw., vol. 18 (6), pp. 1725-1737, 2007.

[52] F. Ornelas-Tellez, E. Sanchez et A. Loukianov, Discrete time neural inverse optimal control for nonlinear systems via passivation, IEEE Trans. Neural Netw. Learn. Syst., vol. 23 (8), pp. 1327-1339, 2012.

[53] T. Das, I.N. Kar et S. Chaudhury, Simple neuron-based adaptive controller for a nonholonomic mobile robot including actuator dynamics, Neurocomputing, vol. 69, pp. 2140-2151, 2006.

[54] Y. Zuo, Y.N. Wang, X.Z. Liu, S.X. Yang, L.H. Huang, X.R. Wu et Z.Y. Wang, Neural network robust control for a nonholonomic mobile robot including actuator dynamics, Int. J. Innovative Comput. Inf. Control, vol. 6, pp. 3437-3449, 2010.

[55] B.S. Park, S.J. Yoo, J.B. Park et Y.H. Choi, A simple adaptive control approach for trajectory tracking of electrically driven nonholonomic mobile robots, IEEE Trans. Control Syst. Technol., vol. 18, pp. 1199-1206, 2010.

[56] B.S. Park, J.B. Park et Y.H. Choi, Robust formation control of electrically driven nonholonomic mobile robots via sliding mode technique, Int. J. Control Autom. Syst., vol. 9, pp. 888-894, 2011.

[57] M.M. Fateh et A. Arab, Robust control of a wheeled mobile robot by voltage control strategy, Nonlinear Dynam. vol. 79, pp. 335-348, 2015.

[58] M. Homayounzad, M. Keshmiri et M. Ghobadi, A robust tracking controller for electrically driven robot manipulators: stability analysis and experiment, Int.J. Autom. Comput, 1vol. 2, pp. 83-92, 2015.

[59] G. Zhong, Y. Kobayashi, Y. Hoshino et T. Emaru, System modeling and tracking control of mobile manipulator subjected to dynamic interaction and uncertainty, Nonlinear Dynam., vol. 73, pp. 167–182, 2013.

[60] A. Zdesar, D. Dovzan et I. Skrjanc, Self-tuning of 2 DOF control based on evolving fuzzy model, Applied Soft Computing, vol. 19, pp. 403-418, 2014.

[61] A. Visioli, Fuzzy logic based set-point weight tuning of PID controllers, IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans, vol. 29 (6), 587-592, 1999.

[62] H. Hultmann Ayala et Leandro dos Santos Coelho, Tuning of PID controller based on a multiobjective genetic algorithm applied to a robotic manipulator, Expert Systems with Applications, vol. 39 (10), pp. 8968-8974, 2012.

[63] A. Rajasekhar, A. Abraham et M. Pant, A Hybrid differential artificial Bee Colony Algorithm based tuning of fractional order controller for permanent magnet synchronous motor drive, Int. J. Mach. Learn. & Cyber., vol. 5, pp. 327-337, 2014.

[64] M. Villarreal-Cervantes et J. Alvarez-Gallegos, On-line PID control tuning for a planar parallel robot using DE variants, Expert Systems with Applications, vol. 64 (1), pp. 444-454, 2016.

[65] T. Mac, C. Copot, D. Tran et R. Keyser, Heuristic approaches in robot path planning: A survey, Robotics and Autonomous Systems, vol. 86, pp. 13-28, 2016.

[66] S. Bououden, M. Chadli et F. Allouani, A New Approach for Fuzzy Predictive Adaptive Controller Design Using Particle Swarm Optimization Algorithm, Int. J. of Innovative Computing Information and Control, vol. 9 (9), pp. 3741-3758, 2013.

[67] S. Bououden, M. Chadli, H. Karimi. An ant colony optimization-based fuzzy predictive control approach for nonlinear processes. Information Sciences, vol. 299, pp. 143-158. 2015.

[68] D. Fister, Jr. Fister, I. Fister et R. Safaric, Parameter tuning of PID controller with reactive nature-inspired algorithms, Robotics and Autonomous Systems, vol. 84, pp. 64-75, 2016.

[69] S. Girma Tewolde et W. Sheng, Robot Path Integration in Manufacturing Processes: Genetic Algorithm Versus Ant Colony Optimization, IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans, vol. 38 (2), pp. 278.287, 2008.

[70] V. Roberge, M. Tarbouchi et G. Labonte, Comparison of Parallel Genetic Algorithm and Particle Swarm Optimization for Real-Time UAV Path Planning, IEEE Transactions on Industrial Informatics, vol. 9 (1), pp. 132-141, 2013.

[71] Kaushik Das Sharma, Amitava Chatterjee et Anjan Rakshit, A PSO. Lyapunov Hybrid Stable Adaptive Fuzzy Tracking Control Approach for Vision-Based Robot Navigation, IEEE Transactions on Instrumentation and Measurement, vol. 61 (7), pp. 1908.1914, 2012.

[72] A. Marco Contreras-Cruz, Victor Ayala-Ramirez et Uriel H. Hernandez-Belmonte, Mobile robot path planning using artificial bee colony and evolutionary programming, Applied Soft Computing, vol. 30, pp. 319-328, 2015.

[73] Hsu-Chih Huang, SoPC-Based Parallel ACO Algorithm and its Application to Optimal Motion Controller Design for Intelligent Omnidirectional Mobile Robots, IEEE Transactions on Industrial Informatics, vol. 9 (4), pp. 1828-1835, 2013.

[74] M. Renaud, Modélisation dynamique des plateformes mobiles de type uni-cycle et voiture. Communication personnelle, 2003.

[75] Y. Yamamoto, Control and coordination of locomotion and manipulation of a wheeled mobile manipulator. Thèse, University of Pennsylvania, 1994.

[76] Desineni Subbaram. Optimal control systems. CRC, 2003.

[77] Brian Anderson. Optimal control, linear quadratic methods. Prentice-Hall, 1989.

[78] D. E. Rumelhart, et J. L. McClelland. Parallel Distributed Processing. MIT Press, Cambridge, MA, 1986.

[79] K. Hornik, M. Stinchcombe, et H. White. Multilayer Feedforward networks are universal approximators, Neural Networks, vol. 2, pp. 359-366, 1989.

[80] K. S. Narendra & K. Parthasarathy, Identification and Control of Dynamical Systems Using Neural Networks. IEEE Trans. on Neural Networks, vol. 1, pp. 4-27, 1990.

[81] R. Hecht-Nielsen, Kolmogorov's mapping neural networks existence theorem. First IEEE International Conference on Neural Networks, vol. 3, pp. 11-14, San Diego, CA, 1987.

[82] K. Funahashi, On the approximate realization of continuous mapping by neural networks. Neural Networks, vol. 2, pp. 183-192, 1989.

[83] G. Cybenko, Approximation by superposition of a sigmoidal function. Technical Report, University of Illinois, Urbana, 1988.

[84] G. Cybenko, Approximation by Superposition of a Sigmoidal Function. Mathematics of Control, Signals, and Systems, vol. 2, pp. 303-314, 1989.

[85] F. Girosi, M. Jones et T. Poggio, Regularization Theory and Neural Networks Architectures. Neural Computation, vol. 7(2), pp. 219-269, 1995.

[86] Tianping Chen et Hong Chen, Approximations of Continuous Functional by Neural Networks with application to dynamical systems, Appeared in IEEE Transactions on Neural Networks, 1993.

[87] M. Kawato, Feedback-error-learning neural network for supervised motor learning. Advanced neural computers, Elsevier Science Publishers, 1990.

[88] H. Khalil, Nonlinear systems. New Jersey, Prentice Hall, 2002.

[89] J. Godjevac. Idées nettes sur la logique floue. Presses polytechniques et universitaires romandes. Lausanne, 1999.

[90] J. Dréo, A. Pétrowski, P. Siarry et E. Taillard, Métaheuristiques pour l'optimisation difficile, Eyrolles, 2005.

[91] M. Dorigo et T. Stutzle, Ant Colony Optimization, Massachusetts Institute of Technology, 2004.

[92] E. Bonabeau, M. Dorigo, et G. Theraulaz, Swarm intelligence: from natural to artificial systems, Oxford University Presses, New York, USA, 1999.

[93] I. Wagner et A. Bruckstein, Special Issue on Ant Robotics, Annals of Mathematics and Artificial Intelligence, vol. 31, pp. 1-4, 2001.

[94] Y. Kanayama, Y. Kimura, F. Miyazaki et T. Noguchi. A Stable tracking control method for an autonomous mobile robot, in Proc. IEEE Int. Conf. Robot. Autom., pp. 384-389, 1990.