REPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE POPULAIRE

MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITÉ MOULOUD MAMMERI DE TIZI OUZOU

Faculté de Génie Electrique et d'Informatique
Department d'Automatique

THÈSE DE DOCTORAT

Présentée pour l'obtention du diplôme de
DOCTORAT 3ÈME CYCLE LMD

SPÉCIALITÉ : AUTOMATIQUE

OPTION : COMMANDE DES SYSTÈMES

PAR :

WELID BENCHOUCHE

Commande prédictive d'un robot mobile: Application à la navigation dans un milieu contraint

---

DEVANT LE JURY :

Président	M. MAIDI Ahmed	Professeur	UMMTO
Rapporteur	MELLAH Rabah	Professeur	UMMTO
Examinateur	Mme ALKAMA Sadia	MCA	UMMTO
Examinateur	LAGHROUCHE Mourad	Professeur	UMMTO
Examinateur	FERGUENE Farid	Professeur	USTHB

Année 2024

Résumé

L'intérêt pour la conception, la fabrication et l'utilisation de robots mobiles autonomes s'est rapidement accru au cours de la dernière décennie. La principale motivation de cet intérêt est le large éventail d'applications potentielles dans lesquelles ces systèmes autonomes peuvent servir. Ces applications incluent mais ne sont pas limitées à, la couverture de zone, les missions de patrouille, la surveillance de périmètre, les missions de recherche et de sauvetage, et la conscience situationnelle. Notre travail a pour objectif principal de réaliser la synthèse de lois de commande non linéaires et de navigation adaptées au contrôle dynamique et la navigation d'un robot mobile non holonomique à entrainement différentiel est abordé. Une solution basée sur l'optimisation pour la commande et la navigation est conçue, analysée et appliquée à un robot mobile non holonomique. L'une des principales motivations pour considérer de telle solution est sa capacité à gérer des systèmes contraints et non linéaires tels que les robots mobiles non holanomiques. De plus, les développements récents des algorithmes d'optimisation dynamique ainsi que du traitement informatique ont facilité l'application des méthodes basées sur l'optimisation.

Pour commencer, nous entreprenons une analyse approfondie de l'état actuel de la robotique mobile ainsi que la définition de certains concepts clés qui sont très importants dans notre travail (Robot mobile, contraintes de mouvement, modélisation, évitement d'obstacles, matériel utilisé,... etc). Cette étude vise à explorer les différents phénomènes impliqués dans la description de la robotique mobile. Par la suite, Nous avons exposé la modélisation cinématique et dynamique d'un robot mobile à entraînement différentiel en utilisant la méthode d'Euler Lagrange avec un teste du modèle en boucle ouverte. Ensuite, Nous avons démontré l'application de la télé opération unilatérale sur un robot mobile à entraînement différentiel en utilisant le robot mobile Qbot2e. Puis, deux problèmes de contrôle d'un robot mobile non holonomiques sont considérés Tout d'abord, ces problèmes de contrôle sont la stabilisation (régulation) du point et le suivi de la trajectoire. Cependant, de nombreux contrôleurs ont été fournis, au début, nous avons commencé par le contrôleur le plus simple de stabilisation de point, à savoir le contrôle PID d'orientation et de translation, après cela, nous avons étendu le contrôleur de stabilisation de point à un contrôleur de stabilisation de séquence de points, qui utilise des segments reliant des points successifs, et ensuite un algorithme pour forcer le robot à rester sur le segment est utilisé pour garantir la stabilité et l'ordre des points de cheminement. Ensuite, nous avons étiré ce contrôleur pour obtenir un contrôleur de suivi de trajectoire. Dans ce dernier, nous avons utilisé un contrôle de linéarisation de la rétroaction pour guider le robot sur une trajectoire prédéfinie. Par ailleurs, nous avons proposé un schéma de contrôle prédictif de modèle (MPC) non linéaire pour remplir les deux tâches de contrôle. Il utilise le modèle de dynamique des erreurs pour piloter le robot, les coûts et les contraintes utilisés sont ceux traditionnellement utilisés dans la littérature pour garantir la stabilité asymptotique du système en boucle fermée. En revanche, nous utilisons un critère de stabilité dans lequel la stabilité asymptotique en boucle fermée peut être garantie en choisissant de manière appropriée la longueur de l'horizon de prédiction du contrôleur MPC. Cette méthode est basée sur la contrôlabilité en temps fini ainsi que sur les limites de la fonction de valeur du MPC. Ensuite, dans le cadre des simulations uniquement, nous avons utilisé une commande prédictive basée sur un modèle de la tension d'alimentation des deux moteurs d'entraînement pour contrôler le robot mobile. Nous avons pris en compte les contraintes sur les variables contrôlées et manipulées, ainsi que la dynamique de l'actionneur du moteur, afin de dériver un modèle dynamique linéaire de l'espace d'état. Les équations cinématiques non linéaires de base ont été linéarisées successivement pour obtenir un modèle espace d'état linéaire. Les modèles dynamiques et cinématiques ont été augmentés pour obtenir un seul modèle linéaire de l'espace d'état. Les résultats ont démontré un suivi très précis et une grande robustesse face aux variations de charge. Afin de simplifier la programmation, nous avons utilisé une boîte à outils appelée CasADi pour implémenter l'algorithme dans l'environnement MATLAB/SIMULINK. Pour permettre l'expérimentation en temps réel, nous avons ajouté un logiciel appelé QUARC à SIMULINK, qui assure la communication avec le robot. De plus, cette thèse construit une base pour l'utilisation du NMPC dans le domaine des robots mobiles non- holonomes pour naviguer (éviter des obstacles) à travers les obstacles avec une stabilité asymptotique.

Abstract

Interest in designing, manufacturing, and using autonomous robots has been rapidly growing during the most recent decade. The main motivation for this interest is the wide range of potential applications these autonomous systems can serve in. The applications include, but are not limited to, area coverage, patrolling missions, perimeter surveillance, search and rescue missions, and situational awareness. In this thesis, the area of control and navigation in a non-holonomic differential drive mobile robot is tackled Herein, an optimization-based solution for control and navigation is designed, analyzed, and applied to such systems. One of the main motivations for considering such solutions is their ability to handle constrained and nonlinear systems such as nonholonomic mobile robots. Moreover, the recent developments in dynamic optimization algorithms as well as in computer processing facilitated the application of such optimization based methods.

We started off with some concepts and historical study on mobile robots. After that, kinematic and dynamic modeling of a differential drive mobile robot was provided. In the next chapter, two control problems of a non-holonomic mobile robot are considered. First, these control problems are point stabilization (regulation) and path-following Here, many controllers have been provided, at the beginning, we started with the point stabilization simplest controller, namely PID orientation and translation control, after that, we stretched the point stabilization controller to a sequence point stabilization controller, which uses segments connecting successive points, and then an algorithm to force the robot to stay on the segment is used to guarantee stability and order of waypoints. Then, we stretched that controller to get a trajectory tracking controller, in the latter, we used feedback linearization control to guide the robot through the predefined trajectory. In the next chapter, we proposed a nonlinear model predictive control (MPC) scheme to fulfill the two control tasks. It utilizes the error dynamics model to drive the robot, costs and constraints used are the ones traditionally used in the literature to guarantee the asymptotic stability of the closed loop system. In contrast, we use a recently developed stability criterion in which the closed loop asymptotic stability can be guaranteed by appropriately choosing the prediction horizon length of the MPC controller This method is based on finite time controllability as well as bounds on the MPC value function. To facilitate coding, A toolbox named CasADi is used to implement the algorithm in MATLAB environment, Afterwards, the next chapter discussed and compared two algorithms to localize the robot and map the environment based on probability theory, namely the Extended Kalman Filter, and the Particle Filter which is also called Fast Slam. Finally in the last chapter, we discussed the methods that allow the robot to plan its motion and search for a suitable path to cross the waypoints and avoid obstacles. Moreover, this thesis constructs a foundation for using NMPC in the area of nonholonomic mobile robots to navigate through obstacles with asymptotic stability

« *Si nous examinons les lois générales de la perception, nous voyons qu'une fois devenues habituelles, les actions deviennent aussi automatiques.* »

Vaclav Havel

Remerciements

Tout d'abord, je remercie Allah le Tout-Puissant pour ma famille, ma santé, mon courage et ma foi qui m'ont permis d'arriver jusqu'à ce jour.

Je remercie Monsieur Rabah Mellah, Professeur à l'Université Mouloud Mammeri de Tizi-Ouzou, pour m'avoir tenu compagnie tout au long de cette thèse et m'avoir fait profiter de ses brillantes connaissances. Qu'il soit également remercié pour sa patience, sa disponibilité constante et les nombreux encouragements qu'il m'a prodigués.

Je tiens à exprimer ma gratitude à Monsieur Mourad Laghrouche, Professeur à l'Université Mouloud Mammeri de Tizi-Ouzou et Président du Vice-Décanat de la Post-Graduation et de la Recherche Scientifique, pour sa disponibilité, son soutien permanent, ses encouragements et ses précieux conseils. Je lui en suis très reconnaissant.

Je tiens à exprimer toute ma gratitude à Monsieur Maidi Ahmed, Professeur à l'Université Mouloud Mammeri de Tizi-Ouzou et Président du Jury, pour sa disponibilité, son soutien permanent, ses encouragements et ses précieux conseils.

Je remercie Monsieur Maidi Ahmed, Professeur, UMMTO ; Madame ALKAMA Sadia, MCA, UMMTO ; Monsieur LAGHROUCHE Mourad, Professeur, UMMTO ; et Monsieur FERGUENE Farid, Professeur, USTHB, pour avoir accepté d'être examinateurs.

Je tiens à exprimer toute ma gratitude à Monsieur Smain Tamchichet, ingénieur chez GE et mon très cher beau-frère, pour son aide inestimable tout au long de l'aventure, et je lui en serai éternellement reconnaissant.

Je tiens à remercier M. Mohamed Salah BENNOUNA, Maître de conférences à l'Université Kasdi Merbah de Ouargla et co-auteur de mes articles, pour avoir répondu calmement et patiemment à mes questions quotidiennes.

Je tiens à remercier Monsieur Mohamed Messaoudi, Maitre-assistant à l'université de Adrar, pour son aide immense pendant toute l'aventure, je serai éternellement reconnaissant.

Je tiens à remercier Monsieur Fouad Berrabah, Professeur et chef du département de génie électrique à l'université de M'sila, pour son aide immense pendant toute l'aventure, je serai éternellement reconnaissant.

Dédicace

A mes parents, Brahim et Fatima,

A mes frères Raouf et Khalil

A mes sœurs Sabrina, Meriem et Djihad

A ma femme Fatima et mon fils Ghaith

A tous mes amis, en particulier, Anis Madani, Mohamed Arbane, Aissat Omar, Khalil Bensalem, Bilal Zitouni, Daoud Nadir, Mahdi Salem, Khaled Benchouche, Djawad Dahmeche, Zekkar Amine, Boudia Assam, Alaa Sebaa, Kheir Eddine et Bachir Bousbaa et tous ceux que je n'ai pas pu citer.

Merci de faire partie de ma vie

Table des matières

Table des figures

Liste des Tableaux

Liste des abréviations

LQR	**L**inear **Q**uadratic **R**egulator	Régulateur linéaire quadratique
MIT	**M**assachusetts **I**nstitute of **T**echnology	Institut de technologie du Massachusetts
MPC	**M**odel **P**redictive **C**ontrol	Commande prédictive
NNMAC	**N**eural **N**etwork **M**odel **A**lgorithm	Commande par algorithme
PID	**P**roportional **I**ntegral **D**erivative	Controlleur proportionnelles intégrale dérivé
LBMPC	**L**earning **B**ased **M**odel **P**redictive **C**ontrol	Commande prédictive basée sur l'apprentissage
WMR	**W**heeled **M**obile **R**obot	Robot Mobile a Roue
LGMD	**L**obule **G**iant **M**oovement **D**etector	Détecteur de mouvement géant de la lobule
PLDM	**P**arallel **L**ine **D**istance **M**easurement	Mesure de distance par lignes parallèles
LED	Light Emitting Diode	Diode électroluminescente
SoC	**S**ystem **o**n a **C**hip	Système sur Puce
CPU	**C**entral **P**rocessing **U**nit	Unité centrale de traitement
GPU	**G**raphical **P**rocessing **U**nit	Unité de traitement graphique
GB	**G**iga **B**yte	**G**iga **o**ctets
RAM	**R**andom **A**ccess **M**emory	Mémoire Vive
BLE	**B**luetooth **L**ow **E**nergy	Bluetooth à faible énergie
GPIO	**G**eneral **P**urpose **I**nput **O**utput	Entrées/sorties à usage général
USB	**U**niversal **S**erial **B**us	Bus universel en série
HDMI	**H**igh **D**efinition **M**ultimedia **I**nterface	Interface multimédia haute définition
CSI	**C**amera **S**erial **I**nterface	Interface série de la caméra
DSI	**D**isplay **S**erial **I**nterface	Interface série d'affichage
TCP/IP	**T**ransmission **C**ontrol **P**rotocol/ **I**nternet **P**rotocol	Protocole de contrôle de transmission/protocole Internet
DC	**D**irect **C**urrent	Courant continu
SLAM	**S**imultaneous **L**ocalization **a**nd Mapping	Localisation et cartographie simultanées
2D	**T**wo **D**imensions	Deux Dimensions
3D	**T**hree **D**imensions	Trois Dimensions
WiFi	**W**ireless **F**idelity	Fidélité sans fil
P2P	**P**eer-**to**-**P**eer	pair-à-pair
WANET	**W**ireless **a**d hoc **Net**work	réseau ad hoc sans fil
HIL	**H**ardware **i**n the **l**oop	Le matériel dans la boucle
RVB	**R**GB **R**ed **G**reen **B**lue	**R**ouge **V**ert **B**leu
PC	**P**ersonal **C**omputer	**O**rdinateur **P**ersonnel
NASA	**N**ational and **A**eronautics **S**pace **A**dministration	Administration Nationale de l'Espace et de l'Aéronautique
LMR	**L**egged **M**obile **R**obot	Robot Mobile a Jambes

UAV	**U**nmanned **A**erial **V**ehicles	Véhicules aériens sans pilote
UAV	**A**utonumous **U**nderground **V**ehicles	véhicules sous-marins autonomes
MAC	**M**odel **A**lgorithm **C**ontrol	Commande par algorithme de modèle
NMPC	**N**onlinear **M**odel **P**redictive **C**ontrol	Commande prédictive non linéaire
IPOPT	**I**nterior **P**oint **OPT**imizer	Optimiseur de points intérieurs
OCP	**O**ptimal **C**ontrol **P**roblem	Problème de commande optimale
SISO	**S**ingle **I**nput **S**ingle **O**utput	Entrée unique, sortie unique
ANFIS	**A**daptive **N**euro **F**uzzy **I**nterference **S**ystem	**S**ystème d'**i**nterférence **a**daptatif **n**euro-**f**lou
LSC	**L**earning **S**ubtractive **C**lustering	**A**pprentissage du **r**egroupement **s**oustractif
CasADi	computer algebra systems for algorithmic differentiation	systèmes d'algèbre par ordinateur pour la différenciation algorithmique
RR	**R**obot **R**adius	**R**ayon du **r**obot
RO	**O**bstacle **r**adius	**R**ayon d'**o**bstacle

Nomenclature

$(x_i - c_{i,j})^2$ la distance quadratique entre la valeur d'entrée x_i et le centre $c_{i,j}$ du j-ième ensemble flou

$2D$ Deux dimension

$3D$ Trois dimension

$[ki_x, ki_y, ki_\theta]$ Les gains du contrôleur cinématique

$[x_d, y_d, \theta_d]$ La pose désirée

α, β, γ Paramètres du sorties

$\dot{\eta}$ Un vecteur réduit pour obtenir un meilleur modèle dynamique adapté à des fins de contrôle

$\dot{\theta}_e$ Erreur de suivi d'orientation

$\dot{Q}^I$ Vecteur des vitesses d'un point donné dans le repère allocentrique

$\dot{Q}^M$ Vecteur des vitesses d'un point donné dans le repère égocentrique

$\dot{x}_c$ vitesse instantanée du point C suivant l'axe des abscisses

$\dot{x}_e$ Erreur de suivi suivant l'axe des abscisses

$\dot{x}_P^M$ vitesse du point P suivant l'axe des abscisses dans le plan égocentrique

$\dot{y}_c$ vitesse instantanée du point C suivant l'axe des ordonnées

$\dot{y}_e$ Erreur de suivi suivant l'axe des ordonnées

$\dot{y}_P$ vitesse du point P suivant l'axe des ordonnées dans le plan allocentrique

$\dot{y}_P^M$ vitesse du point P suivant l'axe des ordonnées dans le plan égocentrique

ℓ Fonction décrivant le coût de fonctionnement

Γ_{jd} Vecteur des paramètres souhaités du régulateur ANFIS

Γ_j Le vecteur des paramètres à ajuster

κ Un instant de temps, utilisé dans la commande prédictive

λ Le vecteur de forces des multiplicateurs de Lagrange

$\Lambda^T(q)$ La matrice associée aux contraintes cinématiques du robot mobile

λ_j Multiplicateur lagrangien qui relie les contraintes aux forces de contrainte

$\mu_{i,j}(x_i)$ le degré d'appartenance de la i-ième entrée x_i à l'ensemble flou j-ième du système flou

ν Loi de commande du contrôleur dynamique

ω Vitesse angulaire du robot mobile

ϕ_L Vitesse angulaire de la roue gauche

ϕ_R Vitesse angulaire de la roue droite

ρ Point de contact entre la roue et le sol

$\sigma_{i,j}$ l'étendue ou la largeur de l'ensemble flou j-ième pour l'entrée i-ième

τ Le vecteur d'entrée

θ Orientation du robot mobile

$\vec{x}$ Vecteur de l'axe horizontal

$\vec{X}_I$ Axe d'abscisses dans le plan allocentrique

$\vec{X}_M$ Axe d'abscisses dans le plan égocentrique
$\vec{y}$ Vecteur de l'axe vertical
$\vec{Y}_I$ Axe des ordonnées dans le plan allocentrique
$\vec{Y}_M$ Axe des ordonnées dans le plan égocentrique
$\zeta(k)$ taux d'apprentissage
a_i Coefficients d'une équation caractéristique
$A_i; B_i, T_i$ Des termes linguistiques associés a des fonctions
A_s Matrice de la dynamique d'erreur linéarisée
$B(q)$ La matrice de transformation d'entrée
C Point du centre de masse du robot mobile Qbot2e
$c_{i,j}$ de la i-ième entrée x_i à l'ensemble flou j-ième du système flou
$c_{i,j}$ le centre ou le sommet de l'ensemble flou j-ième pour l'entrée i-ième
C_i Coefficients liés aux contraintes cinématiques
D Distance entre les deux roues du robot
d Distance entre le point P et une roue $= 0.5D$
E Fonction de coût de l'algorithme d'apprentissage
E_x La première sortie du régulateur ANFIS
E_y La deuxième sortie du régulateur ANFIS
$F(\dot{q})$ La matrice de friction de surface
$F(q(N_c))$ Coût terminal
$G(q)$ Le vecteur gravitationnel
h Distance entre les points P et C
$H(k)$ Matrice jacobienne d'observation du system
H_2 Une approche basée sur l'optimisation
I Inertie totale équivalente du robot mobile $= I_c + M_c h^2 + 2M_w d^2 + 2I_m$
I_c Moment d'inertie par rapport au centre de gravité du robot mobile
I_m Moment d'inertie par rapport au centre de la roue pour chaque roue motrice(avec actionneur)
I_w Moment d'inertie de chaque roue motrice par rapport au centre de gravité du robot mobile
J Fonction objectif $= [J_x, J_y]^T$
J_{N_c} Fonction objective utilisée en MPC
K_1 Le gain du régulateur ANFIS de la première sortie
K_2 Le gain du régulateur ANFIS de la deuxième sortie
$k_j^{'}$ La dérivée partielle de la sortie par rapport à l'entrée multiplier par le taux d'apprentissage
k_{p1} Premier gain du contrôleur dynamique
k_{p2} Deuxième gain du contrôleur dynamique
K_w Paramètre du taux d'apprentissage qui détermine la taille du pas de la mise à jour du poids
Ka Gain de Kalman
ki_θ Le gain du contrôleur qui influence θ_e
ki_x Le gain du contrôleur qui influence x_e
ki_y Le gain du contrôleur qui influence y_e
L Le Lagrangien $= T - V$

M_a Masse totale du robot mobile $= M_c + 2M_w$

M_c Masse du robot mobile sans les roues et les actionneurs $[kg]$

$M_{In}(q)$ La matrice d'inertie symétrique définie positive

M_w Masse des roues du robot mobile $[kg]$

N Noeud Circulaire

N_c Horizon de prédiction

O_i La sortie du i-ième neurone du réseau neuronal

P Point central dans l'axe entre les roues

$P(k)$ Matrice d'estimation de la covariance de l'erreur

P_{opt} Matrice de pondération symétrique définie positive pour l'état finale du robot mobile

Q^I Vecteur des coordonnées d'un point donné dans le repère allocentrique

q^I Le vecteur définissant la position du robot dans le plan allocentrique

Q^M Vecteur des coordonnées d'un point donné dans le repère égocentrique

q_e Le modèle d'erreur de suivi dans le repère égocentrique $= R(\theta)(q_{ref} - q_p)$

Q_i La force non conservatrice du robot

Q_{opt} Matrice de pondération symétrique définie positive pour les états du robot mobile

q_p La posture courante

q_{ref} La posture de référence

r Rayon de roue du Qbot2e

$R(\theta)$ Une matrice de rotation

$R(k)$ Matrice de covariance du bruit de processus

R_{opt} Matrice de pondération symétrique définie positive pour la commande du robot mobile

S Variable de Laplace

$S(q)$ Matrice de transformation de q à η

T L'énergie cinétique du robot

T_e Un temps d'échantillonnage

u Sortie du régulateur ANFIS

u^* Séquence de commande de minimisation qui résulte de la résolution de l'OCP

u_κ Une séquence de commande en boucle ouverte, en MPC

u_{kin} La loi de commande du contrôleur cinématique

V L'énergie potentielle du robot

v Vitesse linéaire du robot mobile

$V(q;\dot{q})$ La matrice centripète et de coriolis

v^{nor} Le poid normalisé

$v_{\rho L}$ Vitesse des points de contact du roue gauche dans le repère du robot

$v_{\rho R}$ Vitesse des points de contact du roue droite dans le repère du robot

V_j Une fonction de Lyapunov

V_{lya} Fonction de Lyapunov

V_L Vitesse linéaire de la roue gauche du robot mobile

v_L vitesse linéaire de la roue gauche du robot mobile

v_p Vitesse du centre de rotation P du robot mobile

v_{ref}	Vitesse linéaire de référence
v_R	vitesse linéaire de la roue droite du robot mobile
W_i	règle de mise à jour des poids
w_i	Le poids du i-ième neurone du réseau neuronal
w_{ref}	Vitesse angulaire de référence
x	Abscisse de la position du robot
x_a	Abscisse de la pose du robot dans le plan allocentrique
x_c	Abscisse du point C
x_i	La i-ième entrée du système flou
y	Ordonnée de la position du robot
y_a	Ordonnée de la pose du robot dans le plan allocentrique
y_c	Ordonnée du point C

symbol description

Introduction générale

La robotique est un domaine multidisciplinaire qui englobe plusieurs domaines d'expertise, notamment, l'informatique, l'ingénierie électrique/électronique, les mathématiques, l'ingénierie mécanique et la conception structurelle.

La robotique mobile représente un aspect important. En effet, bien que les robots mobiles partagent certaines similitudes, ainsi que certains problèmes avec d'autres types de robots, tels que les manipulateurs industriels, ils sont également confrontés à des défis particuliers liés à la mobilité et à la fonctionnalité dans le monde physique. C'est la raison pour laquelle, au cours de ces trois dernières décennies, le contrôle dynamique des robots mobiles a été soigneusement étudié pour leur permettre d'accomplir des tâches plus précises et surtout, variées.

Nous distinguons plusieurs conceptions appropriées aux robots mobiles terrestres, à savoir, l'architecture à roues (WMR) et à jambes (LMR) :

— Les robots mobiles à roues (WMR) sont très utilisés dans la vie active, particulièrement, ils conviennent aux applications courantes avec moins de complexité mécanique et de consommation d'énergie [1] ;
— Les robots mobiles à jambes (LMR) peuvent effectuer des tâches dans des environnements plus hostiles comme les escaliers, les tas de gravats, mais, les robots mobiles à jambes présentent de nombreux inconvénients, notamment leur complexité mécanique, leur difficulté à se déplacer sur certains terrains, leur forte consommation d'énergie, leur faible vitesse de déplacement, la complexité de leur maintenance et leur coût plus élevé que celui d'autres types de robots.

Bien qu'il existe de nombreuses autres options, les systèmes à deux, trois, quatre ou six pattes présentent un intérêt général dans l'accomplissement de la majorité des tâches quotidiennes [1]. Par contre, les robots mobiles n'ayant qu'une seule jambe sont rares car ils ne peuvent se déplacer qu'en sautillant. Ceci, réduit considérablement leur efficacité [1].

Par ailleurs, les robots mobiles comprennent également les véhicules aériens sans pilote (UAV), les véhicules sous-marins autonomes (AUV) et les manipulateurs mobiles à roues ou à jambes [1].

Dans le cadre de ce travail, notre choix s'est porté sur le robot mobile terrestre à roues, à entraînement différentiel (WMR) dont le prototype est largement disponible sur le marché et ce, pour vérifier nos simulations et nos expériences.

Comme, une telle plate-forme est relativement simple à construire, presque tous les roboticiens en herbe en ont créé une. Malgré sa popularité, il convient de noter que cette plate-forme est non-holonomique, avec un comportement non linéaire [2].

En outre, lorsqu'on commande un robot mobile, on cherche souvent en parallèle à lui conférer un certain niveau de capacité ou de sophistication, ce qui contribue

à l'expansion significative du domaine de la robotique mobile. Pour y parvenir, nous devons intégrer des principes et des méthodologies provenant d'autres domaines, tels que l'intelligence artificielle, les sciences cognitives, la psychologie et même le comportement humain et animal.

Ces domaines sont de vastes champs d'investigation à part entière, et leur maîtrise nécessite souvent des années de formation et de pratique spécialisées. A cet effet, plusieurs approches de contrôle ont été proposées (linéaires, non linéaires, cinématiques, dynamiques, basées sur des modèles, sans modèles, adaptatives, directes, indirectes, décentralisées, etc.) pour permettre au robot mobile de naviguer dans son environnement, d'éviter les obstacles, de calculer le chemin le plus court qui le sépare de sa cible, de se recharger en cas de besoin, etc...

Les problèmes de contrôle associés à la classe des robots mobiles à entraînement différentiel ont attiré l'attention des chercheurs au cours des trois dernières décennies. Cet intérêt est dû, d'une part à des préoccupations théoriques et pratiques, d'autre part, la nature "non holonome" de cette catégorie impose des limites aux vitesses acceptables par le système. En outre, la non-holonomie s'avère précieuse car, elle réduit le nombre d'entrées de contrôle, tout en maintenant le système entièrement contrôlable dans l'espace d'état [3]. Cet avantage introduit toutefois une complexité liée au premier objectif de contrôle qui est la stabilisation point à point : la tâche ne peut pas être accomplie avec une simple rétroaction [4].

Dans la littérature, les principaux objectifs de contrôle associée à ce type de systèmes sont la stabilisation du point (régulation), ainsi que, le suivi d'une trajectoire. Le premier vise à conduire le robot d'une position à une autre, le second force le robot à suivre une trajectoire donnée, évoluant dans le temps [1].

Naturellement, la stabilisation point à point est un problème plus simple, où le robot doit seulement partir d'un point initial pour atteindre un point de destination souhaitée. Dans cette classe de problèmes, le comportement du robot entre le point initial et final, ainsi que l'orientation du robot, ne sont pas explicitement contrôlés. La stabilisation point à point peut être traitée comme une sous-classe des problèmes de suivi de chemin ou de régulation de posture, selon que l'objectif est de suivre un chemin ou d'atteindre un point de référence [2]. Le suivi de chemin est le problème de plus haut niveau qui consiste pour un robot à suivre un chemin prédéfini et à atteindre une destination souhaitée. Une forme plus générale de suivi de chemin est le problème de suivi de trajectoire qui est proposé en définissant une loi temporelle sur le chemin désiré et ce, en mettant implicitement une contrainte de vitesse à chaque point d'échantillonnage [2].

En l'absence d'obstacles dans l'espace de travail, les tâches de mouvement de base assignées à un WMR peuvent se réduire au déplacement entre deux postures du robot, tout en poursuivant une trajectoire donnée. En fait, la stabilisation par rétroaction à une posture donnée ne peut pas être obtenue par une commande lisse invariante dans le temps, ce qui rend le deuxième problème plus simple du point de vue de la commande, en raison des propriétés particulières de la cinématique non holonomique. Ceci montre que le problème est véritablement non linéaire, ce qui le rend impossible à résoudre avec les méthodes de conception traditionnelles (même localement) [5].

En revanche, lorsque des obstacles sont présents dans l'environnement de travail, le problème de commande devient également plus complexe. Ainsi, les méthodes de commande traditionnelles peuvent ne pas être suffisantes pour garantir la

sécurité et l'efficacité du mouvement du robot en présence d'obstacles statiques ou dynamiques et imprévisibles. Par conséquent, les techniques de commande avancées ou de planification de trajectoire deviennent alors essentielles pour permettre au robot de naviguer dans un environnement complexe, tout en évitant les obstacles.

Après une première tentative de conception de contrôleurs locaux, le problème du suivi de trajectoire a été résolu de manière indépendante et globale dans De LUCA et al [6] et B. d'Andréa NOVEL et al [7], en utilisant la linéarisation de la rétroaction dynamique. Le paradigme du « backstepping » peut également être utilisé pour synthétiser une technique récursive de suivi de trajectoire de systèmes non holonomes sous forme de chaîne Zhong-Ping JIANG et al [8].

En ce qui concerne la stabilisation de la posture, des contrôleurs à rétroaction discontinus et/ou variant dans le temps ont été proposés. Claude Samson [9, 10] a été le premier à proposer une stabilisation lisse variable dans le temps, tandis que M. AICARDI et al [11], C.C. de WIT et al [12], R.T. M'CLOSKEY et al [13], P. MORIN et al [14], et O.J. SORDALEN et al [15], et d'autres ont utilisé un contrôle discontinu (souvent variable dans le temps) de diverses manières. L'application de la linéarisation par rétroaction dynamique au problème de stabilisation de la posture a été introduite dans Alessandro DE LUCA et al [16], ce qui constitue une nouvelle contribution à cette classe.

Bien que de nombreuses simulations comparatives de certaines des techniques mentionnées ci-dessus soient fournies dans C. Canudas de WIT et al [17], pour un monocycle et dans A. DE LUCA et al [18] pour un véhicule ressemblant à une voiture, les tests expérimentaux approfondis sur un seul véhicule de référence sont limités.

L'objectif de notre travail consiste à évaluer et à comparer l'efficacité des méthodes de contrôle pour le suivi de trajectoires et la stabilisation de la posture dans des situations réelles et ce, tout en soulignant les problèmes de mise en œuvre causés par les non idéalités cinématiques ou dynamiques, tels que la discrétisation, la quantification du signal, la saturation des actionneurs et les lectures imprécises des données par des capteurs. La majorité des WMRs utilisent la cinématique d'un robot à entrainement différentiel, donc toutes les conceptions de contrôle sont présentées spécifiquement pour ce cas et testées expérimentalement sur le prototype de notre laboratoire Qbot2e de Quanser.

La majorité des techniques peuvent cependant être appliquées à des véhicules dont la cinématique est plus complexe.

Bien que les commandes utilisées aient reçu beaucoup d'attention, il reste encore des questions critiques qui n'ont pas été traitées de manière rigoureuse. Parmi ces questions fondamentales, nous avons retenus les suivantes :

- Quand le modèle cinématique est-il suffisant ?
- Quand le modèle dynamique est-il essentiel ?
- Quel est l'impact du modèle dynamique sur le contrôle ?
- Pourquoi la commande prédictive par modèle est naturelle pour la navigation des robots mobiles ?
- Est-il facile de passer de la simulation à l'expérimentation, et quelles sont les difficultés qui se posent ?

Pour répondre à ces questions, un modèle cinématique non linéaire et un modèle dynamique spécial du robot mobile à entraînement différentiel sont développés. De même, un schéma pour une approche par modèle inverse avec une commande d'action proportionnelle pour mettre en œuvre la technique de suivi est conçu pour la dynamique. Par ailleurs, un contrôleur non linéaire asymptotiquement stable est proposé pour le modèle cinématique, basé essentiellement sur le modèle de dynamique d'erreur.

L'ensemble du système est codé sur MATLAB/SIMULINK pour le prototypage rapide des paramètres de commande et la prédiction du comportement du robot. En outre, l'impact du modèle dynamique, a été montré par des simulations dont les résultats obtenus ont fait l'objet de notre publication [19].

Pour valider le travail mentionné, nous avons implémenté un régulateur ANFIS (Adaptive Neuro-Fuzzy Inference System) pour notre contrôleur, que nous nous avons implémenté sur la plateforme Qbot2e.

Un autre type de contrôleurs connu sous le nom de contrôle prédictif du modèle (MPC) est également développé à base du modèle dynamique d'erreur du WMR synthétisé, et sa capacité à naviguer loin des obstacles est montrée et prouvée par des simulations sur MATLAB et ce, en utilisant une boîte à outils pour les problèmes optimaux nommée CasADi. Les résultats obtenus ont également été présentés sous forme d'une communication internationnale référencée [20]. Ensuite, nous avons implémenté un contrôleur MPC non linéaire sur la plateforme Qbot2e pour les deux problèmes de contrôle et pour différents cas, afin de vérifier l'applicabilité des contrôleurs MPC non linéaires dans les travaux expérimentaux. Nous avons utilisé la boîte à outils de contrôle prédictif par modèle non linéaire dans Matlab/Simulink ainsi que les blocs Embotech (FORCESPRO) qui facilitent le prototypage expérimental. Enfin, les performances du MPC ont été testées en tenant compte de la dynamique du robot mobile, afin de vérifier la robustesse du contrôleur.

L'ensemble de nos travaux est regroupé en quatre chapitres. Dans le chapitre I, un état de l'art de la robotique mobile est entrepris. Il est constitué d'une étude permettant de comprendre les phénomènes mis en jeu dans la description de la robotique mobile.

Le chapitre II présente des algorithmes de calcul appropriés au modèles cinématiques directes et inverses du robot mobile à entraînement différentiel, en se basant sur la géométrie simple. La dynamique de la structure mécanique de ce dernier, à vitesse élevée est prise en charge par le modèle dynamique.

Le chapitre III est scindés en deux parties :

1. Des résultats de simulation et expérimentaux qui montrent l'impact du modèle dynamique dans le suivi de la trajectoire, en utilisant une technique de commande de rétroaction stabilisante qui est asymptotiquement stable au sens de lyapunov d'un robot mobile à entraînement différentiel.
2. L'addition d'un régulateur ANFIS pour améliorer notre contrôleur proposé dans le contrôle expérimental du robot mobile Qbot2e.

Le chapitre IV aborde le problème de commande optimale avec des résultats de simulation et expérimentales pour les deux modèles d'intérêt précédent, à savoir, le modèle d'erreur dynamique et le modèle dynamique plus cinématique. Cette stratégie permet de faciliter la navigation d'un robot mobile à entraînement différentiel et ce, en utilisant la commande prédictive par modèle non linéaire pour répondre aux

différents phénomènes non linéaires que se soit, dans l'environnement avec ou sans obstacles.

Enfin, nous terminons notre thèse par une conclusion générale qui récapitule ce qui a été fait et expose de nouvelles perspectives. Après la conclusion générale, nous avons inclus en annexe une description montrant comment notre robot Qbot2e est contrôlé sans fil.

CHAPITRE 1

État de l'art de la robotique mobile

1.1 Introduction

Les robots mobiles sont en eux-mêmes un champ d'étude, ils sont si larges qu'il est impossible de les étudier tous dans un seul article, un seul livre, ou dans notre cas, dans une seule thèse, c'est pourquoi, il faut s'en tenir à un nombre minimal d'objectifs clés (sous-domaines) dans une seule . Dans ce modeste travail, nous limitons nos études au sous-domaine du contrôle. Un type bien connu de robots mobiles, à savoir, l'entraînement différentiel a été choisi pour profiter de son modèle simple, de sa géométrie et de sa disponibilité sur le marché à moindre coût. L'objectif de ce chapitre est de motiver le lecteur sur ce type de véhicules terrestres, leurs propriétés, leurs avantages, leurs inconvénients, et leur caractère unique. Nous nous intéressons à tous les différents facteurs qui interviennent dans sa modélisation et son contrôle. Cet état de l'art ne cherche pas à offrir une vision exhaustive sur des thématiques de recherche associées au domaine de la robotique mobile, mais il présente l'ensemble des verrous scientifiques qu'il reste à lever pour aboutir au développement d'un robot autonome. Parmi ceux-ci, nous nous focalisons alors sur celui de la navigation d'un robot mobile.

1.2 Robot mobile

Les robots mobiles peuvent se déplacer dans leur environnement et ne sont pas confinés à une seule zone. En revanche, les robots industriels sont généralement plus ou moins stationnaires et se composent d'un bras articulé (manipulateur à liaisons multiples) et d'un ensemble de préhension (ou effecteur final), fixés à une surface fixe, à travers une articulation.

Les chercheurs peuvent utiliser un cadre de référence allocentrique (absolu) ou égocentrique représenté à la Figure (1.1) pour étudier les mouvements des robots mobiles. Le cadre de référence allocentrique établit les coordonnées par rapport à un point de référence externe fixe, ce qui permet de bien comprendre la position et le mouvement du robot par rapport à son environnement. En revanche, le cadre de référence égocentrique est placé sur le robot lui-même, ce qui lui permet d'analyser ses propres mouvements et interactions. La sélection de ces cadres est déterminée par les objectifs uniques de l'étude et la nature des tâches du robot, et fournit des informations vitales sur le comportement et la dynamique des robots mobiles.(1.1)

Lorsque l'on parle de robotique mobile, on se concentre souvent sur les robots à roues. Ces systèmes sont plus simples à créer que d'autres types de robots mobiles

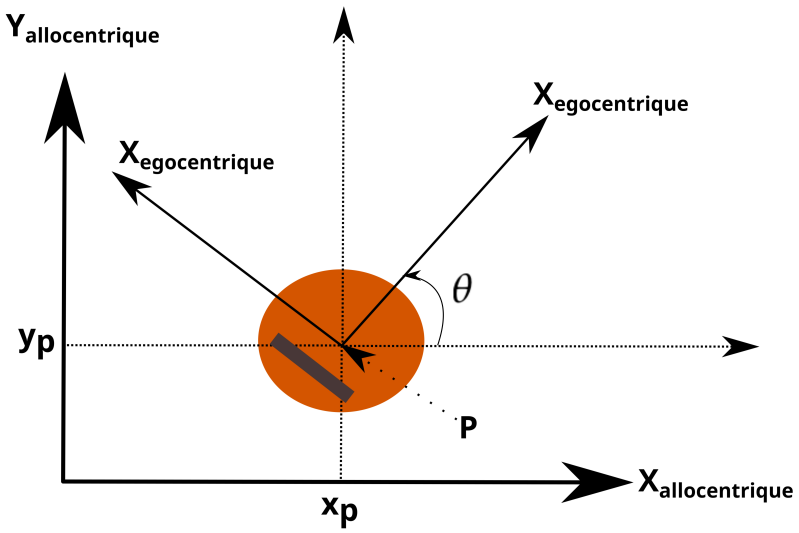

FIGURE 1.1 – Repère allocentrique et égocentrique

et permettent des études de navigation plus rapides. Ce type de robot est fréquemment utilisé dans la recherche sur les systèmes autonomes. On trouve également la robotique mobile avec des jambes, avec principalement la robotique humanoïde, mais aussi des robots avec un plus grand nombre de jambes qui offrent de bonnes propriétés pour la locomotion dans des environnements difficiles (environnements forestiers et agricoles). La stabilisation des mouvements de ce genre de robots est un thème de recherche très important [21]. Finalement, il y a aussi beaucoup d'autres types de robots mobiles (robots marins [22], sous marins [23], drones volants [24], micro [25] et nano robots [26]), Généralement, un tel robot est étudié dans le cadre de thèmes spécifiques avec des problèmes particuliers pour l'application en question(1.2)

Les robots mobiles peuvent fonctionner selon deux modes : à distance (télé opéré) et de manière autonome (commande en boucle fermée). En mode télécommandé, un opérateur donne des instructions au robot à l'aide d'une interface de commande telle qu'un joystick, un clavier ou une souris. Ces instructions sont envoyées au robot par l'intermédiaire d'un lien de communication, tel qu'Internet ou un satellite. Dans ce mode, le robot répond aux instructions de l'opérateur et observe l'environnement qui l'entoure par divers moyens tels que le retour d'image ou le retour haptique. Il convient de noter que le degré de téléopération peut varier, ce qui détermine le niveau de précision requis pour certifier le système comme "robotique" [27], afin de donner les ordres appropriés au contrôle. Les chercheurs dans ce domaine se focalisent sur les défis relatifs aux réseaux de télécommunication, tels que les retards de communication, les problèmes de contrôle et la perte de données. Ils travaillent également sur l'amélioration de la perception de l'environnement par l'opérateur en explorant des aspects tels que les interfaces haptiques et le retour d'effort.

En revanche, en ce qui concerne le mode autonome, le robot doit décider par

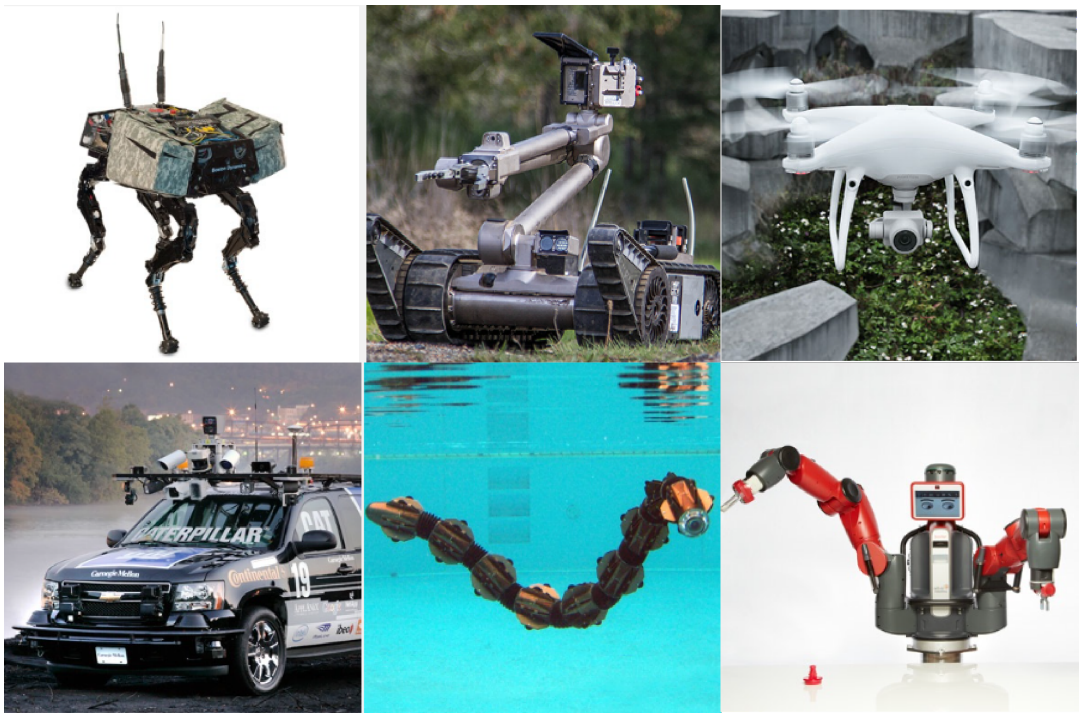

FIGURE 1.2 – Différents types de robots mobiles

lui-même. Cela implique qu'il doit être capable à la fois de bien percevoir son environnement, tout en sachant comment réagir en conséquence, conformément au niveau d'autonomie. C'est à lui qu'il appartient de planifier son itinéraire et de choisir les mouvements qui lui permettront d'atteindre son but. L'étude de ce thème porte principalement sur la localisation et la cartographie de l'environnement du robot autonome, ainsi que sur le contrôle de ces robots (structure de commande, méthodes de commande et planification).

Il est important de distinguer le concept d'autonomie décisionnelle de celui de l'autonomie énergétique lorsqu'il s'agit de robots. Bien que ces deux concepts soient étroitement liés, ils ne doivent pas être confondus. L'autonomie décisionnelle se réfère à la capacité d'un robot à prendre des décisions de manière autonome, tandis que l'autonomie énergétique concerne la capacité du robot à gérer efficacement son énergie, de telle manière à la préserver et éventuellement à se recharger en cas de nécessité.

Pour un robot mobile totalement autonome du point de vue décisionnel, l'une de ses principales préoccupations serait idéalement de pouvoir gérer lui-même ses réserves d'énergie.

1.3 Robot mobile avec entraînement différentiel

Un robot à entraînement différentiel est équipé de deux roues motrices de telle manière que la vitesse globale est répartie entre les roues gauche et droite.

Tous les systèmes évoluant sur un plan $2D$ ont trois degrés de liberté : la translation le long de l'axe x, la translation le long de l'axe y et la rotation autour d'un axe

z normal à $(\vec{x}, \vec{y})$. Une roue traditionnelle, par contre, n'a que deux degrés de mobilité : elle ne peut que se translater (avancer ou reculer) ou tourner sur elle-même. Elle ne peut pas se déplacer horizontalement (latéralement) en dérapant transversalement.

La plupart des robots mobiles "traditionnels" sont incapables d'effectuer un mouvement transversal immédiat (parallèle à l'axe de rotation de la roue) en raison de cette restriction. Une voiture ne peut pas se garer en parallèle tant qu'elle n'a pas effectué un certain nombre de mouvements. Il s'agit d'une limitation commune à tous les robots mobiles équipés de roues différentielles. Les robots mobiles non ergonomiques ont un nombre de degrés de mobilité inférieur au nombre de degrés de liberté, et cette limitation concerne principalement les robots mobiles à roues.

1.4 Contraintes de mouvement

Le mouvement du Robot est limité par des contraintes dynamiques et cinématiques. Les contraintes dynamiques trouvent leur origine dans le modèle dynamique du système où la réponse du système est limitée en raison de son inertie ou des contraintes des actionneurs (par exemple, le couple limité du moteur d'entraînement à cause de ses capacités ou bien pour éviter le glissement des roues). Les contraintes cinématiques trouvent leur origine dans la construction d'un robot et son modèle cinématique. Les contraintes cinématiques peuvent être holonomiques ou non holonomiques. Les contraintes non holonomiques limitent certaines directions de conduite d'un robot mobile. Par contre, les contraintes holonomiques sont liées à la dimensionnalité relative à la description de l'état du système (coordonnées généralisées). Ainsi, dans le cas où ces dernières contraintes se présentent, on peut exploiter le fait que certaines coordonnées de l'état dépendent des autres, pour les éliminer.

Un système est holonomique s'il n'a pas de contraintes cinématiques ou s'il n'a que des contraintes holonomiques. Les systèmes holonomiques n'ont aucune limitation dans leur espace de vitesse, donc toutes les directions de mouvement dans l'espace d'état sont possibles. En revanche le système est non holonomique s'il a des contraintes non holonomiques et ne peut donc pas se déplacer dans une direction arbitraire (par exemple, une voiture ne peut se déplacer qu'en avant ou en arrière et pas dans la direction latérale). En outre, pour les systèmes holonomiques, on peut déterminer un sous-ensemble de coordonnées généralisées indépendantes qui définit un espace dans lequel toutes les directions de mouvement sont possibles. Par contre pour les systèmes non holonomiques, le mouvement du système n'est pas arbitraire ; les contraintes non holonomiques définissent un sous-espace des vitesses possibles à l'instant présent.

Par conséquent, l'état des systèmes holonomiques, dépend directement de la configuration des variables internes du système (rotation des roues, angles des articulations). Ce qui n'est pas le cas pour des systèmes non holonomiques, car on ne garantit pas le retour des variables internes du système aux valeurs initiales, ceci est due au fait qu'on ne garantit pas le retour des variables internes du système aux valeurs initiales, en raison que l'état d'un système non holonomique dépend du chemin parcouru décrit par la séquence des variables internes.

Pour une explication approfondie, se référer au livre de Klancar [28][P,32-44]

1.4.1 Domaines de recherche en robotique mobile

La Figure (1.3) ci-dessous, illustre les différents problèmes de la robotique mobile, classés en 4 niveaux.

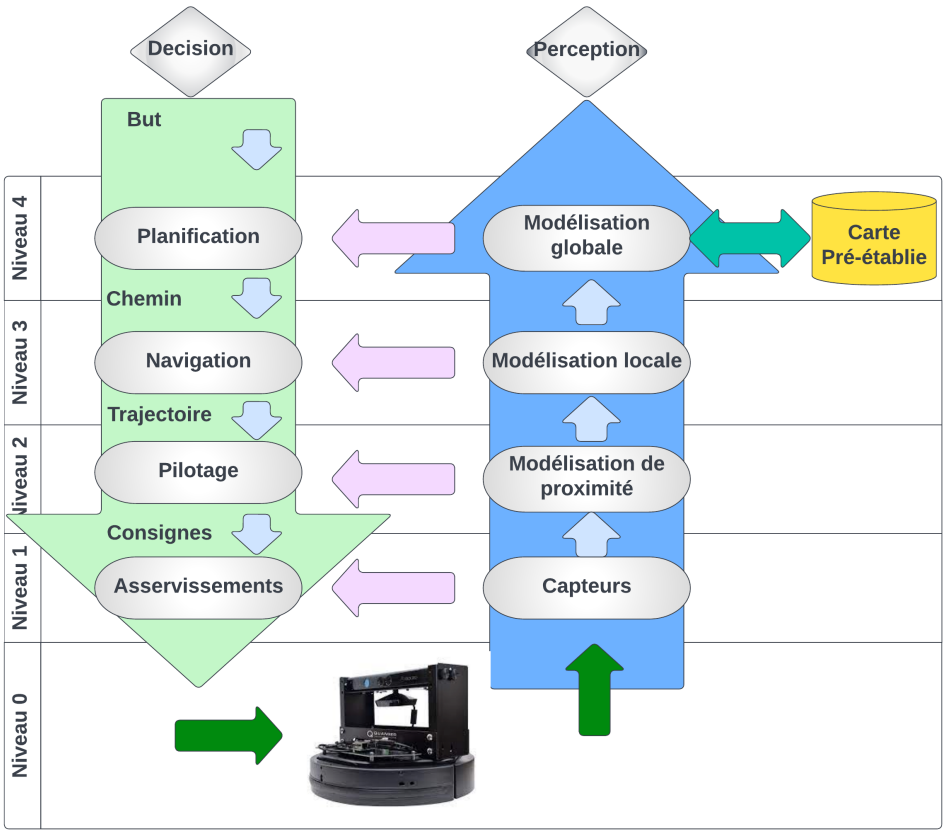

FIGURE 1.3 – Architecture de contrôle de la robotique mobile

Le niveau 0 comprend les parties physiques du robot, telles que le système mécanique articulé et les actionneurs qui lui permettent de se déplacer. C'est la base autour de laquelle toute l'architecture est construite.

Le premier niveau contient les boucles de rétroaction de niveau inférieur du robot, tandis que le niveau de perception contient les capteurs qui fournissent des données utilisées par le module de prise de décision pour faire fonctionner chaque moteur du robot.

Le deuxième niveau correspond au pilote du robot, qui reçoit du navigateur un itinéraire à suivre et utilise les informations du bloc de perception pour produire des ordres à transmettre aux moteurs pour l'asservissement. Le composant de perception analyse les données brutes, par exemple en les segmentant ou en détectant les zones vacantes, et les transmet au pilote. En plus de convertir la trajectoire du navigateur en ordres moteurs, le pilote peut également émettre des jugements tels que des manœuvres d'évitement d'obstacles dans des scénarios d'urgence.

Le troisième niveau est celui du navigateur, qui utilise une représentation de l'environnement local du robot, souvent sous la forme d'une carte, fournie par l'élément de perception de ce niveau. Cette carte est continuellement mise à jour au fur

et à mesure que de nouvelles données provenant de capteurs externes sont retransmises et analysées par les blocs de perception inférieurs. Cette carte locale est utilisée par le navigateur pour construire une trajectoire acceptable pour le robot mobile, lui permettant de naviguer entre les obstacles tout en suivant une trajectoire spécifiée par un niveau supérieur, principalement le planificateur. La trajectoire calculée tient compte à la fois des obstacles et des restrictions cinématiques et dynamiques du robot. Par conséquent, le navigateur est essentiel pour relier ce que nous voulons que le robot exécute (la trajectoire), les restrictions liées aux capacités de mouvement du robot et les contraintes locales dues aux obstacles. Lorsque la trajectoire est déterminée, le navigateur l'envoie au conducteur, qui utilise les actionneurs pour la mettre en œuvre.

Le quatrième et dernier niveau est réservé au planificateur. La section décisionnelle de ce bloc cherche à choisir un objectif à atteindre, qui peut être une destination finale ou une série de destinations à visiter. Sa fonction est de calculer le meilleur chemin pour atteindre cet objectif, qui sera ensuite communiqué sous forme d'instructions au navigateur de troisième niveau. Pour ce faire, le planificateur utilise un modèle global de l'environnement fourni par la composante perception du niveau 4. La partie perception utilise normalement des cartes prédéterminées de sa base de données pour construire cette carte globale, qu'elle peut ensuite intégrer aux informations actuelles, telles que la carte locale, pour mettre à jour la modélisation de l'environnement.

Le travail effectué dans le cadre de cette thèse est strictement consacrée aux niveaux 0, 1, et 2, à savoir la modélisation et pilotage, ainsi, le reste de cet état de l'art traitera le type de robot utilisé, de quelques concepts concernant ce type de robots, des tratégies de commande développées pour ce type de robot et par conséquent traitée à travers le développement de ces stratégies de commande constitue l'objectif principal de cette thèse (la problématique que nous souhaitons résoudre).

1.5 Modélisation cinématique et dynamique

Les modèles de mouvement peuvent décrire la cinématique du robot, et nous nous sommes intéressés aux formalismes mathématiques représentant le mouvement du robot mobile sans tenir compte des causes qui le provoquent à savoir les forces ou les couples. Cependant, le modèle cinématique décrit les relations géométriques entre les paramètres d'entrée (commande) et le comportement du système représenté par ses vitesses. Ces relations sont données par un ensemble d'équations différentielles du premier ordre [29]. Les modèles dynamiques expriment le mouvement du système lorsque des forces ou couples lui sont appliquées, en tenant compte de la physique du mouvement, des différents paramètres d'énergie, de masse et d'inertie du système. Par conséquent, les descriptions relatives aux modèles dynamiques sont données par des équations différentielles du deuxième ordre [30].

1.6 Télé opération :

On voit souvent un jeune enfant mettre des choses dans sa bouche, ce qui est très instinctif et normal. Au contraire, c'est très important, car une fois posé, son goût, son poids, sa raideur et autres sont perçus. Donc, la manipulation d'objets et l'identification de matériaux est une tâche primordiale et essentielle.

S'il n'est pas possible de manipuler un objet directement à mains nues ou avec la bouche, ou tout autre effecteur corporel Figure (1.4)a, une interaction indirecte à l'aide d'un outil est parfois appropriée Figure (1.4)b.. L'interaction à base d'outils est appelée télé opération, lorsque l'outil est divisé en deux parties, Figure (1.4)c., reliés par un module de connexion électrique, appelé contrôleur. Un téléopérateur se compose donc d'une interface opérateur (appareil maître) et d'un robot esclave (appareil esclave), connectés via un contrôleur. L'humain qui tient l'appareil maître s'appelle l'opérateur et l'objet manipulé s'appelle l'environnement (l'esclave).

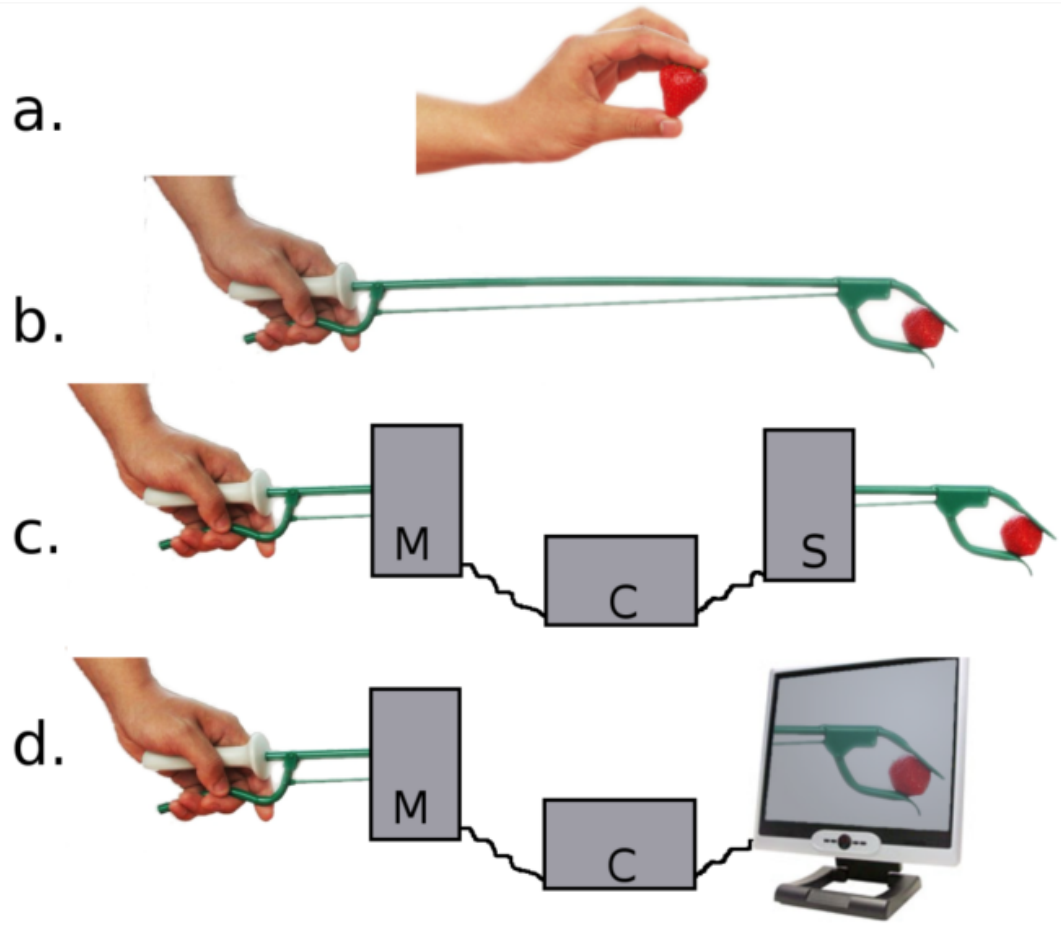

FIGURE 1.4 – Interaction et manipulation : a. Interaction physique directe b. Interaction physique indirecte c. Téléopération d. Interaction en réalité virtuelle. M = appareil maître, S = appareil esclave, C = contrôleur
[31]

La téléopération, qui consiste à utiliser un outil robotique télécommandé pour effectuer une tâche dans un environnement distant, est utilisée depuis 50 ans dans l'industrie nucléaire [32]. La technologie de téléopération haptique est actuellement utilisée dans les sites de recherche nucléaire pour manipuler des matières dangereuses, pour la robotique en haute mer et, dans une certaine mesure, pour les opérations spatiales. La technologie de téléopération est également utilisée dans les "systèmes de conduite par le fil" pour les avions et les prototypes de voitures, où l'interface de commande humaine (par exemple, le volant) est connectée électroniquement à un actionneur esclave (moteur de direction des roues) [33].

Une catégorie importante de téléopérateurs est celle où les forces d'interaction du côté de l'esclave sont renvoyées à l'opérateur via le maître, de sorte que l'opérateur puisse sentir l'objet distant. C'est ce qu'on appelle la "téléopération haptique". Cela ne sera pas étudié dans cette thèse, car notre objectif principal est le contrôle. Une technologie connexe est l'interaction en réalité virtuelle, illustrée à la Figure (1.4)d., où le robot esclave et l'environnement sont remplacés par un modèle informatique.

Les dispositifs haptiques développés jusqu'à présent concentrent leurs applications sur le contrôle des manipulateurs et non sur le contrôle du robot mobile, car les manipulateurs nécessitent des mouvements plus délicats que les robots mobiles. Dans de nombreuses applications, l'utilisation des robots mobiles est limitée aux simples navigations. C'est pourquoi un simple joystick à deux degrés de liberté a été utilisé pour la téléopération du robot mobile. Si nous avions une caméra capable d'effectuer un panoramique et de s'incliner, nous aurions incorporé davantage d'outils à degrés de liberté, mais la nôtre est fixe. Nous n'en avons donc pas besoin [31].

1.7 Commandes développées pour les robots mobiles à entrainement différentiel

Afin de permettre un mouvement efficace, régulier et continu du robot le long de la trajectoire souhaitée, le suivi de la trajectoire est important. Selon le théorème de Brocketts [34], une loi de commande par retour d'état statique, lisse et invariable dans le temps ne peut pas être utilisée pour stabiliser un système nonholonomique à une configuration donnée. Pour surmonter cette limitation, de nombreux chercheurs ont proposé plusieurs stratégies de commande non linéaires pour le suivi de trajectoire. La conception de la commande du robot peut alors se faire sur la base de modèles cinématiques ou à la fois cinématiques et dynamiques.

Dans [35-43], le problème de suivi de trajectoire pour les robots mobiles est résolu uniquement sur la base d'un modèle cinématique. L'algorithme de commande est asymptotiquement stable sous la condition d'une vitesse angulaire limitée.

Kadakkal et Cook [35] ont développé une commande à base de la cinématique du robot, en vue d'assurer une poursuite de trajectoire représentée par une courbe prédéterminée, ainsi un régulateur LQR amélioré est appliqué ce qui traduit la conversion du problème de suivi de trajectoire en un problème de régulation. Dans ce cas, la perturbation est la courbure de la trajectoire.

Liu et al [36] et Wang et al [37] ont présenté une commande de suivi de trajectoire adaptative, la stabilité est améliorée en utilisant une fonction de Lyapunov. Un champ de potentiel artificiel est utilisé pour faire naviguer le robot. L'avantage de cette commande est son adaptation à d'autres robots mobiles. Cependant, pour conserver la fluidité de la trajectoire du robot, une vitesse limitée doit être donnée au robot.

Chang et Meng [38] ont proposé une loi de commande non linéaire de suivi de trajectoire par retour d'état, tandis qu'une fonction de Lyapunov est utilisée pour assurer la stabilité du système.

Blasic [39] a proposé à base de la cinématique du robot mobile, un algorithme de commande non linéaire, construit de telle manière à assurer la stabilité au sens

de Lyapunov. L'algorithme a atteint une stabilité asymptotique basée sur les vitesses de référence.

Dans [40], Amoozgar et Zhang ont présenté une nouvelle méthode de commande appelée commande de guidage du robot mobile. Cette stratégie de commande utilise le concept de la stabilité au sens de Lyapunov pour assurer le guidage du robot mobile vers son orientation à chaque instant.

Comparativement aux stratégies de commande classiques à savoir commande prédictive de modèle, commande de suivi d'état (linéaire et non non linéaire), cet algorithme de commande a amélioré les performances de suivi du robot.

Osmankovic et Velagic [41] ont présenté une commande adaptative basée sur la cinématique du robot et une technique basée sur le gradient (adaptation basée sur la règle MIT) pour l'adaptation de la commande. Les résultats de la simulation montrent que le système est stable et robuste par rapport au contrôleur basé sur la linéarisation de la rétroaction. Cependant, ce contrôleur a donné des commandes de vitesse moins bonnes parfois.

Kunhe et al [42] ont proposé un algorithme de suivi de trajectoire basé sur une commande prédictive de modèle linéaire (MPC) et non linéaire. D'après les résultats de simulation, ils ont indiqué que le MPC non linéaire prend beaucoup de temps de calcul par rapport à un MPC linéaire. Klancar et Skrjanc [43] ont également présenté une approche de commande prédictive de modèle basée sur un modèle d'erreur de suivi linéarisé, les résultats expérimentaux montrent une bonne convergence et stabilité.

En ce qui concerne la cinématique et la dynamique du robot mobile, de nombreuses recherches ont été menées.

Dans [44-52], certains de ces travaux sont présentés. Fierro et Lewis [45, 46] ont présenté une commande adaptative robuste basée sur les réseaux de neurones pour traiter les perturbations en tenant compte de la cinématique et de la dynamique du robot, la commande est un couple qui est difficile à traiter.

Yang et Kim [47] ont développé un algorithme de commande par mode glissant basé sur la cinématique pour suivre une trajectoire de référence donnée. Ils ont présenté le modèle dynamique et l'ont linéarisé en utilisant une méthode de calcul du couple, le schéma de commande proposé était robuste contre les erreurs de conditions initiales, les perturbations de mesure et les bruits dans les données du capteur.

Martins et al [44] ont présenté une commande adaptative de suivi de trajectoire basée sur la dynamique du robot dont ses signaux d'entrée sont les vitesses, ils ont utilisé la théorie de Lyapunov pour améliorer la stabilité du système. Cette stratégie de commande est simulée ainsi qu'appliquée expérimentalement. Canigur et Osmangazi [53] ont présenté une commande adaptative à référence de modèle basée sur la cinématique et la dynamique du robot mobile, pour synthétiser la commande force de ce dernier de telle manière à suivre un modèle de référence présélectionné.

Zhang [54] a proposé une commande par algorithme de réseau de neurones (NNMAC) pour suivre une trajectoire prédéterminée, la commande par algorithme de modèle (MAC) est une commande prédictive. Comme le MAC a besoin d'un modèle bien connu du système pour concevoir la commande, le réseau de neurones a la capacité d'estimer les paramètres inconnus et les caractéristiques non linéaires du robot.

Dans [55], Alessandretti et al ont proposé une commande à prédiction de modèle pour le suivi de trajectoire. Cette commande garantit la convergence asymptotique de la position du robot.

Resende et al [56] ont proposé et validé une commande de trajectoire non linéaire, en utilisant des règles floues pour définir les gains de la commande. Les règles floues ont été conçues pour limiter le signal de commande, ainsi que pour réduire les erreurs provenant de la dynamique du robot, La stabilité de la commande a été étudiée par la théorie de Lyapunov, et le modèle dynamique du robot peut être adapté à d'autres robots.

Dans [48], Shojaei et al ont proposé une commande de suivi de trajectoire qui combine la technique de commande de dynamique inverse et la stratégie de commande PID adaptative, la commande a été simulée et validée en présence d'incertitudes paramétriques et non paramétriques.

Valiloo et al [49] ont proposé une commande à mode glissant avec une performance H_2 généralisée. La commande est divisée en deux parties : la première est la commande cinématique, qui génère les valeurs souhaitées des vitesses (linéaire et angulaire), tandis que la seconde est la commande dynamique basée sur le mode glissant avec la performance H_2 généralisée. La stabilité est discutée en utilisant la théorie de Lyapunov, et les résultats de simulation ont montré une meilleure performance pour traiter le patinage des roues.

Junag et al [50] ont conçu et simulé une commande adaptative à rétroaction de sortie pour un problème de trajectoire. La commande est structurée en deux étapes. La première est la conception d'une commande adaptative de retour d'état pour réaliser un suivi asymptotique, tandis que la seconde est la conception de deux observateurs à gain élevé pour estimer les états inconnus. La stabilité du système est assurée par une fonction de Lyapunov.

Dans [51], Taheri-Kalani et Khosrowjerdi proposent une commande adaptative en présence d'incertitudes cinématiques et dynamiques pour suivre une trajectoire prédéfinie. La commande cinématique est basée sur la linéarisation de la rétroaction, tandis que la commande dynamique est basée sur la commande adaptative de référence de modèle, les incertitudes du robot sont modélisées comme des perturbations localisées.

Dans [52], Aswani et al ont présenté une nouvelle technique de commande appelée commande prédictive de modèle basée sur l'apprentissage (LBMPC) et l'ont appliquée à la commande du mouvement d'un hélicoptère quadrotor. L'algorithme de commande est très performant et robuste. Cette stratégie de commande n'a pas été mise en œuvre jusqu'à présent avec les WMR.

1.8 Évitement d'obstacles

Les robots mobiles à roues sont censés d'exécuter diverses tâches de manière autonome. Pour accomplir ces tâches, les robots doivent se déplacer vers une position et une orientation souhaitées toute en évitant une variété d'obstacles. Les obstacles ont plusieurs classifications. Selon la nature de l'obstacle, il peut être de forme convexe, concave ou les deux. Selon son statut, l'obstacle peut être statique (lorsque sa position et son orientation par rapport à un cadre de coordonnées fixe connu

sont invariantes dans le temps), ou dynamique (lorsque sa position, son orientation ou les deux changent par rapport au temps). Plusieurs approches développées pour l'évitement d'obstacles se sont intéressées à la recherche d'un chemin avec des informations entièrement connues sur l'environnement du robot mobile, ou à la détermination du chemin en temps réel en utilisant les informations des capteurs [57]. Il existe de nombreuses méthodes, dans cette section une revue des algorithmes de commande d'évitement d'obstacles est présentée.

Jung et al [58] ont présenté un algorithme d'évitement des collisions utilisant une approche par champ de potentiel et un modèle dynamique du robot. Pour compenser les incertitudes du robot, un réseau de neurones a été introduit. Sekiguchi et al [59] et Jiang et al [60] ont présenté un algorithme de commande basé sur la cinématique du robot et la méthode du champ de potentiel, le contrôleur sélectionne une fonction de Lyapunov appropriée qui permet au robot d'éviter les obstacles. Deng et al [61] ont proposé un algorithme de commande d'évitement d'obstacles en utilisant un détecteur de mouvement géant de la lobule (LGMD), il s'agit d'une méthode de commande visuelle. Cette méthode est inspirée de la méthode utilisée par les criquets pour détecter et éviter les obstacles.

Outre les caméras, un algorithme de mesure de distance est utilisé pour calculer la route optimale que le robot mobile suivra pour éviter les obstacles. Mester [62] et Rusu et al [63] ont présenté un système de navigation réactif flou pour éviter les obstacles en se basant sur les vitesses angulaires des roues et les données collectées par les capteurs. Le premier a utilisé des caméras de vision stéréo et des capteurs à ultrasons, tandis que le second a utilisé des capteurs infrarouges. Chen et Juang [64] ont présenté deux stratégies d'évitement des obstacles. La première est appelée un modèle d'évitement d'obstacles à courte distance dans lequel le robot utilise les signaux des capteurs à ultrasons, tandis que la seconde est appelée un modèle d'évitement d'obstacles piloté par la cible dans lequel une théorie floue est utilisée avec les signaux des capteurs pour commander la vitesse du robot. Dans [65], Chen et al ont proposé une méthode de détection par imagerie pour éviter les obstacles, combinée à la mesure de distance par lignes parallèles (PLDM) pour détecter la position des obstacles. Cette méthode a l'avantage de n'utiliser qu'une seule webcam, facile à installer et bon marché.

Watanabe et al [66] ont présenté un algorithme de commande d'évitement d'obstacle par contrôleur flou basé sur l'image, le système de commande visuelle est basé sur le principe de détection des bords. Cette méthode présente l'avantage qu'il n'est pas nécessaire de mesurer la distance exacte entre le robot et l'obstacle. Cependant, le démérite de ce système est que l'obstacle est supposé être détectable par la détection des bords. Dans [67], Sirbu et Dobrea ont conçu et implémenté un algorithme de commande d'évitement d'obstacle basé sur un ensemble de capteurs infrarouges et un algorithme génétique. L'avantage majeur de cet algorithme est que les situations d'évitement de collision sont résolues et que le chemin optimal est généré en ligne en utilisant l'algorithme génétique. Kang et Prasad [57] ont présenté un système intégré de commande optimale d'évitement d'obstacles capable d'éviter les obstacles dans un environnement incertain. Le système utilise un algorithme de génération de carte basé sur les mesures des capteurs, un algorithme de planification de trajectoire en ligne et une commande prédictive de modèle. Cet algorithme formule le problème d'évitement d'obstacles comme un problème d'optimisation de trajectoire avec des contraintes non linéaires.

1.9 Matériel utilisé

Au cours des nombreuses années de recherche, on a travaillé sur de nombreux matériels allant de la simple commutation de LED, et leur contrôle à l'aide de cartes Arduino, à des moteurs à courant continu, des moteurs pas à pas, différents types de capteurs, des caméras, des lidars, et enfin un robot mobile à roues qui les incorpore tous (1.5). C'est un projet très intéressant et amusant à faire, un projet qui est venu avec de nombreux problèmes, du matériels défectueux, et des capteurs incalibrables, à des problèmes financiers et de temps que nous n'avons pas à attendre que le produit vienne de Chine par exemple. pour découvrir qu'il n'y a pas de compatibilité logicielle sur internet. mais la principale raison qui n'a pas permis à notre robot de briller, est les problèmes de batterie ou d'alimentation en général, nous ne pourrions jamais calibrer cela, parce que le matériel sur lequel nous pouvions travailler n'est tout simplement pas pour le contrôle de haut niveau en utilisant MATLAB et SIMULINK. Nous avons cependant pu faire une téléopération JOYSTICK en utilisant notre robot dans l'environnement PYTHON, car c'est le principal langage de programmation de la carte mère utilisé "Raspberry Pi 3 model B". Ces problèmes de compatibilité nous ont conduit à travailler sur un robot mobile déjà construit, à savoir le Qbot 2e de QUARC Quanser en collaboration avec YUJIN ROBOT 1.6.

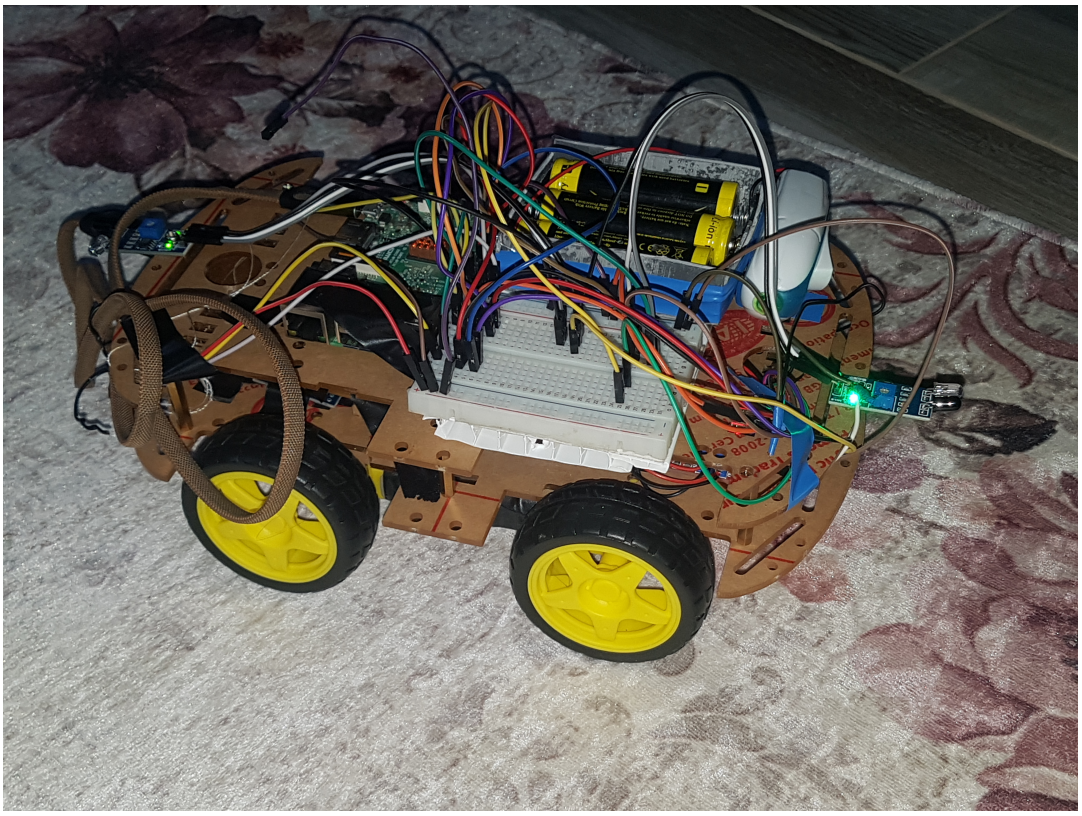

FIGURE 1.5 – Le robot mobile réalisé

1.9.1 Carte Raspberry Pi 3 model B

Le Raspberry Pi, parfois appelé Pi, est un petit ordinateur à bas prix inventé au Royaume-Uni par la Fondation Raspberry Pi.

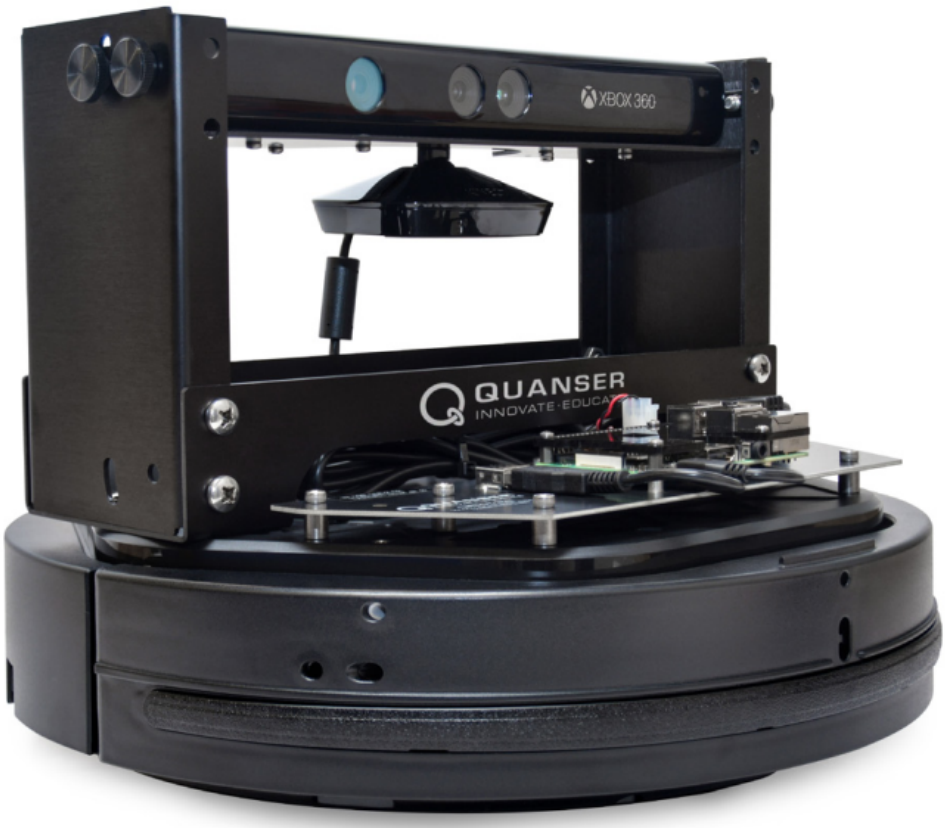

FIGURE 1.6 – Le robot mobile Qbot 2e de QUARC

De la taille d'une carte d'identité, il n'est pas aussi puissant qu'un ordinateur portable ou de bureau ; sa puissance de calcul est plus proche de celle d'un téléphone intelligent. Mais ce qui lui manque en puissance de traitement, il le compense par ses nombreuses fonctionnalités :

— Sa compatibilité avec la programmation en Python
— Les nombreuses façons dont vous pouvez l'utiliser
— sa taille et son coût réduits.
— une interface dans MATLAB/SIMULINK à travers une boîte à outils.

Le Pi, a besoin d'une carte mémoire SD pour fonctionner, cette dernière porte le système d'exploitation du Pi, et est riche en logiciels que l'on peut utiliser pour programmer.

Un système sur puce (SoC) est un circuit électronique miniaturisé qui intègre divers composants d'un ordinateur ou d'un système électronique sur une seule puce. Il comprend souvent une unité centrale de traitement (CPU), un processeur graphique (GPU), de la mémoire et d'autres fonctionnalités.

1.9.2 Caractéristiques du Raspberry Pi 3 modèle B

— Processeur Quad Core 1.2GHz Broadcom $BCM2837\ 64bit$
— 1 GO DE RAM
— LAN sans fil$BCM43438$ et Bluetooth Low Energy (BLE) sur la carte
— Ethernet 100 Base
— GPIO étendu à 40 broches

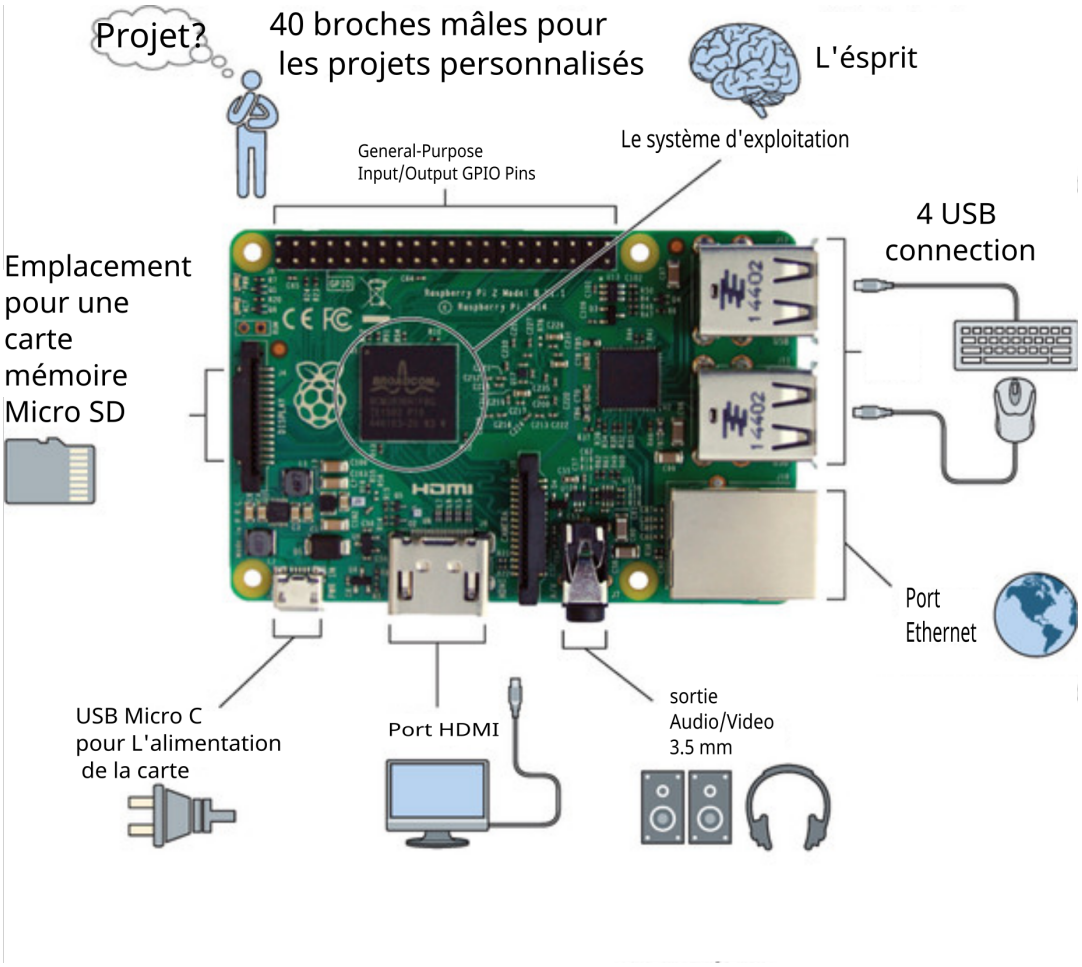

FIGURE 1.7 – La carte Raspberry Pi 3 model B

— 4 ports USB 2
— Sortie stéréo 4 pôles et port vidéo composite
— HDMI pleine grandeur
— Port caméra CSI pour connecter une caméra Raspberry Pi
— Port d'affichage DSI pour connecter un écran tactile Raspberry Pi
— Port Micro SD pour charger votre système d'exploitation et stocker des données
— Source d'alimentation Micro USB commutée améliorée jusqu'à $2,5A$

Toutes les informations techniques sont tirées de la feuille de calcul officielle du Raspberry pi 3 modèle B. [68]

1.9.3 Robot mobile Qbot 2e

Le QBot 2e est un robot terrestre autonome qui est le dernier entraînement différentiel de Quarc. Il utilise une carte d'acquisition de données embarquée et un ordinateur embarqué sans fil (un Raspberry pi 3 modèle B) pour obtenir des mesures des capteurs embarqués et entraîner les moteurs. Dans cette étude, le robot mobile QBot 2e est évalué en fonction de sa capacité à manœuvrer. Ainsi, la téléopération et le contrôle du robot mobile QBot 2e en expérimental sont réalisés. QBot 2e permet un environnement de connexion sans fil. Un modèle Simulink est développé dans

l'environnement MATLAB® 2021a. Le modèle créé est construit avec le logiciel de contrôle Quarc. Le modèle compilé est téléchargé avec une connexion TCP/IP vers QBot 2e et l'application est réalisée sur un ordinateur embarqué (le Raspberry pi 3 modèle B). La plateforme mobile QBot 2e est constituée de deux roues motrices centrales montées sur un axe commun. Cette configuration d'entraînement est connue sous le nom d'entraînement différentiel. Les deux roues motrices sont entraînées indépendamment en avant et en arrière afin d'actionner le robot, qui utilise donc un entraînement différentiel. Le mouvement des roues est réalisé à l'aide de moteurs 12V DC à haute performance. Le QBot 2 convient parfaitement aux applications de recherche telles que la cinématique avant et inverse, la navigation à l'estime et la localisation odométrique, la planification de trajectoire et l'évitement d'obstacles, la cartographie 2D et la carte d'occupation, l'acquisition, le traitement d'images, la localisation et la cartographie simultanées (SLAM), l'architecture de commande de haut niveau des robots mobiles et la commande de véhicules guidée par la vision.

Dans cette thèse, le QBot 2e est utilisé pour explorer ses capacités de manœuvre et de commande à distance. De plus, le QBot 2e est contrôlé en expériemental pour suivre des trajectoiresen utilisant un contrôleur par retour d'état asymptotiquement stable, un contrôleur adaptatif Neuro-Floue de type ANFIS. Enfin, un contrôleur de stationnement parallèle et de suivi de trajectoire utilisant un contrôle prédictif de modèle est réalisé avec succès. En raison des contraintes de temps et Bien que le capteur Kinect du robot permette d'obtenir des informations sur l'environnement, sa précision s'est avérée insuffisante pour une implémentation de l'évitement d'obstacles dans des scénarios expérimentaux. En revanche, plusieurs stratégies de commande du robot différentiel sont testées en simulation dans l'environnement MATLAB/SIMULINK.

1.9.4 Caractéristique du Qbot 2e

La plate-forme mobile Quanser QBot2e est composée de deux roues motrices centrales montées sur un axe commun qui divise le robot en deux parties égales, comme le montre la Figure (1.8). Cette configuration d'entraînement est connue sous le nom d'entraînement différentiel. Des roulettes à l'avant et à l'arrière du robot stabilisent la plate-forme sans compromettre le mouvement. Les deux roues motrices sont entraînées indépendamment en avant et en arrière afin d'actionner le robot. Cette approche de la géométrie des roues des robots mobiles est très courante en raison de sa simplicité et de sa maniabilité.

Le mouvement de chaque roue est mesuré à l'aide de codeurs, et l'orientation du robot représentée par l'angle theta θ est estimée à l'aide du gyroscope intégré. Pour plus d'informations sur la cinématique du robot Qbot2e, ainsi que la façon de générer des commandes des roues en vue d'obtenir des trajectoires de mouvement spécifique, vous pouvez consulter Laboratoire de cinématique avant/arrière et différentielle disponible sur le site de Quanser [69].

Le QBot 2e est livré avec un capteur de vision Microsoft Kinect, illustré à la Figure (1.8)b, qui produit des images en couleur (RVB) ainsi que des informations sur la profondeur. Le principe de fonctionnement de capteur est décrit d'une manière explicite par expériences du Laboratoire de vision informatique du site Quanser [69].

Le QBot 2e est également équipé de capteurs de chocs intégrés (à gauche, à droite et au centre) et de capteurs de falaise (à gauche, à droite et au centre), comme le montrent la Figure (1.8)a et la Figure (1.8)b. Ces capteurs peuvent être utilisés

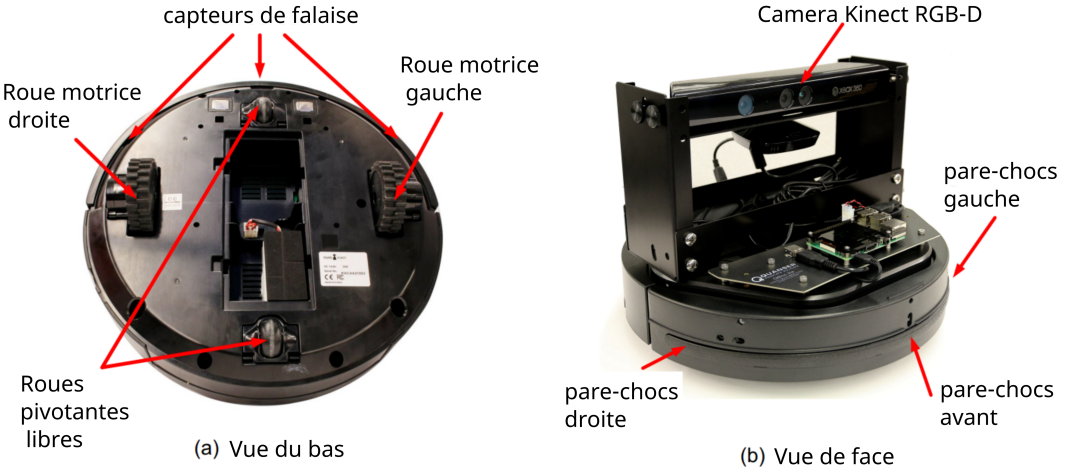

FIGURE 1.8 – Composents du Qbot 2e

dans un algorithme de contrôle pour éviter les obstacles ou prévenir les dommages au robot.

1.10 Problématique

Les problèmes de contrôle appropriés à la navigation d'un robot mobile suivant une trajectoire planifiée à base du modèle de l'environnement pour le positionner en un point déterminé, ou pour suivre une trajectoire prédéfinie parfaitement, ont attiré beaucoup d'attention ces dernières années. Certains solutions ont mis l'accent sur des approches de contrôle à base des capteurs pour interactive le mouvement du robot mobile dans des environnements connus ou inconnus, statiques ou dynamiques à l'aide d'un radar, la vidéo et le flux optique. Néanmoins, la commande du robot mobile est soumise à plusieurs contraintes liées à l'évitement d'obstacles et le changement de la dynamique du robot en fonction de l'état du route dans laquelle il évolue, mise en œuvre matérielle, problèmes d'alimentation et d'étalonnage, imprécision des capteurs, ...etc. Ceci, provoquent des fois plusieurs défauts qui peuvent être représentés par la saturation des actionneurs à cause de l'amplification des gains de commande et la défaillance de certains capteurs. C'est la raison pour laquelle nous nous sommes intéressés dans la cadre de ce travail à la commande prédictive stable et robuste aux erreurs de navigations, elle peut être décrite par un problème de contrôle optimal, en utilisant un algorithme d'optimisation pour calculer les séquences des futures entrées de commande, à base des erreurs de navigation représentées par un modèle de prédiction pour générer les prédictions à optimiser, en minimisant une fonction objective dans le but de satisfaire les contraintes assurant la stabilité du système à commander.

1.11 Conclusion

Ce chapitre présente l'état de l'art sur les robots mobiles à roues. Tout d'abord, il présente le contexte des WMR et quelques définitions majeures dans le domaine de la commande des WMR. Ensuite, une revue des algorithmes de commande de suivi de trajectoire et d'évitement d'obstacles est présentée. Pour chaque section, une conclusion de l'examen est mise en évidence.

Pour le suivi de trajectoire, il s'agit toujours d'un problème ouvert ; toutes les méthodes de commande mentionnées dans la littérature ont des avantages et des inconvénients. Jusqu'à présent, aucune d'entre elles ne peut résoudre le problème parfaitement. La régularité et la vitesse de convergence sont les principaux facteurs significatifs dans le problème du suivi de trajectoire. En effet, une méthode de suivi parfaite doit atteindre ces deux objectifs simultanément.

L'algorithme de commande de suivi de trajectoire peut être basé sur la cinématique ou la dynamique du robot mobile. Les systèmes de commande basés sur la cinématique du robot mobile sont suffisants en cas de faible vitesse et de faible poids. Cependant, lorsqu'un mouvement à grande vitesse et/ou une charge lourde sont requis, il est essentiel de considérer la dynamique du robot en plus de sa cinématique. Dans ce cas, le système de commande sera plus compliqué. La complexité du modèle dynamique est due à la prise en compte de la cinématique et de la dynamique, ainsi que certains paramètres inconnus et certaines incertitudes paramétriques telles que les glissements latéral et longitudinal. Comme certains de ces paramètres étant inconnus, la commande doit être améliorée pour estimer et identifier ces paramètres inconnus. Ainsi, la commande floue, la commande neuronale et la commande adaptative sont efficaces pour estimer ces paramètres et ces incertitudes dans le but de résoudre le problème du suivi de trajectoire. La commande prédictive de modèle peut également être utilisée pour le suivi de trajectoire. Néanmoins, elle présente quelques problèmes de mise en œuvre en raison de la non-linéarité du système. Pour résoudre ce problème, un modèle linéaire approximatif peut être appliqué. Une nouvelle tendance actuelle consiste à utiliser une commande basée sur l'apprentissage. Une commande prédictive de modèle basée sur l'apprentissage (LBMPC) n'a été appliquée qu'à un hélicoptère quadrotor et n'a pas été appliquée jusqu'à présent aux WMR.

Pour la question de l'évitement des obstacles, les stratégies de commande les plus utilisées, sont basées principalement sur la cinématique du robot mobile, les approches du champ de potentiel et les règles floues avec des capteurs infrarouges ou ultrasoniques. Ces méthodes se concentrent sur la recherche d'un chemin vers la position désirée en tenant compte des contraintes dynamiques différentielles.

Depuis quelques années, de nombreux chercheurs à formuler le problème de l'évitement d'obstacles en utilisant des techniques d'optimisation, dans le but de trouver un chemin optimal en ligne, même en présence d'une dynamique non linéaireet des contraintes complexes. De même, l'utilisation d'un algorithme de commande basé sur la vision a récemment présenté de nombreux avantages, mais il s'agit toujours d'un domaine difficile.

Enfin, le contrôle est une chose fascinante à faire avec les robots, mais il ne peut être fait que si nous avons un modèle mathématique. C'est la raison pour laquelle la modélisation du robot mobile à entraînement différentiel fera l'objet du prochain chapitre.

CHAPITRE 2

Modèle mathématique du robot mobile Qbot2e à entraînement différentiel

2.1 Introduction

Un robot à entraînement différentiel est un robot doté de deux roues commandables. Pour manœuvrer un robot à entraînement différentiel dans un plan, le robot a besoin d'une vitesse linéaire v et d'un sens d'orientation θ. En contrôlant la vitesse et l'orientation, il est possible de planifier la trajectoire du robot.

Ce chapitre présente les modèles cinématique et dynamique d'un robot mobile à roues, utilisé dans cette thèse, à savoir le Qbot 2e. On commence à donner des informations générales sur la relation entre les repères et leurs matrices de rotation, puis sur les contraintes nonholonomiques qui affectent le mouvement du robot, et enfin sur les modèles cinématiques et dynamiques qui jouent un rôle important dans la résolution les problèmes de commande.

2.2 Modèle du robot mobile Qbot2e

2.2.1 Systèmes de coordonnées

Afin de décrire la position du WMR dans son environnement, deux systèmes de coordonnées (repères) différents doivent être définis.

— Système de coordonnées inertiel (Allocentrique) : Ce système de coordonnées est un repère global qui est fixé dans l'environnement ou le plan dans lequel le WMR se déplace. De plus, ce repère est considéré comme le repère de référence. et est désigné par $\vec{X}_I, \vec{Y}_I$

— Système de coordonnées du robot (Égocentrique) : Ce système de coordonnées est un repère local attaché au WMR, et qui se déplace donc avec lui. Ce repère est dénoté comme $\vec{X}_M, \vec{Y}_M$

Les deux repères définis sont représentés sur la Figure (2.1). L'origine du repère du robot est définie comme étant le point central P sur l'axe entre les roues. Le centre de masse C du robot est supposé sur l'axe de symétrie, à une distance h de l'origine P.

Comme l'illustre la Figure (2.1), la position et l'orientation du robot dans le repère inertiel peuvent être définies comme suit :

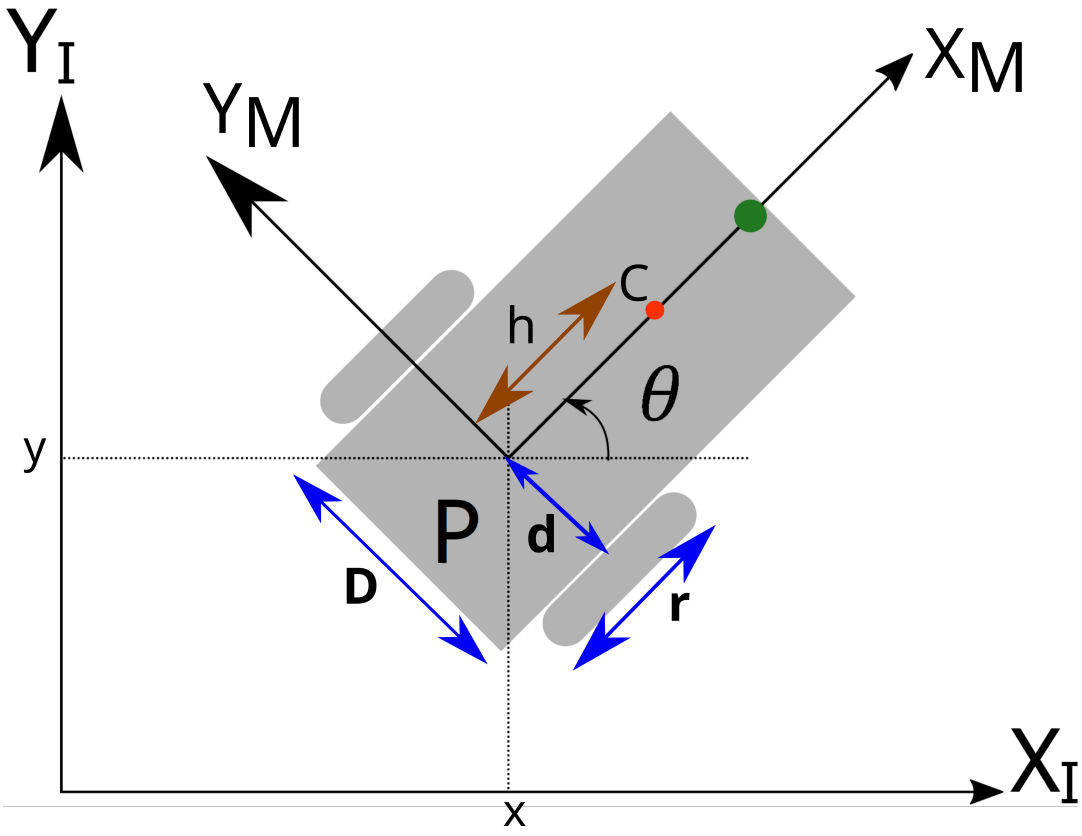

FIGURE 2.1 – Le robot mobile à entraînement différentiel dans la référence globale et locale

$$q^I = [x, y, \theta]^T \tag{2.1}$$

La question importante qui doit être expliquée à ce stade est le mappage entre ces deux repères. La position d'un point quelconque du robot peut être définie dans le repère du robot et le repère inertiel comme suit.

Supposons que $Q^M = [x^M, y^M, \theta^M]^T$, et $Q^I = [x^I, y^I, \theta^I]^T$ sont les coordonnées du point donné dans le repère du robot et le repère inertiel, respectivement.

Alors, les deux coordonnées sont liées par la transformation suivante :

$$Q^I = R(\theta)Q^M \tag{2.2}$$

Où $R(\theta)$ est la matrice de rotation orthogonale

$$R(\theta) = \begin{bmatrix} cos(\theta) & -sin(\theta) & 0 \\ sin(\theta) & cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.3}$$

Cette transformation permettra également de traiter le mouvement entre les repères. Donc :

$$\dot{Q}^I = R(\theta)\dot{Q}^M \tag{2.4}$$

Nous verrons dans la suite que l'équation (2.4) est très importante pour la dérivation des modèles cinématique et dynamique du WMR car elle décrit la relation entre les vitesses dans le repère inertiel et le repère local.

2.2.2 Contraintes de mouvement

Le mouvement d'un robot mobile à entraînement différentiel est caractérisé par deux équations de contraintes nonholonomiques, qui sont obtenues par deux hypothèses principales :

— La contrainte empêchant les déplacements latérales : Cette contrainte signifie simplement que le robot ne peut se déplacer que dans un mouvement courbe (en avant et en arrière) mais pas latéralement. Dans le repère du robot, cette condition signifie que la vitesse du point central P est nulle le long de l'axe latéral, donc :

$$\dot{y}_p^M = 0 \tag{2.5}$$

En utilisant la matrice de rotation orthogonale $R(\theta)$, la vitesse dans le repère inertiel est la suivante :

$$\dot{y}_p cos(\theta) - \dot{x}_p sin(\theta) = 0 \tag{2.6}$$

— La contrainte de roulement pur : Cette contrainte représente le fait que chaque roue maintient un seul point de contact ρ avec le sol, comme le montre la Figure (2.2). Il n'y a pas de glissement de la roue dans son axe longitudinal $(\vec{X}_M)$ ni de dérapage dans son axe orthogonal $(\vec{Y}_M)$. Les vitesses des points de contact dans le repère du robot sont liées aux vitesses des roues par la formule (2.7) :

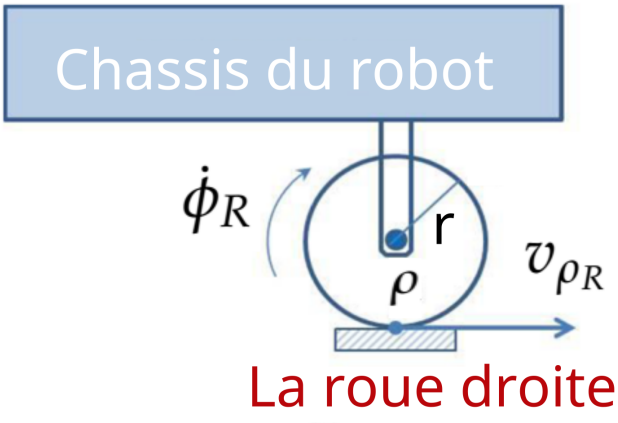

FIGURE 2.2 – La contrainte de roulement pur

$$\begin{cases} v_{\rho_R} = r\dot{\phi}_R \\ v_{\rho_L} = r\dot{\phi}_L \end{cases} \tag{2.7}$$

Dans le repère inertiel, ces vitesses peuvent être calculées en fonction des vitesses du point central P du robot :

$$\begin{cases} \dot{x}_{\rho_R} = \dot{x}_p + d\dot{\theta}cos(\theta) \\ \dot{y}_{\rho_R} = \dot{y}_p + d\dot{\theta}sin(\theta) \end{cases} \tag{2.8}$$

$$\begin{cases} \dot{x}_{\rho_L} = \dot{x}_p - d\dot{\theta}cos(\theta) \\ \dot{y}_{\rho_L} = \dot{y}_p - d\dot{\theta}sin(\theta) \end{cases} \tag{2.9}$$

Où : d est la distance des roues par rapport à leur axe central commun P.
En utilisant la matrice de rotation $R(\theta)$, les équations de contrainte de roulement sont formulées comme suit :

$$\begin{cases} \dot{x}_{\rho_R}cos(\theta) + \dot{y}_{\rho_R}sin(\theta) = r\dot{\phi_R} \\ \dot{x}_{\rho_L}cos(\theta) + \dot{y}_{\rho_L}sin(\theta) = r\dot{\phi_L} \end{cases} \tag{2.10}$$

Maintenant, en substituant (2.8) et (2.9) dans (2.10), on obtient :

$$\begin{cases} (\dot{x}_p + d\dot{\theta}cos(\theta))cos(\theta) + (\dot{y}_p + d\dot{\theta}sin(\theta))sin(\theta) = r\dot{\phi_R} \\ (\dot{x}_p - d\dot{\theta}cos(\theta))cos(\theta) + (\dot{y}_p - d\dot{\theta}sin(\theta))(\theta) = r\dot{\phi_L} \end{cases} \tag{2.11}$$

Menant à :

$$\begin{cases} \dot{x}_pcos(\theta) + d\dot{\theta}cos(\theta)cos(\theta) + \dot{y}_psin(\theta) + d\dot{\theta}sin(\theta)sin(\theta) - r\dot{\phi}_R = 0 \\ \dot{x}_pcos(\theta) - d\dot{\theta}cos(\theta)cos(\theta) + \dot{y}_psin(\theta) - d\dot{\theta}sin(\theta)sin(\theta) - r\dot{\phi}_L = 0 \end{cases} \tag{2.12}$$

Donc

$$\begin{cases} \dot{x}_pcos(\theta) + \dot{y}_psin(\theta) + d\dot{\theta} - r\dot{\phi}_R = 0 \\ \dot{x}_pcos(\theta) + \dot{y}_psin(\theta) - d\dot{\theta} - r\dot{\phi}_L = 0 \end{cases} \tag{2.13}$$

les trois équations de contrainte peuvent être écrites dans la forme matricielle suivante :

$$\Lambda(q)\dot{q} = 0 \tag{2.14}$$

Où

$$\Lambda(q) = \begin{bmatrix} -sin(\theta) & cos(\theta) & 0 & 0 & 0 \\ cos(\theta) & sin(\theta) & d & -r & 0 \\ cos(\theta) & sin(\theta) & -d & 0 & -r \end{bmatrix} \tag{2.15}$$

et

$$\dot{q} = \begin{bmatrix} \dot{x}_p & \dot{y}_p & \dot{\theta} & \dot{\phi}_R & \dot{\phi}_L \end{bmatrix} \tag{2.16}$$

et

$$\begin{cases} v_R = r\dot{\phi}_R \\ v_L = r\dot{\phi}_L \end{cases} \tag{2.17}$$

La matrice de contraintes ci-dessus $\Lambda(q)$ q sera utilisée dans la section suivante pour la modélisation dynamique du WMR.

2.2.3 Modèle cinématique

La modélisation cinématique se concentre sur l'étude du mouvement des systèmes mécaniques sans prendre en compte les forces qui influencent ce mouvement. Dans le cas d'un WMR (Robot Mobile à Roues), l'objectif principal de la modélisation cinématique est de décrire les vitesses du robot en fonction des vitesses de rotation des roues motrices et des paramètres géométriques du robot.

La moyenne de la vitesse linéaire de chaque roue motrice dans le repère du robot est donc la vitesse linéaire du WMR dans le repère du robot.

$$v = \frac{v_R + v_L}{2} = r\frac{(\dot{\phi}_R + \dot{\phi}_L)}{2} \tag{2.18}$$

Par conséquent, la vitesse angulaire du robot mobile est donnée par :

$$\omega = \frac{v_R - v_L}{D} = \frac{r}{D}(\dot{\phi}_R - \dot{\phi}_L) \tag{2.19}$$

Les vitesses du robot mobile dans le repère du robot peuvent maintenant être représentées en termes de vitesses du point central P dans le repère du robot comme suit :

$$\begin{cases} \dot{x}_p^M = r\frac{(\dot{\phi}_R+\dot{\phi}_L)}{2} \\ \dot{y}_p^M = 0 \\ \dot{\theta} = \omega = \frac{r}{D}\frac{(\dot{\phi}_R-\dot{\phi}_L)}{2} \end{cases} \tag{2.20}$$

Ainsi,

$$\begin{bmatrix} \dot{x}_p^M \\ \dot{y}_p^M \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \frac{r}{2} & \frac{r}{2} \\ 0 & 0 \\ \frac{r}{D} & -\frac{r}{D} \end{bmatrix} \begin{bmatrix} \dot{\phi}_R \\ \dot{\phi}_L \end{bmatrix} \tag{2.21}$$

Les vitesses du robot mobile peuvent également être obtenues dans le repère inertiel comme suit :

$$\dot{q}^I = \begin{bmatrix} \dot{x}_p^M \\ \dot{y}_p^M \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \frac{r}{2}cos(\theta) & \frac{r}{2}cos(\theta) \\ \frac{r}{2}sin(\theta) & \frac{r}{2}sin(\theta) \\ \frac{r}{D} & -\frac{r}{D} \end{bmatrix} \begin{bmatrix} \dot{\phi}_R \\ \dot{\phi}_L \end{bmatrix} \tag{2.22}$$

L'équation (2.22) représente la cinématique directe différentielle d'un robot mobile à deux roues à entraînement différentiel. Une autre forme alternative pour le modèle cinématique peut être obtenue en représentant les vitesses du WMR en termes de vitesses linéaires et angulaires du WMR dans le repère du robot.

$$\dot{q}^I = \begin{bmatrix} \dot{x}_p^M \\ \dot{y}_p^M \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} cos(\theta) & 0 \\ sin(\theta) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{2.23}$$

2.2.4 Modèle dynamique

La dynamique d'un robot est essentielle dans la conception d'un contrôleur pour le robot. La modélisation dynamique en général est l'étude du mouvement approprié au système dans lequel les forces ou couples de commande imposés par les actionneurs sont modélisés en tenant compte des énergies et des vitesses associées aux mouvements. La principale différence entre les deux modélisations, dynamique et cinématique est qu'en modélisation cinématique, nous étudions le mouvement sans tenir compte des forces ou couples qui l'affectent et nous nous contentons de traiter les relations géométriques du système.

Le modèle dynamique d'un robot mobile joue un rôle crucial dans l'analyse et la simulation du mouvement du robot, ainsi que dans la conception d'algorithmes de contrôle du mouvement. Il fournit une représentation mathématique des forces et des moments qui influencent le mouvement du robot, permettant ainsi d'étudier et de prédire son comportement dans différentes situations. Ce modèle dynamique est essentiel pour améliorer la compréhension du mouvement du robot et pour développer des stratégies de contrôle efficaces.

En outre un robot mobile ayant un espace de configuration L à n dimensions avec des coordonnées généralisées $(q_1; q_2; ...; q_n)$ et soumis à m contraintes, son modèle dynamique peut être décrit par l'équation dynamique générale suivante :

$$M_{In}(q)\ddot{q} + V(q, \dot{q}) + F(\dot{q}) + G(q) = B(q)\tau - \Lambda^T(q)\lambda \tag{2.24}$$

Où :
— $M_{In}(q)$ est la matrice d'inertie symétrique définie positive
— $V(q; \dot{q})$ est la matrice centripète et de coriolis
— $F(\dot{q})$ est la matrice de friction de surface
— $G(q)$ est le vecteur gravitationnel
— $B(q)$ est la matrice de transformation d'entrée
— τ est le vecteur d'entrée
— $\Lambda^T(q)$ est la matrice associée aux contraintes cinématiques
— λ est le vecteur de forces des multiplicateurs de Lagrange

L'approche Lagrangienne basée sur l'énergie est utilisée pour exprimer la dynamique de ce robot. En effet, la formalisme mathématique de Lagrange, permet d'obtenir systématiquement les équations du mouvement en considérant les énergies cinétique et potentielle du système à modéliser.

L'équation de Lagrange peut être écrite sous la forme suivante :

$$\frac{d}{dt}(\frac{\partial L}{\partial \dot{q}_i}) - \frac{\partial L}{\partial q_i} = \sum_{j=1}^{n} \lambda_j a_{ji} + Q_i \qquad i = 1, 2, ...n \tag{2.25}$$

Où :
— $q_1; q_2; ...q_n$ sont les coordonnées généralisées

— $L = T - V$ est le Lagrangien qui est calculé à partir de la différence entre l'énergie cinétique et l'énergie potentielle du système.
— λj est le multiplicateur lagrangien qui relie les contraintes aux forces de contrainte.
— Q_i est la force non conservatrice du système.

En plus de ces n équations, nous avons m équations des contraintes à résoudre, ce qui donne $(m + n)$ inconnues. La première étape pour trouver l'équation dynamique consiste à trouver les énergies cinétique et potentielle du système.

Étant donné que le mouvement est limité au sol, l'énergie potentielle du système est nulle.

La fonction d'énergie cinétique du robot peut être dérivée en fonction des vitesses indiquées dans la Figure (2.3) .

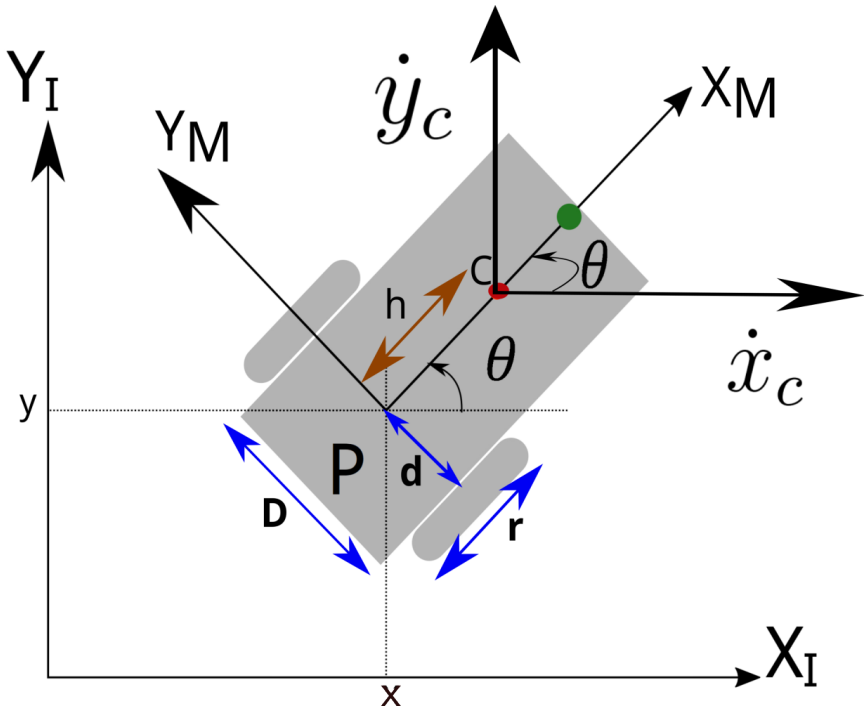

FIGURE 2.3 – Diagramme de corps libre de robot pour la modélisation dynamique de Lagrange

Pour trouver la vitesse des points P et C, considérant les coordonnées du point C données par :

$$x_c = x_p + hcos(\theta) \tag{2.26}$$

$$y_c = y_p + hsin(\theta) \tag{2.27}$$

Par conséquent :

$$\dot{x}_c = \dot{x}_p - h\dot{\theta}sin(\theta) \tag{2.28}$$

$$\dot{y}_c = \dot{y}_p + h\dot{\theta}cos(\theta) \tag{2.29}$$

En réarrangeant l'équation (2.28) et l'équation (2.29), la vitesse du centre de rotation P du robot mobile sera expriméé par :

$$v_p = (\dot{x}_c + h\dot{\theta}sin(\theta))\vec{i} + (\dot{y}_c - h\dot{\theta}cos(\theta))\vec{j} \tag{2.30}$$

L'énergie cinétique du robot mobile est la somme de l'énergie cinétique de la plate-forme du robot mobile et les deux roues droite et gauche qui sont données respectivement par ces expressions :

$$T = \frac{1}{2}M_c v_p^2 + \frac{1}{2}I_c\dot{\theta}^2 \tag{2.31}$$

$$T = \frac{1}{2}M_w v_{wR}^2 + \frac{1}{2}I_m\dot{\theta}^2 + \frac{1}{2}I_w\dot{\phi}_R^2 \tag{2.32}$$

$$T = \frac{1}{2}M_w v_{wL}^2 + \frac{1}{2}I_m\dot{\theta}^2 + \frac{1}{2}I_w\dot{\phi}_L^2 \tag{2.33}$$

où, M_c et I_c sont respectivement la masse et le moment d'inertie par rapport au centre de gravité du robot mobile, appropriés au chassis de ce dernier. M_w et I_w sont respectivement la masse et le moment d'inertie par rapport au centre de la roue pour chaque roue motrice (avec actionneur), I_m est le moment d'inertie de chaque roue motrice par rapprt au centre de gravité du robot mobile.

Toutes les vitesses sont d'abord exprimées en fonction des coordonnées généralisées en utilisant l'équation générale de la vitesse dans le repère inertiel.

$$v_i^2 = \dot{x}_i^2 + \dot{y}_i^2 \tag{2.34}$$

Les composantes x_c et y_c du centre de masse et des roues droite et gauche peuvent être obtenues en termes de coordonnées généralisées comme suit :

$$\begin{cases} x_c = x_p + hcos(\theta) \\ y_c = y_p + hsin(\theta) \end{cases} \tag{2.35}$$

$$\begin{cases} x_{wR} = x_p + dcos(\theta) \\ y_{wR} = y_p - dsin(\theta) \end{cases} \tag{2.36}$$

$$\begin{cases} x_{wL} = x_p - dcos(\theta) \\ y_{wL} = y_p + dsin(\theta) \end{cases} \tag{2.37}$$

En utilisant les équations (2.31)-(2.37), l'énergie cinétique totale du robot mobile est exprimée par la relation suivante :

$$T = \frac{1}{2}M_a(\dot{x}_p^2 + \dot{y}_p^2) + M_c h\dot{\theta}(\dot{y}_p cos(\theta) - \dot{x}_p sin(\theta) + \frac{1}{2}I_w(\dot{\phi}_R^2 + \dot{\phi}_L^2) + \frac{1}{2}I\dot{\theta}^2 \quad (2.38)$$

où les nouveaux paramètres suivants sont introduits
— $M_a = M_c + 2M_w$ est la masse totale du robot,
— $I = I_c + M_c h^2 + 2M_w d^2 + 2I_m$ est l'inertie totale équivalente
Maintenant, nous utilisons l'équation (2.25) pour trouver les équations dynamiques en utilisant les coordonnées généralisées et le lagrangien ci-dessus comme suit :

$$\frac{\partial L}{\partial \dot{x}_p} = M_a\dot{x}_p - M_c h\dot{\theta}sin(\theta) \quad (2.39)$$
$$\frac{\partial L}{\partial \dot{y}_p} = M_a\dot{y}_p + M_c h\dot{\theta}cos(\theta) \quad (2.40)$$
$$\frac{\partial L}{\partial \dot{\theta}} = M_c h(\dot{y}_p cos(\theta) - \dot{x}_p sin(\theta)) + I\dot{\theta} \quad (2.41)$$
$$\frac{\partial L}{\partial \dot{\phi}_R} = I_w\dot{\phi}_R \quad (2.42)$$
$$\frac{\partial L}{\partial \dot{\phi}_L} = I_w\dot{\phi}_L \quad (2.43)$$
$$\frac{\partial L}{\partial x_p} = 0 \quad (2.44)$$
$$\frac{\partial L}{\partial y_p} = 0 \quad (2.45)$$
$$\frac{\partial L}{\partial \theta} = -M_c h\dot{\theta}(\dot{x}_p cos(\theta) + \dot{y}_p sin(\theta)) \quad (2.46)$$
$$\frac{\partial L}{\partial \phi_R} = 0 \quad (2.47)$$
$$\frac{\partial L}{\partial \phi_L} = 0 \quad (2.48)$$
$$\frac{d}{dt}(\frac{\partial L}{\partial \dot{x}_p}) = M_a\ddot{x}_p - M_c h\ddot{\theta}sin(\theta) - M_c h\dot{\theta}^2 cos(\theta) \quad (2.49)$$
$$\frac{d}{dt}(\frac{\partial L}{\partial \dot{y}_p}) = M_a\ddot{y}_p + M_c h\ddot{\theta}cos(\theta) - M_c h\dot{\theta}^2 sin(\theta) \quad (2.50)$$
$$\frac{d}{dt}(\frac{\partial L}{\partial \dot{\theta}}) = M_c h(\ddot{y}_p cos(\theta) - \ddot{x}_p sin(\theta)) - M_c h\dot{\theta}(\dot{y}_p sin(\theta) + \dot{x}_p cos(\theta)) + I\ddot{\theta} \quad (2.51)$$
$$\frac{d}{dt}(\frac{\partial L}{\partial \dot{\phi}_R}) = I_w\ddot{\phi}_R \quad (2.52)$$
$$\frac{d}{dt}(\frac{\partial L}{\partial \dot{\phi}_L}) = I_w\ddot{\phi}_L \quad (2.53)$$

En substituant les équations (2.39)-(2.53) et les équations qui y figurent dans l'équation de Lagrange (2.25), nous nous obtenons les équations de mouvement relatif au robot mobile :

$$M_a\ddot{x}_p - M_ch\ddot{\theta}sin(\theta) - M_ch\dot{\theta}^2cos(\theta) = C_1 \tag{2.54}$$

$$M_a\ddot{y}_p + M_ch\ddot{\theta}cos(\theta) - M_ch\dot{\theta}^2sin(\theta) = C_2 \tag{2.55}$$

$$I\ddot{\theta} - M_ch\ddot{x}_psin(\theta) + M_ch\ddot{y}_pcos(\theta) = C_3 \tag{2.56}$$

$$I_w\ddot{\phi}_R = \tau_R + C_4 \tag{2.57}$$

$$I_w\ddot{\phi}_L = \tau_L + C_5 \tag{2.58}$$

où $(C_1, C_2, C_3, C_4, C_5)$, sont des coefficients liés aux contraintes cinématiques, qui peuvent être écrits en termes de vecteur de multiplicateurs de Lagrange λ et de matrice de contraintes cinématiques Λ introduite dans (2.15)

$$\Lambda^T(q) = \begin{bmatrix} C_1 \\ C_2 \\ C_3 \\ C_4 \\ C_5 \end{bmatrix} \tag{2.59}$$

Maintenant, les équations de mouvement obtenues, de (2.54) à (2.58), peuvent être représentées sous la forme générale donnée par l'équation (2.24) comme suit

$$M_{In}(q)\ddot{q} + V(q,\dot{q}) = B(q)\tau - \Lambda^T(q)\lambda \tag{2.60}$$

où

$$M_{In}(q) = \begin{bmatrix} M_a & 0 & -M_chsin(\theta) & 0 & 0 \\ 0 & M_a & M_chcos(\theta) & 0 & 0 \\ -M_chsin(\theta) & M_chcos(\theta) & I & 0 & 0 \\ 0 & 0 & 0 & I_w & 0 \\ 0 & 0 & 0 & 0 & I_w \end{bmatrix} \tag{2.61}$$

$$V(q,\dot{q}) = \begin{bmatrix} 0 & 0 & -M_ch\dot{\theta}cos(\theta) & 0 & 0 \\ 0 & 0 & -M_ch\dot{\theta}sin(\theta) & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{2.62}$$

$$B(q) = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{2.63}$$

$$\Lambda^T(q)\lambda = \begin{bmatrix} -sin(\theta) & cos(\theta) & cos(\theta) \\ cos(\theta) & sin(\theta) & sin(\theta) \\ 0 & d & -d \\ 0 & -r & 0 \\ 0 & 0 & -r \end{bmatrix} \times \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \lambda_4 \\ \lambda_5 \end{bmatrix} \tag{2.64}$$

Ensuite, le système décrit par l'équation (2.60) est transformé en une forme alternative qui est plus pratique pour le contrôle et la simulation. L'objectif principal est d'éliminer le terme de contrainte $\Lambda^T\lambda$ dans l'équation (2.60) puisque les multiplicateurs de Lagrange λ_i sont inconnus. Cela se fait d'abord en définissant le vecteur réduit

$$\dot{\eta} = \begin{bmatrix} \dot{\phi}_R \\ \dot{\phi}_L \end{bmatrix} \tag{2.65}$$

Ensuite, en exprimant les vitesses des coordonnées généralisées à l'aide du modèle cinématique avant (2.22). Nous avons alors

$$\begin{bmatrix} \dot{x}_a \\ \dot{y}_a \\ \dot{\theta} \\ \dot{\phi}_R \\ \dot{\phi}_L \end{bmatrix} = \frac{1}{2} \begin{bmatrix} rcos(\theta) & rcos(\theta) \\ rsin(\theta) & rsin(\theta) \\ \frac{r}{d} & -\frac{r}{d} \\ 2 & 0 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} \dot{\phi}_R \\ \dot{\phi}_L \end{bmatrix} \tag{2.66}$$

Ceci peut être écrit sous la forme

$$\dot{q} = S(q)\eta \tag{2.67}$$

On peut vérifier que la matrice de transformation $S(q)$ est dans l'espace nul de la matrice de contrainte $\Lambda(q)$. Par conséquent, nous avons

$$S^T(q)\Lambda^T(q) = 0 \tag{2.68}$$

Ensuite, en prenant la dérivée temporelle de l'équation (2.67), on obtient

$$\ddot{q} = \dot{S}(q)\eta + S(q)\dot{\eta} \tag{2.69}$$

En substituant les équations (2.67) et (2.69) dans l'équation principale (2.60), on obtient

$$M_{In}(q)[\dot{S}(q)\eta + S(q)\dot{\eta}] + V(q,\dot{q})[S(q)\eta] = B(q)\tau - \Lambda^T(q)\lambda \tag{2.70}$$

Ensuite, en réarrangeant l'équation et en multipliant les deux côtés par $S^T(q)$, on obtient

$$S^T(q)M_{In}(q)S(q)\dot{\eta} + S^T(q)[M_{In}(q)\dot{S}(q)+ V(q,\dot{q})S(q)]\eta = S^T(q)B(q)\tau - S^T(q)\Lambda^T(q)\lambda \quad (2.71)$$

où le dernier terme est identiquement nul. Définissons maintenant les nouvelles matrices

$$\begin{cases} \overline{M}_{In} = S^T(q)M_{In}(q)S(q) \\ \overline{V} = S^T(q)M_{In}(q)\dot{S(q)} + S^T(q)V(q,\dot{q})S(q) \\ \overline{B} = S^T B(q) \end{cases} \quad (2.72)$$

Les équations dynamiques sont réduites à la forme compacte suivante :

$$\overline{M}(q)\dot{\eta} + \overline{V}(q,\dot{q})\eta = \overline{B}(q)\tau \quad (2.73)$$

où

$$\overline{M}(q) = \begin{bmatrix} I_w + \frac{r^2}{4d^2}(M_a d^2 + I) & \frac{r^2}{4d^2}(M_a d^2 - I) \\ \frac{r^2}{4d^2}(M_a d^2 - I) & I_w + \frac{r^2}{4d^2}(M_a d^2 + I) \end{bmatrix} \quad (2.74)$$

$$\overline{V}(q,\dot{q}) = \begin{bmatrix} 0 & \frac{r^2}{2d}M_c h\dot{\theta} \\ -\frac{r^2}{2d}M_c h\dot{\theta} & 0 \end{bmatrix} \quad (2.75)$$

$$\overline{B}(q) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad (2.76)$$

L'équation (2.73) montre que la dynamique du WMR est exprimée uniquement en fonction des vitesses angulaires des roues droite et gauche, de la vitesse angulaire du robot $\dot{theta}$ et des couples du moteur d'entraînement

Les équations de mouvement (2.73) peuvent également être transformées sous une autre forme représentée par les vitesses linéaires et angulaires (v, ω) du WMR. En utilisant les équations du modèle cinématique (2.18) et (2.19), on peut facilement montrer que les équations du modèle (2.73) peuvent être réarrangées sous la forme compacte suivante :

$$\begin{cases} (M_a + \frac{2I_w}{r^2})\dot{v} - M_c h\omega^2 = \frac{1}{r}(\tau_R + \tau_L) \\ (I + \frac{2d^2}{r^2}I_w)\dot{\omega} + M_c h\omega v = \frac{d}{r}(\tau_R - \tau_L) \end{cases} \quad (2.77)$$

Ensuite, ces deux équations peuvent être écrites sous forme matricielle comme suit

$$\begin{bmatrix} M_a + \frac{2I_w}{r^2} & 0 \\ 0 & I + \frac{2d^2}{r^2}I_w \end{bmatrix} \begin{bmatrix} \dot{v} \\ \dot{\omega} \end{bmatrix} + \begin{bmatrix} 0 & -M_c h\omega \\ M_c h\omega & M_c h v \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} = \frac{1}{r}\begin{bmatrix} 1 & 1 \\ d & -d \end{bmatrix} \begin{bmatrix} \tau_R \\ \tau_L \end{bmatrix} \quad (2.78)$$

Afin de représenter ce système sous forme d'espace d'états standard, nous pouvons réécrire les équations (2.78) sous la forme générale $\dot{x} = Ax + Bu$, où x est le vecteur d'état,A est la matrice d'état, B est la matrice d'entrée, et u est la commande. En effectuant cette transformation, nous obtenons :

$$\dot{x} = \begin{bmatrix} \dot{v} \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} M_a + \frac{2I_w}{r^2} & 0 \\ 0 & I + \frac{2d^2}{r^2} I_w \end{bmatrix}^{-1} \begin{bmatrix} 0 & M_c h\omega \\ -M_c h\omega & -M_c hv \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} + \frac{1}{r} \begin{bmatrix} M_a + \frac{2I_w}{r^2} & 0 \\ 0 & I + \frac{2d^2}{r^2} I_w \end{bmatrix}^{-1} \begin{bmatrix} 1 & 1 \\ d & -d \end{bmatrix} \begin{bmatrix} \tau_R \\ \tau_L \end{bmatrix} \quad (2.79)$$

Ce qui se traduit par :

$$\dot{x} = \begin{bmatrix} \dot{v} \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} 0 & \frac{M_c hr^2 w}{M_a r^2 + 2I_w} \\ -\frac{M_c hr^2 w}{2I_w d^2 + Ir^2} & -\frac{M_c hr^2 v}{2I_w d^2 + Ir^2} \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} + \begin{bmatrix} \frac{r^2}{M_a r^2 + 2I_w} & \frac{r^2}{M_a r^2 + 2I_w} \\ -\frac{dr^2}{2I_w d^2 + Ir^2} & \frac{dr^2}{2I_w d^2 + Ir^2} \end{bmatrix} \begin{bmatrix} \tau_R \\ \tau_L \end{bmatrix} \quad (2.80)$$

2.2.5 Exemple du comportement

Une validation initiale du modèle obtenu ci-dessus a été réalisée par calcul pour des scénarios où nous pouvons anticiper le comportement du système réel. Les premières valeurs des variables d'état en régime permanent ont été déterminées pour différentes combinaisons de paramètres et de tensions d'alimentation du moteur.

Les valeurs des paramètres énumérés dans les tableaux (2.1) et (2.2) sont utilisées dans tous les calculs. Ces valeurs sont sélectionnées de manière à correspondre, au moins approximativement, aux valeurs dérivées du manuel livré avec le robot mobile Qbot2e de Quanser. Les spécifications géométriques et autres du châssis sont répertoriées dans le Tableau 2.1.

TABLE 2.1 – Paramètres du robot

Notation	Valeur	Dimension	Signification
M_w	0.2	Kg	La masse de roue
M_c	4.5	Kg	La masse de châssis
h	0.01	m	distance entre le centre des roues et le centre de masse du châssis
l	0.235	m	la distance roue à roue
I	3	$Kg.m^2$	Moment d'inertie total
r	0.15	m	diamètre des roues

Les simulations ont été effectuées à l'aide de l'environnement *MATLAB*.

Quatre scénarios ont été exécutés pour montrer les quatres types de mouvement que le robot peut effectuer, à savoir vers l'avant, l'arrière, dans le sens des aiguilles d'une montre et dans le sens contraire des aiguilles d'une montre.

2.2.5.1 Résultats du scénario 1

TABLE 2.2 – Paramètres des simulations

Scénario	Temps de simulation	Condition initiale	Couple des moteurs $\tau = \begin{pmatrix} \tau_R \\ \tau_L \end{pmatrix}$
1	$2s$	$q_0 = [0,0,0]^T$	$\tau = [2,2]^T$
2	$2s$	$q_0 = [0,0,0]^T$	$\tau = [1.5,2]^T$
3	$2s$	$q_0 = [0,0,0]^T$	$\tau = [2,1.5]^T$
4	$2s$	$q_0 = [0,0,0]^T$	$\tau = [-2,-2]^T$

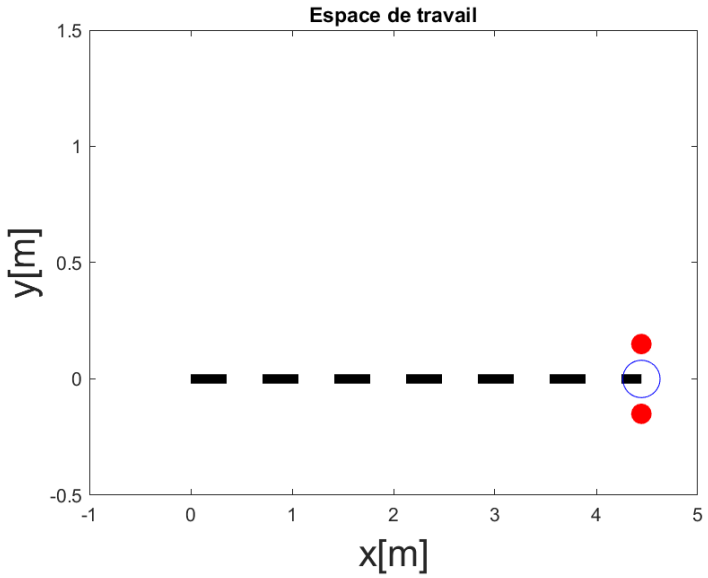

FIGURE 2.4 – Resultats du scénario 1

Comme nous pouvons le voir sur la Figure (2.4), le robot avance comme prévu car les couples des roues gauche et droite sont identiques.

2.2.5.2 Résultats du scénario 2

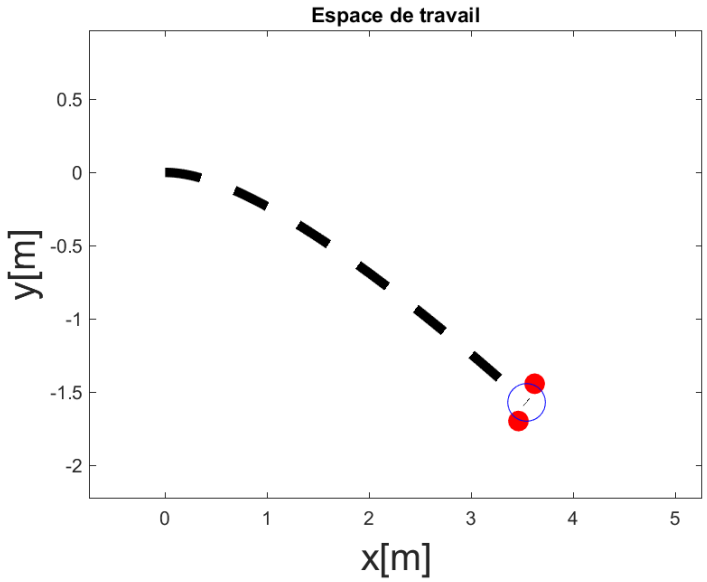

FIGURE 2.5 – Resultats du scénario 2

Comme nous pouvons le voir sur la Figure (2.5), le robot tourne dans le sens des aiguilles d'une montre comme prévu car le couple de la roue gauche est supérieur à celui de la roue droite.

2.2.5.3 Résultats du scénario 3

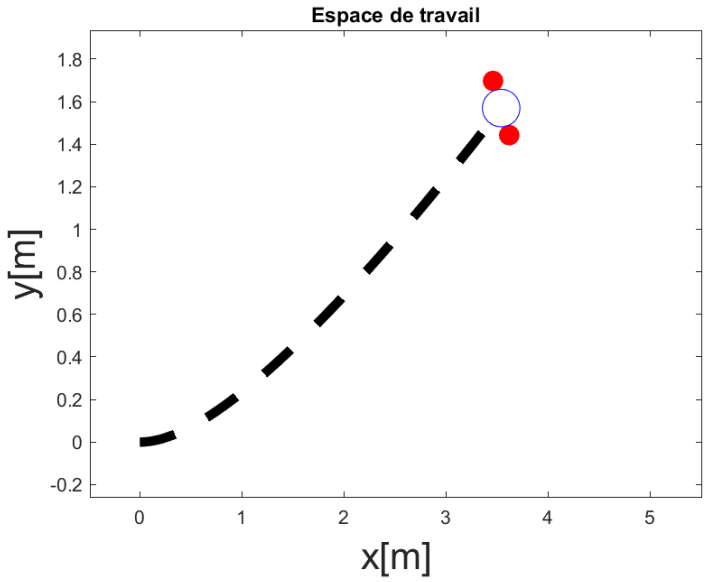

FIGURE 2.6 – Resultats du scénario 3

Comme vous pouvez le voir sur la Figure (2.6), le robot tourne dans le sens inverse des aiguilles d'une montre comme prévu car le couple de la roue droite est supérieur à celui de la roue gauche.

2.2.5.4 Résultats du scénario 4

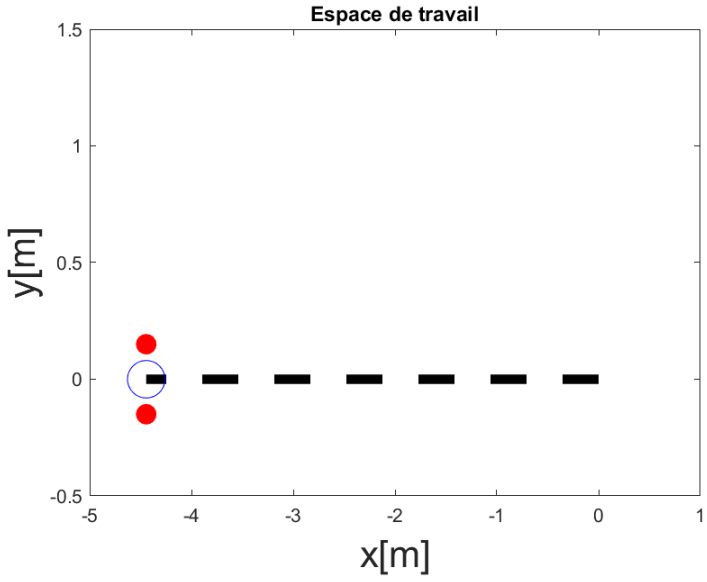

FIGURE 2.7 – Resultats du scénario 4 (Marche arrière)

Comme nous pouvons le voir sur la Figure (2.7), le robot recule comme prévu car les couples des roues gauche et droite sont identiques et negatives.

2.3 Conclusion

Dans ce chapitre, nous avons présenté en détaille les formalismes mathématiques appropriés aux modèles cinématique et dynamique d'un robot mobile à entraînement différentiel. La modélisation dynamique de ce dernier a été effectuée en utilisant la méthode de Lagrange. Cette modélisation a permis de mettre le caractère cohérent avec le comportement attendu. D'autr part, elle peut être une étape vers la synthèse des contrôleurs adaptés à la navigation et au suivi de trajectoire du robot mobile à entraînement différentiel, qui feront l'objet des deux prochains chapitres.

CHAPITRE 3

Commande stabilisante Pour Le Robot Mobile Qbot2e

3.1 Introduction

Historiquement, les premières tentatives d'approche pour le suivi de trajectoire des robots mobiles à entraînement différentiel, sont basées sur l'utilisation de la cinématique. Bien que son efficacité soit reconnue dans des situations particulières, elle présente des défauts majeurs qui la condamnent comme outil général, car le modèle cinématique non linéaire ne suffira pour la conception du contrôle que si la boucle de vitesse interne (dynamique) est plus rapide que la boucle de contrôle externe [70] ; Il s'agit d'un problème important lorsqu'on utilise des contrôleurs cinématiques pour des robots qui ne disposent pas d'un contrôleur de bas niveau, comme les robots mobiles contrôlés par la carte Arduino.

Dans ce chapitre, MATLAB/SIMULINK est utilisé en vue de déterminer l'impact du modèle dynamique du robot mobile sur le comportement du contrôleur contrôle par retour d'état asymptotiquement stable au sens de lyapunov (contrôleur cinématique pour stabiliser la position), ainsi qu'un contrôleur neuro-flou (ANFIS) développés par rapport au problème de suivi de trajectoire. Nous incorporons un contrôleur dynamique pour chacun des contrôleurs proposés afin de contrôler les vitesses du robot Par conséquent les tests de simulation et expérimentaux effecués, nous ont permis de déduire que ce contrôleur proposé a une excellente caractéristique dynamique, simple, réponse rapide, capacité stable pour le suivi de trajectoire, et erreur de suivi négligeable. L'approche de contrôle est vérifiée en utilisant l'analyse de stabilité de Lyapunov [19].

De plus, pour le contrôleur à rétroaction , une comparaison entre deux stratégies de contrôle dans le calcul de leur contrôleur, dont l'une utilise le modèle dynamique et l'autre non, montre que l'erreur de suivi est moins bonne dans le cas où la stratégie de contrôle ne prend pas en compte le modèle dynamique, car plusieurs perturbations et phénomènes non linéaires ne sont pas pris en compte, confirmant ainsi la supériorité de l'approche proposée en termes de précision, avec une différence négligeable dans les calculs [19].

Pour toutes ces raisons, nous avons dû essayer cette technique sur un vrai robot afin de mesurer ses performances. Nous présenterons donc notre travail et montrerons les résultats de la simulation, ainsi que les résultats expérimentaux sur le robot mobile Qbot2e.

Les résultats expérimentaux n'étant pas aussi bons que les résultats de la simulation, nous avons découvert que le problème venait des gains du contrôleur proposé, et nous avons donc décidé d'utiliser une technique d'adaptation des gains à la place du contrôleur cinématique à rétroaction. Nous avons donc décidé d'utiliser une technique d'adaptation du gain au lieu du contrôleur cinématique à rétroaction. Le contrôleur que nous avons proposé est la technique du système d'inférence neuro-floue adaptatif (ANFIS). Nous discutons brièvement de ses origines, de ses techniques, de son algorithme d'apprentissage et de son étude de stabilité.

Enfin, nous fournissons des résultats expérimentaux qui valident la supériorité du nouveau contrôleur par rapport au contrôleur précédent.

Selon [4, 71], les systèmes non-holonomiques ne peuvent pas être stabilisés par une loi de contrôle de retour d'état continuellement différentiable et invariable dans le temps. Cela indique que ce problème est véritablement non linéaire. Par conséquent, pour répondre au problème de la complexité de la dynamique du robot mobile dans l'élaboration des lois de commande, il s'avère nécessaire d'opter pour des techniques innovantes.

Puisque nous travaillons sur un robot mobile de classe non holonomique qui utilise un entraînement différentiel, l'objectif principal de ce chapitre est d'exploiter le modèle dynamique développé dans le chapitre 2 dans la synthèse d'une stratégie de contrôle appropriée pour le robot mobile, en utilisant un contrôleur cinématique qui est asymptotiquement stable, et un contrôleur dynamique basé sur la conception de contrôle par le modèle inverse inclus dans notre article [19].

Puis, la présentation de la technique de régulation ANFIS, à savoir, sa technique d'apprentissage, et son étude de stabilité.

Ensuite, une application du régulateur ANFIS sur le contrôle suggéré est utilisée pour la commande des vitesses du robot.

Dans la deuxième partie, nous illustrerons les résultats de simulation du notre article, ainsi que les résultats expérimentaux du même contrôleur sur le robot mobile Qbot2e.

Enfin, une illustration des résultats expérimentaux de l'ANFIS est présentée pour conclure ce chapitre.

3.2 Conception du contrôleur stabilisant suggéré

La stabilisation d'un point fait référence à la stabilisation du robot par rapport à ses coordonnées cartésiennes et son orientation préalablement connues. Le suivi de trajectoire fait référence au déplacement d'un système robotique dans un itinéraire indépendamment du temps.

Le problème du suivi de trajectoire est similaire à celui du suivi de chemin, mais dans un temps prédéfini. Un problème régulier de commande du mouvement est le suivi de trajectoire, qui concerne la conception d'une loi de commande qui force le robot mobile à atteindre et à rester sur une référence paramétrée dans le temps (c'est-à-dire une trajectoire géométrique avec une régulation temporelle associée) [72].

En général, le problème du suivi de trajectoire revient à concevoir un contrôleur de modèle d'erreur dynamique, en poursuivant les étapes suivantes.

D'abord, concevoir le contrôleur cinématique pour rendre l'erreur de position asymptotiquement stable. Ensuite, concevoir un contrôleur dynamique pour calculer les couples de sorte que les vitesses du robot mobile convergent vers les entrées de vitesses données par le contrôleur cinématique.

En d'autres termes, nous divisons la conception du contrôleur en deux étapes :
— Le contrôleur cinématique asymptotiquement stable selon Lyapunov pour contrôler la position du robot.
— Le contrôleur dynamique pour commander la vitesse (c'est-à-dire que la vitesse de sortie du contrôleur cinématique est la vitesse d'entrée du contrôleur dynamique).
Le schéma de contrôle ressemble donc à celui donné la Figure (3.1).

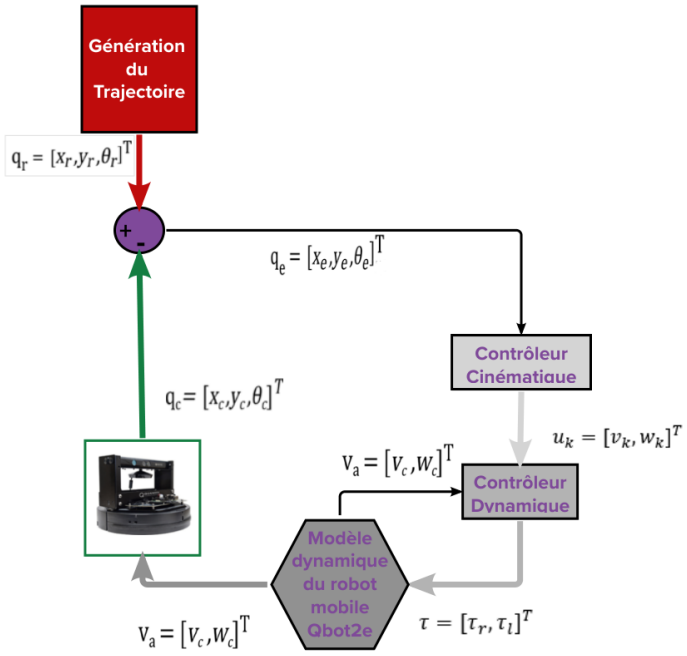

FIGURE 3.1 – Architecture du système de contrôle

3.2.1 Conception du contrôleur cinématique

Le contrôleur cinématique qu'on a dévellopé utilise deux postures pour le robot mobile à savoir : la posture de référence $q_{ref} = [x_{ref}, y_{ref}, \theta_{ref}]^T$ et la position courante du robot sous la forme $q_p = [x_p, y_p, \theta_p]^T$. Par la suite, on définit le modèle d'erreur de suivi en fonction du repère lié à la plateforme mobile ou repère local(égocentrique) comme suit :

$$q_e = \begin{bmatrix} x_e \\ y_e \\ \theta_e \end{bmatrix} = R(\theta)(q_{ref} - q_p) = R(\theta) \begin{bmatrix} x_{ref} - x_p \\ y_{ref} - y_p \\ \theta_{ref} - \theta_p \end{bmatrix} \tag{3.1}$$

La différenciation par rapport au temps du modèle d'erreur donné précédemment (3.1), nous permet d'obtenir la dynamique d'erreur pour le problème de suivi qui peut être écrite comme suit :

$$\dot{x}_e = \omega_p y_e - v_p + v_{ref} cos(\theta_e) \tag{3.2}$$

$$\dot{y}_e = -\omega_p x_e + v_{ref} sin(\theta_e) \tag{3.3}$$

$$\dot{\theta}_e = \omega_{ref} - \omega_p \tag{3.4}$$

Pour vérifier les calculs relatifs aux expressions précédentes, considérons la relations (3.1) et l'égalité de la contrainte cinématique $\dot{y}_p cos(\theta) - \dot{x}_p sin(\theta) = 0$,donc :

$$\begin{aligned}
\dot{x}_e &= (\dot{x}_{ref} - \dot{x}_p)cos(\theta_p) + (\dot{y}_{ref} - \dot{y}_p)sin(\theta_p) - (x_{ref} - x_p)\dot{\theta}_p sin(\theta_p) + (y_{ref} - y_p)\dot{\theta}_p cos(\theta_p) \\
&= y_e\omega_p - v_p + \dot{x}_{ref}cos(\theta_p) + \dot{y}_{ref}sin(\theta_p) \\
&= y_e\omega_p - v_p + \dot{x}_{ref}cos(\theta_{ref} - \theta_e) + \dot{y}_{ref}sin(\theta_{ref} - \theta_e) \\
&= y_e\omega_p - v_p + \dot{x}_{ref}(cos(\theta_{ref})cos(\theta_e) + (sin(\theta_{ref})sin(\theta_e)) + \dot{y}_{ref}(sin(\theta_{ref})cos(\theta_e) - cos(\theta_{ref})sin(\theta_e)) \\
&= y_e\omega_p - v_p + (\dot{x}_{ref}cos(\theta_{ref}) + \dot{y}_{ref}sin(\theta_{ref})cos(\theta_e) + (\dot{x}_{ref}sin(\theta_{ref}) - \dot{y}_{ref}cos(\theta_{ref}))sin(\theta_e) \\
&= y_e\omega_p - v_p + v_{ref}cos(\theta_e)
\end{aligned} \tag{3.5}$$

$$\begin{aligned}
\dot{y}_e &= -(\dot{x}_{ref} - \dot{x}_p)sin(\theta_p) + (\dot{y}_{ref} - \dot{y}_p)cos(\theta_p) - (x_{ref} - x_p)\dot{\theta}_p cos(\theta_p) - (y_{ref} - y_p)\dot{\theta}_p sin(\theta_p) \\
&= -x_e\omega_p + \dot{x}_p sin(\theta_p) - \dot{y}_p cos(\theta_p) - \dot{x}_{ref}sin(\theta_p + \dot{y}_{ref}cos(\theta_p) \\
&= -x_e\omega_p - \dot{x}_{ref}sin(\theta_{ref} - \theta_e) + \dot{y}_{ref}cos(\theta_{ref} - \theta_e) \\
&= -x_e\omega_p - \dot{x}_{ref}(sin(\theta_{ref})cos(\theta_e) - (cos(\theta_{ref})sin(\theta_e)) + \dot{y}_{ref}(cos(\theta_{ref})cos(\theta_e) + sin(\theta_{ref})sin(\theta_e)) \\
&= -x_e\omega_p + (\dot{x}_{ref}cos(\theta_{ref}) + \dot{y}_{ref}sin(\theta_{ref})sin(\theta_e) + (\dot{y}_{ref}cos(\theta_{ref}) - \dot{x}_{ref}sin(\theta_{ref}))cos(\theta_e) \\
&= -x_e\omega_p + v_{ref}sin(\theta_e)
\end{aligned} \tag{3.6}$$

$$\dot{\theta}_e = \dot{\theta}_{ref} - \dot{\theta}_p = \omega_{ref} - \omega_p \tag{3.7}$$

Puisque $R(\theta)$ est inversible et non singulier, en utilisant la loi de contrôle suggérée " u_{kin} " (3.12), et en utilisant la fonction scalaire " V_{Lya} " (3.13), et les hypothèses de la " Proposition II " ci-dessous, alors :

$$\lim_{t\to\infty}(|x_e| + |y_e| + |\theta_e|) = 0 \tag{3.8}$$

Ceci conduit à :

$$\lim_{t\to\infty} x_p = x_{ref} \tag{3.9}$$

$$\lim_{t\to\infty} y_p = y_{ref} \tag{3.10}$$

$$\lim_{t\to\infty} \theta_p = \theta_{ref} \tag{3.11}$$

A partir de [73], on montre et on prouve que si les deux vitesses de référence (linéaire v_{ref} et angulaire ω_{ref}) sont continues et bornées, les dérivées de ces vitesses notées respectivement $\dot{v}_{ref}$ et $\dot{\omega}_{ref}$ sont suffisamment petites et $v_{ref} > 0$ ainsi que les gains ki_x, ki_y et ki_θ sont bornés et positive, alors l'utilisation de la loi de commande u_{kin} garantira la stabilité asymptotique au sens de **Lyapunov** en utilisant la fonction de **Lyapunov** V_{lya}, où [74]

$$u_{kin} = \begin{bmatrix} v_{ref}cos(\theta_e) + ki_x x_e \\ \omega_{ref} + ki_y v_{ref} y_e + ki_\theta v_{ref} sin(\theta_e) \end{bmatrix} \tag{3.12}$$

La solidité de cette règle de contrôle (3.12) est établie par les propositions $I et II$ suivantes :

Proposition I : Si nous utilisons la loi de commande (3.12), $q_e = 0$ est un point d'équilibre stable si la vitesse de référence $v_{ref} > 0$.

Preuve : Proposons une fonction scalaire V_{lya} comme fonction de Lyapunov candidate [75] :

$$V_{lya} = \frac{1}{2}(x_e^2 + y_e^2) + \frac{1}{ki_y}(1 - cos(\theta_e) \tag{3.13}$$

une deuxième forme du système dynamique d'erreur donné par les équations (3.2),(3.3),et (3.4) est donnée ci-dessous :

$$\dot{x}_e = (\omega_{ref} + v_{ref}(ki_y y_e + ki_\theta sin(\theta_e)))y_e - ki_x x_e \tag{3.14}$$

$$\dot{y}_e = -(\omega_{ref} + v_{ref}(ki_y y_e + ki_\theta sin(\theta_e)))x_e + v_{ref} sin(\theta_e) \tag{3.15}$$

$$\dot{\theta}_e = -v_{ref}(ki_y y_e + ki_\theta sin(\theta_e)) \tag{3.16}$$

Il est clair que $V_{lya} \geq 0$. Si $q_e = 0, V_{lya} = 0$. Si $q_e \neq 0, V_{lya} > 0$. En outre, on utilise les équations : (3.14), (3.15), (3.16).

$$\begin{aligned} \dot{V}_{lya} &= \dot{x}_e x_e + \dot{y}_e y_e + \frac{\dot{\theta}_e sin\theta_e}{ki_y} \\ &= [(w_{ref} + v_{ref}(ki_y y_e + ki_\theta sin\theta_e))y_e - ki_x x_e]x_e \\ &+ [-(w_{ref} + v_{ref}(ki_y y_e + ki_\theta sin\theta_e))x_e + v_{ref} sin\theta_e]y_e \\ &+ [-v_{ref}(ki_y y_e + ki_\theta sin\theta_e)]\frac{sin\theta_e}{ki_y} \\ &= -ki_x x_e^2 - \frac{v_{ref} ki_\theta sin^2\theta_e}{ki_y} \leq 0 \end{aligned} \tag{3.17}$$

Alors, V_{lya} devient une fonction de Lyapunov.

La proposition suivante démontre que la stabilité asymptotique uniforme au voisinage de $q_e = 0$ sous certaines conditions :

Proposition II : Supposons que :
— v_{ref}, et w_{ref}, sont continus,
— $v_{ref}, w_{ref}, ki_x, ki_y$ et ki_θ sont bornés et positive,
— $\dot{v}_{ref}$, et $\dot{w}_{ref}$, sont suffisamment petits.
Sous ces conditions, $q_e = 0$ est uniformément asymptotiquement stable sur $[0, \infty)$.

Preuve : En linéarisant le système d'équations différentielles (3.14), (3.15), (3.16) autour de $q_e = 0$.

$$\dot{q}_e = A_s q_e \tag{3.18}$$

Oû,

$$A_s = \begin{bmatrix} -ki_x & w_{ref} & 0 \\ -w_{ref} & 0 & v_{ref} \\ 0 & -v_{ref} ki_y & -v_{ref} ki_\theta \end{bmatrix} \tag{3.19}$$

Alors, $A_s(.)$ est continuellement différentiable et est borné. L'équation caractéristique de A_s est :

$$a_3 S^3 + a_2 S^2 + a_1 S + a_0 = 0 \tag{3.20}$$

Oû :

$$\begin{cases} a_3 = 1 \\ a_2 = ki_\theta v_{ref} + ki_x \\ a_1 = ki_y v_{ref}^2 + ki_x ki_\theta v_{ref} + w_{ref}^2 \\ a_0 = ki_x ki_y v_{ref}^2 + w_{ref}^2 ki_\theta v_{ref} \end{cases} \tag{3.21}$$

Puisque tous les coefficients a_i sont positifs et que $a_1 a_2 - a_0 a_3 > 0$, les parties réelles de toutes les racines sont négatives par le critère de Routh-Hurwitz [75].

3.2.2 Conception du contrôleur dynamique

La section précédente portait sur l'élaboration d'une loi de commande en vitesse $u_{kin}(t)$ définie dans l'équation (3.12) pour le modèle cinématique du robot mobile.

Dans cette section, nous illustrons notre contribution à la littérature qui consiste à convertir cette loi de commande en vitesse en une loi de commande de couple τ pour le robot mobile.

Pour ce faire, considérons les équations du mouvement de la plate-forme du robot mobile non holonomique qui est donnée par l'équation (2.67) et l'équation (2.73) et elles sont montrées ci-dessous :

$$\dot{q} = S(q)\eta_{kin} \tag{3.22}$$

Où : $\eta_{kin} = [v, w]^T$

$$\overline{M}(q)\dot{\eta}_{kin} + \overline{V}(q, \dot{q})\eta_{kin} = \overline{B}(q)\tau \tag{3.23}$$

Si nous choisissons τ comme suit :

$$\tau = \overline{B}^{-1}(q)[\overline{M}(q)\nu(t) + \overline{V}(q, \dot{q})\eta_{kin}(t)] \tag{3.24}$$

Alors, Equation (3.24) devient :

$$\dot{\eta}_{kin}(t) = \nu \tag{3.25}$$

Par consequent, nous pouvons choisir $\nu = u_{kin}$ du contrôleur cinématique

Le contrôleur cinématique proposé dans la section précédente ne donne pas de résultats satisfaisants lors de l'expérimentation sur le robot Qbot2e, c'était juste un résultat qui assure la poursuite de trajectoire avec une certaine erreur. Pour améliorer les performances du robot, nous avons utilisé un contrôle adaptatif (ANFIS). Ce type de contrôle permet d'ajuster les vitesses linéaire et angulaire du robot en fonction de l'environnement et des conditions de fonctionnement.

L'utilisation de la commande adaptative avec ANFIS a produit de résultats où l'erreur de poursuite est très réduite. Le robot a pu atteindre ses objectifs avec plus de précision et d'efficacité.

Par conséquent, nous présenterons brièvement ce qu'est l'ANFIS, puis comment en tirer parti dans notre scénario, et ensuite prouver sa stabilité. Enfin, nous passerons à la section des résultats pour montrer sa supériorité par rapport au contrôleur proposé précédemment, en particulier dans l'expérimentation.

3.3 Téchnique Neuro-floue

3.3.1 Introduction

Un système neuro-flou est un système flou qui utilise un algorithme d'apprentissage dérivé ou inspiré de la théorie des réseaux neuronaux pour déterminer ses paramètres (ensembles flous et règles floues) en traitant des échantillons de données [76].

Réseaux de neuronnes

Les réseaux neuronaux sont un sous-domaine du modèle de machine learning (Apprentissage de la machine) qui s'inspire de la façon dont le cerveau humain fonctionne. Ils sont composés de couches de neurones artificiels interconnectés, qui sont des unités de traitement simples effectuant un calcul spécifique sur des données d'entrée et transmettant le résultat à la couche suivante. Chaque neurone reçoit des données de plusieurs sources, les traite à l'aide d'une fonction d'activation et transmet le résultat aux autres neurones de la couche suivante [77]. La structure et le fonctionnement des réseaux neuronaux s'inspirent de la structure et du fonctionnement du cerveau humain, qui est composé de milliards de neurones reliés par des synapses [77]. Les réseaux neuronaux traitent et intègrent des informations provenant de sources multiples pour résoudre des tâches complexes.

Les réseaux de neurones sont utilisés depuis des décennies dans diverses applications, telles que la reconnaissance d'images [78] et de la parole, le traitement du langage naturel [79] et les tâches de prédiction [80].

Il existe plusieurs types de réseaux neuronaux, notamment les réseaux neuronaux à anticipation (Feedforward neural networks) [81], les réseaux neuronaux convolutifs (Convolutional neural networks) [82] et les réseaux neuronaux récurrents (Recurrent neural networks) [83]. Chacun de ces types de réseaux neuronaux est bien adapté à des types de tâches et de structures de données spécifiques.

Le réseau neuronal comporte deux parties essentielles, le training et l'optimisation. L'entraînement d'un réseau neuronal consiste à ajuster les poids et les biais du réseau pour minimiser l'erreur entre la sortie prédite et la sortie réelle sur un ensemble de données de formation [77]. Ce processus est appelé optimisation et est généralement réalisé à l'aide d'un algorithme d'optimisation tel que la descente de gradient [84].

La fonction d'activation [85] est un composant crucial des réseaux neuronaux, car elle introduit la non-linéarité dans le réseau et lui permet d'apprendre et de modéliser des relations complexes dans les données. Les fonctions d'activation sont appliquées élément par élément à la sortie de chaque neurone du réseau et déterminent la sortie du neurone en fonction de son entrée.

Il existe plusieurs types de fonctions d'activation pouvant être utilisées dans les réseaux neuronaux, notamment sigmoïde, tanh, ReLU et Leaky ReLU [85]. Chaque fonction d'activation possède ses propres caractéristiques et est bien adaptée à des types de tâches spécifiques.

La technique de rétropropagation des erreurs est une approche d'apprentissage supervisée. elle permet d'ajuster les poids et les biais du réseau neuronal pour minimiser l'erreur quadratique entre les sorties réelles et calculées. Par conséquent, l'erreur est calculée pour chaque paire d'entrée/sortie. Si la sortie du réseau diffère de la sortie réelle, les pondérations et les biais sont ajustés en temps réel pour minimiser l'erreur.

Logique floue

La logique floue suscite actuellement l'intérêt des chercheurs, ingénieurs et professionnels de l'industrie, cet intérêt général est principalement dû au fait que la logique floue a pour but de formaliser des méthodes empiriques, de généraliser des modes de raisonnement naturel, d'automatiser la prise de décision dans leur domaine et de créer des systèmes artificiels. qui effectuent les tâches généralement effectuées par les humains[86].

Le réglage par la logique floue se rapproche du processus de prise de décision humain, car il traite les variables non plus comme des valeurs logiques, mais comme des variables linguistiques, telles que "aller beaucoup plus vite" ou "freiner à fond"[86].

Par ailleurs, ces variables linguistiques sont utilisées dans le cadre de règles qui font référence à une connaissance spécifique du comportement du système à ajuster. En effet, en se basant sur ce principe, les systèmes flous et les règles qui leur sont associées forment une classe de fonctions non linéaires, permettant d'entraîner une vaste gamme de modèles et de correcteurs par apprentissage [87].

Principe de la logique floue

La logique floue est en réalité une extension de la logique binaire, reposant sur un ensemble de notions fondamentales développées pour justifier et expliquer certaines notions clés. Pour bien comprendre le réglage par la logique floue, il est essentiel de maîtriser les éléments suivants [86] :

— Les variables le flou
— Les règles d'inférence
— Les opérateurs flous

Variables floues

Contrairement aux variables binaires qui ont deux états distincts, "vrai" ou "faux", les variables floues possèdent une gamme continue de valeurs entre ces deux extrêmes. Cela est rendu possible grâce à deux concepts clés :

— Les fonctions d'appartenance qui déterminent le degré de vérité d'une variable en fonction de la valeur d'entrée. Ces fonctions permettent de quantifier la mesure dans laquelle une valeur est "vraie" pour une variable floue donnée.
— Les ensembles flous qui regroupent un ensemble de variables floues associées à une valeur d'entrée spécifique. Ces ensembles permettent de représenter la variation continue des valeurs entre les états "vrai" et "faux" pour une variable donnée.

Ces deux aspects, les fonctions d'appartenance et les ensembles flous, sont essentiels pour caractériser les variables floues et leur permettre de prendre en compte une gradation continue de valeurs.

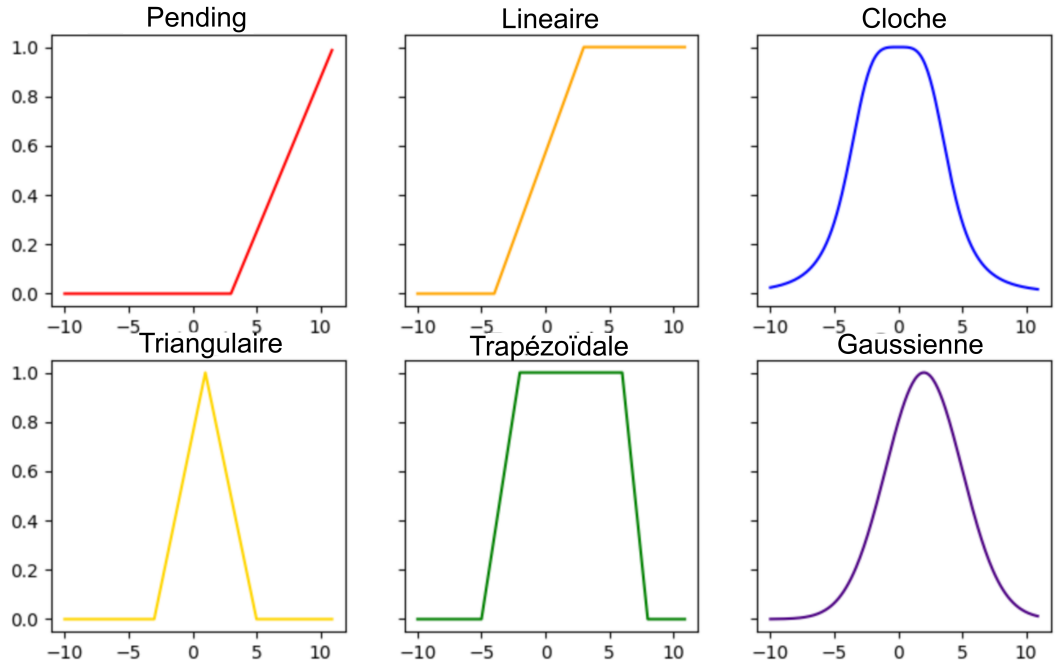

FIGURE 3.2 – 6 types de fonctions d'appartenance [88]

Règles d'inférence

Les règles d'inférence sont un système de règles floues qui modélisent la relation entre les variables d'entrée et de sortie d'un système spécifique, en utilisant des

règles linguistiques [89]. Cela permet une interprétation et une analyse plus facile des résultats.

Opérateurs flous

Les règles d'inférence font appel aux opérateurs présentes dans le tableau (3.1), qui s'appliquent aux variables floues :

Les opérations de minimum et de maximum sont simples à calculer, mais elles privilégient l'une des deux variables. En revanche, les opérations de produit et de valeur moyenne sont plus complexes, mais elles tiennent compte des valeurs des deux variables [89].

Un système flou est un système qui se base sur des connaissances spécifiques. Son architecture de base, telle qu'illustrée dans la Figure (3.3), est principalement constituée de quatre modules essentiels : la Fuzzification, la Base des règles, le Raisonnement flou et la Défuzzification [87].

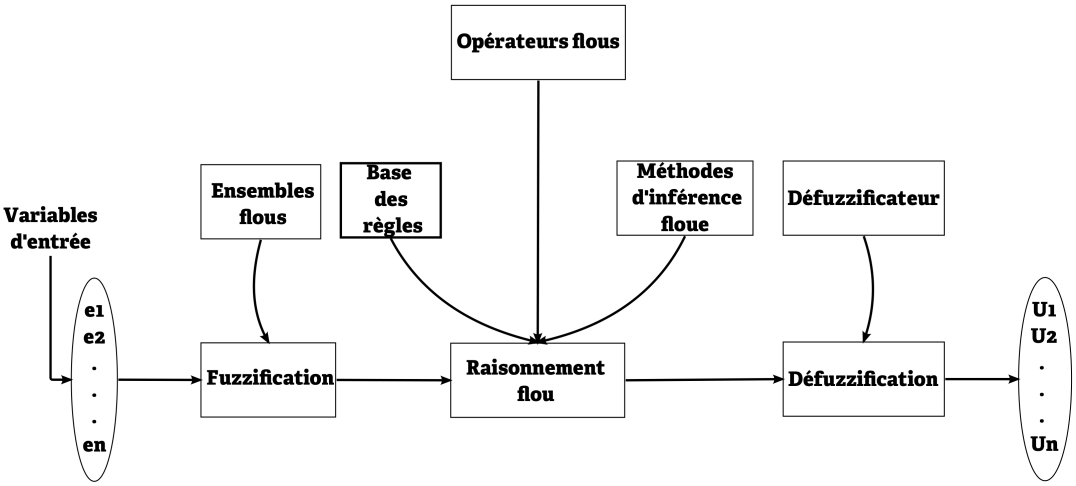

FIGURE 3.3 – Architecture de base d'un système flou

Jusqu'aux années 1990, les techniques neurales et floues étaient perçues comme des approches distinctes, chacune ayant ses propres avantages et limitations, et s'appliquant à des domaines spécifiques.

En ce qui concerne le contrôle des processus, la commande floue se révélait particulièrement adaptée à une interaction avec un opérateur humain, grâce à sa capacité à traiter des termes linguistiques. Cependant, l'utilisation de fonctions d'appartenance et la formulation des règles laissaient place à une certaine subjectivité,

TABLE 3.1 – Les opérateurs floues

Opérateur	Opération sur le degré de vérité des variables
ET	Minimum
	Produit
OU	Maximum
	Valeur moyenne
NON	Complément à un

ce qui pouvait conduire à des solutions éloignées de l'optimum et présentant des imprécisions.

Les réseaux neuromimétiques (réseaux hiérarchiques), en revanche, étaient idéaux lorsque seules des données numériques étaient disponibles pour développer des contrôles. Ces réseaux sont appris par exemple en minimisant la fonction de coût et donnent une bonne qualité de généralisation. En revanche, il était particulièrement difficile d'incorporer des connaissances a priori dans les deux approches, car la structure interne des réseaux neuronaux et des systèmes flous était souvent considérée comme une "boîte noire" difficilement interprétable. Toutefois, ces deux méthodes présentent des similitudes :

— Les deux méthodes, utilisées pour le contrôle, possèdent de bonnes propriétés de robustesse et sont capables d'extrapolation et de généralisation.
— Aucune des deux méthodes n'exige un modèle mathématique précis du système à contrôler, ce qui permet de les appliquer facilement aux systèmes non linéaires.

Ainsi, au début des années 1990, l'idée d'optimiser un système d'inférence floue en utilisant des méthodes d'apprentissage supervisé, comme la rétropropagation du gradient, a émergé naturellement. Cette approche cherchait à ajuster automatiquement les paramètres du système d'inférence floue.

Il existe trois architectures neuro-floues qui permettent d'optimiser un système d'inférence floue : l'architecture Nomura, l'architecture LSC et l'architecture ANFIS.

L'architecture Nomura est une méthode autocorrectrice d'un système d'inférence floue dans laquelle le concept de "neuro-flou" est abordé. Elle fut présentée par les scientifiques japonais Hiroyoshi Nomura [90], Isao Hayashi et Noburu Wakami.

L'architecture LSC tire son nom du laboratoire universitaire Systèmes Complexes d'Evry Val d'Essonne, qui l'a initialement proposée [91]. Elle présente des similitudes avec l'architecture Nomura, mais se distingue par le niveau de l'algorithme d'apprentissage qui est entièrement implémenté en ligne, en minimisant la fonction de coût E pour générer et ajuster les paramètres w_i de la partie conclusion des règles.

Les fonctions d'appartenance de la prémisse proposées par cette architecture sont de type gaussien et sont définies par la relation suivante :

$$\mu_{i,j}(x_i) = \exp(-\frac{(x_i - c_{i,j}^2)}{2\sigma_{i,j}}) \tag{3.26}$$

Le processus d'apprentissage est basé sur la descente de gradient, où chaque paramètre w_i est modifié par sa rétropropagation proportionnelle dans l'erreur globale E :

$$E = \frac{1}{2}(y - y_{ref}) \tag{3.27}$$

$$W_i(t+1) = W_i(t) - \frac{K_w O_i}{\sum_{k=1}^{n}(O_k)}.(y - y_{ref}) \quad k = 1, 2, ..., n \tag{3.28}$$

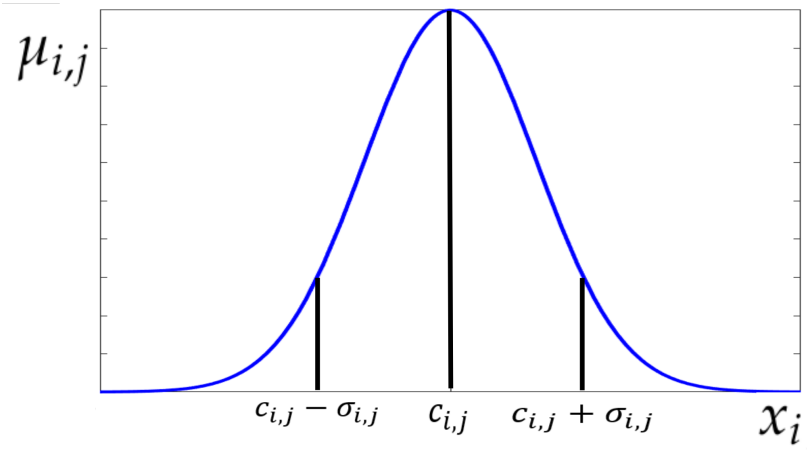

FIGURE 3.4 – Représentation du système d'inférence floue

3.3.2 Architecture ANFIS

L'ANFIS, une architecture appartenant à une catégorie de réseaux adaptatifs [92], est fonctionnellement équivalente à un système d'inférence floue de type Sugeno, connu sous le nom de ANFIS (Adaptive-Network-based Fuzzy Inference System) [93]. Cette approche utilise la méthode des moindres carrés combinée à la rétropropagation du gradient pour l'apprentissage. Pour simplifier la compréhension sans perdre en généralité, nous considérons un système avec deux entrées x_1 et x_2, ainsi qu'une sortie y. Nous examinons également un modèle flou de type Takagi-Sugeno1 pour ce système, composé des deux règles suivantes :

$$\begin{array}{llllllllll} Si & x_1 & est & A_1 & et & x_2 & est & B_1 & alors & y_1 = F_1(x_1, x_2) = \alpha_1 x_1 + \beta_1 x_2 + \gamma_1 \\ Si & x_1 & est & A_2 & et & x_2 & est & B_2 & alors & y_2 = F_2(x_1, x_2) = \alpha_2 x_1 + \beta_2 x_2 + \gamma_2 \end{array} \tag{3.29}$$

Oû :

- x_1, x_2 : Les variables d'entrée du système.
- A_1, A_2,B_1, B_2 : Les ensembles flous associés aux variables d'entrée x_1 et x_2.
- y_1, y_2 : Les variables de sortie du système, représentant les résultats.
- $F_1(x_1, x_2)$, $F_2(x_1, x_2)$: Les fonctions d'appartenance floues décrivant la relation entre les ensembles flous d'entrée et de sortie.
- α_i, β_i, γ_i : Les coefficients de pondération ou les paramètres des fonctions d'appartenance floues.

Jang a proposé de représenter cette base de règles par le réseau adaptatif de la Figure (3.5) [93]

La méthode ANFIS repose sur l'utilisation de réseaux multicouches appelés "adaptative networks", dans lesquels chaque cellule exécute sa propre fonction en respectant les paramètres qui lui sont attribués. Ainsi, l'application de l'exemple mentionné peut être visualisée dans la Figure suivante (3.5) : Le réseau adaptatif ANFIS est un réseau multicouche dans lequel les connexions ont un poids de 1 ou ne sont pas pondérées du tout. Les nœuds du réseau sont de deux types différents : les nœuds carrés, également appelés nœuds adaptatifs, qui contiennent des paramètres, et les nœuds circulaires, également appelés nœuds fixes, qui n'ont pas de paramètres. Cependant, chaque nœud, qu'il soit carré ou circulaire, applique une fonction à ses signaux d'entrée. La sortie O_i^k du nœud i de la couche k (appelé nœud

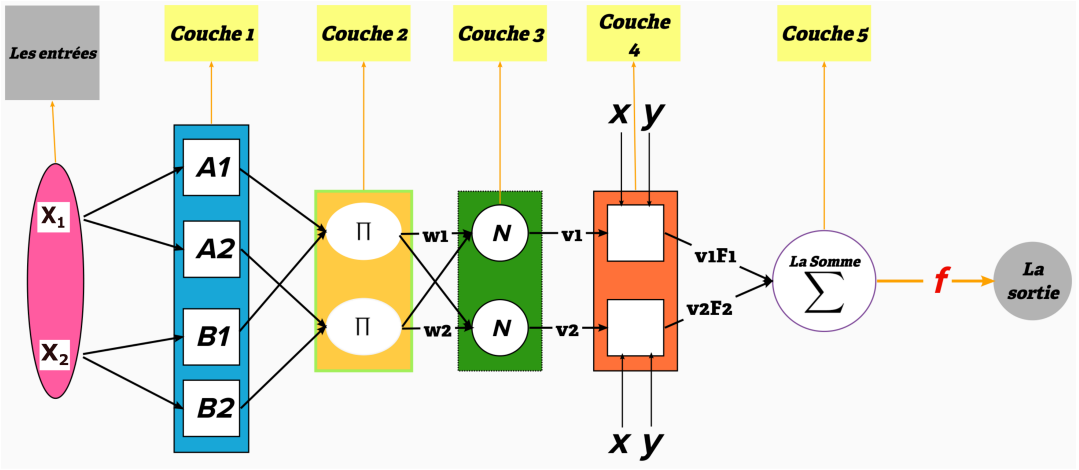

FIGURE 3.5 – Réseau ANFIS équivalent au raisonnement flou ANFIS

(i,k)) dépend à la fois des signaux provenant de la couche $k-1$ et des paramètres du nœud (i,k). En d'autres termes :

$$O_i^k = f(O_1^{k-1}, O_2^{k-1}, ..., O_{n_{k-1}}^{k-1}, \alpha, \beta, \gamma, ...) \tag{3.30}$$

Oû le nombre de nœuds dans la couche $k-1$ est noté n_{k-1}, et les paramètres du nœud (i,k) sont notés $\alpha, \beta, \gamma, ...$ Pour les nœuds circulaires, ces paramètres n'existent pas. Dans le réseau illustré dans la Figure (3.5), les nœuds d'une même couche ont des fonctions qui proviennent d'une même famille, que nous détaillons ci-dessous :

Couche N^o1 **:** (Couche de Fuzzification) Chaque nœud de cette couche est un nœud carré avec une fonction :

$$O_i^1 = \nu_{A_i}(x) \tag{3.31}$$

Le nœud i prend en entrée la valeur x, et A_i est le terme linguistique associé à sa fonction. Le résultat produit par ce nœud représente le degré de satisfaction du qualificatif A_i par la valeur x. En d'autres termes, O_i^1 représente le degré d'appartenance de x à A_i. Les paramètres de ce nœud sont ceux de la fonction d'appartenance correspondante, qui sont généralement des gaussiennes ou des fonctions cloches, et sont utilisées fréquemment dans la méthode ANFIS.

Couche N^o2 **:** Chaque nœud i de cette couche est un nœud circulaire, également appelé $\prod$, qui produit en sortie le produit de ses entrées. Ce produit représente le degré d'activation d'une règle.

$$w_i = \mu_{A_i}(x_1).\mu_{B_i}(x_2) \tag{3.32}$$

Couche N^o3 **:** Chaque nœud de cette couche est un nœud circulaire appelé N. Les cellules de cette couche calculent le rapport entre les valeurs de vérité de chaque règle par rapport à la somme de toutes les valeurs de vérité données par chaque règle. Elles estiment ainsi le poids "normalisé" de chaque règle.

$$v_i^{nor} = \frac{w_i}{w_1 + w_2} \tag{3.33}$$

Couche N^o4 **:** Chaque nœud de cette couche est un nœud carré avec une fonction qui effectue le calcul :

$$O_i^4 = v_i^{nor}.F_i = v_i^{nor}(\alpha_i x_1 + \beta_i x_2 + \gamma_i), i = 1, 2 \tag{3.34}$$

où v_i est la sortie de la couche N^o3, et $\alpha_i, \beta_i, \gamma_i$ est l'ensemble des paramètres de sortie de la règle i.

Couche N^o5 **:** (Couche de défuzzification) Le nœud unique de cette couche est un nœud circulaire qui effectue la somme des signaux provenant de la couche N^o4, c'est-à-dire,

$$O_1^5 = y = \sum_i v_i^{nor}.F_i \tag{3.35}$$

Nous remarquons que la sortie globale du réseau est équivalente à la sortie du modèle de Takagi Sugeno $1^{er} ordre$.

Aucun problème particulier ne se pose dans la généralisation du réseau à un système à entrées multiples. Le nombre de nœuds dans la couche N^o1 reste égal au nombre total de termes linguistiques définis. Le nombre de nœuds dans les couches $N^o2, 3et4$ reste également égal au nombre de règles floues.

L'exemple de réseau ANFIS illustré dans la Figure (3.6) correspond à un système d'inférence flou (SIF) avec 2 entrées et 9 règles. Chaque entrée est associée à 3 fonctions d'appartenance de type cloche, ce qui divise l'espace d'entrée en 9 sous-espaces flous, couvrant chacun une règle floue.

3.3.3 Technique d'apprentissage

L'une des principales caractéristiques des RNA est leur capacité à apprendre. Les réseaux de neurones artificiels peuvent apprendre et détecter les relations entre les entrées et les sorties pour généraliser les solutions et produire des sorties proches des sorties souhaitées .

Suite au développement de l'approche ANFIS, un certain nombre de méthodes ont été proposées pour l'apprentissage des règles et l'obtention d'un ensemble optimal de règles. Par exemple, [94] ont proposé de fusionner les modèles Min-Max et ANFIS pour obtenir un réseau neuro-flou et déterminer un ensemble optimal de règles floues.

[95] ont présenté l'application de la méthode Levenberg-Marquardt, qui est essentiellement une technique des moindres carrés non linéaires, pour l'apprentissage de la structure du réseau ANFIS. Dans un autre article, [96] a présenté un schéma pour la sélection des entrées et [97] a utilisé la carte de Kohonen pour la formation.

Dans cette thèse, le Gradient Décent (GD) est utilisé pour mettre à jour tous les paramètres de la structure ANFIS.

L'apprentissage à partir d'ensembles de données consiste à identifier les paramètres des antécédents et des conséquences compte tenu d'une structure de réseau fixe. L'algorithme d'apprentissage commence par construire un réseau initial, puis on applique la méthode d'apprentissage par rétropropagation de l'erreur. En utilisant une règle d'apprentissage hybride qui combine des algorithmes de descente de gradient avec une estimation des moindres carrés (MC).

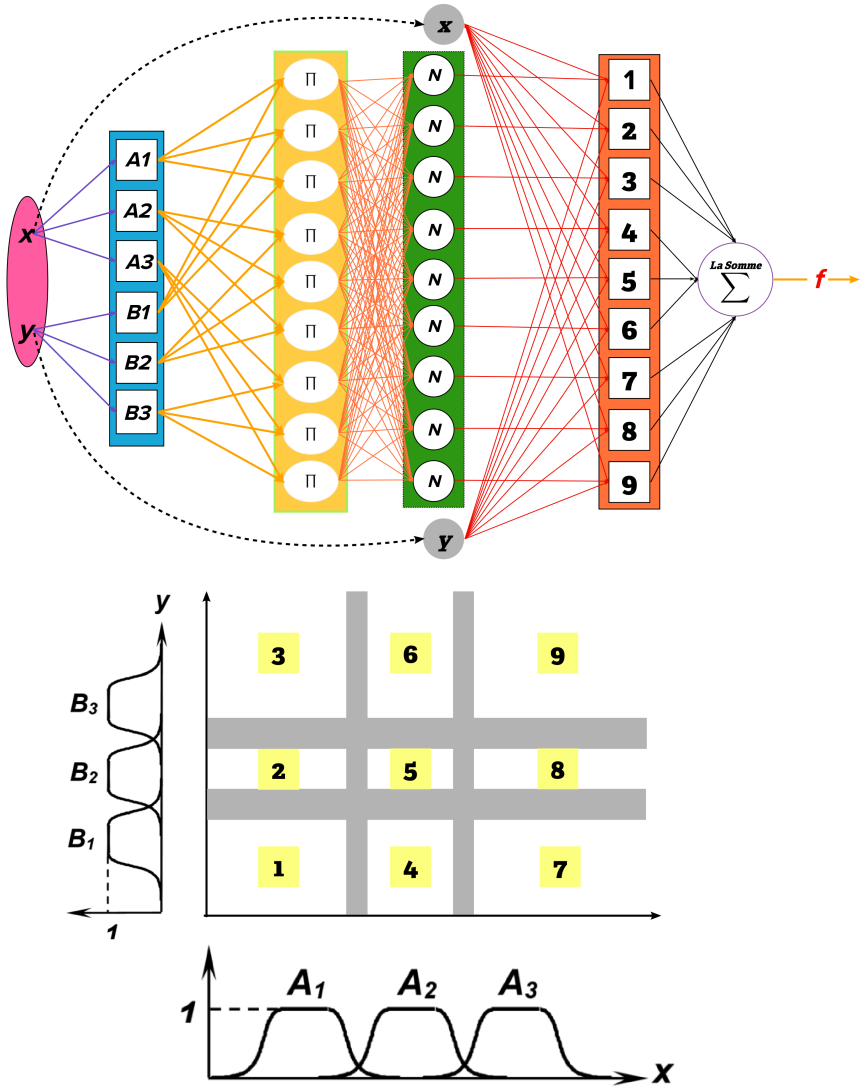

FIGURE 3.6 – Une structure ANFIS 2-entrées avec 9 règles

Dans l'architecture ANFIS, l'apprentissage est basé sur une approche hybride qui applique les moindres carrés combinés à la rétropropagation des gradients tout au long du processus d'apprentissage. Les paramètres de la conclusion de chaque règle ont une relation linéaire avec la sortie du système de raisonnement flou. Dans la direction "avant" du réseau, les paramètres de la prémisse de la règle sont ajustés par la méthode des moindres carrés, puis par la méthode du gradient dans le sens "retour" du réseau, à chaque itération de l'algorithme . Le tableau (3.2) résume les différentes étapes appropriées à l'algorithme d'apprentissage.

3.3.4 Régulateur ANFIS pour le robot mobile à entrainement différentiel

Dans cette section, nous développons l'algorithme d'apprentissage pour le réajustement des 24 paramètres pour le contrôle des vitesses $[v_p, w_p]$.

Dans ce cas, nous n'incluons que les erreurs de position à savoir : selon l'axe x (x_e) et selon l'axe y (y_e) ainsi que leur dérivée notée respectivement $\dot{x}_e$ et $\dot{y}_e$, à l'exclusion de l'erreur d'orientation thêta θ_e. Ainsi, nous présentons un nouveau vecteur

TABLE 3.2 – La méthode hybride utilisé dans ANFIS

	Avant (une seule fois)	Arrière (une seule fois)
Paramètres de l'antécédent	Fixés	Descente du gradient
Paramètres de la conclusion	Moindres carrées	Fixés

pour notre objectif, comme suit :

$$err_q = \begin{bmatrix} x_e \\ \dot{x}_e \\ y_e \\ \dot{y}_e \end{bmatrix} \tag{3.36}$$

Considérons un réseau de neurones flou ANFIS du premier ordre avec un système d'inférence floue de type Takagi-Sugeno, doté de quatre entrées $x_e, \dot{x}_e, y_e$ et $\dot{y}_e$ et de deux sorties v_p et w_p.

Rappelez-vous que cette architecture est en fait 2 architectures en une. En d'autres termes, les entrées 1 et 2 sont totalement découplées des entrées 3 et 4, comme vous pouvez le voir sur la Figure (3.7), Ainsi, il s'agit en fait de 2 contrôleurs ANFIS ayant 2 entrées (erreur de position et sa dérivée) pour x et y respectivement qui sont construits de manière identique. Ainsi, quel que soit l'algorithme que nous utilisons pour les 2 premières entrées, nous utilisons le même pour les deux autres entrées 3 et 4, cette clarification sera utile lorsque nous expliquerons l'algorithme d'apprentissage que nous avons utilisé. Nous allons le dériver pour l'erreur selon l'axe x . C'est-à-dire en optimisant $\alpha_i, \beta_i, \gamma_i$ pour $i = 1, 2, ..., 4$ et c'est exactement la même chose pour l'erreur selon l'axe y. A savoir l'optimisation de $\alpha_i, \beta_i, \gamma_i$ pour $i = 5, 6, ..., 8$

Dans ce cas, nous considérons que les paramètres de la prémisse sont fixes, alors que ceux de la conséquence sont ajustés en minimisant la fonction objectif suivante :

$$J = \begin{bmatrix} J_x \\ J_y \end{bmatrix} \tag{3.37}$$

Oû :

$$\begin{cases} J_x = \frac{1}{2} v_e^T v_e \\ J_y = \frac{1}{2} w_e^T w_e \end{cases} \tag{3.38}$$

Où, v_e est la différence entre la vitesse linéaire du robot et celle de référence $v_e = v_p - v_{ref}$, et w_e est la différence entre la vitesse linéaire du robot et celle de référence $w_e = w_p - w_{ref}$.

soit Γ_j le vecteur des paramètres à ajuster. L'objectif est de trouver les paramètres $\alpha_j, \beta_j, et \gamma_j$, du vecteur Γ_j en utilisant la méthode de la descente du gradient comme [97, 98].

Les règles floues sont définies comme suit :

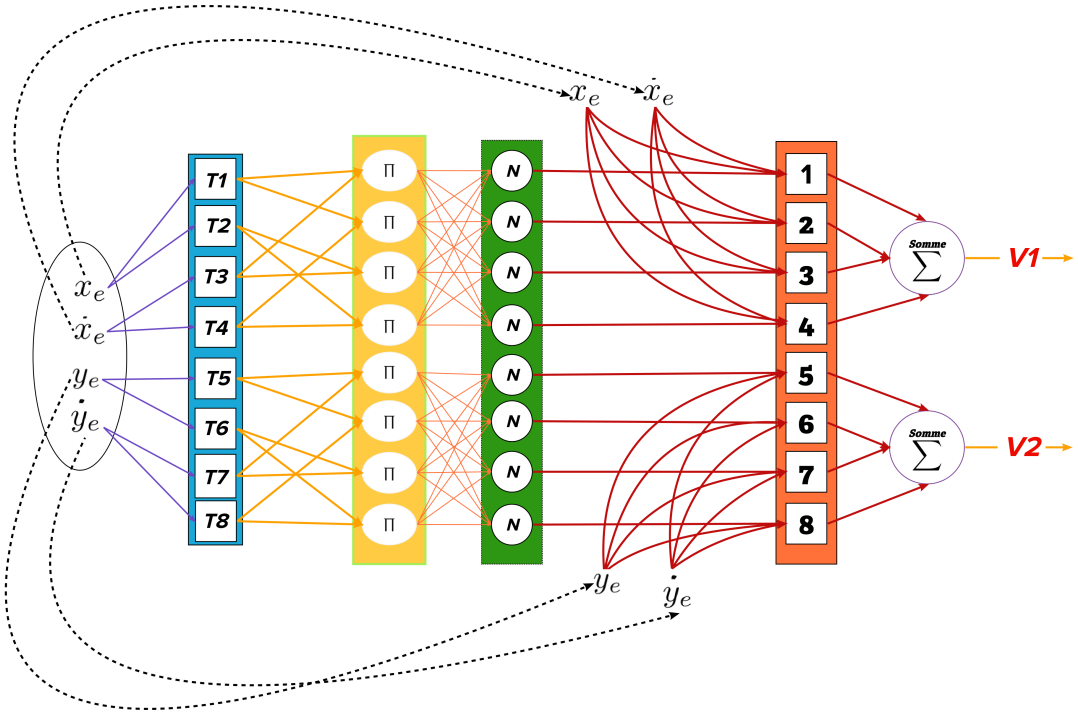

FIGURE 3.7 – Structure du régulateur ANFIS suggérée

$$
\begin{array}{llllllll}
Si & x_e & est & T_1 & et & \dot{x}_e & est & T_3 \quad alors \quad V1_1 = \alpha_1 x_e + \beta_1 \dot{x}_e + \gamma_1 \\
Si & x_e & est & T_1 & et & \dot{x}_e & est & T_4 \quad alors \quad V1_2 = \alpha_2 x_e + \beta_2 \dot{x}_e + \gamma_2 \\
Si & x_e & est & T_2 & et & \dot{x}_e & est & T_3 \quad alors \quad V1_3 = \alpha_3 x_e + \beta_3 \dot{x}_e + \gamma_3 \\
Si & x_e & est & T_2 & et & \dot{x}_e & est & T_4 \quad alors \quad V1_4 = \alpha_4 x_e + \beta_4 \dot{x}_e + \gamma_1 \\
Si & y_e & est & T_5 & et & \dot{y}_e & est & T_7 \quad alors \quad V2_1 = \alpha_5 y_e + \beta_5 \dot{y}_e + \gamma_5 \\
Si & y_e & est & T_5 & et & \dot{y}_e & est & T_8 \quad alors \quad V2_2 = \alpha_6 y_e + \beta_6 \dot{y}_e + \gamma_6 \\
Si & y_e & est & T_6 & et & \dot{y}_e & est & T_7 \quad alors \quad V2_3 = \alpha_7 y_e + \beta_7 \dot{y}_e + \gamma_7 \\
Si & y_e & est & T_6 & et & \dot{y}_e & est & T_8 \quad alors \quad V2_4 = \alpha_8 y_e + \beta_8 \dot{y}_e + \gamma_8
\end{array}
\tag{3.39}
$$

Où T_i sont des sous-ensembles flous, $\alpha_i, \beta_i, \gamma_i$ sont les paramètres de la conclusion de la j^{me} règle estimés durant l'apprentissage.

Nous associons deux ensembles flous pour chaque entrée $x_e, \dot{x}_e, y_e$et $\dot{y}_e$, nommés $N(ngatif)$ et $P(ositif)$. μ_P et μ_N sont les degrés d'appartenance des entrées aux sous-ensembles flous T_i, définis par les fonctions d'appartenance suivantes :

$$
\mu_P(e_x) = \begin{cases} 0 & \text{si } x_e < -1 \\ 0.5x_e + 0.5, & \text{si } -1 < x_e < 1, \\ 1 & \text{si } x_e > 1. \end{cases} \tag{3.40}
$$

$$
\mu_N(\dot{e}_x) = \begin{cases} 1 & \text{si } \dot{x}_e < -1 \\ -0.5\dot{x}_e + 0.5, & \text{si } -1 < \dot{x}_e < 1, \\ 0 & \text{si } \dot{x}_e > 1. \end{cases} \tag{3.41}
$$

$$\mu_P(e_y) = \begin{cases} 0 & \text{si } y_e < -1 \\ 0.5y_e + 0.5, & \text{si } -1 < y_e < 1, \\ 1 & \text{si } y_e > 1. \end{cases} \tag{3.42}$$

$$\mu_N(\dot{e}_y) = \begin{cases} 1 & \text{si } \dot{y}_e < -1 \\ -0.5\dot{y}_e + 0.5, & \text{si } -1 < \dot{y}_e < 1, \\ 0 & \text{si } \dot{y}_e > 1. \end{cases} \tag{3.43}$$

Les figures suivantes représentent les ensembles floues utilisés.

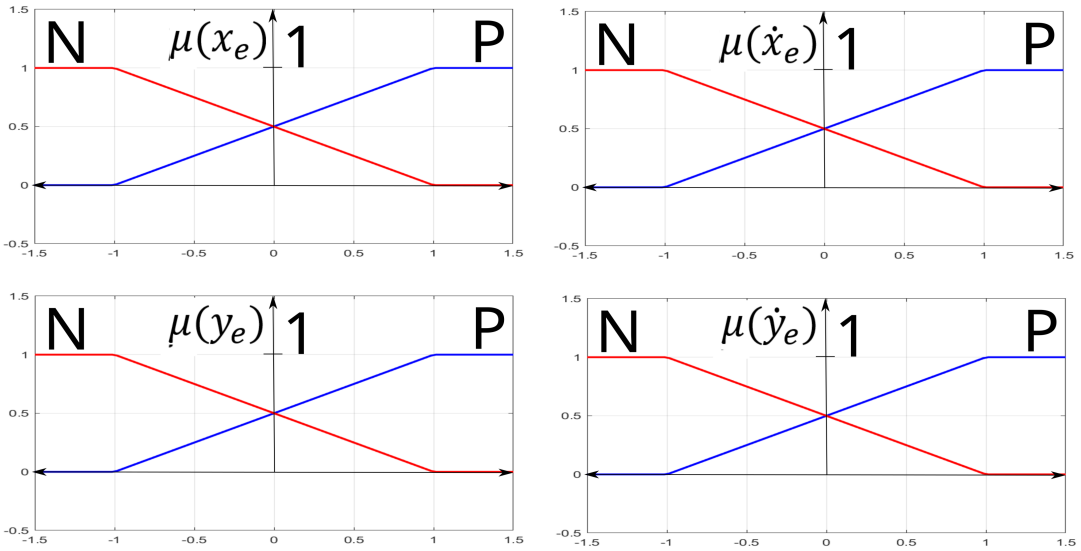

FIGURE 3.8 – Fonctions d'appartenance

En utilisant la méthode de déffuzification par centre de gravité, la valeur numérique de la sortie u_{anf} est donnée par :

$$u_{anf} = \begin{bmatrix} v_{anf} = \frac{\sum_{j=1}^{4} V1_j w_j}{\sum_{j=1}^{4} w_j} \\ w_{anf} = \frac{\sum_{j=1}^{4} V2_j w_j}{\sum_{j=5}^{8} w_j} \end{bmatrix} \tag{3.44}$$

Oû :

$$\begin{cases} w_1 = & \mu_P(x_e).\mu_P(\dot{x}_e) = \mu_{T_1}(x_e).\mu_{T_3}(\dot{x}_e) \\ w_2 = & \mu_P(x_e).\mu_P(\dot{x}_e) = \mu_{T_1}(x_e).\mu_{T_4}(\dot{x}_e) \\ w_3 = & \mu_N(x_e).\mu_N(\dot{x}_e) = \mu_{T_2}(x_e).\mu_{T_3}(\dot{x}_e) \\ w_4 = & \mu_N(x_e).\mu_N(\dot{x}_e) = \mu_{T_3}(x_e).\mu_{T_4}(\dot{x}_e) \\ w_5 = & \mu_P(y_e).\mu_P(\dot{y}_e) = \mu_{T_5}(y_e).\mu_{T_7}(\dot{y}_e) \\ w_6 = & \mu_P(y_e).\mu_P(\dot{y}_e) = \mu_{T_5}(y_e).\mu_{T_8}(\dot{y}_e) \\ w_7 = & \mu_N(y_e).\mu_N(\dot{y}_e) = \mu_{T_6}(y_e).\mu_{T_7}(\dot{y}_e) \\ w_8 = & \mu_N(y_e).\mu_N(\dot{y}_e) = \mu_{T_7}(y_e).\mu_{T_8}(\dot{y}_e) \end{cases} \tag{3.45}$$

3.3.5 Algorithme d'apprentissage

Le mécanisme d'apprentissage permet de déterminer les paramètres de la prémisse et de la conséquence. Dans cette partie, nous présentons le mécanisme d'apprentissage pour le contrôleur cinématique. Ainsi, $q_x = [e_x, \dot{e}_x]^T$ et v_{anf} sont respectivement les entrées et sorties du premier contrôleur proposé, et $q_y = [e_y, \dot{e}_y]^T$ et w_{anf} sont respectivement les entrées et sorties du premier contrôleur proposé.

Nous n'illustrerons que la première commande ANFIS qui contrôle v_p, et il en va de même pour la seconde qui contrôle la vitesse angulaire w_p

Dans ce cas, nous envisageons que les paramètres de la prémisse sont fixes, alors que ceux de la conséquence sont ajustés en minimisant la fonction de coût suivante [98] :

$$J_x = \frac{1}{2} v_e^T v_e \tag{3.46}$$

Soit Γ_j le vecteur à minimiser, par la manipulation des paramètres $\alpha_i, \beta_i, \gamma_i$ par rétro propagation qui minimisent le vecteur Γ en utilisant une technique d'optimisation. A savoir, la descente de gradient comme suit [99] :

$$\Gamma_j(k+1) = \Gamma_j(k) - \zeta(k) \frac{\partial J}{\partial \Gamma_{j_x}} \tag{3.47}$$

Où $\zeta(k)$ est le taux d'apprentissage. le filtre de Kalman étendu consiste à linéariser la sortie autour de l'entrée de contrôle à chaque période d'échantillonnage [10, 21-23]. Cela revient à écrire :

$$\frac{\partial J}{\partial \Gamma_j} = \frac{\partial j_x}{\partial q_x} \frac{\partial q_x}{\partial \Gamma} = \frac{\partial j_x}{\partial v_e} \frac{\partial v_e}{\partial q_x} \frac{\partial q_x}{\partial \Gamma} \tag{3.48}$$

$$\frac{\partial J}{\partial \Gamma_j} = -v_e \frac{\partial v_p}{\partial q_x} \frac{\partial q_x}{\partial \Gamma_j} \tag{3.49}$$

$$\frac{\partial J}{\partial \Gamma_j} = -v_e \frac{\partial v_p}{\partial q_x} \Psi_x \tag{3.50}$$

D'après les équations (3.47), (3.48), (3.49) et (3.50), il s'ensuit :

$$\Gamma_j(k+1) = \Gamma_j(k) + k_j^{'} \Psi_{x(j)}^T v_e \tag{3.51}$$

Où :

$$\Psi_{x(j)}^T = \frac{\partial q_x}{\partial \Gamma_j} = \begin{bmatrix} \frac{\partial q_x}{\partial \alpha_i} \\ \frac{\partial q_x}{\partial \beta_i} \\ \frac{\partial q_x}{\partial \gamma_i} \end{bmatrix} \tag{3.52}$$

$$k_j^{'} = \zeta(k) \frac{\partial v_e}{\partial q_x} \tag{3.53}$$

Dans notre cas, $\frac{\partial v_e}{\partial q_x}$ ne peuvent pas être évalués, mais peuvent être estimés à l'aide des équations du filtre de Kalman étendu.

L'équation (3.51) peut être identifiée à l'équation du filtre de Kalman étendu [100] :

$$\Gamma_j(k+1) = \Gamma_j(k) + ka(k)v_e \tag{3.54}$$

Où $Ka(k)$ est le gain de Kalman défini comme suit :

$$ka(k) = \frac{P(k)H^T(k)}{H(k)P(k)H^T(k) + R(k)} \tag{3.55}$$

Où $H(k)$ est la matrice jacobienne (matrice d'observation du système), $P(k)$ est la matrice d'estimation de la covariance de l'erreur et $R(k)$ est la matrice de covariance du bruit de processus.

En prenant $H^T(k) = \Psi_j^T$,$P(k) = \lambda_1$et $R(k) = \lambda_2$, le gain $ka(k)$ peut être écrit :

$$ka(k) = \frac{\lambda_1}{\Psi_j \lambda_1 \Psi_j^T + \lambda_2} \Psi_j^T v_e \tag{3.56}$$

Par conséquent, l'équation (3.54) se réduit :

$$\Gamma_j(k+1) = \Gamma_j(k) + \frac{\lambda_1}{\Psi_j \lambda_1 \Psi_j^T + \lambda_2} \Psi_j^T v_e \tag{3.57}$$

Par identification entre les équations (3.51) et (3.57), on obtient :

$$k_j^{'} = \frac{\lambda_1}{\Psi_j \lambda_1 \Psi_j^T + \lambda_2} \tag{3.58}$$

Par conséquent, le vecteur des paramètres Γ_j peut être estimé par la formule suivante :

$$\Gamma_j(k+1) = \Gamma_j(k) + k_j^{'} \Psi_j^T v_e \tag{3.59}$$

Pour un temps d'échantillonnage très court T_e, on a :

$$\dot{\Gamma}_j = \frac{\Gamma_j(k+1) - \Gamma_j(k)}{T_e} = \frac{k_j^{'} \Psi_j^T v_e}{T_e} = K_1 \Psi_j^T v_e \tag{3.60}$$

Oû :

$$K_1 = \frac{k_j^{'}}{T_e} \tag{3.61}$$

D'oû :

$$\dot{\Gamma}_j = K_1 \Psi_j^T err = \Psi_j^T E_v \tag{3.62}$$

Avec : $E_v = K_1 v_e$

Que $\tilde{\Gamma}_j = \Gamma_{jd} - \Gamma_j$, où Γ_j est le vecteur des paramètres et Γ_{jd} le vecteur des paramètres souhaités.

Cela implique :

$$\tilde{\dot{\Gamma}}_j = \dot{\Gamma}_{jd} - \dot{\Gamma}_j = -\Psi_j^T E_v \tag{3.63}$$

Où : $E_v = v_{ref} - v_{anf}$ est l'erreur entre la sortie souhaitée du contrôleur v_{ref} et la sortie réelle v_{anf}. Pour une variation linéaire, elle est définie par :

$$E_v = v_{ref} - v_{anf} = \sum_{j=1}^{4}((\Psi_j)\Gamma_{jd} - (\Psi_j)\Gamma_j) = \sum_{j=1}^{4}((\Psi_j)(\Gamma_{jd} - \Gamma_j)) = \sum_{j=1}^{4}((\Psi_j)(\tilde{\Gamma}_j) \tag{3.64}$$

De la même manière :

$$E_w = w_{ref} - w_{anf} = \sum_{j=1}^{4}((\Psi_j)\Gamma_{jd} - (\Psi_j)\Gamma_j) = \sum_{j=1}^{4}((\Psi_j)(\Gamma_{jd} - \Gamma_j)) = \sum_{j=1}^{4}((\Psi_j)(\tilde{\Gamma}_j) \tag{3.65}$$

Avec : $E_w = K_2 w_e$

3.3.6 Analyse de la stabilité du système de régulation

Considérons la fonction de Lyapunov suivante [101] :

$$V_j = \frac{1}{2}\sum_{j=1}^{4}((\tilde{\Gamma}_j)^T(\tilde{\Gamma}_j)) \tag{3.66}$$

En différentiant V_j par rapport au temps, on obtient :

$$\dot{V}_j = \sum_{j=1}^{4}((\dot{\tilde{\Gamma}}_j^T)(\dot{\tilde{\Gamma}}_j)) \tag{3.67}$$

A partir des équations (3.63) , (3.64), (3.65) et (3.67) , on arrive à :

$$\dot{V}_j = \sum_{1}^{4}((-(\Psi_j^T)E_v)T(\tilde{\Gamma}_j)) = -(E_v)^T\sum_{j=1}^{4}((\Psi_j)(\tilde{\Gamma}_j)) = -(E_v)^T(E_v) \tag{3.68}$$

En conséquence :

$$\dot{V}_j = -(E_v)^T(E_v) \leq 0 \tag{3.69}$$

D'après l'équation (3.69) nous trouvons que $\dot{V}_j \leq 0$, nous concluons donc que le système est asymptotiquement stable au sens de Lyapunov [98, 102].

De la même manière, la deuxième sortie E_w est également asymptotiquement stable en utilisant la même fonction candidate de Lyapunov et la même procédure

3.4 Résultats de simulation et expérimentaux

Dans cette section, nous proposons d'illustrer en simulation et en expérimentation les stratégies de navigation présentées dans ce chapitre. Néanmoins la stratégie de commande neuro-floue est utilisée juste en expérimentation en vue d'améliorer les performances de poursuite de la commande par retour d'état dans le cas expérimental.

3.4.1 Résultats de simulation

Dans cette section, nous illustrons le performance du contrôleur proposé par quelques résultats de simulation pour le suivi des trajectoires de formes rectangulaire et infinie. Le contrôleur proposé est simulé à l'aide de MATLAB/SIMULINK, les paramètres pour le robot mobile et les moteurs DC sont inspirées des références [103] et [104]. Plusieurs simulations ont été effectuées avec différents paramètres, confirment que la trajectoire réelle du robot mobile conforme à la trajectoire de référence.

Le contrôleur suggéré est testé avec une connaissance parfaite des paramètres ; la pose initiale du robot était $q_0 = [0,1,0]^T$ et la position de référence initiale était $q_0^{ref} = [0,0.75,\pi/4]^T$. Les trajectoires des polygones sont représentées à l'aide d'équations paramétriques conformément à la formulation suivante :

$$traj = [(x(0)\,y(0)),(x(1)\,y(1)),...,(x(t)\,y(t))]$$

Où t est le temps de simulation, (x_i, y_i) représente la pose souhaitée au i-ième instant.

Les performances à atteindre sont décrites par la trajectoire des lemniscates simulée avec une vitesse constante $v_d = 0,4m/s$, et représentée par les équations ci-dessous :

$$x_d = 0 + \sin\left(\frac{t}{20}\right) \tag{3.70}$$

$$y_d = 0.8 + 2\sin\left(\frac{t}{10}\right) \tag{3.71}$$

$$\theta_d = atan2\,(y,x) \tag{3.72}$$

L'application de la loi de commande relative au contrôleur proposé permet d'obtenir les performances montrés sur les Figures (3.9)-(3.11). La Figure (3.9) représente un exemple de tâche pour illustrer le suivi de la trajectoire. Les graphes présentés sur la Figure (3.10) correspondent aux profils des erreurs en position selon les deux axes x et y et orientation θ. Quant aux graphes présentés sur la Figure (3.11) correspondent aux profils des vitesses linéaire et angulaire, réelles et désirées du robot mobile. Les Figures (3.12) - (3.14) représentent les résultats de simulation effectuées avec la trajectoire des lemniscates (Inifini). Ces résultats graphiques sont ordonnés dans le même ordre que précédemment.

Afin de tester la robustesse de cette stratégie de commande, nous avons perturbé la chaîne de commande grâce à l'introduction d'une plage de variation sur les différents paramètres liés au fonctionnement du contrôleur, et les meilleures valeurs que nous avons choisi sont données dans le Tableau 3.3. Les changements effecués sur les paramètres du contrôleur, doivent être suffisamment petits pour éviter un comportement oscillatoire et éventuellement l'instabilité.

3.4.1.1 Premier scénario : forme rectangulaire

La première trajectoire de simulation est d'une forme rectangulaire qui a la configuration suivante

$$traj_{rec} = [(0,0),(1.8,0),(1.8,1.8),(0,1.8),(0,0)]$$

La trajectoire du robot est représentée par une ligne continue et la trajectoire de référence est donnée par des points. La direction du mouvement est indiquée par des flèches et une numérotation successive. Le cercle vert indique la position initiale du robot q_0 et le cercle rouge indique la position initiale de la trajectoire de référence q_0^{ref} . On doit faire attention à l'erreur de position élevée au début, qui est due au fait que la trajectoire de référence commence à un point éloigné de la position initiale. Par la suite, l'erreur de position est maintenue dans des limites acceptables par le contrôleur de retour et le robot peut atteindre l'objectif en peu de temps, environ $8.5s$. Après $10s$, le robot était déjà sur la piste et est resté sur place pendant les $190s$ restantes de la simulation.

Une autre observation à noter est la petite erreur lorsque le robot tourne, qui peut être réduite en diminuant la vitesse de rotation, donnant ainsi le temps au robot de répondre au signal de commande.

TABLE 3.3 – Paramètres des simulations de la linéarisation de rétro-action

Paramètre	Valeur	Unité
k_{p1} 1er gain du contrôleur dynamique (pour l'erreur de vitesse linéaire)	10	sans unité
k_{p2} 2ème gain du contrôleur dynamique (pour l'erreur de vitesse angulaire)	10	sans unité
ki_x Gain du contrôleur cinématique	10	$1/s$
ki_y Gain du contrôleur cinématique	$6.4e-3$	$1/cm$
ki_θ Gain du contrôleur cinématique	0.16	$1/cm$

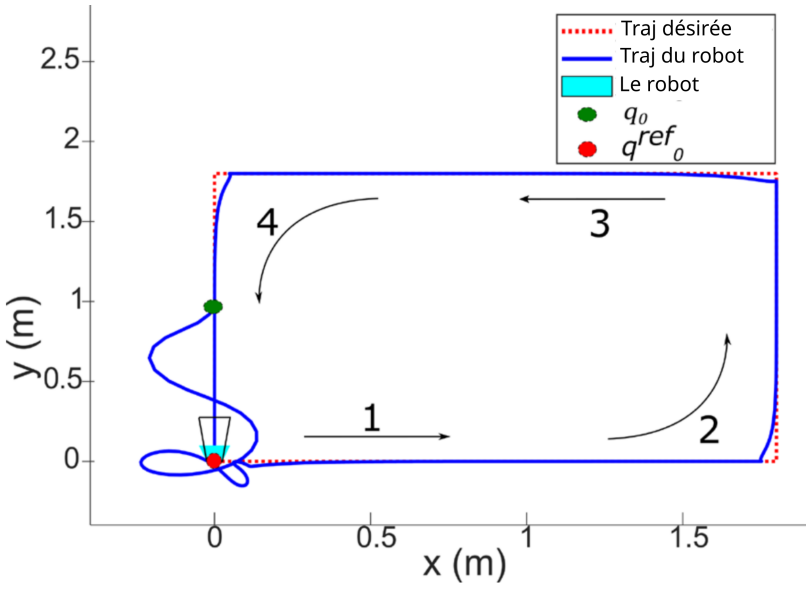

FIGURE 3.9 – $[x(t), y(t)]$ Courbe pour une trajectoire rectangulaire

Par la suite, les trois composantes de l'erreur sont représentées sur la Figure (3.10).

Comme on peut le voir sur la Figure (3.10), la convergence des erreurs est correctement assurée par le contrôleur. Les erreurs longitudinales et latérales sont inférieures à $0,1m$. De plus, comme on peut le voir dans le deuxième graphe (e_y) ,

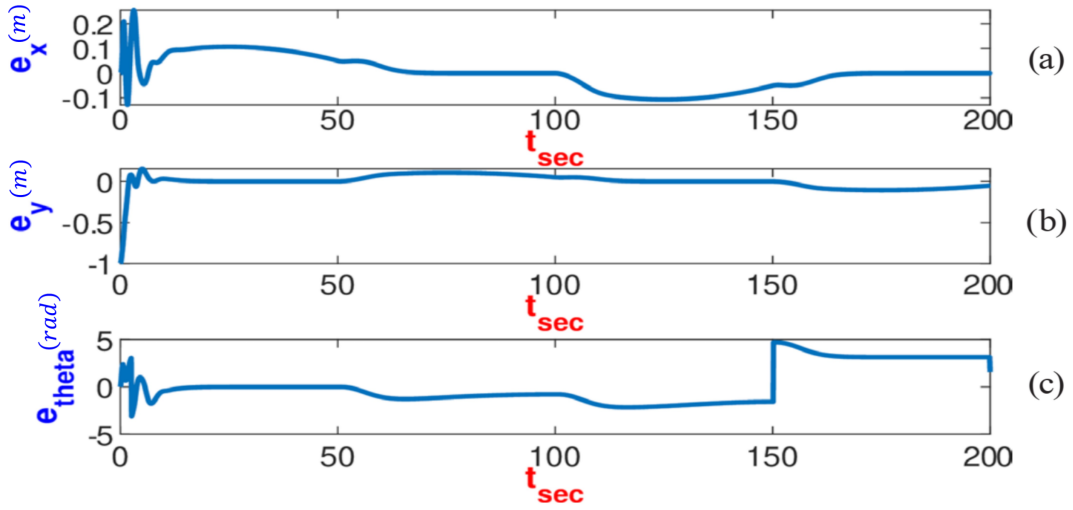

FIGURE 3.10 – Résultats des erreurs du robot : (a) erreur longitudinale du robot e_x , (b) erreur latérale du robot e_y , (c) erreur d'orientation du robot e_θ

l'erreur commence à partir de -1 qui est la différence entre le point de départ du robot $(y = 1)$ et le point de départ de la trajectoire de référence $(y = 0)$ qui devient nulle après un court instant et reste très proche de celle-ci pendant toute la simulation.

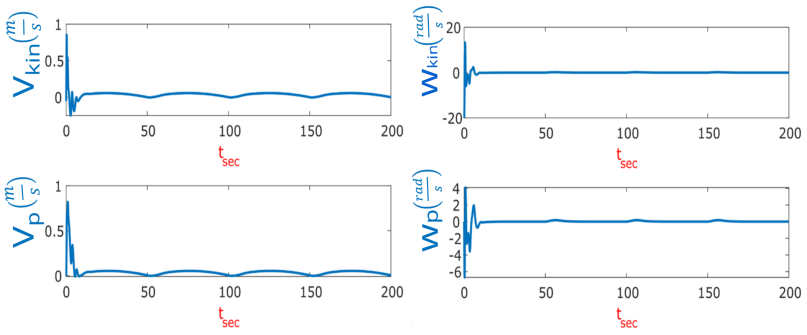

FIGURE 3.11 – Vitesses linéaires et angulaires du contrôleur et du robot

La Figure (3.11) illustre les vitesses linéaires et angulaires, désirées et réelles du robot mobile.

On constate que la dynamique des vitesses réelles varie peu par rapport à celle des vitesses désirées, ce qui signifie que le contrôleur effectue un suivi de trajectoire.

Dans l'intervalle ci-dessous, les vitesses linéaires v_{kin} et v_p ne sont pas nulle et les vitesses angulaires ω_{kin} et ω_p sont nulles, c'est là que le robot se déplace en ligne droite

$$]0, 45[\cup]55, 95[\cup]105, 145[\cup]155, 195[$$

. Dans l'intervalle ci-dessous, les vitesses linéaires v_{kin} et v_p sont nulles, et les vitesses angulaires ω_{kin} et ω_p sont non nulles, c'est là que le robot tourne

$$]45, 55[\cup]95, 105[\cup]135, 145[\cup]195, 200[$$

3.4.1.2 Deuxième scénario : trajectoire en forme de 8

Comme précédemment, la trajectoire du robot est représentée par une ligne continue et la trajectoire de référence est donnée par des points. La direction du mouvement est indiquée par des flèches et une numérotation successive. Le cercle orange indique la position initiale du robot et le cercle rouge indique la position initiale de la trajectoire de référence.

Nous remarquons, d'apès la Figure (3.12), l'erreur de suivi en position est maintenue dans des limites acceptables par le contrôleur de rétroaction $(0.1m)$ et le robot peut atteindre l'objectif en un temps réduit, environ $4.5s$. C'est la raison pour laquelle il était plus rapide que le scénario précédent car l'état initial du robot était très proche de l'état initial approprié à la référence.

Après la $5^{me}s$, le robot était déjà sur la piste et y est resté pendant les $105,5s$ restants de la simulation. Ensuite, les trois composantes de l'erreur sont montrées sur la Figure (3.13).

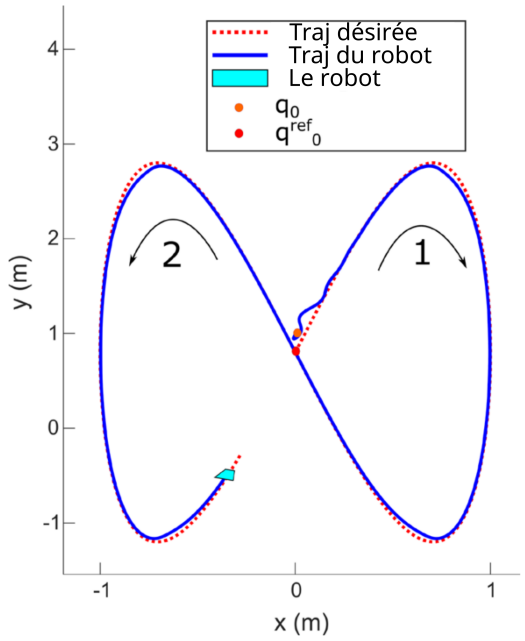

FIGURE 3.12 – $[x(t), y(t)]$ Courbe pour une trajectoire de forme 8

On conclut que, la convergence des erreurs est correctement assurée par le contrôleur. Une observation à noter est la différence d'erreurs lorsque la référence est un polygone, et lorsqu'il s'agit d'un lemniscate, les erreurs dans un polygone sont seulement perceptibles dans les virages, au contraire, dans les lemniscates, les erreurs sont toujours non nulles parce que les vitesses linéaire et angulaire sont toutes les deux non nulles. Les erreurs longitudinale et latérale sont inférieures à $0,1m$ pour l'ensemble de la simulation après la mise en place du robot sur la trajectoire (i.e., après $4,5s$). Quant à l'erreur d'orientation, elle est inférieure à $4rad$ depuis le début.

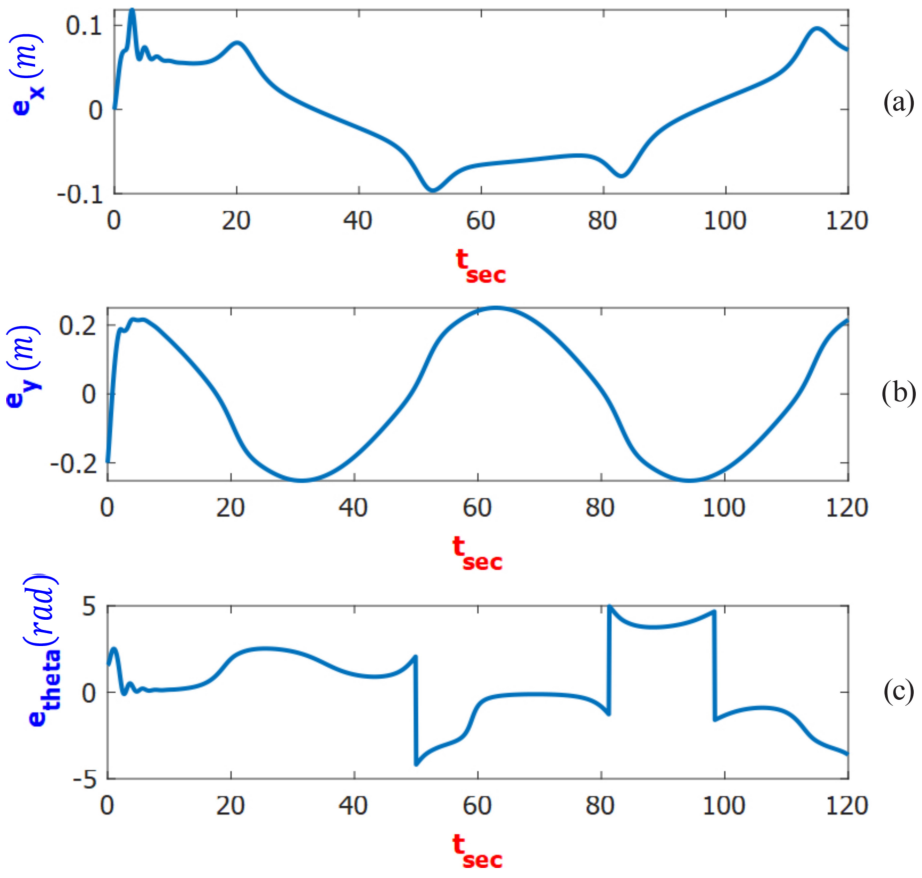

FIGURE 3.13 – Résultats des erreurs du robot : (a) erreur longitudinale du robot e_x , (b) erreur latérale du robot e_y , (c) erreur d'orientation du robot e_θ

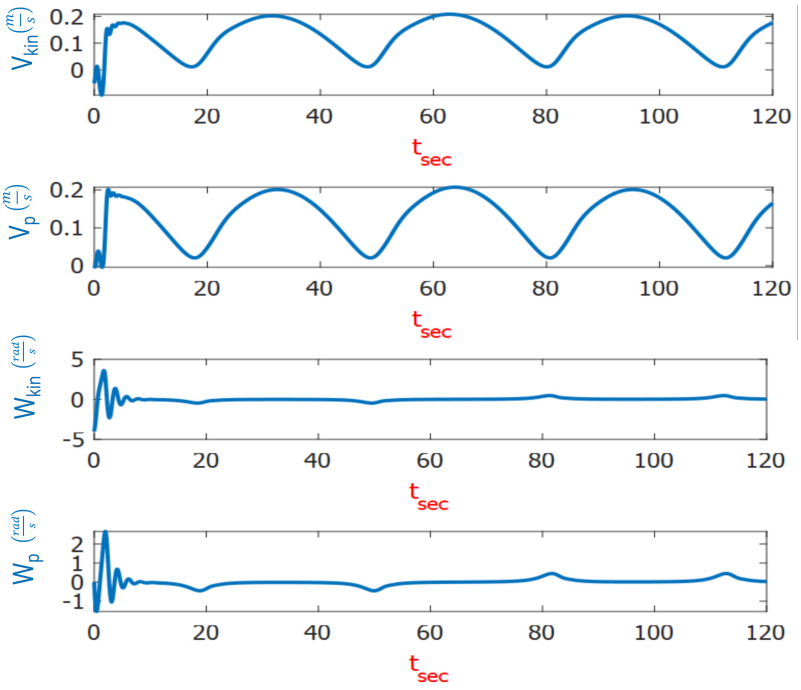

FIGURE 3.14 – Vitesses linéaires et angulaires du contrôleur et du robot

La Figure (3.14) illustre les différentes trajectoires des vitesses linéaires et angulaires de telle manière que les deux premières trajectoires correspondent respectivement à la la vitesse linéaire désirée la vitesse linéaire réelle du robot. Quant aux deux dernières trajectoires correspondent respectivement à la vitesse angulaire désirée et la vitesse angulaire réelle du robot. A travers ces résultats, nous constatons que les preformances de poursuite en vitesse sont assurées que soient pour les vitesses linéaires ou angulaires, ce qui signifie que le contrôleur contrôle réellement les vitesses linéaire et angulaire du robot mobile. Par ailleurs les vitesses linéaires v_{kin} et v_p sont positives et les vitesses ω_{kin} et ω_p sont nulles. Ce qui signifie que le robot mobile se déplace en ligne droite dans l'intervalle

$$(]0, 18[\cup]22, 48[\cup]52, 78[\cup]82, 111[\cup]113, 120[)$$

D'autre part, les vitesses linéaires v_{kin} et v_p sont presque nulles, et les vitesses angulaires ω_{kin} et ω_p sont non nulles, ce qui signifie que le robot mobile tourne, dans l'intervalle

$$(]18, 22[\cup]48, 52[\cup]79, 83[\cup]109, 111[)$$

On peut également constater que les vitesses angulaires sont négatives dans les intervalles $]18, 22[$ et $]48, 52[$, et que cela est dû au fait que le robot mobile tourne à l'encontre de la direction positive de thêta, c'est-à-dire dans le sens des aiguilles d'une montre, alors que les vitesses angulaires dans $]79, 83[$ et $]109, 111[$ sont positives, car le robot mobile tourne avec la direction positive de thêta (c'est-à-dire dans le sens inverse des aiguilles d'une montre).

3.4.2 Étude comparative entre la commande de position uniquement et la commande de position et de vitesse

Pour tester la robustesse des contrôleurs proposés, nous allons montrer la différence entre deux schémas de contrôle. Le premier est celui où nous utilisons uniquement le contrôleur cinématique, et le second est celui où nous utilisons les deux contrôleurs (cinématique et dynamique).

Pour montrer la différence entre les deux, nous commencerons par les tracés de suivi de trajectoire, puis les erreurs de suivi, ensuite les vitesses (linéaire et angulaire), et enfin un tableau qui inclut des métriques pour quantifier les performances de nos contrôleurs.

Les Figures (3.15) et (3.16) montrent respectivement les résultats obtenus avec les contrôleurs (cinématique et dynamique) et le contrôleur cinématique uniquement pour une trajectoire de référence rectangulaire et une trajectoire de référence lemniscate. Par évidence, nous constatons que le système dynamique et le contrôleur dynamique prennent en compte les non-linéarités du système ou lorsque les moteurs ne sont pas assez puissants pour permettre de négliger la dynamique, et donnent un résultat plus proche du robot réel, en particulier surtout dans les 10 premières secondes de la simulation.

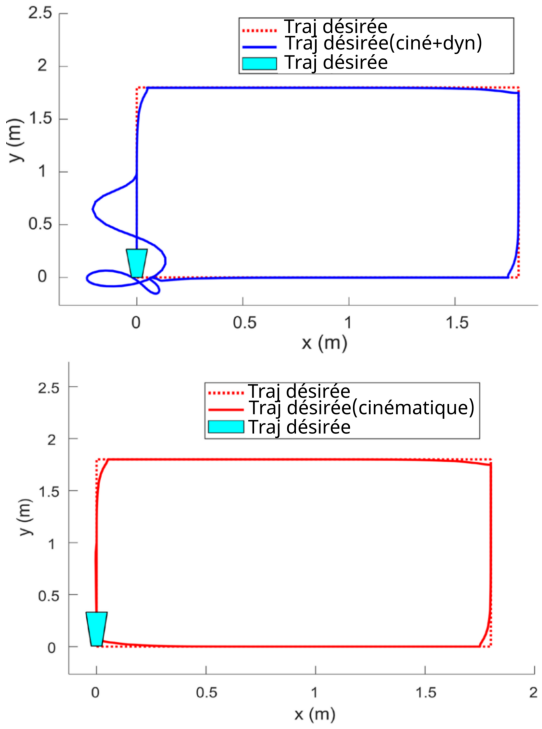

FIGURE 3.15 – Comportement du contrôleur (dynamique et cinématique (bleu)) à gauche par rapport à un contrôleur (cinématique uniquement (rouge)) à droite pour la même trajectoire de référence du rectangle

Ces résultats graphiques permettent de relever les caractéristiques dynamiques relatives aux simulations effectuées. Il est très évident qu'au début de la simulation, le contrôleur cinématique ne montre pas d'erreur de suivi, alors que, en considérant la dynamique, qui présente des oscillations au démarrage du robot mobile, qui sont

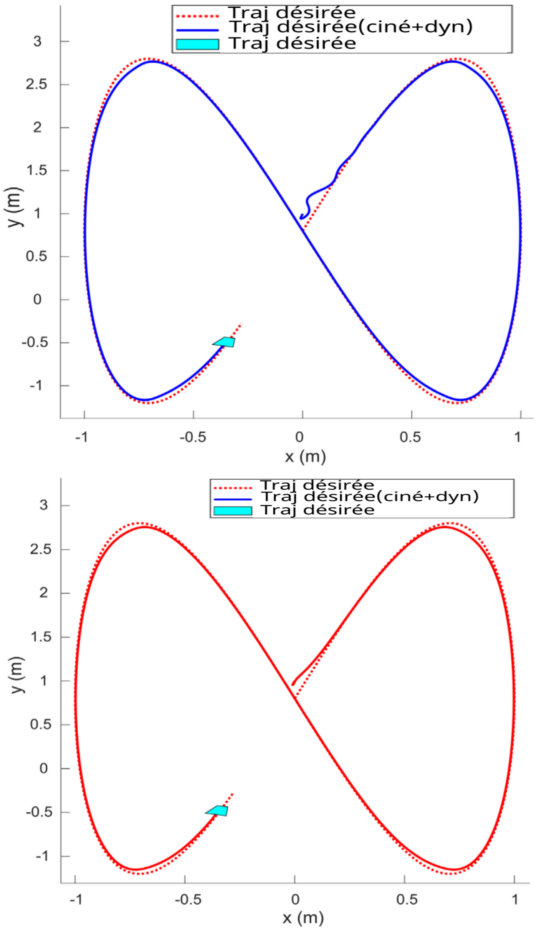

FIGURE 3.16 – Comportement du contrôleur (dynamique et cinématique (bleu)) à gauche par rapport à un contrôleur (cinématique uniquement (rouge)) à droite pour la même trajectoire de référence du forme 8

dûes aux non-linéarités, aux paramètres du contrôleur, aux saturations, ce qui sont très important lors des implémentations expérimentales.

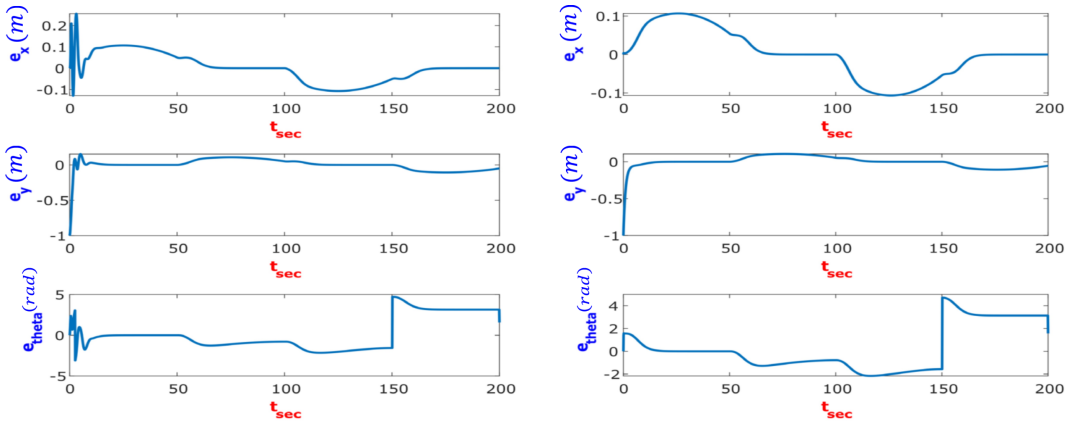

FIGURE 3.17 – Une comparaison entre les erreurs de suivi des deux contrôleurs (proposé à gauche et cinématique uniquement à droite) pour le scénario du rectangle

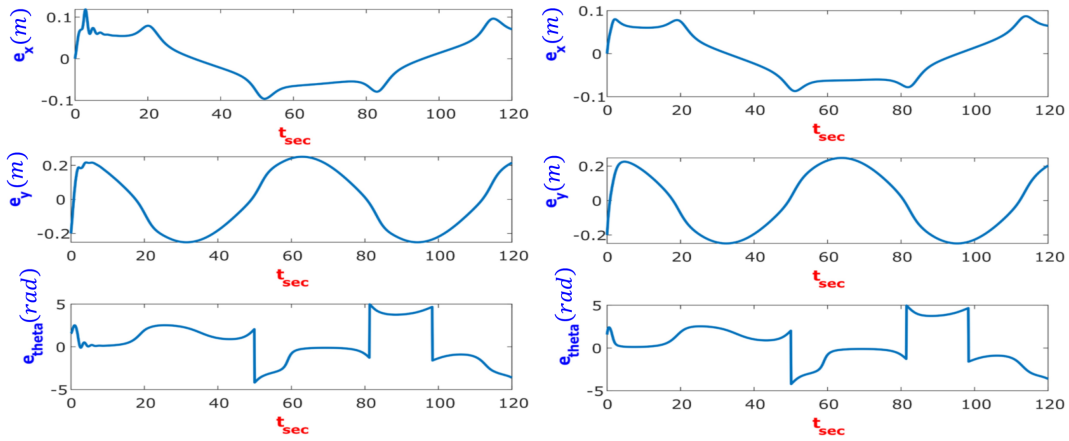

FIGURE 3.18 – Une comparaison entre les erreurs de suivi des deux contrôleurs (proposé à gauche et cinématique uniquement à droite) pour le scénario du forme 8

Les Figures (3.17) et (3.18) illustrent une comparaison entre les erreurs de suivi par application des deux contrôleurs pour les deux scénarios décrits précédemment : à gauche, on a représenté les erreurs de suivi, relatives aux contrôleurs (cinématique et dynamique), par contre à droite, on a représenté les erreurs de suivi, obtenues par le contrôleur (cinématique uniquement). A partir de ces résultats graphiques, on peut aussi constater lorsque le robot démarre, la différence d'erreur est très nette. Enfin, les tracés des vitesses linéaires et angulaires sont représentés par les Figures (3.19) et (3.20).

Les graphes présentés sur les Figures (3.19) et (3.20), correspondent aux vitesses linéaires et angulaires souhaitées et réelles du robot mobile respectivement pour les trajectoire de référence rectangle et lemniscate. Pour chaque figure, les résultats obtenus par application du contrôleur proposé sont représentés sur la gauche et ceux obtenus par le contrôleur cinématique sont données sur la droite.

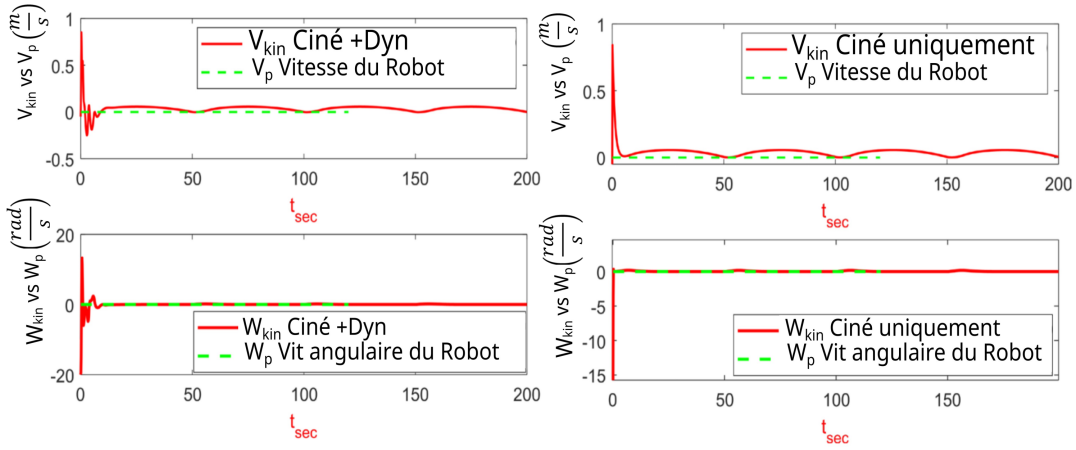

FIGURE 3.19 – Comparaison entre les vitesses linéaires (en haut) et les vitesses angulaires (en bas) des deux contrôleurs (proposé à gauche et cinématique uniquement à droite) pour le scénario du rectangle

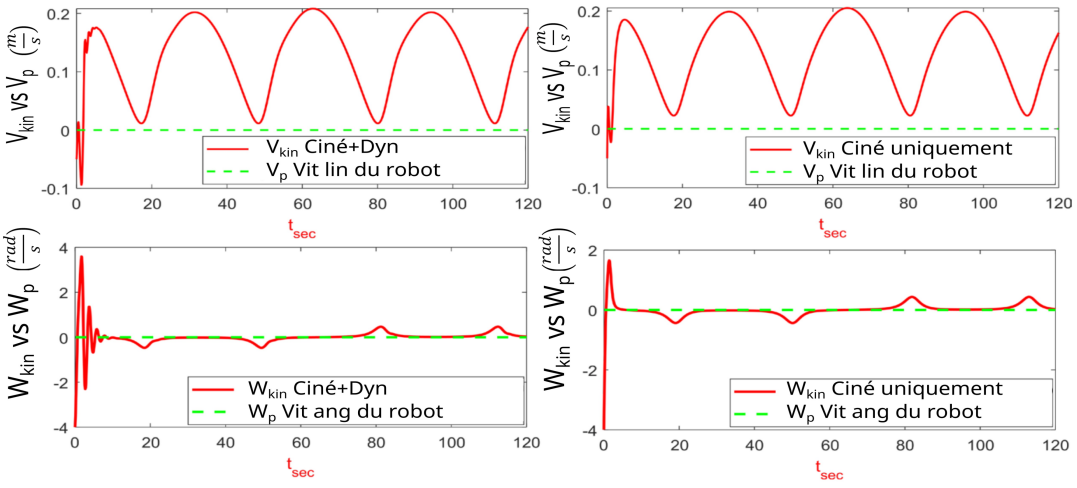

FIGURE 3.20 – Comparaison entre les vitesses linéaires (en haut) et les vitesses angulaires (en bas) des deux contrôleurs (proposé à gauche et cinématique uniquement à droite) pour le scénario du forme 8

Ces résultats graphiques permettent de mettre en évidance que le contrôleur proposé prend en compte les vitesses, ce qui est un avantage par rapport au contrôleur cinématique qui n'a aucune influence sur celles-ci.

3.4.2.1 Comparaison des performances des contrôleurs

Il est important de noter que chaque métrique évaluée offre un aperçu unique de la performance du système de contrôle. Par conséquent, l'analyse des différentes métriques permettra une compréhension plus approfondie des forces et des faiblesses de chaque approche de contrôle dans les scénarios de suivi de trajectoire rectangulaire et de lemniscates. Les deux tableaux présentés seront discutés pour identifier les avantages et les limitations de chaque méthode, ainsi que les implications pratiques pour le contrôle en expérimental des robots mobiles.

A- Trajectoire rectangulaire

Métrique	Cinématique et Dynamique	Cinématique
Erreur Moyenne de Suivi	0.0928 [m]	0.0913 [m]
Erreur RMS de Suivi	0.1109 [m]	0.1038 [m]
Erreur en régime permanent	0.0504 [m]	0.0544 [m]
Dépassement Maximal	1.0000 %	1.0000 %

Dans le scénario de suivi de trajectoire rectangulaire, les deux contrôleurs présentent des performances similaires en termes d'erreur moyenne de suivi de trajectoire et d'erreur RMS. Cependant, le contrôleur basé uniquement sur la cinématique présente une légèrement meilleure erreur en régime permanent, indiquant une convergence légèrement plus rapide vers la trajectoire désirée à long terme. Le dépassement maximal reste identique pour les deux contrôleurs.

Discussion des Métriques

1. **Erreur Moyenne de Suivi de Trajectoire** : Bien que les deux contrôleurs présentent des résultats similaires en termes d'erreur moyenne de suivi de trajectoire, il est important de noter que le contrôleur cinématique + dynamique offre une meilleure représentation du comportement réel du robot. Cela est dû à l'incorporation du modèle dynamique, qui prend en compte les effets des forces et des moments sur le mouvement du robot, ce qui peut conduire à des trajectoires plus précises.
2. **Erreur RMS de Suivi de Trajectoire** : Les deux contrôleurs montrent des performances comparables en termes d'erreur RMS de suivi de trajectoire. Cependant, le contrôleur cinématique + dynamique pourrait offrir une meilleure robustesse face aux perturbations externes en raison de sa capacité à modéliser les dynamiques du système.
3. **Erreur en régime permanent** : Le contrôleur cinématique + dynamique présente une erreur en régime permanent légèrement supérieure, ce qui peut être attribué à une meilleure représentation des propriétés dynamiques du robot. Cependant, cette différence est négligeable et n'affecte pas significativement la performance globale.
4. **Dépassement Maximal** : Bien que les deux contrôleurs présentent des dépassements maximaux similaires, le contrôleur cinématique + dynamique

est susceptible de mieux reproduire le comportement réel du robot en raison de l'incorporation du modèle dynamique, ce qui pourrait conduire à une meilleure gestion des dépassements dans des situations réelles.

La comparaison des performances met en évidence l'importance du modèle dynamique dans la représentation précise du comportement d'un robot. Bien que le contrôleur basé uniquement sur la cinématique offre des performances acceptables, l'ajout du modèle dynamique permet d'améliorer la précision, la robustesse et la fidélité de la simulation aux conditions réelles.

B- Trajectoire de forme en 8

Métrique	**Cinématique et Dynamique**	**Cinématique**
Erreur Moyenne de Suivi de Trajectoire	0.1745 [m]	0.1751 [m]
Erreur RMS de Suivi de Trajectoire	0.1855 [m]	0.1848 [m]
Erreur en régime permanent	0.2267 [m]	0.2132 [m]
Dépassement Maximal	0.2582 %	0.2552 %

Discussion des Métriques

1. **Erreur Moyenne de Suivi de Trajectoire** : Bien que les deux contrôleurs présentent des résultats similaires en termes d'erreur moyenne de suivi de trajectoire, il est important de noter que le contrôleur cinématique + dynamique offre une meilleure représentation du comportement réel du robot. Cela est dû à l'incorporation du modèle dynamique, qui prend en compte les effets des forces et des moments sur le mouvement du robot, ce qui peut conduire à des trajectoires plus précises.
2. **Erreur RMS (Root Mean Square) de Suivi de Trajectoire** : Les deux contrôleurs montrent des performances comparables en termes d'erreur RMS de suivi de trajectoire. Cependant, le contrôleur cinématique + dynamique pourrait offrir une meilleure robustesse face aux perturbations externes en raison de sa capacité à modéliser les dynamiques du système.
3. **Erreur en régime permanent** : Le contrôleur cinématique + dynamique présente une erreur en régime permanent légèrement supérieure, ce qui peut être attribué à une meilleure représentation des propriétés dynamiques du robot. Cependant, cette différence est négligeable et n'affecte pas significativement la performance globale.
4. **Dépassement Maximal** : Bien que les deux contrôleurs présentent des dépassements maximaux similaires, le contrôleur cinématique + dynamique est susceptible de mieux reproduire le comportement réel du robot en raison de l'incorporation du modèle dynamique, ce qui pourrait conduire à une meilleure gestion des dépassements dans des situations réelles.

La comparaison des performances met en évidence l'importance du modèle dynamique dans la représentation précise du comportement d'un robot. Bien que le contrôleur basé uniquement sur la cinématique offre des performances acceptables, l'ajout du modèle dynamique permet d'améliorer la précision, la robustesse et la fidélité de la simulation aux conditions réelles.

3.4.3 Résultats expérimentaux

Dans cette section, nous illustrons des résultats expérimentaux du contrôleur suggéré sans adaptation des paramètres sur le Qbot2e, puis avec adaptation des paramètres pour le suivi d'une trajectoire Circulaire. Le contrôleur proposé est simulé à l'aide du logiciel Quarc et des blocs dans l'environnement MATLAB/SIMULINK. Les caractéristiques du robot Qbot2e sont tirées de [105].

3.4.3.1 Résultats expérimental du contrôleur cinématique et dynamique

Après de nombreux tests et modifications, nous avons réussi à obtenir un suivi acceptable de la trajectoire circulaire. Ces résultats seront présentés ci-dessous.

Les postures initiale et de référence du robot ont été fixées à $q_{ref} = [1, 0, 0]^T$ et la posture initiale du robot a été fixée à $q_0 = [1, 0, 0]^T$. les équations de la trajectoire de référence utilisées pour la trajectoire circulaire sont données par (3.73) :

$$\begin{cases} x_d = sin(t) \\ y_d = cos(t) \\ \theta_d = arctan2(\dot{y}_d, \dot{x}_d) \end{cases} \tag{3.73}$$

Les résultats expérimentaux du contrôleur suggéré sur le Qbot2e dans le scénario de forme circulaire sont représentés dans les Figure (3.21) et (3.22).

La Figure (3.21) montre le comportement de Qbot2e par rapport à la trajectoire de référence, la trajectoire de Qbot2e est représentée par une ligne noire, la trajectoire de référence est représentée par des cercles verts.

Comme on peut le voir sur la Figure (3.21), le Qbot2e commence à reculer parce que l'erreur d'orientation était si grande ($e_\theta = \frac{\pi}{2}$), puis, il a commencé à avancer vers la trajectoire de référence et a réussi à l'atteindre et à y rester. Le seul inconvénient de ce contrôleur dans ce scénario est qu'il n'a pas pu rattraper la trajectoire au bon moment, car la trajectoire a une vitesse constante, et le contrôleur a des paramètres fixes. Dans d'autres expériences, nous avons augmenté les paramètres du contrôleur de $[1, 1]^T$ à $[1.3, 1.3]^T$, le robot a réussi à atteindre la trajectoire plus rapidement, et n'a eu aucun problème à la rattraper, mais il avait un comportement oscillatoire, à cause du contrôleur qui essaie constamment de rattraper les coordonnées $[x, y]$, mais θ changeant à chaque instant, le contrôleur pense qu'il n'est pas sur la bonne voie, encore une fois, tout ceci est dû à la nature non flexible des paramètres de notre contrôleur.

Un comportement oscillatoire n'est jamais une bonne idée en phase d'expérimentation ou de pilotage en temps réel du robot. Par conséquent, nous avons choisi d'utiliser $[1, 1]^T$ avec une petite erreur au lieu d'avoir un comportement oscillatoire qui pourrait nuire à notre robot. Néanmoins, notre contrôleur a fait un très bon travail et nous en sommes satisfaits.

La Figure (3.22) illustre les tracés des erreurs, l'erreur de x en rouge, l'erreur de y en vert, et l'erreur de θ en bleu, comme il est montré, les erreurs en $[x, y]^T$ convergent et seulement après une seule second, ils sont inférieures à $\pm 0.25 mètres$ et reste là pour la simulation entière, ceci valide les simulations obtenue avec le contrôleur proposé.

L'erreur dans θ est maintenue à une valeur non nulle car le robot n'a pas pu rattraper la trajectoire à temps, c'est-à-dire, $[x, y]^T$, c'est pourquoi l'orientation de référence est différente de l'orientation réelle à tout moment.

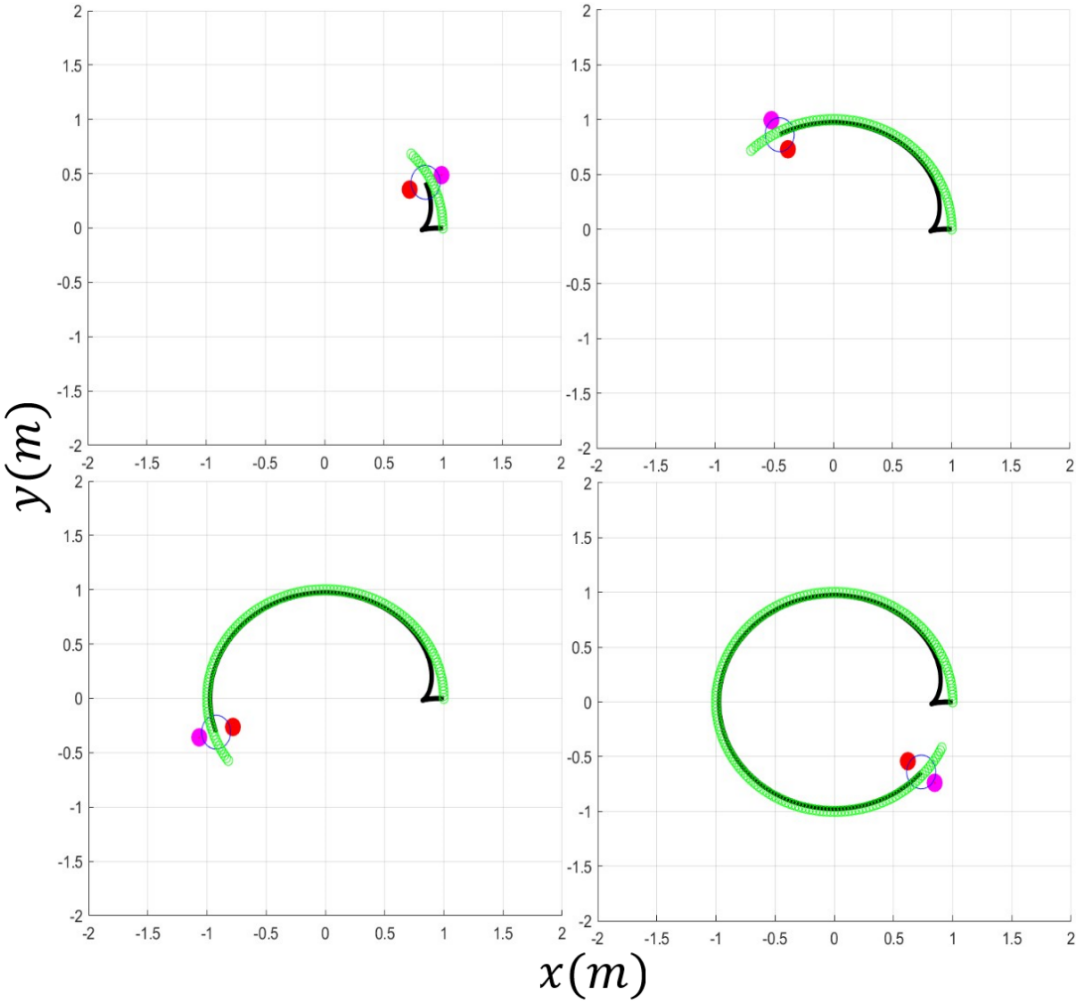

FIGURE 3.21 – Comportement de la trajectoire de Qbot2e avec le contrôleur suggéré

Pour illustrer le problème avec notre résultat expérimental sur ce contrôleur, une Figure (3.23) avec la trajectoire de référence plus la trajectoire actuelle du Qbot2e est présentée.

La trajectoire de référence en bleu contre la trajectoire actuelle en rouge par rapport à l'état x est montrée en haut, tandis que pour l'état y au milieu, et enfin pour l'état θ en bas.

La Figure (3.23) illustre clairement le retard dans chaque état.

Comme nous l'avons dit précédemment, cette solution n'est pas tout à fait optimale, c'est pourquoi nous avons dû la résoudre en ajoutant une régulation adaptative en ligne des paramètres, les résultats obtenus après cette opération seront décrits ci-après.

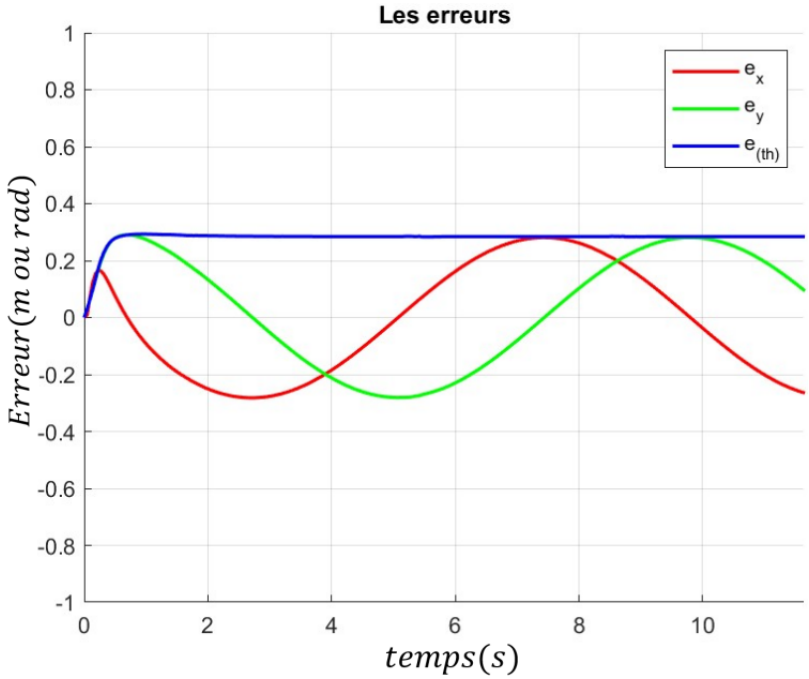

FIGURE 3.22 – Comportement de l'erreur de position avec le contrôleur suggéré

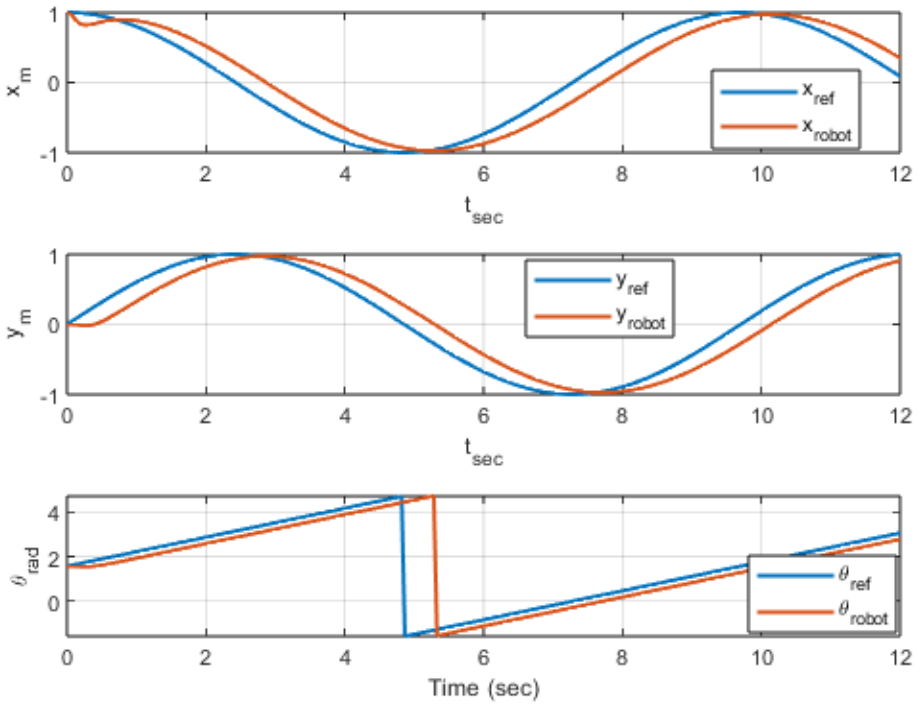

FIGURE 3.23 – Comportement des erreurs de position et d'orientation avec le contrôleur suggéré

3.4.3.2 Résultats expérimental du contrôleur ANFIS et dynamique

Les postures initiale et de référence du robot ont été fixées à $q_{ref} = [1, 0, 0]^T$ et la posture initiale du robot a été fixée à $q_0 = [1, 0, 0]^T$. les équations de la trajectoire de référence utilisées pour la trajectoire circulaire sont données par (3.74) :

$$\begin{cases} x_d = sin(t) \\ y_d = cos(t) \\ \theta_d = arctan2(\dot{y}_d, \dot{x}_d) \end{cases} \tag{3.74}$$

Les résultats expérimentaux du contrôleur suggéré sur le Qbot2e dans le scénario de forme circulaire sont représentés dans les Figure (3.24) et (3.25).

La Figure (3.24) montre le comportement de Qbot2e par rapport à la trajectoire de référence, la trajectoire de Qbot2e est représentée par une ligne noire, la trajectoire de référence est représentée par des cercles verts.

Comme on peut le voir, le robot suit très bien la trajectoire avec une erreur presque nulle, ceci n'a été rendu possible qu'après l'utilisation du contrôleur ANFIS.

La Figure (3.25) illustre les tracés des erreurs, l'erreur de x en rouge, l'erreur de y en vert, et l'erreur de θ en bleu, comme il est montré, les erreurs convergent et seulement après une seule second, ils sont inférieures à $\pm 0.02 mètres$ et reste là pour la simulation entière, ceci montre l'efficacité et la robustesse du contrôleur avec la technique ANFIS proposée.

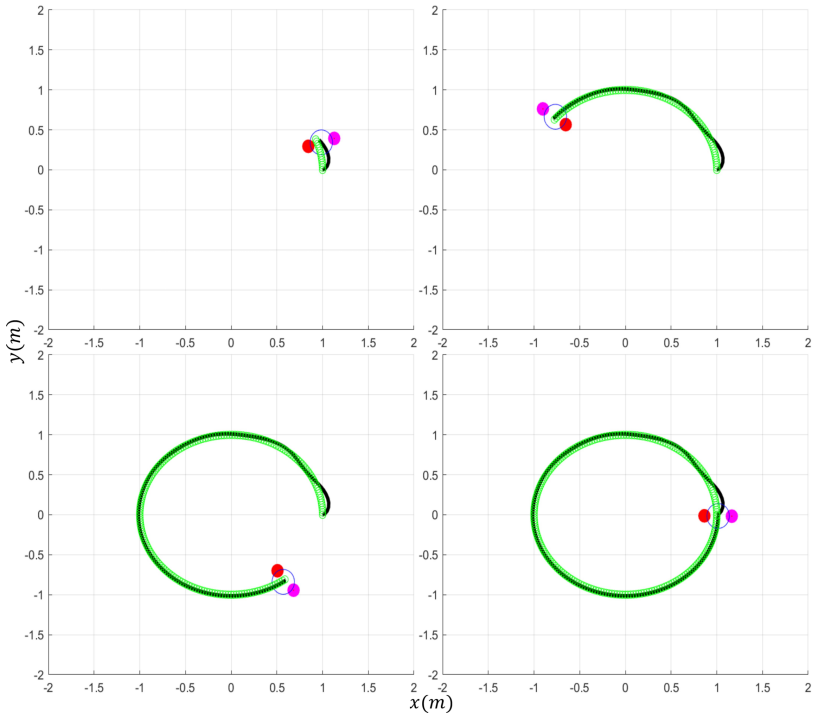

FIGURE 3.24 – Comportement de la trajectoire de Qbot2e avec le contrôleur suggéré

La Figure (3.26) illustre clairement le retard éliminé grâce au régulateur ANFIS dans chaque état par rapport au resultats précédentes .

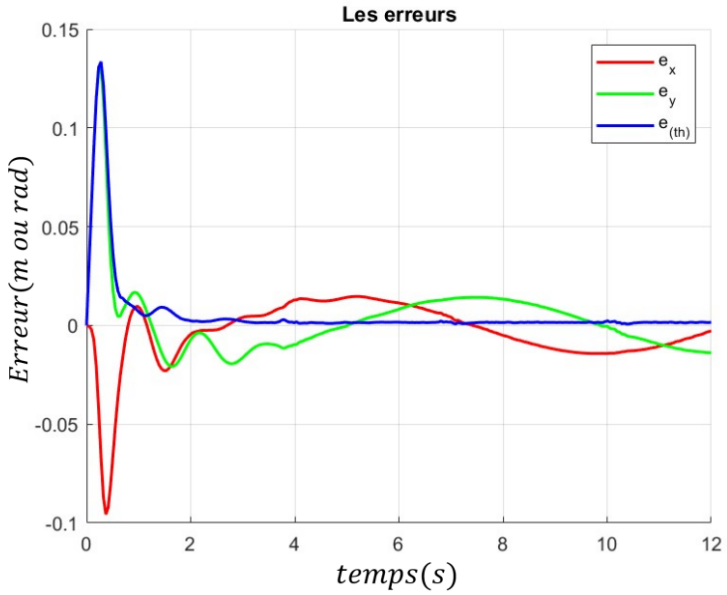

FIGURE 3.25 – Comportement de l'erreur de position avec le contrôleur suggéré

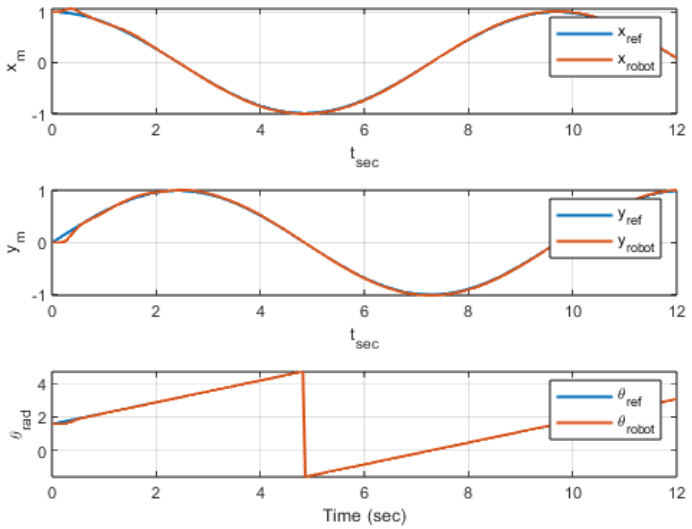

FIGURE 3.26 – Comportement de les erreurs des positions et d'orientation avec le contrôleur suggéré

3.4.3.3 Comparaison des Performances des Contrôleurs

Dans cette section, nous comparons les performances des contrôleurs de rétroaction et des contrôleurs ANFIS pour la navigation d'un robot mobile. Les résultats obtenus pour chaque contrôleur sont résumés dans le tableau ci-dessous :

Métrique	**Contrôleur de Rétroaction**	**Contrôleur ANFIS**
Erreur Moyenne de Suivi	0.2781 [m]	0.0183 [m]
Erreur RMS de Suivi	0.2795 [m]	0.0267 [m]
Erreur en régime permanent	0.2808 [m]	0.0142 [m]
Dépassement Maximal	0.9161 %	0.9831 %

TABLE 3.4 – Comparaison des performances des contrôleurs de rétroaction et des contrôleurs ANFIS

La comparaison des performances des contrôleurs révèle plusieurs différences significatives :

Le contrôleur ANFIS affiche une erreur moyenne et une erreur RMS de suivi de trajectoire beaucoup plus faibles par rapport au contrôleur de rétroaction, indiquant une meilleure précision dans le suivi de la trajectoire de référence.

Erreur en régime permanent : De même, le contrôleur ANFIS présente une erreur en régime permanent inférieure à celle du contrôleur de rétroaction, ce qui suggère une meilleure précision de positionnement une fois le système stabilisé.

Dépassement Maximal : Le contrôleur ANFIS affiche également un dépassement maximal légèrement plus élevé que le contrôleur de rétroaction, indiquant une réponse légèrement plus agressive du système dans certaines situations.

Ces résultats suggèrent que le contrôleur ANFIS offre généralement une meilleure performance en termes de précision de suivi de trajectoire et de positionnement par rapport au contrôleur de rétroaction. Cependant, il présente également un léger compromis en termes de dépassement maximal.

3.5 Conclusion

Le travail présenté dans ce chapitre se concentre sur l'implémentation de la stratégie de commande par retour d'état asymptotiquement stable à la commande dynamique d'un robot mobile à entraînement différentiel cascadé avec un contrôleur asymptotiquement stable pour la cinématique du robot, pour le suivi de trajectoire de référence rectangle et lemniscate. et la même stratégie de commande pour le suivi de trajectoire circulaire pour le robot Qbot2e de Quanser.

Cette stratégie de commande utilise deux contrôleurs dont l'un est un contrôleur de haut niveau et l'autre est un contrôleur bas niveau. Le contrôleur de haut niveau calcule les vitesses linéaires et angulaires puis les envoie au contrôleur de bas niveau qui commande à son tour les moteurs du robot mobile.

Par ailleurs, cette stratégie de commande est très sensible aux paramètres de contrôle (c'est une de ses limites), c'est pour cela qu'on a utilisé un régulateur ANFIS pour éliminer cette limitation dans l'étude expérimentale.

L'erreur de suivi de trajectoire produite dans le virage est due au rayon de braquage du robot mobile. Cette erreur s'avère être la principale limite de cette stratégie

de contrôle. Cependant, cette erreur dans le virage peut être réduite en diminuant la vitesse de braquage, donnant ainsi le temps au robot de répondre au signal de contrôle.

La prise en compte de la dynamique du robot mobile par cette stratégie de commande proposée la rend plus adaptée aux résultats expérimentaux puisqu'elle prend en compte les forces internes du système. Ces non-linéarités ne peuvent pas être négligées. Les résultats expérimentaux obtenus confirment cette contrainte.

Dans le prochain chapitre, nous aborderons la commande prédictive par modèle d'un robot mobile en utilisant la contrainte du coût terminal pour réduire l'erreur de suivi de trajectoire lors des virages, comme vu dans ce chapitre.

CHAPITRE 4

Commande prédictive non linéaire pour la navigation d'un robot mobile à entrainement différentiel

4.1 Introduction

La commande prédictive par modèle (MPC) est l'une des techniques de commande avancée les plus fréquemment utilisée dans l'industrie. L'objectif de la MPC est de calculer une séquence de commande future dans un horizon temporel spécifié de sorte que la prédiction de la sortie du système à contrôler soit proche d'une valeur de référence. Cet objectif est atteint en minimisant une fonction de coût par rapport aux actions de commande futures à travers une étape d'optimisation en ligne, où un ensemble d'actions de commande et de contraintes d'état du système sont satisfaites [106].

Plusieurs variantes de MPC sont disponibles dans la littérature à savoir : le MPC linéaire et le MPC non linéaire (NMPC), ont été utilisées pour résoudre les deux principaux objectifs de commande des robots mobiles.

Le MPC linéaire utilise une version linéarisée des équations de mouvement du robot, ce qui lui permet d'être utilisé uniquement pour le problème de suivi de trajectoire [43, 107] ou son cas particulier non paramétré dans le temps connu sous le nom de suivi de chemin [106, 108].

Le NMPC, qui utilise le modèle de mouvement non linéaire explicite des robots, a été utilisée pour les problèmes de suivi [109, 110] ; les problèmes de régulation [111, 112] ; et les deux [113].

L'utilisation d'un horizon de contrôle prédictif pose un problème de stabilité comme mentionné dans [111]. Il a été démontré que la stabilité peut être assurée en utilisant des contraintes d'égalité d'état final pour un horizon décroissant fini [114, 115]. Une analyse plus poussée montre qu'en ajoutant une pénalité d'état final, la limite d'égalité d'état final peut être relaxée comme une inégalité d'état final [109, 111].

Un autre critère de stabilité a été introduit dans [113], centré sur la commande prédictive contractive au premier état. Dans [109, 111, 114] il est indiqué que l'obtention de la stabilité à l'aide de la contrainte d'égalité terminale prend beaucoup de

temps ce qui fait pratiquement une tâche impossible à gérer. Néanmoins, de nombreux progiciels d'optimisation dynamique qui mettent en œuvre la commande prédictive de modèle non linéaire ont été développés en raison de l'évolution du matériel et du développement d'algorithmes numériques efficaces [116, 117]. En comparaison avec les paquets d'optimisation développés précédemment, un paquet récemment développé (CasADi) [118], qui implémente facilement les problèmes de NMPC, s'est avéré être un logiciel libre, convivial, extensible sur le plan informatique [118]. Il a été noté, selon la littérature décrite ci-dessus, qu'une étude utilisant la NMPC en temps réel est nécessaire pour les deux objectifs clés de contrôle des robots mobiles non holonomiques, où des contraintes d'égalité finale, stabilisantes sont considérées.

Dans ce chapitre, nous souhaitons exposer notre principale contribution à la thèse, qui consiste à proposer un NMPC qui utilise une contrainte d'état final pour la stabilité, une distance de norme 2 entre la pose du robot et la contrainte de pose de l'obstacle pour la tâche de navigation avec évitement d'obstacle. Un modèle non linéaire de la cinématique du robot mobile est utilisé. Les contraintes des variables de contrôle sont également prises en compte et une fonction objectif quadratique est proposée pour calculer l'ensemble des signaux de contrôle, et ce dernier est résolu en utilisant la technique de tir multiple dans la boîte à outils CasADi car elle réduit le temps de simulation de 10 à 20 fois par rapport à la technique de tir simple ; le solveur utilisé pour le problème de contrôle optimal est l'IPOPT.

4.2 Formulation mathématique du contrôleur NMPC

Nous présentons d'une manière explicite une brève formulation mathématique de MPC. Comme discuté précédemment, dans MPC, l'entrée de commande appliquée au système est obtenue en résolvant un problème de commande optimal (OCP) en boucle ouverte à horizon fini, à chaque instant de décision.

Nous considérons maintenant la dynamique discrète (4.1) à un instant κ comme suit :

$$q(\kappa+1) = f(q(\kappa), u(\kappa)) \quad q(0) = q_0 \tag{4.1}$$

Ensuite, pour l'horizon de prédiction $N_c \in N$ et la séquence de commande en boucle ouverte

$$u = [u(0) \quad u(1) \quad ... \quad u(N-1)] \qquad \in \quad U^{N_c} \tag{4.2}$$

Le problème de commande optimale en ligne (OCP) de MPC peut être décrit comme suit :

$$\min_{u \in \Re^{n_u \times N_c}} J_{N_c}(q(\kappa), u)$$

$$soumis - \grave{a},$$

$$\begin{cases} q(0) = q_0 & \\ q(\kappa+1) = f(q(\kappa), u(\kappa)) & \forall \kappa \in [0, 1, ..., N_c - 1] \\ q(\kappa) \in Q & \forall \kappa \in [1, 2, ..., N_c] \\ u(\kappa) \in U & \forall \kappa \in [0, 1, ..., N_c - 1] \end{cases} \tag{4.3}$$

Où la fonction objective $J_{N_c}(q(\kappa), u) : X \times U^N \longrightarrow \Re^{*+}$ est généralement donnée par :

$$J_{N_c}(q(\kappa), u) = \sum_{\kappa=0}^{N_c-1} (\ell(q(\kappa), u(\kappa)) + F(q(N_c)) \tag{4.4}$$

Le premier terme de la fonction objective (coût) (4.4) est appelé coûts de fonctionnement et est calculé en pénalisant la norme euclidienne de l'écart de la prédiction de l'état du système $q(.)$ et de son état de référence $q_{ref}(.)$. Ceci se traduit par en pénalisant la norme euclidienne de l'écart de la commande $u(.)$ par rapport à sa référence $u_{ref}(.)$. Le terme $F(q(N_c))$ est appelé coût terminal et il correspond à l'écart de la dernière entrée de la trajectoire prédite par rapport à la référence. Le coût de fonctionnement $\ell(.) : Q \times U \longrightarrow \Re^{*+}$ est généralement donné comme suit :

$$\ell(q(.), u(.)) = \|q(.) - q_{ref}(.)\|^2_{Q_{opt}} + \|u(.) - u_{ref}(.)\|^2_{R_{opt}} \tag{4.5}$$

Où Q_{opt} et R_{opt} sont des matrices de pondération symétriques définies positives de dimensions appropriées.

Il faut mentionner ici que, dans le cas d'une référence statique (régulation MPC), la référence de commande $u_{ref}(.) = 0$.

De plus, dans le cas du suivi, la référence de commande $u_{ref}(.)$ est la commande nominale à diriger l'état du système le long de la trajectoire de référence.

Comme il ressort de (4.5), la déviation de l'état q vers la référence q_{ref} ainsi que la déviation de la commande u vers la référence u_{ref} sont pénalisées le long de la trajectoire de prédiction. Alors que pénaliser l'écart d'état par rapport à sa référence est intuitif, pénaliser l'écart de la commande présente des avantages de calcul, c'est-à-dire que pénaliser la variable de commande peut rendre la solution optimale du problème de commande plus facile. De plus, lorsque la variable de commande est pénalisée, les valeurs de commande avec une énergie coûteuse peuvent être évitées [111]. Enfin, le terme de coût terminal $F(.) : Q \longrightarrow \Re^{*+}$ est donné par :

$$F(q(N_c)) = \|q(N_c) - q_{ref}(N_c)\|^2_{P_{opt}} \tag{4.6}$$

Où P_{opt} est une matrice de pondération définie positive pénalisant l'écart de la dernière entrée de la prédiction d'état, c'est-à-dire $q(N_c)$, par rapport à sa référence $q_{ref}(N_c)$. Comme nous le verrons plus loin dans ce chapitre, le coût terminal (4.6) est utilisé principalement pour assurer la stabilité en boucle fermée du MPC.

La séquence de commande de minimisation résultant de la résolution de l'OCP (4.3) est indiquée par

$$u^* := [u^*(0) \quad u^*(1) \quad ... \quad u^*(N_c - 1)] \quad \in U^{N_c} \tag{4.7}$$

Où $u^*(0)$ est l'action de commande à appliquer sur le système. L'algorithme sur le Tableau (4.1) résume le schéma MPC pour les systèmes non linéaires, par exemple. Equation (4.1). Voir également la Figure (4.1) pour un exemple de deux itérations MPC appliquées à un système monoentrée et monosortie (SISO).

TABLE 4.1 – L'algorithme de schéma MPC pour les systèmes non linéaires

1 : pour chaque instant d'échantillonnage $n = 0, 1, 2, ...$ faire
2 : Mesurer l'état actuel $q_p(n) := q(n) \in Q$ du système (4.1)
3 : définir $q_0 = q_p(n)$
4 : Trouver la séquence de commande minimisante de l'équation (4.7), qui satisfait $J_{N_c}(q_p, u^* = V_{N_c}(q_p))$
5 : Définir la loi de commande de rétroaction MPC $\mho_{N_c} : Q \longrightarrow U$ à q_p par $\mho_{N_c}(q_p) := u*(0)$
6 : Appliquer $\mho_{N_c}(q_p)$ au systeme (4.1)
7 : fin pour

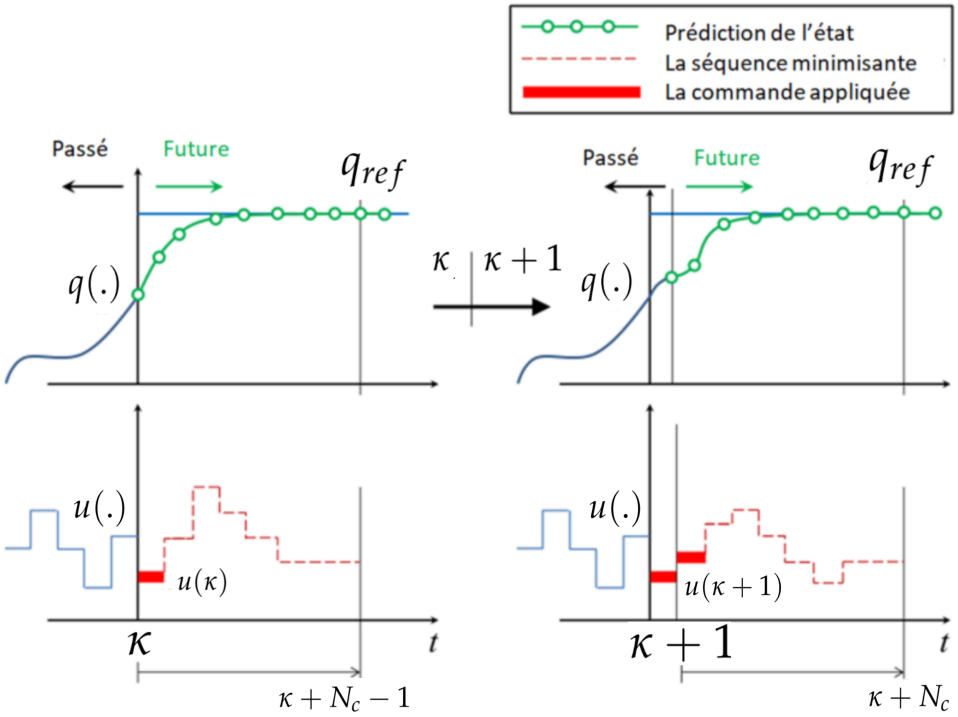

FIGURE 4.1 – Illustration de deux itérations MPC pour un système SISO simple

Sur la base de la fonction de coût introduite (4.4), une fonction de valeur (optimale) correspondante $V_{N_c} : Q \longrightarrow \Re^{*+} \cup \infty$ est définie pour un horizon de prédiction donné $N_c \in N$ comme suit :

$$V_{N_c}(q(0)) = \min_{u \in 2 \times N_c} J_{N_c}(q(0), u) = J_{N_c}(q_0, u^*) \tag{4.8}$$

En plus des contraintes d'état et de commande présentées dans OCP (4.3), qui représentent principalement les limites physiques de l'état et de la commande, l'OCP peut également être soumis à des contraintes terminales d'égalité ou d'inégalité. La contrainte d'égalité terminale peut être écrite comme suit : $q(N_c) - q_{ref}(N_c) = 0$, où cette contrainte nécessite que la dernière entrée de la trajectoire prédite soit égale à sa référence. De plus, la contrainte d'inégalité terminale peut être exprimée comme

suit : $q(N_c) \in \Omega(q_{ref}(N_c))$,où $\Omega(.) \subset Q$. Cette contrainte nécessite que la dernière entrée de la trajectoire prédite soit dans une région autour de la référence $q_{ref}(N_c)$. Les contraintes terminales d'égalité et d'inégalité sont utilisées dans la littérature pour garantir la stabilité du MPC en boucle fermée ; elles sont appelées contraintes de stabilisation terminales.

4.3 Application de la commande NMPC sur le robot mobile

Dans ce qui suit, une illustration des deux objectifs de contrôle, à savoir la stabilisation des points et le suivi de trajectoire, est présentée sur la classe de robots à entraînement différentiel des robots mobiles non holonomiques.

4.4 Stabilisation de la posture et suivi de la trajectoire

Afin d'illustrer les deux problèmes de commande du robot mobile, un robot de référence, représenté sur la Figure (4.2), est défini avec un vecteur d'état de référence $q_{ref} = [x_{ref}, y_{ref}, \theta_{ref}]^T$ et un vecteur de commande de référence $u_{ref} = [v_{ref}, \omega_{ref}]^T$, soumis à la même contrainte que le système (2.23) qui est exprimée pour plus de facilité par la relation suivante :

$$\dot{q} = \begin{bmatrix} \dot{x}_p \\ \dot{y}_p \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} cos(\theta) & 0 \\ sin(\theta) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{4.9}$$

Ainsi, son modèle de mouvement cinématique du robot de référence peut être donné comme suit :

$$\dot{q_{ref}} = \begin{bmatrix} \dot{x}_{p_{ref}} \\ \dot{y}_{p_{ref}} \\ \dot{\theta}_{ref} \end{bmatrix} = \begin{bmatrix} cos(\theta_{ref}) & 0 \\ sin(\theta_{ref}) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_{ref} \\ \omega_{ref} \end{bmatrix} \tag{4.10}$$

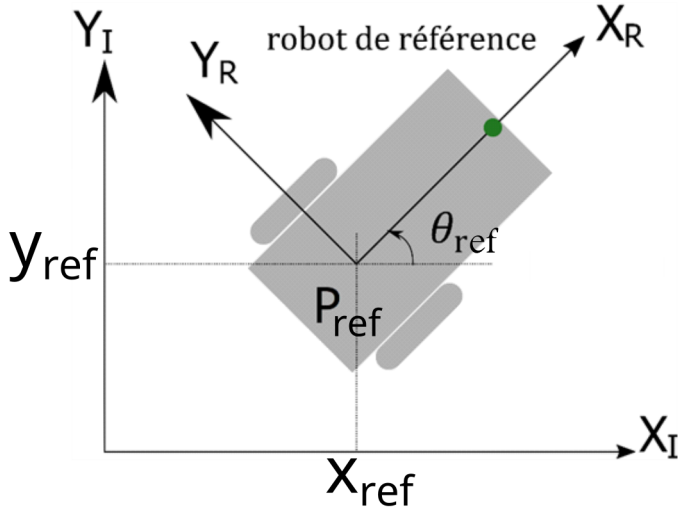

FIGURE 4.2 – Cinématique d'un robot de référence à entraînement différentiel

À ce stade, le problème de stabilisation du point convient au cas où le vecteur d'état de référence q_{ref} a une valeur constante correspondant à la position cible souhaitée, et que le vecteur de contrôle u_{ref} a des valeurs nulles pour la référence des vitesses linéaires et angulaires. En revanche, pour le problème de suivi de trajectoire, les vecteurs q_{ref} et u_{ref} ont des valeurs qui varient dans le temps en fonction de la trajectoire de référence choisie. Dans les deux cas, l'objectif de la stratégie de commande est de contrôler la dynamique du système représentée par la relation (4.9) en vue de suivre la trajectoire de référence donnée par l'équation (4.10). Par conséquent, un vecteur d'état d'erreur q_e peut être défini comme donné par l'expression (3.1) en utilisant la relation suivante :

$$q_e = \begin{bmatrix} x_e \\ y_e \\ \theta_e \end{bmatrix} = R(\theta)(q_{ref} - q_p) = R(\theta) \begin{bmatrix} x_{ref} - x_p \\ y_{ref} - y_p \\ \theta_{ref} - \theta_p \end{bmatrix} \tag{4.11}$$

On peut facilement voir que les deux objectifs de contrôle peuvent être atteints en ramenant le vecteur d'état q_e à zéro. Le modèle dynamique d'erreur est obtenu en différenciant (Equations (3.2), (3.3), et (3.4)) ce qui conduit au système d'équations d'erreur suivant :

$$\dot{q}_e = \begin{bmatrix} \dot{x}_e \\ \dot{y}_e \\ \dot{\theta}_e \end{bmatrix} = \begin{bmatrix} \omega y_e - v_p + v_{ref} cos(\theta_e) \\ -\omega x_e + v_{ref} sin(\theta_e) \\ \omega_{ref} - \omega \end{bmatrix} \tag{4.12}$$

Une version linéarisée du modèle d'erreur (4.12) a la forme suivante :

$$\dot{q}_e = \begin{bmatrix} \dot{x}_e \\ \dot{y}_e \\ \dot{\theta}_e \end{bmatrix} = \begin{bmatrix} 0 & \omega_{ref} & 0 \\ -\omega_{ref} & 0 & v_{ref} \\ 0 & 0 & 0 \end{bmatrix} q_e + \begin{bmatrix} -1 & 0 \\ 0 & 0 \\ 0 & -1 \end{bmatrix} u_e \tag{4.13}$$

Où u_e est définie comme :

$$u_e = \begin{bmatrix} -v + v_{ref} cos(\theta_e) \\ \omega_{ref} - \omega \end{bmatrix} \tag{4.14}$$

On peut facilement vérifier que la commandabilité du modèle (4.13) est perdue lorsque les valeurs appropriées aux vitesses de reférence linéaire et angulaire s'approchent de l'origine, ce qui élimine l'applicabilité de la commande de la stabilisation du point. À ce stade, il faut mentionner que pour la commande de la stabilisation ponctuelle, une variante du modèle de mouvement pour (4.12) défini par les coordonnées polaires a également été considérée dans [111] ; cependant, dans le travail présenté ici, le modèle (4.12) a été uniquement utilisé pour atteindre les deux objectifs de contrôle. Ainsi, la structure du contrôleur n'a pas besoin d'être modifiée lors de l'application des deux objectifs de contrôle.

4.5 Commande prédictive NMPC stabilisante

Dans cette section, une brève description du schéma NMPC pour les robots mobiles non holonomiques est présentée, mettant en évidence les hypothèses nécessaires pour la preuve de la stabilité des contraintes d'égalité terminale. La forme générale du système de contrôle non linéaire donné en équation (4.9) peut être exprimée comme suit :

$$\dot{q} = f(q(t), u(t)) \tag{4.15}$$

où $q(t) \in \Re^n$ et $u(t) \in \Re^m$ sont les vecteurs respectivement d'état de dimension n et de commande de dimension m, respectivement. L'objectif de la commande est de calculer une entrée de commande admissible $u(t)$ pour amener le système (4.15) à se déplacer vers le point d'équilibre défini par $(q_e(t) = 0$ et $u_e(t) = 0)$, où $q_e(t)$ représente la différence entre le vecteur d'état du système et celui de référence, par contre $u_e(t)$ représente la différence entre le vecteur des entrées de commande du système et celui des entrées de commande de référence. L'objectif de l'algorithme de contrôle est de minimiser une fonction de coût pondérée exprimée par cette relation :

$$J(t, q_e(\tau_{opt}), u_e(\tau_{opt})) = \int_t^{t+T} \ell(\tau_{opt}, q_e(\tau_{opt}), u_e(\tau_{opt}))d\tau_{opt} \tag{4.16}$$

où le coût de fonctionnement ℓ est donné par :

$$\ell(\tau_{opt}, q_e(\tau_{opt}), u_e(\tau_{opt})) = q_e(\tau_{opt})^T Q_{opt} q_e(\tau_{opt}) + u_e(\tau_{opt})^T R_{opt} u_e(\tau_{opt}) \tag{4.17}$$

Il est appelé le coût de fonctionnement. Q_{opt} et R_{opt} sont les matrices de poids symétriques définies positives.

Il est bien connu que la stabilité du contrôleur trouvé dans (4.16) n'est pas garantie en raison de l'utilisation d'un horizon décroissant fini [109].

La stabilité du NMPC peut être garantie en ajoutant un terme de pénalité d'état terminal à la fonction de coût et une contrainte d'état terminal à l'optimisation dans le contrôleur NMPC [109]. Par conséquent, la fonction de coût pour le problème de suivi sera modifiée comme suit :

$$J(t, q_e(\tau_{opt}), u_e(\tau_{opt})) = g(q_e(t+T) + \int_t^{t+T} \ell(\tau_{opt}, q_e(\tau_{opt}), u_e(\tau_{opt}))d\tau_{opt} \tag{4.18}$$

où $g(q_e(t+T))$ est la pénalité de l'état terminal et est supposé être une fonction continue et différentiable,$g(0) = 0$, et $g(q_e(t)) > 0 \ \forall \ q_e(t) \neq 0$, et T est l'horizon de prédiction.

Dans le cadre de l'optimisation décrit dans l'Équation (4.18), la fonction de pénalité $g(q_e(t+T))$ est utilisée pour quantifier l'écart de l'état du système par rapport à l'état terminal désiré au temps $t+T$. Ici, $q_e(t+T)$ représente l'erreur ou l'écart par rapport à l'état terminal désiré.

La fonction de pénalité g attribue une pénalité à cette erreur en fonction de sa magnitude, garantissant que les écarts par rapport à l'état terminal désiré sont correctement pris en compte dans le processus d'optimisation.

La fonction de pénalité g est définie comme suit :

$$g(q_e(t+T)) = \frac{1}{2}(q_e(t+T))^2 \tag{4.19}$$

Cette fonction de pénalité joue un rôle crucial dans la définition des objectifs d'optimisation, garantissant que les actions de contrôle résultantes guident efficacement le système vers l'état terminal désiré tout en minimisant les écarts.

Ainsi, au temps t, le problème d'optimisation en ligne et en boucle ouverte du contrôleur NMPC peut être décrit comme suit :

$$\begin{gathered}\min_u J_(t, q_e(\tau_{opt}), u_e(\tau_{opt})) \\ soumis - à \\ \begin{cases} \dot{q}(\tau_{opt}) = f(q(\tau_{opt}), u(\tau_{opt})) \\ u(\tau_{opt}) \in U, (\tau_{opt} \in [t, t+T]) \\ q_e(t+T) = 0 \end{cases}\end{gathered} \tag{4.20}$$

où $0 \in U \in \Re^m$ est un ensemble compact spécifiant les limites de saturation de l'entrée de commande, et $q_e(t+T) = 0$ définit les contraintes d'égalité de l'état terminal.

Comme indiqué dans [115], la stabilité du contrôleur peut être prouvée si les deux hypothèses suivantes sont satisfaites.

— Le vecteur d'état $q_{ref} \in Q$ est un point d'équilibre pour une valeur de commande admissible $u_{ref} \in U$, où q_{ref} est le vecteur d'état de référence, et $Q \in \Re^n$ est l'ensemble de l'espace d'état pour le vecteur d'état $q(t)$; cela signifie qu'il existe une valeur de commande $u_{ref} \in U$ telle que $f(q_{ref}, u_{ref}) = q_{ref}$.

— La fonction de coût de fonctionnement $\ell : X \times U \longrightarrow \Re^{*+}$ satisfait $\ell(q_{ref}, u_{ref}) = 0$ sous la forme $u_{ref} \in U$ obtenue à partir de la première hypothèse.

Ces hypothèses peuvent être facilement vérifiées dans notre cas. Une autre preuve de stabilité est incluse dans [111].

4.6 Évitement des obstacles

Comme le montre la Figure (4.3), nous sommes dans un cas où nous avons un obstacle à éviter sur la trajectoire planifiée du robot de telle sorte que le robot mobile démarre à partir d'un état (x, y) pour le conduire à la trajecoire de référence tout en évitant l'obstacle rencontré. Compte tenu de ces considérations, il s'avère nécessaire d'ajouter les obstacles à éviter comme une autre contrainte dans la formulation MPC.

Supposons que notre robot est représenté par un cercle et que l'obstacle est également représenté par un cercle et que nous ne voulons pas que ces deux cercles se croisent.

Supposons que le rayon du robot est de RR et le rayon de l'obstacle de RO. Donc la distance euclidienne entre les centres des deux doit être supérieure ou égale à la somme des deux rayons. Par conséquent, nous devons imposer la contrainte de chemin suivante

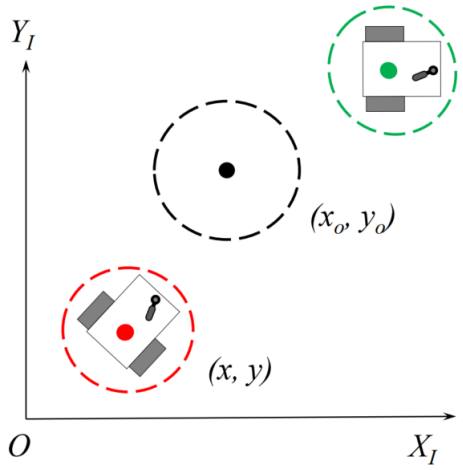

FIGURE 4.3 – Evitement d'un obstacle

$$\sqrt{(x - x_{obs})^2 + (y - y_{obs})^2} \geqslant RR + RO \tag{4.21}$$

Cela conduit à la contrainte de l'inégalité

$$-\sqrt{(x - x_{obs})^2 + (y - y_{obs})^2} + (RR + RO) \leqslant 0 \tag{4.22}$$

Le OCP devient :

$$\begin{gathered}\min_u J_(t, q_e(\tau_{opt}), u_e(\tau_{opt})) \\ soumis - à \\ \begin{cases} \dot{q}(\tau_{opt}) = f(q(\tau_{opt}), u(\tau_{opt})) \\ u(\tau_{opt}) \in U, (\tau_{opt} \in [t, t+T]) \\ q_e(t+T) = 0 \\ -\sqrt{(x - x_{obs})^2 + (y - y_{obs})^2} + (RR + RO) \leqslant 0 \end{cases}\end{gathered} \tag{4.23}$$

Le code développé instancie une routine d'exécution du NMPC à chaque étape des simulations est analysé dans la section suivante.

4.7 Résultats de la simulation et expérimentaux sur le Qbot2e

Comme dans le cas du chapitre précédent, cette section est subdivisée en deux sous-sections. La première sous-section traite des résultats de la simulation, qui ont

fait l'objet d'une publication [20], et la seconde sous-section traite les résultats expérimentaux obtenus par application des stratégies de commande développées dans ce chapitre sur un robot mobile Qbot2e de Quanser.

Toutes les simulations et les expérimentations ont été faites dans l'environnement MATLAB/SIMULINK, la connexion au Robot a été assurée via le logiciel Quarc fourni avec le robot.

Le fonctionnement du robot avec ce contrôleur a nécessité des mois d'essais et d'échecs, mais nous avons finalement trouvé une solution à chaque problème rencontré.

Une vidéo complète a été uploadée sur YouTube montrant les performances du robot en expérimental pour chaque scénario. L'URL fournie dans [119] vous conduit à la vidéo décrivant ces résultats.

4.7.1 Résultats de simulation

Les simulations que nous présenterons dans cette partie visent à évaluer les deux objectifs de contrôle du robot mobile à entraînement différentiel, à savoir stabilisation des points et le suivi de trajectoire en considérant le modèle (4.9) comme étant une approximation de son mouvement. Les paramètres d'optimisation en ligne, y compris la fréquence d'échantillonnage, le nombre de pas de l'horizon de prédiction N et les matrices de poids Q_{opt}, R_{opt} et P_{opt} sont choisis de manière à obtenir des performances satisfaisantes du contrôleur. Les résultats de simulation appropriés à la stabilisation des points sont présentés en premier, suivis par les résultats du suivi de la trajectoire, en considérant d'abord un environnement sans obstacles puis avec les obstacles tout en respectant le même ordre de la présentation des résultats de simulation.

par ailleurs, pour toutes les simulations, la matrice de pondération $Q_{opt} = P_{opt}$.

4.7.1.1 Résultats de la stabilisation des points sans obstacles

Dans le but de montrer les performances du contrôleur NMPC mis en œuvre pour la stabilisation des points, les résultats du robot exécutant la stabilisation du stationnement, sont présentés.

Dans ces simulations, le robot effectue 3 scénarios, il commence par la pose initiale, $q_0 = [0, 0, 0]^T(m, m, rad)$, après cela il va à $q_1 = [8.5, 8.5, 0]^T(m, m, rad)$ puis à $q_2 = [0.5, 8, \frac{\pi}{2}]^T(m, m, rad)$, finalement à $q_3 = [8, 1, \pi]^T(m, m, rad)$ et s'arrête là.

Pour le premier scénario, le robot est commandé pour se stabiliser à la pose $q_1 = [8.5, 8.5, 0]^T(m, m, rad)$ (un stationnement parallèle) venant de la pose initiale q_0.

Pour le deuxième scénario, le robot est commandé pour se stabiliser à la pose $q_2 = [0.5, 8, \frac{\pi}{2}]^T(m, m, rad)$ venant de la pose $q_1 = [8.5, 8.5, 0]^T(m, m, rad)$.

Pour le troisième scénario, le robot est commandé pour se stabiliser à la pose $q_3 = [8, 1, \pi]^T(m, m, rad)$ venant de la pose $q_2 = [0.5, 8, \frac{\pi}{2}]^T(m, m, rad)$.

Le pas de temps de mise à jour du contrôleur est choisi pour être de $0,3$ seconde avec le nombre d'étapes de prédiction $N = 30$, l'horizon de prédiction de temps est $T = 9$ secondes. Le poids des matrices Q_{opt} et R_{opt} de la fonction de coût (4.17)

sont choisies comme des matrices diagonales avec des éléments diagonaux définis comme $(15, 10, 0.1)$ pour Q_{opt}, et comme $(0.002, 0.002)$ pour R_{opt}.

$$Q_{opt} = \begin{bmatrix} 15 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & 0.1 \end{bmatrix} ; R_{opt} = \begin{bmatrix} 0.002 & 0 \\ 0 & 0.002 \end{bmatrix}$$

Afin d'obtenir la localisation exacte du robot et la satisfaction de la saturation de ses actionneurs, les limites de saturation du contrôleur, pour la vitesse linéaire v et la vitesse angulaire ω, sont définis comme suit :

$$\begin{bmatrix} -0.75 \\ -0.98 \end{bmatrix} \leq \begin{bmatrix} v\,(m/s) \\ \omega\,(rad/s) \end{bmatrix} \leq \begin{bmatrix} 0.75 \\ 0.98 \end{bmatrix} \tag{4.24}$$

Les résultats obtenus sont résumés dans les Figures (4.4) et (4.5). La Figure (4.4) (meilleure vue en couleurs) montre les trajectoires exécutées par le robot. La position et l'orientation du robot sont représentées respectivement par un triangle dont le centre de la base du triangle est situé à la position du robot, et le sommet du triangle représente l'orientation du robot.

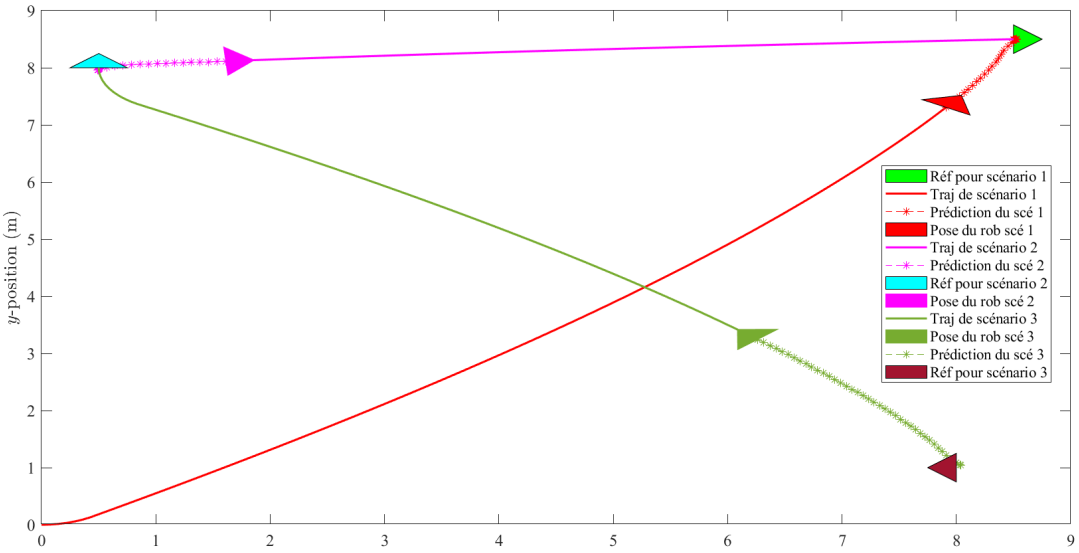

FIGURE 4.4 – Trajectoires exécutées en stabilisation ponctuelle

La position de départ du robot est point de départ du figure ou q_0 ;

Les triangles représentent le robot, les lignes représentent la trajectoire effectuée par le robot et les étoiles représentent la trajectoire prédite.

La couleur rouge est utilisée pour le premier scénario, la couleur rose pour le deuxième scénario et la couleur verte foncée pour le troisième et dernier scénario.

Le triangle vert clair représente la référence pour le premier scénario q_1, le bleu ciel pour la référence du deuxième scénario q_2 et le grenat foncé pour la référence du troisième scénario q_3.

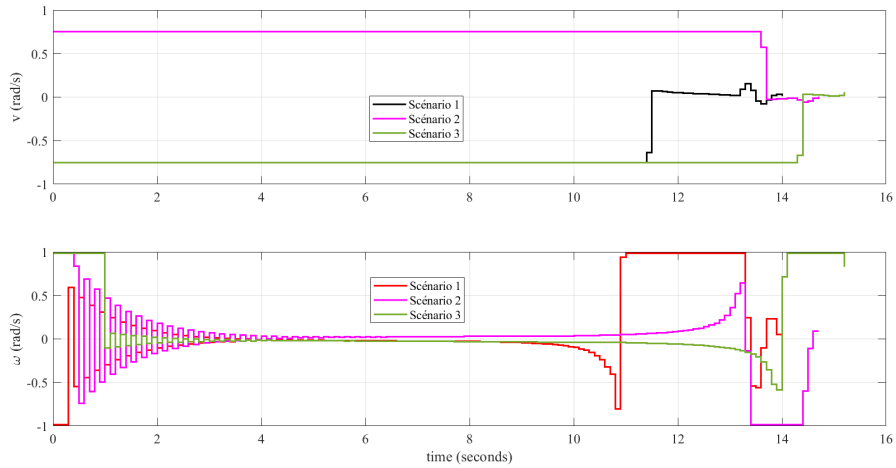

FIGURE 4.5 – Signaux de contrôle de la stabilisation des points

Comme on peut le voir dans la Figure (4.4), le contrôleur peut stabiliser le robot dans la position souhaitée dans tous les cas, au moyen de mouvements vers l'avant et vers l'arrière.

Les résultats montrent les performances de poursuite du contrôleur avec un coût de calcul raisonnable, pour le cas de la stabilisation ponctuelle. La Figure (4.5) (meilleure vue en couleurs) montre que les vitesses linéaires et angulaires du robot conforment aux limites de saturation données par (4.24)

4.7.1.2 Résultats du suivi de la trajectoire sans obstacles

Les performances du contrôleur NMPC, pour le suivi des trajectoires, ont été évaluées en considérant deux trajectoires de référence, à savoir les trajectoires de forme circulaire (4.25) et de forme octogonale (4.26).

Les paramètres des trajectoires (4.25) et (4.26) sont choisis de telle sorte que les vitesses linéaire et angulaire de référence ne dépassent pas les limites de saturation données par (4.24).

Comme dans le cas de la stabilisation ponctuelle, le robot mobile part de la pose initiale $q_0 = [0, 0, 0]^T(m, m, rad)$ L'environnement de performances est décrit par :

$$\begin{cases} x_{ref}(t) = 0.3 + 2sin(0.25t) \\ y_{ref}(t) = 2.3 + 2cos(0.25t) \end{cases} \tag{4.25}$$

$$\begin{cases} x_{ref}(t) = 0.3 + 1.5sin(0.3t) \\ y_{ref}(t) = 0.3 + 2.5cos(0.15t) \end{cases} \tag{4.26}$$

Le choix du pas de temps de mise à jour du contrôleur de $0,2$ seconde avec un nombre de pas de prédiction $N = 30$, conduit à un horizon de prédiction $T = 6$ secondes. Les points de départ des trajectoires de référence (4.25) et (4.26) ont été sélectionnés de manière à ce qu'une erreur initiale soit imposée au problème de suivi.

Pour le suivi de forme circulaire, les matrices de poids Q_{opt} et R_{opt} de la fonction de coût (4.18) sont choisies comme matrices diagonales avec des éléments diagonaux définis comme $(30, 30, 0.1)$ pour Q_{opt}, et comme $(50, 50)$ pour R_{opt}.

$$Q_{opt} = \begin{bmatrix} 30 & 0 & 0 \\ 0 & 30 & 0 \\ 0 & 0 & 0.1 \end{bmatrix} ; R_{opt} = \begin{bmatrix} 50 & 0 \\ 0 & 50 \end{bmatrix}$$

Pour le suivi de forme octogonale, les éléments diagonaux de Q_{opt} sont $(30, 30, 0.1)$, et les éléments diagonaux de R_{opt} sont $(9, 5)$.

$$Q_{opt} = \begin{bmatrix} 30 & 0 & 0 \\ 0 & 30 & 0 \\ 0 & 0 & 0.1 \end{bmatrix} ; R_{opt} = \begin{bmatrix} 9 & 0 \\ 0 & 5 \end{bmatrix}$$

Les Figures (4.6) et (4.7) montrent respectivement la trajectoire du robot et les signaux de commande représentés par les vitesses linéaire et angulaire, pour la trajectoire circulaire. Les Figures (4.8) et (4.9) présentent des résultats semblables aux résultats données respectivement par les Figures (4.6) et (4.7) pour la trajectoire de forme infini.

L'emplacement de départ du robot pour une référence circulaire est $q_0 = [2, 4, pi]^T$. et pour une référence en forme infini, l'état initiale est $q_0 = [2, 1, -\frac{\pi}{1,5}]$ La position et l'orientation du robot sont représentées respectivement par un triangle dont le centre de la base du triangle est situé à la position du robot, et le sommet du triangle représente l'orientation du robot.

Le triangle rouge représente le robot, le triangle grenat foncé représente l'état initiale, le triangle vert clair représente l'état finale, la ligne rouge représente la trajectoire effectuée par le robot, la ligne rose ou verte foncée représente la trajectoire de référence, et les étoiles représentent la trajectoire prédite.

Les Figures (4.7) et (4.9) montrent que les vitesses du robot convergent vers leurs valeurs de référence, et ne dépassent jamais leurs valeurs limites fixées par les contraintes de vitesses, malgré l'erreur initiale qu'elles présentaient.

Pour le suivi de la forme circulaire, on a observé que la valeur de l'erreur de position en régime permanent était comprise entre $(\pm 3, 27cm)$ pour x et y, et l'erreur d'orientation entre $(\pm 0, 04rad)$, le coût moyen de calcul par pas de temps étant de $(21millisecondes)$.

D'autre part, dans le cas du suivi de forme octogonale, la valeur de l'erreur de position en régime permanent a été observée à $(\pm 2, 75cm)$, et l'erreur d'orientation à $(\pm 0, 07rad)$, où le coût moyen de calcul par pas de temps est de $(21millisecondes)$ dans ce cas.

4.7.1.3 Résultats de la stabilisation des points avec obstacles

Les performances de poursuite du contrôleur NMPC mis en œuvre pour la stabilisation des points en évitant les obstacles sont présentées dans cette section. Le robot mobile part de la pose initiale $q_{ref} = [0, 0, 0]^T(m, m, rad)$.

Le robot mobile est commandé pour se stabiliser à la pose $q_{ref} = [8.5, 9.5, 0]^T$.

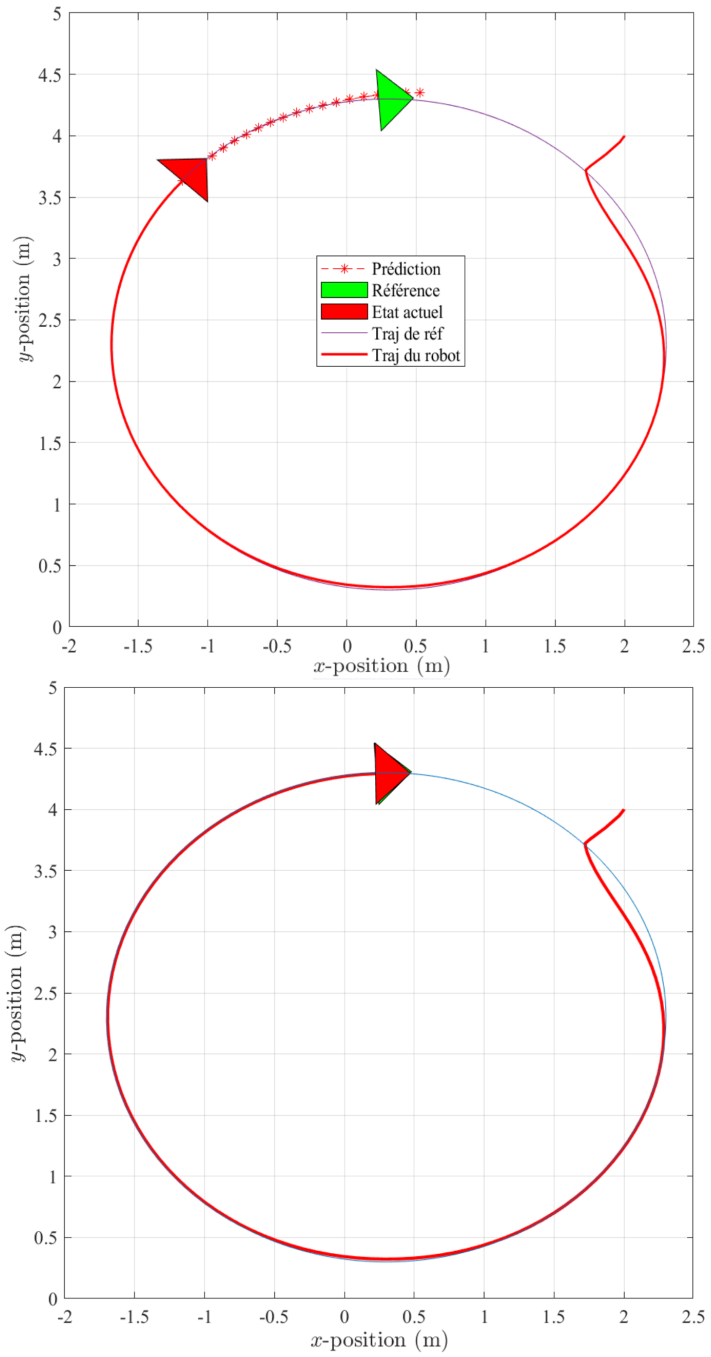

FIGURE 4.6 – Tracé de la trajectoire circulaire

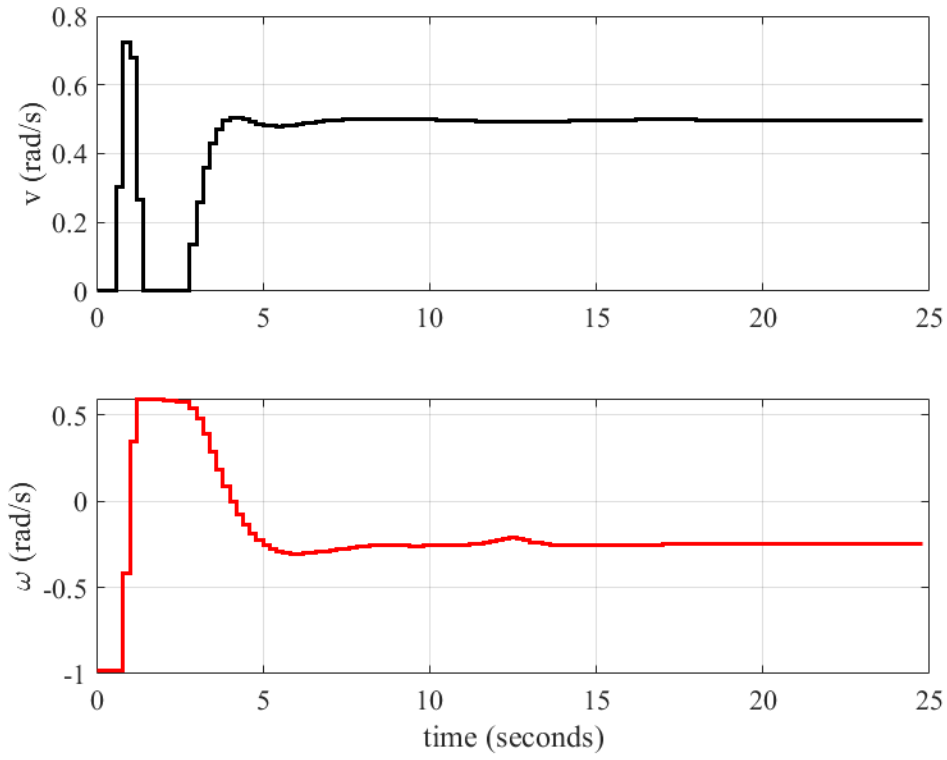

FIGURE 4.7 – Vitesses linéaire et angulaire de la trajectoire circulaire en forme de huit

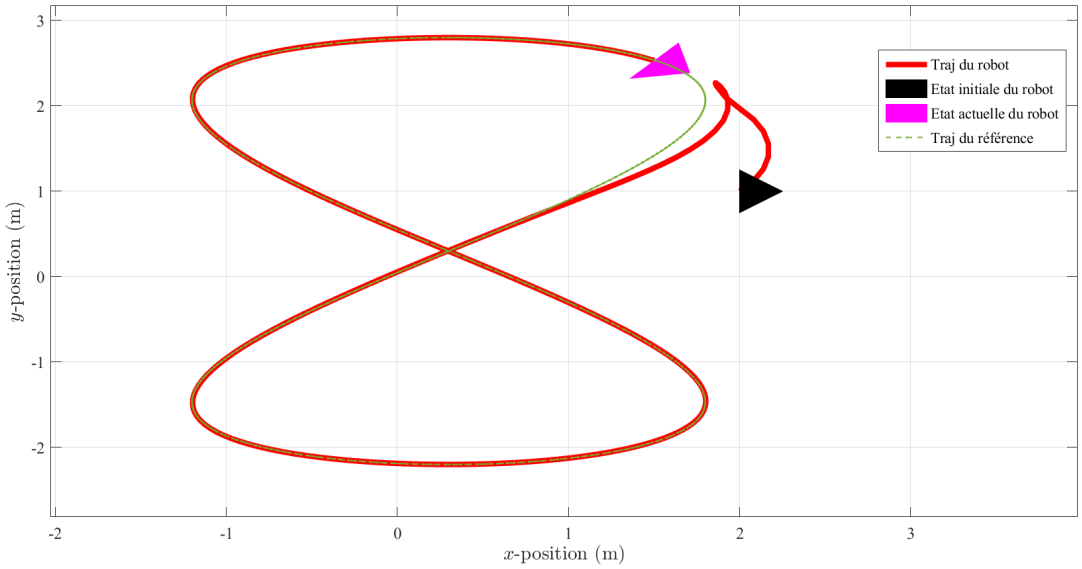

FIGURE 4.8 – Tracé de la trajectoire de forme infini

Le choix du pas de temps de mise à jour du contrôleur de $0,2$ seconde avec un nombre de pas de prédiction $N = 20$, conduit à un horizon de prédiction $T = 4$ secondes.

Les matrices de poids Q_{opt} et R_{opt} de la fonction de coût (4.18) sont choisies

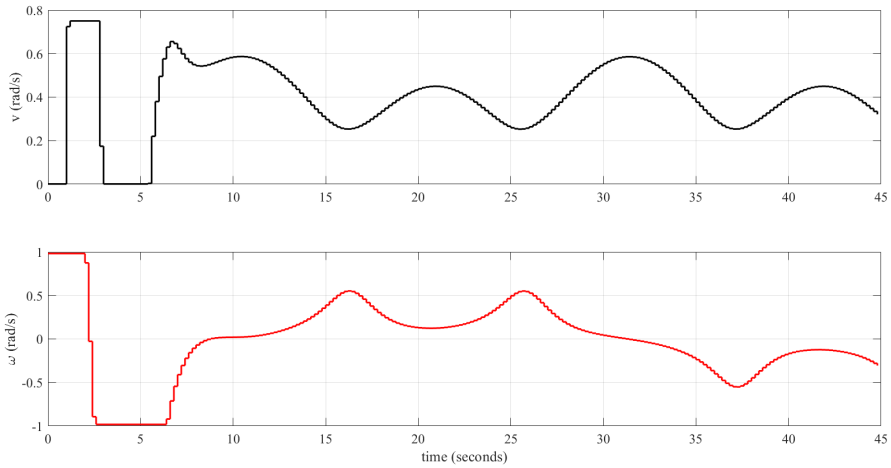

FIGURE 4.9 – Vitesses linéaire et angulaire de la trajectoire circulaire en forme infini

comme matrices diagonales dont éléments diagonaux sont les suivantes : $(15, 10, 0.1)$ pour Q_{opt}, et $(0.002, 0.002)$ pour R_{opt}.

Pour obtenir une localisation précise du robot mobile et satisfaire les limites de saturation de ses actionneurs, les limites de saturation du contrôleur, pour les commandes de vitesse linéaire v et de vitesse angulaire ω, sont fixées comme suit :

$$\begin{bmatrix} -0.75 \\ -\frac{753}{767} \end{bmatrix} \leq \begin{bmatrix} v\,(m/s) \\ w\,(rad/s) \end{bmatrix} \leq \begin{bmatrix} 0.75 \\ \frac{753}{767} \end{bmatrix} \tag{4.27}$$

Les résultats sont résumés dans les Figures (4.10) et (4.11). La Figure (4.10) montre les trajectoires affichées par le robot (en rouge). La position et l'orientation du robot sont illustrées par un triangle rouge, les cercles montrent la prédiction de l'état, l'obstacle est représenté par un disque bleu et la position de référence par un triangle vert.

Comme le montre la Figure (4.10), le controlleur peut parfaitement stabiliser le robot dans la position requise. Les résultats montrent que le contrôleur fonctionne de manière satisfaisante dans ce cas. La Figure (4.11) montre que les vitesses linéaire et angulaire réelles du robot mobile respectent les limites de saturation spécifiée par l'equation (4.27).

4.7.1.4 Résultats du suivi de la trajectoire avec obstacles

Pour le suivi des trajectoires, les performances des contrôleurs NMPC ont été illustrées par la prise en compte de deux trajectoires de référence, à savoir (4.25) et (4.26).

Les paramètres des trajectoires (4.25) et (4.26) sont choisis de manière à ce que les vitesses linéaire et angulaire de référence ne dépassent pas les limites données par (4.27), pour la trajectoire circulaire, le robot part de la pose initiale $q_0 = [0, 0, 0]^T (m, m, rad)$, et pour la trajectoire en forme de huit, la pose initiale est $q_0 = [1, 2, 0]^T (m, m, rad)$.

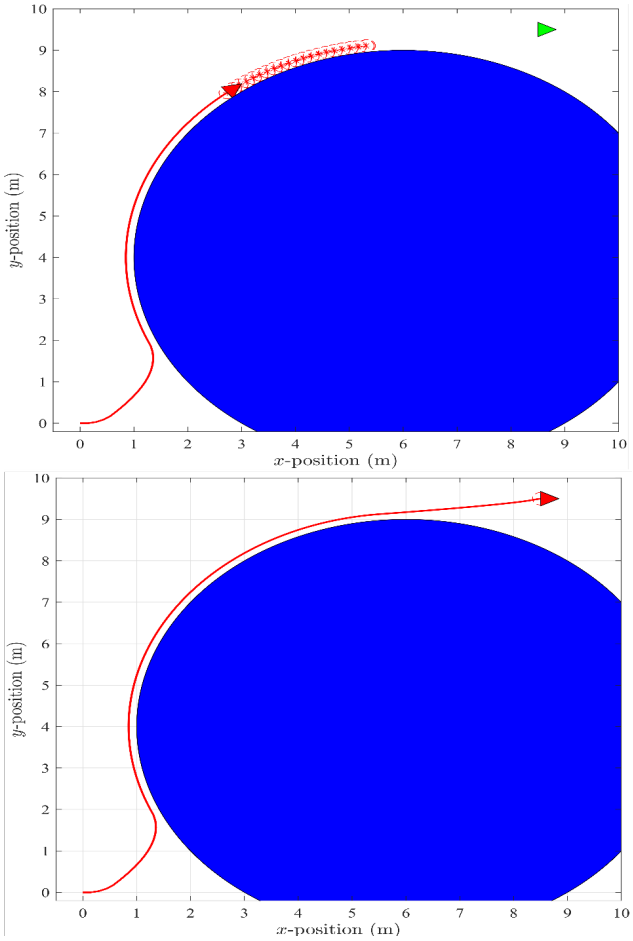

FIGURE 4.10 – Résultats de la trajectoire de stabilisation des points avec obstacle

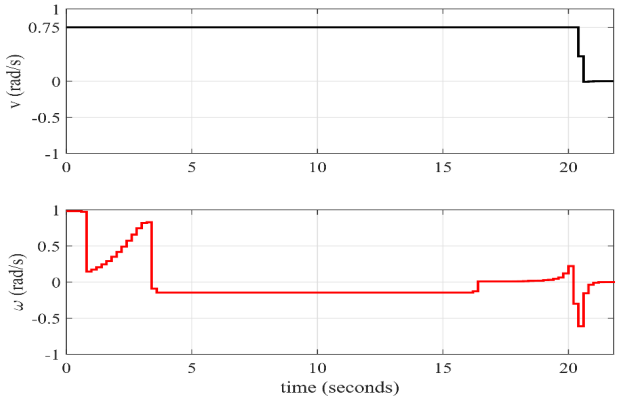

FIGURE 4.11 – Vitesses du robot de stabilisation des points avec obstacle

Le choix du pas de temps de mise à jour du contrôleur de $0,2$ seconde avec un nombre de pas de prédiction $N = 20$, conduit à un horizon de prédiction $T = 4$ secondes. secondes. Les points de départ des trajectoires de référence (4.25) et (4.26) ont été sélectionnés de manière à ce qu'une erreur initiale soit différente de 0.

Pour le suivi de la forme circulaire, les matrices de poids Q_{opt} et R_{opt} de la fonction objectif (4.18) sont des matrices diagonales dont les éléments diagonaux sont donnés comme suit : $(30, 30, 0.2)$ pour Q_{opt}, et $(50, 50)$ pour R_{opt}.

Pour le suivi de la forme huit, les éléments diagonaux de Q_{opt} sont $(30, 30, 0, 2)$, et les éléments diagonaux de R_{opt} sont $(9, 5)$.

Les graphes présentés sur les Figures (4.12) et (4.14) correspondent à un exemple de trajectoire à suivre respectivement en forme circulaire et en forme huit. La trajectoire de référence est indiquée en blue pointillé et les obstacles à éviter sont donnés par des disques bleus.

Les graphes présentés sur les Figures (4.13) et (4.15) correspondent aux profils des signaux de commande représentés par les vitesses linéaire et angulaire, résultant du suivi de la trajectoire de référence.

Ces resultats graphiques permettent de relever les caractéristiques dynamiques relatives aux erreurs.

On constate que la dynamique des erreurs de vecteur d'état q_e est maintenue dans les limites acceptables, ainsi que les vitesses du robot mobile sont maintenues dans les limites indiquées dans (4.27).

Le coût moyen de calcul par pas de temps pour le suivi de forme circulaire est de $(24, 2ms)$, la valeur de l'erreur de position en régime permanent a été observée à l'intérieur $(\pm 3, 5mm)$ et l'erreur d'orientation a été observée à l'intérieur $(\pm 0, 04rad)$, sauf lorsque le robot évite des obstacles.

Dans le cas du suivi du trajectoire de forme huit, les coûts moyens de calcul du temps $(23, 2ms)$ se sont avérés être à l'intérieur de $(\pm 4, 7mm)$ et les erreurs d'orientation $(\pm 0, 065rad)$, sauf lorsque le robot évite des obstacles.

4.7.2 Résultats expérimentaux

Les expérimentations que nous présenterons dans cette deuxième partie visent à valider les deux objectifs de contrôle du robot mobile à entraînement différentiel, à savoir : stabilisation des points et le suivi de trajectoire en utilisant le Robot Qbot2e de Quanser.

Les paramètres d'optimisation en ligne, y compris la fréquence d'échantillonnage, le nombre de pas de l'horizon de prédiction N et les matrices de poids Q_{opt}, R_{opt} et P_{opt} nous ont donné beaucoup de mal à trouver un choix optimal pour eux. Le problème est dûe au fait que nous utilisons ceux qui donnaient de bons résultats en simulation. Mais, en pratique le robot mobile présente un comportement indésirable, en raison que ces paramètres d'optimisation étaient grands, ce qui fait que la fonction de coût est toujours grande.

Après avoir réduit les valeurs diagonales dans les matrices de poids, nous avons commencé à voir un meilleur comportement, qui a conduit, après de nombreuses expériences, à l'excellent comportement qui sera décrit prochainement.

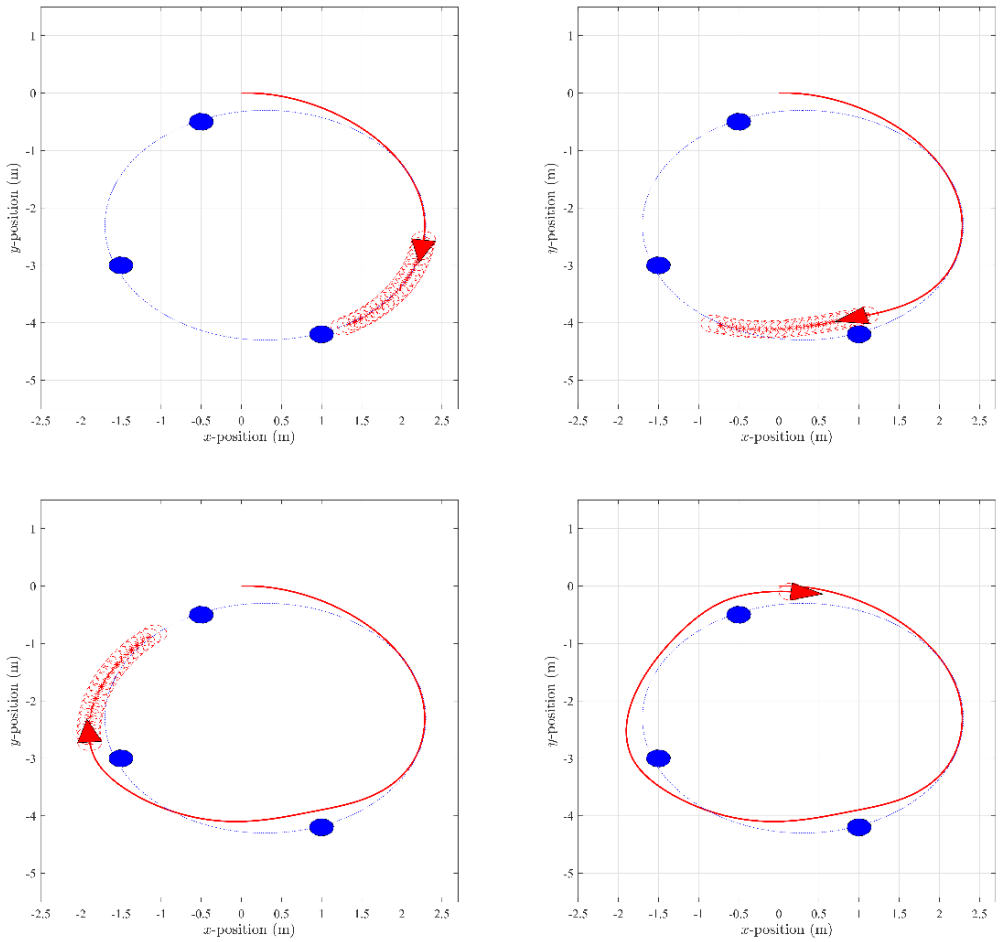

FIGURE 4.12 – Résultats du suivi de trajectoire et de la navigation de forme circulaire

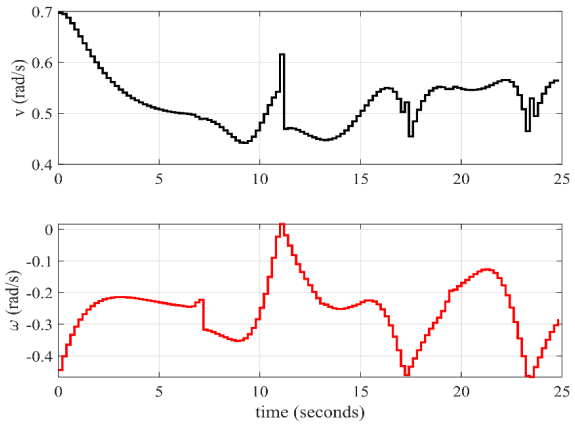

FIGURE 4.13 – Résultats du suivi de la trajectoire de la forme circulaire et des vitesses de navigation

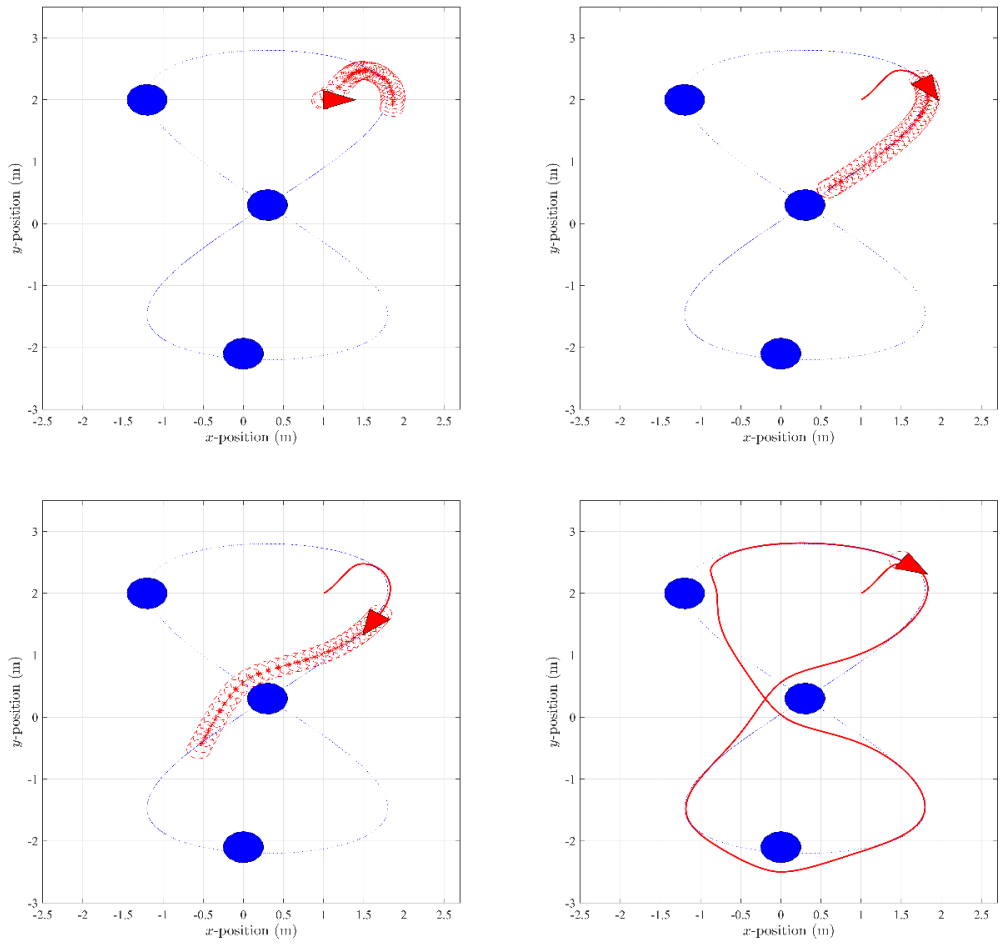

FIGURE 4.14 – Résultats de suivi de trajectoire et de navigation pour une trajectoire de forme 8

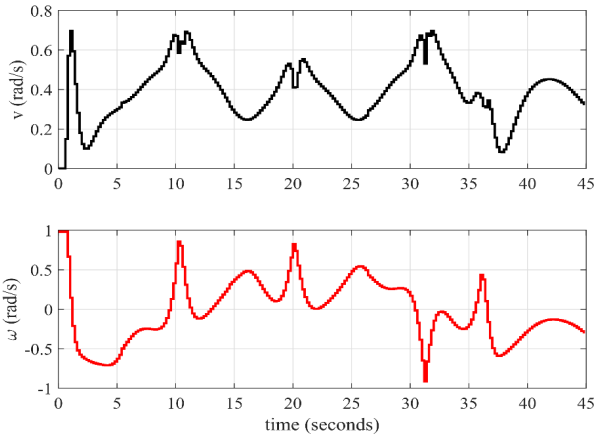

FIGURE 4.15 – Résultats des vitesses du robot pour le suivi de trajectoire et la navigation dans le cas d'une trajectoire de forme 8

Les résultats expérimentaux ont été illustrés en commençant par les résultats de stabilisation du point, puis les résultats de suivi de ligne, et enfin le suivi d'une trajectoire Infini.

Notre but était de généraliser ces résultats à l'évitement d'obstacles, mais malheureusement, les capteurs étaient absents de notre robot et donnaient la plupart du temps des lectures bizarres, ce qui nous a conduit à garder cela comme un travail futur.

4.7.2.1 Stabilisation du point

Dans le but de valider les performances du contrôleur NMPC mis en œuvre pour la stabilisation des points dans la partie précédente, les résultats expérimentaux obtenus du robot Qbot2e exécutant la stabilisation du stationnement, sont présentés.

Dans cet expérimentation, le robot commence par la pose initiale, $q_0 = [0, 0, 0]^T(m, m, rad)$, il est commandé de telle manière à se stabiliser à la pose $q_{ref} = [2, 2, 0]^T$ (un stationnement parallèle).

Le pas de temps de mise à jour du contrôleur est choisi pour être de $0,1$ seconde avec le nombre d'étapes de prédiction $N = 10$, l'horizon de prédiction de temps est $T = 1$ secondes. Le poids des matrices Q_{opt} et R_{opt} de la fonction de coût (4.17) sont choisies comme des matrices diagonales avec des éléments diagonaux sont définis comme suit : $(5.8, 8, 1.5)$ pour Q_{opt}, et $(2.02, 2.02)$ pour R_{opt}.

$$Q_{opt} = \begin{bmatrix} 5.8 & 0 & 0 \\ 0 & 8 & 0 \\ 0 & 0 & 1.5 \end{bmatrix}; R_{opt} = \begin{bmatrix} 2.02 & 0 \\ 0 & 2.02 \end{bmatrix}$$

Afin d'obtenir la localisation exacte du robot et la satisfaction de la saturation de ses actionneurs, les limites de saturation du contrôleur, pour la vitesse linéaire v et la vitesse angulaire ω, sont définis comme suit :

$$\begin{bmatrix} -0.1 \\ -\pi/2 \end{bmatrix} \leq \begin{bmatrix} v\,(m/s) \\ \omega\,(rad/s) \end{bmatrix} \leq \begin{bmatrix} 0.3 \\ \pi/2 \end{bmatrix} \tag{4.28}$$

Les résultats expérimentaux obtenus sont résumés dans les Figures (4.16), (4.17) et (4.18). Sur la Figure (4.16) (meilleure vue en couleurs) le cercle mauve représente le corps du robot, les 2 grands cercles rouges représentent les roues droite et gauche, le petit cercle rouge situé à $[2, 2]$ représente la position finale que le robot est commandé à atteindre. La ligne noir représente la trajectoire effectuée par le robot.

La Figure (4.17) illustre les profils des erreurs en position suivant l'axe des x en rouge, l'axe des y en vert, et en orientation θ en noir.

La Figure (4.18) illustre les profils des signaux de commande qui correspondent aux vitesses linéaire et angulaire. Ces résultats graphiques permettent de relever que le profil de la vitesse linéaire se situe dans la zone limitée par la vitesse maximale V_{max} et la vitesse minimale V_{min}. C'est le même cas pour la vitesse angulaire qui se situe dans la zone limitée par la vitesse angulaire maximale ω_{max} et la vitesse angulaire minimale ω_{min}.

La position de départ du robot est point de départ du figure ou q_0 ;

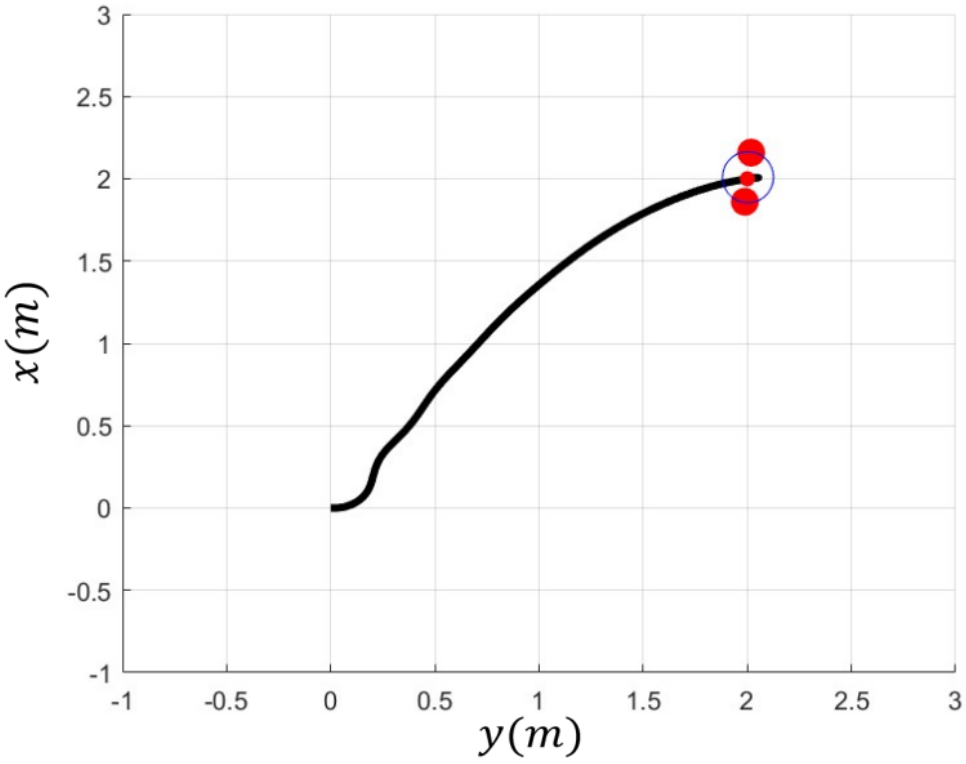

FIGURE 4.16 – Trajectoires exécutées en stabilisation ponctuelle

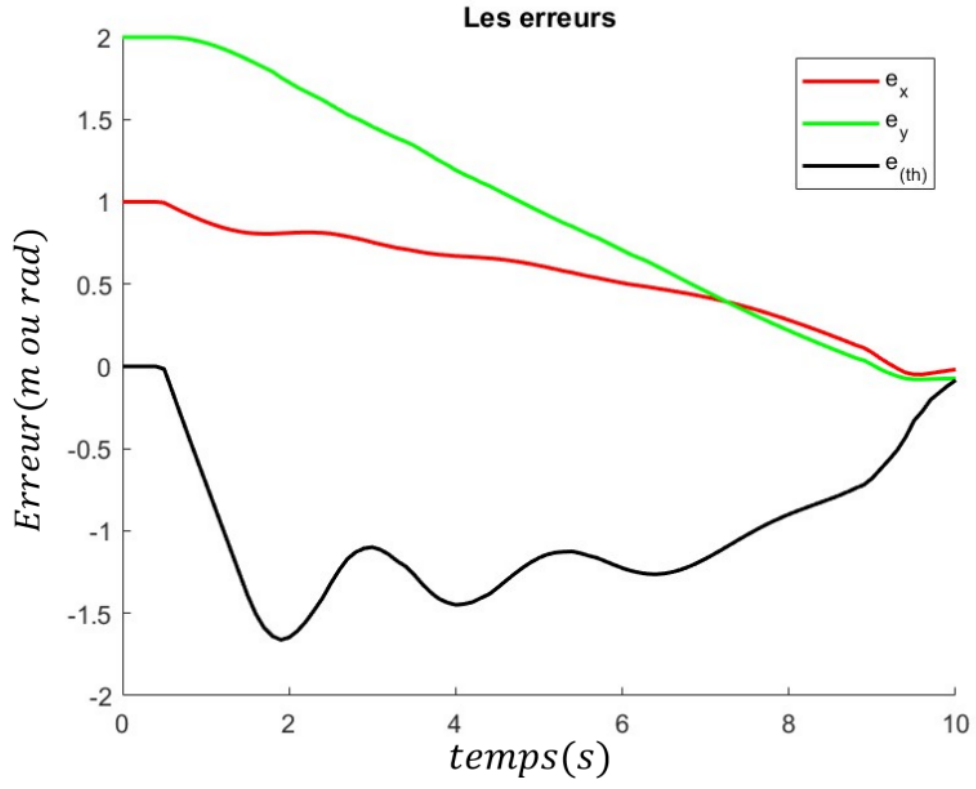

FIGURE 4.17 – Tracé des erreurs pour la stabilisation du point

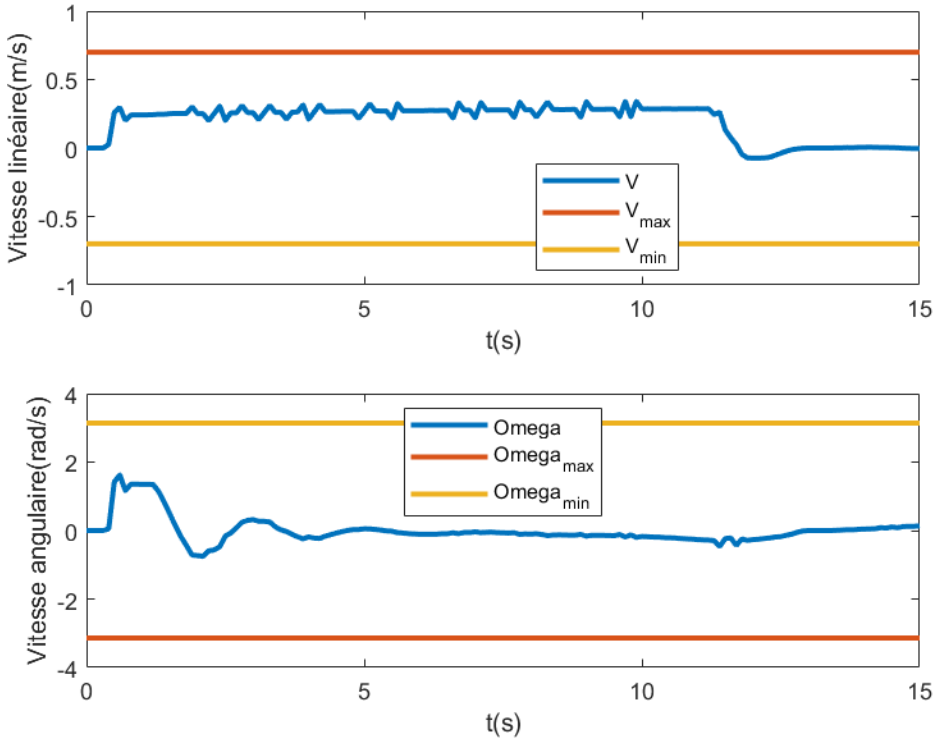

FIGURE 4.18 – Signaux de contrôle de la stabilisation des points

Comme on peut le voir dans la Figure (4.16), le contrôleur peut stabiliser le robot dans la position souhaitée.

Les résultats montrent les performances de poursuite satisfaisantes du contrôleur avec un coût de calcul raisonnable, pour le cas de la stabilisation ponctuelle. La Figure (4.18) (meilleure vue en couleurs) montre que les vitesses linéaires et angulaires du robot satisfont aux limites de saturation données par (4.28)

4.7.2.2 Suivi d'une ligne

Dans cette sous-section, on cherche à suivre une droite d'équation :

$$y_{ref} = 1 \tag{4.29}$$

ce qui revient par exemple à s'engager sur une route principale après avoir été sur une route secondaire.

Le robot mobile doit partir de la position $q_0 = [0, 0, 0]^T$, et doit rattraper la route avec l'équation (4.29) le plus rapidement possible, et rester sur la ligne pendant les 9 mètres.

Les résultats expérimentaux montrent de bonnes performances de poursuite, comme illustré par la Figure (4.19).

Le pas de temps de mise à jour du contrôleur est choisi pour être de $0,1$ seconde avec le nombre d'étapes de prédiction $N = 10$, l'horizon de prédiction de temps est $T = 1$ secondes. Le poids des matrices Q_{opt} et R_{opt} de la fonction de coût (4.17) sont

choisies comme des matrices diagonales dont des éléments diagonaux sont données par $(5.8, 8, 1.5)$ pour Q_{opt}, et par $(2.02, 2.02)$ pour R_{opt}.

$$Q_{opt} = \begin{bmatrix} 5.8 & 0 & 0 \\ 0 & 8 & 0 \\ 0 & 0 & 1.5 \end{bmatrix} ; R_{opt} = \begin{bmatrix} 2.02 & 0 \\ 0 & 2.02 \end{bmatrix}$$

Afin d'obtenir la localisation exacte du robot et la satisfaction de la saturation de ses actionneurs, les limites de saturation du contrôleur, pour la vitesse linéaire v et la vitesse angulaire ω, sont définis par (4.28).

Les résultats obtenus sont résumés dans les Figures (4.19)- (4.21). Sur la Figure (4.19) (meilleure vue en couleurs) le cercle mauve représente le corps du robot, les 2 cercles rouges représentent les roues droite et gauche, les cercles verte représente la trajectoire de référence que le robot est commandé à suivre. La ligne noir représente la trajectoire effectuée par le robot.

La Figure (4.20) illustre les erreurs suivant l'axe des x en rouge, l'axe des y en vert, et d'orientation θ en mauve.

La Figure (4.21) représente les signaux de commande pour la vitesse linéaire en haut et la vitesse angulaire en bas. Les vitesses sont représentées en bleu, V_{max} et V_{min} respectivement en rouge et orange, ω_{max} et ω_{min} respectivement en orange et rouge.

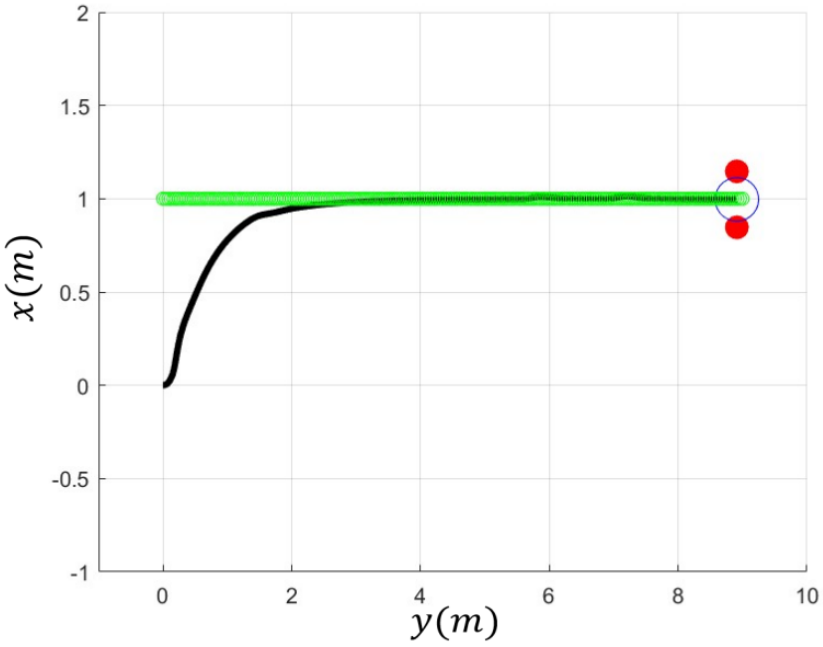

FIGURE 4.19 – Trajectoires exécutées en suivi d'une ligne

Comme on peut le constater sur la Figure (4.19), le Qbot2e rattrape la trajectoire très rapidement, et le suivi était acceptable. Pour tester la robustesse de la stratégie de commande adoptée, nous avons injecté 2 défauts dans θ à des temps respectifs de $29sec$ et $36sec$, ce qui a entraîné le robot à perdre la trajectoire pendant un petit

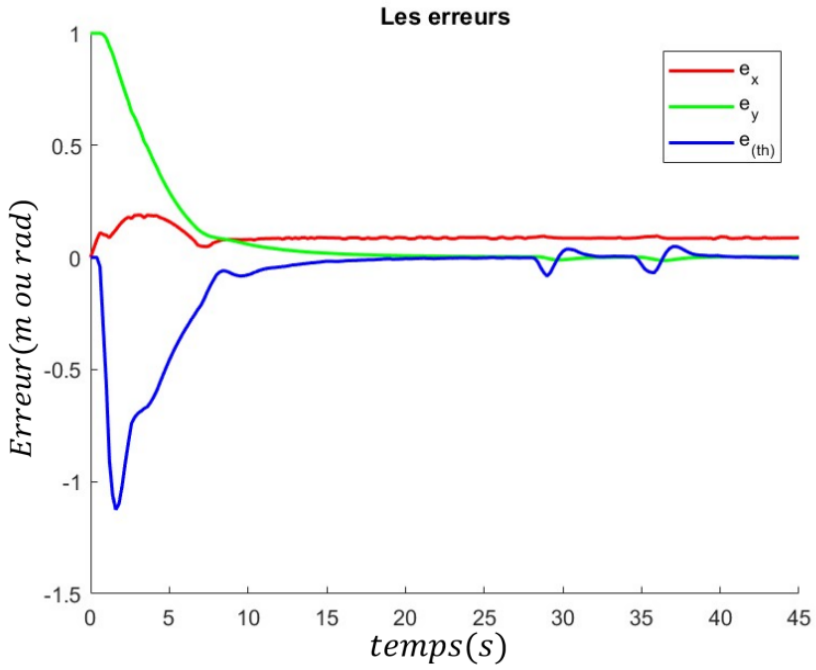

FIGURE 4.20 – Tracé des erreurs pour la stabilisation du point

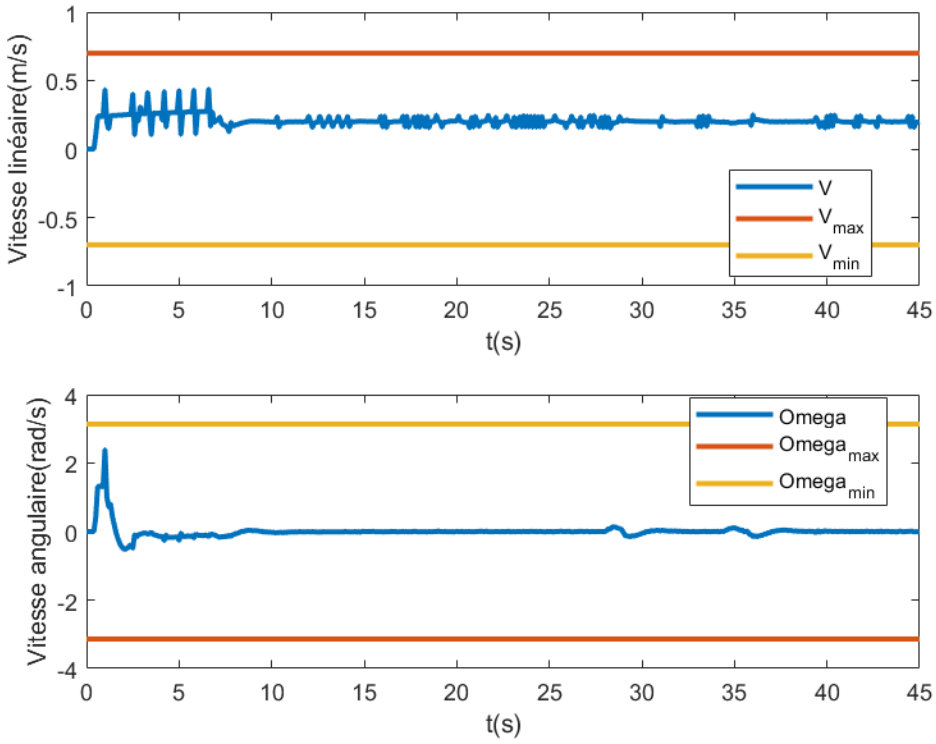

FIGURE 4.21 – Signaux de contrôle de la stabilisation des points

moment, mais le contrôleur rattrape le défaut tout aussi rapidement, et garde les erreurs très petites et convergeant vers zéro.

Les résultats montrent les performances de poursuite satisfaisantes du contrôleur avec un coût de calcul raisonnable et une tolérance aux erreurs de navigation. La Figure (4.18) (meilleure vue en couleurs) montre que les vitesses linéaires et angulaires du robot satisfont aux limites de saturation données par (4.28).

4.7.2.3 Suivi de la trajectoire d'une forme à 8

Les performances du contrôleur NMPC, pour le suivi des trajectoires, a été évaluée en considérant deux trajectoires de référence, à savoir les trajectoires de forme octogonale (4.30).

$$\begin{cases} x_{ref} = 0.3 + 3.5sin(0.1t) \\ y_{ref} = 0.3 + 3.5cos(0.05t) \end{cases} \tag{4.30}$$

Les paramètres de la trajectoire (4.30) est choisis de telle sorte que les vitesses linéaire et angulaire de référence ne violent pas (4.24).

Le pas de temps de mise à jour du contrôleur est choisi pour être de $0,1$ seconde avec un nombre de pas de prédiction $N = 10$, conduisant à un horizon de prédiction $T = 1$ secondes. Dans cette expérimentation, le robot Qbot2e part de $q_0 = [0, 3, 0]^T$ et est commandé pour suivre la trajectoire définie par le système d'équations (4.30).

Les matrices de poids Q_{opt} et R_{opt} de la fonction de coût (4.18) sont choisies comme matrices diagonales avec des éléments diagonaux définis comme $(6.5, 4, 2)$ pour Q_{opt}, et comme $(2, 2)$ pour R_{opt}.

$$Q_{opt} = \begin{bmatrix} 6.5 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 2 \end{bmatrix} ; R_{opt} = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$$

Les Figures (4.22)-(4.24) montrent la trajectoire du robot, les erreurs de navigations, et les signaux de commande linéaires et angulaires respectivement.

La Figure (4.22) illustre la réponse du Qbot2e par rapport a une trajectoire Lemniscate (de forme 8). Le robot est marqué d'un cercle bleu, et ses roues sont marquées de deux cercles rouges, les cercles verts représentent la trajectoire de référence et la ligne noire représente la trajectoire du robot.

Sur la Figure (4.23), on a observé que la valeur de l'erreur de position en régime permanent était comprise entre $(\pm 2cm)$ pour x et y, et l'erreur d'orientation entre $(\pm 1rad)$, le coût moyen de calcul par pas de temps étant de $(25millisecondes)$.

La Figure (4.24) montre que les vitesses du Qbot2e convergent vers leurs valeurs de référence, et ne dépasse jamais leurs valeurs limites fixées par les contraintes de vitesses, malgré l'erreur initiale qu'elles présentaient.

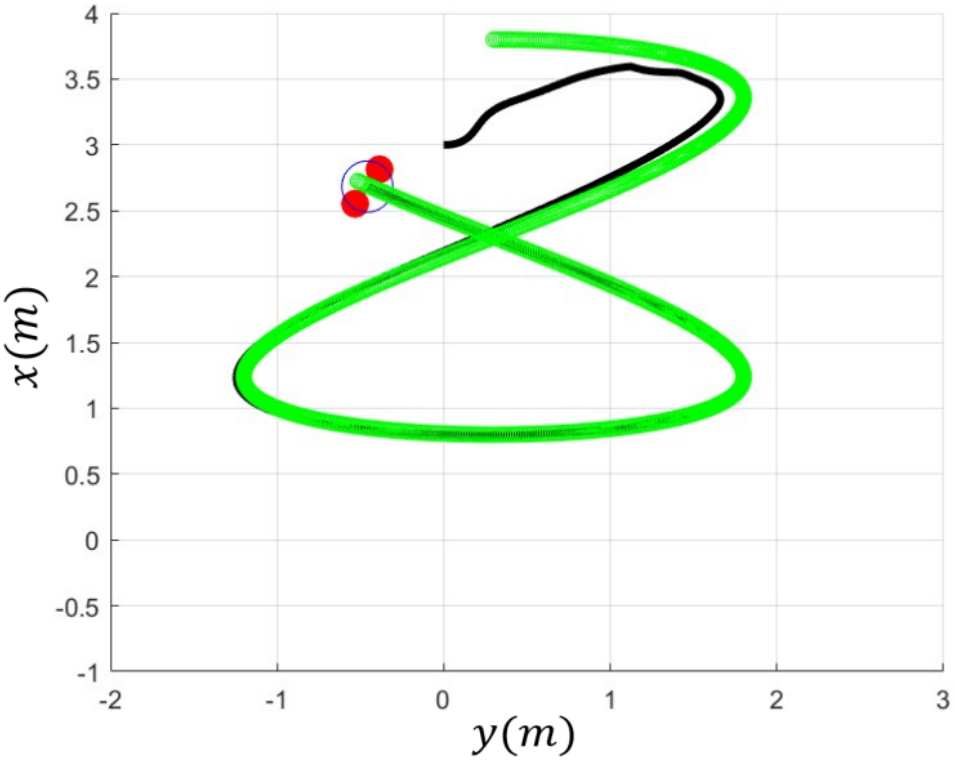

FIGURE 4.22 – Trajectoires exécutées en suivi d'une trajectoire Lemniscate

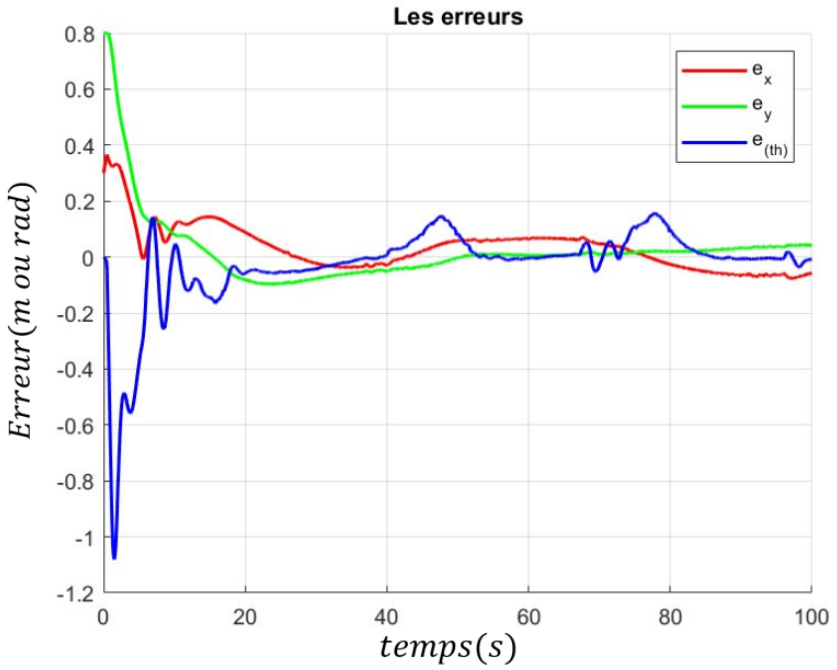

FIGURE 4.23 – Tracé des erreurs pour le suivi du trajectoire Lemniscate

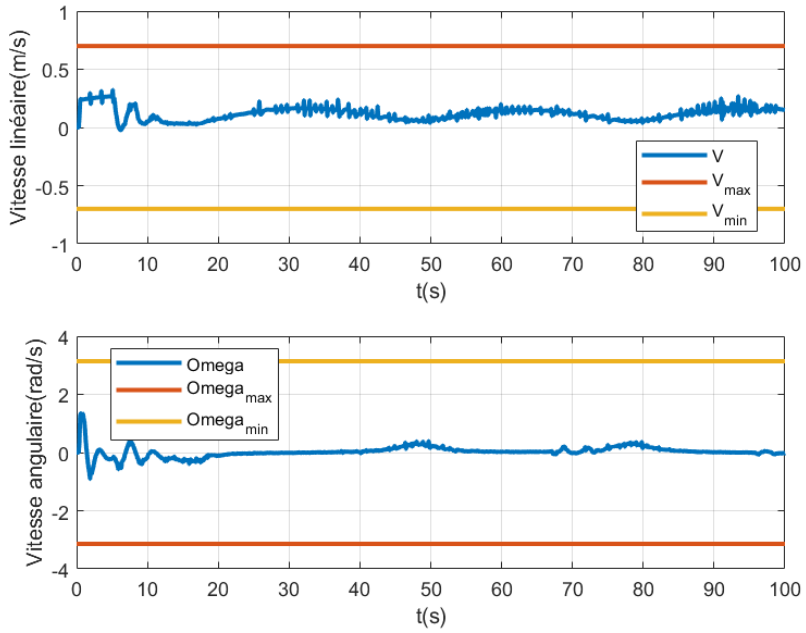

FIGURE 4.24 – Signaux de contrôle de la suivi du trajectoire Lemniscate

4.8 Commande prédictive MPC stabilisante en considérant la cinématique et la dynamique

Cette section traite le suivi de la trajectoire du robot à entraînement différentiel à l'aide d'un modèle mathématique régissant la dynamique (2.78) et la cinématique (2.23) . Ces deux modèles sont considérés avec la dynamique de l'actionneur qui est le moteur à courant continu pour dériver un modèle dynamique linéaire d'espace d'état. Les équations cinématiques non linéaires de base sont linéarisées en un modèle d'espace d'état successivement linéarisé. Les modèles dynamique et cinématique sont augmentés pour dériver un seul modèle linéaire d'espace d'état. Le suivi de la référence est réalisé par une commande prédictive par modèle de la tension d'alimentation des deux moteurs d'entraînement en tenant compte des contraintes sur les variables contrôlées et les variables manipulées. Des résultats de simulation sont fournis pour démontrer les performances de poursuite de la stratégie de contrôle proposée dans l'environnement de simulation MATLAB. Pour mieux comprendre comment ce modèle est dérivé, veuillez vous référer à [120].

Le modèle augmenté est un modèle à espace d'état variant dans le temps avec 9 états (courants i_L et i_R, les couples τ_R, τ_L, vitesses linéaires et angulaires v, ω et coordonnées x, y, θ), deux variables de commande (entrée de commande de la tension du moteur U_L, U_R) et trois sorties (position dans les directions x et y et orientation θ mesurée à partir de la direction x), il est illustré ci-dessous :

$$\begin{aligned} \overline{q}(\kappa+1) &= \overline{A}(\kappa)\overline{q}(\kappa) + \overline{B}(\kappa)\overline{u}(\kappa) \\ \overline{y}(\kappa) &= \overline{C}(\kappa)\overline{q}(\kappa) \end{aligned} \tag{4.31}$$

Le modèle augmenté est un modèle basé sur les erreurs dont les variables d'état

sont des déviations par rapport aux variables de référence. Les variables de référence peuvent être vues comme un robot idéal suivant une trajectoire de référence variable dans le temps. Ces vitesses de référence v_{ref}, ω_{ref} et cet angle d'orientation θ_{ref} peuvent être calculés à partir des équations (4.32),(4.33) et (4.34) à partir des entrées de référence (coordonnées de position du robot x_{ref}, y_{ref}).

$$v_{ref} = \sqrt{\dot{x}_{ref}(t)^2 + \dot{y}_{ref}(t)^2} \tag{4.32}$$

$$\theta_{ref}(t) = arctan2(\dot{y}_{ref}(t), \dot{x}_{ref}(t)) \tag{4.33}$$

$$\omega_{ref}(t) = \dot{\theta}_{ref}(t) = \frac{\dot{x}_{ref}(t)\ddot{y}_{ref}(t) - \dot{y}_{ref}(t)\ddot{x}_{ref}(t)}{v_{ref}(t)} \tag{4.34}$$

Le suivi de la trajectoire du robot mobile est réalisé par une commande prédictive de modèle appliquée au modèle linéaire variant dans le temps, équation (4.31), avec une fonction de coût comme dans l'équation (4.18) en tenant compte des contraintes, équation (4.20). À chaque instant, l'algorithme MPC calculera les entrées de commande optimales (entrées de commande de la tension du moteur u_L et u_R). Le schéma de commande global est illustré à la Figure (4.25).

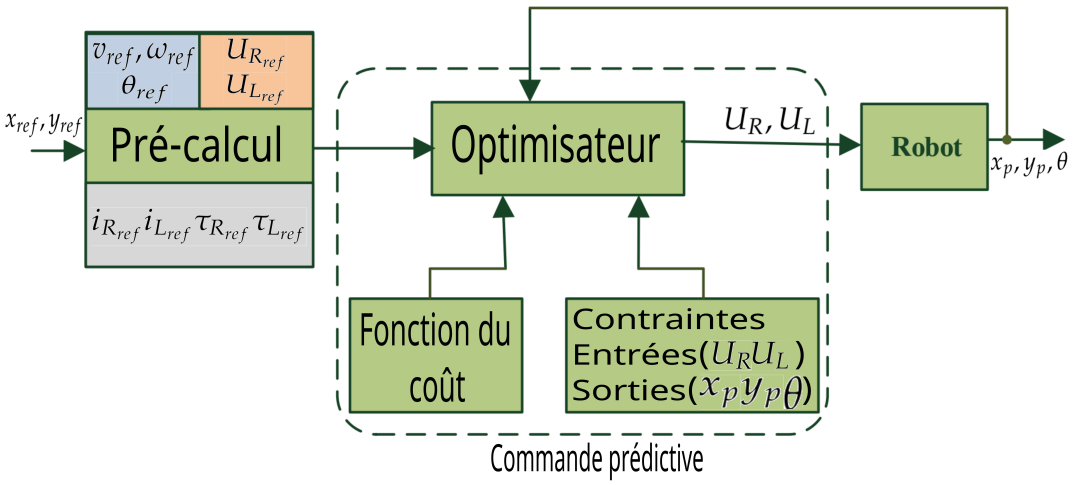

FIGURE 4.25 – Schéma de contrôle global

4.9 Résultats de la Simulation

Les paramètres du châssis sont donnés au tableau (2.1). Les paramètres nécessaires pour les moteurs à courant continu avec description de la source de tension commune sont indiqués dans le tableau (4.2). Nous considérons des moteurs identiques avec des paramètres identiques.

Pour le contrôleur MPC on a choisi :

$$Q_{opt} = diag(1, 10, 10)$$

$$R_{opt} = diag(10, 10)$$

TABLE 4.2 – Régime permanent pour les tensions des moteurs

Notation	Valeur	Dimension	Signification
R_{mot}	2	Ω	résistivité de l'enroulement du moteur
L_{mot}	0.05	H	L'inductance du moteur
D	0.1	m	distance entre le centre des roues
K_{mot}	0.1	$kg.m^2.s^{-2}.A^{-1}$	La constante électromotrice
R_z	0.2	Ω	Résistance de la source
U_0	10	V	Tension de la source
J_{mot}	0.025	$kg.m^2$	moment d'inertie total du rotor et du réducteur
kr_{mot}	0.005	$kg.m^2.s^{-1}$	coefficient de la résistance à la rotation du rotor et de la boîte de vitesses
P_G	25	$---$	rapport de transmission de la boîte de vitesses

Ces valeurs sont choisies de manière à correspondre approximativement aux valeurs physiques réelles du robot mobile.

La trajectoire utilisée ici est choisie d'une manière proche de la réalité, où l'on fixe des points de contrôle et on les relie par des lignes ou des splines, ou par interpolation, puis on extrait x_{ref}, y_{ref} de la trajectoire résultante, ensuite, on calcule θ_{ref} et on demande au robot de suivre cette trajectoire.

Les résultats de simulation présentés ont pour but de montrer la robustesse du robot en présence de charges de masse sur le robot et de problèmes de basses tensions alimentant les actionneurs, nous avons choisi ces deux paramètres car ils sont les plus responsables de l'instabilité des autres contrôleurs.

Dans les Figures (4.26)-(4.28), la trajectoire simulée est comparée à la trajectoire souhaitée (trajectoire de référence) dans le graphique du haut, sous celui-ci, les vitesses linéaire et angulaire sont comparées à leurs vitesses de référence, respectivement de gauche à droite, et dans le graphique du bas, les entrées de commande et la référence sont comparées aux entrées souhaitées à gauche, et les courants des moteurs sont comparés aux courants de référence à droite.

Des contraintes ont été appliquées aux variables contrôlées (tensions de commande des roues droite et gauche) et aux vitesses (linéaire et angulaire). Les contraintes de la tension de commande des moteurs ont été fixées à $[0, 10]$ puisque la tension de la source est de $10V$ et qu'aucun mouvement arrière du moteur n'a été supposé. Les vitesses ont été fixées à $[0, 0.2]$ pour v et $[-45, 45]$ pour ω. La trajectoire a été choisie de manière à ce que nous puissions voir la réponse du robot lors d'un changement soudain de position et d'orientation du robot.

Les Figures (4.26)-(4.28), que le contrôleur fait un excellent travail pour différentes charges sur le robot, lorsque la masse est de $3kg$ le robot utilise des tensions minimales (inférieures ou égales à $2V$ pour toute la simulation), lorsque la masse est de $10kg$ le robot utilise des tensions plus élevées (inférieures ou égales à $5V$ pour toute la simulation), enfin, lorsque la masse est de $35kg$, le robot utilise des tensions maximales (inférieures ou égales à $10V$ pour l'ensemble de la simulation), nous avons rencontré quelques problèmes lorsque la masse est supérieure à $35kg$,

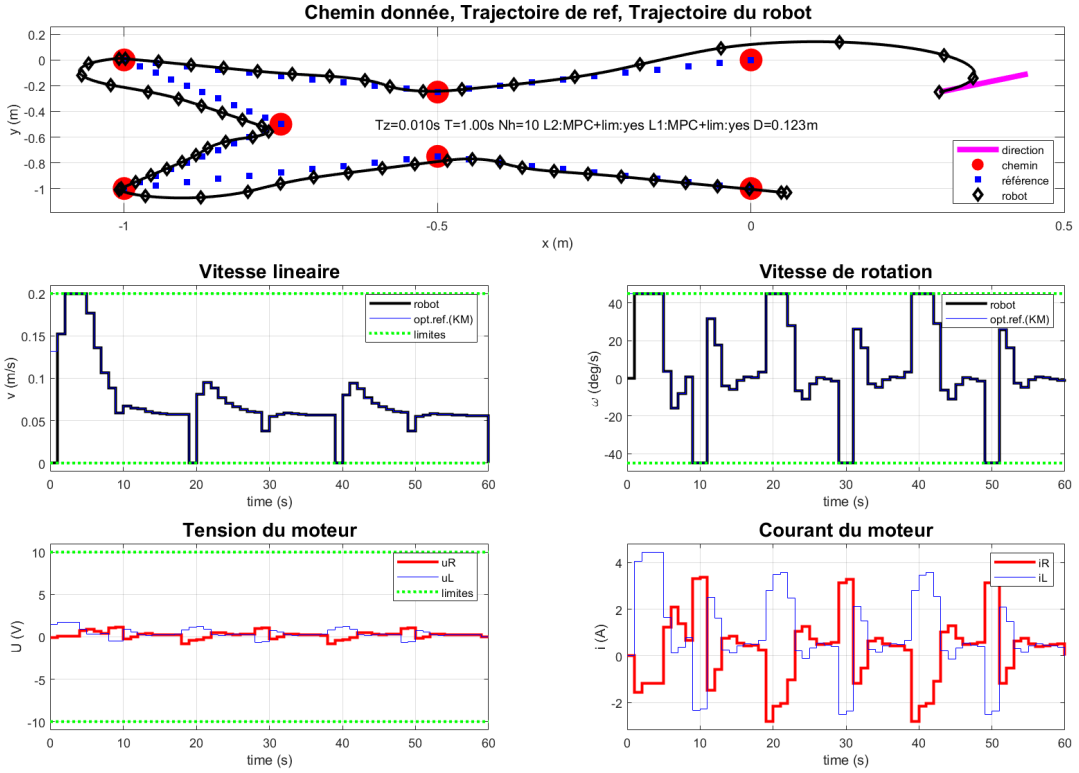

FIGURE 4.26 – Résultats pour $U_0 = 10V, M_c = 3kg$

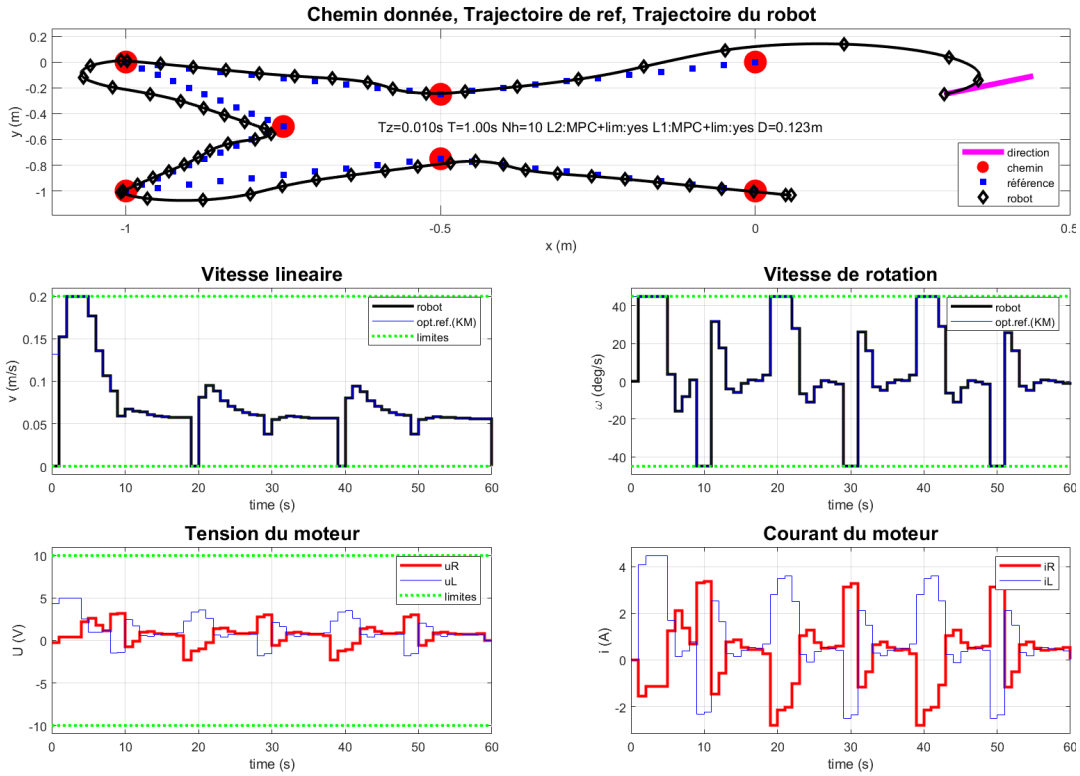

FIGURE 4.27 – Résultats pour $U_0 = 10V, M_c = 10kg$

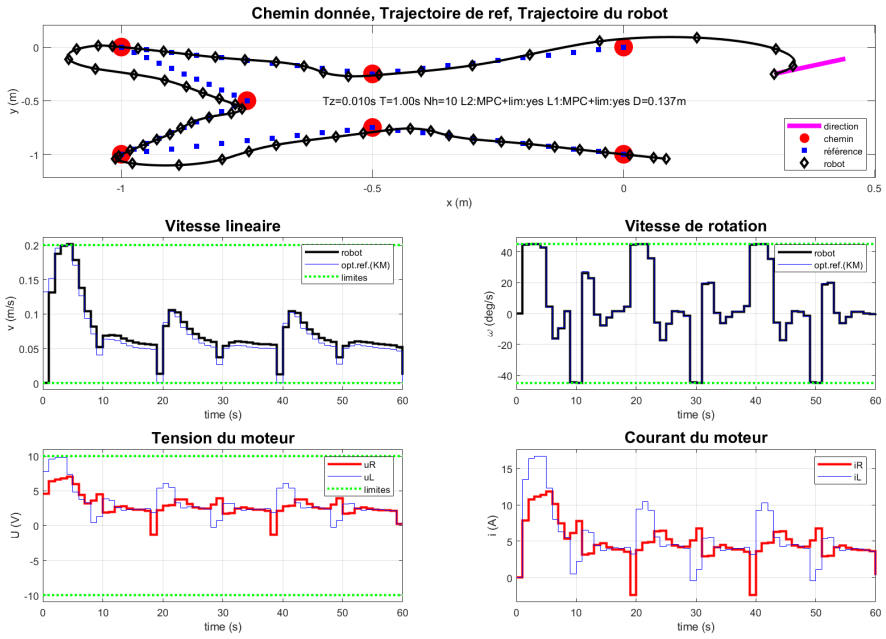

FIGURE 4.28 – Résultats pour $U_0 = 10V, M_c = 35kg$

ce qui est un très bon travail pour un moteur de $10V$, les contraintes limites sur les entrées et les vitesses ont été satisfaites et le suivi était assez acceptable.

4.10 Conclusion

Compte tenu des performances limitées des commandes cinématiques d'un robot mobile, nous avons implémenté une commande prédictive non linéaire pour la navigation d'un robot mobile (NMPC).Cette statégie de commande est conditionnée par la stabilité des contraintes des états terminaux.

De plus, les trajectoires utilisées sur le robot mobile doivent être suffisamment riches en termes de généralité, allant de tâches simples au suivi de trajectoires complexes. Les simulations ont été réalisées sur la plateforme Matlab/Simulink et les expérimentations sur la plateforme du robot mobile Qbot2e.

Dans le cas de la stabilisation ponctuelle, le robot a été commandé pour effectuer un stationnement en avant et en parallèle avec différentes orientations alors que dans le problème du suivi de la trajectoire, le robot a été commandé pour suivre une forme circulaire et octogonale.

Les résultats de simulation sur MATLAB et les expérimentations sur le vrai robot *Qbot2e* obtenus nous ont permis de constater :

— L'amélioration des performances dynamiques du système commandé et notamment en termes de stabilisation des points et un meilleur suivi de trajectoire en régime permanent avec de faibles erreurs de poursuite.

— La commande prédictive est parfaite pour les mises en œuvre faciles en expérimental, grâce à la gestion des contraintes et à la nature prédictive qui

contribue à la stabilité. Tout cela n'est possible que grâce à la puissance de calcul des ordinateurs actuels.

— La prise en charge des contraintes d'évitement d'obstacles et de la variabilité de paramètres du robot.

— Dans la dernière section, un modèle linéaire variant dans le temps est utilisé en considérant à la fois la cinématique et la dynamique du robot mobile, ce qui permettra de suivre la trajectoire du robot mobile en contrôlant la tension de commande des moteurs.

— Des contraintes ont été considérées pour la variable de contrôle et satisfaites, nous avons montré que le contrôleur fait un bon suivi dans des masses allant de $3kg$ à $35kg$.

Conclusions et Perspectives

Le robot Qbot2e peut être contrôlé selon plusieurs stratégies qui correspondent à différentes méthodes de génération de couples dynamiques, de vitesses linéaires et angulaires.

Dans le cadre de cette thèse, nous nous sommes intéressés d'une part, à évaluer l'intérêt des commandes non linéaires, que ce soit pour la stabilisation point à point ou le suivi de trajectoire, et d'autre part, à examiner la contribution des stratégies de commande neuro-floues et optimales à ces problèmes et ce, en mettant considérablement l'accent au résultats expérimentaux.

Dans cette section, nous conclurons la thèse en résumant les principaux résultats, puis, nous présenterons quelques perspectives de recherches futures pour compléter et améliorer ce travail.

Cette thèse contribue à la conception de systèmes de contrôle avancé pour la navigation d'un robot mobile à roues. Les résultats expérimentaux ont confirmé l'efficacité et la fiabilité de chaque méthode que nous avons suggérées. L'application de ces méthodes peut réduire de manière significative la tâche laborieuse de préprogrammation manuelle d'un robot et permettre à l'homme d'aller au-delà de ses capacités physiques.

Dans cette thèse, nous nous sommes penchés sur l'état de l'art concernant le sujet de recherche. Le premier chapitre a fourni une vue d'ensemble des concepts pertinents et des avancements dans la robotique mobile, en particulièrement les robots mobiles à entraînement différentiel. Nous avons exploré la littérature existante sur les DDMR, les contraintes de mouvement, la modélisation cinématique et dynamique. En outre, nous avons discuté des stratégies d'évitement des obstacles et du matériel utilisé dans notre étude. En examinant de manière approfondie l'état actuel du domaine, nous avons jeté des bases solides pour les chapitres suivants, ce qui nous a permis d'aborder les domaines problématiques identifiés. Cette exploration de l'état de l'art a non seulement amélioré notre compréhension du sujet, mais a également apporté des informations précieuses au domaine plus large de la robotique mobile.

Dans le deuxième chapitre, nous avons largement abordé la conception des modèles cinématiques et dynamique qui traduisent le comportement du robot mobile à entrainement différentiel Qbot2e. Des lois « géométriques » et « Euler Lagrange » sont successivement utilisées pour les modélisation cinématique et dynamique. De même, nous avons fait appel à des manipulations mathématiques utilisant la théorie des vecteurs, pour nous débarrasser de la matrice des contraintes et enfin, obtenir un modèle approprié à des fins de contrôle.

Par la suite, nous avons étudié explicitement, le contrôle autonome pour suivre une trajectoire prédéfinie. Pour ce faire, nous avons opté pour deux techniques dont la première est une commande par retour d'état asymptotiquement stable au sens de

Lyapunov pour stabiliser la posture du robot. En outre, nous avons ajouté un contrôleur dynamique qui contrôle les vitesses de notre robot et qui est basé sur le contrôle proportionnel et la dynamique inverse.Les résultats expérimentaux ont montré que nous n'étions pas satisfaits des performances de notre contrôleur. Nous avons donc remplacé notre contrôleur cinématique par un contrôleur intégralement inspirée d'un contrôleur adaptatif (ANFIS), qui fait le même travail, mais en mieuxAu niveau de contrôle autonome, nous avons créé une stratégie de contrôle basée sur le modèle pour synthétiser la dynamique d'erreur entre la posture de référence et la position actuelle du robot mobile afin, d'effectuer le suivi de trajectoire. La stabilité asymptotique de ce schéma de contrôle est ensuite assurée au sens de Lyapunov.

Ensuite, pour contrôler le modèle dynamique, nous avons conçu un contrôleur de modèle inverse avec un régulateur proportionnel. Par la suite, nous avons réalisé des simulations comparatives sur le contrôleur, en prenant en compte le modèle dynamique sans en considération le mode dynamique. L'objectif principale est de démontrer l'impact de la modélisation dynamique dans la commande des robots mobiles.

Dans notre cas, le contrôleur suggéré est très sensible aux paramètres de commande dans l'expérimental. Le schéma de commande est donc inadéquat pour l'expérimental. C'est la raison qui nous a conduit à définir une nouvelle méthodologie plus appropriée qui est une principale contribution de notre travail. Elle est basée sur l'utilisation de la logique floue en combinaison avec des réseaux de neurones comme outils d'inférences et d'optimisation. Cela, nous a permis de vérifier nos résultats de simulation précédents à l'aide du contrôleur ANFIS. C'est ce qui a été développé dans le chapitre III.

Dans le chapitre IV, nous avons mis l'accent pour une solution basée sur l'optimisation et ce, pour le contrôle du Qbot2e.

Nous nous sommes intéressés au contrôle optimal NMPC qui a l'avantage de prise en compte explicite des contraintes, ce qui nous a amené à la conception d'un contrôleur NMPC stable avec une contrainte terminale. Le but est de réduire à zéro les états d'erreurs de suivi déduits du modèle cinématique pour assurer la stabilité, tout en adressant deux problèmes de contrôle pour éviter les obstacles et surtout, respecter les contraintes actionneurs et non holonomiques. Après expérimentation, Nous avons constaté que la cinématique de ce contrôleur est suffisante, il n'a même pas besoin de contrôle dynamique.

En utilisant le même contrôleur NMPC dans le contexte précédent, et en ajustant les paramètres de contrôle, nous avons réussi à améliorer les performances de poursuite lors de l'expérimentation sur la plateforme Qbot2e.

Par ailleurs, on a implémenté un contrôleur MPC stable pour piloter le système des deux modèles cinématique et dynamique du robot mobile. Ce contrôleur à neuf états assure le suivi des trajectoires complexes afin de quantifier la robustesse du MPC aux virages difficiles et aux changements de charges sur le robot.

En ce qui concerne les notes clés apprises au cours de l'élaboration de ce projet, nous avons constaté que :

— L'ajout du modèle dynamique dans la première section a démontré la nécessité de son intégration dans l'architecture de contrôle ainsi que, la robustesse qui doit être recherchée ;

— Le contrôleur du modèle dynamique doit être traité avec précaution, surtout lors de l'expérimentation en temps réel. C'est pourquoi, nous avons rajouté par la suite, dans la troisième partie, un régulateur ANFIS pour l'étude expérimentale ;
— L'implémentation de notre projet dans la quatrième partie, autour de l'architecture de contrôle prédictif, nous a permis de surmonter les deux problèmes de contrôle par des simulations et expérimentations sous environnement Matlab. Ceci, nous a permis d'éviter les obstacles et d'augmenter l'autonomie du robot, en lui donnant la liberté de choisir ses propres trajectoires. Par conséquent, si le robot se trouve dans une situation inattendue, comme un obstacle devant lui, il peut effectuer des manœuvres adéquate sans avoir besoin d'une programmation spéciale. Cette capacité à adapter son comportement à la situation indique une prise de décision autonome. La connaissance à tirer de cette partie est que ce contrôleur est naturel pour l'expérimentation, à condition que les paramètres de contrôle soient maintenus bas et dans de bonnes limites. Après cela, les mêmes paramètres pourraient même être généralisés à tous les types de contrôle sur la même plateforme. Pour obtenir un bon résultat, nous avons continué à ajuster les paramètres de contrôle de régulation du NMPC. Les mêmes paramètres nous ont donné aussi de résultats concluants dans tous les autres problèmes que nous avons essayés (suivi de ligne, suivi de trajectoire Lemniscates etc.).

Une vidéo complète sur YouTube [119], montrant les performances du robot Qbot2e pour les résultats expérimentaux, a été uploadée.

En termes de perspectives, nous envisageons d'améliorer notre méthode en l'intégrant de manière plus efficace dans un système entièrement autonome. Notre objectif est d'augmenter encore la fiabilité et l'adaptabilité de notre navigateur à différents types de robots et d'environnements et ce, en développant une bibliothèque de familles de trajectoires pour adapter au mieux le comportement du robot à une situation spécifique. Nous pouvons citer le type de mission (suivi d'un parcours, exploration, phase de stationnement, etc.) et d'environnement (jonché ou non, surface glissante, etc.). De plus, nous visons à améliorer l'algorithme d'optimisation en développant une approche qui combine les avantages des méthodes déterministes (performance de convergence en l'absence d'obstacles) avec les avantages des méthodes stochastiques (gestion des minima locaux causés par les obstacles).

En outre, pour atteindre une autonomie complète, le développement d'un planificateur de chemin est essentiel. L'idée est de confier au planificateur de trajectoire la tâche de déterminer une série de points pour le robot mobile, sur la base d'une perception générale de l'environnement dans lequel évolue le robot. L'intégration de ces modules dans le robot nous permettra donc de vérifier expérimentalement notre approche.

Enfin, en termes de contrôle, la fusion de capteurs est un excellent outil à ajouter pour améliorer les performances des robots mobiles car des défauts et des erreurs peuvent se produire, et cet outil les compense.

Bibliographie

[1] Ulrich NEHMZOW. *Mobile robotics : a practical introduction*. Springer Science et Business Media, 2012. DOI : https://doi.org/10.1007/978-1-4471-0025-6.

[2] Alessandro DE LUCA, Giuseppe ORIOLO et Marilena VENDITTELLI. « Control of Wheeled Mobile Robots : An Experimental Overview ». In : (2001). Sous la dir. de Salvatore NICOSIA et al., p. 181-226. DOI : 10.1007/3-540-45000-9_8. URL : https://doi.org/10.1007/3-540-45000-9_8.

[3] Spyros G TZAFESTAS. « Mobile robot control and navigation : A global overview ». In : *Journal of Intelligent & Robotic Systems* 91 (2018), p. 35-58. DOI : https://doi.org/10.1007/s10846-018-0805-9.

[4] Eduardo D. SONTAG. « Stability and Feedback Stabilization ». In : *Encyclopedia of Complexity and Systems Science*. Sous la dir. de Robert A. MEYERS. New York, NY : Springer New York, 2009, p. 8616-8630. ISBN : 978-0-387-30440-3. DOI : 10.1007/978-0-387-30440-3_515. URL : https://doi.org/10.1007/978-0-387-30440-3_515.

[5] G. CAMPION, B. d'Andrea NOVEL et G. BASTIN. « Modelling and state feedback control of nonholonomic mechanical systems ». In : *[1991] Proceedings of the 30th IEEE Conference on Decision and Control*. 1991, 1184-1189 vol.2. DOI : 10.1109/CDC.1991.261553.

[6] De LUCA et al. « Control of nonholonomic systems via dynamic compensation ». eng. In : *Kybernetika* 29.6 (1993), p. 593-608. URL : http://eudml.org/doc/28187.

[7] B. d'Andréa NOVEL, G. CAMPION et G. BASTIN. « Control of Nonholonomic Wheeled Mobile Robots by State Feedback Linearization ». In : *The International Journal of Robotics Research* 14.6 (1995), p. 543-559. DOI : 10.1177/027836499501400602. URL : https://doi.org/10.1177/027836499501400602.

[8] Zhong-Ping JIANG et H. NIJMEIJER. « A recursive technique for tracking control of nonholonomic systems in chained form ». In : *IEEE Transactions on Automatic Control* 44.2 (1999), p. 265-279. DOI : 10.1109/9.746253.

[9] C. SAMSON. « Control of chained systems application to path following and time-varying point-stabilization of mobile robots ». In : *IEEE Transactions on Automatic Control* 40.1 (1995), p. 64-77. DOI : 10.1109/9.362899.

[10] Claude SAMSON. « Time-varying Feedback Stabilization of Car-like Wheeled Mobile Robots ». In : *The International Journal of Robotics Research* 12.1 (1993), p. 55-64. DOI : 10.1177/027836499301200104. URL : https://doi.org/10.1177/027836499301200104.

[11] M. AICARDI et al. « Closed loop steering of unicycle like vehicles via Lyapunov techniques ». In : *IEEE Robotics and Automation Magazine* 2.1 (1995), p. 27-35. DOI : 10.1109/100.388294.

[12] C.C. de WIT et O.J. SORDALEN. « Exponential stabilization of mobile robots with nonholonomic constraints ». In : *IEEE Transactions on Automatic Control* 37.11 (1992), p. 1791-1797. DOI : 10.1109/9.173153.

[13] R.T. M'CLOSKEY et R.M. MURRAY. « Exponential stabilization of driftless nonlinear control systems using homogeneous feedback ». In : *IEEE Transactions on Automatic Control* 42.5 (1997), p. 614-628. DOI : 10.1109/9.580865.

[14] P. MORIN et C. SAMSON. « Application of Backstepping Techniques to the Time-Varying Exponential Stabilisation of Chained Form Systems ». In : *European Journal of Control* 3.1 (1997), p. 15-36. ISSN : 0947-3580. DOI : https://doi.org/10.1016/S0947-3580(97)70059-1. URL : https://www.sciencedirect.com/science/article/pii/S0947358097700591.

[15] O.J. SORDALEN et O. EGELAND. « Exponential stabilization of nonholonomic chained systems ». In : *IEEE Transactions on Automatic Control* 40.1 (1995), p. 35-49. DOI : 10.1109/9.362901.

[16] Alessandro DE LUCA, Giuseppe ORIOLO et Marilena VENDITTELLI. « Stabilization of the Unicycle Via Dynamic Feedback Linearization ». In : *IFAC Proceedings Volumes* 33.27 (2000). 6th IFAC Symposium on Robot Control (SYROCO 2000), Vienna, Austria, 21-23 September 2000, p. 687-692. ISSN : 1474-6670. DOI : https://doi.org/10.1016/S1474-6670(17)38011-4. URL : https://www.sciencedirect.com/science/article/pii/S1474667017380114.

[17] C. Canudas de WIT et al. « Nonlinear control design for mobile robots ». In : *Recent trends in mobile robots*. World Scientific, 1993, p. 121-156. DOI : 10.1142/9789814354301_0005. URL : https://www.worldscientific.com/doi/abs/10.1142/9789814354301_0005.

[18] A. DE LUCA, G. ORIOLO et C. SAMSON. « Feedback control of a nonholonomic car-like robot ». In : (1998). Sous la dir. de J. P. LAUMOND, p. 171-253. DOI : 10.1007/BFb0036073. URL : https://doi.org/10.1007/BFb0036073.

[19] Welid BENCHOUCHE, Rabah MELLAH et Mohammed Salah BENNOUNA. « The Impact of the Dynamic Model in Feedback Linearization Trajectory Tracking of a Mobile Robot ». In : *Periodica Polytechnica Electrical Engineering and Computer Science* 65.4 (2021), 329–343. DOI : 10.3311/PPee.17127. URL : https://pp.bme.hu/eecs/article/view/17127.

[20] Welid BENCHOUCHE, Rabah MELLAH et Mohammed Salah BENNOUNA. « Navigation of a Differential Drive Mobile Robot Using Nonlinear Model Predictive Control ». In : *Conference Proceedings ICCSA'2021*, p. 78. URL : https://ceur-ws.org/Vol-2904/55.pdf.

[21] J. FORET, O. BRUNEAU et J.G. FONTAINE. « Unified approach for m-stability analysis and control of legged robots ». In : *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No.03CH37453)*. T. 1. 2003, 106-111 vol.1. DOI : 10.1109/IROS.2003.1250613.

[22] Jawhar GHOMMAM et Gerard POISSON. « Motion coordination control of multiple marine crafts ». In : *2008 10th IEEE International Workshop on Advanced Motion Control*. 2008, p. 44-49. DOI : 10.1109/AMC.2008.4516039.

[23] Vincent CREUZE, Olivier PARODI et Xianbo XIANG. « Design, simulation and experimental results of Taipan 300, a new Autonomous Underwater Vehicle prototype ». In : *OCEANS 2009-EUROPE*. 2009, p. 1-6. DOI : 10.1109/OCEANSE.2009.5278200.

[24] Rahul GOEL et al. « Modeling, Simulation and Flight Testing of an Autonomous Quadrotor ». In : mai 2009.

[25] Hao ZHOU et al. « Modeling and Experimental Characterization of Propulsion of a Spiral-Type Microrobot for Medical Use in Gastrointestinal Tract ». In : *IEEE Transactions on Biomedical Engineering* 60.6 (2013), p. 1751-1759. DOI : 10.1109/TBME.2012.2228001.

[26] B. H. RAGHUNATH et al. « Mathematical Modeling and Simulation of a Nanorobot Using Nano-hive Tool for Medical Applications ». In : *Nanoelectronics, Circuits and Communication Systems*. Sous la dir. de Vijay NATH et J.K. MANDAL. Singapore : Springer Singapore, 2021, p. 325-345.

[27] N. DIOLAITI et C. MELCHIORRI. « Teleoperation of a mobile robot through haptic feedback ». In : *IEEE International Workshop HAVE Haptic Virtual Environments and Their*. 2002, p. 67-72. DOI : 10.1109/HAVE.2002.1106916.

[28] Gregor KLANCAR et al. *Wheeled mobile robotics : from fundamentals towards autonomous systems*. Butterworth-Heinemann, 2017.

[29] Kenneth WALDRON et James SCHMIEDELER. « Kinematics ». In : *Springer Handbook of Robotics*. Sous la dir. de Bruno SICILIANO et Oussama KHATIB. Berlin, Heidelberg : Springer Berlin Heidelberg, 2008, p. 9-33. ISBN : 978-3-540-30301-5. DOI : 10.1007/978-3-540-30301-5_2. URL : https://doi.org/10.1007/978-3-540-30301-5_2.

[30] Ashitava GHOSAL. *Robotics : fundamental concepts and analysis*. Oxford university press, 2006.

[31] Jangmyung LEE, Hanuel YOON et Donghyuk LEE. « A Stable Tele-operation of a Mobile Robot with the Haptic Feedback ». In : *IFAC-PapersOnLine* 51.22 (2018). 12th IFAC Symposium on Robot Control SYROCO 2018, p. 13-18. ISSN : 2405-8963. DOI : https://doi.org/10.1016/j.ifacol.2018.11.511. URL : https://www.sciencedirect.com/science/article/pii/S2405896318332178.

[32] Grigore BURDEA, Paul RICHARD et Philippe COIFFET. « Multimodal virtual reality : Input-output devices, system integration, and human factors ». In : *International Journal of Human–Computer Interaction* 8.1 (1996), p. 5-24. DOI : 10.1080/10447319609526138. URL : https://doi.org/10.1080/10447319609526138.

[33] M. BERTOLUZZO et al. « Drive-by-wire systems for ground vehicles ». In : *2004 IEEE International Symposium on Industrial Electronics*. T. 1. 2004, 711-716 vol. 1. DOI : 10.1109/ISIE.2004.1571893.

[34] F.H. CLARKE, Yu.S. LEDYAEV et R.J. STERN. « Asymptotic Stability and Smooth Lyapunov Functions ». In : *Journal of Differential Equations* 149.1 (1998), p. 69-114. ISSN : 0022-0396. DOI : https://doi.org/10.1006/jdeq.1998.3476. URL : https://www.sciencedirect.com/science/article/pii/S0022039698934763.

[35] Vijay KADAKKAL et Gerald COOK. « Use of a preview control scheme with knowledge of future trajectory information for a lane tracking controller on a wheeled mobile robot ». In : *2008 34th Annual Conference of IEEE Industrial Electronics*. 2008, p. 1692-1697. DOI : 10.1109/IECON.2008.4758208.

[36] Zhen-yu LIU et al. « Trajectory Tracking Control of Wheeled Mobile Robots Based on the Artificial Potential Field ». In : *2008 Fourth International Conference on Natural Computation*. T. 7. 2008, p. 382-387. DOI : 10.1109/ICNC.2008.41.

[37] Xiao SHEN et Wuxi SHI. « Adaptive Trajectory Tracking Control of Wheeled Mobile Robot ». In : *2019 Chinese Control And Decision Conference (CCDC)*. 2019, p. 5161-5165. DOI : 10.1109/CCDC.2019.8833019.

[38] Jiang CHANG et Qingxin MENG. « Trajectory tracking control of nonholonomic wheeled mobile robots ». In : *The 2010 IEEE International Conference on Information and Automation*. 2010, p. 688-692. DOI : 10.1109/ICINFA.2010.5512422.

[39] Sašo BLAŽIČ. « A novel trajectory-tracking control law for wheeled mobile robots ». In : *Robotics and Autonomous Systems* 59.11 (2011), p. 1001-1007. ISSN : 0921-8890. DOI : https://doi.org/10.1016/j.robot.2011.06.005. URL : https://www.sciencedirect.com/science/article/pii/S0921889011001023.

[40] M. H. AMOOZGAR et Y. M. ZHANG. « Trajectory tracking of Wheeled Mobile Robots : A kinematical approach ». In : *Proceedings of 2012 IEEE/ASME 8th IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications*. 2012, p. 275-280. DOI : 10.1109/MESA.2012.6275574.

[41] Dinko OSMANKOVIC´ et Jasmin VELAGIĆ. « Gradient based adaptive trajectory tracking control for mobile robots ». In : *2013 XXIV International Conference on Information, Communication and Automation Technologies (ICAT)*. 2013, p. 1-6. DOI : 10.1109/ICAT.2013.6684086.

[42] F KÜNHE, J GOMES et W FETTER. « Mobile robot trajectory tracking using model predictive control ». In : *II IEEE latin-american robotics symposium*. T. 51. Citeseer. 2005.

[43] Gregor KLANČAR et Igor ŠKRJANC. « Tracking-error model-based predictive control for mobile robots in real time ». In : *Robotics and Autonomous Systems* 55.6 (2007), p. 460-469. ISSN : 0921-8890. DOI : https://doi.org/10.1016/j.robot.2007.01.002. URL : https://www.sciencedirect.com/science/article/pii/S0921889007000140.

[44] Felipe N. MARTINS et al. « An adaptive dynamic controller for autonomous mobile robot trajectory tracking ». In : *Control Engineering Practice* 16.11 (2008), p. 1354-1363. ISSN : 0967-0661. DOI : https://doi.org/10.1016/j.conengprac.2008.03.004. URL : https://www.sciencedirect.com/science/article/pii/S0967066108000373.

[45] R. FIERRO et F.L. LEWIS. « Control of a nonholonomic mobile robot : backstepping kinematics into dynamics ». In : 4 (1995), 3805-3810 vol.4. DOI : 10.1109/CDC.1995.479190.

[46] R. FIERRO et F.L. LEWIS. « Control of a nonholonomic mobile robot using neural networks ». In : *IEEE Transactions on Neural Networks* 9.4 (1998), p. 589-600. DOI : 10.1109/72.701173.

[47] Jong-Min YANG et Jong-Hwan KIM. « Sliding mode control for trajectory tracking of nonholonomic wheeled mobile robots ». In : *IEEE Transactions on Robotics and Automation* 15.3 (1999), p. 578-587. DOI : 10.1109/70.768190.

[48] Khoshnam SHOJAEI, Alireza Mohammad SHAHRI et Behzad TABIBIAN. « Design and implementation of an inverse dynamics controller for uncertain nonholonomic robotic systems ». In : *Journal of Intelligent and Robotic Systems* 71.1 (2013), p. 65-83. DOI : https://doi.org/10.1007/s10846-012-9762-x.

[49] Shirin VALILOO, Ehsan Khadem OLAMA et Amir Khadem OLAMA. « A sliding mode controller with generalized H2 performance for dynamic of nonholonomic mobile robot ». In : *2013 3rd Joint Conference of AI and Robotics and 5th RoboCup Iran Open International Symposium*. 2013, p. 1-7. DOI : 10.1109/RIOS.2013.6595329.

[50] Jiangshuai HUANG et al. « Adaptive output feedback tracking control of a nonholonomic mobile robot ». In : *Automatica* 50.3 (2014), p. 821-831. ISSN : 0005-1098. DOI : https://doi.org/10.1016/j.automatica.2013.12.036. URL : https://www.sciencedirect.com/science/article/pii/S0005109813005955.

[51] J TAHERI-KALANI et MJ KHOSROWJERDI. « Adaptive trajectory tracking control of wheeled mobile robots with disturbance observer ». In : *International Journal of Adaptive Control and Signal Processing* 28.1 (2014), p. 14-27. DOI : https://doi.org/10.1002/acs.2382.

[52] Anil ASWANI, Patrick BOUFFARD et Claire TOMLIN. « Extensions of learning-based model predictive control for real-time application to a quadrotor helicopter ». In : *2012 American Control Conference (ACC)*. 2012, p. 4661-4666. DOI : 10.1109/ACC.2012.6315483.

[53] Emine CANIGUR et Metin OZKAN. « Model reference adaptive control of a nonholonomic wheeled mobile robot for trajectory tracking ». In : *2012 International Symposium on Innovations in Intelligent Systems and Applications*. 2012, p. 1-5. DOI : 10.1109/INISTA.2012.6247005.

[54] YUANLIANG ZHANG. « Tracking control for wheeled mobile robots using neural network model algorithm control ». In : *Journal of Theoretical and Applied Information Technology* 46.2 (2012), p. 794-199.

[55] Andrea ALESSANDRETTI, A. Pedro AGUIAR et Colin N. JONES. « Trajectory-tracking and path-following controllers for constrained underactuated vehicles using Model Predictive Control ». In : *2013 European Control Conference (ECC)*. 2013, p. 1371-1376. DOI : 10.23919/ECC.2013.6669717.

[56] Cassius Z. RESENDE, Ricardo CARELLI et Mário SARCINELLI-FILHO. « A nonlinear trajectory tracking controller for mobile robots with velocity limitation via fuzzy gains ». In : *Control Engineering Practice* 21.10 (2013), p. 1302-1309. ISSN : 0967-0661. DOI : https://doi.org/10.1016/j.conengprac.2013.05.012. URL : https://www.sciencedirect.com/science/article/pii/S0967066113001056.

[57] Keeryun KANG et JVR PRASAD. « Development and flight test evaluations of an autonomous obstacle avoidance system for a rotary-wing UAV ». In : *Unmanned Systems* 1.01 (2013), p. 3-19. DOI : https://doi.org/10.1142/S2301385013500015.

[58] Seul JUNG, Eun Soo JANG et T.C. HSIA. « Collision Avoidance of a Mobile Robot Using Intelligent Hybrid Force Control Technique ». In : *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*. 2005, p. 4418-4423. DOI : 10.1109/ROBOT.2005.1570800.

[59] Katsuya SEKIGUCHI, Mingcong DENG et Akira INOUE. « Obstacle Avoidance and Two Wheeled Mobile Robot Control Using Potential Function ». In : *2006 IEEE International Conference on Industrial Technology*. 2006, p. 2314-2319. DOI : 10.1109/ICIT.2006.372703.

[60] Jiang LIHUA, Deng MINGCONG et Inoue AKIRA. « SVR based Obstacle Avoidance and Control of a Two Wheeled Mobile Robot ». In : IEEE, sept. 2007. DOI : 10.1109/icicic.2007.553. URL : https://cir.nii.ac.jp/crid/1360855568938876160.

[61] Mingcong DENG et al. « An obstacle avoidance method for two wheeled mobile robot ». In : *2007 IEEE International Conference on Networking, Sensing and Control*. 2007, p. 689-692. DOI : 10.1109/ICNSC.2007.372863.

[62] Gyula MESTER. « Obstacle - slope avoidance and velocity control of wheeled mobile robots using fuzzy reasoning ». In : *2009 International Conference on Intelligent Engineering Systems*. 2009, p. 245-249. DOI : 10.1109/INES.2009.4924770.

[63] C. G. RUSU, I. T. BIROU et E. SZÖKE. « Fuzzy based obstacle avoidance system for autonomous mobile robot ». In : *2010 IEEE International Conference*

on Automation, Quality and Testing, Robotics (AQTR). T. 1. 2010, p. 1-6. DOI : 10.1109/AQTR.2010.5520862.

[64] Yen-Sheng CHEN et Jih-Gau JUANG. « Intelligent obstacle avoidance control strategy for wheeled mobile robot ». In : *2009 ICCAS-SICE*. 2009, p. 3199-3204.

[65] Yi-Han CHEN et al. « Image-based obstacle avoidance and path-planning system ». In : *2013 International Conference on System Science and Engineering (ICSSE)*. 2013, p. 205-209. DOI : 10.1109/ICSSE.2013.6614660.

[66] Keigo WATANABE, Tatsuya KATO et Shoichi MAEYAMA. « Obstacle avoidance for mobile robots using an image-based fuzzy controller ». In : *IECON 2013 - 39th Annual Conference of the IEEE Industrial Electronics Society*. 2013, p. 6392-6397. DOI : 10.1109/IECON.2013.6700188.

[67] Adriana SÎRBU et Dan-Marius DOBREA. « Real-time genetic obstacle avoidance controller for a differential wheeled exploratory robot ». In : *International Symposium on Signals, Circuits and Systems ISSCS2013*. 2013, p. 1-4. DOI : 10.1109/ISSCS.2013.6651201.

[68] RASPBERRYPI. *Raspberry Pi 3 Model B*. Rapp. tech. Stontronics, 2016. URL : Web:www.stontronics.co.uk.

[69] QUANSER. *Qbot 2e Mobile Robot*. Rapp. tech. Quanser, 2016. URL : https://www.quanser.com/products/qbot-2e/.

[70] Rached DHAOUADI et A Abu HATAB. « Dynamic modelling of differential-drive mobile robots using lagrange and newton-euler methodologies : A unified framework ». In : *Advances in Robotics and Automation* 2.2 (2013), p. 1-7. DOI : 10.4172/2168-9695.1000107.

[71] G. ORIOLO, A. DE LUCA et M. VENDITTELLI. « WMR control via dynamic feedback linearization : design, implementation, and experimental validation ». In : *IEEE Transactions on Control Systems Technology* 10.6 (2002), p. 835-852. DOI : 10.1109/TCST.2002.804116.

[72] K. Rahul SHARMA, František DUŠEK et Daniel HONC. « Comparitive study of predictive controllers for trajectory tracking of non-holonomic mobile robot ». In : *2017 21st International Conference on Process Control (PC)*. 2017, p. 197-203. DOI : 10.1109/PC.2017.7976213.

[73] ZHONG-PING JIANGDAGGER et Henk NIJMEIJER. « Tracking control of mobile robots : A case study in backstepping ». In : *Automatica* 33.7 (1997), p. 1393-1399. DOI : https://doi.org/10.1016/S0005-1098(97)00055-1.

[74] Y. KANAYAMA et al. « A stable tracking control method for an autonomous mobile robot ». In : *Proceedings., IEEE International Conference on Robotics and Automation*. 1990, 384-389 vol.1. DOI : 10.1109/ROBOT.1990.126006.

[75] Arjan VAN DER SCHAFT. « Book review : Nonlinear systems analysis / by M. Vidyasagar. - Englewood Cliffs, NJ : Prentice-Hall, 1993. - ISBN 0-13-623463-1 ». Undefined. In : *Automatica* 30.10 (1994), p. 1631-1632. ISSN : 0005-1098. DOI : 10.1016/0005-1098(94)90103-1.

[76] H.R. BERENJI et P. KHEDKAR. « Learning and tuning fuzzy logic controllers through reinforcements ». In : *IEEE Transactions on Neural Networks* 3.5 (1992), p. 724-740. DOI : 10.1109/72.159061.

[77] Chris M BISHOP. « Neural networks and their applications ». In : *Review of scientific instruments* 65.6 (1994), p. 1803-1832. DOI : https://doi.org/10.1063/1.1144830.

[78] Samer HIJAZI, Rishi KUMAR, Chris ROWEN et al. « Using convolutional neural networks for image recognition ». In : *Cadence Design Systems Inc. : San Jose, CA, USA* 9 (2015), p. 1-12.

[79] G.K. VENAYAGAMOORTHY, V. MOONASAR et K. SANDRASEGARAN. « Voice recognition using neural networks ». In : *Proceedings of the 1998 South African Symposium on Communications and Signal Processing-COMSIG '98 (Cat. No. 98EX214)*. 1998, p. 29-32. DOI : 10.1109/COMSIG.1998.736916.

[80] N. KARUNANITHI, D. WHITLEY et Y.K. MALAIYA. « Using neural networks in reliability prediction ». In : *IEEE Software* 9.4 (1992), p. 53-59. DOI : 10.1109/52.143107.

[81] Daniel SVOZIL, Vladimír KVASNICKA et Jiri POSPICHAL. « Introduction to multi-layer feed-forward neural networks ». In : *Chemometrics and Intelligent Laboratory Systems* 39.1 (1997), p. 43-62. ISSN : 0169-7439. DOI : https://doi.org/10.1016/S0169-7439(97)00061-0. URL : https://www.sciencedirect.com/science/article/pii/S0169743997000610.

[82] Jiuxiang GU et al. « Recent advances in convolutional neural networks ». In : *Pattern Recognition* 77 (2018), p. 354-377. ISSN : 0031-3203. DOI : https://doi.org/10.1016/j.patcog.2017.10.013. URL : https://www.sciencedirect.com/science/article/pii/S0031320317304120.

[83] Larry R MEDSKER et LC JAIN. « Recurrent neural networks ». In : *Design and Applications* 5 (2001), p. 64-67. DOI : https://doi.org/10.1201/9781003040620.

[84] RAMANUJAM et SADAYAPPAN. « Optimization by neural networks ». In : *IEEE 1988 International Conference on Neural Networks*. 1988, 325-332 vol.2. DOI : 10.1109/ICNN.1988.23944.

[85] Sagar SHARMA, Simone SHARMA et Anidhya ATHAIYA. « Activation functions in neural networks ». In : *towards data science* 6.12 (2017), p. 310-316. URL : https://ijeast.com/papers/310-316,Tesma412,IJEAST.pdf.

[86] Lotfi A. ZADEH. « Fuzzy Logic, Neural Networks, and Soft Computing ». In : *Commun. ACM* 37.3 (mars 1994), 77–84. ISSN : 0001-0782. DOI : 10.1145/175247.175255. URL : https://doi.org/10.1145/175247.175255.

[87] Bart KOSKO et Satoru ISAKA. « Fuzzy logic ». In : *Scientific American* 269.1 (1993), p. 76-81. URL : https://www.amazon.com/Fuzzy-Thinking-Bart-Kosko/dp/B006UQ16YE.

[88] Manuel SANTOS. *Membership-functions*. 2022. URL : https://github.com/VManuelSM/Membership-functions/commits?author=VManuelSM.

[89] Omar TOUAMI et al. « An Application of a Fuzzy TOPSIS Multi-Criteria Decision Analysis Algorithm for Augmented Reality Maintenance Aid Systems Selection ». In : *2022 2nd International Conference on Advanced Electrical Engineering (ICAEE)*. 2022, p. 1-6. DOI : 10.1109/ICAEE53772.2022.9962127.

[90] H. NOMURA, I. HAYASHI et N. WAKAMI. « A learning method of fuzzy inference rules by descent method ». In : *[1992 Proceedings] IEEE International Conference on Fuzzy Systems*. 1992, p. 203-210. DOI : 10.1109/FUZZY.1992.258618.

[91] Mahieddine BENREGUIEG et al. « Fuzzy navigation strategy : application to two distinct autonomous mobile robots ». In : *Robotica* 15.6 (1997), 609–615. DOI : 10.1017/S0263574797000738.

[92] J.-S.R. JANG et Chuen-Tsai SUN. « Neuro-fuzzy modeling and control ». In : *Proceedings of the IEEE* 83.3 (1995), p. 378-406. DOI : 10.1109/5.364486.

[93] J.-S.R. JANG. « ANFIS : adaptive-network-based fuzzy inference system ». In : *IEEE Transactions on Systems, Man, and Cybernetics* 23.3 (1993), p. 665-685. DOI : 10.1109/21.256541.

[94] F.M. FRATTALE MASCIOLI, G.M. VARAZI et G. MARTINELLI. « Constructive algorithm for neuro-fuzzy networks ». In : *Proceedings of 6th International*

Fuzzy Systems Conference. T. 1. 1997, 459-464 vol.1. DOI : 10.1109/FUZZY.1997.616411.

[95] Jyh-Shing Roger JANG et E. MIZUTANI. « Levenberg-Marquardt method for ANFIS learning ». In : *Proceedings of North American Fuzzy Information Processing*. 1996, p. 87-91. DOI : 10.1109/NAFIPS.1996.534709.

[96] J.-S.R. JANG. « Input selection for ANFIS learning ». In : *Proceedings of IEEE 5th International Fuzzy Systems*. T. 2. 1996, 1493-1499 vol.2. DOI : 10.1109/FUZZY.1996.552396.

[97] Manish KUMAR et Devendra P GARG. « Intelligent learning of fuzzy logic controllers via neural network and genetic algorithm ». In : *Proceedings of*. 2004, p. 1-8. URL : https://www.iosrjournals.org/iosr-jece/papers/sicete-volume8/88.pdf.

[98] Rabah MELLAH, Saïd GUERMAH et Redouane TOUMI. « Adaptive control of bilateral teleoperation system with compensatory neural-fuzzy controllers ». In : *International Journal of Control, Automation and Systems* 15.4 (2017), p. 1949-1959. DOI : 10.1007/s12555-015-0309-3. URL : https://doi.org/10.1007/s12555-015-0309-3.

[99] Sebastian RUDER. « An overview of gradient descent optimization algorithms ». In : *arXiv preprint arXiv :1609.04747* (2016). DOI : https://doi.org/10.48550/arXiv.1609.04747.

[100] Simon S HAYKIN et Simon S HAYKIN. *Kalman filtering and neural networks*. T. 284. Wiley Online Library, 2001. DOI : Kalmanfilteringandneuralnetworks.

[101] L. PENG et Peng-Yung WOO. « Neural-fuzzy control system for robotic manipulators ». In : *IEEE Control Systems Magazine* 22.1 (2002), p. 53-63. DOI : 10.1109/37.980247.

[102] F LEWIS, C ABDALLAH et D DAWSON. « Control of robot ». In : *Manipulators, Editorial Maxwell McMillan, Canada* (1993), p. 25-36.

[103] M.L. CORRADINI et G. ORLANDO. « Control of mobile robots with uncertainties in the dynamical model : a discrete time sliding mode approach with experimental results ». In : *Control Engineering Practice* 10.1 (2002). Modelling and Control in Biomedical Systems, p. 23-34. ISSN : 0967-0661. DOI : https://doi.org/10.1016/S0967-0661(01)00109-5. URL : https://www.sciencedirect.com/science/article/pii/S0967066101001095.

[104] T. DAS et I.N. KAR. « Design and implementation of an adaptive fuzzy logic-based controller for wheeled mobile robots ». In : *IEEE Transactions on Control Systems Technology* 14.3 (2006), p. 501-510. DOI : 10.1109/TCST.2006.872536.

[105] Sebastian DUDZIK. « Application of the Motion Capture System to Estimate the Accuracy of a Wheeled Mobile Robot Localization ». In : *Energies* 13.23 (2020), p. 6437. DOI : https://doi.org/10.3390/en13236437.

[106] Guilherme V. RAFFO et al. « A Predictive Controller for Autonomous Vehicle Path Tracking ». In : *IEEE Transactions on Intelligent Transportation Systems* 10.1 (2009), p. 92-102. DOI : 10.1109/TITS.2008.2011697.

[107] P. FALCONE et al. « A linear time varying model predictive control approach to the integrated vehicle dynamics control problem in autonomous systems ». In : *2007 46th IEEE Conference on Decision and Control*. 2007, p. 2980-2985. DOI : 10.1109/CDC.2007.4434137.

[108] J. BACKMAN, T. OKSANEN et A. VISALA. « Navigation system for agricultural machines : Nonlinear Model Predictive path tracking ». In : *Computers and Electronics in Agriculture* 82 (2012), p. 32-43. ISSN : 0168-1699. DOI :

https://doi.org/10.1016/j.compag.2011.12.009. URL : https://www.sciencedirect.com/science/article/pii/S0168169911003218.

[109] Dongbing GU et Huosheng HU. « Receding horizon tracking control of wheeled mobile robots ». In : *IEEE Transactions on Control Systems Technology* 14.4 (2006), p. 743-749. DOI : 10.1109/TCST.2006.872512.

[110] Heonyoung LIM et al. « Nonlinear Model Predictive Controller Design with Obstacle Avoidance for a Mobile Robot ». In : *2008 IEEE/ASME International Conference on Mechtronic and Embedded Systems and Applications*. 2008, p. 494-499. DOI : 10.1109/MESA.2008.4735699.

[111] Dongbing GU et Huosheng HU. « A stabilizing receding horizon regulator for nonholonomic mobile robots ». In : *IEEE Transactions on Robotics* 21.5 (2005), p. 1022-1028. DOI : 10.1109/TRO.2005.851357.

[112] F. KUHNE, W.F. LAGES et J.M.G. da SILVA. « Point stabilization of mobile robots with nonlinear model predictive control ». In : *IEEE International Conference Mechatronics and Automation, 2005*. T. 3. 2005, 1163-1168 Vol. 3. DOI : 10.1109/ICMA.2005.1626717.

[113] Feng XIE et Rafael FIERRO. « First-state contractive model predictive control of nonholonomic mobile robots ». In : *2008 American Control Conference*. 2008, p. 3494-3499. DOI : 10.1109/ACC.2008.4587034.

[114] J.B. RAWLINGS et K.R. MUSKE. « The stability of constrained receding horizon control ». In : *IEEE Transactions on Automatic Control* 38.10 (1993), p. 1512-1516. DOI : 10.1109/9.241565.

[115] Lars GRÜNE et Jürgen PANNEK. « Nonlinear Model Predictive Control ». In : *Nonlinear Model Predictive Control : Theory and Algorithms*. London : Springer London, 2011, p. 43-66. ISBN : 978-0-85729-501-9. DOI : 10.1007/978-0-85729-501-9_3. URL : https://doi.org/10.1007/978-0-85729-501-9_3.

[116] Daniel B. LEINEWEBER et al. « An efficient multiple shooting based reduced SQP strategy for large-scale dynamic process optimization. Part 1 : theoretical aspects ». In : *Computers and Chemical Engineering* 27.2 (2003), p. 157-166. ISSN : 0098-1354. DOI : https://doi.org/10.1016/S0098-1354(02)00158-8. URL : https://www.sciencedirect.com/science/article/pii/S0098135402001588.

[117] Levente L. SIMON, Zoltan K. NAGY et Konrad HUNGERBUEHLER. « Swelling Constrained Control of an Industrial Batch Reactor Using a Dedicated NMPC Environment : OptCon ». In : *Nonlinear Model Predictive Control : Towards New Challenging Applications*. Sous la dir. de Lalo MAGNI, Davide Martino RAIMONDO et Frank ALLGÖWER. Berlin, Heidelberg : Springer Berlin Heidelberg, 2009, p. 531-539. ISBN : 978-3-642-01094-1. DOI : 10.1007/978-3-642-01094-1_43. URL : https://doi.org/10.1007/978-3-642-01094-1_43.

[118] Joel ANDERSSON, Johan ÅKESSON et Moritz DIEHL. « CasADi : A Symbolic Package for Automatic Differentiation andOptimal Control ». In : *Recent Advances in Algorithmic Differentiation*. Sous la dir. de Shaun FORTH et al. Berlin, Heidelberg : Springer Berlin Heidelberg, 2012, p. 297-307. ISBN : 978-3-642-30023-3.

[119] Welid BENCHOUCHE. *My experimental results*. 2023. URL : https://youtu.be/gKz79gdeUTw.

[120] F DUŠEK, D HONC et P ROZSÍVAL. « Mathematical model of differentially steered mobile robot ». In : *18th International Conference on Process Control, Tatranská Lomnica, Slovakia*. 2011, p. 221-229.

[121] Hocine KHATI. « Commande d'une architecture de téléopération par la carte FPGA ». Thèse de doct. UNIVERSITE MOULOUD MAMMERI TIZI-OUZOU, 2020. URL : https://www.ummto.dz/dspace/handle/ummto/11723.

[122] W. GORDON KRUBERG. *Gumstix DuoVero*. Rapp. tech. Gumstix, 2016. URL : https://www.gumstix.com/community/support/hardware/duovero/.

[123] QUANSER. *Quarc real time control software*. Rapp. tech. Quanser, Quarc, 2016. URL : https://www.quanser.com/products/quarc-real-time-control-software/.

Appendices

Commande à distance du robot mobile Qbot2e

1- Introduction

Un système de Commande à distance (téléopération) est une technologie qui permet à un opérateur humain de piloter des éléments à distance tout en se séparant de la zone de travail et de la machine sous contrôle. Ainsi, la commande à distance réduit les dangers associés aux environnements hasardeux et à la manipulation de produits toxiques [121]. Un système de téléopération se compose d'un maître qui passe des commandes, d'un robot esclave qui effectue le travail et d'un canal de communication qui permet aux deux (maître/esclave) d'interagir [121].

Le maître peut être un simple interrupteur marche/arrêt, un clavier, un téléphone portable, un joystick ou même un autre robot. Dans notre cas, comme nous n'avions pas d'autre robot, nous avons utilisé le maître comme clavier, et téléphone, nous n'avions pas non plus de joystick, mais l'avoir aurait été un grand avantage puisque nous pouvons utiliser la modulation de largeur d'impulsion sur le joystick pour changer les vitesses du robot en téléopération, simplement en utilisant un gain de glissière, et le module joystick qui vient avec la boîte à outils aérospatiale dans Simulink.

Comme mentionné précédemment (à l'état de l'art), le QBot 2e utilise une carte d'acquisition de données embarquée et un ordinateur embarqué sans fil pour mesurer les lectures des capteurs embarqués et les tics des encodeurs des moteurs d'entraînement. La connexion avec le QBot 2e s'effectue dans un environnement sans fil via le chipset WiFi "the Cypress CYW43455" embarqué dans le Raspberry Pi 3 modèle B. Un modèle Simulink est développé dans l'environnement MATLAB® 2021a. Le modèle créé est construit avec le logiciel de contrôle Quarc qui est inclus avec le Robot. Le modèle compilé est téléchargé avec une connexion TCP/IP vers QBot 2e et l'application est réalisée sur un ordinateur embarqué qui utilise un système d'exploitation Linux.

Le système informatique embarqué monté sur le robot mobile utilise le Gumstix DuoVero [122] pour exécuter QUARC et s'interfacer avec la carte d'acquisition de données QBot 2 [123].

2- Mise en place d'une connexion sans fil

La connexion entre l'ordinateur hôte et le QBot 2e est une communication sans fil, le QBot 2e communique avec l'ordinateur hôte en utilisant une connexion TCP/IP sans fil ad-hoc de type peer-to-peer (P2P). Le réseau établi est appelé WANET. Le robot mobile téléopéré QBot 2e est illustré à la (1). Le robot est livré avec un Routeur, ce dernier a le pont par défaut 192.168.2.1, le robot est pré-configuré à l'adresse ip : 192.168.2.19 . L'adaptateur wifi de l'ordinateur hôte doit être réglé sur l'adresse ip : 192.168.2.10 . Tous les masques de sous-réseau sont réglés sur : 255.255.255.0 .

Le QBot 2 peut être combiné à d'autres unités QBot pour une téléopération bilatérale, à des drones QBall 2 et QBall-X4, ou à des véhicules autonomes tiers, et une plateforme multi-agents à architecture ouverte peut être construite pour la recherche.

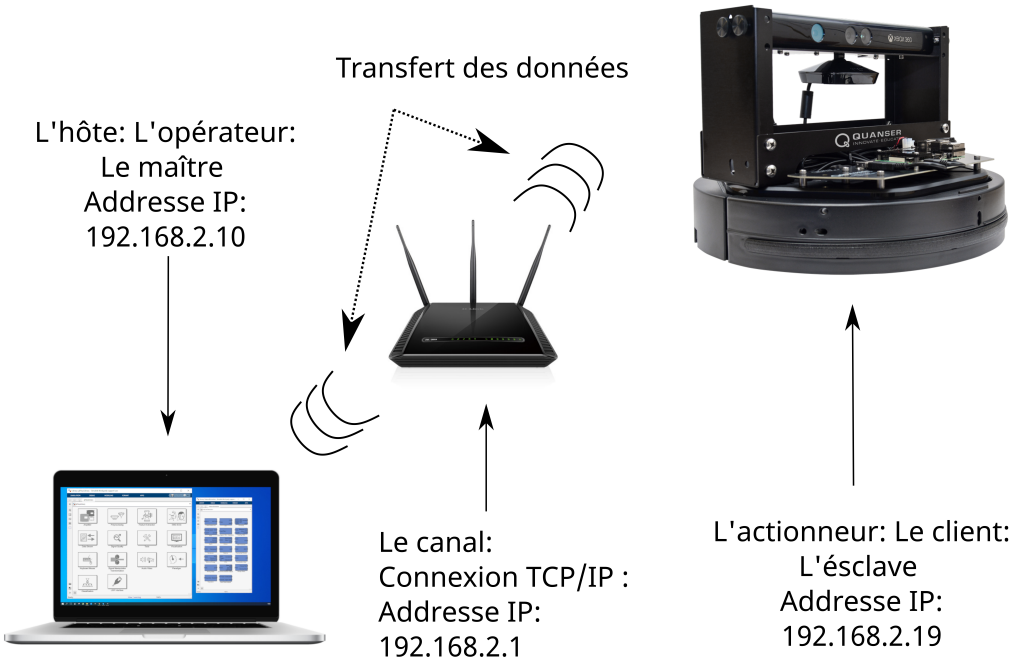

FIGURE 1 – Le système de Télé Opération du robot Qbot 2e

3- Structure du logiciel Quarc

Le logiciel de contrôle Quanser est un logiciel de contrôle rapide pour exécuter le code généré par Simulink sous Windows [123]. QUARC est l'approche la plus efficace pour utiliser Simulink® afin de concevoir, construire, déployer et valider des applications en expérimental sur du matériel [123]. Il est livré avec différentes versions pour chaque version de Matlab. Pour Matlab 2021a, Quarc 2021 SP1(4.1.3669) doit être installé sur l'ordinateur hôte. Quarc permet de régler les paramètres des applications en temps réel.

4- Ensemble de blocs Quarc utilisé

Normalement, comme Quanser est une grande entreprise avec de nombreux types de robots, chacun avec ses caractéristiques, ses capteurs, ses composants, etc..., elle devrait inclure dans son logiciel beaucoup de blocs pour couvrir tous ces types de robots, donc, nous allons parler de ceux que nous avons utilisés pour rendre notre robot fonctionnel, qui sont :

1. Bloc d'initialisation du matériel en boucle (HIL) : Le bloc HIL Initialize configure les pilotes et l'interface matérielle du matériel pour le QBot 2e
2. Lecture/écriture HIL : Les blocs HIL Read/Write sont utilisés pour lire les données sensorielles et piloter les moteurs.
3. Le bloc Host Initialize peut être utilisé pour utiliser des périphériques d'entrée externes, tels qu'un clavier (Host Keyboard) ou un joystick (Host Game Controller)
4. Video3D Initialize : Utilisé pour initialiser le capteur Kinect. La fréquence d'images maximale et la résolution sont définies dans ce bloc.

5. Video3D Capture : Capturez les informations RVB, de profondeur et autres en modifiant le type de flux.
6. Video Compressed Display : Transmet les données d'entrée compressées (RVB ou profondeur) de QBot 2e au PC et les affiche sur le moniteur.

5- résultats

Dans cette section, le robot mobile QBot 2e est commandé à distance pour explorer sa capacité à manœuvrer. Pour cela, un modèle Simulink est développé, illustré à la Figure (2).

Le bloc Scheme de la figure comprend 5 sous-blocs, le premier est le sous-bloc HIL, représenté en vert foncé, Il est responsable de l'importation de la bonne configuration pour le robot Quanser que l'on a, pour notre cas ,nous choisissons parmi les paramètres Qbot 2e comme indiqué dans la Figure (3). Une fois choisi, une image du robot apparaît dans le bas de la fenêtre du masque, une configuration par défaut des E/S sera également importée, pendant ce temps, les utilisateurs avancés peuvent changer ces paramètres à leur convenance, par exemple, attacher un capteur à l'ordinateur embarqué sur le robot, et activer les broches attachées à ce capteur vous donnerait accès à ses lectures.

Le deuxième sous-bloc est le HIL Read (lecture/écriture HIL), représenté en couleur cyan, il est responsable de l'émission des commandes de vitesse aux actionneurs Qbot 2e (block HIL Write), tout en fournissant à l'utilisateur des lectures sensorielles à une fréquence très élevée (HIL Read), comme présenté sur la Figure (4).

Les troisième et quatrième sous-blocs sont l'Initialisation du HOST (Initiliaze HOST), représentés respectivement en couleur verte et la Normalisation en couleur grise, ils activent les commandes d'entrée du clavier ou du joystick dans l'expériemental à une fréquence très élevée, ses paramètres vous permettent de choisir les touches ou les boutons que vous souhaitez activer, comme le montre la Figure (5). Dans cette Figure, nous pouvons voir que les valeurs de la vitesse linéaire sont de $0,18m/s$ et de la vitesse angulaire de $0,6rad/s$, ceci parce que nous appuyons sur la flèche vers le haut et la flèche vers la gauche du clavier, nous pouvons modifier ces valeurs selon nos préférences à partir des gains du sous-bloc de normalisation.

Le cinquième et dernier sous-bloc est constitué des sous-blocs 3D Initialize, Capture, et Compress, respectivement de gauche à droite, le premier, le deuxième et le cinquième, comme illustré dans la Figure (6). Le bloc 3D Initialize prépare la caméra, en configurant sa résolution, le type de flux, etc. Le bloc Capture permet de collecter les données Kinect et de les renvoyer à l'ordinateur par le biais du bloc Video Compressed représenté en rouge sur la Figure (6).

Le schéma de l'opération Tele supposée est présenté dans la Figure (7)

Le diagramme de l'opération Tele supposée est montré dans la Figure (7) L'opérateur utilise les informations de vision fournies sous forme de flux vidéo avec une résolution de $640x320$ pour surveiller et contrôler le robot mobile. Le capteur de ticks de l'encodeur attaché au robot mobile compte le nombre de ticks que les roues droite et gauche ont croisés, qui sont renvoyés au maître sous forme de vitesses après avoir été convertis en vitesses linéaires et angulaires.

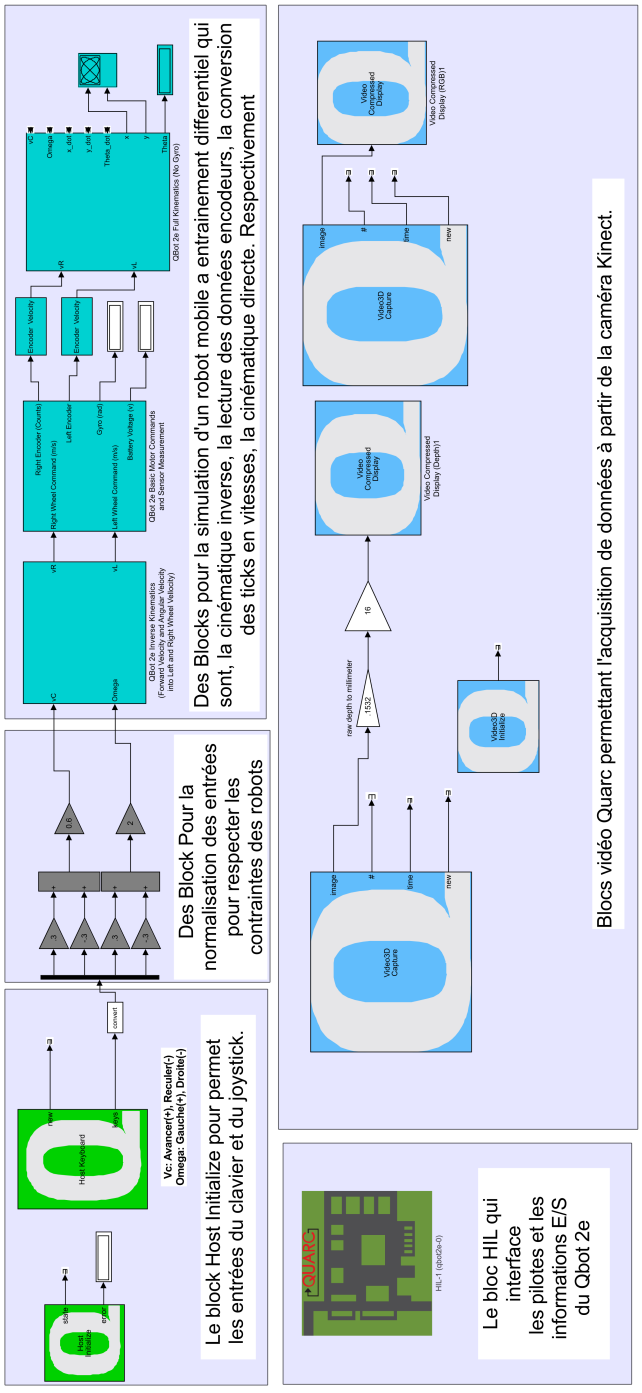

FIGURE 2 – Shemaa block Simulink de la Télé Opération du robot Qbot 2e

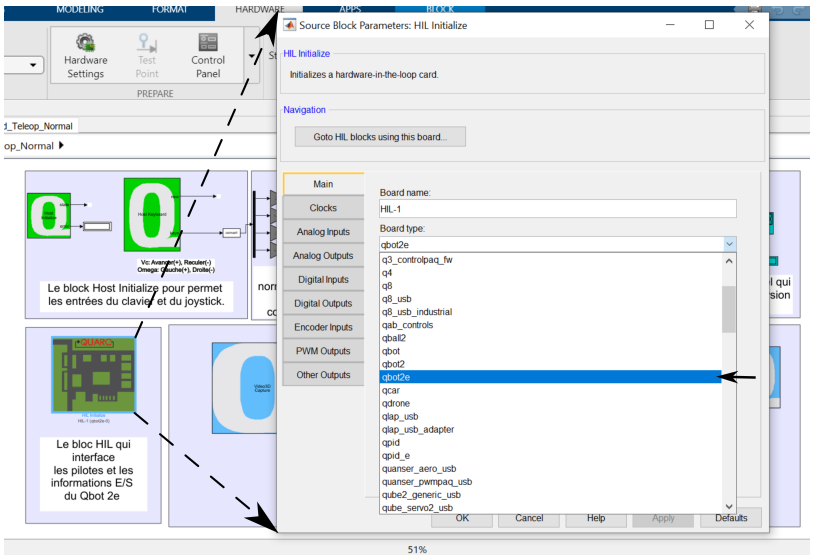

FIGURE 3 – Sous-block HIL Initialize

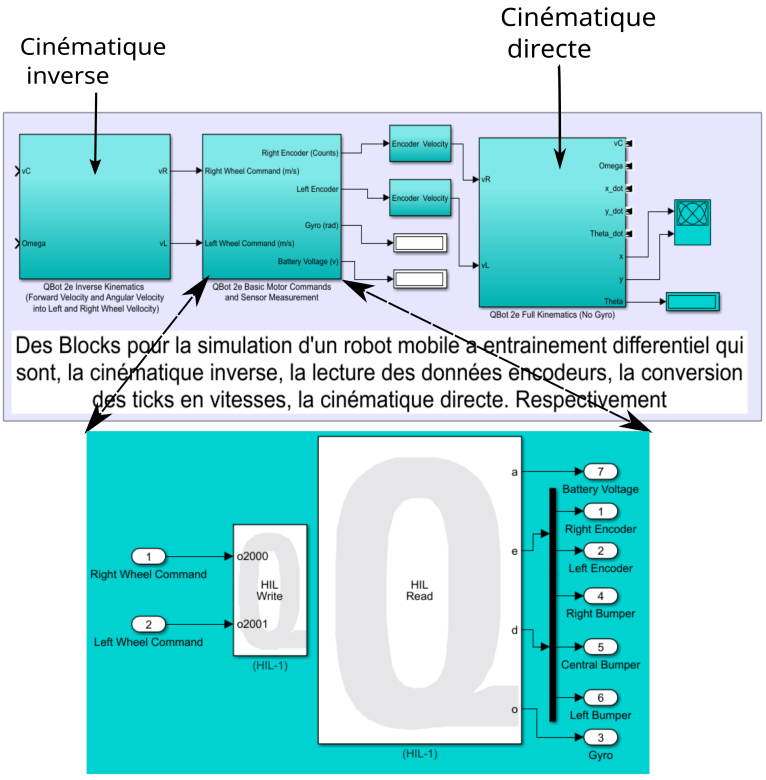

FIGURE 4 – Sous-block HIL Read/Write

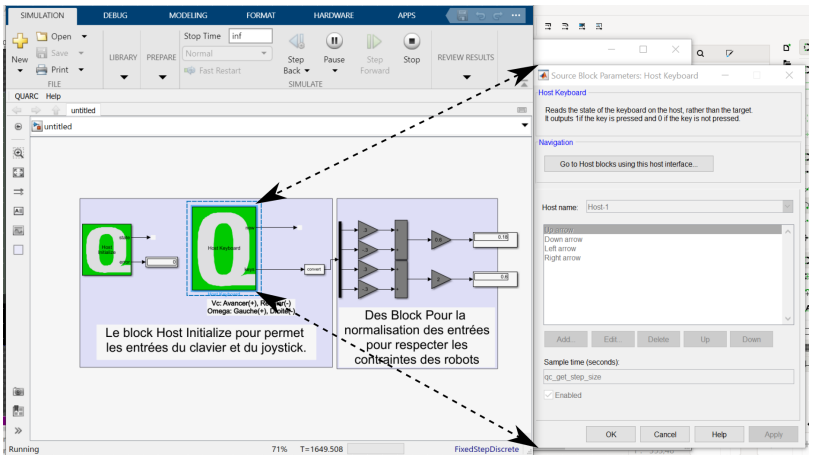

FIGURE 5 – Sous-block HOST

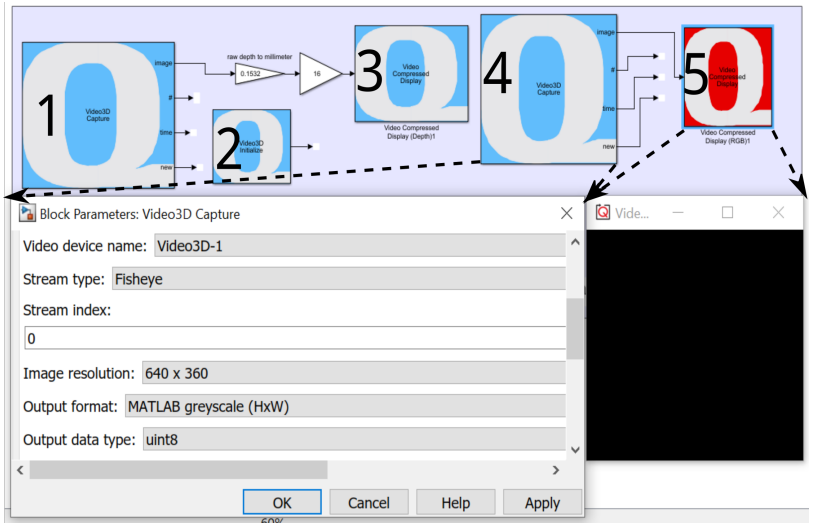

FIGURE 6 – Sous-block Visualisation

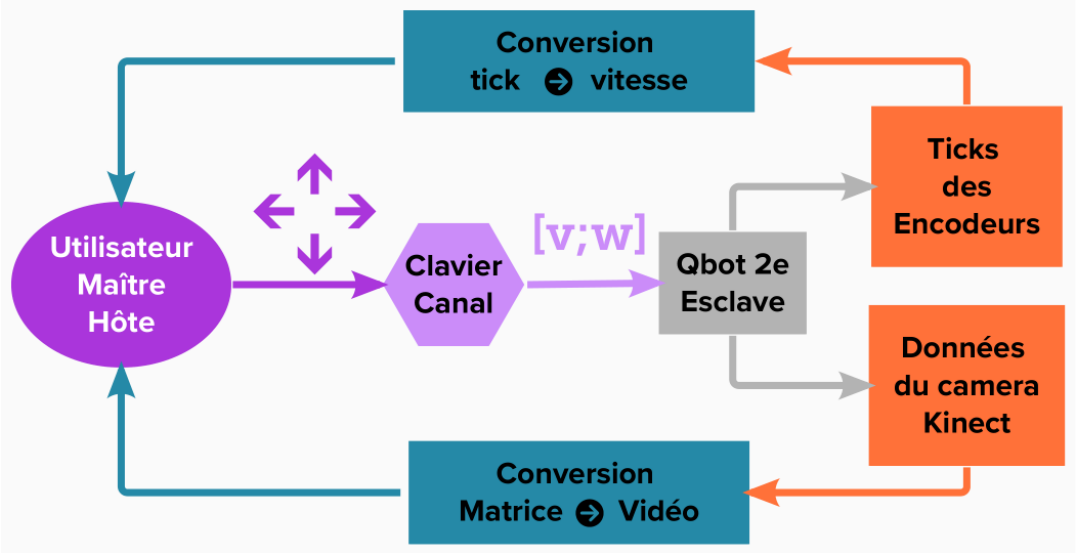

FIGURE 7 – Diagram de la Télé Opération

5- Conclusion

Dans cet annexe, un clavier a été utilisé pour la télé-opération d'un robot mobile. L'opérateur observe l'environnement de conduite à travers la caméra Kinect montée sur le Qbot 2e. La capacité de prototypage rapide du modèle conçu permet un contrôle précis et stable du robot mobile, et les sorties du modèle sont renvoyées avec précision à l'opérateur.