

**UNIVERSITÉ DU QUÉBEC EN ABITIBI-TÉMISCAMINGUE**

**SUIVI DE TRAJECTOIRES D'UN ROBOT MOBILE OMNIDIRECTIONNEL EN  
UTILISANT LE RÉSEAU DE NEURONES**

**MÉMOIRE**

**PRÉSENTÉ**

**COMME EXIGENCE PARTIELLE  
DE LA MAÎTRISE EN INGÉNIERIE**

**PAR**

**Mohamed Taher Rouissi**

**Juin 2022**



## DÉDICACE

À mes très aimables parents :  
Aucune dédicace ne saurait exprimer  
à sa juste valeur tout l'amour,  
toute la gratitude,  
et toute la reconnaissance que je porte !

À toute ma famille,  
pour m'avoir encouragé à suivre cette voie.

Mohamed Taher ROUSSI

## REMERCIEMENTS

Je tiens tout d'abord à remercier mon directeur de recherche M. Mohamad Saad de m'avoir accordé l'opportunité de faire partie de son équipe de recherche ainsi que pour tous les atouts et les efforts fournis pour me superviser.

J'exprime en outre, ma profonde gratitude à M. Nahi Kandil pour son aide, ses conseils et ses idées pour mon projet.

Mes remerciements vont aussi à mes amis Ahmed Kharrat et Mahmoud El Sayyah pour leurs aides tout au long de mon projet.

Finalement, un grand merci à toute les membres de ma famille et mes amis pour leurs soutiens quotidiens.

## TABLE DES MATIÈRES

DÉDICACE .....	<i>iii</i>
REMERCIEMENTS.....	<i>iv</i>
TABLE DES MATIÈRES .....	<i>v</i>
LISTE DES FIGURES.....	<i>viii</i>
LISTE DES TABLEAUX.....	<i>xii</i>
RÉSUMÉ .....	<i>I</i>
ABSTRACT.....	<i>ii</i>
INTRODUCTION GÉNÉRALE .....	<i>1</i>
CHAPITRE I ÉTAT DE L'ART.....	<i>6</i>
1.1 Introduction.....	<i>6</i>
1.2 Types des roues.....	<i>6</i>
1.2.1 Roues conventionnelles .....	<i>7</i>
1.2.2 Roues omnidirectionnelles:.....	<i>7</i>
1.3 Types des robots .....	<i>8</i>
1.3.1 Robot unicycle .....	<i>8</i>
1.3.2 Robot tricycle.....	<i>9</i>
1.3.3 Robot voiture .....	<i>9</i>
1.3.4 Robot omnidirectionnel .....	<i>10</i>
1.4 Comparaison entre les types des robots .....	<i>10</i>
1.5 Éléments constitutifs d'un robot .....	<i>12</i>

1.6	Fonctionnement du robot étudié .....	13
1.6.1	Capteurs infrarouges de mesure de distance .....	16
1.6.2	Encodeur incrémental .....	16
1.6.3	Capteur anticollision .....	16
1.6.4	Capteur de proximité inductif analogique .....	17
1.6.5	Capteurs diffus .....	17
1.7	Défis .....	18
1.7.1	Suivi de chemin .....	18
1.7.2	Suivi de trajectoire .....	18
1.7.3	Stabilisation des configurations fixes .....	19
1.8	Travaux reliés.....	19
1.8.1	Contrôle du suivi basé sur le modèle cinématique .....	19
1.8.2	Contrôleur de suivi de trajectoire par Backstepping.....	20
1.8.3	Contrôleurs de suivi de trajectoire par la mode glissante .....	21
1.8.4	Contrôleurs basés sur la logique floue de suivi de trajectoire .....	22
1.8.5	Contrôleurs adaptatifs de suivi de trajectoire .....	23
1.8.6	Contrôleurs de suivi de trajectoire basé sur le réseau de neurones.....	24
<b>CHAPITRE II MATÉRIEL ET MÉTHODES.....</b>		<b>26</b>
2.1	Introduction.....	26
2.2	Modélisation .....	26
2.2.1	Systèmes de coordonnées .....	27
2.2.2	Modélisation cinématique.....	28
2.3	Conception des régulateurs .....	35
2.3.1	Régulateur à base PI .....	35
2.3.2	Régulateurs de réseau de neurones : .....	40

CHAPITRE III	RÉSULTATS ET DISCUSSIONS.....	62
3.1	Introduction.....	62
3.2	Résultats de simulation sur SIMULINK.....	62
3.2.1	En utilisant le régulateur à base PI.....	63
3.2.2	En utilisant le régulateur de réseau de neurones entraîné hors ligne .	<a href="#">6968</a>
3.2.3	En utilisant le régulateur PI et le réseau de neurones entraîné en ligne	<a href="#">7572</a>
3.2.4	Interprétation.....	<a href="#">8277</a>
3.3	Résultats Expérimentales .....	<a href="#">8479</a>
3.3.1	En utilisant le régulateur à base PI.....	<a href="#">8479</a>
3.3.2	En utilisant le régulateur de réseau de neurones entraîné hors ligne .	<a href="#">9386</a>
3.3.3	En utilisant le régulateur PI et réseau de neurones entraîné en ligne	<a href="#">9990</a>
3.3.4	Interprétation.....	<a href="#">10495</a>
	CONCLUSIONS ET PERSPECTIVES.....	<a href="#">10797</a>
	RÉFÉRENCES.....	<a href="#">10898</a>
	ANNEXES .....	<a href="#">112102</a>

## LISTE DES FIGURES

FIGURE 1. 1 ROUE CONVENTIONNELLE [3] .....	7
FIGURE 1. 2 ROUE OMNIDIRECTIONNELLE .....	8
FIGURE 1. 3 ROBOT DE TYPE UNICYCLE.....	9
FIGURE 1. 4 ROBOT DE TYPE UNICYCLE.....	9
FIGURE 1. 5 ROBOT DE TYPE VOITURE.....	10
FIGURE 1. 6 ROBOT DE TYPE OMNIDIRECTIONNEL .....	10
FIGURE 1. 7 CONSTITUANTS MATÉRIELS D'UN ROBOT [27] .....	12
FIGURE 1. 8 MODULE DE L'UNITÉ D'ENTRAÎNEMENT DU ROBOT [28] .....	14
FIGURE 1. 9 EMPLACEMENT DES CAPTEURS .....	17
FIGURE 2. 1 ROBOT DANS LES REPÈRES .....	27
FIGURE 2. 2 MODÉLISATION DES VITESSES DU ROBOT [16] .....	29
FIGURE 2. 3 GÉOMÉTRIE DE LA PLATEFORME DU ROBOT MOBILE OMNIDIRECTIONNEL [16] .....	30
FIGURE 2. 4 STRUCTURE DU RÉGULATEUR À BASE PI.....	36
FIGURE 2. 5 MODÈLE MATHÉMATIQUE D'UN NEURONE [29] .....	41
FIGURE 2. 6 RÉSEAU DE NEURONES À UNE COUCHE AVEC S NEURONES [29].....	42
FIGURE 2. 7 UN RÉSEAU DE NEURONES À DEUX COUCHES [29].....	43
FIGURE 2. 8 STRUCTURE D'APPROXIMATION DES FONCTIONS POUR LES RÉSEAUX DE NEURONES.....	45
FIGURE 2. 9 RÉSEAU DE NEURONES À DEUX COUCHES [29] .....	46
FIGURE 2. 10 SCHÉMA FONCTIONNEL DE L'ALGORITHME DE RÉTROPROPAGATION.....	51
FIGURE 2. 11 ALGORITHME GÉNÉRAL DE MISE EN ŒUVRE DE LA FORMATION DES RÉSEAUX DE NEURONES PAR RÉTROPROPAGATION.....	52

FIGURE 2. 12 RÉSEAU DE NEURONES À DEUX COUCHES AVEC DEUX ENTRÉES ET UNE SORTIE [29].....	53
FIGURE 2. 13 MODÈLE DE RÉSEAU DE NEURONES UTILISÉ .....	56
FIGURE 2. 14 STRUCTURE DU RÉGULATEUR DE RÉSEAU DE NEURONES .....	57
FIGURE 2. 15 STRUCTURE DU RÉGULATEUR À BASE PI ET RÉSEAU DE NEURONES .....	59
FIGURE 3. 1 TRAJECTION LINÉAIRE DU ROBOT DANS LE PLAN XY .....	63
FIGURE 3. 2 ÉVOLUTION DES COMPOSANTES DU ROBOT AU COURS DU TEMPS .....	64
FIGURE 3. 3 ÉVOLUTION DES ERREURS AU COURS DU TEMPS .....	64
FIGURE 3. 4 TRAJECTOIRE CIRCULAIRE DU ROBOT DANS LE PLAN XY .....	65
FIGURE 3. 5 ÉVOLUTION DES COMPOSANTES DU ROBOT AU COURS DU TEMPS .....	66
FIGURE 3. 6 ÉVOLUTION DES ERREURS AU COURS DU TEMPS .....	66
FIGURE 3. 7 TRAJECTOIRE CIRCULAIRE DU ROBOT DANS LE PLAN XY .....	68
FIGURE 3. 8 ÉVOLUTION DES COMPOSANTES DU ROBOT AU COURS DU TEMPS .....	68
FIGURE 3. 9 ÉVOLUTION DES ERREURS AU COURS DU TEMPS .....	69
FIGURE 3. 10 TRAJECTOIRE LINÉAIRE DU ROBOT DANS LE PLAN XY .....	70
FIGURE 3. 11 ÉVOLUTION DES COMPOSANTES DU ROBOT AU COURS DU TEMPS .....	70
FIGURE 3. 12 ÉVOLUTION DES ERREURS AU COURS DU TEMPS .....	71
FIGURE 3. 13 TRAJECTOIRE CIRCULAIRE DU ROBOT DANS LE PLAN XY .....	72
FIGURE 3. 14 ÉVOLUTION DES COMPOSANTES AU COURS DU TEMPS .....	72
FIGURE 3. 15 ÉVOLUTION DES ERREURS AU COURS DU TEMPS .....	73
FIGURE 3. 16 TRAJECTOIRE ELLIPTIQUE DU ROBOT DANS LE PLAN XY .....	74
FIGURE 3. 17 ÉVOLUTION DES COMPOSANTES DU ROBOT AU COURS DU TEMPS .....	74
FIGURE 3. 18 ÉVOLUTION DES ERREURS AU COURS DU TEMPS .....	75
FIGURE 3. 19 TRAJECTOIRE LINÉAIRE DU ROBOT DANS LE PLAN XY .....	76
FIGURE 3. 20 ÉVOLUTION DES COMPOSANTES DU ROBOT DANS LE PLAN XY .....	76
FIGURE 3. 21 ÉVOLUTION DES ERREURS AU COURS DU TEMPS .....	77
FIGURE 3. 22 TRAJECTOIRE CIRCULAIRE DU ROBOT DANS LE PLAN XY .....	78

FIGURE 3. 23 ÉVOLUTION DES COMPOSANTES DU ROBOT AU COURS DU TEMPS .....	78
FIGURE 3. 24 ÉVOLUTION DES ERREURS AU COURS DU TEMPS .....	79
FIGURE 3. 25 TRAJECTOIRE ELLIPTIQUE DU ROBOT DANS LE PLAN XY .....	80
FIGURE 3. 26 ÉVOLUTION DES COMPOSANTES DU ROBOT AU COURS DU TEMPS .....	81
FIGURE 3. 27 ÉVOLUTION DES ERREURS AU COURS DU TEMPS .....	81
FIGURE 3. 28 COMPARAISON ENTRE LES 3 RÉGULATEURS POUR UNE TRAJECTOIRE LINÉAIRE .....	82
FIGURE 3. 29 COMPARAISON ENTRE LES 3 RÉGULATEURS POUR UNE TRAJECTOIRE CIRCULAIRE .....	83
FIGURE 3. 30 COMPARAISON ENTRE LES 3 RÉGULATEURS POUR UNE TRAJECTOIRE ELLIPTIQUE .....	83
FIGURE 3. 31 TRAJECTOIRE LINÉAIRE DU ROBOT DANS LE PLAN XY .....	85
FIGURE 3. 32 ÉVOLUTION DES COMPOSANTES DU ROBOT DANS LE PLAN XY .....	85
FIGURE 3. 33 ÉVOLUTION DES ERREURS AU COURS DU TEMPS .....	86
FIGURE 3. 34 TRAJECTOIRE CIRCULAIRE DU ROBOT DANS LE PLAN XY .....	87
FIGURE 3. 35 ÉVOLUTION DES ERREURS AU COURS DU TEMPS .....	88
FIGURE 3. 36 TRAJECTOIRE CIRCULAIRE DU ROBOT DANS LE PLAN XY .....	89
FIGURE 3. 37 ÉVOLUTION DES COMPOSANTES DU ROBOT AU COURS DU TEMPS .....	89
FIGURE 3. 38 ÉVOLUTION DES ERREURS AU COURS DU TEMPS .....	90
FIGURE 3. 39 TRAJECTOIRE ELLIPTIQUE DU ROBOT DANS LE PLAN XY .....	91
FIGURE 3. 40 ÉVOLUTION DES COMPOSANTES DU ROBOT AU COURS DU TEMPS .....	92
FIGURE 3. 41 ÉVOLUTION DES ERREURS AU COURS DU TEMPS .....	92
FIGURE 3. 42 TRAJECTOIRE LINÉAIRE DU ROBOT DANS LE PLAN XY .....	93
FIGURE 3. 43 ÉVOLUTION DES COMPOSANTES DU ROBOT DANS LE PLAN XY .....	94
FIGURE 3. 44 ÉVOLUTION DES ERREURS AU COURS DU TEMPS .....	94
FIGURE 3. 45 TRAJECTOIRE CIRCULAIRE DU ROBOT DANS LE PLAN XY .....	95
FIGURE 3. 46 ÉVOLUTION DES COMPOSANTES DU ROBOT DANS LE PLAN XY .....	96

FIGURE 3. 47 ÉVOLUTION DES ERREURS AU COURS DU TEMPS.....	96
FIGURE 3. 48 TRAJECTOIRE ELLIPTIQUE DU ROBOT DANS LE PLAN XY .....	97
FIGURE 3. 49 ÉVOLUTION DES COMPOSANTES DU ROBOT AU COURS DU TEMPS .....	98
FIGURE 3. 50 ÉVOLUTION DES ERREURS AU COURS DU TEMPS.....	98
FIGURE 3. 51 TRAJECTOIRE LINÉAIRE DU ROBOT DANS LE PLAN XY .....	99
FIGURE 3. 52 ÉVOLUTION DES COMPOSANTES DU ROBOT AU COURS DU TEMPS .....	100
FIGURE 3. 53 ÉVOLUTION DES ERREURS AU COURS DU TEMPS.....	100
FIGURE 3.54 TRAJECTOIRE CIRCULAIRE DU ROBOT DANS LE PLAN XY .....	101
FIGURE 3.55 ÉVOLUTION DES COMPOSANTES DU ROBOT AU COURS DU TEMPS.....	102
FIGURE 3. 56 ÉVOLUTION DES ERREURS AU COURS DU TEMPS.....	102
FIGURE 3. 57 TRAJECTOIRE ELLIPTIQUE DU ROBOT DANS LE PLAN XY .....	103
FIGURE 3. 58 ÉVOLUTION DES COMPOSANTES DU ROBOT AU COURS DU TEMPS .....	103
FIGURE 3. 59 ÉVOLUTION DES ERREURS AU COURS DU TEMPS.....	104
FIGURE 3. 60 COMPARAISON ENTRE LES 3 RÉGULATEURS POUR UNE TRAJECTOIRE LINÉAIRE.....	105
FIGURE 3. 61 COMPARAISON ENTRE LES 3 RÉGULATEURS POUR UNE TRAJECTOIRE CIRCULAIRE .....	105
FIGURE 3. 62 COMPARAISON ENTRE LES 3 RÉGULATEURS POUR UNE TRAJECTOIRE ELLIPTIQUE .....	106

## LISTE DES TABLEAUX

TABLEAU 1. 1 COMPARAISON DES DIFFÉRENTS TYPES DES ROBOTS.....	11
---	----

## RÉSUMÉ

Le présent travail porte principalement sur la conception et la mise en œuvre d'un contrôleur basé sur un réseau de neurones pour le suivi de la trajectoire d'un robot mobile omnidirectionnel. Trois algorithmes différents de contrôle de suivi: le premier est basé sur le régulateur PI. Les deux autres contrôleurs proposés sont des contrôleurs de réseaux de neurones: un réseau de neurones entraîné hors ligne pour remplacer le PI et un réseau de neurones entraîné en ligne combiné avec le PI.

L'intégration d'un contrôleur de réseau de neurones et du contrôleur cinématique présente l'avantage de traiter les incertitudes et les perturbations non modélisées et non structurées du système. Les avantages et les inconvénients des deux algorithmes de contrôle proposés seront discutés avec des illustrations.

Une modélisation complète du système, comprenant la cinématique du robot a été réalisée, et les résultats sont comparés pour vérifier la précision de chaque méthode. La simulation du modèle de robot et des différents contrôleurs a été réalisée à l'aide de Matlab/Simulink. Les contrôleurs de suivi de trajectoire proposés seront testés sur la plate-forme du robot mobile ROBOTINO, un robot mobile utilisé à des fins éducatives, de recherche et de formation par le centre de l'entreprise FESTO DIACTEC, et tous les résultats de suivi de trajectoire avec différentes trajectoires de référence sont présentés. La simulation du modèle du robot et des différents contrôleurs a été réalisée à l'aide de Matlab et de son toolbox Simulink. Plusieurs tests ont été effectués pour visualiser la trajectoire suivie par le robot et d'autres ont été réalisés sur le modèle physique pour valider les résultats théoriques et pratiques.

Mots clés: Robot mobile omnidirectionnel, robot holonome, PI, réseau de neurones.

## ABSTRACT

The present work focuses on the design and implementation of a neural network-based controller for tracking the trajectory of an omnidirectional mobile robot. Three different tracking control algorithms: the first one is based on the PI controller. The proposed other two controllers are neural network controllers: an offline trained neural network to replace the PI and an online trained neural network combined with the PI.

The integration of a neural network controller and the kinematic controller has the advantage of dealing with unmodeled and unstructured uncertainties and disturbances in the system. The advantages and disadvantages of the two proposed control algorithms will be discussed with illustrations.

A complete modeling of the system, including the robot kinematics has been performed, and the results are compared to verify the accuracy of each method. The simulation of the robot model and the different controllers was performed using Matlab/Simulink.

The proposed trajectory tracking controllers will be tested on the ROBOTINO mobile robot platform, a mobile robot used for educational, research and training purposes by the FESTO DIACTEC company center, and all the trajectory tracking results with different reference trajectories are presented. The simulation of the robot model and the different controllers was performed using Matlab and its toolbox Simulink. Several tests will be performed to visualize the trajectory tracked by the robot and tests must be done on the physical model to validate the theoretical and practical results.

Key words: Omnidirectional mobile robot, holonomic robot, PI, neural network.

## INTRODUCTION GÉNÉRALE

Ces dernières années, les robots mobiles sont largement intégrés dans notre société actuelle dans de nombreux lieux publics tels que les centres commerciaux et les aéroports ainsi que les industries, l'agriculture, les mines et dans les domaines militaires [1, 2]. Les robots mobiles sont les plus utilisés grâce à leur capacité à évoluer dans ces environnements, qu'ils soient terrestres, aériens ou maritimes. Les robots mobiles à roues sont le type de systèmes robotiques mobiles que nous envisageons dans ce projet car ils sont les plus largement utilisés parmi les classes de robots mobiles. Cela est dû à leurs contrôleurs simples, à leurs caractéristiques d'économie d'énergie et à leurs rapidités surtout sur les surfaces lisses.

Ils se développent rapidement, ainsi que leurs structures matérielles et logicielles, pour atteindre leurs principaux objectifs. L'un des objectifs importants est de résoudre les problèmes affectant le comportement de mobilité des robots. En conséquence, de nombreuses plateformes complexes ont été développées pour atteindre 3 degrés de mobilité (pour éviter les caractéristiques non holonomes). Nous pouvons classer les systèmes de robots mobiles à roues en deux catégories en fonction de leur type de roues: les roues conventionnelles et les roues suédoises. Pour la roue conventionnelle, le contact entre la roue et le sol est censé satisfaire le roulage pur sans contrainte de glissement. Cela implique que la vitesse du point de contact est égale à zéro et les composantes de la vitesse parallèle et orthogonale au plan de roue sont égales à zéro.

Au cours des deux dernières décennies, un certain nombre d'efforts de recherche considérables portant sur la mobilité des robots mobiles à roues holonomes a été réalisé.

Le comportement de mobilité du robot mobile à roues dépend principalement du type de ses roues et leurs vitesses de déplacement. Les roues omnidirectionnelles sont les roues les plus utilisés. Pour une roue suédoise, une seule composante de la vitesse du point de contact de la roue avec le sol est supposée égale à zéro le long du mouvement. La direction de cette composante nulle est arbitraire mais fixe par rapport à l'orientation de la roue. Sa construction consiste en des rouleaux montés autour de la roue principale. Elle a 3 degrés de mobilité. Les roues omnidirectionnelles montrent leurs performances efficaces avec différentes plateformes du robot mobile.

La majorité des robots ne peuvent pas conduire dans toutes les directions possibles. Pour cette raison, ces robots sont appelés « non holonomes ». En revanche, un robot « holonome » ou omnidirectionnel est capable de se déplacer dans toutes les directions.

La plupart des robots « non holonomes » ne peuvent pas conduire dans une direction perpendiculaire à leurs roues motrices. Par exemple, un robot à entraînement différentiel peut avancer et reculer, se déplacer dans une courbe ou tourner sur place, mais il ne peut pas conduire latéralement. Cependant, malgré ses complexités mécaniques, les robots omnidirectionnels sont stables et capable de conduire dans n'importe quelle direction dans un plan 2D et de tourner sur soi-même.

Les robots mobiles omnidirectionnels sont de plus en plus populaires dans de nombreuses applications, en particulier dans les espaces réduits, car ils peuvent effectuer des mouvements de translation et de rotation indépendamment et simultanément. En d'autres termes, ils peuvent se déplacer dans n'importe quelle direction avec un angle d'orientation.

Le robot mobile omnidirectionnel est constitué d'une plateforme équipée de 3 roues orientables décentrées et placées en triangle équilatérale. Chacune de ces roues est entraînée indépendamment par un moteur à courant continu qui est à son tour alimenté

par une tension de commande. Les principaux inconvénients des variantes de conception mentionnées sont la forte consommation d'énergie des actionneurs et la nécessité d'un entretien fréquent. C'est pourquoi les robots mobiles à roues sont devenus l'un des systèmes d'ingénierie complexes à développer. Jusqu'à présent, la réduction de cette complexité n'a pas reçu beaucoup d'attention car certains robots sont des prototypes expérimentaux qui ne sont pas exposés aux exigences rigoureuses des produits commerciaux.

Durant les dernières années, le nombre des publications et recherches scientifiques est renforcé dans l'ingénierie robotique. Ces recherches se concentrent principalement sur la résolution des problèmes de commande des systèmes non-linéaire dans le but d'élaborer les techniques de commande par la réduction des influences des perturbations extérieures et intérieures, la non-linéarité, les erreurs de modélisations, l'incertitudes des paramètres du système physique et les erreurs des mesures des capteurs. Dans l'ingénierie robotique, les problèmes des suivis de trajectoire et de chemin restent toujours les tâches les plus importantes pour les robots mobiles.

Les travaux menés sur l'élaboration des lois de commande pour les systèmes non linéaires ont permis de proposer plusieurs lois. Malgré tout ceci, il n'existe pas de technique standard de synthèse de lois face à un problème précis dans le cas des systèmes non linéaires.

Les capacités d'apprentissage et les bonnes performances pour l'approximation de fonctions non linéaires sont la raison principale de la combinaison du contrôleur de réseau de neurones avec le régulateur PI.

Récemment, l'intérêt pour le contrôle de suivi des robots mobiles s'est accru avec diverses contributions théoriques et pratiques. Les approches de contrôle de suivi pour les robots mobiles sont principalement divisées en six approches :

- Linéarisation par retour d'état
- Contrôle mode glissant
- Contrôle Backstepping
- Couple pré-calculé
- Contrôle adaptatif
- Contrôle intelligent

L'objectif général de ce travail est de diriger le robot « ROBOTINO » à partir d'une loi de commande par réseau de neurones utilisant l'approximation du modèle direct du robot par réseau de neurones qui améliore la performance de suivi du régulateur PI pour qu'il suive la trajectoire désirée. Les objectifs spécifiques sont :

- Modéliser le robot étudié en tenant compte la non-linéarité de système
- Trouver une loi de commande robuste qui tient compte des aléas du système telles que les incertitudes des paramètres, les perturbations extérieures et la forme non linéaire du modèle du robot.
- Éliminer l'erreur entre la position réelle du robot et la position virtuelle (position de référence) lorsque le temps tend vers l'infini.
- Éliminer l'erreur de l'angle de l'orientation du robot.
- Assurer la stabilité du système par une fonction de Lyapunov.
- Analyse comparative détaillée entre deux contrôleurs de suivi traditionnels et le contrôleur proposé, en soulignant les avantages et les inconvénients de chacun d'eux.

Le travail comprend le développement et la construction de deux différents contrôleurs de suivi de trajectoire basés sur des réseaux de neurones. Le premier contrôleur est un contrôleur cinématique à base PI. Le deuxième contrôleur est un régulateur de réseaux de neurones entraîné hors ligne, et le troisième contrôleur est une combinaison de régulateur PI et du contrôleur de réseau de neurones entraîné en ligne.

Notre travail intitulé « Suivi de trajectoires d'un robot mobile omnidirectionnelle en utilisant le réseau de neurones » est une contribution parmi tant d'autre que nous apportons en ce qui concerne la commande des robots mobiles. Ce projet est proposé par deux des capacités les plus importantes des réseaux de neurones, ses capacités d'apprentissage et ses bonnes performances pour l'approximation de fonctions non linéaires. Ces deux capacités sont la raison principale de la combinaison du contrôleur de réseau de neurones avec le PI. Afin d'atteindre notre objectif de recherche, nous avons ordonné le travail comme suit: dans le premier chapitre, nous présentons les types des roues et des robots en comparant les caractéristiques de performance des robots. Dans le deuxième, on a développé le modèle cinématique du robot étudié, et on a développé les différentes lois de commande utilisées. Dans le troisième chapitre, on a présenté les résultats de simulation et les résultats expérimentaux des différentes lois de commande avec différentes trajectoires.

# CHAPITRE I

## ÉTAT DE L'ART

### 1.1 Introduction

L'étude des robots mobiles consiste à étudier et comprendre les différents types des robots, les structures et les roues utilisées, en outre, la structure et l'architecture du robot. La première partie de ce chapitre décrit les différents types de roues utilisées dans les robots mobiles ainsi qu'une classification des robots et une comparaison entre eux. Puis, on présente les constituants généraux des robots mobiles autonomes ainsi que le robot étudié. Parmi ces constituants se trouvent la structure mécanique, les actionneurs, les capteurs et le cerveau. Ensuite, on illustre les différents problèmes de commande que l'on peut faire avec le robot, en particulier les problèmes techniques, tels que le suivi de trajectoire, le suivi de chemin et la stabilisation des configurations fixes. Enfin, on va clôturer ce chapitre par un résumé des lois de commandes les plus utilisées pour assurer le contrôle des robots mobiles omnidirectionnels.

### 1.2 Types des roues

Un robot mobile à roues est un véhicule à roues capable de se déplacer de manière autonome sans conducteur humain externe, car il est équipé, pour son déplacement, de moteurs pilotés par un ordinateur embarqué. On peut classer les systèmes de robots

mobiles à roues en deux catégories en fonction de leur type de roues: les roues conventionnelles et les roues suédoises [3, 4].

### 1.2.1 Roues conventionnelles

Pour la roue conventionnelle, le contact entre la roue et le sol est supposé satisfaire la contrainte de roulement pur sans glissement. Cela implique que la vitesse du point de contact est égale à zéro et que les composantes de cette vitesse parallèles et orthogonales au plan de la roue sont égales à zéro. La Figure 1.1 présente la forme schématique d'une roue conventionnelle.

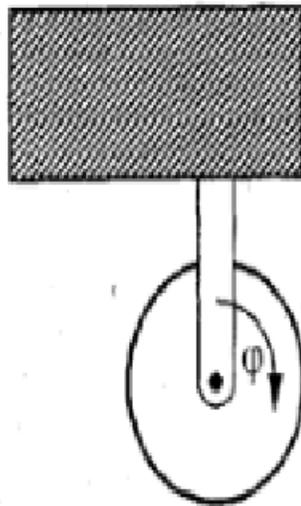


Figure 1. 1 Roue conventionnelle [3]

### 1.2.2 Roues omnidirectionnelles:

Pour une roue suédoise, une seule composante de la vitesse du point de contact de la roue avec le sol est supposée être égale à zéro le long du mouvement. La direction de cette composante nulle est arbitraire, mais elle est fixe par rapport à l'orientation de la roue.

La roue omnidirectionnelle est constituée de rouleaux assemblés autour de chaque roue principale.



Figure 1. 2 Roue omnidirectionnelle

### 1.3 Types des robots

Il existe plusieurs modèles des robots mobiles à roues. On peut les classifier selon le nombre, le type et l'emplacement des roues par rapport aux plateformes des robots en quatre classes principales:

#### 1.3.1 Robot unicycle

Un robot unicycle est un robot non-holonyme. En effet, il est impossible de se déplacer latéralement dans la direction perpendiculaire aux roues. Il peut se déplacer d'un point à un autre par une suite de lignes droites et de rotations simples. Il se compose principalement de deux roues motrices chacune est actionnée par un moteur à courant continu et une roue folle (roue libre) pour assurer sa stabilité. Son centre de rotation est situé sur l'axe reliant les deux roues motrices. Cependant, sa commande est plus simple (Figure 1.3).

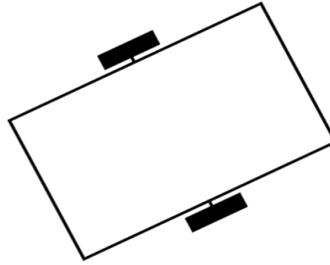


Figure 1. 3 Robot de type unicycle

### 1.3.2 Robot tricycle

Ce type des robots est non-holonyme. En effet, il est impossible de se déplacer latéralement dans la direction perpendiculaire aux roues. Sa commande est compliquée. Généralement, il est impossible de faire des rotations simples. Sa structure est constituée d'une roue centrée et orientable placée sur l'axe longitudinal et deux roues fixes placées sur un même axe. Le mouvement du robot est donné par l'orientation de la roue orientable et par la vitesse des deux roues fixes (Figure 1.4).

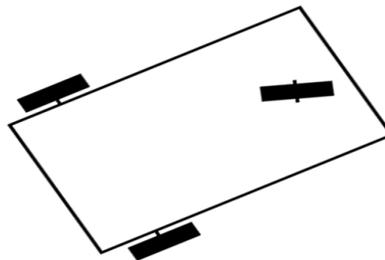


Figure 1. 4 Robot de type unicycle

### 1.3.3 Robot voiture

Un robot de type voiture est semblable au robot de type tricycle, mais il est plus stable mécaniquement. Il se compose de deux roues fixes placées sur un même axe et de deux roues centrées orientables placées sur un même axe, ce qui lui donne un point d'appui supplémentaire (Figure 1.5).

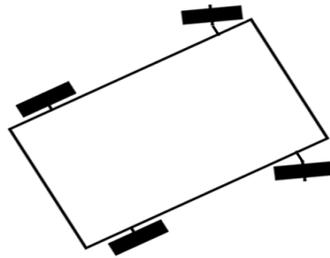


Figure 1. 5 Robot de type voiture

#### 1.3.4 Robot omnidirectionnel

Un robot omnidirectionnel est un robot holonome puisqu'il peut se déplacer librement dans toutes les directions et de tourner sur soi-même malgré sa complexité mécanique. Généralement, le robot omnidirectionnel est constitué de trois (3) roues décentrées orientables placées en triangle équilatéral (Figure 1.6).

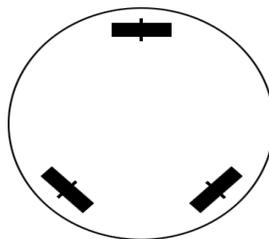


Figure 1. 6 Robot de type omnidirectionnel

#### 1.4 Comparaison entre les types des robots

Les critères de l'holonomie, la stabilité et la structure mécanique sont les critères principaux pour la performance des différents types des robots mobiles à roues. Le Tableau 1 représente une comparaison entre les types des robots mobiles à roues selon différentes caractéristiques [26].

Tableau 1. 1 Comparaison des différents types des robots

Type de Robot	Caractéristiques
<b>Robot Unicycle</b>	<ul style="list-style-type: none"> <li>- Non-holonome</li> <li>+ Stable</li> <li>+ Rotation sur soi-même</li> <li>+ Complexité mécanique faible</li> </ul>
<b>Robot Tricycle</b>	<ul style="list-style-type: none"> <li>- Non-holonome</li> <li>- Peu stable</li> <li>- Pas de rotation sur soi-même</li> <li>+ Complexité mécanique modérée</li> </ul>
<b>Robot Voiture</b>	<ul style="list-style-type: none"> <li>- Non-holonome</li> <li>+ Stable</li> <li>- Pas de rotation sur soi-même</li> <li>+ Complexité mécanique modérée</li> </ul>
<b>Robot Omnidirectionnel</b>	<ul style="list-style-type: none"> <li>+ Holonome</li> <li>+ Stable</li> <li>+ Rotation sur soi-même</li> <li>- Complexité mécanique importante</li> </ul>

## 1.5 Éléments constitutifs d'un robot

Généralement, la composition d'un robot autonome se base sur l'assemblage de pièces électromécaniques pour former la structure du robot et de pièces électroniques qui jouent le rôle de cerveau. Le robot possède une source d'énergie (généralement une batterie d'accumulateurs électriques embarquée).

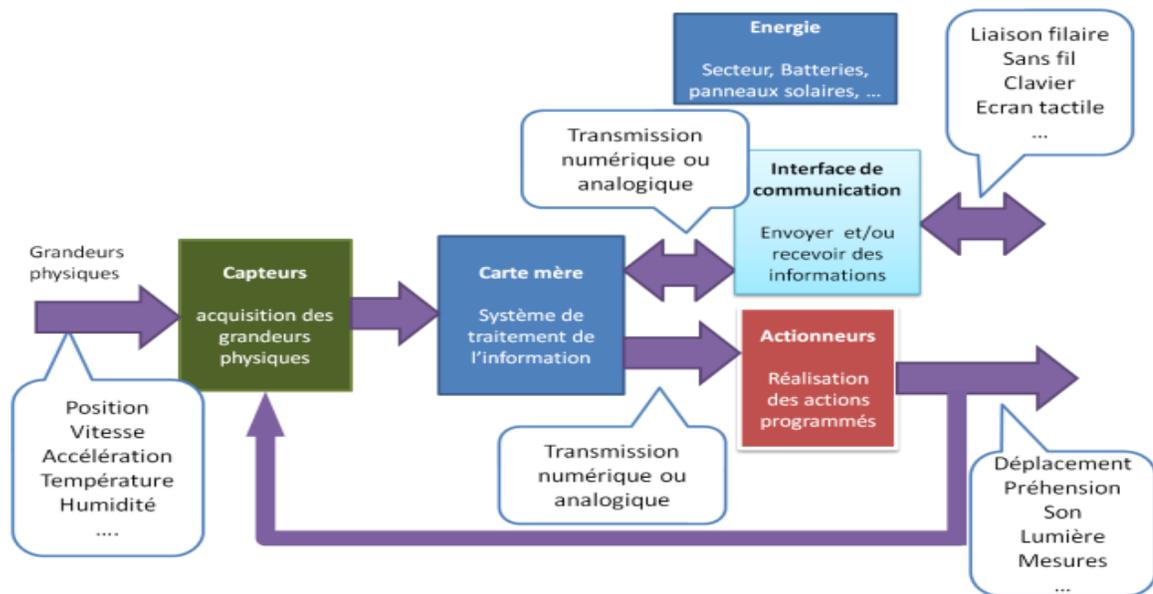


Figure 1. 7 Constituants matériels d'un robot [27]

Le système robotique se compose principalement de quatre groupes (Figure 1.7):

- La structure mécanique : elle représente le squelette du robot. L'importance de ce groupe s'incarne aux choix des articulations qui permettent de donner les mouvements possibles au robot c'est-à-dire le nombre de degré de liberté du système.
- Les actionneurs : ils sont les servo-moteurs qui permettent au système robotique d'effectuer les actions commandées par le cerveau.

- Les capteurs : ils représentent le dispositif sensoriel du système robotique. Ils permettent de recevoir les informations environnementales du robot.
- Le cerveau : c'est le dispositif qui permet au système robotique d'analyser les données reçues par les capteurs et d'envoyer les ordres de commandes aux actionneurs.

### 1.6 Fonctionnement du robot étudié

Robotino est un système de robot mobile entièrement fonctionnel et de haute qualité avec entraînement omnidirectionnel. Les trois unités d'entraînement permettent un mouvement dans toutes les directions avant, arrière et latéral et le robot peut également être tourné sur place. Il est également équipé d'une webcam et de plusieurs types de capteurs, analogues à mesure de la distance, par exemple binaires pour la protection contre les collisions, et numériques pour mesurer la vitesse réelle. Cela permet de répondre à toutes les exigences de ce type de système.

Le système peut être mis en service immédiatement sans PC. Le contrôleur de Robotino se compose d'un PC embarqué avec une carte Compact Flash, sur laquelle plusieurs applications de démonstration et le système d'exploitation (Linux) sont déjà installés. Robotino est autonome. De nombreux capteurs, une caméra et un contrôleur haute performance fournissent au système "l'intelligence" nécessaire.

Le contrôleur est accessible directement via un réseau local sans fil (WLAN). Lorsqu'il est correctement programmé, Robotino accomplit les tâches assignées de manière autonome. Des capteurs et des actionneurs supplémentaires peuvent être connectés via une interface E/S.

Les batteries rechargeables, les unités d'entraînement et la caméra sont montées sur le châssis, où se trouvent également les capteurs de mesure de distance et le capteur anti-

collision de mesure de la distance. Le châssis offre un espace supplémentaire et des options de montage pour d'autres accessoires, capteurs et/ou actionneurs.

Les composants du système les plus sensibles, tels que le contrôleur, le module d'E/S et les interfaces, sont situés sur le pont de commande. Le pont de commande est relié aux autres modules du système au moyen d'un connecteur. Robotino est entraîné par trois unités d'entraînement indépendantes et omnidirectionnelles. Elles sont montées à un angle de  $120^\circ$  l'un par rapport à l'autre.

Chacune des trois unités d'entraînement est composée des éléments suivants (Figure 1.8):

- Moteur à courant continu
- Réducteur avec un rapport de 32:1
- Rouleau toutes directions
- Courroie crantée
- Encodeur incrémental

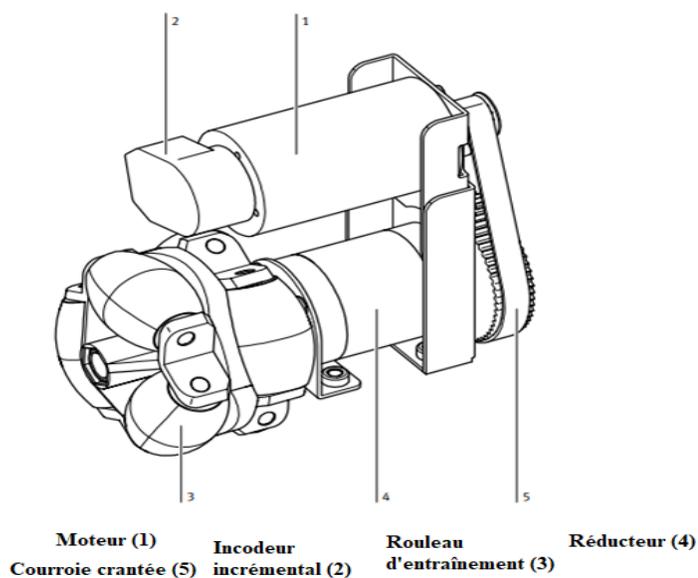


Figure 1. 8 Module de l'unité d'entraînement du robot [28]

L'unité de commande de Robotino se compose de trois éléments :

- Un processeur PC 104, compatible avec MOPSlcdVE, 300 MHz, et un système d'exploitation Linux avec noyau en temps réel.
- Une carte compact flash avec API C++ pour le contrôle de Robotino.
- Un point d'accès LAN sans fil

Il est important de savoir que :

- La carte de circuit E/S établit la communication entre l'unité de commande et les capteurs, les unités d'entraînement et l'interface E/S incluse dans Robotino.
- Chacun des moteurs des unités d'entraînement individuelles est contrôlé par un contrôleur PID. Chaque moteur peut être contrôlé individuellement.
- Les signaux provenant du codeur, de tous les capteurs installés de façon permanente et de tous les capteurs et actionneurs connectés à l'interface E/S sont transmis à l'unité de commande ou aux actionneurs supplémentaires.
- L'alimentation est assurée par deux batteries rechargeables de 12 V d'une capacité de 4 Ah. Les deux batteries rechargeables sont montées sur le châssis.
- Des capteurs (Figure 1.9) permettant de mesurer la distance aux objets et de détecter la vitesse du moteur sont intégrés à Robotino. Un capteur anticollision monté sur la circonférence du châssis signale tout contact avec des objets.

### 1.6.1 Capteurs infrarouges de mesure de distance

Robotino est équipé de 9 capteurs infrarouges de mesure de distance qui sont montés dans le châssis à un angle de 40° les uns par rapport aux autres. Grâce à ces capteurs, Robotino peut scruter toutes les zones environnantes à la recherche d'objets. Chacun des capteurs peut être interrogé individuellement via la carte de circuit imprimé E/S. Il est ainsi possible d'éviter les obstacles, de respecter les distances et de se diriger vers une cible sélectionnée.

Les capteurs sont capables d'effectuer des mesures précises ou relatives de la distance d'objets à des distances de 4 à 30 cm. La connexion du capteur est particulièrement simple et ne comprend qu'un seul signal de sortie analogique et une alimentation. L'électronique d'évaluation des capteurs détermine la distance et la lit sous forme de signal analogique.

### 1.6.2 Encodeur incrémental

La vitesse réelle de chaque moteur individuel est mesurée en RPM par le codeur incrémental. Si la vitesse réelle du moteur s'écarte du point de consigne, elle peut être ajustée pour correspondre à la valeur cible au moyen d'un contrôleur PID.

### 1.6.3 Capteur anticollision

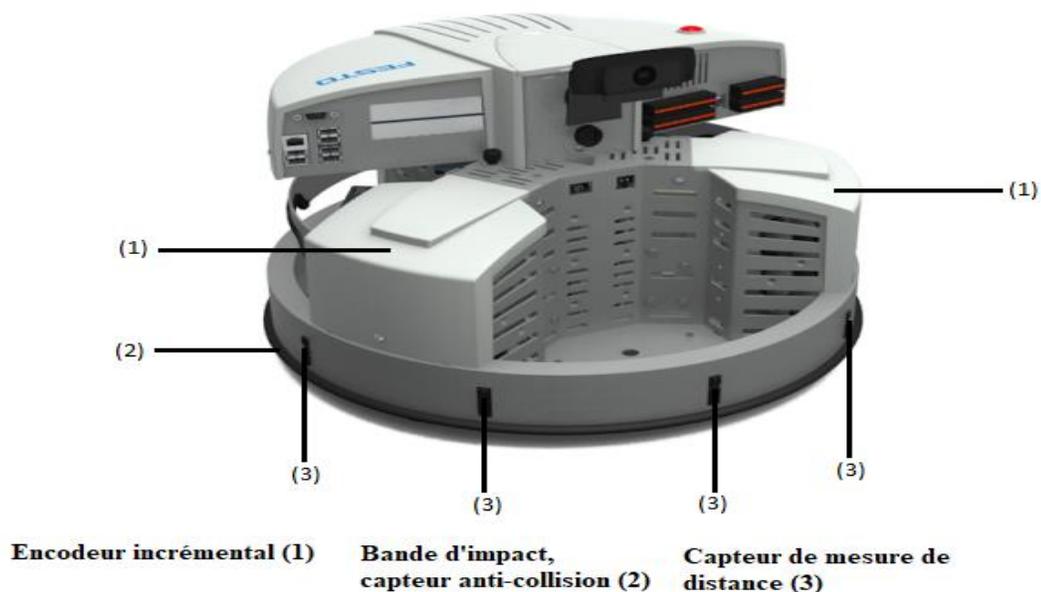
Le capteur anticollision est composé d'une bande de commutation qui est fixée sur toute la circonférence du châssis. Une chambre de commutation est située à l'intérieur d'un profilé en plastique. Deux surfaces conductrices sont situées à l'intérieur de la chambre, entre lesquelles un espace donné est maintenu. Ces surfaces sont court-circuitées lorsqu'une pression même minimale est appliquée sur la bande. Un signal reconnaissable de manière fiable est ainsi transmis à l'unité de commande. Les collisions avec des objets en tout point du boîtier sont détectées.

#### 1.6.4 Capteur de proximité inductif analogique

Le capteur de proximité inductif sert à détecter les objets métalliques sur le sol et est utilisé pour le contrôle de la trajectoire continue. Il lit des signaux d'intensité variable selon qu'il se trouve au milieu ou au bord de la bande métallique. Le suivi du chemin peut ainsi être contrôlé de manière différenciée. Le capteur de proximité inductif doit être fixé sur le support prévu à cet effet et doit être connecté à l'interface E/S.

#### 1.6.5 Capteurs diffus

Le suivi de trajectoire peut également être mis en œuvre avec les deux capteurs diffus inclus. Des câbles de fibre optique flexibles sont connectés à une unité de fibre optique qui fonctionne avec une lumière rouge visible. La lumière réfléchie est détectée. Des surfaces et des couleurs différentes produisent différents degrés de réflexion. Cependant, les différences graduelles dans la lumière réfléchie ne peuvent pas être détectées. Les capteurs doivent être fixés sur les supports prévus à cet effet et connectés à l'interface E/S.



**Figure 1. 9 Emplacement des capteurs**

L'unité de commande est équipée d'un emplacement dans lequel une carte PC a déjà été insérée. Cette carte PC contient le système d'exploitation, les bibliothèques de fonctions et les programmes inclus.

## 1.7 Défis

### 1.7.1 Suivi de chemin

Étant donné une courbe « $\zeta$ » du plan, une vitesse d'avancement non nulle  $v$  pour le robot mobile, ainsi qu'un point « $C$ » fixe sur ce robot, on souhaite que le point « $C$ » suive la courbe « $\zeta$ » lorsque le robot roule à la vitesse  $v$ , la variable que l'on doit régler à zéro est donc la distance du point « $C$ » à la courbe. Ce type de problème correspond à l'exécution d'une trajectoire indépendamment du temps, comme dans l'exemple de la conduite sur route lorsque le conducteur cherche à maintenir une distance constante par rapport à des marquages au sol [5].

### 1.7.2 Suivi de trajectoire

Ce problème se distingue du précédent par le fait que la vitesse d'avancement du véhicule n'est plus fixée à l'avancement, du fait que l'on cherche également à réguler la distance parcourue le long de la courbe « $\zeta$ ». Cet objectif nécessite de doter la courbe « $\zeta$ » d'une loi horaire, c'est-à-dire de la paramétrer par la variable temporelle « $t$ ». Cela revient à définir une trajectoire  $t \rightarrow (x_r(t), y_r(t))$  par rapport à un repère  $R$ . On souhaite alors réguler à zéro le vecteur d'erreur  $(x_r(t) - x(t), y_r(t) - y(t))$ , où  $(x(t), y(t))$  désignent les coordonnées du point « $C$ » dans le repère  $R$  à l'instant  $t$ . Le problème peut également être interprété comme celui consistant à asservir le véhicule à un véhicule de référence, dont la trajectoire est donnée par  $t \rightarrow (x_r(t), y_r(t))$  [6].

### 1.7.3 Stabilisation des configurations fixes

Autrement dit commande point à point. Dans ce cas une configuration dite de référence est choisie et on cherche à établir une loi de commande permettant au robot d'atteindre la référence.

Les lois de commande que nous avons recensées ici sont celles couramment rencontrées dans la littérature pour résoudre le problème de suivi de chemin, de poursuite de trajectoire et de stabilisation de configuration fixe des robots mobiles [7].

## 1.8 Travaux reliés

Il y a beaucoup de publications sur la résolution du problème du mouvement des robots mobiles holonomes et de nombreux contrôleurs non linéaires qui ont été proposés dans la littérature pour le suivi de trajectoires de ces robots. Certains types de ces contrôleurs de suivi de trajectoire seront brièvement expliqués dans cette section afin de clarifier les algorithmes de contrôle proposés dans ce projet.

### 1.8.1 Contrôle du suivi basé sur le modèle cinématique

Plusieurs études ont été basées sur le modèle cinématique, telles que [8], [9] et [10]. La loi de contrôle proposée dans ces travaux ne tient pas compte des effets de la dynamique des robots. Les paramètres appropriés du contrôleur ont été déterminés en linéarisant les équations différentielles du système et en trouvant une condition d'amortissement critique. La nécessité de limiter la vitesse et l'accélération est également discutée dans ces travaux [8, 9, 10].

Pour calculer les entrées de contrôle des véhicules, on suppose qu'il y a un suivi parfait de la vitesse. Cette approche pose trois problèmes : premièrement, le problème du suivi de la vitesse parfaite ne tient pas dans la pratique. Deuxièmement, les perturbations sont ignorées et troisièmement, une connaissance complète de la dynamique est nécessaire. Afin d'améliorer les performances de l'algorithme de contrôle proposé,

d'autres contrôleurs incluant la dynamique du véhicule et pouvant traiter les perturbations sont proposés.

### 1.8.2 Contrôleur de suivi de trajectoire par Backstepping

Le travail, qui est une extension dynamique permettant l'intégration d'un contrôleur cinématique et d'un contrôleur de couple pour les robots mobiles holonomes, a été proposé dans [11], [12] et [13]. Un contrôleur combiné cinématique/couple est développé en utilisant la commande par Backstepping et la stabilité est garantie par la théorie de Lyapunov. L'algorithme de contrôle proposé peut être appliqué à trois problèmes de navigation de robots : le suivi d'une trajectoire de référence, le suivi de la trajectoire et la stabilisation autour d'une position de référence. Cet algorithme de contrôle fournit une méthode rigoureuse de prise en compte de la dynamique spécifique du véhicule pour convertir la commande du système de direction en entrée de contrôle pour le véhicule réel. Tout d'abord, les entrées de contrôle de la vitesse de retour d'état sont conçues pour le système de direction cinématique afin de rendre l'erreur de position asymptotiquement stable. Ensuite, un contrôleur de couple calculé est conçu de telle sorte que les vitesses du robot mobile convergent vers les entrées de vitesse données. Cette approche de contrôle peut être appliquée à une classe de commandes de vitesse du système cinématique lisse et par conséquent, la même procédure de conception fonctionne pour les trois problèmes de navigation de base.

Cette loi de contrôle, qui prend en compte la dynamique complète du robot mobile, est calculée en utilisant l'approche du Backstepping. La méthode est valable tant que les entrées de contrôle de la vitesse sont lisses et limitées et que la dynamique est entièrement connue. En fait, la connaissance parfaite des paramètres du robot mobile est impossible à atteindre. Par exemple, la friction est très difficile à modéliser dans ces systèmes. Pour faire face à ce problème, un contrôleur adaptatif robuste doit être conçu de manière à pouvoir traiter les incertitudes et les perturbations inconnues non

modélisées. Il n'est pas nécessaire d'avoir des informations sur les paramètres dynamiques du robot mobile.

Des méthodes de commande robustes telles que la commande par mode glissant et des méthodes de commande adaptatives et intelligentes telles que la logique floue et les contrôleurs de réseau de neurones sont les solutions possibles à ce problème et seront expliquées dans le reste de ce chapitre.

### 1.8.3 Contrôleurs de suivi de trajectoire par la mode glissante

Une commande de suivi robuste pour un robot mobile holonome à roues est proposée dans [14], [15] et [16]. Une nouvelle loi de contrôle du glissement a été proposée dans ces travaux pour stabiliser asymptotiquement le robot mobile sur une trajectoire désirée. Une grande difficulté dans le contrôle des robots mobiles dans le monde réel est qu'il y a des incertitudes dans leur modèle. Compte tenu des caractéristiques intrinsèques des robots mobiles telles que la dynamique actuelle du véhicule, l'inertie et les limites de puissance des actionneurs et les erreurs de localisation, leurs équations dynamiques ne peuvent être décrites comme un modèle mathématique simplifié. Les avantages de l'utilisation de la commande en mode glissant pour la solution comprennent la rapidité, une bonne réponse transitoire et la robustesse par rapport aux variations des paramètres. Le modèle dynamique du robot est utilisé dans ce travail pour décrire son comportement en présence de perturbations limitées. Au moyen d'une méthode de calcul du couple, la dynamique de l'erreur du robot mobile est linéarisée et la loi de contrôle du mode glissant est appliquée pour stabiliser le robot à une trajectoire de référence et compenser les perturbations existantes. Ainsi, le schéma de commande proposé utilise la méthode du couple précalculé pour la linéarisation de l'équation dynamique et la théorie du mode de glissement pour la commande robuste.

Cette loi de commande proposée a la capacité de résoudre le problème de suivi de trajectoire sur la base d'une modélisation dynamique, le comportement du contrôleur du robot mobile est robuste contre les erreurs dans la condition initiale et les

perturbations externes telles que les erreurs d'intégration, le bruit dans le signal de commande, les erreurs de localisation, etc. [14].

Le principal problème de cette approche de contrôle est qu'elle nécessite le modèle dynamique précis du robot pour la partie du couple pré-calculée. Lorsque nous voulons effectuer la linéarisation par rétroaction du modèle dynamique non linéaire du système, nous devons connaître la dynamique exacte du robot. Cet inconvénient de cette loi de commande nous conduit à utiliser un contrôleur adaptatif ou intelligent avec capacité d'apprentissage afin que nous puissions apprendre la dynamique du robot en ligne sans le savoir pour la conception du contrôleur. De nombreuses méthodes de contrôle adaptatif et intelligent peuvent être utilisées pour résoudre ce problème.

#### 1.8.4 Contrôleurs basés sur la logique floue de suivi de trajectoire

Une commande de suivi de trajectoire pour un robot mobile à roues utilisant la commande par logique floue est proposée dans [17], [18] et [19]. Dans [17], un algorithme de contrôle basé sur les erreurs de postures des robots mobiles qui alimentent la logique floue, qui génère des signaux de correction pour les vitesses des moteurs, est proposé. La stratégie de contrôle est basée sur un profil de vitesse d'avance et le signal de correction de la vitesse généré par la logique floue en fonction des erreurs de position. Les résultats de la simulation et des expériences démontrent l'efficacité de l'algorithme proposé, qui a prouvé les bons résultats de suivi et a montré la robustesse de l'algorithme face aux incertitudes du modèle du système.

L'algorithme proposé dans [19] porte sur le développement d'un contrôleur d'algorithme de génétique floue pour un robot mobile à roues 2-DOF. Les entrées globales sont une position de référence et une vitesse de référence, qui sont des variables temporelles. La sortie globale est une posture actuelle ; la position est estimée par un algorithme d'estimation. L'algorithme d'estimation détermine la position actuelle du robot en temps réel en ajoutant les données de la position croissante à celles de la position précédente pendant la période d'échantillonnage. Le contrôleur de suivi fait

converger l'erreur de position à zéro. Afin de réduire l'erreur de position, une compensation des vitesses sur la trajectoire est nécessaire. Par conséquent, un contrôleur utilisant un algorithme flou génétique est proposé pour donner une compensation de vitesse dans ce système. Les variables d'entrée de deux contrôleurs à logique floue sont des erreurs de position à chaque instant d'échantillonnage. Les valeurs de sortie sont des vitesses de compensation. Des algorithmes génétiques sont mis en œuvre pour ajuster le gain de sortie de la logique floue.

Le contrôleur à logique floue proposé dans [18] est basé sur une approche de Backstepping pour assurer une stabilisation asymptotique de la position et de l'orientation du robot autour de la trajectoire souhaitée, en tenant compte de la cinématique et de la dynamique du véhicule. Le système d'inférence de Mamadani est utilisé pour construire le contrôleur avec neuf règles « si alors » et la méthode du centroïde de la zone comme stratégie de défuzzification où les couples et vitesses d'entrées sont considérés comme des variables linguistiques. De nombreux chercheurs ont utilisé uniquement le modèle cinématique (système de direction) pour résoudre le problème du contrôle de suivi, où la vitesse, utilisée comme contrôle d'entrée, est censée être fournie par le robot mobile dont la dynamique des actionneurs est négligée. Des prototypes réels ont actionné des roues dont le taux de glissement, le roulement, le moment d'inertie et la répartition des masses contribuent aux forces exercées sur la structure du véhicule, affectant ainsi la précision et la pleine manœuvrabilité du robot. Le modèle dynamique du robot est utilisé dans ce travail pour convertir le système de direction en véhicule réel. Les fonctions d'appartenance triangulaire et trapézoïdale sont utilisées dans cette conception, ainsi que trois partitions floues et neuf règles.

#### 1.8.5 Contrôleurs adaptatifs de suivi de trajectoire

Des méthodes de contrôle adaptatives pour le suivi de la trajectoire d'un robot mobile à roues sont proposées dans [20], [21], [22] et [23]. Dans [20] une règle de contrôle adaptatif, au niveau de la dynamique, pour les robots mobiles dont les paramètres

dynamiques sont inconnus, est proposée. Les commandes adaptatives sont dérivées pour les robots mobiles, pour le suivi d'une trajectoire de référence et la stabilisation à une position fixe. Pour le problème de suivi, le contrôleur garantit la convergence asymptotique de l'erreur de suivi vers zéro. Pour la stabilisation, le problème est converti en un problème de suivi équivalent, en utilisant une erreur de retour d'état variable dans le temps, avant que le contrôle de suivi ne soit appliqué. Le contrôleur désigné assure la mise à zéro asymptotique de l'erreur de stabilisation. Les lois de contrôle proposées comprennent un limiteur de vitesse/accélération qui empêche les roues du robot de glisser.

Un contrôleur de suivi adaptatif simple est présenté dans [22] basé sur le modèle cinématique. Un champ de potentiel artificiel est utilisé pour diriger le robot à roues dans le contrôleur. Une conception simple, une convergence rapide et une adaptabilité à d'autres mobiles sont des avantages évidents. La stabilité est prouvée par l'utilisation d'une fonction de Lyapunov. Un contrôleur dynamique adaptatif pour le suivi de trajectoire des robots mobiles holonomes est présenté dans [23]. Le contrôleur est entièrement développé en temps discret et les fonctions dynamiques non linéaires du robot sont supposées inconnues. Un réseau de neurones à fonction de base radiale gaussienne est utilisé pour l'approximation de la fonction, et ses pondérations sont estimées de manière stochastique en temps réel. La loi de contrôle proposée prend en considération l'incertitude des estimations, ce qui permet d'améliorer les performances de suivi. La méthode proposée est vérifiée par des simulations réalistes et des analyses de Monte Carlo.

#### 1.8.6 Contrôleurs de suivi de trajectoire basé sur le réseau de neurones

De nos jours, les réseaux de neurones se sont montrés d'être une approche intéressante pour résoudre des problèmes de contrôle complexes. Les contrôleurs de réseaux de neurones sont généralement basés sur la propriété d'approximation des fonctions et la capacité d'apprentissage du réseau de neurones. L'utilisation de contrôleurs de réseaux

de neurones pour le suivi de la trajectoire des robots mobiles a été proposée dans [24] et [25].

Le contrôleur de réseau neuronal proposé dans [25] est basé sur la propriété d'approximation de la fonction du réseau de neurones et peut traiter les perturbations limitées non modélisées et la dynamique non modélisée du robot mobile. Le réseau de neurones est combiné avec le contrôleur de mode glissant pour apprendre la dynamique complète du robot mobile et convertir la vitesse de sortie du contrôleur de mode glissant en une entrée de couple pour le véhicule réel. L'avantage d'avoir des réseaux de neurones dans cette approche est qu'il n'est pas nécessaire de connaître le modèle dynamique du robot et que le réseau de neurones l'apprendra en ligne sans connaissance de la dynamique.

Un contrôleur basé sur un réseau de neurones à ondelettes pour les robots mobiles est proposé dans [24]. L'ouvrage présente un schéma de commande prédictive pour les robots mobiles qui présentent une complexité, une non-linéarité et une incertitude. Un réseau neuronal à rétro propagation multicouche est utilisé comme modèle pour la dynamique non linéaire du robot. Le réseau de neurones est construit par la décomposition orthogonale en ondelettes pour former un réseau de neurones en ondelettes qui peut surmonter les problèmes causés par les minimums locaux d'optimisation. Le réseau d'ondelettes est également utile pour déterminer le nombre de nœuds cachés et la valeur initiale des pondérations.

## CHAPITRE II

### MATÉRIEL ET MÉTHODES

#### 2.1 Introduction

La première étape dans l'étude d'un robot mobile consiste à donner son modèle dans le repère global. Une fois ceci est fait, il sera possible de trouver des lois de commandes.

Dans la première partie de ce chapitre, on va présenter le modèle cinématique du robot mobile omnidirectionnel en traitant les relations entre les vitesses et les positions sans tenir compte des forces qui les affectent. Dans la deuxième partie, on va proposer une loi de commande à base d'un proportionnel-intégrateur (PI) et on va vérifier sa stabilité par la théorie de Lyapunov. Puis on va présenter le contexte général de réseau de neurones. Ensuite, on va élaborer un régulateur à base du réseau de neurones pour remplacer le régulateur à base PI. Enfin on va élaborer un régulateur en combinant les deux régulateurs précédents.

#### 2.2 Modélisation

La conception, le développement, la modification et le contrôle d'un système mécatronique nécessitent une compréhension et une représentation appropriée d'un système: plus précisément, un "modèle" du système est nécessaire. Tout modèle est une idéalisation du système réel. Un système mécatronique ou robotique peut être constitué de plusieurs types de composants différents, et il est appelé système mixte.

De cette manière, les modèles de composants peuvent être intégrés de manière pratique pour obtenir le modèle global. La modélisation d'une plateforme robotique mobile consiste en une modélisation cinématique du système. La modélisation cinématique traite des relations géométriques qui régissent le système et étudie les mathématiques du mouvement sans tenir compte des forces qui l'affectent.

### 2.2.1 Systèmes de coordonnées

La première étape de la modélisation mécanique consiste à définir les systèmes de coordonnées appropriés pour la plateforme du robot, qui seront décrits dans la section suivante.

La fonction principale des systèmes de coordonnées est de représenter la position du robot. Les deux systèmes de coordonnées suivants sont utilisés pour la modélisation et le contrôle du robot mobile :

- Repère inertiel  $\{X_I, Y_I\}$  : c'est le système de coordonnées fixe dans le plan du robot.
- Repère du robot :  $\{X_R, Y_R\}$  : c'est le système de coordonnées attaché au robot, appelé aussi repère mobile.

Les deux systèmes de coordonnées ci-dessus sont illustrés dans la Figure 2.1.

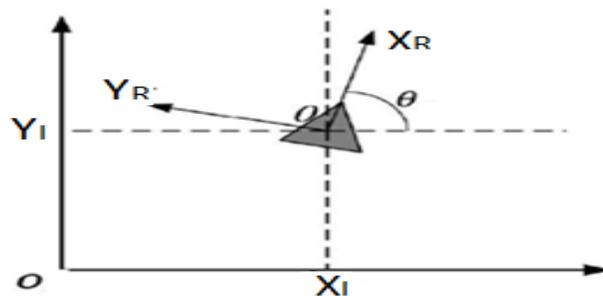


Figure 2. 1 Robot dans les repères

L'introduction de ces systèmes de coordonnées est utile pour la modélisation cinématique du robot qui sera présentée dans la section suivante. La question importante qui doit être expliquée est la correspondance entre ces deux repères. La position du robot dans le cadre inertiel et le cadre du robot peut être définie comme suit:

$$q_I = [x_I \quad y_I \quad \theta_I]^T$$

$$q_R = [x \quad y \quad \theta]^T$$

$$q_R = R(\theta) q_I \quad (2.1)$$

Avec :

$$R(\theta) = \begin{bmatrix} \cos \theta & \sin \theta \\ -\sin \theta & \cos \theta \end{bmatrix} \text{ s'appelle la matrice de transformation.}$$

### 2.2.2 Modélisation cinématique

La figure 2.2 représente les vitesses et les forces appliquées sur le robot mobile omnidirectionnel étudié dans le plan XY.

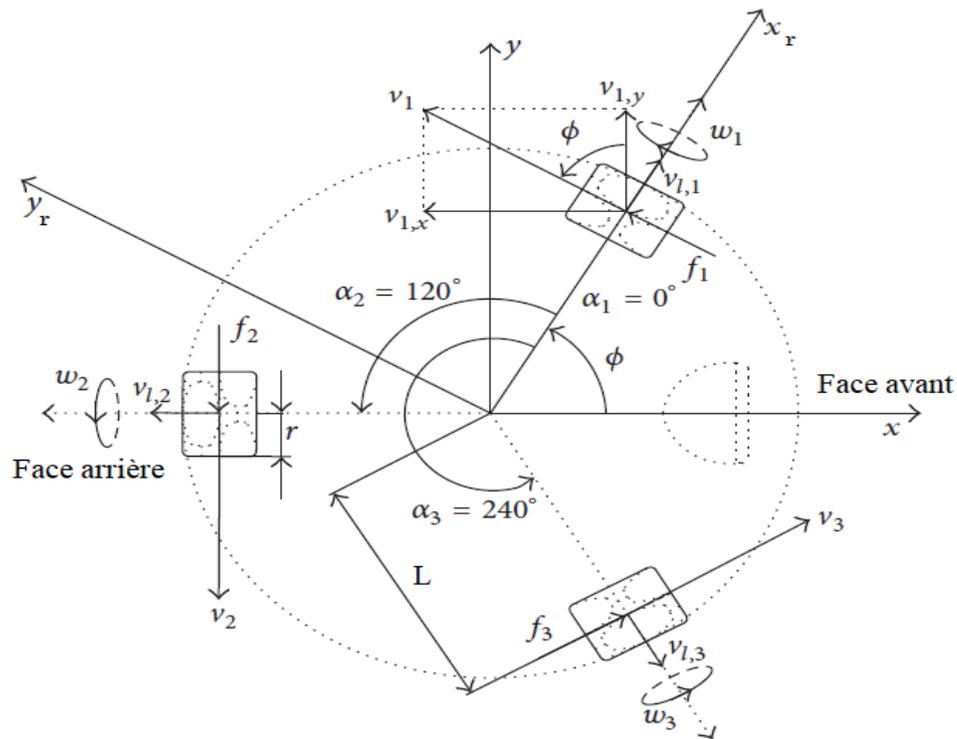


Figure 2. 2 Modélisation des vitesses du robot [16]

Avec :

$v_i$  : les vitesses de translation des roues avec :  $i = 1,2,3$

$\omega_i$  : les vitesses de rotation des roues

$v_{l,i}$  : les vitesses latérales des roues

$f_i$  : les forces motrices

$\alpha_i$  : les angles qui décrivent les positions des roues par rapport au repère mobile

$L$  : la distance entre le centre de masse du robot et chaque roue

$r$  : le rayon des roues

Maintenant, nous avons acquis une compréhension de base du robot omnidirectionnel, nous pouvons étudier la cinématique du véhicule. Si nous voulons prescrire les mouvements du robot dans l'environnement, nous devons savoir comment ces variables sont liées aux variables primaires que nous pouvons contrôler: les positions angulaires et les vitesses des arbres de roue. Il faut donc développer un modèle cinématique du robot.

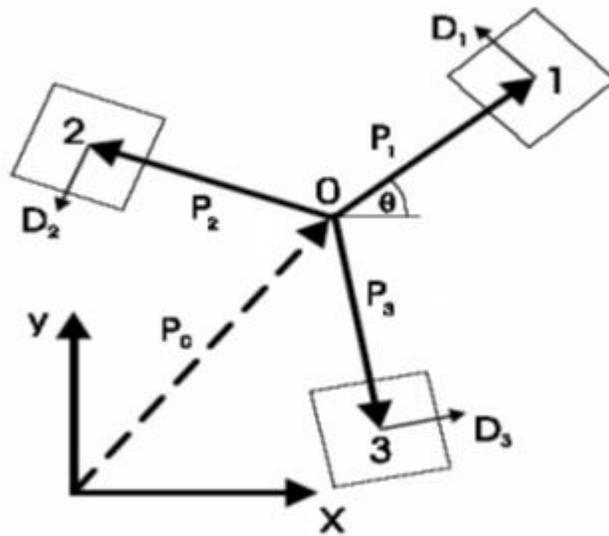


Figure 2. 3 Géométrie de la plateforme du robot mobile omnidirectionnel [16]

Soit :

$P_0$  : la position du centre de masse par rapport au repère globale avec  $P_0 = [x \ y]^t$

$P_i$  : la position de chaque roue par rapport au centre de masse du robot avec  $i = 1,2,3$

$$P_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix} = R(\theta)L \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad (2.2)$$

$$P_1 = R(\theta)L \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} L \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$P_1 = L \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$P_2 = R\left(\frac{2\pi}{3}\right) L \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} \frac{-1}{2} & \frac{-\sqrt{3}}{2} \\ \frac{\sqrt{3}}{2} & \frac{-1}{2} \end{bmatrix} L \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$P_2 = \frac{L}{2} \begin{bmatrix} -1 \\ \sqrt{3} \end{bmatrix}$$

$$P_3 = R\left(\frac{4\pi}{3}\right) L \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} \frac{-1}{2} & \frac{\sqrt{3}}{2} \\ \frac{-\sqrt{3}}{2} & \frac{-1}{2} \end{bmatrix} L \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$P_3 = \frac{-L}{2} \begin{bmatrix} 1 \\ \sqrt{3} \end{bmatrix}$$

Soit  $D_i$  : le vecteur unitaire normal de chaque roue, il représente la direction de translation avec  $i = 1,2,3$ .

$$D_i = \frac{1}{L} R\left(\frac{\pi}{2}\right) P_i \quad (2.3)$$

$$\text{Et } R\left(\frac{\pi}{2}\right) = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$$

$$D_1 = \frac{1}{L} R\left(\frac{\pi}{2}\right) P_1 = \frac{1}{L} \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} L \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$D_1 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$D_2 = \frac{1}{L} R\left(\frac{\pi}{2}\right) P_2 = \frac{1}{L} \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \frac{L}{2} \begin{bmatrix} -1 \\ \sqrt{3} \end{bmatrix}$$

$$D_2 = \frac{-1}{2} \begin{bmatrix} \sqrt{3} \\ 1 \end{bmatrix}$$

$$D_3 = \frac{1}{L} R\left(\frac{\pi}{2}\right) P_3 = \frac{1}{L} \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \frac{-L}{2} \begin{bmatrix} 1 \\ \sqrt{3} \end{bmatrix}$$

$$D_3 = \frac{1}{2} \begin{bmatrix} \sqrt{3} \\ -1 \end{bmatrix}$$

Soit  $r_i$  : la position de chaque roue par rapport au repère globale avec  $i = 1,2,3$ .

$$r_i = P_0 R(\gamma_i) P_i \quad (2.4)$$

$$\text{Avec : } \gamma_i = \theta + \alpha_i \text{ et } \begin{cases} \alpha_1 = 0 \\ \alpha_2 = \frac{2\pi}{3} \\ \alpha_3 = \frac{4\pi}{3} \end{cases} \text{ et } \begin{cases} \gamma_1 = \theta \\ \gamma_2 = \theta + \frac{2\pi}{3} \\ \gamma_3 = \theta + \frac{4\pi}{3} \end{cases}$$

La vitesse de chaque roue (vitesse de point) par rapport au repère globale est donnée par:

$$v_i = \dot{P}_0 + \dot{R}(\gamma_i) P_i \quad (2.5)$$

La vitesse de translation de chaque roue par rapport au repère globale est donnée par:

$$V_i = (\dot{P}_0 + \dot{R}(\gamma_i) P_i)^t (R(\gamma_i) D_i)$$

$$V_i = \dot{P}_0^t + P_i^t \dot{R}(\gamma_i) (R(\gamma_i) D_i)$$

$$V_i = \dot{P}_0^t R(\gamma_i) D_i + P_i^t \dot{R}^t(\gamma_i) R(\gamma_i) D_i$$

Avec :  $P_i^t \dot{R}^t(\gamma_i) R(\gamma_i) D_i = L \dot{\theta}$  représente la vitesse angulaire du robot

$$V_i = \dot{P}_0^t R(\gamma_i) D_i + L \dot{\theta} \quad (2.6)$$

$$P_0 = \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix}; \dot{P}_0 = \begin{bmatrix} \ddot{x} \\ \ddot{y} \end{bmatrix}; \dot{P}_0^t = [\dot{x} \quad \dot{y}]$$

$$V_1 = \dot{P}_0^t R(\gamma_1) D_1 + L \dot{\theta}$$

$$V_1 = \dot{P}_0^t R(\theta) D_1 + L \dot{\theta}$$

$$V_1 = -\sin \theta \dot{x} + \cos \theta \dot{y} + L \dot{\theta}$$

$$V_2 = \dot{P}_0^t R(\gamma_2) D_2 + L \dot{\theta}$$

$$V_2 = \dot{P}_0^t R\left(\theta + \frac{2\pi}{3}\right) D_2 + L \dot{\theta}$$

$$V_2 = \left(\frac{-\sqrt{3}}{2} \cos \theta + \frac{1}{2} \sin \theta\right) \dot{x} - \left(\frac{1}{2} \cos \theta + \frac{\sqrt{3}}{2} \sin \theta\right) \dot{y} + L \dot{\theta}$$

$$V_2 = -\sin\left(\frac{\pi}{3} - \theta\right) \dot{x} - \cos\left(\frac{\pi}{3} - \theta\right) \dot{y} + L \dot{\theta}$$

$$V_3 = \dot{P}_0^t R(\gamma_3) D_3 + L \dot{\theta}$$

$$V_3 = \dot{P}_0^t R\left(\theta + \frac{4\pi}{3}\right) D_3 + L \dot{\theta}$$

$$V_3 = \left(\frac{\sqrt{3}}{2} \cos \theta + \frac{1}{2} \sin \theta\right) \dot{x} + \left(\frac{1}{2} \cos \theta - \frac{\sqrt{3}}{2} \sin \theta\right) \dot{y} + L \dot{\theta}$$

$$V_3 = \sin\left(\frac{\pi}{3} + \theta\right) \dot{x} + \cos\left(\frac{\pi}{3} + \theta\right) \dot{y} + L \dot{\theta}$$

$$\begin{bmatrix} V_1 \\ V_2 \\ V_3 \end{bmatrix} = \begin{bmatrix} -\sin \theta & \cos \theta & L \\ -\sin\left(\frac{\pi}{3} - \theta\right) & \cos\left(\frac{\pi}{3} - \theta\right) & L \\ \sin\left(\frac{\pi}{3} + \theta\right) & -\cos\left(\frac{\pi}{3} + \theta\right) & L \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} \quad (2.7)$$

Avec:  $V_i = r \omega_i$        $\omega_i$ : la vitesse angulaire de la roue

$$r_1 = r_2 = r_3 = r$$

$$V = \begin{bmatrix} V_1 \\ V_2 \\ V_2 \end{bmatrix} = P(\theta) \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = r \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{bmatrix} \quad (2.8)$$

Avec :

$$P(\theta) = \begin{bmatrix} -\sin \theta & \cos \theta & L \\ -\sin\left(\frac{\pi}{3} - \theta\right) & \cos\left(\frac{\pi}{3} - \theta\right) & L \\ \sin\left(\frac{\pi}{3} + \theta\right) & -\cos\left(\frac{\pi}{3} + \theta\right) & L \end{bmatrix}$$

Donc :

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = P^{-1}(\theta) \begin{bmatrix} V_1 \\ V_2 \\ V_3 \end{bmatrix} \quad (2.9)$$

D'après l'équation (2.1) on a :

$$\dot{q} = \dot{q}_R = R(\theta) \dot{q}_I \quad (2.10)$$

Soit  $U = \dot{q}_I = [V_x \ V_y \ \omega]^T$  le vecteur de vitesse de commande

Avec :

$V_x$  : représente la vitesse linéaire du robot suivant x

$V_y$  : représente la vitesse linéaire du robot suivant y

$\omega$  : représente la vitesse angulaire du robot

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \dot{x}_I \\ \dot{y}_I \\ \dot{\theta}_I \end{pmatrix}$$

Le modèle cinématique du robot mobile omnidirectionnel est donné par la formule suivante:

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} V_x \\ V_y \\ \omega \end{pmatrix} \quad (2.11)$$

D'après (2.1) et (2.11) la relation entre les vitesses des roues  $V$  et les vitesses de commandes  $U$  est donnée par la formule suivante:

$$V = \begin{pmatrix} -\sin \alpha_1 & \cos \alpha_1 & L \\ -\sin \alpha_2 & \cos \alpha_2 & L \\ -\sin \alpha_3 & \cos \alpha_3 & L \end{pmatrix} U \quad (2.12)$$

## 2.3 Conception des régulateurs

### 2.3.1 Régulateur à base PI

Soit :  $[x_d(t) \ y_d(t) \ \theta_d(t)]^t$  la trajectoire désirée.

L'erreur de suivi de trajectoire (erreur de position) est donnée par la formule suivante:

$$e = \begin{bmatrix} x_e(t) \\ y_e(t) \\ \theta_e(t) \end{bmatrix} = \begin{bmatrix} x(t) \\ y(t) \\ \theta(t) \end{bmatrix} - \begin{bmatrix} x_d(t) \\ y_d(t) \\ \theta_d(t) \end{bmatrix} \quad (2.13)$$

La dérivée de l'erreur est:

$$\dot{e} = \begin{bmatrix} \dot{x}_e \\ \dot{y}_e \\ \dot{\theta}_e \end{bmatrix} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} - \begin{bmatrix} \dot{x}_d \\ \dot{y}_d \\ \dot{\theta}_d \end{bmatrix} = P^{-1}(\theta) \begin{bmatrix} V_1 \\ V_2 \\ V_3 \end{bmatrix} - \begin{bmatrix} \dot{x}_d \\ \dot{y}_d \\ \dot{\theta}_d \end{bmatrix} \quad (2.14)$$

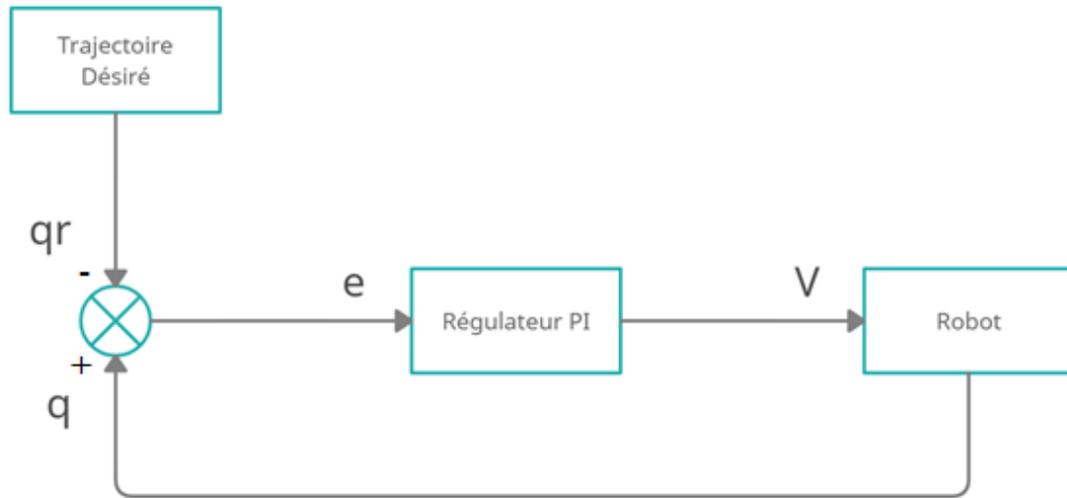


Figure 2. 4 Structure du régulateur à base PI

De même, l'objectif de contrôle est de trouver un ensemble de vecteurs de vitesse de contrôle  $V = [V_1 \ V_2 \ V_3]^t$  de sorte que le système d'erreur en boucle fermée est asymptotiquement stable.

On propose:

$$\begin{bmatrix} V_1 \\ V_2 \\ V_3 \end{bmatrix} = P(\theta) \left( -K_P \begin{bmatrix} x_e \\ y_e \\ \theta_e \end{bmatrix} - K_I \begin{bmatrix} \int x_e dt \\ \int y_e dt \\ \int \theta_e dt \end{bmatrix} + \begin{bmatrix} \dot{x}_d \\ \dot{y}_d \\ \dot{\theta}_d \end{bmatrix} \right) \quad (2.15)$$

$$\text{Soit: } S = K_P \begin{bmatrix} x_e \\ y_e \\ \theta_e \end{bmatrix} + K_I \begin{bmatrix} \int x_e dt \\ \int y_e dt \\ \int \theta_e dt \end{bmatrix}$$

Donc :

$$V = P(\theta) \left( -S + \begin{bmatrix} \dot{x}_d \\ \dot{y}_d \\ \dot{\theta}_d \end{bmatrix} \right) \quad (2.16)$$

Qui conduit au système d'erreur en boucle fermée fondamental:

$$\begin{bmatrix} \dot{x}_e \\ \dot{y}_e \\ \dot{\theta}_e \end{bmatrix} = P^{-1}(\theta) \begin{bmatrix} V_1 \\ V_2 \\ V_3 \end{bmatrix} - \begin{bmatrix} \dot{x}_d \\ \dot{y}_d \\ \dot{\theta}_d \end{bmatrix}$$

$$\begin{bmatrix} \dot{x}_e \\ \dot{y}_e \\ \dot{\theta}_e \end{bmatrix} = P^{-1}(\theta) P(\theta) \left( -K_P \begin{bmatrix} x_e \\ y_e \\ \theta_e \end{bmatrix} - K_I \begin{bmatrix} \int x_e dt \\ \int y_e dt \\ \int \theta_e dt \end{bmatrix} + \begin{bmatrix} \dot{x}_d \\ \dot{y}_d \\ \dot{\theta}_d \end{bmatrix} \right) - \begin{bmatrix} \dot{x}_d \\ \dot{y}_d \\ \dot{\theta}_d \end{bmatrix}$$

$$\begin{bmatrix} \dot{x}_e \\ \dot{y}_e \\ \dot{\theta}_e \end{bmatrix} = -K_P \begin{bmatrix} x_e \\ y_e \\ \theta_e \end{bmatrix} - K_I \begin{bmatrix} \int x_e dt \\ \int y_e dt \\ \int \theta_e dt \end{bmatrix} + \begin{bmatrix} \dot{x}_d \\ \dot{y}_d \\ \dot{\theta}_d \end{bmatrix} - \begin{bmatrix} \dot{x}_d \\ \dot{y}_d \\ \dot{\theta}_d \end{bmatrix}$$

$$\begin{bmatrix} \dot{x}_e \\ \dot{y}_e \\ \dot{\theta}_e \end{bmatrix} = -K_P \begin{bmatrix} x_e \\ y_e \\ \theta_e \end{bmatrix} - K_I \begin{bmatrix} \int x_e dt \\ \int y_e dt \\ \int \theta_e dt \end{bmatrix} \quad (2.17)$$

Maintenant, il faut vérifier la stabilité de régulateur proposé.

Soit la fonction  $V_L$  avec :

$$V_L = \frac{1}{2} \begin{bmatrix} x_e & y_e & \theta_e \end{bmatrix} \begin{bmatrix} x_e \\ y_e \\ \theta_e \end{bmatrix} + \frac{1}{2} \begin{bmatrix} \int x_e dt & \int y_e dt & \int \theta_e dt \end{bmatrix} K_I \begin{bmatrix} \int x_e dt \\ \int y_e dt \\ \int \theta_e dt \end{bmatrix} \quad (2.18)$$

$$V_L = \frac{1}{2} (x_e^2 + y_e^2 + \theta_e^2) + \frac{1}{2} K_I \left( \left( \int x_e dt \right)^2 + \left( \int y_e dt \right)^2 + \left( \int \theta_e dt \right)^2 \right)$$

Donc  $\forall t > 0, V_L > 0$

La dérivée de  $V_L$  est :

$$\begin{aligned} \dot{V}_L &= \frac{1}{2} \left( \begin{bmatrix} \dot{x}_e & \dot{y}_e & \dot{\theta}_e \end{bmatrix} \begin{bmatrix} x_e \\ y_e \\ \theta_e \end{bmatrix} + \begin{bmatrix} x_e & y_e & \theta_e \end{bmatrix} \begin{bmatrix} \dot{x}_e \\ \dot{y}_e \\ \dot{\theta}_e \end{bmatrix} \right) \\ &+ \frac{1}{2} K_I \left( \begin{bmatrix} x_e & y_e & \theta_e \end{bmatrix} \begin{bmatrix} \int x_e dt \\ \int y_e dt \\ \int \theta_e dt \end{bmatrix} + \begin{bmatrix} \int x_e dt & \int y_e dt & \int \theta_e dt \end{bmatrix} \begin{bmatrix} x_e \\ y_e \\ \theta_e \end{bmatrix} \right) \end{aligned}$$

De même :

$$\begin{aligned}
& \dot{V}_L \\
&= \frac{1}{2} \left( -K_p [x_e \quad y_e \quad \theta_e] - K_I \left[ \int x_e dt \quad \int y_e dt \quad \int \theta_e dt \right] \right) \begin{bmatrix} x_e \\ y_e \\ \theta_e \end{bmatrix} \\
&+ [x_e \quad y_e \quad \theta_e] \left( -K_p \begin{bmatrix} x_e \\ y_e \\ \theta_e \end{bmatrix} - K_I \begin{bmatrix} \int x_e dt \\ \int y_e dt \\ \int \theta_e dt \end{bmatrix} \right) \\
&+ \frac{1}{2} K_I \left( [x_e \quad y_e \quad \theta_e] \begin{bmatrix} \int x_e dt \\ \int y_e dt \\ \int \theta_e dt \end{bmatrix} + \left[ \int x_e dt \quad \int y_e dt \quad \int \theta_e dt \right] \begin{bmatrix} x_e \\ y_e \\ \theta_e \end{bmatrix} \right)
\end{aligned}$$

$$\begin{aligned}
& \dot{V}_L \\
&= -\frac{1}{2} K_p [x_e \quad y_e \quad \theta_e] \begin{bmatrix} x_e \\ y_e \\ \theta_e \end{bmatrix} - \frac{1}{2} K_I \left[ \int x_e dt \quad \int y_e dt \quad \int \theta_e dt \right] \begin{bmatrix} x_e \\ y_e \\ \theta_e \end{bmatrix} \\
&- \frac{1}{2} K_p [x_e \quad y_e \quad \theta_e] \begin{bmatrix} x_e \\ y_e \\ \theta_e \end{bmatrix} - \frac{1}{2} K_I [x_e \quad y_e \quad \theta_e] \begin{bmatrix} \int x_e dt \\ \int y_e dt \\ \int \theta_e dt \end{bmatrix} \\
&+ \frac{1}{2} K_I [x_e \quad y_e \quad \theta_e] \begin{bmatrix} \int x_e dt \\ \int y_e dt \\ \int \theta_e dt \end{bmatrix} + \frac{1}{2} K_I \left[ \int x_e dt \quad \int y_e dt \quad \int \theta_e dt \right] \begin{bmatrix} x_e \\ y_e \\ \theta_e \end{bmatrix}
\end{aligned}$$

$$\dot{V}_L = -K_P [x_e \quad y_e \quad \theta_e] \begin{bmatrix} x_e \\ y_e \\ \theta_e \end{bmatrix}$$

$$\dot{V}_L = -K_P (x_e^2 + y_e^2 + \theta_e^2)$$

Donc  $\forall t > 0, \dot{V}_L < 0$

Donc  $V_L$  est une fonction de Lyapunov.

La stabilité asymptotique du système d'erreur en boucle fermée peut être obtenue en sélectionnant la fonction quadratique de Lyapunov  $V_L$ .

### 2.3.2 Régulateurs de réseau de neurones :

#### 2.3.2.1 Contexte de réseaux de neurones

Avant de commencer à expliquer les détails de la conception d'un contrôleur à réseau de neurones pour le suivi de trajectoire de ce système du robot mobile, nous présentons dans la section suivante un bref aperçu des réseaux de neurones (RN), incluant les différentes topologies, propriétés, sélection et formation des poids, ainsi que l'application des réseaux de neurones aux problèmes de contrôle.

Les RN artificiels sont modélisés sur les processus biologiques de traitement de l'information, notamment le système nerveux et son unité de base, le neurone. Les signaux sont propagés sous la forme de différences de potentiel entre l'intérieur et l'extérieur des cellules. Le modèle mathématique d'un neurone est représenté dans la figure suivante :

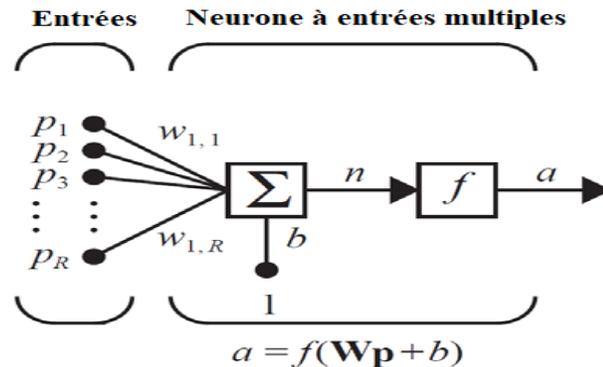


Figure 2. 5 Modèle mathématique d'un neurone [29]

Les poids d'entrées  $w_j$ , le seuil de déclenchement  $b$  (aussi appelé le bias), la somme des entrées pondérées et la fonction d'activation non linéaire  $f$  sont illustrées dans la figure ci-dessus. Si les entrées de la cellule sont  $n$  signaux à l'instant  $k$ ,  $x_1(k), x_2(k), x_3(k), \dots, x_n(k)$ , et la sortie est le scalaire  $y(k)$ , l'équation mathématique du neurone peut être écrite comme suit:

$$y(k) = f\left(\sum_{j=1}^n w_j x_j(k) + b\right) \quad (2.19)$$

Les fonctions d'activation sont sélectionnées en fonction de l'application, mais certains choix fréquents sont les suivants :

- Linéaire :  $y = x$
- Sigmoidé :  $y = \frac{1}{1+e^{-x}}$
- Sigmoidé symétrique :  $y = \frac{1-e^{-x}}{1+e^{-x}}$
- Tangente hyperbolique :  $y = \frac{e^x - e^{-x}}{e^x + e^{-x}}$

L'intention de la fonction d'activation est de modéliser le comportement non linéaire de la cellule où il n'y a pas de sortie en dessous d'une certaine valeur de l'argument. Les

fonctions sigmoïdes sont une classe générale de fonctions monotones non décroissantes prenant des valeurs bornées. Pour de nombreux algorithmes de formation de RN, y compris l'algorithme de rétropropagation, la dérivée de la fonction d'activation est nécessaire, la fonction d'activation sélectionnée doit donc être différentiable. La Figure 2.6 illustre un réseau neuronal à une couche composé de  $s$  cellules:

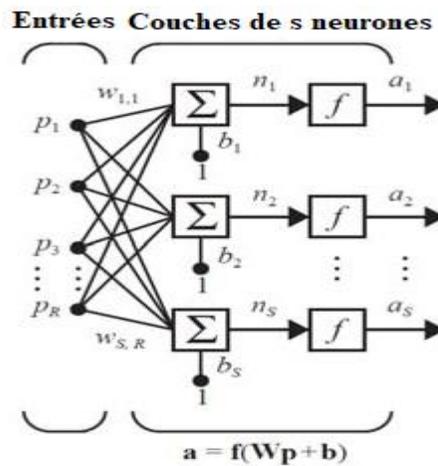


Figure 2. 6 Réseau de neurones à une couche avec  $s$  neurones [29]

L'équation de ce réseau est donnée par:

$$y_l(k) = f\left(\sum_{j=1}^n w_{lj} x_j(k) + b_l\right) \quad (2.20)$$

Il est pratique d'écrire les poids et les seuils sous forme de matrice et de vecteur. En définissant la matrice des poids et des seuils comme suit:

$$W^T = \begin{bmatrix} w_{11} & w_{12} & \dots & w_{1n} \\ w_{21} & w_{22} & \dots & w_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{L1} & w_{L2} & \dots & w_{Ln} \end{bmatrix} \quad (2.21)$$

$$b_w = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_L \end{bmatrix} \quad (2.22)$$

Nous pouvons écrire le vecteur de sortie  $y(t) = [y_0 \ y_1 \ y_2 \ \dots \ y_n]^T$  comme suit :

$$y(t) = f(W^T x + b_w) \quad (2.23)$$

Un RN à deux couches qui comporte deux couches de neurones, une couche comportant  $L$  neurones alimentant la deuxième couche comportant  $m$  neurones, est illustré dans la figure ci-dessous.

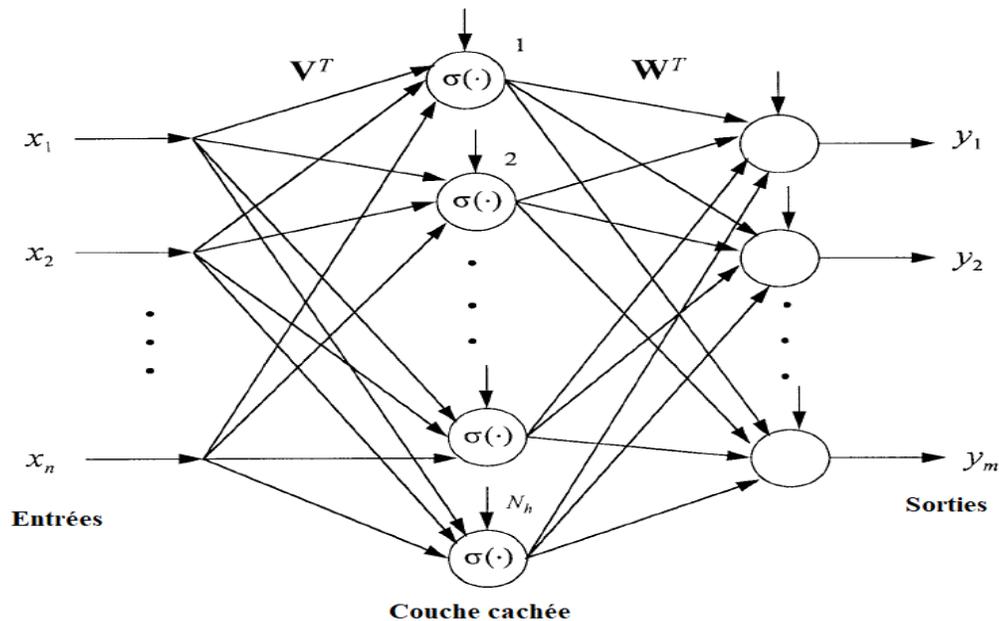


Figure 2. 7 Un réseau de neurones à deux couches [29]

La première couche est appelée la couche cachée et la deuxième couche est appelée la couche de sortie. Un réseau de neurones à plusieurs couches s'appelle un perceptron multicouche et sa puissance de calcul est nettement supérieure à celle d'un réseau à une couche. La sortie du réseau à deux couches ci-dessus est donnée par l'équation suivante:

$$y_i = \sigma \left( \sum_{l=1}^L w_{il} \sigma \left( \sum_{j=1}^n v_{lj} x_j + v_{l0} \right) + w_{i0} \right) \quad (2.24)$$

Si nous définissons la sortie de la couche cachée comme  $z_l$ , nous pouvons écrire l'équation ci-dessus comme suit:

$$z_l = \sigma \left( \sum_{j=1}^n v_{lj} x_j + v_{l0} \right) \quad l = 1, 2, \dots, L \quad (2.25)$$

$$y_i = \sigma \left( \sum_{l=1}^L w_{il} z_l + w_{i0} \right) \quad i = 1, 2, \dots, m \quad (2.26)$$

Les deux équations ci-dessus peuvent être écrites sous forme de matrice. Il est important de mentionner que les poids de la couche d'entrée à la couche cachée seront sélectionnés de manière aléatoire tandis que les poids de la couche cachée à la couche de sortie seront réglés à l'aide de méthodes d'apprentissage appropriées. Cela minimisera la complexité de calcul de l'utilisation de RN dans les applications de contrôle.

### 2.3.2.2 Propriétés des réseaux de neurones

Les réseaux neuronaux sont des systèmes distribués non linéaires complexes, et par conséquent, ils ont un large éventail d'applications. Deux des propriétés les plus importantes des réseaux de neurones sont : la classification (pour la reconnaissance des formes) et l'approximation de fonctions. Une propriété fondamentale dans les applications de contrôle en boucle fermée des réseaux de neurones est la propriété d'approximation de fonction universelle des réseaux de neurones ayant au moins deux couches. (Les RN à une couche n'ont pas la capacité d'approximation universelle des fonctions).

La théorie de base de l'approximation universelle dit que toute fonction lisse  $f(x)$  peut être approximé de façon arbitraire sur un ensemble compact en utilisant un RN à deux couches avec des poids appropriés. Spécifiquement, soit  $f(x)$  une fonction générale lisse. Ensuite, étant donné un ensemble compact  $S$  et un nombre positif  $\epsilon_N$ , il existe un RN à deux couches tel que:

$$f(x) = W^T \sigma(V^T x) + \epsilon \quad (2.27)$$

Avec  $\|\epsilon\| = \epsilon_N$  pour tout  $x$  dans l'ensemble compact  $S$ , pour une valeur  $L$  de neurones de couche cachée suffisamment grande. La valeur  $\epsilon$  est appelée l'erreur d'approximation de la fonction RN et elle diminue lorsque le nombre de neurones de la couche cachée  $L$  augmente. D'autre part, sur l'ensemble compact  $S$ , lorsque  $S$  devient plus grand, le  $L$  requis augmente généralement en conséquence. Le réseau neuronal agissant comme un approximateur de fonction est illustré à la Figure 2.8 :

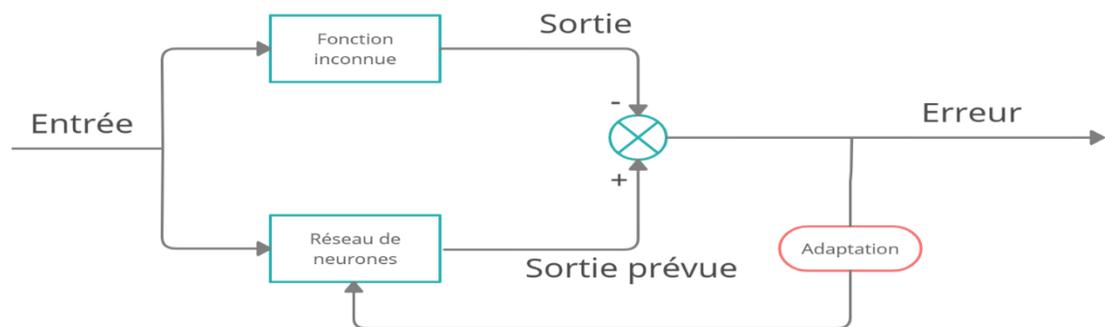


Figure 2. 8 Structure d'approximation des fonctions pour les réseaux de neurones

Même si le théorème ci-dessus dit qu'il existe un RN qui approxime toute fonction  $f(x)$ , il faut noter qu'il ne montre pas comment déterminer les poids requis. En fait, il n'est pas facile de déterminer les poids de façon qu'un RN approxime effectivement une fonction donnée  $f(x)$  d'assez près. Dans la section suivante, nous montrerons comment y parvenir en utilisant l'accord par rétropropagation.

### 2.3.2.3 Entraînement du réseau de neurones.

Comme il a été mentionné dans la section précédente, pour qu'un RN fonctionne comme souhaité, il est nécessaire de déterminer des poids et des seuils appropriés qui garantissent les performances souhaitées. Pendant des années, cela a été un problème spécifique pour un RN multicouche, où l'on ne savait pas relier la part de l'erreur aux poids des différentes couches. Aujourd'hui, ces problèmes ont été en grande partie résolus et il existe de très bons algorithmes pour la sélection et le réglage des poids des RN. L'algorithme de formation par rétropropagation est l'un des algorithmes les plus connus en raison de sa simplicité et de sa puissance. La dérivation de cette méthode est basée sur la méthode du gradient décent qui n'est pas incluse dans ce texte. L'algorithme de rétropropagation pour un réseau de neurones à deux couches illustrées à la Figure 2.9 utilisant une fonction d'activation sigmoïde est expliqué comme suit:

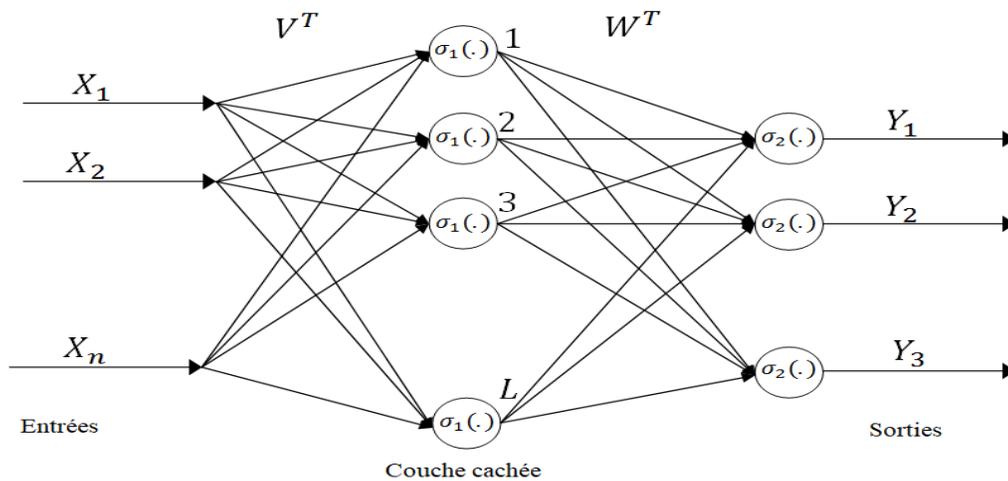


Figure 2. 9 Réseau de neurones à deux couches [29]

La sortie du réseau de neurones à deux couches ci-dessus est donnée par l'équation suivante:

$$y_i = \sigma\left(\sum_{l=1}^L W_{il} \sigma\left(\sum_{j=1}^n v_{lj} x_j + v_{l0}\right) + w_{i0}\right) \quad i = 1, 2, \dots, m$$

La dérivation est largement simplifiée par la définition de certains paramètres intermédiaires. Dans la Figure 2.9 on peut appeler la couche des pondérations  $V_{lj}$  la première couche et la couche des pondérations  $W_{il}$  la deuxième couche. L'entrée de la première couche est  $x_j$ . On définit l'entrée de la deuxième couche comme:

$$z_l = \sigma\left(\sum_{j=1}^L v_{lj}x_j + v_{l0}\right) \quad l = 1, 2, \dots, L$$

Les seuils peuvent être traités plus facilement en définissant  $x_0 = 1$  et  $z_0 = 1$ . Alors on peut dire:

$$y_i = \sigma\left(\sum_{l=1}^n w_{il}z_l\right)$$

$$z_l = \sigma\left(\sum_{j=1}^n v_{lj}x_j\right)$$

On définit la sortie de la première et la deuxième couche respectivement comme:

$$u_i^2 = \sum_{l=0}^L w_{il}z_l$$

$$u_l^2 = \sum_{j=0}^n v_{lj}x_j$$

Alors on peut écrire :

$$y_i = \sigma(u_i^2)$$

$$z_l = \sigma(u_l^1)$$

En dérivant l'algorithme de rétropropagation, nous aurons l'occasion de différencier les fonctions d'activation. On note donc que:

$$\frac{\partial y_i}{\partial W_{il}} = \sigma'(u_i^2) z_l$$

$$\frac{\partial y_i}{\partial z_l} = \sigma'(u_i^2) W_{il}$$

$$\frac{\partial z_l}{\partial V_{lj}} = \sigma'(u_l^1) x_j$$

$$\frac{\partial z_l}{\partial x_j} = \sigma'(u_l^1) V_{lj}$$

Avec  $\sigma'()$  est le dérivé de la fonction d'activation. L'algorithme de rétropropagation est un algorithme d'ajustement de la pondération basé sur la méthode de la descente gradient:

$$W_{il}(k+1) = W_{il}(k) - \eta \frac{\partial E(k)}{\partial W_{il}}$$

$$V_{lj}(k+1) = V_{lj}(k) - \eta \frac{\partial E(k)}{\partial V_{lj}}$$

Soit  $X$  un vecteur d'entrée et  $Y$  un vecteur de sortie souhaité associé pour le réseau. On définit le moindre carré de l'erreur de sortie du réseau de neurones comme :

$$E(k) = \frac{1}{2} e^T(k) e(k) = \frac{1}{2} \sum_{l=1}^m e_l(k)^2$$

$$e_i(k) = Y_i(k) - y_i(k)$$

Avec  $y_i(k)$  est calculé à l'aide de l'équation de propagation directe du réseau de neurones avec les composantes du modèle d'entrée  $X_j$  comme les entrées  $x_i(k)$ .

Le gradient requis  $E(k)$  peut être déterminé facilement en utilisant la règle de chaîne.

En particulier pour les gains de la deuxième couche on a:

$$\frac{\partial E}{\partial W_{il}} = \frac{\partial E}{\partial u_i^2} \times \frac{\partial u_i^2}{\partial W_{il}} = \left[ \frac{\partial E}{\partial e_i} \times \frac{\partial e_i}{\partial y_i} \times \frac{\partial y_i}{\partial u_i^2} \right] \times \frac{\partial u_i^2}{\partial W_{il}}$$

En utilisant les équations ci-dessus, on a:

$$\frac{\partial E}{\partial u_i^2} = -\sigma'(u_i^2)e_i$$

$$\frac{\partial E}{\partial W_{il}} = -z_l \sigma'(u_i^2)e_i$$

De même pour les poids de la première couche, on a:

$$\frac{\partial E}{\partial V_{lj}} = \frac{\partial E}{\partial u_l^1} \times \frac{\partial u_l^1}{\partial V_{lj}} = \left[ \sum_{i=1}^m \frac{\partial E}{\partial u_i^2} \times \frac{\partial u_i^2}{\partial z_l} \times \frac{\partial z_l}{\partial u_l^1} \right] \times \frac{\partial u_l^1}{\partial V_{lj}}$$

Donc on a:

$$\frac{\partial E}{\partial V_{lj}} = \left[ \sum_{i=1}^m (-\sigma'(u_i^2)e_i) \times W_{il} \times \sigma'(u_l^1) \right] \times x_j$$

$$\frac{\partial E}{\partial V_{lj}} = -x_j \sigma'(u_l^1) \sum_{i=1}^m (-\sigma'(u_i^2)e_i) \times W_{il}$$

Les équations ci-dessus peuvent être considérablement simplifiées en introduisant la notation d'une récurion à rebours à travers le réseau. Ainsi, on définit l'erreur propagée à rebours pour les couches deux et une, respectivement comme:

$$\delta_i^2 = -\frac{\partial E}{\partial u_i^2} = \sigma(u_i^2)e_i$$

$$\delta_i^1 = -\frac{\partial E}{\partial u_i^1} = \sigma'(u_i^1) \sum_{i=1}^m \delta_i^2 \times W_{il}$$

En supposant que les fonctions d'activation sigmoïde utilisent les erreurs retro propagées peuvent être calculées comme:

$$\delta_i^2 = y_i(1 - y_i)e_i$$

$$\delta_i^1 = z_i(1 - z_i) \sum_{i=1}^m \delta_i^2 \times W_{il}$$

En combinant les équations ci-dessus, on peut écrire les formules de l'algorithme de rétropropagation comme suit:

$$W_{il}(k+1) = W_{il}(k) - \eta z_i \delta_i^2$$

$$V_{ij}(k+1) = V_{ij}(k) - \eta x_j \delta_i^1$$

Dans de nombreuses applications, le réseau de neurones n'a pas de fonction d'activation dans la couche de sortie. Il faut alors utiliser simplement  $\delta_i^2 = e_i$  dans les équations de la rétropropagation.

Les formules générales et les procédures de rétropropagation sont indiquées dans le diagramme de la Figure 2.10.

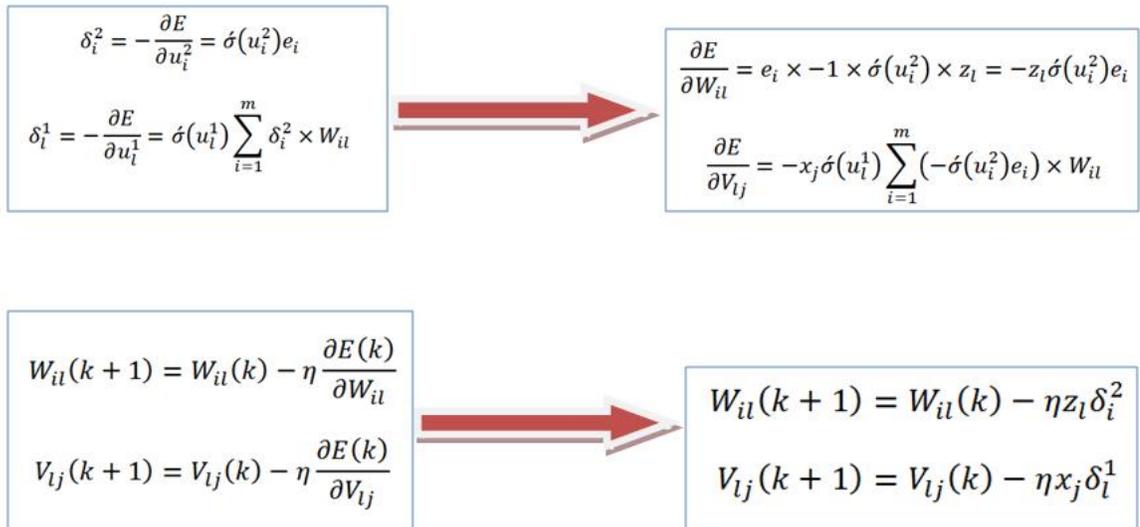


Figure 2. 10 Schéma fonctionnel de l'algorithme de rétropropagation

Une version améliorée de l'algorithme ci-dessus, qui est basée sur la méthode du gradient décent, est donnée par l'algorithme du gradient de Momentum:

$$V(k) = \beta V(k) + \eta(1 - \beta) X e^T(k)$$

Avec le paramètre de Momentum  $\beta < 1$  et un taux d'apprentissage  $\eta < 1$ .  $\beta$  est généralement sélectionné près de 1. Le Momentum ajoute un effet de mémoire afin que le réseau de neurones réponde non seulement au gradient local, mais aussi aux tendances récentes de la surface d'erreur. Sans Momentum, le réseau risque de rester bloqué dans les minimums locaux, mais l'ajout d'un Momentum peut aider le réseau de neurones à passer les minimums locaux. L'algorithme ci-dessus est la méthode que nous utilisons pour entraîner le réseau de neurones requis dans l'application du contrôleur de suivi de la trajectoire du robot mobile.

### 2.3.2.4 Développement et mise en œuvre de réseaux de neurones

Afin de construire un réseau de neurones pour l'approximation d'une fonction générale à entrées multiples et sorties multiples, il faut disposer d'un algorithme pour initialiser le réseau de neurones, utiliser l'algorithme de rétropropagation pour entraîner le réseau et utiliser l'erreur d'approximation du réseau de neurones pour montrer ses performances.

L'algorithme utilisé pour construire le réseau de neurones et le mettre en œuvre au moyen d'un progiciel (software) approprié pour l'approximation de la fonction est illustré sur la Figure 2.11:

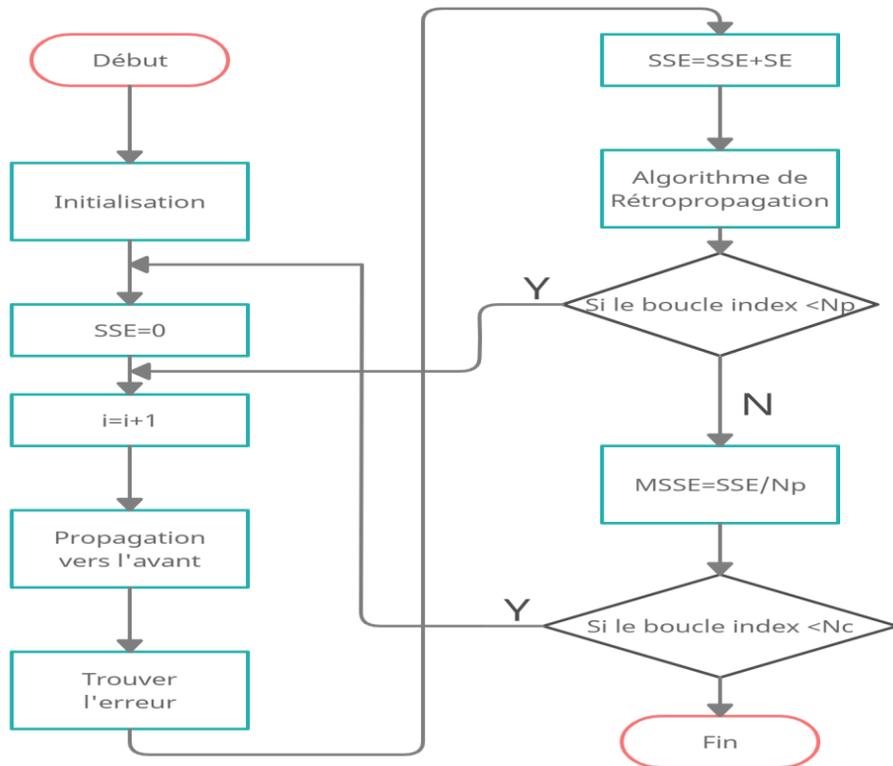


Figure 2. 11 Algorithme général de mise en œuvre de la formation des réseaux de neurones par rétropropagation

Différents blocs de l'organigramme ci-dessus sont responsables d'une tâche spécifique qui sera expliquée dans la partie restante de cette section. Un réseau de neurones typique à deux couches avec deux entrées et une sortie qui utilise l'algorithme ci-dessus pour son apprentissage est présenté dans la Figure 2.12:

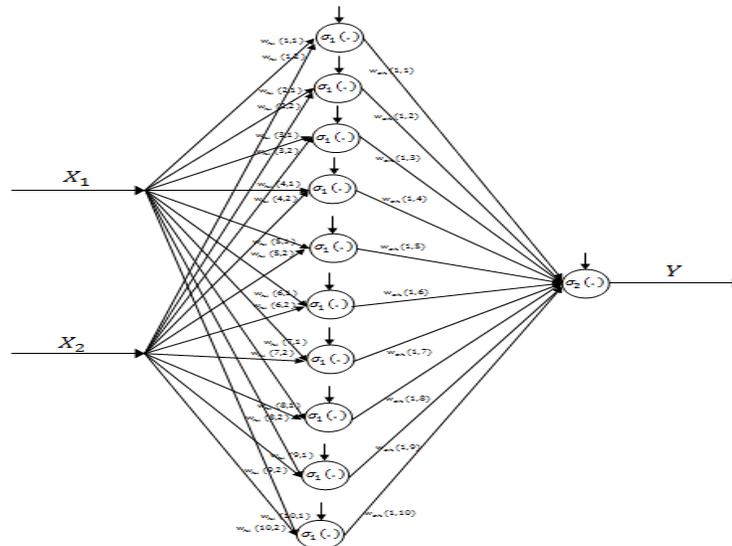


Figure 2. 12 Réseau de neurones à deux couches avec deux entrées et une sortie [29]

Comme le montre la figure ci-dessus, la première partie de l'algorithme consiste à initialiser le réseau, c'est à dire à définir le nombre d'entrées, le nombre de sorties et le nombre de neurones dans la couche cachée, ainsi que les taux d'apprentissage et d'impulsion. La terminologie suivante est utilisée dans ce travail pour définir les différents paramètres des réseaux de neurones mis en œuvre :

$N_i$  : le nombre d'entrées dans le réseau de neurones

$N_o$  : le nombre de sorties du réseau de neurones

$N_h$  : le nombre de neurones dans la couche cachée

$\eta$  : le taux d'apprentissage

$\beta$  : le rythme de Momentum

$N_p$  : le nombre de modèles de données d'entrée

$N_c$  : le nombre de cycles pour entraîner le réseau de neurones

$Whi$  : la couche cachée pour entrer les pondérations de la couche

$Woh$  : les pondérations de la couche de sortie à la couche cachée

La partie suivante de l'initialisation du réseau consiste à définir la fonction d'activation à utiliser pour les fonctions de la couche cachée et de la couche de sortie. L'étape suivante consiste à construire la partie de l'apprentissage et de la formation qui comprend les parties suivantes:

- Une boucle pour le nombre de cycles que nous voulons répéter le même ensemble de données d'entrée pour entraîner le réseau et atteindre l'erreur désirable. Le nombre de répétitions de cette boucle est  $N_c$  et l'indice de la boucle est  $j$ .
- Une boucle pour le nombre de modèles de données dont nous disposons dans chaque cycle. Cette boucle lira toutes les données dans un cycle. Le nombre de répétitions de cette boucle est  $N_p$  et l'indice de la boucle est  $i$ .

Comme on peut le voir sur l'organigramme de la Figure 2.12, la boucle qui lit les modèles de données dans chaque cycle se trouve à l'intérieur d'une autre boucle qui répète l'ensemble des données pour un certain nombre de cycles. La première chose à faire avant d'effectuer la rétropropagation pour chaque modèle de données est de faire la propagation vers l'avant et de trouver l'erreur entre la sortie du réseau de neurones et

la fonction réelle. La propagation vers l'avant est effectuée à l'aide des équations suivantes:

$$y_i = \sigma\left(\sum_{l=1}^L W_{oh_{il}} \sigma\left(\sum_{j=1}^n W_{hi_{lj}} x_j + v_{l0}\right) + w_{i0}\right)$$

$$z_l = \sigma\left(\sum_{j=1}^n w_{hi_{lj}} x_j + v_{l0}\right) \quad l = 1, 2, \dots, L$$

$$y_i = \sigma\left(\sum_{l=1}^L W_{oh_{il}} z_l + w_{i0}\right) \quad l = 0, 1, 2, \dots, m$$

Les équations ci-dessus concernent le cas général d'avoir  $i$  sorties,  $L$  neurones de la couche cachée et  $n$  entrées.

### 2.3.2.5 Régulateur de réseau de neurones entraîné hors ligne

Les définitions et la terminologie suivantes seront utilisées pour déterminer l'algorithme d'apprentissage du contrôleur de réseau de neurones selon la figure.

$$\text{On a } S = K_P \begin{bmatrix} x_e \\ y_e \\ \theta_e \end{bmatrix} + K_I \begin{bmatrix} \int x_e dt \\ \int y_e dt \\ \int \theta_e dt \end{bmatrix}$$

Soit  $S = \begin{bmatrix} S_1 \\ S_2 \\ S_3 \end{bmatrix}$ , on veut remplacer par un réseau de neurone  $\hat{S} = \begin{bmatrix} \hat{S}_1 \\ \hat{S}_2 \\ \hat{S}_3 \end{bmatrix}$  de mêmes entrées

et de mêmes sorties.

$$\text{On a l'erreur } e = \begin{bmatrix} x_e(t) \\ y_e(t) \\ \theta_e(t) \end{bmatrix} = \begin{bmatrix} x(t) \\ y(t) \\ \theta(t) \end{bmatrix} - \begin{bmatrix} x_d(t) \\ y_d(t) \\ \theta_d(t) \end{bmatrix}$$

Dans cette méthode, le contrôleur est entraîné hors ligne pour obtenir les valeurs des poids qui permettent à l'erreur de position  $e$  s'approche de zéro. Le RN se compose de 3 couches de neurones:

- Une couche d'entrée: dans cette couche, il y a 3 neurones, l'entrée de ces neurones sont les erreurs  $e = \begin{bmatrix} x_e(t) \\ y_e(t) \\ \theta_e(t) \end{bmatrix}$ .
- Une couche cachée: dans cette couche, il y a  $N = 7$ . Elles sont reliées aux sorties de la couche d'entrée par les poids  $V_n$ , avec  $n = 1, \dots, N$ . Les sorties de ces neurones sont actionnées par la fonction sigmoïde. Les sorties de ces 7 neurones de la couche cachée sont transmises à la couche de sortie par les poids  $W_n$ .
- Une couche de sortie : cette couche est composée de 3 neurones. Les entrées de ces neurones sont les sorties de ces neurones sont  $z_n$ , avec  $n = 1, \dots, N$  de la couche cachée. Ces neurones sont actionnés par fonction linéaire. Les poids associés aux entrées sont  $W_n$ . La sortie de cette couche est  $\hat{S}$ .

La topologie de réseau neuronal que nous souhaitons utiliser est illustrée à la Figure 2.13 :

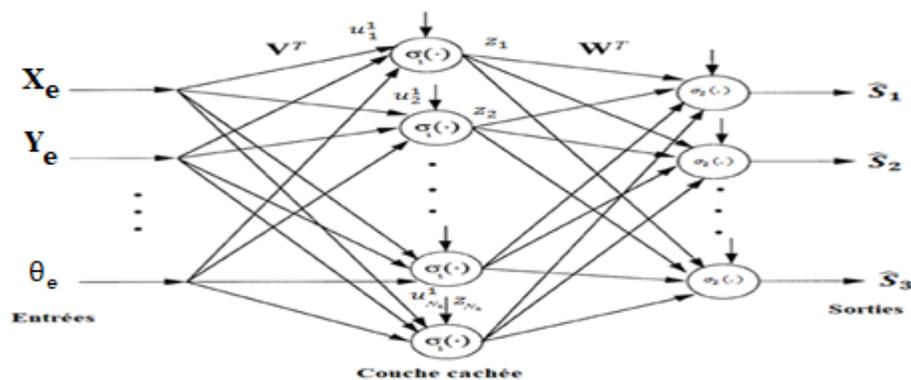


Figure 2. 13 Modèle de réseau de neurones utilisé

On remplace le régulateur à base PI par le régulateur de réseau de neurones entraîné hors ligne.

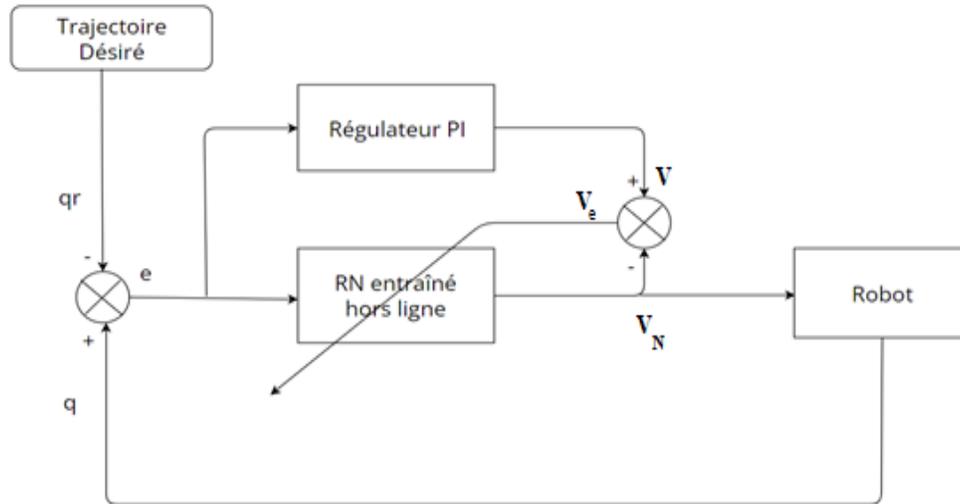


Figure 2. 14 Structure du régulateur de Réseau de neurones

L'équation de la loi de commande est :

$$V_N = P(\theta) \left( -\hat{S} + \begin{bmatrix} \dot{x}_d \\ \dot{y}_d \\ \dot{\theta}_d \end{bmatrix} \right)$$

L'erreur à minimiser pendant la formation est:

$$E_V = \frac{1}{2} (V_e)^2 = \frac{1}{2} (V - V_N)^2$$

La modification des poids est donnée par:

$$W_{il}(k+1) = W_{il}(k) - \eta z_l \delta_i^2$$

$$V_{lj}(k+1) = V_{lj}(k) - \eta e_j \delta_l^1$$

Ce réseau est entraîné avec l'algorithme de rétropropagation de Levenberg-Marquardt, sauf si la mémoire est insuffisante, auquel cas la rétropropagation à gradient conjugué gradué sera utilisée.

Les étapes suivantes visent à former le réseau de neurones et à le préparer à l'approximation en ligne du modèle de système :

- Entraînement hors ligne du réseau de neurones avec des entrées sinusoïdales à différentes fréquences pour exciter toutes les fréquences modales du système.
- Optimisation des paramètres du réseau de neurones pour atteindre la meilleure performance d'approximation possible pour une entrée qui contient toutes les fréquences du système.
- Tester les performances d'apprentissage en ligne du réseau de neurones avec les poids du réseau obtenues lors de la formation hors ligne et les comparer avec les sorties des modèles de robots réels.

#### 2.3.2.6 Régulateur à base PI et réseau de neurones entraîné en ligne

Pour augmenter les performances de régulateur à base PI, on le combine avec le régulateur de réseau de neurones entraîné en ligne.

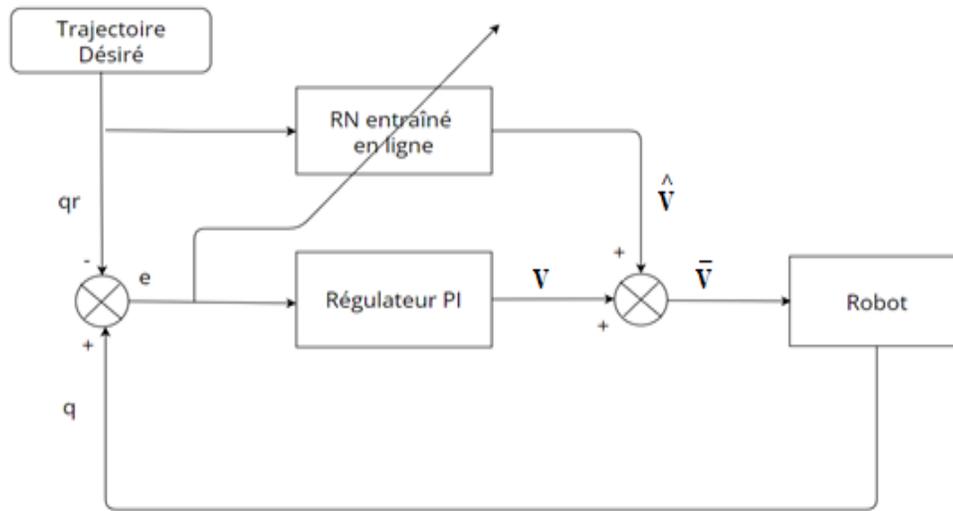


Figure 2. 15 Structure du régulateur à base PI et Réseau de neurones

L'équation de la loi de commande est :

$$\bar{v} = v + \hat{v}$$

L'erreur à minimiser pendant la formation est:

$$E = \frac{1}{2} e^2 = \frac{1}{2} (q - q_d)^2$$

Dans cette méthode, le contrôleur est entraîné hors ligne pour obtenir les valeurs des poids qui permettent à l'erreur de position  $e$  s'approche de zéro. Le RN se compose de 3 couches de neurones:

- Une couche d'entrée: dans cette couche, il y a 3 neurones, l'entrée de ces

$$\text{neurones sont les erreurs } q_d = \begin{bmatrix} x_d(t) \\ y_d(t) \\ \theta_d(t) \end{bmatrix}.$$

- Une couche cachée: dans cette couche, il y a  $N = 7$ . Elles sont reliées aux sorties de la couche d'entrée par les poids  $V_n$ , avec  $n = 1, \dots, N$ . Les sorties de

ces neurones sont actionnées par la fonction sigmoïde. Les sorties de ces 7 neurones de la couche cachée sont transmises à la couche de sortie par les poids  $W_n$ .

- Une couche de sortie : cette couche est composée de 3 neurones. Les entrées de ces neurones sont les sorties de ces neurones sont  $z_n$ , avec  $n = 1, \dots, N$  de la couche cachée. Ces neurones sont actionnés par fonction linéaire. Les poids associés aux entrées sont  $W_n$ . La sortie de cette couche est  $\hat{V}$ .

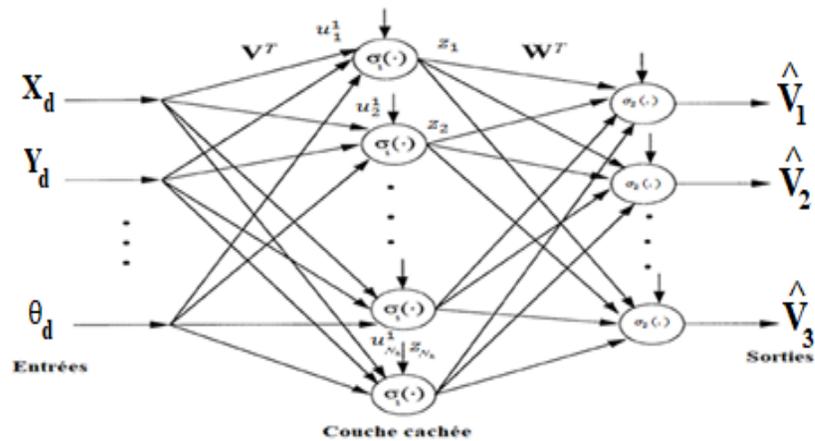


Figure 2.16 Modèle de réseau de neurones utilisé

La modification des poids est donnée par:

$$W_{il}(k+1) = W_{il}(k) - \eta z_l \delta_i^2$$

$$V_{lj}(k+1) = V_{lj}(k) - \eta q_j \delta_l^1$$

Les entrées de réseau de neurones entraîné en ligne sont les références de trajectoire désirée.

Pour les deux réseaux de neurones en ligne et hors ligne sont des réseaux à deux couches avec des neurones cachés sigmoïdes et des neurones de sortie linéaires. Ils peuvent s'adapter à des problèmes de cartographie multidimensionnelle de manière arbitraire, à condition de disposer de données cohérentes et d'un nombre suffisant de neurones dans sa couche cachée.

Ce réseau est entraîné avec l'algorithme de rétropropagation de Levenberg-Marquardt, sauf si la mémoire est insuffisante, auquel cas la rétropropagation à gradient conjugué gradué sera utilisée.

## CHAPITRE III

### RÉSULTATS ET DISCUSSIONS

#### 3.1 Introduction

Après avoir modélisé le système du robot mobile omnidirectionnel et proposé les lois de commande qu'on utilise, il faut les valider en simulations en utilisant les logiciels de simulations et expérimentalement sur le modèle physique du robot « Robotino » avec 3 trajectoires différentes (linéaire, circulaire et elliptique) en présentant les courbes d'évolution de la position du robot, l'évolution des composantes et des erreurs, et les vitesses du robot et des roues.

Dans la première partie de ce chapitre, on va présenter les résultats de simulation en utilisant le logiciel MATLAB/SIMULINK. Dans la deuxième partie de ce chapitre, on va présenter les résultats expérimentaux du robot en utilisant le régulateur à base PI, le régulateur de réseau de neurones et en combinant le régulateur à base PI et le réseau de neurones entraîné en ligne. Enfin on a présenté une comparaison entre ces 3 régulateurs.

#### 3.2 Résultats de simulation sur SIMULINK

Avant de valider les résultats expérimentalement, on a fait des simulations en utilisant le logiciel MATLAB/SIMULINK.

### 3.2.1 En utilisant le régulateur à base PI

Dans cette partie, on va présenter les résultats de simulation du SIMULINK de trois trajectoires différentes en utilisant le régulateur à base PI.

Avec  $K_P = \text{diag}\{10,10,10\}$  et  $K_I = \text{diag}\{2,2,2\}$

- Trajectoire linéaire :

L'équation de la trajectoire est de la forme :

$$\begin{cases} X_d = Y_{d0} + V t \\ Y_d = Y_{d0} + V t \\ \theta_d = \frac{\pi}{4} \end{cases}$$

Avec :

$V = 2$ ;  $X_{d0} = 0$  et  $Y_{d0} = 0$

Les positions initiales du robot sont:  $\begin{cases} X_0 = 0.5 \\ Y_0 = 1 \\ \theta_0 = 0 \end{cases}$

La courbe suivante représente la variation de la position du robot le long de la trajectoire linéaire dans le plan XY en utilisant le régulateur à base PI.

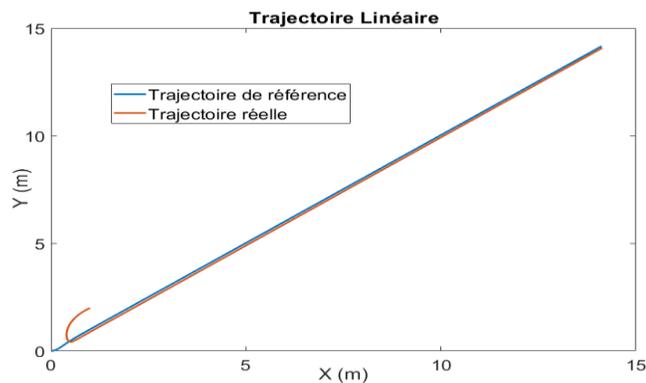


Figure 3.1 Trajection linéaire du robot dans le plan XY

Le courbe suivant représente l'évolution des positions de référence et les positions réelles du robot en fonction du temps pour la trajectoire linéaire en utilisant le contrôleur PI.

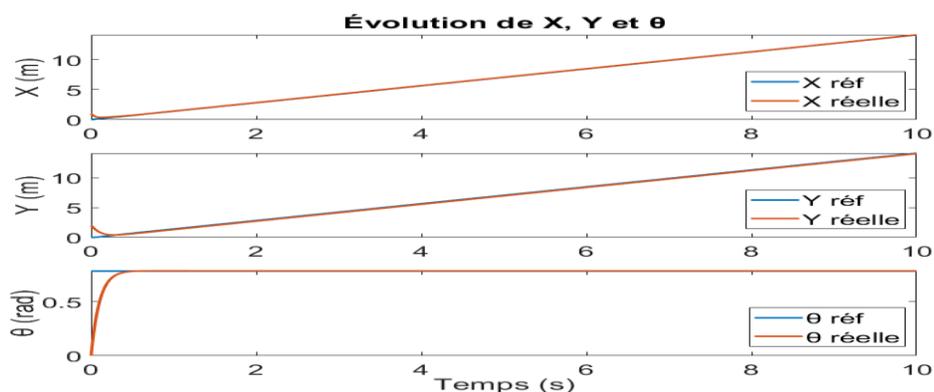


Figure 3. 2 Évolution des composantes du robot au cours du temps

Le courbe suivant représente l'évolution des erreurs des positions au cours du temps pour la trajectoire linéaire en utilisant le contrôleur PI.

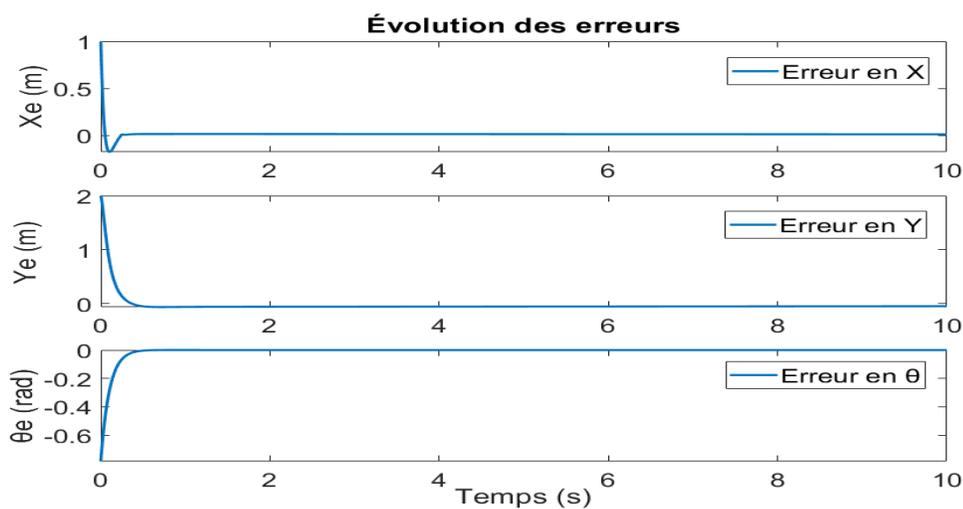


Figure 3. 3 Évolution des erreurs au cours du temps

Les figures 3.1, 3.2 et 3.3 illustrent les résultats de la simulation du suivi de la trajectoire linéaire pour le robot mobile omnidirectionnel. Ces résultats indiquent le suivi de la trajectoire du contrôleur cinématique du régulateur proposé.

- Trajectoire circulaire :

L'équation de la trajectoire est de la forme :

$$\begin{cases} X_d = X_{d0} + r \cos(\omega_0 + \omega_r t) \\ Y_d = Y_{d0} + r \sin(\omega_0 + \omega_r t) \\ \theta_d = 0 \end{cases}$$

Avec :  $\omega_0 = 0$ ;  $\omega_r = 1$ ;  $r = 0.5$ ;  $X_{d0} = 0$ ;  $Y_{d0} = 0$

Les positions initiales du robot sont:  $\begin{cases} X_0 = 0.5 \\ Y_0 = 0.5 \\ \theta_0 = 0 \end{cases}$

La courbe suivante représente la variation de la position du robot le long de la trajectoire circulaire dans le plan XY en utilisant le régulateur à base PI.

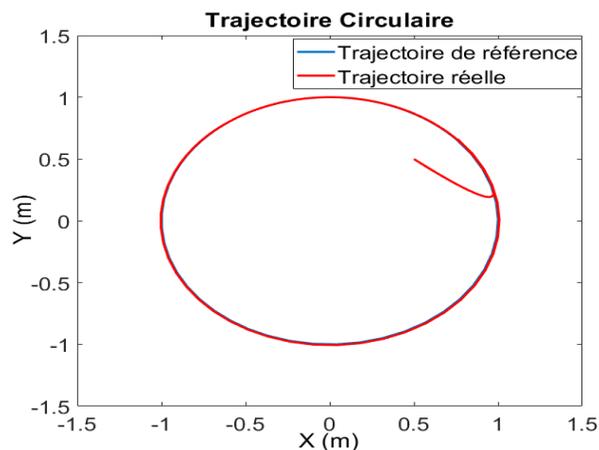


Figure 3. 4 Trajectoire circulaire du robot dans le plan XY

Le courbe suivant représente l'évolution des positions de référence et les positions réelles du robot en fonction du temps pour la trajectoire circulaire en utilisant le contrôleur PI.

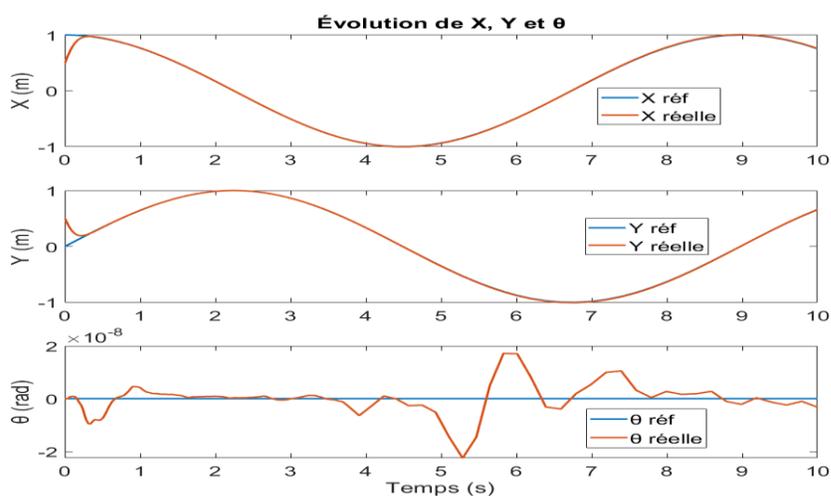


Figure 3. 5 Évolution des composantes du robot au cours du temps

Le courbe suivant représente l'évolution des erreurs des positions au cours du temps pour la trajectoire circulaire en utilisant le contrôleur PI.

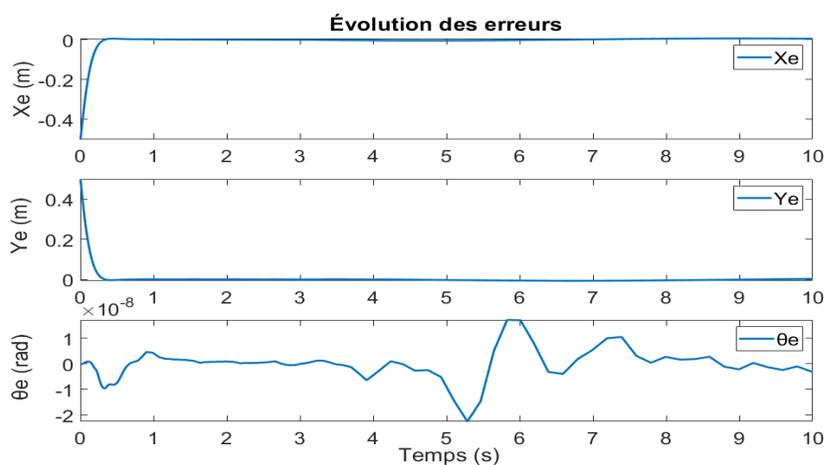


Figure 3. 6 Évolution des erreurs au cours du temps

Les figures 3.4, 3.5 et 3.6 illustrent les résultats de la simulation du suivi de la trajectoire circulaire pour le robot mobile omnidirectionnel. Ces résultats indiquent le suivi de la trajectoire du contrôleur cinématique du régulateur proposé.

- Trajectoire elliptique:

L'équation de la trajectoire est de la forme :

$$\begin{cases} X_d = X_{d0} + r_1 \cos(\omega_0 + \omega_r t) \\ Y_d = Y_{d0} + r_2 \sin(\omega_0 + \omega_r t) \\ \theta_d = 0 \end{cases}$$

Avec :  $\omega_0 = 0.1$ ;  $\omega_r = 0.35$ ;  $r_1 = 0.5$ ;  $r_2 = 0.5$   $X_{d0} = 0$  et  $Y_{d0} = 0$

Les positions initiales du robot sont:  $\begin{cases} X_0 = 0 \\ Y_0 = 0 \\ \theta_0 = 0 \end{cases}$

La courbe suivante représente la variation de la position du robot le long de la trajectoire elliptique dans le plan XY en utilisant le régulateur à base PI.

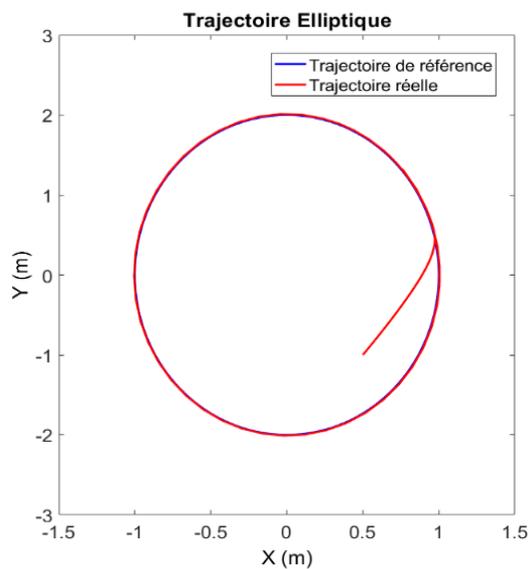


Figure 3. 7 Trajectoire circulaire du robot dans le plan XY

Le courbe suivant représente l'évolution des positions de référence et les positions réelles du robot en fonction du temps pour la trajectoire elliptique en utilisant le contrôleur PI.

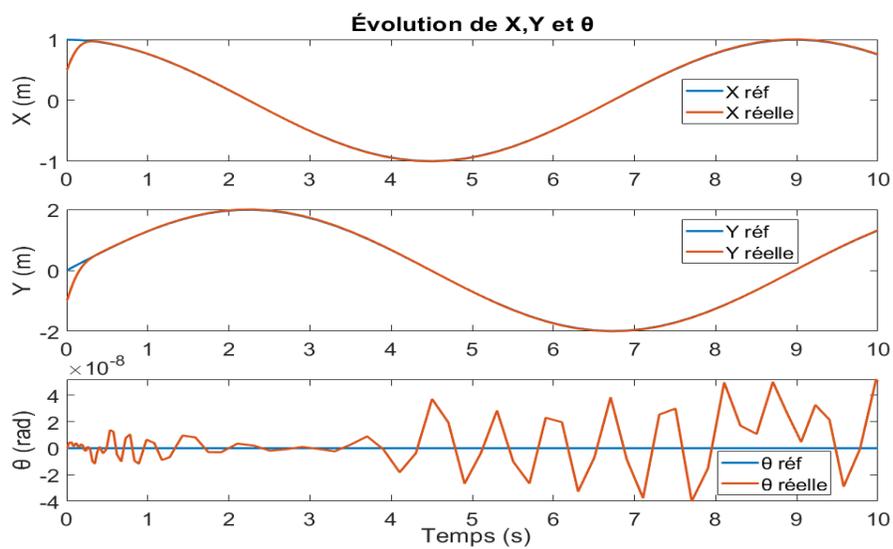


Figure 3. 8 Évolution des composantes du robot au cours du temps

Le courbe suivant représente l'évolution des erreurs des positions au cours du temps pour la trajectoire elliptique en utilisant le contrôleur PI.

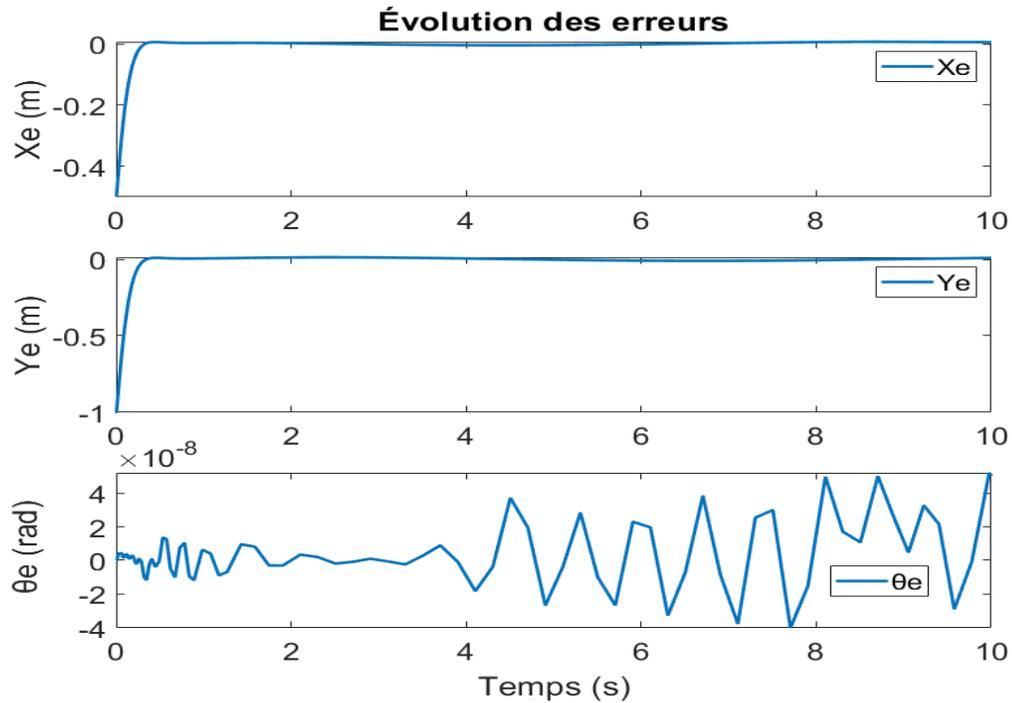


Figure 3. 9 Évolution des erreurs au cours du temps

Les figures 3.7, 3.8 et 3.9 illustrent les résultats de la simulation du suivi de la trajectoire elliptique pour le robot mobile omnidirectionnel. Ces résultats indiquent le suivi de la trajectoire du contrôleur cinématique du régulateur proposé.

### 3.2.2 En utilisant le régulateur de réseau de neurones entraîné hors ligne

Dans cette partie, on a présenté les résultats de simulation du SIMULINK de trois trajectoires différentes en utilisant le régulateur de réseau de neurones.

Avec  $\eta = 0.3$  et  $\beta = 0.7$

- Trajectoire linéaire :

La courbe suivante représente la variation de la position du robot le long de la trajectoire linéaire dans le plan XY en utilisant le régulateur de réseau de neurones entraîné hors ligne.

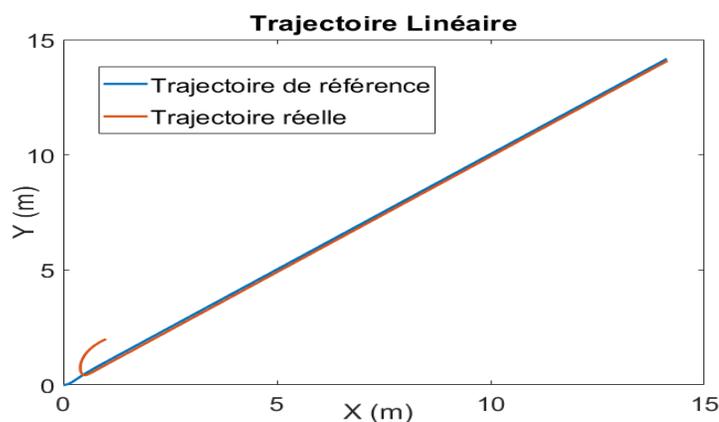


Figure 3. 10 Trajectoire linéaire du robot dans le plan XY

Le courbe suivant représente l'évolution des positions de référence et les positions réelles du robot en fonction du temps pour la trajectoire linéaire en utilisant le contrôleur de réseau de neurones entraîné hors ligne.

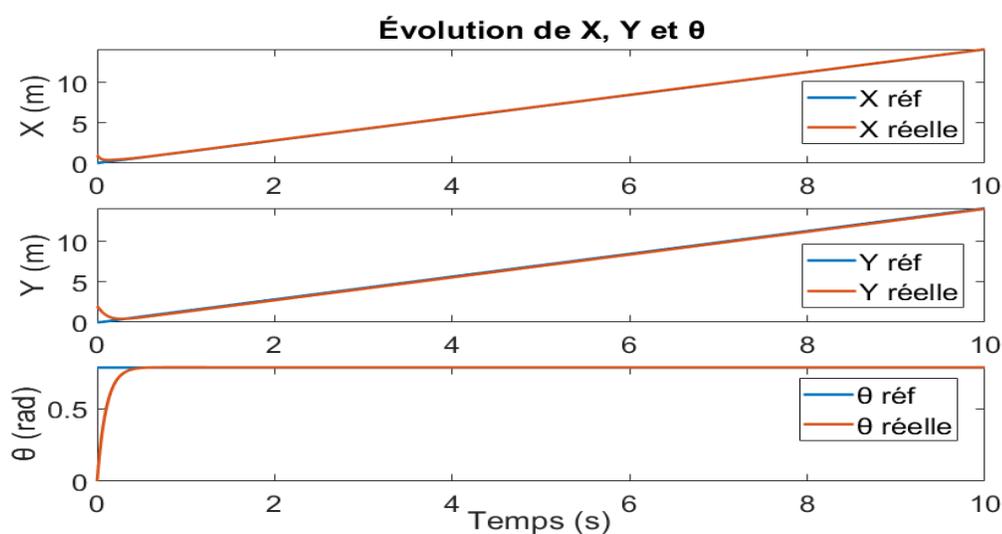


Figure 3. 11 Évolution des composantes du robot au cours du temps

Le courbe suivant représente l'évolution des erreurs des positions au cours du temps pour la trajectoire linéaire en utilisant le contrôleur de réseau de neurones entraîné hors ligne.

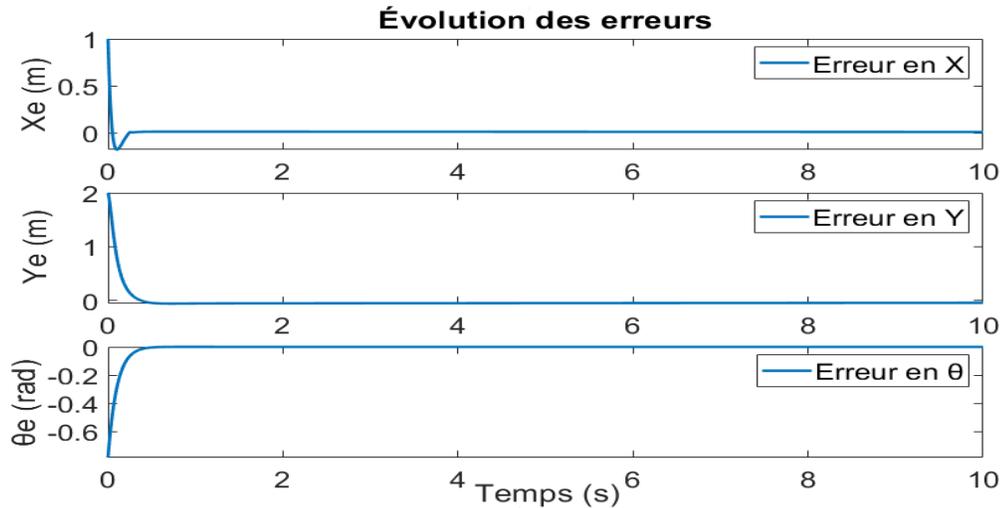


Figure 3. 12 Évolution des erreurs au cours du temps

Les figures 3.10, 3.11 et 3.12 illustrent les résultats de la simulation du suivi de la trajectoire linéaire pour le robot mobile omnidirectionnel. Ces résultats indiquent le suivi de la trajectoire du contrôleur cinématique du régulateur proposé de manière très proche du régulateur PI.

- Trajectoire circulaire :

La courbe suivante représente la variation de la position du robot le long de la trajectoire circulaire dans le plan XY en utilisant le régulateur de réseau de neurones entraîné hors ligne.

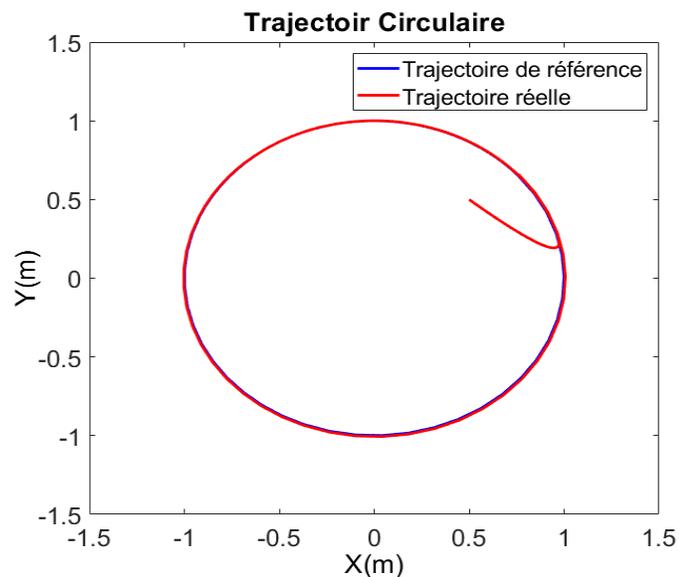


Figure 3. 13 Trajectoire circulaire du robot dans le plan XY

Le courbe suivant représente l'évolution des positions de référence et les positions réelles du robot en fonction du temps pour la trajectoire circulaire en utilisant le contrôleur de réseau de neurones entraîné hors ligne.

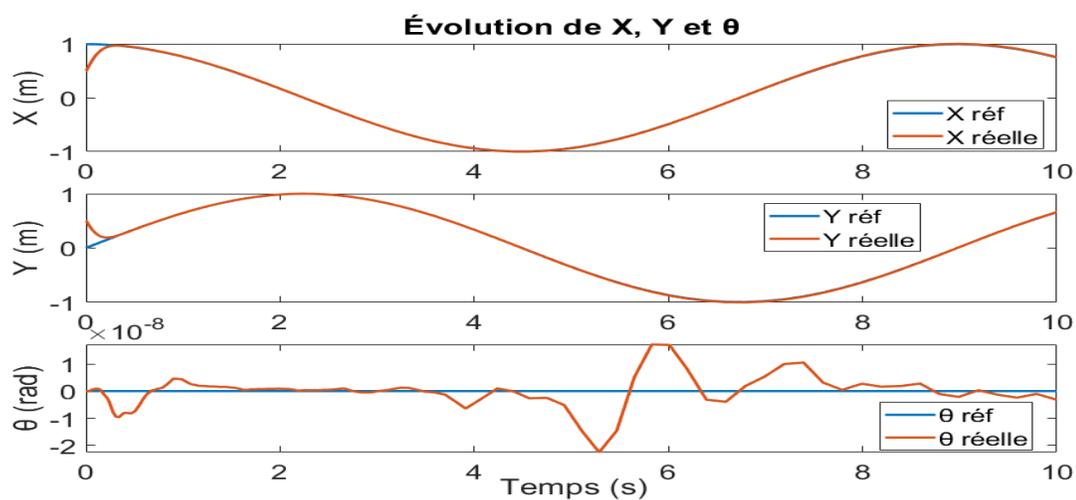


Figure 3. 14 Évolution des composantes au cours du temps

Le courbe suivant représente l'évolution des erreurs des positions au cours du temps pour la trajectoire circulaire en utilisant le contrôleur de réseau de neurones.

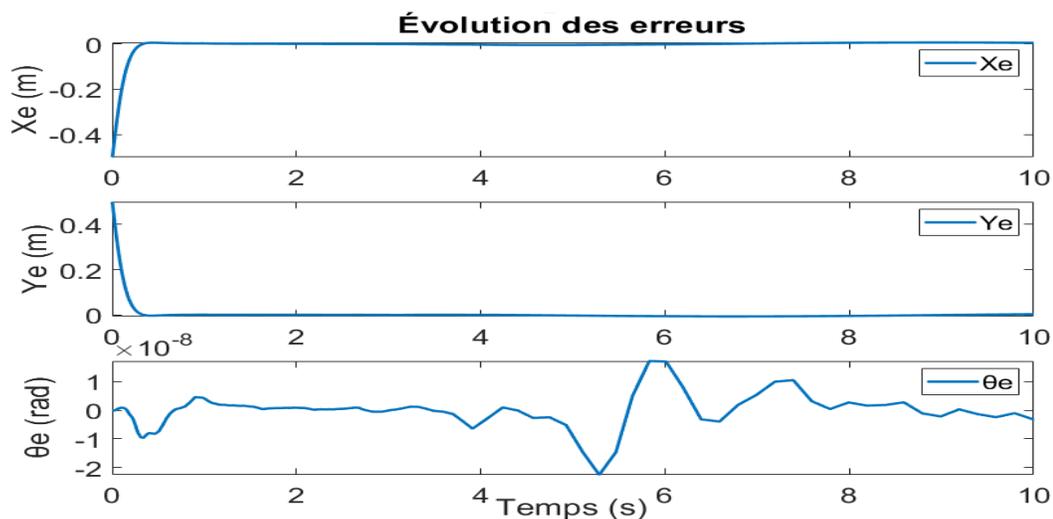


Figure 3. 15 Évolution des erreurs au cours du temps

Les figures 3.13, 3.14 et 3.15 illustrent les résultats de la simulation du suivi de la trajectoire circulaire pour le robot mobile omnidirectionnel. Ces résultats indiquent le suivi de la trajectoire du contrôleur cinématique du régulateur proposé de manière très proche que celle de PI.

- Trajectoire elliptique:

La courbe suivante représente la variation de la position du robot le long de la trajectoire elliptique dans le plan XY en utilisant le régulateur de réseau de neurones entraîné hors ligne.

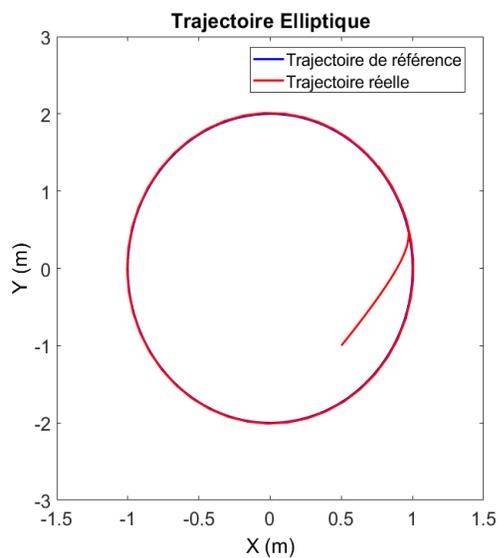


Figure 3. 16 Trajectoire elliptique du robot dans le plan XY

Le courbe suivant représente l'évolution des positions de référence et les positions réelles du robot en fonction du temps pour la trajectoire elliptique en utilisant le contrôleur de réseau de neurones entraîné hors ligne.

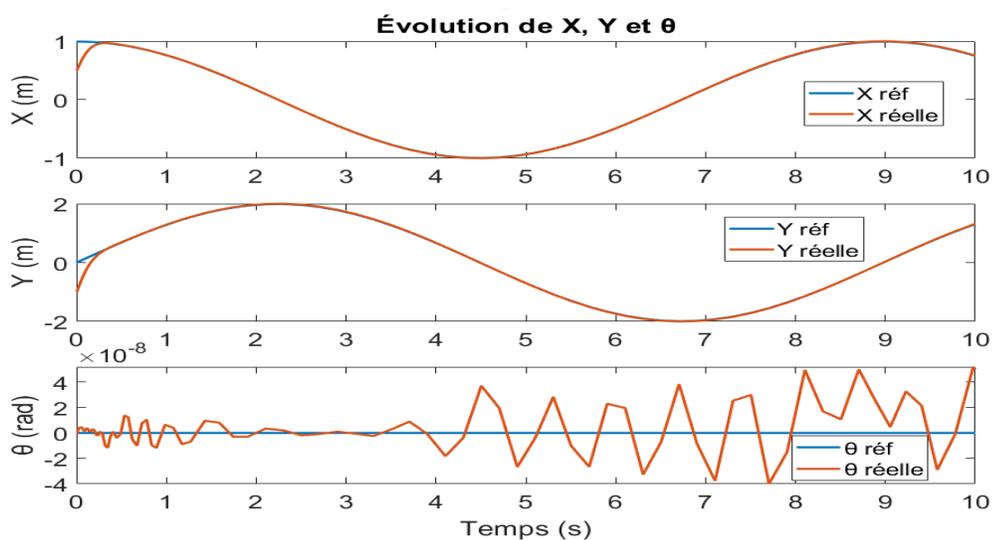


Figure 3. 17 Évolution des composantes du robot au cours du temps

Le courbe suivant représente l'évolution des erreurs des positions au cours du temps pour la trajectoire elliptique en utilisant le contrôleur de réseau de neurones entraîné hors ligne.

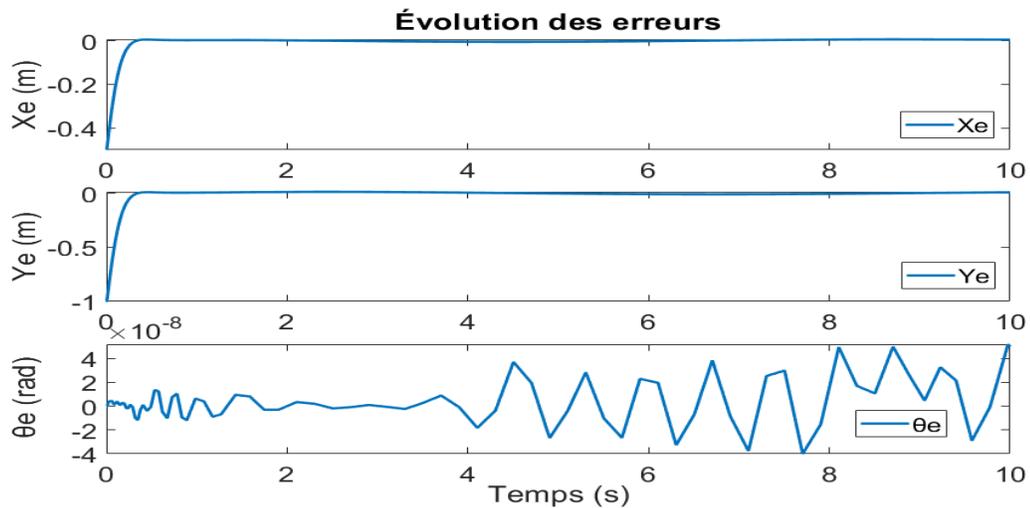


Figure 3. 18 Évolution des erreurs au cours du temps

Les figures 3.16, 3.17 et 3.18 illustrent les résultats de la simulation du suivi de la trajectoire elliptique pour le robot mobile omnidirectionnel. Ces résultats indiquent le suivi de la trajectoire du contrôleur cinématique du régulateur proposé de manière très proche de celle de PI.

3.2.3 En utilisant le régulateur PI et le réseau de neurones entraîné en ligne  
 Dans cette partie, on a présenté les résultats de simulation du SIMULINK de trois trajectoires différentes en combinant le régulateur à base PI et le réseau de neurones.

Avec  $\eta = 0.3$  et  $\beta = 0.7$

- Trajectoire linéaire :

La courbe suivante représente la variation de la position du robot le long de la trajectoire linéaire dans le plan XY en utilisant le régulateur à base PI combiné avec le réseau de neurones entraîné en ligne.

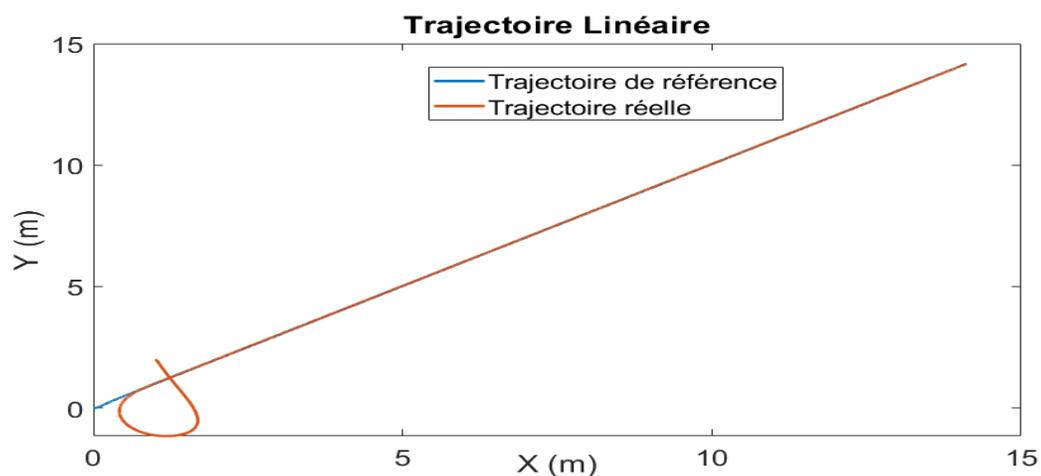


Figure 3. 19 Trajectoire linéaire du robot dans le plan XY

Le courbe suivant représente l'évolution des positions de référence et les positions réelles du robot en fonction du temps pour la trajectoire linéaire en utilisant le contrôleur PI combiné avec le réseau de neurones entraîné en ligne.

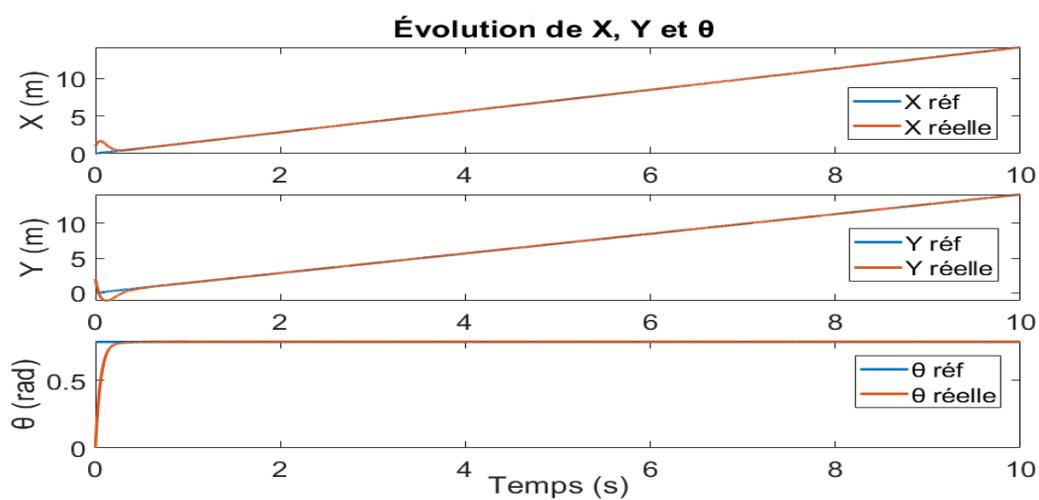


Figure 3. 20 Évolution des composantes du robot dans le plan XY

Le courbe suivant représente l'évolution des erreurs des positions au cours du temps pour la trajectoire linéaire en utilisant le contrôleur PI combiné avec le réseau de neurones entraîné en ligne.

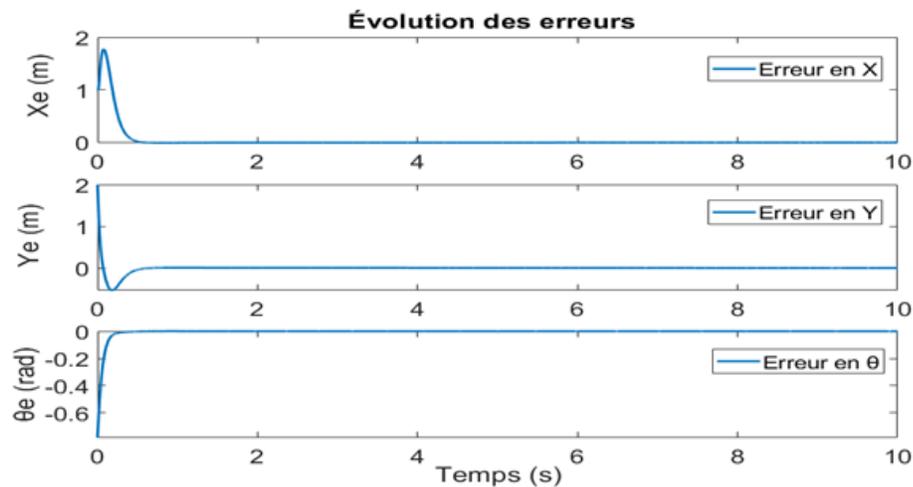


Figure 3. 21 Évolution des erreurs au cours du temps

Les figures 3.19, 3.20 et 3.21 illustrent les résultats de la simulation du suivi de la trajectoire linéaire pour le robot mobile omnidirectionnel. Ces résultats indiquent le suivi de la trajectoire du contrôleur cinématique du régulateur proposé. Le régulateur de la combinaison de PI et de réseau de neurones entraîné en ligne donne des résultats plus précis que le PI et le RN entraîné hors ligne. Ce régulateur a une réponse plus rapide par rapport à l'autres et a réussi à minimiser les erreurs de position.

- Trajectoire circulaire :

La courbe suivante représente la variation de la position du robot le long de la trajectoire circulaire dans le plan XY en utilisant le régulateur à base PI combiné avec le réseau de neurones entraîné en ligne.

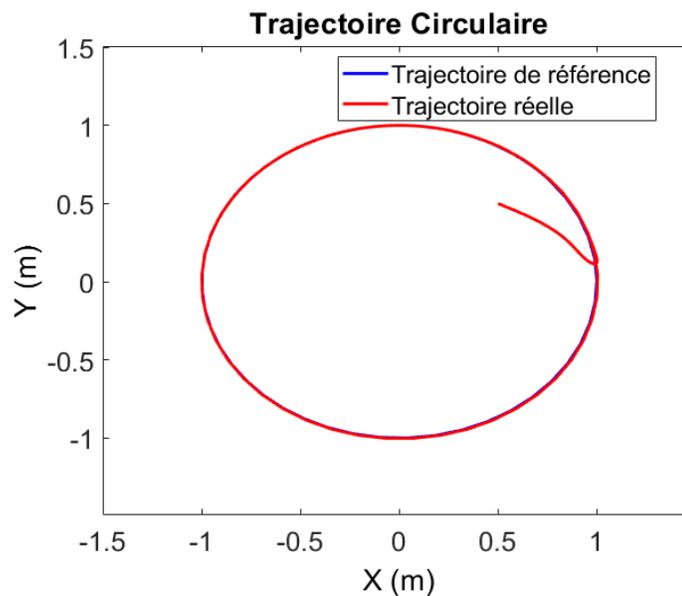


Figure 3. 22 Trajectoire circulaire du robot dans le plan XY

Le courbe suivant représente l'évolution des positions de référence et les positions réelles du robot en fonction du temps pour la trajectoire circulaire en utilisant le contrôleur PI combiné avec le réseau de neurones entraîné en ligne.

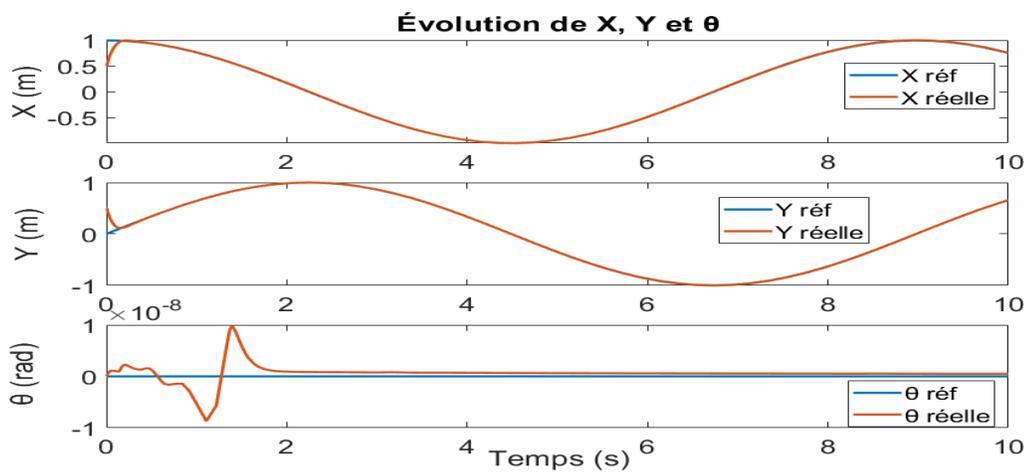


Figure 3. 23 Évolution des composantes du robot au cours du temps

Le courbe suivant représente l'évolution des erreurs des positions au cours du temps pour la trajectoire circulaire en utilisant le contrôleur PI combiné avec le réseau de neurones entraîné en ligne.

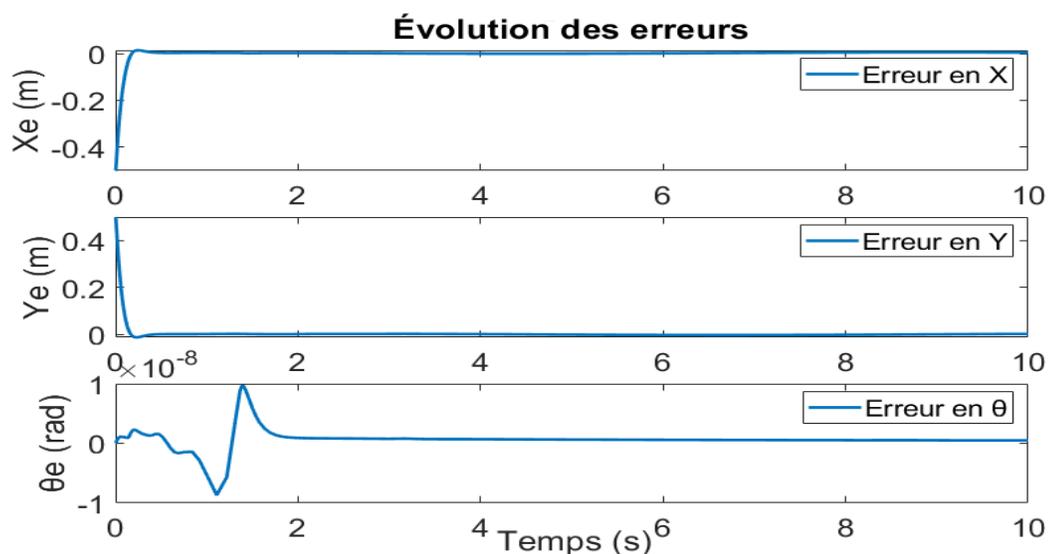


Figure 3. 24 Évolution des erreurs au cours du temps

Les figures 3.22, 3.23 et 3.24 illustrent les résultats de la simulation du suivi de la trajectoire circulaire pour le robot mobile omnidirectionnel. Ces résultats indiquent le suivi de la trajectoire du contrôleur cinématique du régulateur proposé. Le régulateur de la combinaison de PI et de réseau de neurones entraîné en ligne donne des résultats plus précis que le PI et le RN entraîné hors ligne. Ce régulateur a une réponse plus rapide par rapport à l'autres et a réussi à minimiser les erreurs de position.

- Trajectoire elliptique:

La courbe suivante représente la variation de la position du robot le long de la trajectoire elliptique dans le plan XY en utilisant le régulateur à base PI combiné avec le réseau de neurones entraîné en ligne.

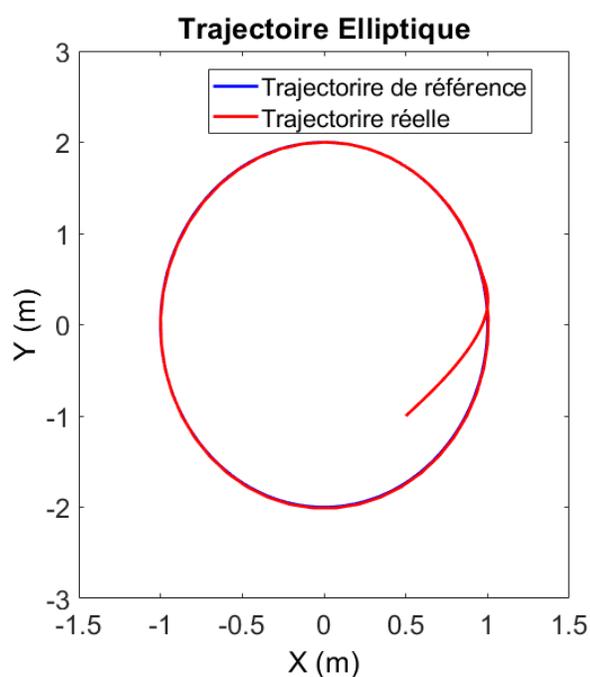


Figure 3. 25 Trajectoire elliptique du robot dans le plan XY

Le courbe suivant représente l'évolution des positions de référence et les positions réelles du robot en fonction du temps pour la trajectoire elliptique en utilisant le contrôleur PI combiné avec le réseau de neurones entraîné en ligne.

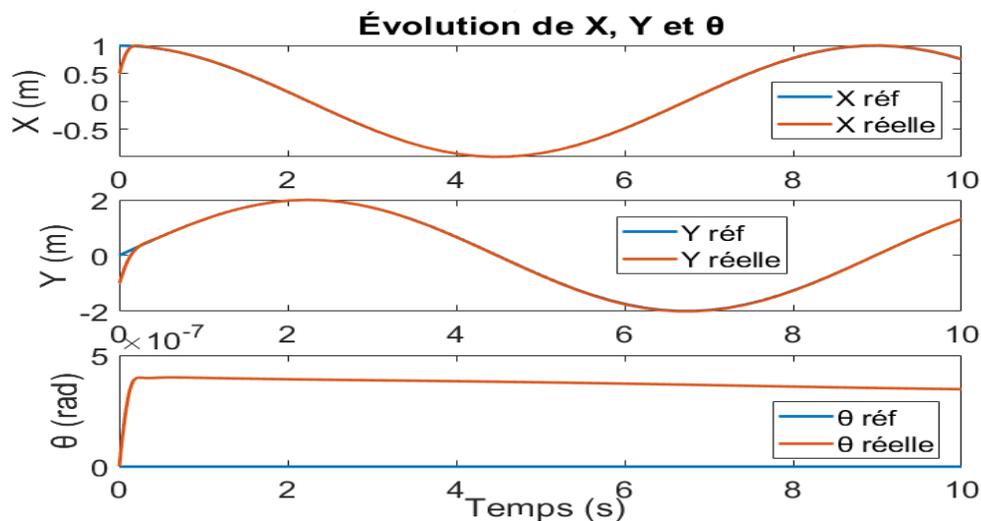


Figure 3. 26 Évolution des composantes du robot au cours du temps

Le courbe suivant représente l'évolution des erreurs des positions au cours du temps pour la trajectoire elliptique en utilisant le contrôleur PI combiné avec le réseau de neurones entraîné en ligne.

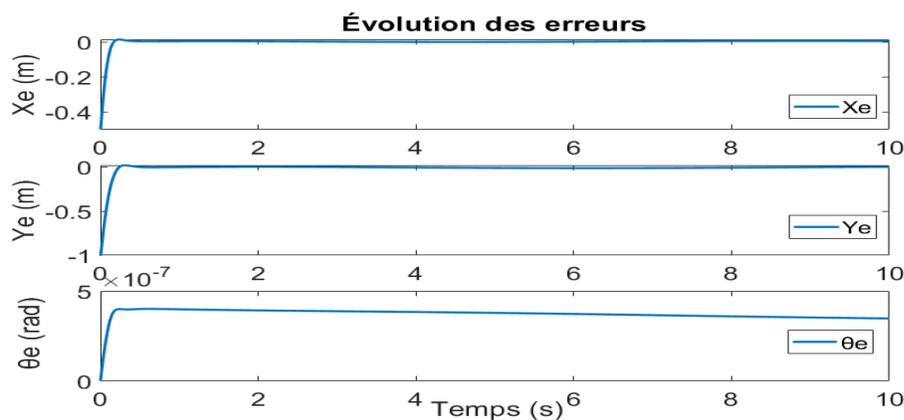


Figure 3. 27 Évolution des erreurs au cours du temps

Les figures 3.19, 3.20 et 3.21 illustrent les résultats de la simulation du suivi de la trajectoire elliptique pour le robot mobile omnidirectionnel. Ces résultats indiquent le suivi de la trajectoire du contrôleur cinématique du régulateur proposé. Le régulateur

de la combinaison de PI et de réseau de neurones entraîné en ligne donne des résultats plus précis que le PI et le RN entraîné hors ligne. Ce régulateur a une réponse plus rapide par rapport à l'autres et a réussi à minimiser les erreurs de position.

### 3.2.4 Interprétation

Les figures suivantes représentent une comparaison entre les variations de la position du robot omnidirectionnel en utilisant les différents régulateurs pour 3 différentes trajectoires.

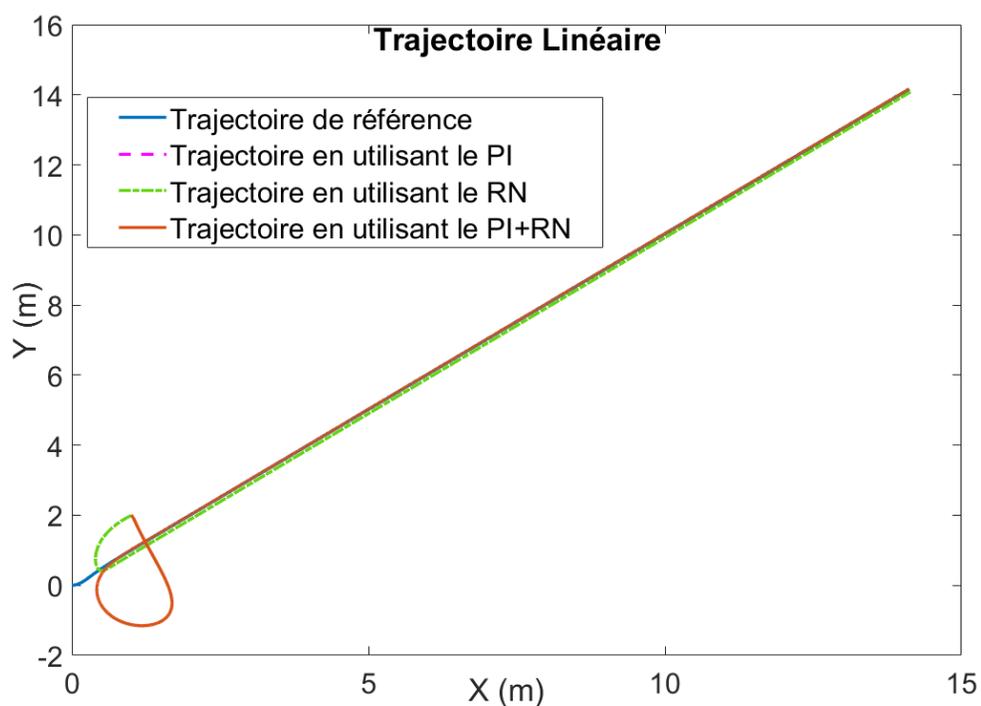


Figure 3. 28 Comparaison entre les 3 régulateurs pour une trajectoire linéaire

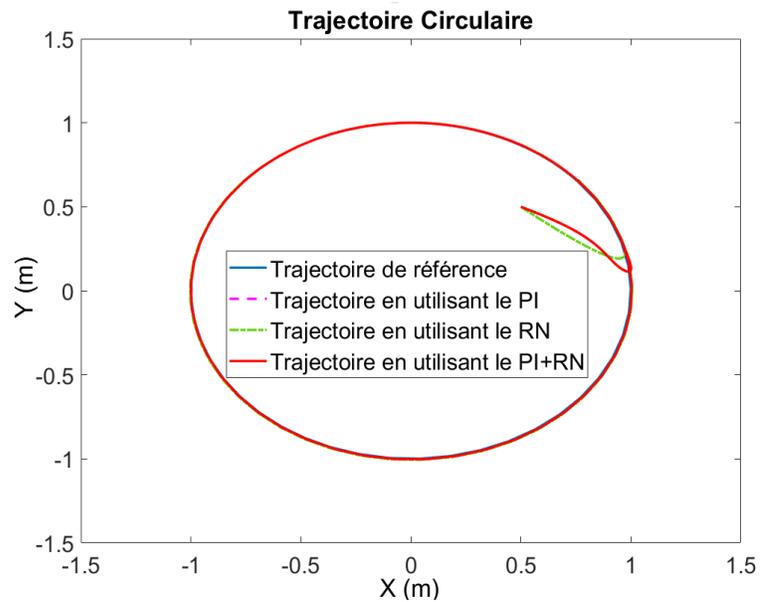


Figure 3. 29 Comparaison entre les 3 régulateurs pour une trajectoire circulaire

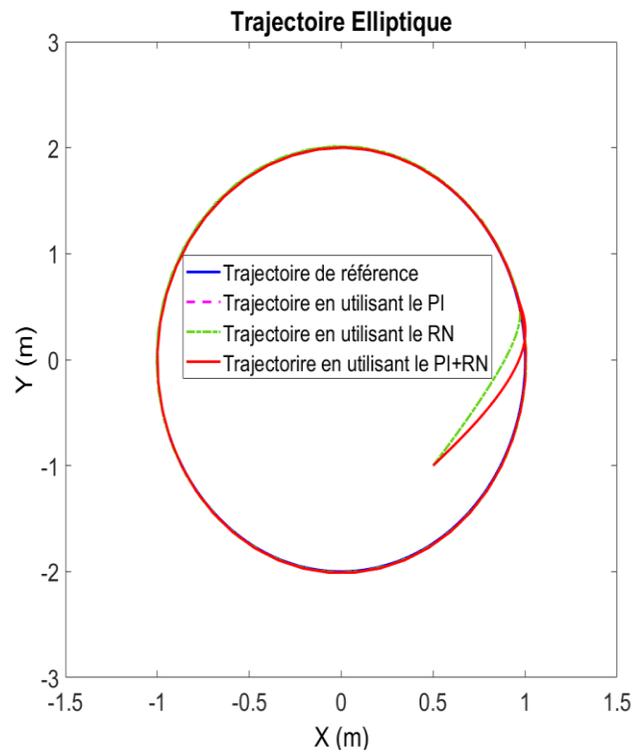


Figure 3. 30 Comparaison entre les 3 régulateurs PI pour une trajectoire elliptique

Le régulateur de réseau de neurones entraîné hors-ligne donne des résultats très proches à celle du régulateur à base PI. On peut interpréter que ce régulateur de réseau de neurones est un compensateur de régulateur à base PI et il peut le remplacer.

Le régulateur de la combinaison du régulateur à base PI et le réseau de neurones entraîné en ligne a une réponse plus rapide seul et il donne des résultats mieux que celle de PI.

### 3.3 Résultats Expérimentales

#### 3.3.1 En utilisant le régulateur à base PI

Dans cette partie, on a présente les résultats expérimentaux des (3) trajectoires différentes en utilisant le régulateur à base PI.

Avec  $K_p = \text{diag}\{2,2,2\}$  et  $K_I = \text{diag}\{0.2,0.2,0.2\}$

- Trajectoire linéaire :

L'équation de la trajectoire est de la forme :

$$\begin{cases} X_d = Y_{d0} + V t \\ Y_d = Y_{d0} + V t \\ \theta_d = \frac{\pi}{2} \end{cases}$$

Avec :

$V = 0.4$ ;  $X_{d0} = 0$  et  $Y_{d0} = 0$

Les positions initiales du robot sont:  $\begin{cases} X_0 = 0.5 \\ Y_0 = 0.8 \\ \theta_0 = 0 \end{cases}$

La courbe suivante représente la variation de la position du robot le long de la trajectoire linéaire dans le plan XY en utilisant le régulateur à base PI.

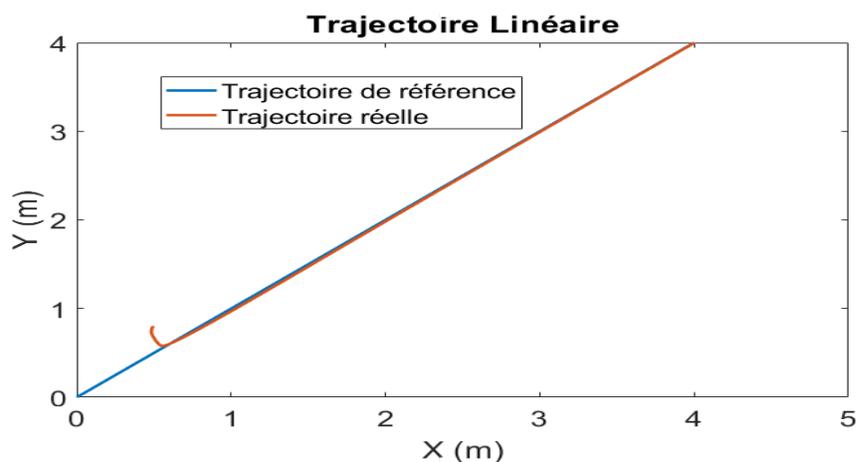


Figure 3. 31 Trajectoire linéaire du robot dans le plan XY

Le courbe suivant représente l'évolution des positions de référence et les positions réelles du robot en fonction du temps pour la trajectoire linéaire en utilisant le contrôleur PI.

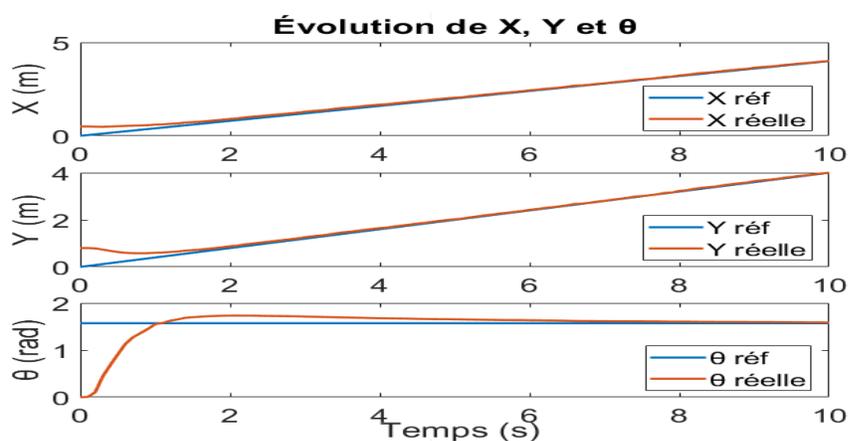


Figure 3. 32 Évolution des composantes du robot dans le plan XY

Le courbe suivant représente l'évolution des erreurs des positions au cours du temps pour la trajectoire linéaire en utilisant le contrôleur PI.

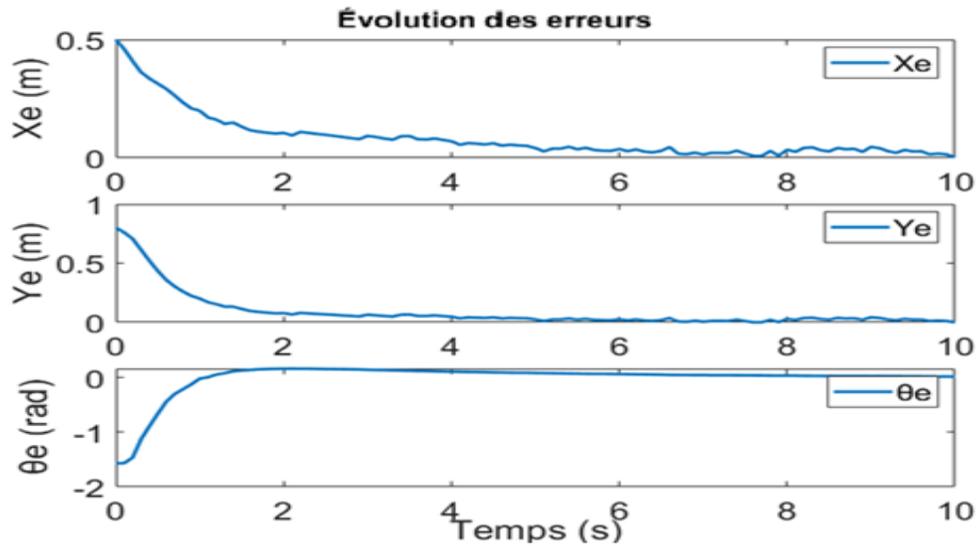


Figure 3. 33 Évolution des erreurs au cours du temps

Les figures 3.30, 3.31 et 3.32 illustrent les résultats expérimentaux du suivi de la trajectoire linéaire pour le robot mobile omnidirectionnel. Ces résultats indiquent le suivi de la trajectoire du contrôleur cinématique du régulateur proposé. On remarque que la réponse du régulateur PI est lent et l'erreur de position est un peu grande.

- Trajectoire circulaire :

Pour vérifier la performance de PI, on a essayé deux (2) vitesses différentes :

- Avec une petite vitesse

L'équation de la trajectoire est de la forme :

$$\begin{cases} X_d = X_{d0} + r \cos(\omega_0 + \omega_r t) \\ Y_d = Y_{d0} + r \sin(\omega_0 + \omega_r t) \\ \theta_d = \frac{\pi}{4} \end{cases}$$

Avec :  $\omega_0 = 0$ ;  $\omega_r = 0.2$ ;  $r = 0.5$ ;  $X_{d0} = 0$ ;  $Y_{d0} = 0$

Les positions initiales du robot sont:  $\begin{cases} X_0 = 0.5 \\ Y_0 = 0.5 \\ \theta_0 = 0 \end{cases}$

La courbe suivante représente la variation de la position du robot le long de la trajectoire circulaire dans le plan XY pour une petite vitesse en utilisant le régulateur à base PI.

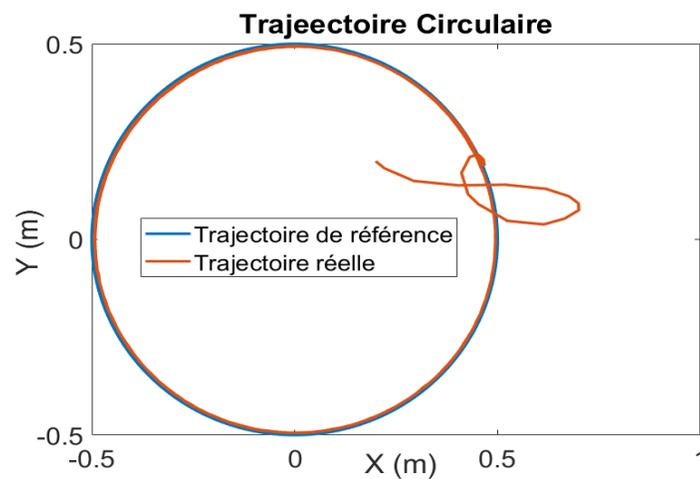


Figure 3. 34 Trajectoire circulaire du robot dans le plan XY

Le courbe suivant représente l'évolution des erreurs des positions au cours du temps pour la trajectoire circulaire pour une petite vitesse en utilisant le contrôleur PI.

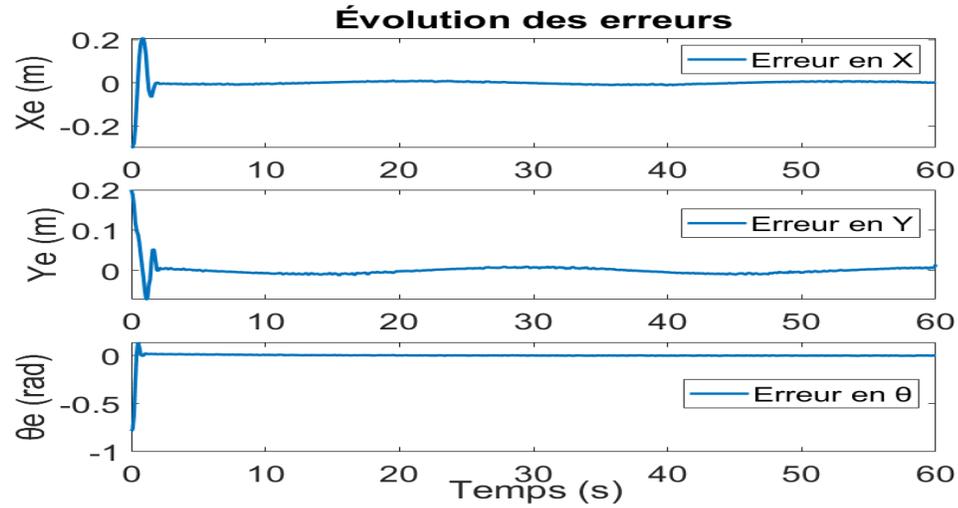


Figure 3.35 Évolution des erreurs au cours du temps

➤ Avec une vitesse plus grande

L'équation de la trajectoire est de la forme :

$$\begin{cases} X_d = X_{d0} + r \cos(\omega_0 + \omega_r t) \\ Y_d = Y_{d0} + r \sin(\omega_0 + \omega_r t) \\ \theta_d = \frac{\pi}{4} \end{cases}$$

Avec :  $\omega_0 = 0.5$ ;  $\omega_r = 1$ ;  $r = 0.5$ ;  $X_{d0} = 0$ ;  $Y_{d0} = 0$

Les positions initiales du robot sont:  $\begin{cases} X_0 = 0.5 \\ Y_0 = 0.5 \\ \theta_0 = 0 \end{cases}$

La courbe suivante représente la variation de la position du robot le long de la trajectoire circulaire dans le plan XY pour une grande vitesse en utilisant le régulateur à base PI.

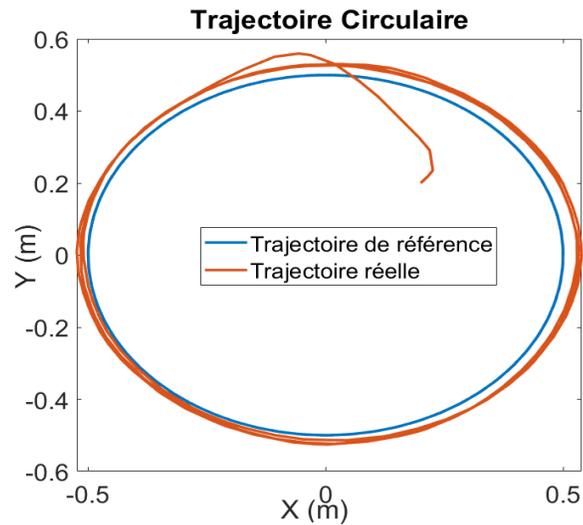


Figure 3. 36 Trajectoire circulaire du robot dans le plan XY

Le courbe suivant représente l'évolution des positions de référence et les positions réelles du robot en fonction du temps pour la trajectoire circulaire en utilisant le contrôleur PI.

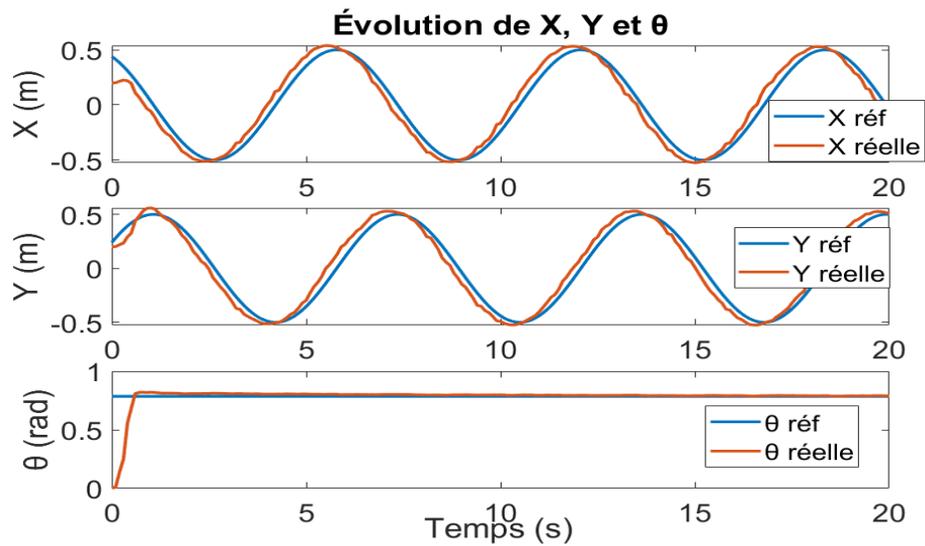


Figure 3. 37 Évolution des composantes du robot au cours du temps

Le courbe suivant représente l'évolution des erreurs des positions au cours du temps pour la trajectoire circulaire pour une grande vitesse en utilisant le contrôleur PI.

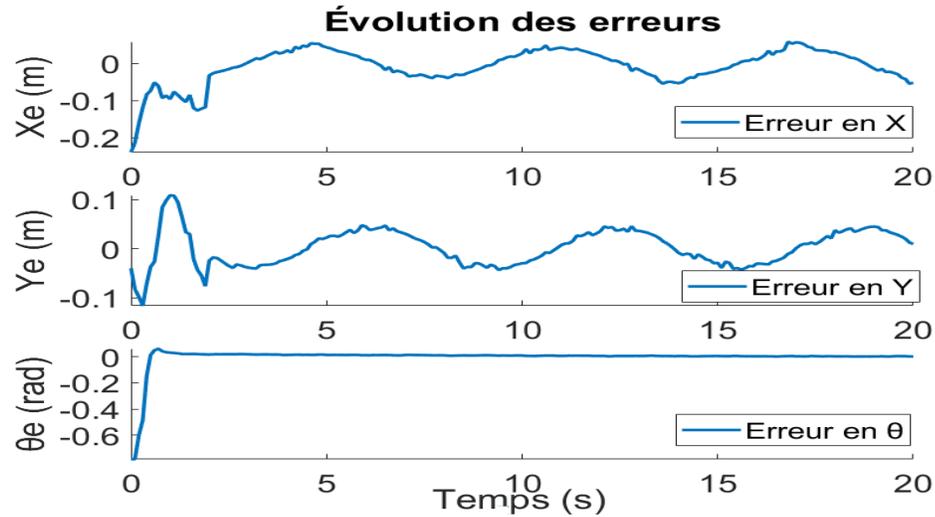


Figure 3.38 Évolution des erreurs au cours du temps

Les figures 3.36, 3.37 et 3.38 illustrent les résultats expérimentaux du suivi de la trajectoire circulaire pour le robot mobile omnidirectionnel. Ces résultats indiquent le suivi de la trajectoire du contrôleur cinématique du régulateur proposé. On remarque que la réponse du régulateur PI est lent et l'erreur de position est un peu grande.

On remarque que les régulateur PI est limité si on utilise une grande vitesse.

- Trajectoire elliptique :

L'équation de la trajectoire est de la forme :

$$\begin{cases} X_d = X_{d0} + r_1 \cos(\omega_0 + \omega_r t) \\ Y_d = Y_{d0} + r_2 \sin(\omega_0 + \omega_r t) \\ \theta_d = \frac{\pi}{4} \end{cases}$$

Avec :  $\omega_0 = 0.1$ ;  $\omega_r = 1$ ;  $r_1 = 0.6$ ;  $r_2 = 0.3$   $X_{d0} = 0$  et  $Y_{d0} = 0$

Les positions initiales du robot sont: 
$$\begin{cases} X_0 = 0 \\ Y_0 = 0 \\ \theta_0 = 0 \end{cases}$$

La courbe suivante représente la variation de la position du robot le long de la trajectoire elliptique dans le plan XY en utilisant le régulateur à base PI.

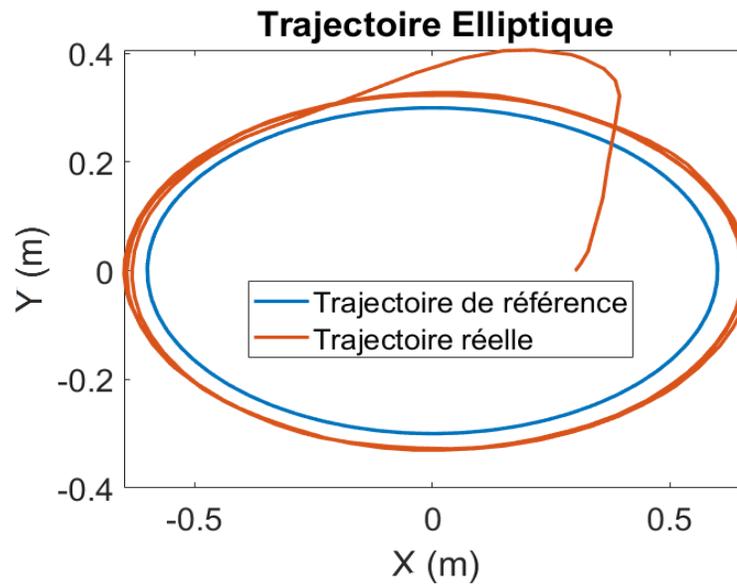


Figure 3. 39 Trajectoire elliptique du robot dans le plan XY

Le courbe suivant représente l'évolution des positions de référence et les positions réelles du robot en fonction du temps pour la trajectoire elliptique en utilisant le contrôleur PI.

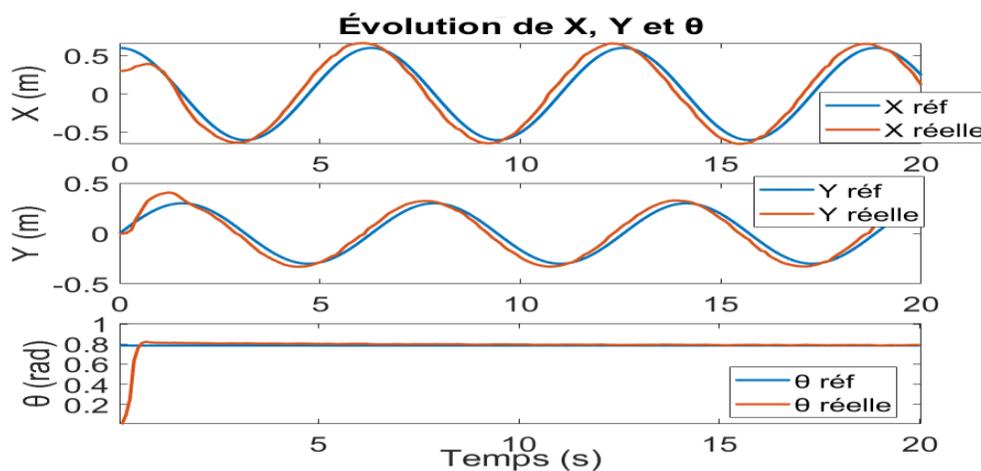


Figure 3. 40 Évolution des composantes du robot au cours du temps

Le courbe suivant représente l'évolution des erreurs des positions au cours du temps pour la trajectoire elliptique en utilisant le contrôleur PI.

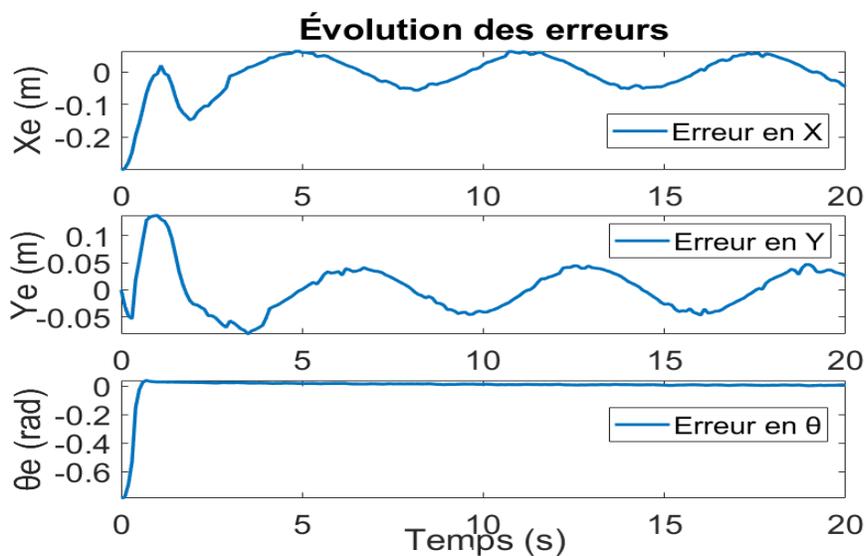


Figure 3. 41 Évolution des erreurs au cours du temps

Les figures 3.39, 3.40 et 3.41 illustrent les résultats expérimentaux du suivi de la trajectoire elliptique pour le robot mobile omnidirectionnel. Ces résultats indiquent le

suivi de la trajectoire du contrôleur cinématique du régulateur proposé. On remarque que la réponse du régulateur PI est lente et l'erreur de position est un peu grande.

### 3.3.2 En utilisant le régulateur de réseau de neurones entraîné hors ligne

Dans cette partie, on présente les résultats expérimentaux des (3) trajectoires différentes en utilisant le régulateur du réseau de neurones.

Avec  $\eta = 0.3$  et  $\beta = 0.7$

- Trajectoire linéaire :

La courbe suivante représente la variation de la position du robot le long de la trajectoire linéaire dans le plan XY en utilisant le régulateur de réseau de neurones entraîné hors ligne.

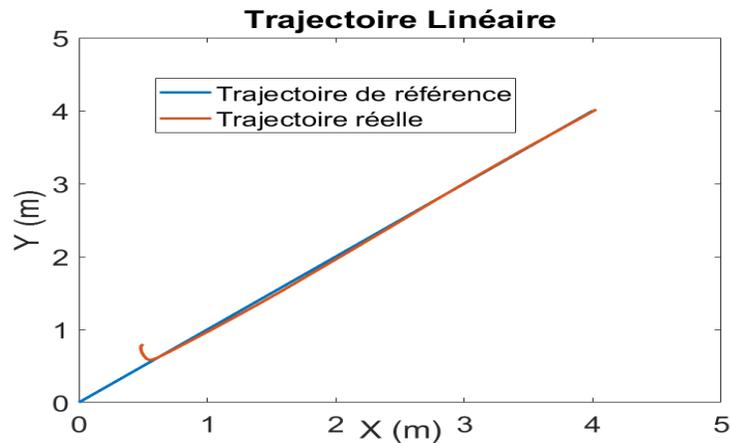


Figure 3. 42 Trajectoire linéaire du robot dans le plan XY

La courbe suivante représente l'évolution des positions de référence et les positions réelles du robot en fonction du temps pour la trajectoire linéaire en utilisant le contrôleur de réseau de neurones entraîné hors ligne.

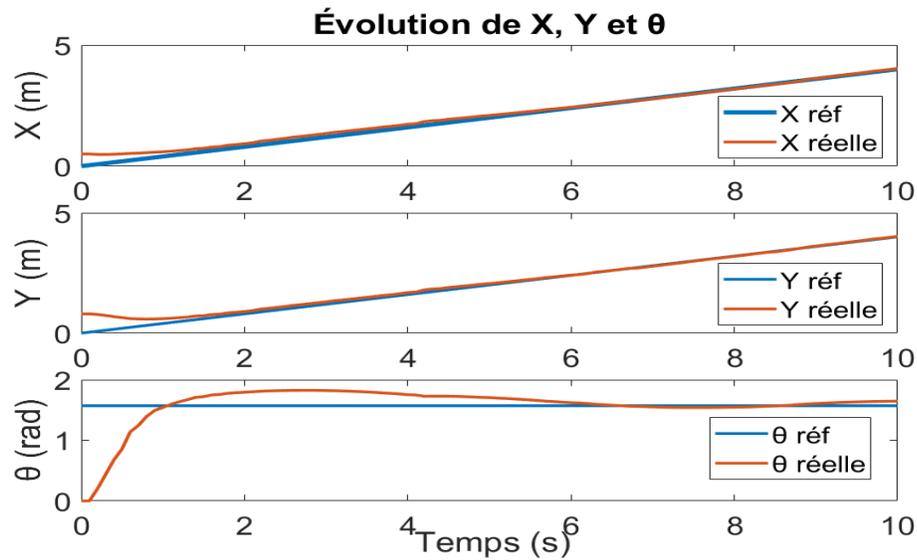


Figure 3. 43 Évolution des composantes du robot dans le plan XY

Le courbe suivant représente l'évolution des erreurs des positions au cours du temps pour la trajectoire linéaire en utilisant le contrôleur de réseau de neurones entraîné hors ligne.

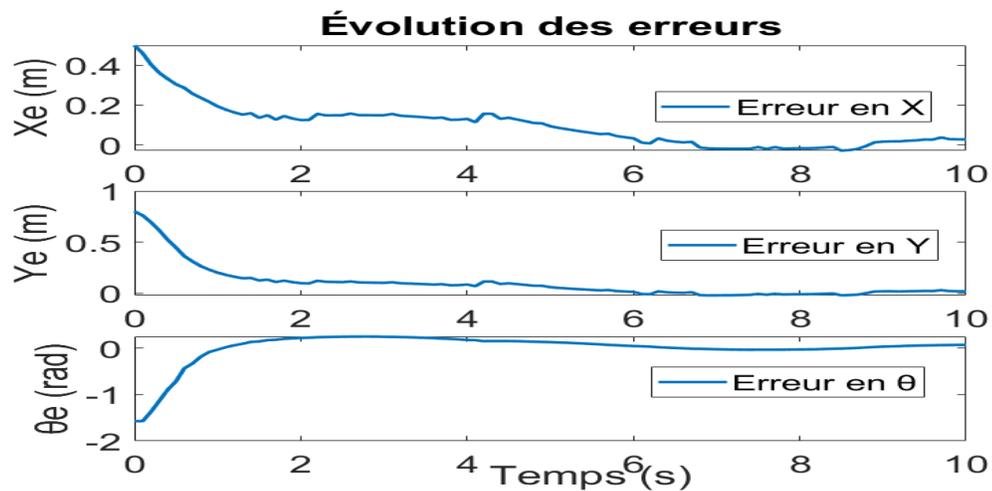


Figure 3. 44 Évolution des erreurs au cours du temps

Les figures 3.42, 3.43 et 3.44 illustrent les résultats expérimentaux du suivi de la trajectoire linéaire pour le robot mobile omnidirectionnel. Ces résultats indiquent le suivi de la trajectoire du contrôleur cinématique du régulateur proposé. On remarque que la réponse du régulateur de réseau de neurones entraîné hors ligne est lente et l'erreur de position est un peu grande, ces résultats sont très proches aux résultats de régulateurs PI.

- Trajectoire circulaire :

La courbe suivante représente la variation de la position du robot le long de la trajectoire circulaire dans le plan XY en utilisant le régulateur de réseau de neurones entraîné hors ligne.

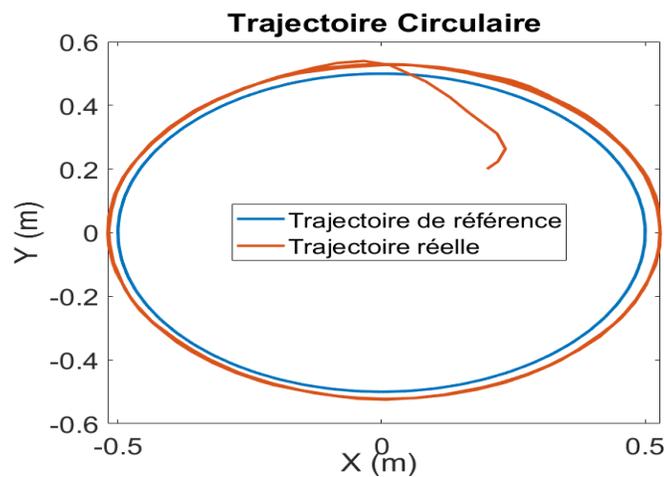


Figure 3. 45 Trajectoire circulaire du robot dans le plan XY

La courbe suivante représente l'évolution des positions de référence et les positions réelles du robot en fonction du temps pour la trajectoire circulaire en utilisant le contrôleur de réseau de neurones entraîné hors ligne.

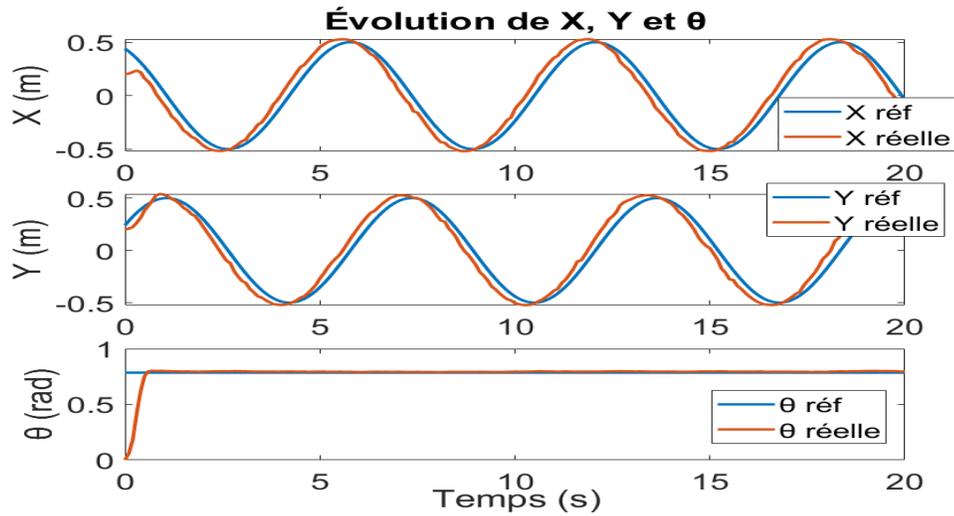


Figure 3. 46 Évolution des composantes du robot dans le plan XY

Le courbe suivant représente l'évolution des erreurs des positions au cours du temps pour la trajectoire circulaire en utilisant le contrôleur de réseau de neurones entraîné hors ligne.

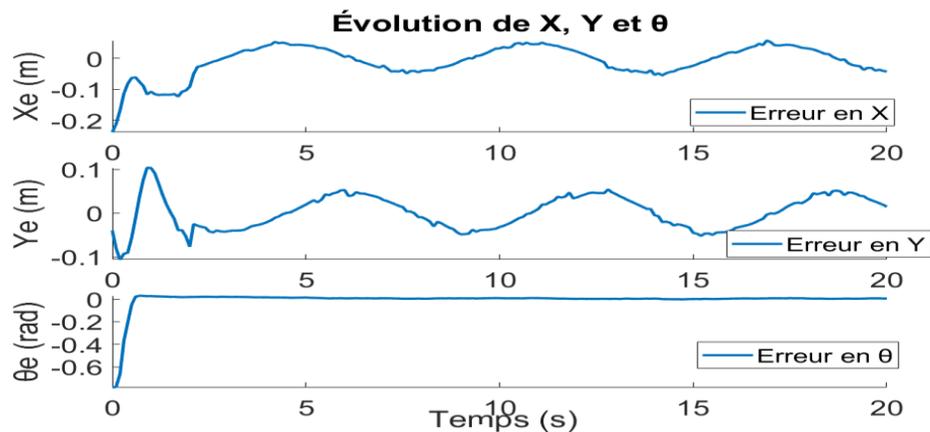


Figure 3. 47 Évolution des erreurs au cours du temps

Les figures 3.45, 3.46 et 3.47 illustrent les résultats expérimentaux du suivi de la trajectoire circulaire pour le robot mobile omnidirectionnel. Ces résultats indiquent le suivi de la trajectoire du contrôleur cinématique du régulateur proposé. On remarque

que la réponse du régulateur de réseau de neurones entraîné hors ligne est lente et l'erreur de position est un peu grande, ces résultats sont très proches aux résultats de régulateurs PI.

- Trajectoire elliptique :

La courbe suivante représente la variation de la position du robot le long de la trajectoire elliptique dans le plan XY en utilisant le régulateur de réseau de neurones entraîné hors ligne.

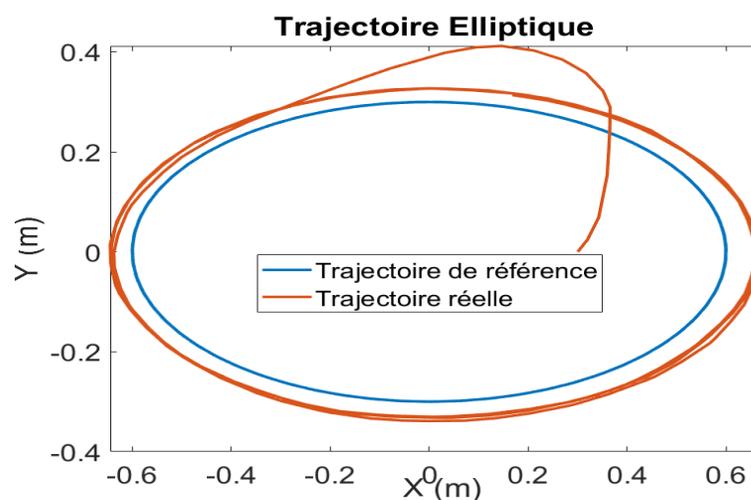


Figure 3. 48 Trajectoire elliptique du robot dans le plan XY

La courbe suivante représente l'évolution des positions de référence et les positions réelles du robot en fonction du temps pour la trajectoire elliptique en utilisant le contrôleur de réseau de neurones entraîné hors ligne.

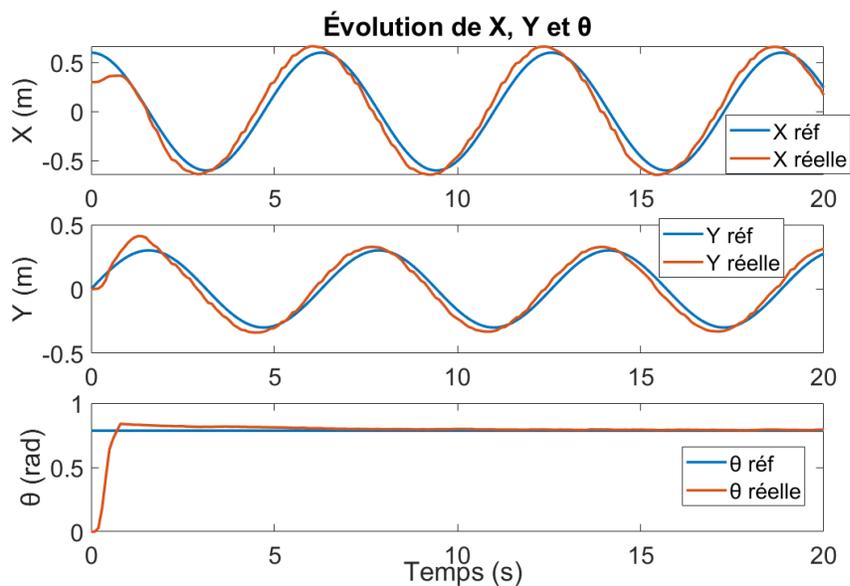


Figure 3.49 Évolution des composantes du robot au cours du temps

Le courbe suivant représente l'évolution des erreurs des positions au cours du temps pour la trajectoire elliptique en utilisant le contrôleur de réseau de neurones entraîné hors ligne.

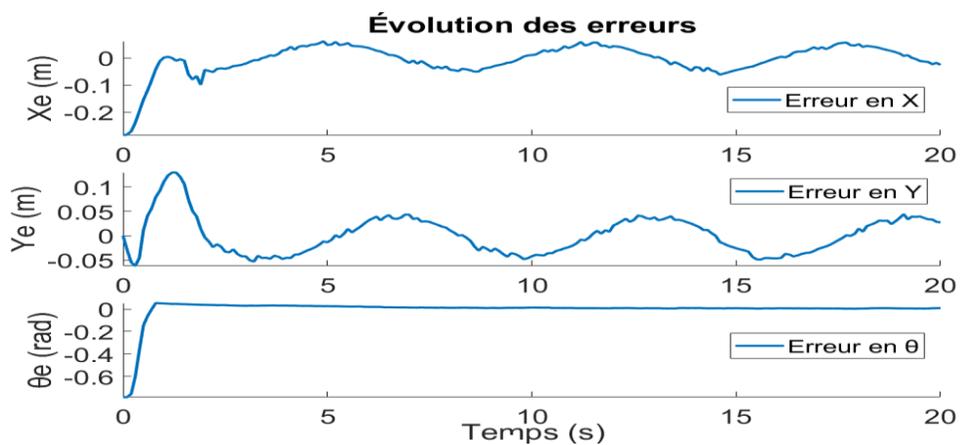


Figure 3.50 Évolution des erreurs au cours du temps

Les figures 3.48, 3.49 et 3.50 illustrent les résultats expérimentaux du suivi de la trajectoire elliptique pour le robot mobile omnidirectionnel. Ces résultats indiquent le suivi de la trajectoire du contrôleur cinématique du régulateur proposé. On remarque que la réponse du régulateur de réseau de neurones entraîné hors ligne est lente et l'erreur de position est un peu grande, ces résultats sont très proches aux résultats de régulateurs PI.

### 3.3.3 En utilisant le régulateur PI et réseau de neurones entraîné en ligne

Dans cette partie, on présente les résultats expérimentaux de 3 trajectoires différentes en combinant le régulateur à base PI et le réseau de neurones.

Avec  $\eta = 0.3$  et  $\beta = 0.7$

- Trajectoire linéaire :

La courbe suivante représente la variation de la position du robot le long de la trajectoire linéaire dans le plan XY en utilisant le régulateur PI combiné avec le réseau de neurones entraîné en ligne.

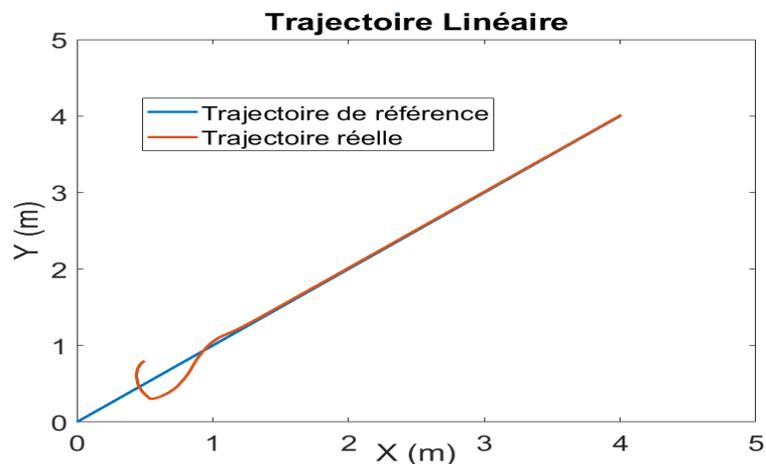


Figure 3. 51 Trajectoire linéaire du robot dans le plan XY

La courbe suivante représente l'évolution des positions de référence et les positions réelles du robot en fonction du temps pour la trajectoire linéaire en utilisant le contrôleur PI combiné avec le réseau de neurones entraîné en ligne.

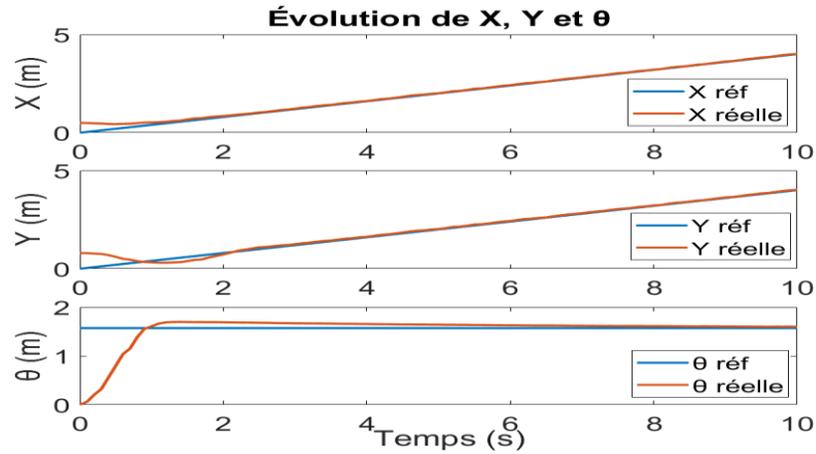


Figure 3. 52 Évolution des composantes du robot au cours du temps

Le courbe suivant représente l'évolution des erreurs des positions au cours du temps pour la trajectoire linéaire en utilisant le contrôleur PI combiné avec le réseau de neurones entraîné en ligne.

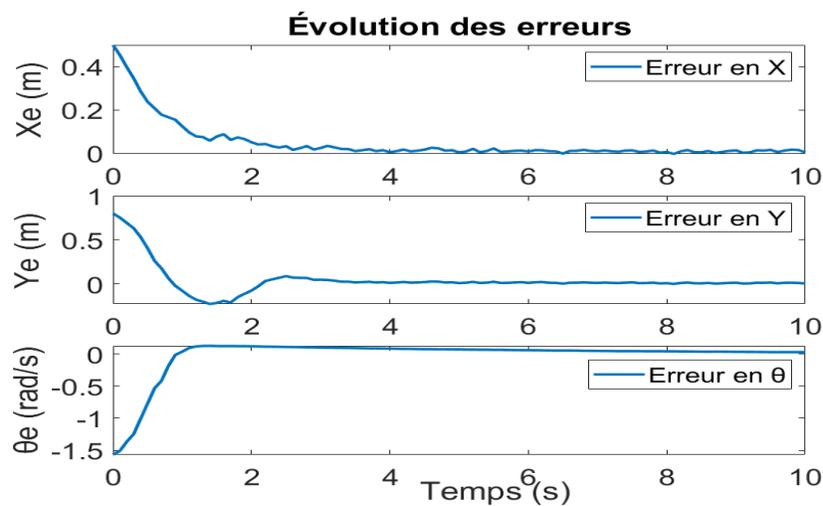


Figure 3. 53 Évolution des erreurs au cours du temps

Les figures 3.51, 3.52 et 3.53 illustrent les résultats expérimentaux du suivi de la trajectoire linéaire pour le robot mobile omnidirectionnel. Ces résultats indiquent le suivi de la trajectoire du contrôleur cinématique du régulateur proposé. On remarque que la réponse du régulateur de la combinaison PI et de réseau de neurones entraîné en ligne est plus rapide que celle de PI et de Réseau de neurones entraîné hors ligne. Ce régulateur a réussi à minimiser l'erreur de position.

- Trajectoire circulaire :

La courbe suivante représente la variation de la position du robot le long de la trajectoire circulaire dans le plan XY en utilisant le régulateur PI combiné avec le réseau de neurones entraîné en ligne.

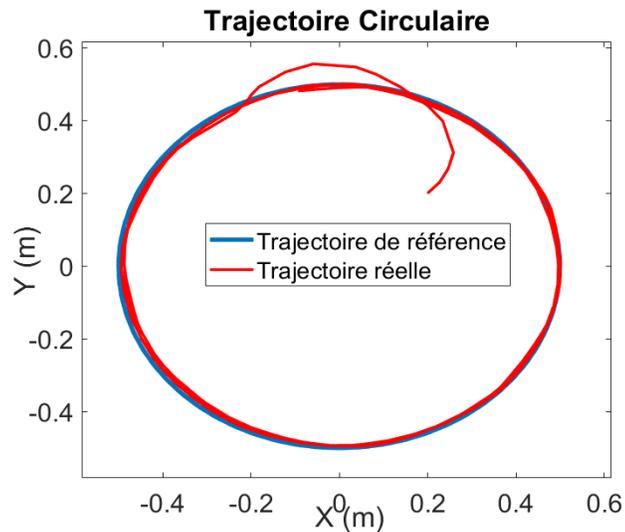


Figure 3.54 Trajectoire circulaire du robot dans le plan XY

La courbe suivante représente l'évolution des positions de référence et les positions réelles du robot en fonction du temps pour la trajectoire circulaire en utilisant le contrôleur PI combiné avec le réseau de neurones entraîné en ligne.

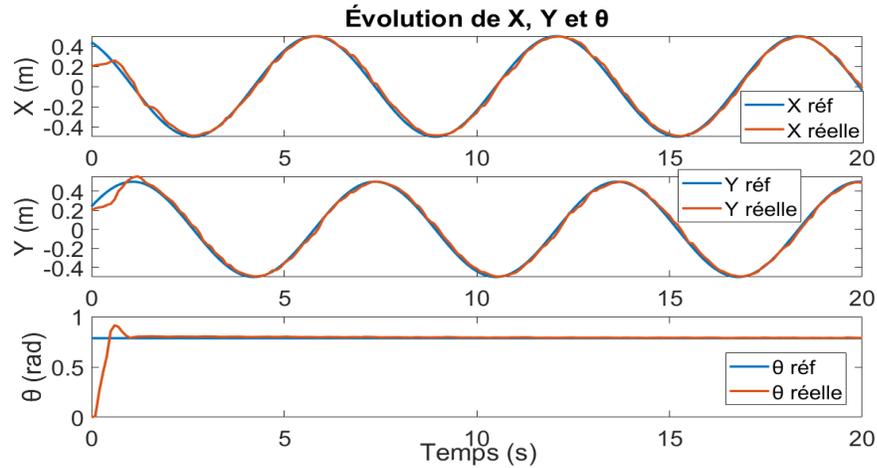


Figure 3.55 Évolution des composantes du robot au cours du temps

Le courbe suivant représente l'évolution des erreurs des positions au cours du temps pour la trajectoire circulaire en utilisant le contrôleur PI combiné avec le réseau de neurones entraîné en ligne.

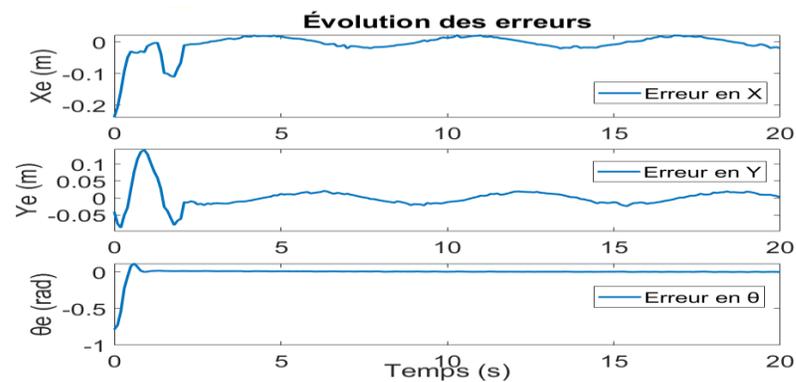


Figure 3. 56 Évolution des erreurs au cours du temps

Les figures 3.51, 3.52 et 3.53 illustrent les résultats expérimentaux du suivi de la trajectoire circulaire pour le robot mobile omnidirectionnel. Ces résultats indiquent le suivi de la trajectoire du contrôleur cinématique du régulateur proposé. On remarque que la réponse du régulateur de la combinaison PI et de réseau de neurones entraîné en

ligne est plus rapide que celle de PI et de Réseau de neurones entraîné hors ligne. Ce régulateur a réussi à minimiser l'erreur de position.

- Trajectoire elliptique :

La courbe suivante représente la variation de la position du robot le long de la trajectoire elliptique dans le plan XY en utilisant le régulateur PI combiné avec le réseau de neurones entraîné en ligne.

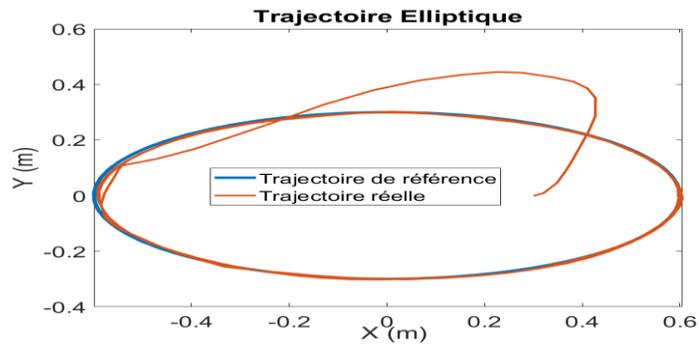


Figure 3. 57 Trajectoire elliptique du robot dans le plan XY

La courbe suivante représente l'évolution des positions de référence et les positions réelles du robot en fonction du temps pour la trajectoire elliptique en utilisant le contrôleur PI combiné avec le réseau de neurones entraîné en ligne.

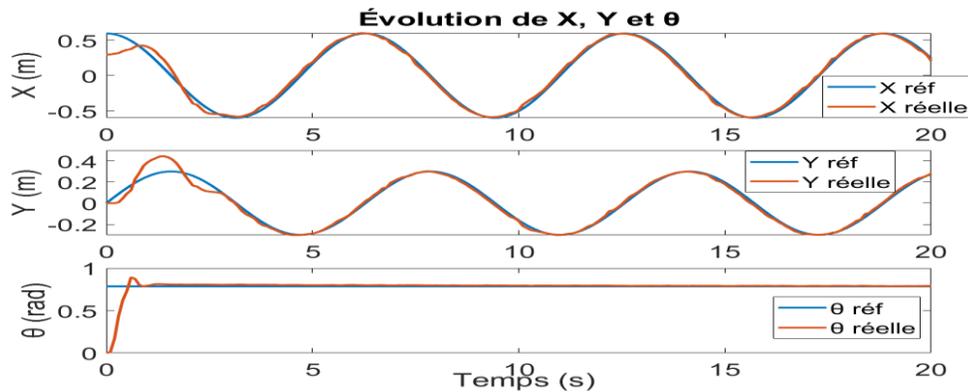


Figure 3. 58 Évolution des composantes du robot au cours du temps

Le courbe suivant représente l'évolution des erreurs des positions au cours du temps pour la trajectoire elliptique en utilisant le contrôleur PI combiné avec le réseau de neurones entraîné en ligne.

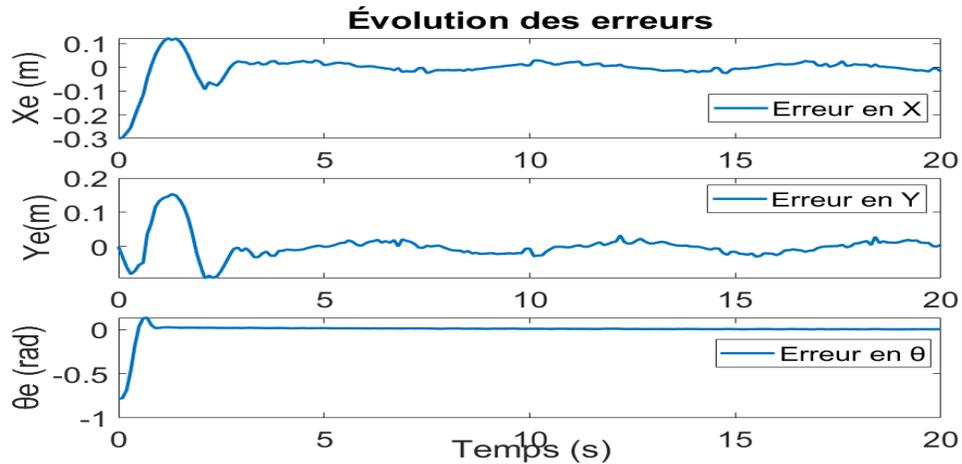


Figure 3. 59 Évolution des erreurs au cours du temps

Les figures 3.57, 3.58 et 3.59 illustrent les résultats expérimentaux du suivi de la trajectoire linéaire pour le robot mobile omnidirectionnel. Ces résultats indiquent le suivi de la trajectoire du contrôleur cinématique du régulateur proposé. On remarque que la réponse du régulateur de la combinaison PI et de réseau de neurones entraîné en ligne est plus rapide que celle de PI et de Réseau de neurones entraîné hors ligne. Ce régulateur a réussi à minimiser l'erreur de position.

### 3.3.4 Interprétation

Les figures suivantes représentent une comparaison entre les variations de la position du robot omnidirectionnel en utilisant les différents régulateurs pour 3 différentes trajectoires.

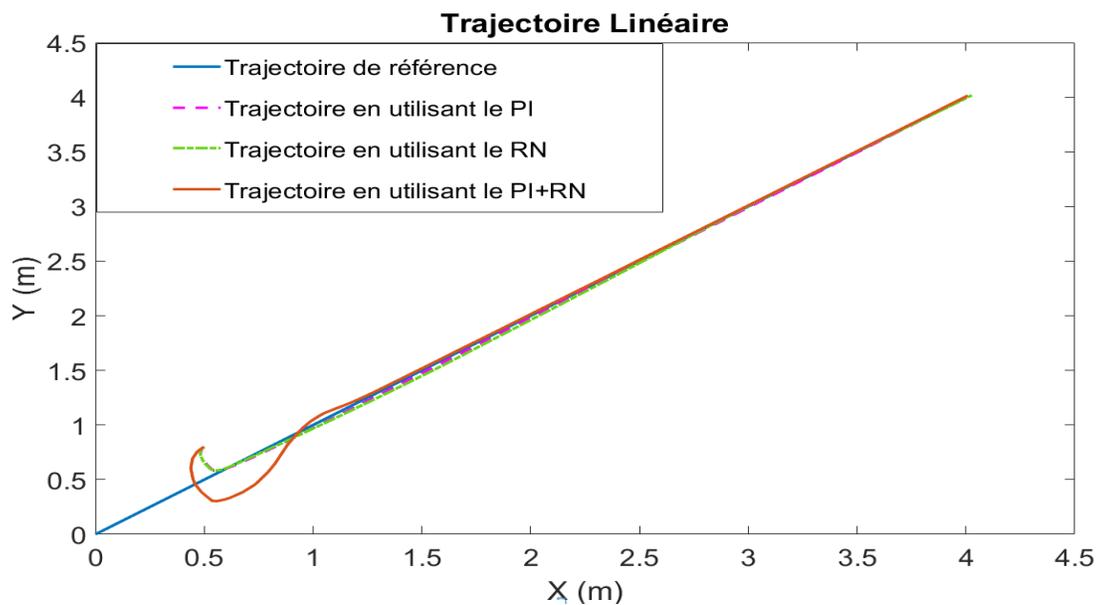


Figure 3. 60 Comparaison entre les 3 régulateurs pour une trajectoire linéaire

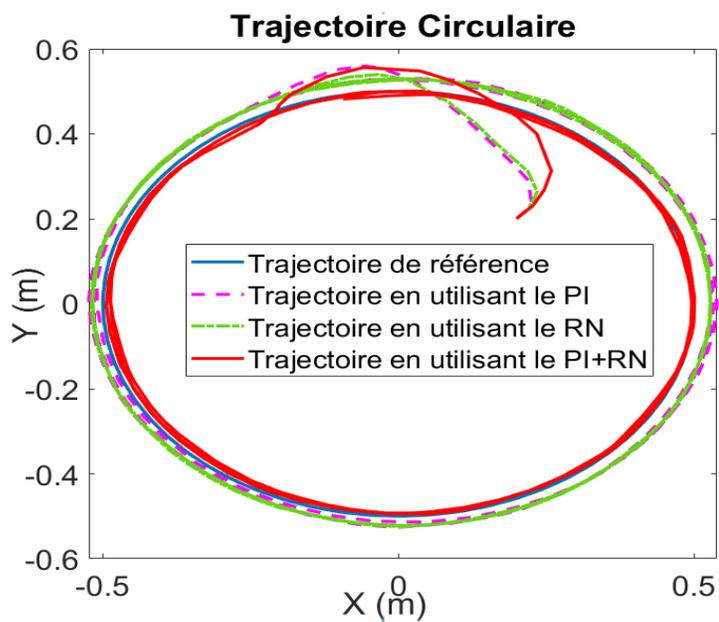


Figure 3. 61 Comparaison entre les 3 régulateurs pour une trajectoire circulaire

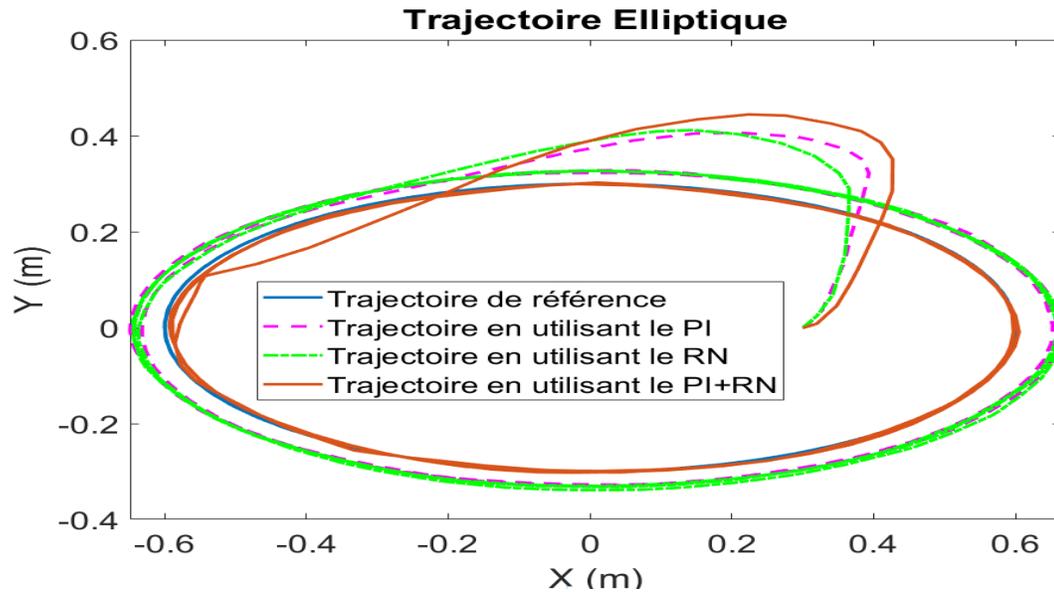


Figure 3. 62 Comparaison entre les 3 régulateurs pour une trajectoire elliptique

Les résultats expérimentaux prouvent les résultats de simulation:

- Le réseau de neurones entraîné hors ligne utilisé est un compensateur de régulateur à base PI et il peut le remplacer.
- La réponse du régulateur de la combinaison de PI et le réseau de neurones entraîné en ligne est plus rapide et donne des résultats plus précis et mieux que celle de PI seul.
- Le PI est limité: en augmentant les vitesses du robot, es erreurs deviennent plus grandes.

## CONCLUSIONS ET PERSPECTIVES

Ce projet porte une présentation de notre sujet de recherche intitulé « Suivi de trajectoire d'un robot mobile omnidirectionnel en utilisant le réseau de neurones ». Dans la première partie, nous avons fait une revue de littérature sur les robots mobiles, ainsi que quelques commandes des systèmes robotiques mobiles qui nous ont données une idée claire et précise sur le type de commande à utiliser. Ensuite nous avons fait une modélisation cinématique du robot étudié et nous avons élaboré les lois de commandes utilisées.

Un contrôleur de suivi de trajectoire basé sur le PI et deux contrôleurs différents basés sur des réseaux de neurones ont été proposés dans cet ouvrage. Nous avons fait une analyse comparative entre ces contrôleurs. Le régulateur à base PI prouve qu'il a des limites surtout en augmentant les vitesses de robot, les erreurs deviennent plus grandes. On a remplacé le PI par un régulateur de réseau de neurones entraîné hors ligne, ce dernier donne presque les mêmes résultats que le PI puisqu'il réagit de la même manière. Enfin on a combiné le régulateur à base PI avec un réseau de neurones entraîné en ligne. Ce régulateur a réussi à minimiser les erreurs de suivi de trajectoire et à augmenter les performances de régulateur à base PI.

Ce projet présenté a été satisfaisant mais il y a toujours des points à améliorer, il est possible d'utiliser d'autres entrées de réseau de neurones en essayant:

- Le réseau de neurones en modèle inverse
- Le réseau de neurones adaptatives

## RÉFÉRENCES

- [1] R. Siegwart and I. R. Nourbakhsh, "Introduction to Autonomous Mobile Robot, 1a edição," ed: MIT Press, Cambridge, Massachusetts, EUA, 2004.
- [2] C. L. Breazeal, Designing sociable robots. MIT press, 2004.
- [3] K. Kanjanawanishkul, "Omnidirectional wheeled mobile robots: wheel types and practical applications," International Journal of Advanced Mechatronic Systems, vol. 6, no. 6, pp. 289-302, 2015.
- [4] K. Shabalina, A. Sagitov, and E. Magid, "Comparative analysis of mobile robot wheels design," in 2018 11th International Conference on Developments in eSystems Engineering (DeSE), 2018: IEEE, pp. 175-179.
- [5] W. Y. Loong, L. Z. Long, and L. C. Hun, "A star path following mobile robot," in 2011 4th International conference on mechatronics (ICOM), 2011: IEEE, pp. 1-7.
- [6] Y. Ha and S. Yuta, "Trajectory tracking control for navigation of self-contained mobile inverse pendulum," in Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'94), 1994, vol. 3: IEEE, pp. 1875-1882.
- [7] F. Guerin, "Commande conjuguée d'un robot mobile: modélisation dynamique et vision artificielle," Le Havre, 2004.
- [8] G. Indiveri, "Swedish wheeled omnidirectional mobile robots: Kinematics analysis and control," IEEE transactions on robotics, vol. 25, no. 1, pp. 164-171, 2009.

- [9] C.-C. Tsai, L.-B. Jiang, T.-Y. Wang, and T.-S. Wang, "Kinematics control of an omnidirectional mobile robot," in Proceedings of 2005 CACS Automatic Control Conference Tainan, 2005, pp. 18-19.
- [10] D. Garcia-Sillas, E. Gorrostieta-Hurtado, J. Vargas, J. Rodríguez-Reséndiz, and S. Tovar, "Kinematics modeling and simulation of an autonomous omnidirectional mobile robot," *Ingeniería e Investigación*, vol. 35, no. 2, pp. 74-79, 2015.
- [11] Y. Zhao, N. Chen, and Y. Tai, "Trajectory tracking control of wheeled mobile robot based on fractional order backstepping," in 2016 Chinese Control and Decision Conference (CCDC), 2016: IEEE, pp. 6730-6734.
- [12] S.-H. Chen and J.-C. Juang, "Integrated backstepping mobile robot controller design by applying sum of squares approach," *SICE Journal of Control, Measurement, and System Integration*, vol. 3, no. 6, pp. 442-447, 2010.
- [13] S.-H. Chen, J.-C. Juang, and S.-H. Su, "Backstepping control with sum of squares design for omni-directional mobile robots," in 2009 ICCAS-SICE, 2009: IEEE, pp. 545-550.
- [14] C. Ren, X. Li, X. Yang, and S. Ma, "Extended state observer-based sliding mode control of an omnidirectional mobile robot with friction compensation," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 12, pp. 9480-9489, 2019.
- [15] S.-T. Kao, W.-J. Chiou, and M.-T. Ho, "Integral sliding mode control for trajectory tracking control of an omnidirectional mobile robot," in 2011 8th Asian Control Conference (ASCC), 2011: IEEE, pp. 765-770.
- [16] P. H. K. Khanh, N. T. Trung, P. T. Doan, and N. Hung, "Trajectory tracking control of omnidirectional mobile robot using sliding mode controller," in 2013 13th International Conference on Control, Automation and Systems (ICCAS 2013), 2013: IEEE, pp. 1170-1175.

- [17] N. Hacene and B. Mendil, "Fuzzy behavior-based control of three wheeled omnidirectional mobile robot," *International Journal of Automation and Computing*, vol. 16, no. 2, pp. 163-185, 2019.
- [18] H.-C. Huang and C.-H. Chiang, "Backstepping holonomic tracking control of wheeled robots using an evolutionary fuzzy system with qualified ant colony optimization," *International Journal of Fuzzy Systems*, vol. 18, no. 1, pp. 28-40, 2016.
- [19] S. Oltean, M. Dulău, and R. Puskas, "Position control of Robotino mobile robot using fuzzy logic," in *2010 IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR)*, 2010, vol. 1: IEEE, pp. 1-6.
- [20] J.-T. Huang, T. Van Hung, and M.-L. Tseng, "Smooth switching robust adaptive control for omnidirectional mobile robots," *IEEE Transactions on Control Systems Technology*, vol. 23, no. 5, pp. 1986-1993, 2015.
- [21] H.-C. Huang and C.-C. Tsai, "FPGA implementation of an embedded robust adaptive controller for autonomous omnidirectional mobile platform," *IEEE Transactions on Industrial Electronics*, vol. 56, no. 5, pp. 1604-1616, 2008.
- [22] H.-C. Huang and C.-C. Tsai, "Adaptive trajectory tracking and stabilization for omnidirectional mobile robot with dynamic effect and uncertainties," *IFAC Proceedings Volumes*, vol. 41, no. 2, pp. 5383-5388, 2008.
- [23] C. Treesatayapun, "A discrete-time stable controller for an omni-directional mobile robot based on an approximated model," *Control Engineering Practice*, vol. 19, no. 2, pp. 194-203, 2011.
- [24] C.-C. Tsai, Y.-S. Chen, and F.-C. Tai, "Intelligent adaptive distributed consensus formation control for uncertain networked heterogeneous Swedish-wheeled omnidirectional multi-robots," in *2016 55th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE)*, 2016: IEEE, pp. 154-159.
- [25] T. D. Viet, P. T. Doan, N. Hung, H. K. Kim, and S. B. Kim, "Tracking control of a three-wheeled omnidirectional mobile manipulator system with

disturbance and friction," *Journal of mechanical science and technology*, vol. 26, no. 7, pp. 2197-2211, 2012.

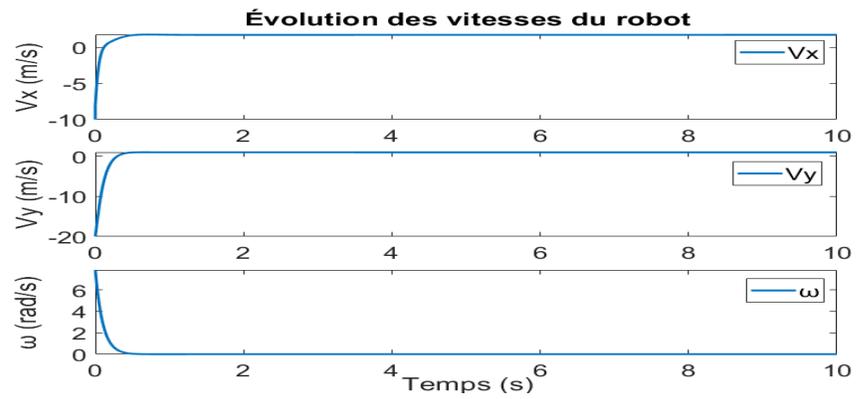
[26] <https://intelligence-artificielle-robotique.weebly.com>

[27] ISN - Robotique - Constituants d'un robot (silanus.fr)

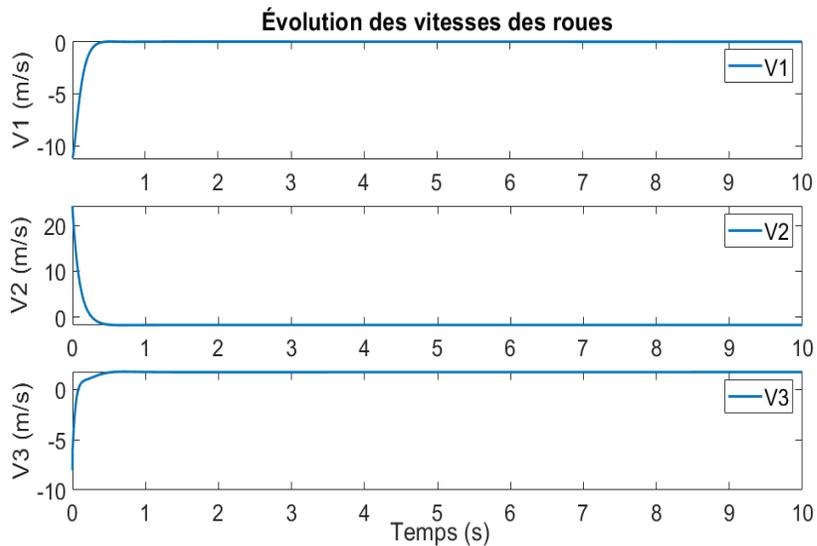
[28] <https://www.festo.com/>

[29] P.J. Braspenning, F. Thuijsman, A.J.M.M. Weijters, "An Introduction to ANN Theory and Practice," *Spring Science & Business Media*, Jun2, 1995.

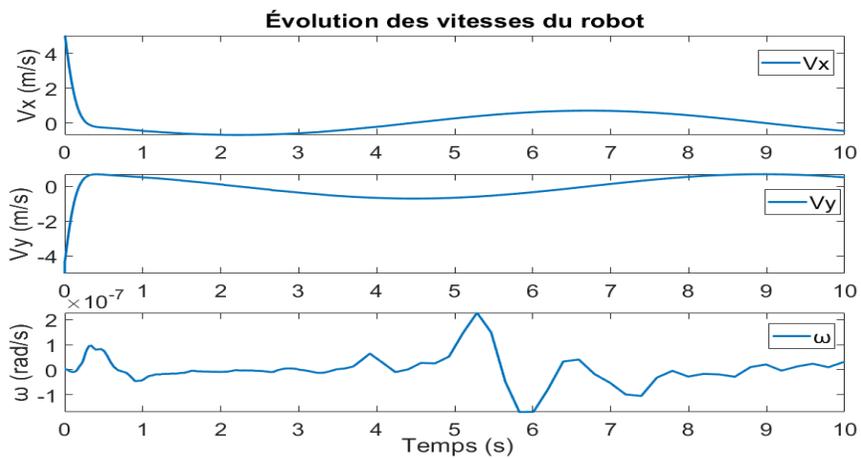
## ANNEXES



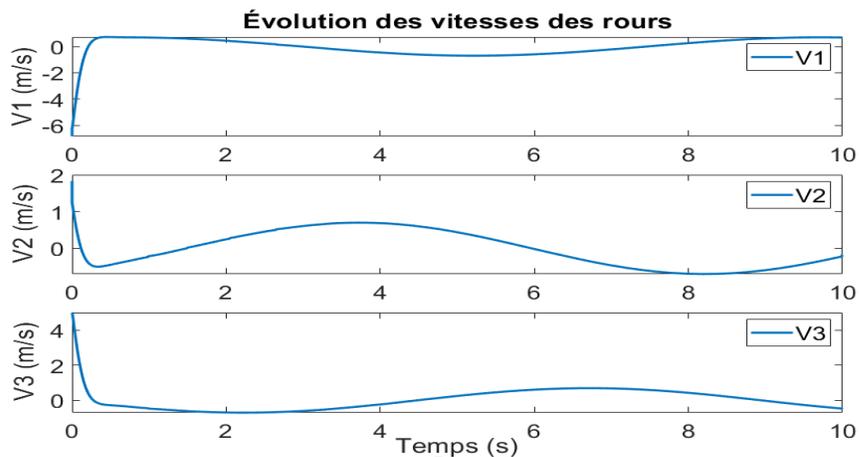
Évolution des vitesses du robot au cours du temps en utilisant le PI pour une trajectoire linéaire avec SIMULINK



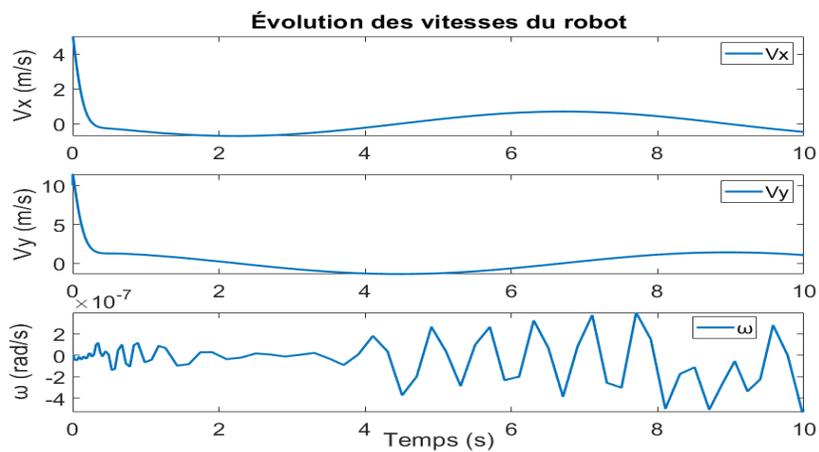
Évolution des vitesses des roues au cours du temps en utilisant le PI pour une trajectoire linéaire avec SIMULINK



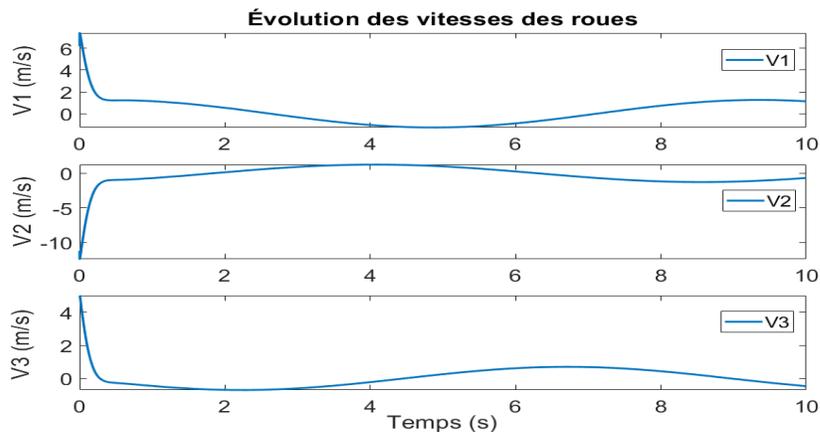
Évolution des vitesses du robot au cours du temps en utilisant le PI pour une trajectoire circulaire avec SIMULINK



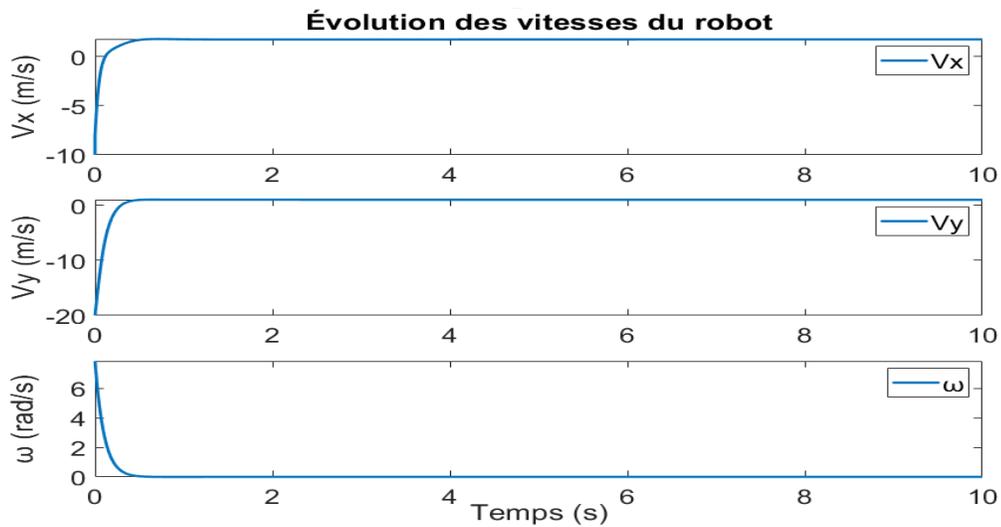
Évolution des vitesses des roues au cours du temps en utilisant le PI pour une trajectoire circulaire avec SIMULINK



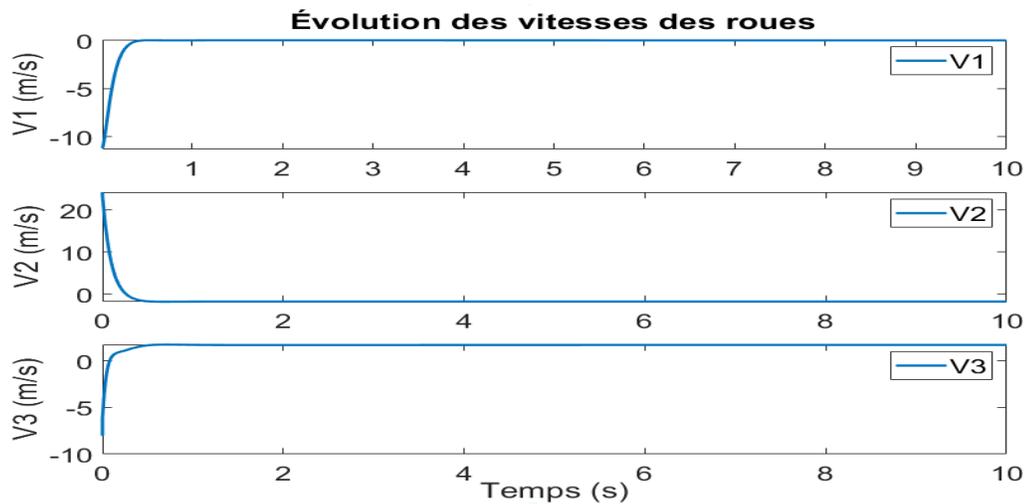
Évolution des vitesses du robot au cours du temps en utilisant le PI pour une trajectoire elliptique avec SIMULINK



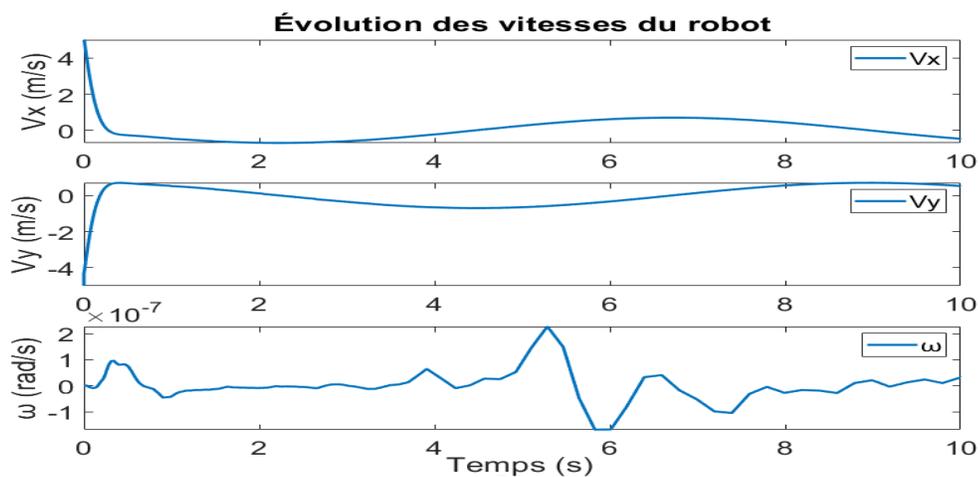
Évolution des vitesses des roues au cours du temps en utilisant le PI pour une trajectoire elliptique avec SIMULINK



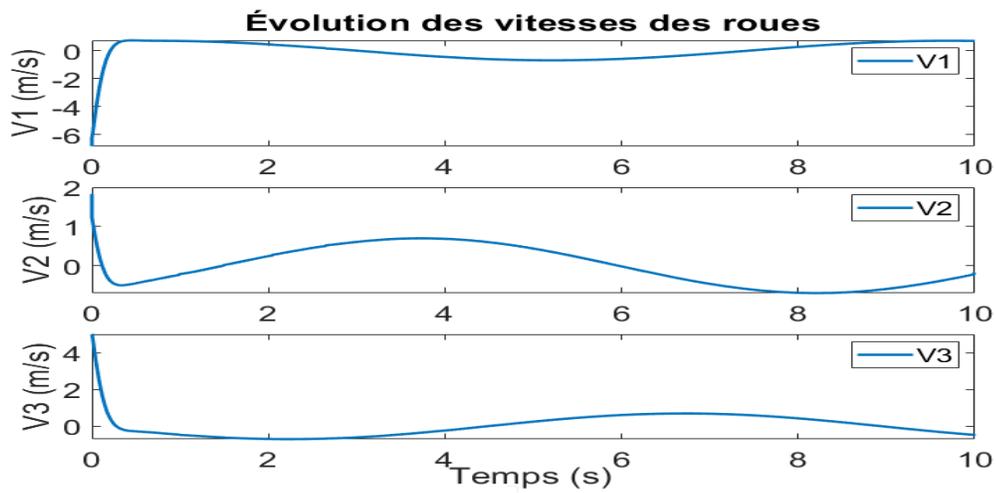
Évolution des erreurs du robot au cours du temps en utilisant le RN pour une trajectoire linéaire avec SIMULINK



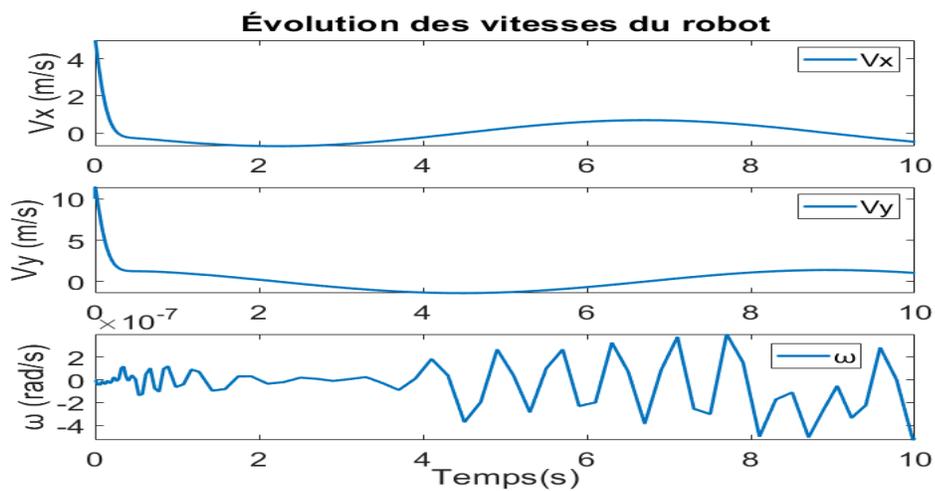
Évolution des vitesses des roues au cours du temps en utilisant le RN pour une trajectoire linéaire avec SIMULINK



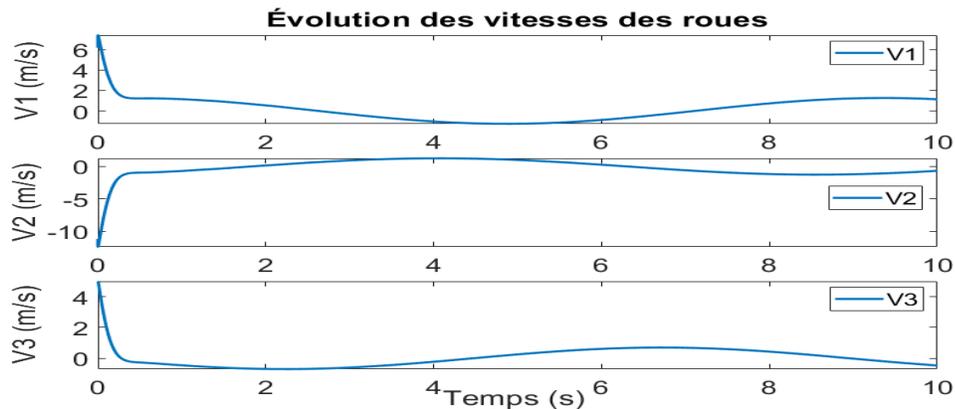
Évolution des vitesses du robot au cours du temps en utilisant le RN pour une trajectoire circulaire avec SIMULINK



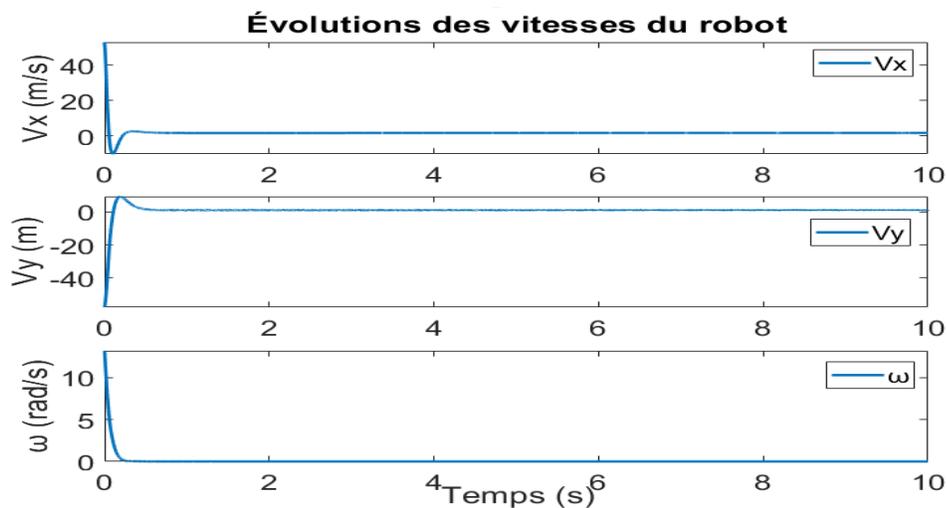
Évolution des vitesses des roues au cours du temps en utilisant le RN pour une trajectoire circulaire avec SIMULINK



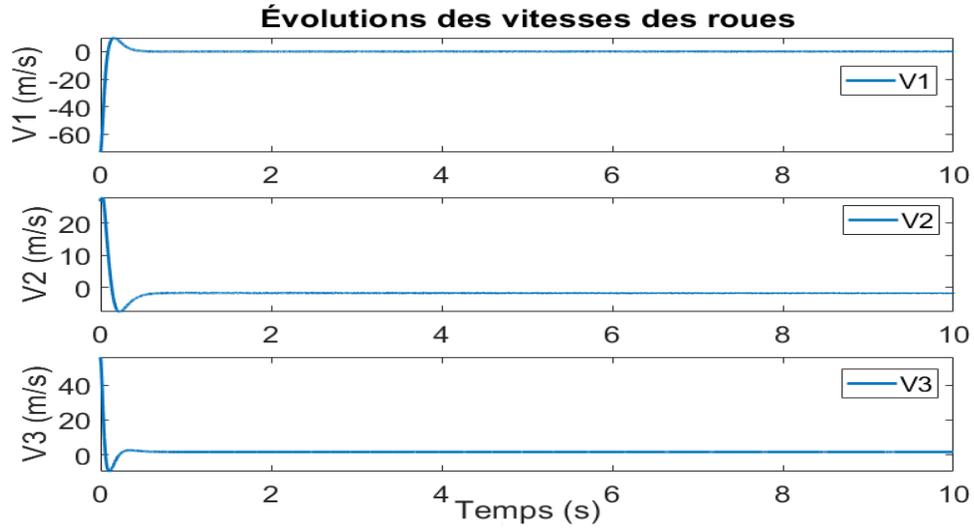
Évolution des vitesses du robot au cours du temps en utilisant le RN pour une trajectoire elliptique avec SIMULINK



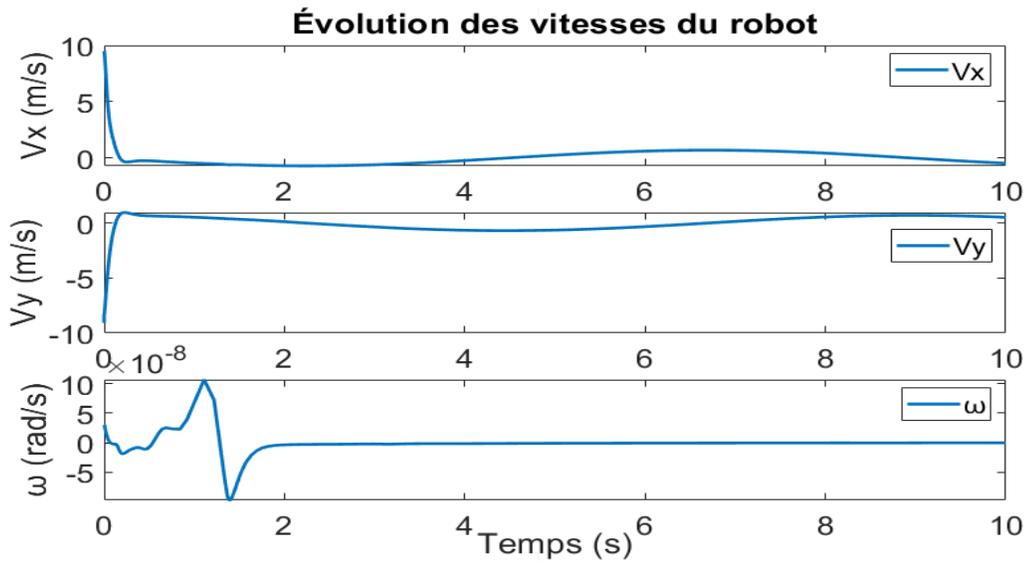
Évolution des vitesses des roues au cours du temps en utilisant le RN pour une trajectoire elliptique avec SIMULINK



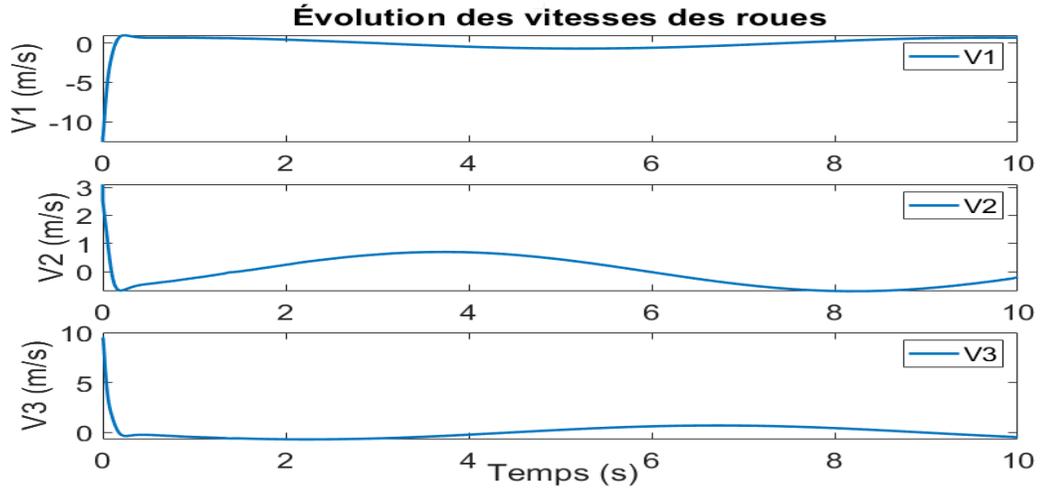
Évolution des vitesses du robot au cours du temps en utilisant le PI et le RN pour une trajectoire linéaire avec SIMULINK



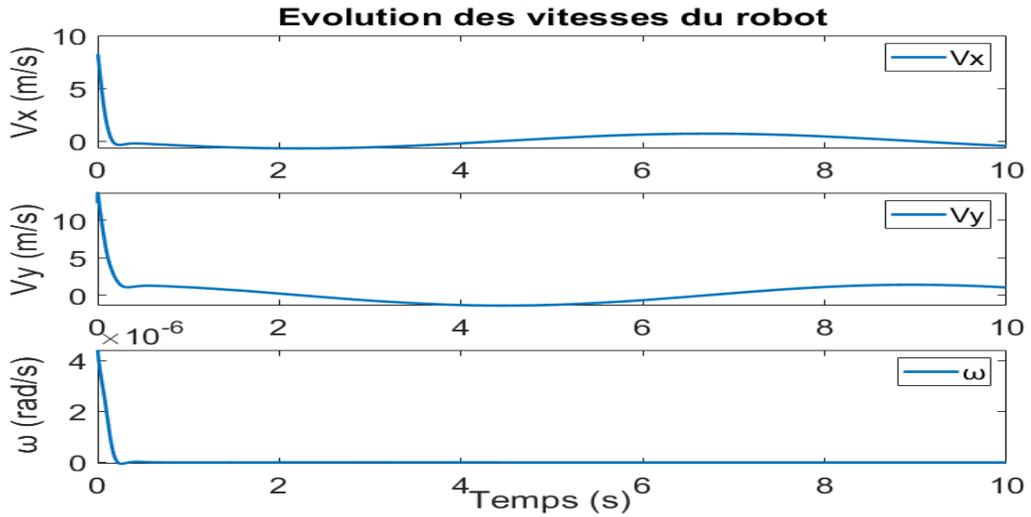
Évolution des vitesses des roues au cours du temps en utilisant le PI et le RN pour une trajectoire linéaire avec SIMULINK



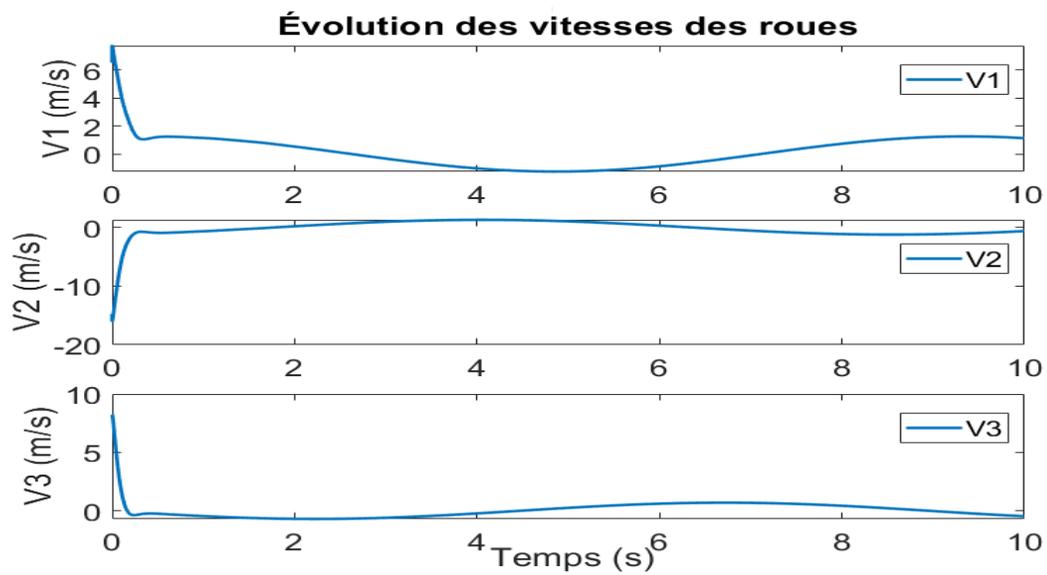
Évolution des vitesses du robot au cours du temps en utilisant le PI et le RN pour une trajectoire circulaire avec SIMULINK



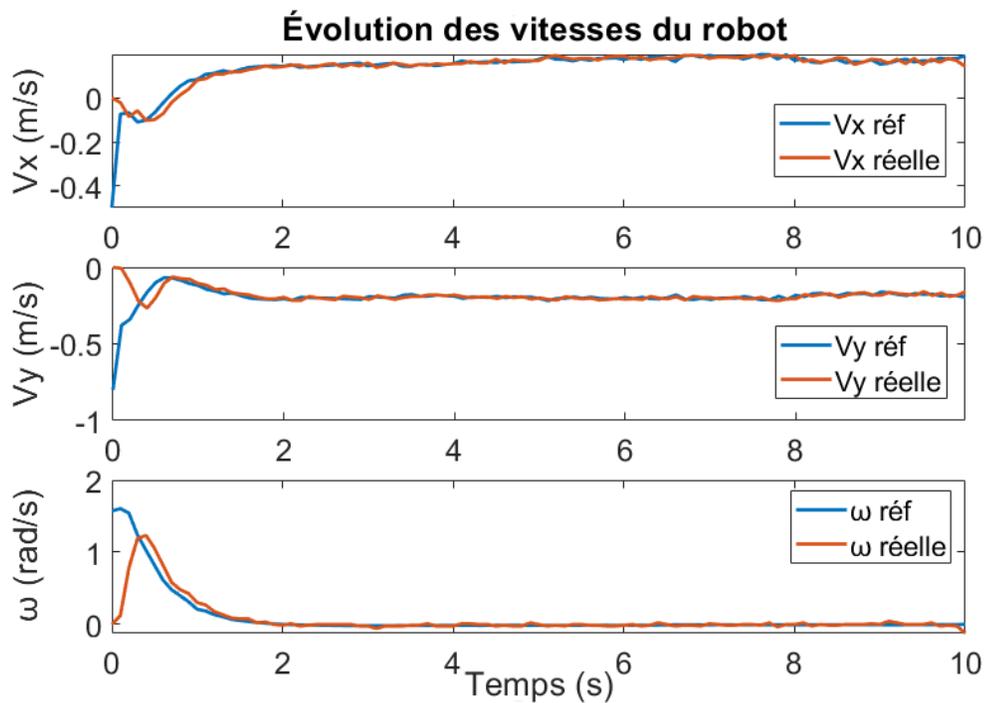
Évolution des vitesses des roues au cours du temps en utilisant le PI et le RN pour une trajectoire circulaire avec SIMULINK



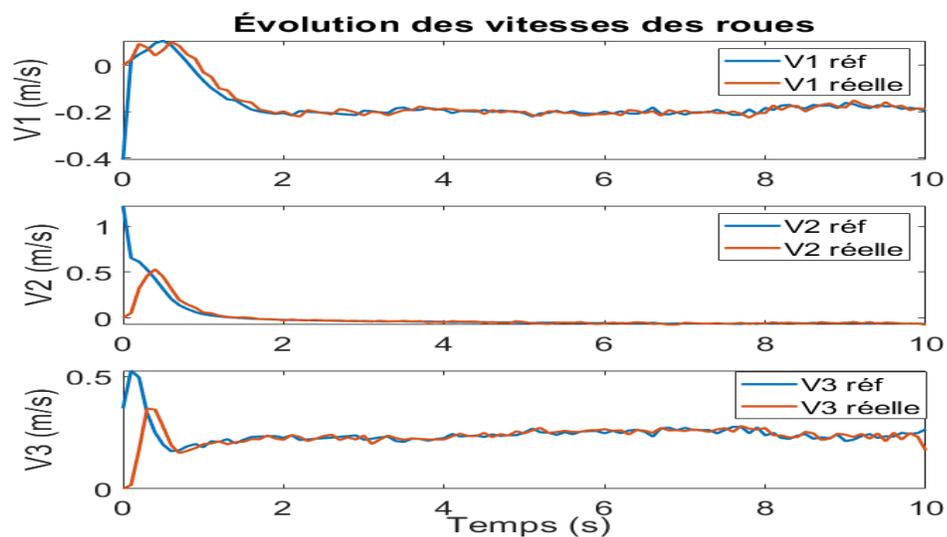
Évolution des vitesses du robot au cours du temps en utilisant le PI et le RN pour une trajectoire elliptique avec SIMULINK



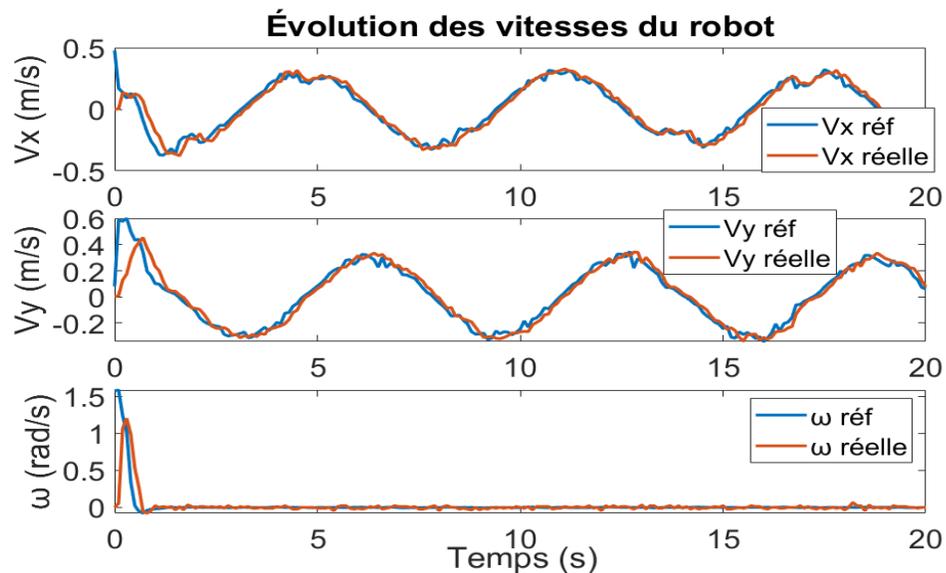
Évolution des vitesses des roues au cours du temps en utilisant le PI et le RN pour une trajectoire elliptique avec SIMULINK



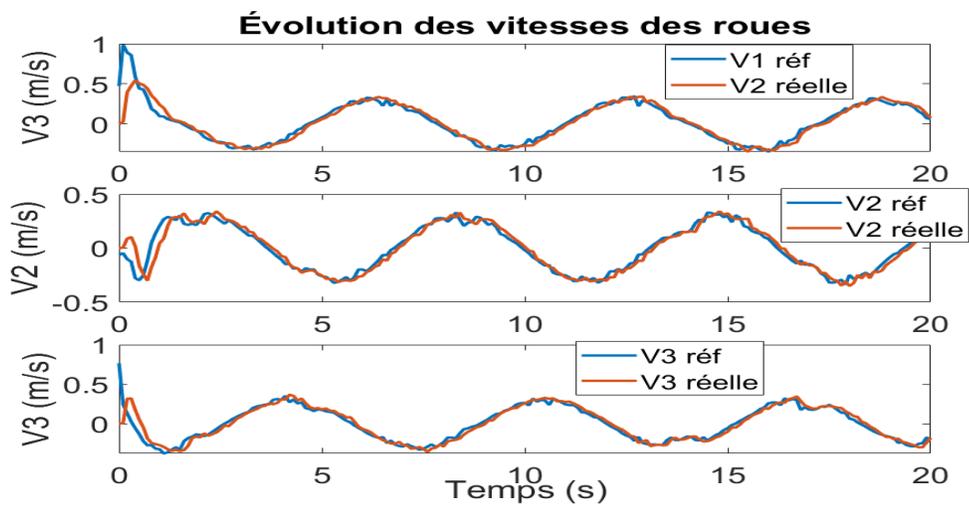
Évolution des vitesses du robot au cours du temps en utilisant le PI pour une trajectoire linéaire expérimentale



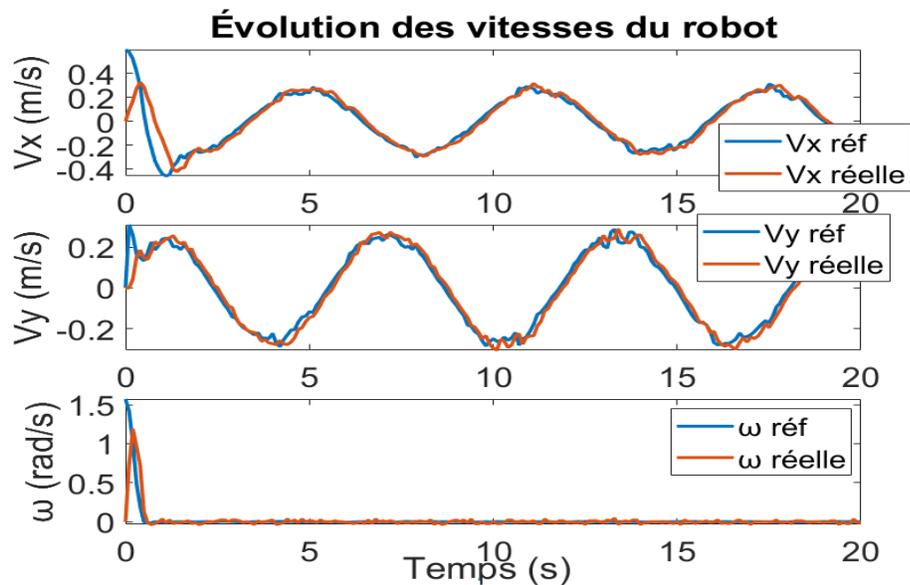
Évolution des vitesses des roues au cours du temps en utilisant le PI pour une trajectoire linéaire expérimentale



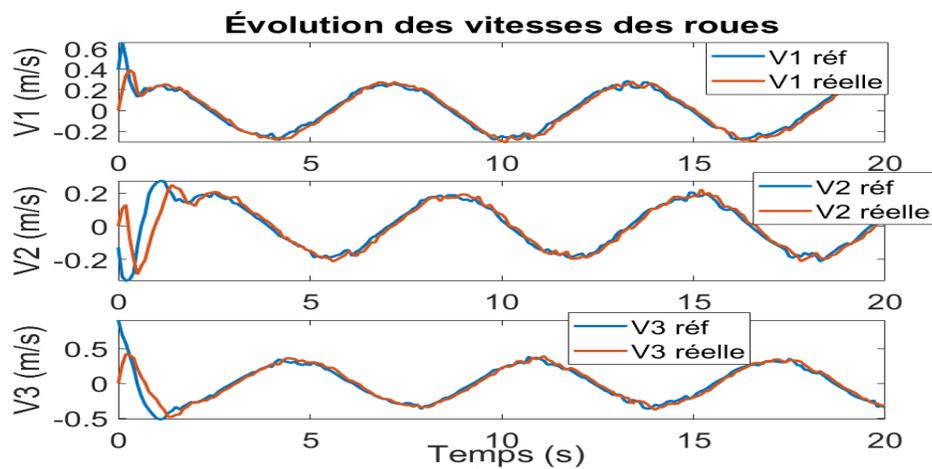
Évolution des vitesses du robot au cours du temps en utilisant le PI pour une trajectoire circulaire expérimentale



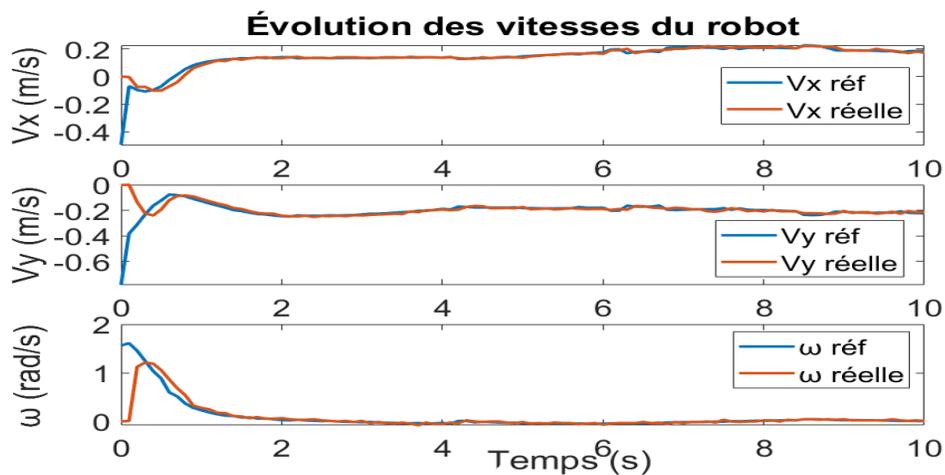
Évolution des vitesses des roues au cours du temps en utilisant le PI pour une trajectoire circulaire expérimentale



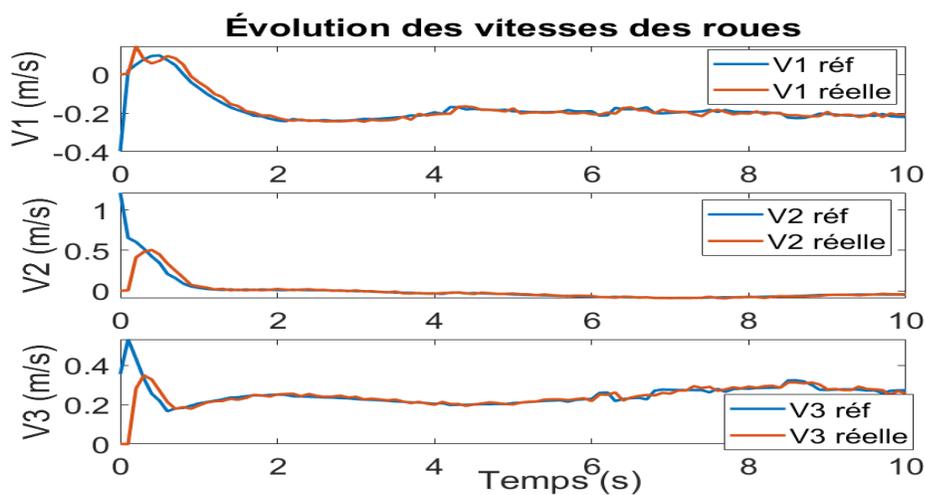
Évolution des vitesses du robot au cours du temps en utilisant le PI pour une trajectoire elliptique expérimentale



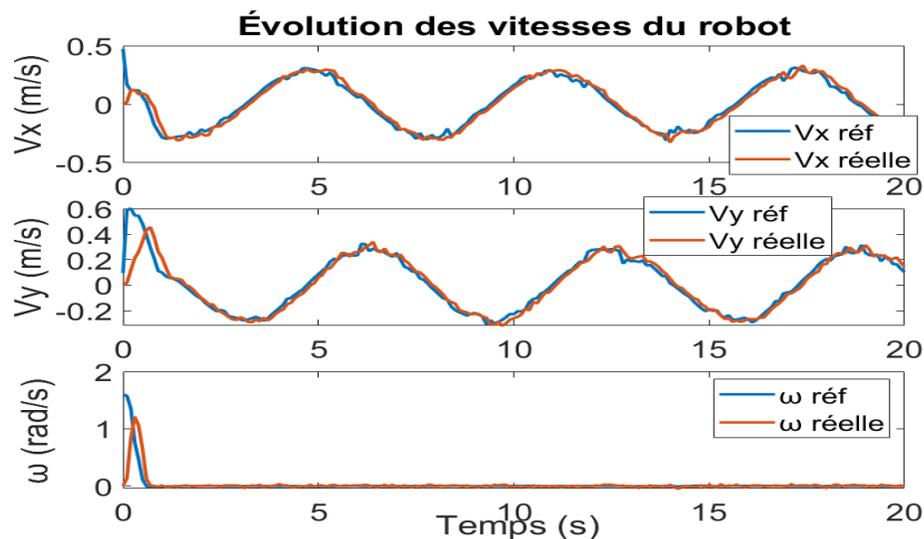
Évolution des vitesses des roues au cours du temps en utilisant le PI pour une trajectoire elliptique expérimentale



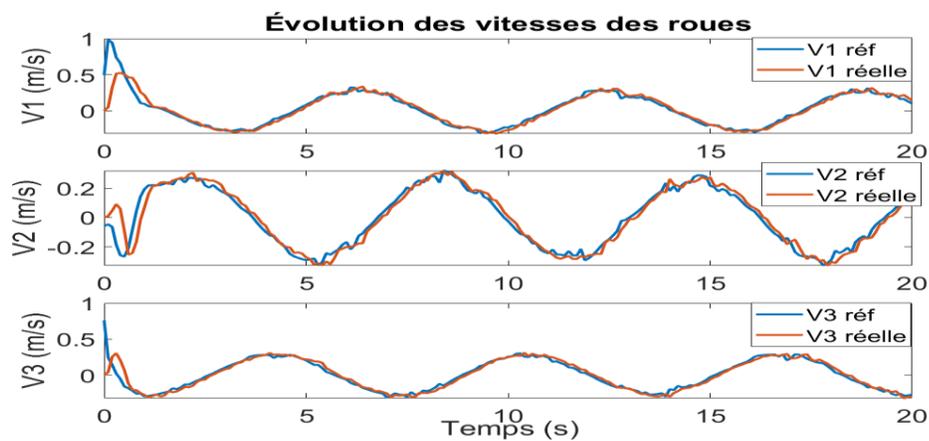
Évolution des vitesses des robots au cours du temps en utilisant le RN pour une trajectoire linéaire expérimentale



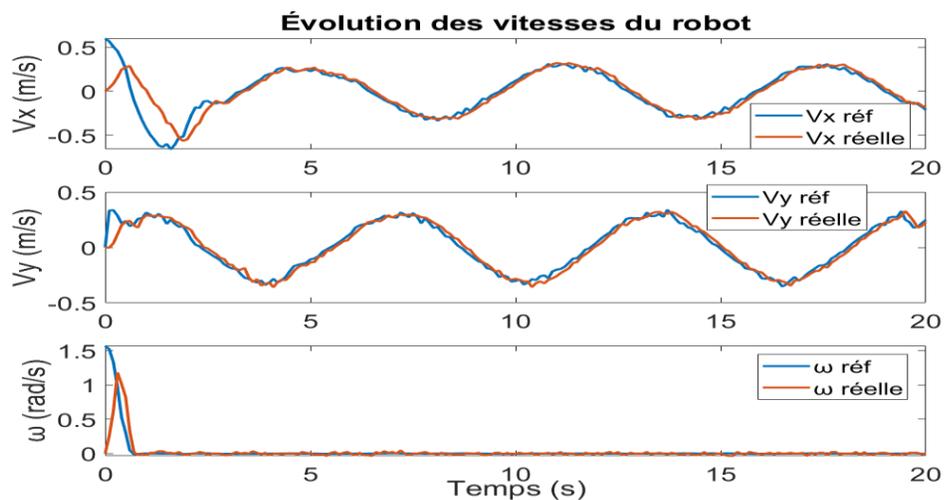
Évolution des vitesses des roues au cours du temps en utilisant le RN pour une trajectoire linéaire expérimentale



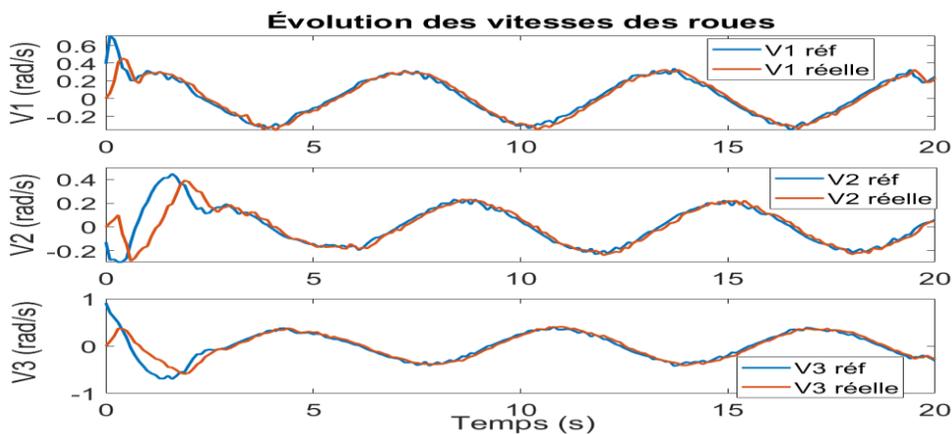
Évolution des vitesses du robot au cours du temps en utilisant le RN pour une trajectoire circulaire expérimentale



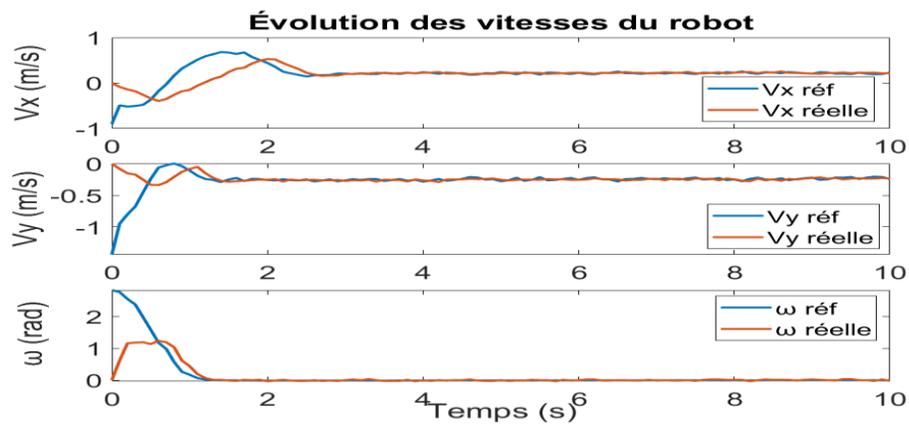
Évolution des vitesses des roues au cours du temps en utilisant le RN pour une trajectoire circulaire expérimentale



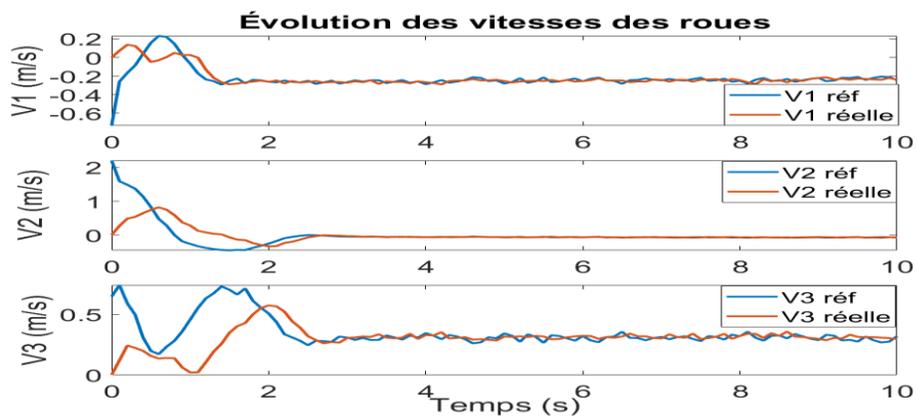
Évolution des vitesses du robot au cours du temps en utilisant le RN pour une trajectoire elliptique expérimentale



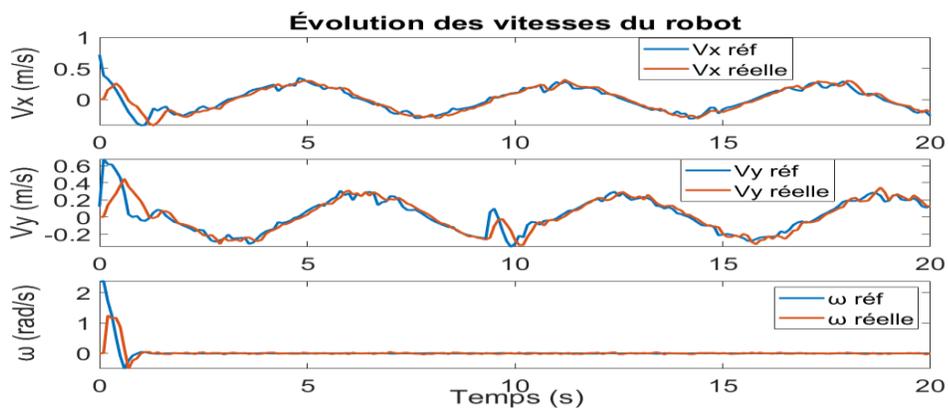
Évolution des vitesses des roues au cours du temps en utilisant le RN pour une trajectoire elliptique expérimentale



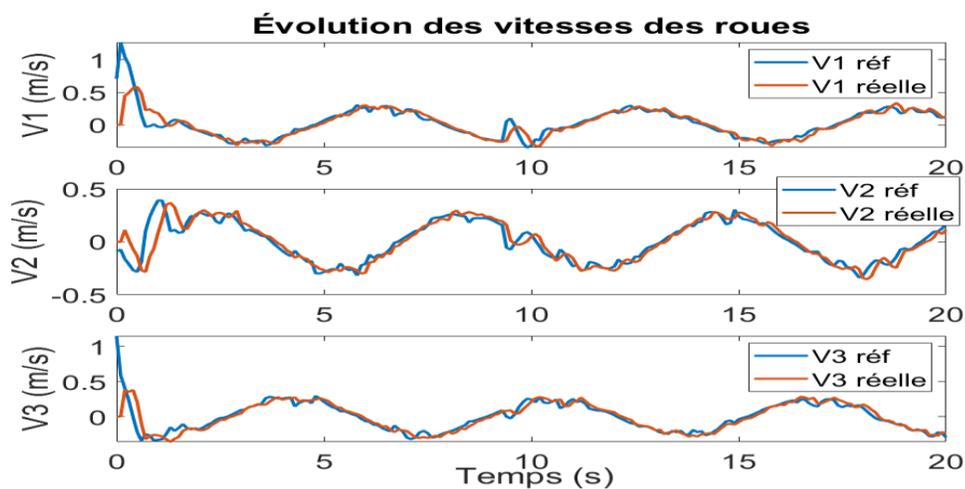
Évolution des vitesses du robot au cours du temps en utilisant le PI et RN pour une trajectoire linéaire expérimentale



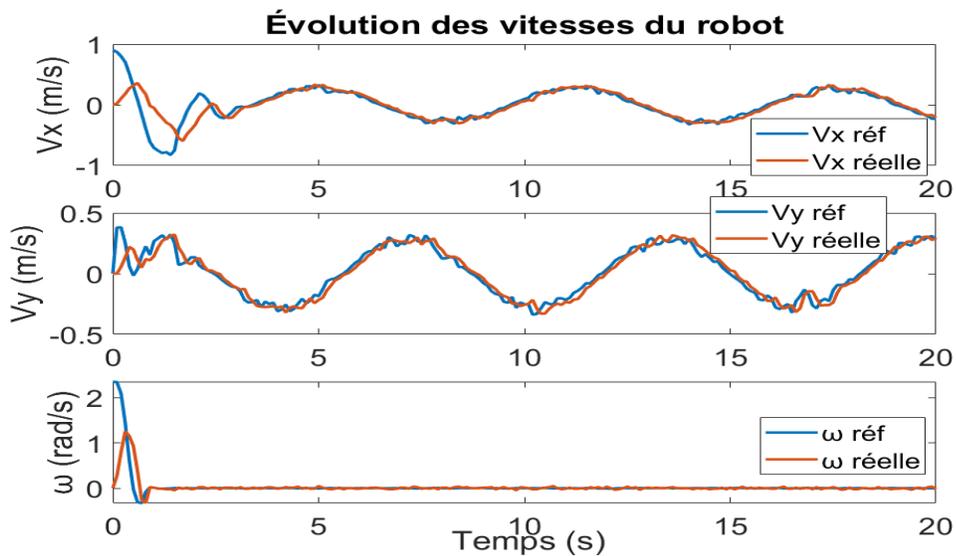
Évolution des vitesses des roues au cours du temps en utilisant le PI et RN pour une trajectoire linéaire expérimentale



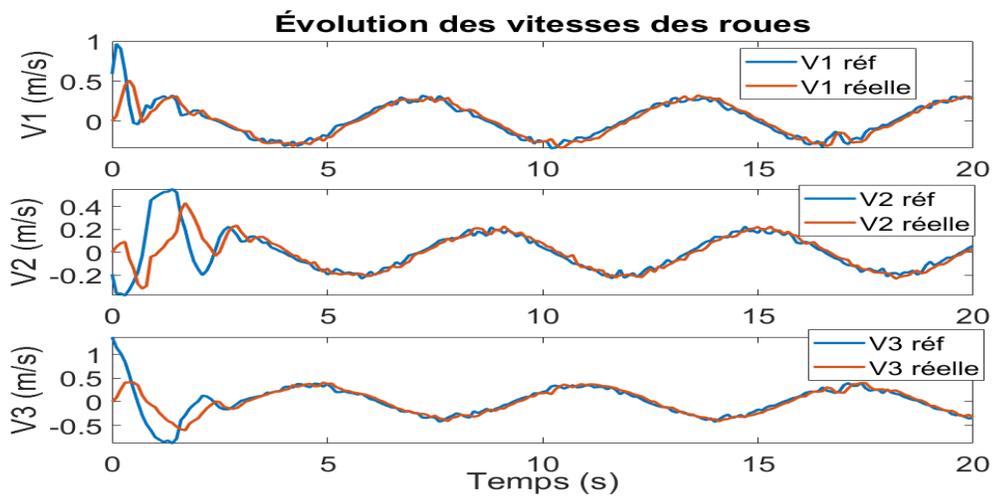
Évolution des vitesses du robot au cours du temps en utilisant le PI et RN pour une trajectoire circulaire expérimentale



Évolution des vitesses des roues au cours du temps en utilisant le PI et RN pour une trajectoire circulaire expérimentale



Évolution des vitesses du robot au cours du temps en utilisant le PI et RN pour une trajectoire elliptique expérimentale



Évolution des vitesses des roues au cours du temps en utilisant le PI et RN pour une trajectoire elliptique expérimentale