REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
الجمهورية الجزائرية الديمقراطية الشعبية

MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH

HIGHER SCHOOL IN APPLIED SCIENCES
--T L E M C E N--

المدرسة العليا في العلوم التطبيقية
École Supérieure en Sciences Appliquées

وزارة التعليم العالي والبحث العلمي

المدرسة العليا في العلوم التطبيقية
-تلمسان-

Mémoire de fin d'étude

Pour l'obtention du diplôme d'Ingénieur

Filière : Génie électrique

Spécialité : Automatique

Présenté par : BOUCHOUK Meroua

Thème

Commande et planification de trajectoire pour un escadron de drone

Soutenu publiquement, le 29 / 09 /2020 , devant le jury composé de :

M. MERAD Lotfi	Professeur	ESSA. Tlemcen	Président
M.TAHOUR Ahmed	Professeur	ESSA. Tlemcen	Directeur de mémoire
M.TADJINE Mohamed	Professeur	ENP. Alger	Co- Directeur de mémoire
M.MOKHTARI Mohamed Rida	MCB	ESSA. Tlemcen	Examinateur 1
M. ABDI Sidi Mohamed	MCB	ESSA. Tlemcen	Examinateur 2

Année universitaire : 2019 /2020

DÉDICACE

Je dédie ce travail

A mes chers parents qui m'ont toujours apporté leur amour et leur soutien pour affronter les difficultés de la vie.

A mes chers frères, et à ma chère sœur.

A tous les membres de ma famille : oncles, tantes, cousins et cousines.

A tous les automaticiens et les automaticiennes de promo 2015/2020.

A tous mes amis

A tous qui m'ont connu et aidé de près ou de loin pour réaliser ce modeste travail.

A tous ceux qui liront ce travail au future, N'hésitez pas à m'écrire si vous avez besoin d'aide.

REMERCIEMENTS

Celui qui ne remercie pas pour la petite chose, ne remercie pas pour la grande chose et celui qui ne remercie pas les gens, ne remercie pas Allah. Prophète Mohammed que la prière d'Allah et son salut soient sur lui.

Je tiens à remercier très vivement mes encadreurs Monsieur TADJINE Mohamed Professeur à l'École Nationale Polytechnique, Alger d'avoir proposé le sujet sur lequel j'ai travaillé, et qui a assuré la direction et l'encadrement du travail présenté dans ce mémoire. C'est avec beaucoup de chance que j'ai eu l'honneur et le plaisir de travailler sous sa direction. Monsieur TAHOUR Ahmed Professeur à Ecole Supérieure en Science Appliquée, Tlemcen qui m'a fait profiter de ses larges connaissances et ses précieux conseils. J'ai beaucoup apprécié sa méthode de travail, son enthousiasme et sa qualité de correction de documents scientifiques.

Je tiens mes plus vifs et profonds remerciements sont adressés à Monsieur ALLAM Ahmed de École Nationale Polytechnique, Bordj El-Bahri, Alger pour ses efforts, ses encouragements et son soutien permanent et qui a été toujours à l'écoute et prêt pour donner des conseils et pour sa pédagogie à diriger mon travail.

Mes remerciements s'adressent aussi à Monsieur Mehdi Zareb, Docteur à l'université Mustapha Stambouli Mascara et Monsieur BOUZID Yasser de École Nationale Polytechnique, Bordj El-Bahri, Alger. Pour l'intérêt particulier qu'ils ont porté à mon travail, leurs conseils et leur temps perdu pour répondre a mes questions.

Mes remerciements vont également a Monsieur KECHIDA Ahmed, Maitre de recherche au Centre de technologies industrielles CRTI, ex-CSC de m'avoir accueillie au centre. Ainsi, le temps, l'attention, l'intérêt qui il avait bien voulu me témoigner n'ont pas été perdus. Ils m'ont donné envie de persévérer dans ce domaine de robotique et d'aviation.

Je remercier trés chaleureusement Monsieur GOUBAA Abderrahim Ingénieur diplômée de l'institut des sciences appliqués et des technologies (INSAT), Tunisie pour tous ces efforts et son temps.

Mes remerciements s'adressent également à Monsieur le président du jury Monsieur MERAD Lotfi Professeur et chef de département du second cycle à l'ESSA de Tlemcen et à mes examinateurs Monsieur MOKHTARI Mohamed Rida Maitre de conférence B et Monsieur ABDI Sidi Mohamed Maitre de conférence B qui m'ont fait l'honneur d'évaluer ce travail.

ملخص

يندرج العمل في هذه الأطروحة في اطار التحكم بأنظمة التعاون متعددة الروبوتات، ويدرس مشاكل تتبع المسار والتحكم في التنسيق لأنظمة الطائرات بدون طيار الفردية و المتعددة. تتم معالجة مشاكل التحكم هذه من أجل نوع الدرون رباعي المحركات.
لذا كخطوة أولى في الأطروحة الحالية، تم تصميم نموذج رباعي الدوران بالإضافة إلى التحكم فيه باستعمال التحكم الانزلاقي و تحكم خطي، ثم يتم استخدام خوارزمية التوافق و التعاون و استراتيجية القائد الافتراضي من أجل تنفيذ المهام بطريقة منسقة، و خاصة للحفاظ على شكل المجموع أثناء التنقل . ينتهي العمل بمحاكاة ثلاثية الأبعاد تحت بيئة تحكم الروبوتات.
الكلمات الرئيسية: الطائرات بدون طيار، السرب، خوارزمية الإجماع، التحكم في وضع الانزلاق، متلاب، تحكم خطي.

ABSTRACT

The work presented in this thesis concerns the multi-robot cooperation systems, and studies trajectory tracking and coordination control problems for single and multi unmanned aerial vehicle (UAV) systems. These control problems are addressed for quadrotor type.
So as first step, the quadrotor is modeled as well as controlled. Secondly, in order to carry out tasks in a coordinated way, and especially to keep the geometrical configuration of the formation assured during the navigation. we have developed a cooperative strategies using the consensus algorithme. Through a this algorithme, a common understanding of the reference frame is achieved and enables the quadrotors to move accordingly. Hence, the vehicles follow the trajectory of the reference movement while keeping the geometric configuration in a simple feed forward manner during the navigation time. The work ends with a 3D simulation under the ROS environment.

Keywords: UAV, swarm, consensus algorithm,relative state feedback, sliding mode control, PID, ROS, MATLAB, Gazebo.

RÉSUMÉ

Les travaux présentés dans ce mémoire concernent les systèmes de coopération multi-robots, et étudient les problèmes de suivi de trajectoire et de contrôle pour les systèmes de véhicules aériens sans pilote (UAV). Ces problèmes de contrôle sont résolus pour le type quadrirotor.
Ainsi, comme une première étape de notre travail, le quadrirotor est modélisé ainsi que contrôlé. Deuxièmement, afin d'effectuer les tâches de manière coordonnée, et surtout de conserver la configuration géométrique de la formation pendant la navigation, nous avons développé une stratégie de coopération en utilisant l'algorithme du consensus. Grâce à cet algorithme, une stratégie commune de la compréhension à base d'un système de référence permet aux quadrotors de se déplacer en conséquence. Ainsi, les véhicules suivent la trajectoire du mouvement de référence tout en conservant la géométrie de la formation de manière simple pendant le temps de navigation. Le travail se termine par une simulation 3D dans l'environnement ROS.

Mots-clés : UAV, coopération, escadrone, algorithme de consensus,commande par retour d'etat,commande par retour des sortie contrôle mode glissant, PID, ROS, MATLAB, Gazebo.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

LISTE DES SIGLES ET ABRÉVIATIONS

UAV	Unmanned Aerial Vehicle
PID	Proportional Integral Derivative
LQR	Linear Quadratic Regulator
LQG	Linear Quadratic Gaussian
RPC	Linear Quadratic Estimator
PD	Proportional Derivative
3D	3 Dimensional cartesian space
ROS	Robot Operating System
TCP	Trés Courte Portée
MALE	Moyenne Altitude et Longue Endurance
HALE	Haute Altitude et Longue Endurance
SLAM	Simultaneous Localization And Mapping
ROS	Robot Operating System
GPS	Global Positioning System
IMU	International Mathematical Union
RGB	red, green, blue
MAV	Micro Air Vehicule

LISTE DES ANNEXES

Introduction

Ces dernières années, les problèmes de contrôle coopératif avec les systémes multirobots ont attiré beaucoup d'attention de la part de nombreux chercheurs [Ren, 2007b]. Ces technologies de contrôle collaboratif devraient être appliquées aux véhicules réels tels que les drones, les satellites artificiels et les robots d'observation mobiles autonomes, et les réseaux de capteurs distribués. Surtout en ce moment pour augmenter les mesures à prendre pour désinfecter les espaces publics. Faire respecter les mesures de distanciation sociale ce qui permettait de limiter l'exposition des travailleurs et des usagers à afin d'empêcher la propagation de virus covid19.

Les problèmes de contrôle de formation pour un système multi-véhicules ont été très bien étudiés ces dernières années, et de nombreux algorithmes de contrôle ont été développés pour ces problèmes [Murry, 2007]. Les algorithmes de consensus avec leur approche distribuée sont l'une des commandes proposées, a l'avantage d'avoir un réseau flexibel. La plupart des travaux des commandes en utilisant un algorithme basé sur le consensus, supposent que les véhicules sont exprimé sous la forme d'un système de $1^{ére}$ ordre. Malgré que Les résultats peut être directement étendu à un système de $2^{éme}$ ordre avec des changement de varaible. La plupart des chercheurs ont utilisé des moyens tels que l'inégalité matricielle linéaire (LMI) pour simplifier les calculs.

Le quadrirotor est un UAV de la famille multirotor utilisé dans les domaines de la surveillance et de la reconnaissance à distance des réseaux routiers et attaques aériennes. Ce type de drone est l'un des types préférés de création des coopérations pour les chercheurs, car ils ont des modèles dynamiques précis et des caractéristiques de stabilité favorables ainsi qu'un vol stationnaire à proximité de sites spécifiès par rapport à d'autres drones rotatifs et a aile fixe. Un autre raison de gagner en popularité, ils ont un coût moindre et une structure simple dans leur conception. Par conséquent, pour les études de recherche et l'utilisation commerciale, les chercheurs de contrôle sont intéressés a développer de nouveaux contrôleurs pour les drones quadrotor afin de fournir des performances bien formées pour diverses tâches compliquées à l'intérieur et à l'extérieur, telles que les tâches de patrouille, les activités agricoles, services de livraison, de surveillance et de sauvetage [Castillo, 2004], [Tayebi und McGilvra, 2006] . Tout au long de notre travail, une approche basée sur l'application du consonsus sur la

position. En cela, une formation consistes en drones qudrirotors avec des positions relatives souhaitées définies dans un cadre de coordonnées communes, cette approche appelée virtuel

référentiel. Le mouvement de la formation est réalisé par un mouvement du référentiel virtuel parceque les quadrotors doivent conserver leurs positions assignées. Un observateur incluses afin d'assurer la surveillance de la mission.
Donc notre mémoire s'articule autour de ces trois chapitres :

— chapitre I :
Résume toutes les notions de base nécessaires à la compréhension du domaine des drones, ainsi qu'un panorama de techniques classiques et récentes appliquées dans la planification de la trajectoire et les coopérations des robots .

— chapitre II :
Introduit tout d'abord, les notions et le concept des algorithmes de consensus. Par la suite, une modélisation du drone quadrirotor est introduit avec un modèle dynamique basé sur les équations de Newton-Euler. Nous élaborerons ensuite 2 types de commandes PID / MODE GLISSANT pour asservir la position et l'orientation du quadrirotor.

— chapitre III :
Comporte plusieurs simulations en boucle fermée qui montreront la stabilité et la robustesse des commandes sur un seul quadrirotor. Des interprétations des résultats pour chaque commande, par la suite une implémentation d'algorithmes de consensus avec plusieurs scénarios. Nous discutons également de l'influence des paramètres sur les résultats obtenus. Nous allons aussi présenter l'environnement de simulation ROS ou nous avons testé toutes les commandes en temps réel.

— Enfin, nous conclurons le manuscrit par une synthèse des résultats obtenus ainsi que par la proposition de quelques perspectives.

CHAPITRE 1 : REVUE DE LITTÉRATURE

1.1 Introduction

Au cours des trois dernières décennies, l'industrie aéronautique s'est concentrée sur la création des méthodes de vol qui n'impliquent pas de facteur humain à l'intérieur des avions, en développant des solutions pour le vol sans pilote. La nécessité de progresser dans le domaine des drones est motivée par le désire de maintenir les pilotes humains à l'abri des dangers.
Les systèmes des drones sont utiles aussi dans les missions militaires et les missions de recherche et de sauvetage à haut risque. Les drones peuvent être aussi utilisés dans un contexte civil pour des missions telles que la surveillance du trafic, la cartographie ou le pistage des animaux. On peut même mentionné le rôle des drones dans la lutte contre le virus de corona cette année.
Dans ce chapitre, nous présentons de façon générale les formations des groupes des robots, les types des formations puis nous présentons les architectures et les stratégies de commande.

1.2 Motivation et types des comportements :

Même si un seul drone est déjà capable d'accomplir différentes missions par lui-même, un vol en formation est une obligation dans le domaine de commande des UAV généralement faire appel à un groupe de drones , au lieu d'en utiliser qu'un seul, est motivé par deux facteur majeurs :
Soit la tâche à exécuter nécessite la coopération d'un nombre minimal des robots , soit l'amélioration d'un certain nombre de performances liées à l'exécution des tâches à réaliser par exemple :

- ***La rapidité*** : c'est-à-dire obtenir un niveau des performances élevés en tirant parti de la parallélisation des tâches, par exemple la désinfection des zones à risques, comme les hôpitaux ou les transports en commun peut être réalisée par un seul robot (drone) mais l'ajout d'autres robots va accélérer l'exécution de la tâche.
- ***Robustesse-fiabilité*** : les performances de contrôle peuvent être très peu affectées en cas de défaillance d'un robot (drone) .
- ***Flexibilité*** : possibilité d'exécuter les tâches désirées, de diverses manières.
- ***Emergence*** : produire une performance qualitativement supérieure à celle de l'addi-

tion des unités.

1.3 Un peu d'histoire sur les systèmes sans pilot

Dans la construction de systèmes sans pilote,le problème a toujours été un problème de capacité technologique. pour construire un système sans pilote efficace, il faut mettre en place des moyens de contrôle des véhicules dans un environnement incertain tout au long de la mission. Le contrôle des véhicules comprend la navigation ainsi que l'emploi d'armes.

L'une des premières idées développées par les Autrichiens et employés lors de leur assaut sur Venise en 1849, devait utiliser un sans pilote ballon (Figure.1.1) pour flotter sur le territoire ennemi et larguer des bombes, qui ont explosé à l'impact [Zhang u. a., 2005].

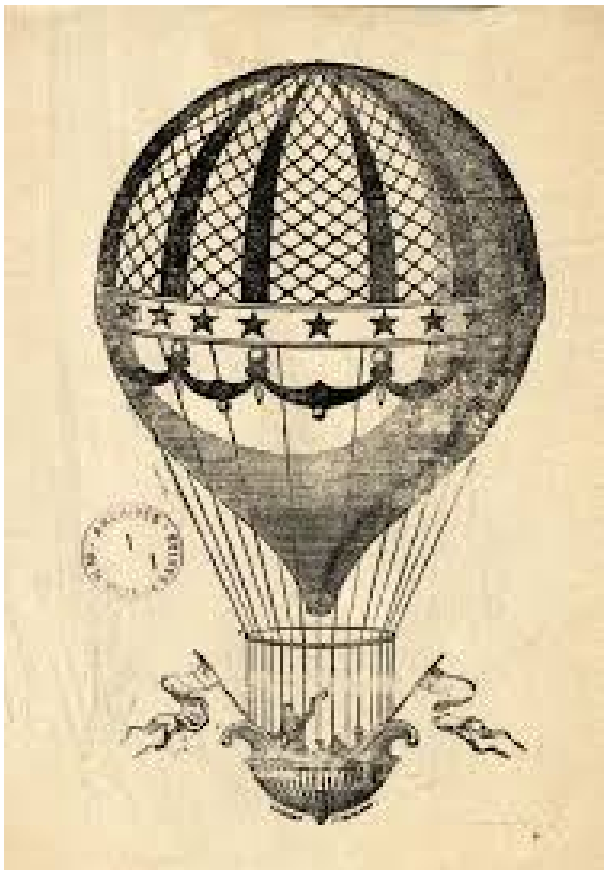


Figure 1.1 le ballon sans pilote

le problème opérationnel évident était la difficulté à contrôler la trajectoire des ballons. La portée opérationnelle était également très limitée, car pour l'emploi d'armes, les bombe sont été libérés par l'utilisation d'une charge électrique sur un fil de cuivre.

Au début de la Première Guerre mondiale, la torpille aérienne Kettering (Bug) (Figure.1.2) a été développé. Il a été conçu pour être lancé à partir d'un rail, survoler le territoire ennemi et plongez dans les lignes ennemies. Il transportait 180 livres d'explosifs puissants qui ont explosé

Figure 1.2 Kettering Bug un premier vol le 2 octobre 1918

lorsque le véhicule a heurté le sol. Le contrôle de la trajectoire du bug Kettering était en boucle ouverte, basé sur la direction de lancement et dépendante des vents en route. L'utilisation de mitrailleuses reposait sur l'estimation de la distance de déplacement en calculant la rotation du ventilateur, puis en arrêtant le moteur et en relâchant les ailes au point calculé.
En 1915, Nikola Tesla a eu une vision d'une flotte d'avions militaires sans pilote et en 1919, le premier UAV a été développé par Elmer Sperry, qui a été utilisé pour couler un navire de guerre allemand capturé. L'États-unis est le premier pays qui a vu le potentiel élevé des véhicules sans pilote pendant la seconde guerre mondiale, des UAV ont été utilisés pour attaquer des cibles navales et terrestres. Ils sont également été utilisés pour la reconnaissance aérienne et la pratique des cibles. Les avions habités ont été utilisés pour contrôler les armes sans pilote par une liaison radio. L'Allemagne a utilisé des armes telles que Henschel Hs 293 (Figure.1.3) Bombe guidée sans fil air-navire pour attaquer un certain nombre de navires. Pendant la même période, les États-Unis ont développé de nombreux systèmes sans pilote différents .

Le développement du pilote automatique est fortement corrélé au développement de l'UAV. La société d'Elmer Sperry a été la première à produire un pilote automatique capable de voler

Figure 1.3 Henschel Hs 293 la bombe planante téléguidée anti-navires développée en Allemagne 1940

de manière autonome pendant trois heures en ligne droite sans être supervisé par un humain. En 1933, le pilote automatique de Sperry (Figure.1.4) était capable de voler sur un cap vrai et de maintenir l'altitude, par rapport au cap gyroscopique de la première version. L'évolution actuelle des pilotes automatiques est en étroite relation avec le développement de systèmes de communication fiables. Bien que les premiers drones aient été contrôlés à distance par un opérateur humain, ils sont maintenant capables de recevoir un plan de vol et, sur cette base, de calculer la trajectoire de vol et de la suivre pour accomplir la mission. Dans les pilotes automatiques modernes, le facteur humain joue un rôle secondaire en supervisant le système et en contrôlant les équipements de bord, comme les caméras et les capteurs.

1.4 Gereralités sur les formations

Dans la mise en oeuvre d'un systéme multi-robots nous pouvons distiguer deux type de comportement

- le comportements compétitifs : chaque robots cherche a améliorer ses perfermance au detriment des autres .
- les comportement collaborative : les robots prenant en considiration les objectifs des autres afin d'améliorer un resultat global

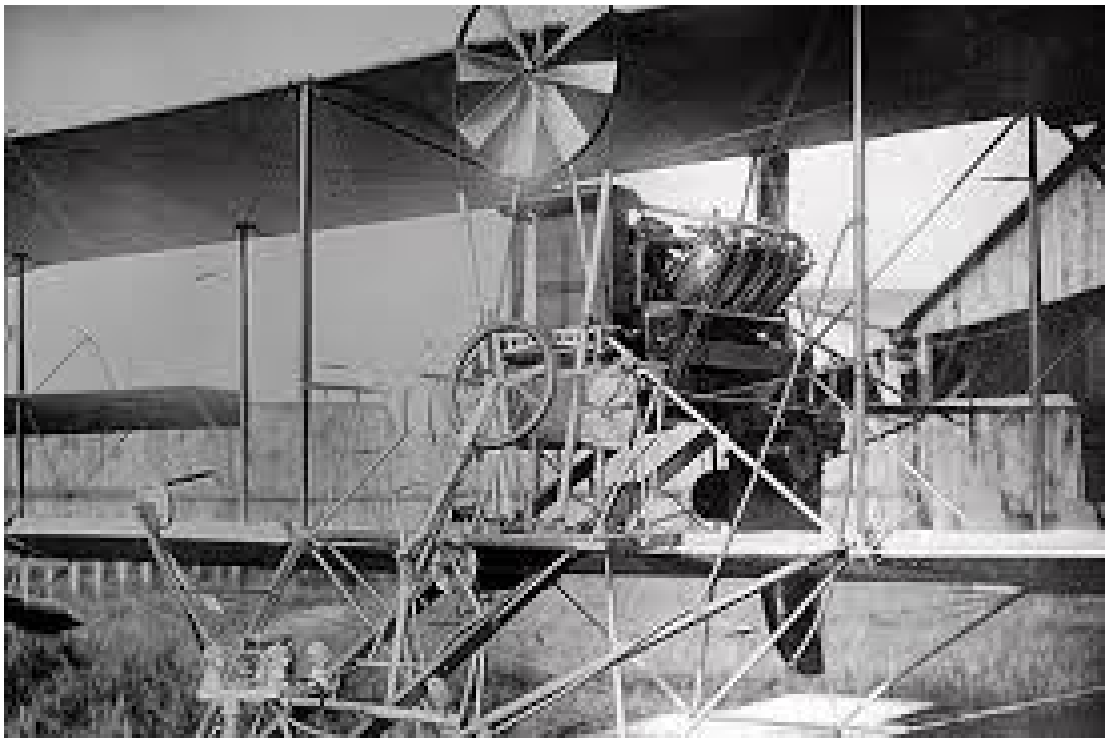


Figure 1.4 pilote automatique Sperry 1933

1.4.1 Système multi-robots homogènes

Un système multi-robots homogènes est un ensemble de robots qui ont les mêmes caractéristiques, ils doivent coopérer entre eux ; ils s'organisent automatiquement pour soulever, porter, pousser et déposer l'objet qu'un seul robot ne pourrait réaliser tout seul.dans l'ensemble on peux faire la distionction en fonction du milieu dans lequel evoluent les robots.

1. ***Robots terrestres*** **:** L'utilisation de la formation pour les robots terrestres a été assez peu étudiée et essentiellement pour des systèmes simples. Les robots mobiles holonomes ou non holonomes ont été largement utilisés pour démontrer l'efficacité de différentes méthodes. La majorité des études porte sur la coordination et la coopération entre plusieurs robots autonomes.
2. ***Robots volants*** **:** Les robots aériens ou drones « Unmanned Aerial Vehicle, UAV » offrent une très grande variété de types. Ils sont généralement classés selon leur taille et leur endurance. On eut ainsi distinguer
   — ***les drones HALE (Haute Altitude Longue Endurance)***
   — ***Les drones MALE (Moyenne Altitude Longue Endurance)***,
   les drones de courte et moyenne portée et les mini drones. Ils peuvent également étre caractérisés par leur fonction : drones stratégiques, drones tactiques ou drones

de combat « Unmanned Combat Air Vehicle, UCAV ». Les types d'engins vont des avions à ailes fixes aux voilures tournantes, en passant par des systèmes hybrides. Les drones stratégiques sont des drones HALE et ne sont pas vraiment concernés par les problématiques de vols en formation, car ils sont principalement destinés à des missions de reconnaissance ou de guerre électronique où un seul avion est généralement suffisant. Ils pourront néanmoins être intéressants à l'avenir dans le cas de ravitaillements en vol, où ils pourraient servir d'avions tanker.

1.4.2 Systèmes multi-robots hétérogènes

Un SMR hétérogènes est un ensemble de robots mixtes, c'est-à-dire les robots ne sont pas identiques, il peut y avoir des robots fixes et des robots mobiles.
Par exemple dans le cas de transport d'un objet chaque robot s'occupe de sa propre sous tache, les robots manipulateurs fixes s'occupent du dépôt et le soulèvement d'objet et les robots mobiles s'occupent de son transport.

1.5 Navigation des robots

La navigation des robots consiste à trouver un mouvement qui amène le robot d'une configuration initiale à une configuration finale mais il faut comprendre aussi que le robot est doté de capacités de perception, de décision et d'action (capteur) qui lui permettent d'agir de manière autonome dans son environnement en fonction de la perception .[M.HAZERCHI, 2012]

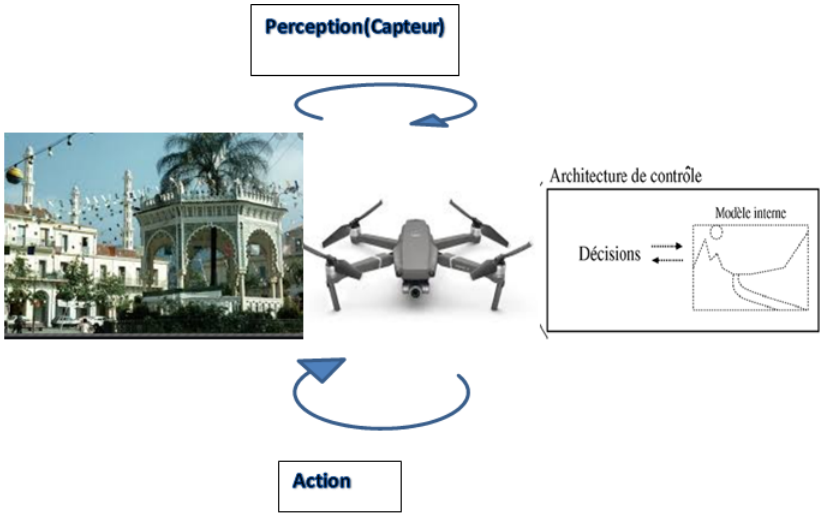


Figure 1.5 Schéma des interactions d'un drone avec son environnement

Généralement, ceci exige deux domaines : la planification de trajectoires et la réaction pour éviter des obstacles. De ce fait, pour différencier les techniques de navigation , on peut distinguer en principe trois approches :

— Approches délibératives (globales) :
Ces approches travaillent sur une représentation complète des obstacles et de l'espace libre . Le temps de calcul de l'espace des configurations croit exponentiellement avec le nombre de degrés de liberté . Ainsi les méthodes globales ne sont applicables qu'à des robots ayant un faible nombre de degrés de liberté . Le fonctionnement est basé sur un cycle rigide de modélisation de l'environnement, planification des actions au sein de cette représentation tout en évitant les obstacles, puis exécution du plan.

— Approche réactive (locales) :
Elle ne suppose aucun modèle a priori de l'environnement. Cette architecture s'appuie sur le couplage étroit entre les capteurs et les actionneurs, pour générer en continu les commandes. Elle consiste à offrir un ensemble de primitives plus réactives. Elles correspondent à des sous-tâches (convergence vers une cible, suivre un mur, éviter un obstacle) dont l'enchaînement est du ressort d'un planificateur de tâches ayant décomposé la tâche globale.

— Approche hybride :
C'est une combinaison des deux méthodes précédentes.

1.5.1 méthodes de planification de la trajectoire des robots

La planification de trajectoire pour les robots est un des aspects les plus importants dans la recherche de la navigation des robots.
Les méthodes de planification de trajectoire de robot peuvent être classifiées dans différentes sortes basées sur différentes situations. Selon l'environnement où le robot est localisé, les méthodes de planification de trajectoire peuvent être classifiées selon les deux types suivants :

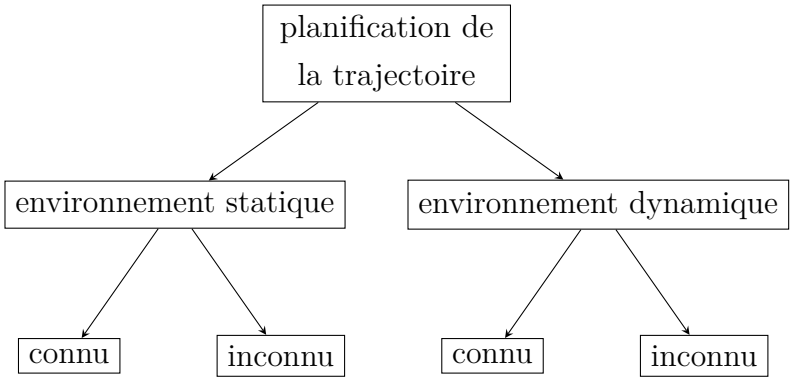
— Planification de la trajectoire du robot dans un environnement statique qui contient seulement les obstacles statiques
— planification de la trajectoire du robot dans un environnement dynamique qui contient des obstacles statiques et dynamiques.

Chacun des deux types pourrait être encore divisé en deux sous-groupes :

— planification de la trajectoire du robot dans un environnement complètement connu

dans lequel le robot connaît déjà l'endroit des obstacles avant qu'il commence à se déplacer. La trajectoire pour le robot pourrait être le résultat optimisé global parce que l'environnement entier est connu.

— planification de la trajectoire du robot dans un environnement partiellement connu ou incertain, dans lequel le robot perçoit l'environnement à l'aide de ses capteurs pour acquérir les informations locales de l'endroit, de la forme et de la taille des obstacles, et utilise alors ces informations pour procéder à une planification locale de la trajectoire



Organigrame de classification des méthodes de planification

1.6 Conclusion

Nous avons présenté une grande diversité des familles des drones à voilure fixe et tournantes, qui ont chacune leurs spécifications techniques.
le chapitre suivant fait l'objet de la modélisation d'un drone de type quadrotor.

CHAPITRE 2 MODÉLISATION ET COMMANDE COOPERATIVE D'UNE FORMATION DE DRONE

2.1 Introduction

Dans ce chapitre nous allons nous intéresser a la commande d'une formation de drone, nous allons donnée une introduction mathématique à deux aspects majeurs sous-jacents du contrôle coopératif, à savoir la modélisation des réseaux et les algorithmes de consensus. La terminologie pour la description des graphes est définie. De plus, une preuve de convergence pour un algorithme de consensus simple est présenté.

2.2 Modélisation du réseau

2.2.1 Préliminaires : théorie des graphes algébriques

[1] Un graphe non orienté peut modéliser mathématiquement un réseau de communication entre n véhicules indépendants où i = 1, 2, ... n est un ensemble fini et non vide des noeuds. Un graph G est une paire d'ensembles (V, E) où V est un ensemble fini non vide d'éléments appelés sommets , et E est un ensemble de paires non ordonnées de sommets distincts appelés arêtes. Les ensembles V et E sont l'ensemble des sommets et l'ensemble des arêtes de G, et sont souvent désignés par $V(G)$ et $E(G)$, respectivement.

Un exemple de graphique est présenté à (la Figure.2.1).

Le nombre de sommets d'un graphique est l'ordre du graphique ; il est généralement désigné par n et le nombre d'arêtes par m , la notation standard pour l'ensemble des sommets est

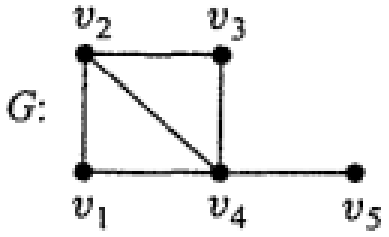
$V = \{V1, V2, \ldots, Vn\}$ et pour le jeu de bord est $E = \{e1, e2, \ldots, em\}$.

Les sommets arbitraires sont fréquemment représenté par u ,v ,w ... et les arêtes par e , f ,...

2.2.2 La matrice adjacente

Une représentation équivalente des graphs est la matrice d'adjacente $A = [a_{ji}] \in R^{n \otimes n}$. Les éléments de A sont définis comme $a_{ij} = a_{ji} = 1$ si $(j, i) \in E$ et a $_{ij} = a_{ji} = 0$ sinon

1. TOPIC IN ALGEBRIQUE GRAPH THEORY [Beineke, 2004]



$$V = \{v_1, v_2, v_3, v_4, v_5\}$$

$$E = \{v_1v_2, v_1v_4, v_2v_3, v_2v_4, v_3v_4, v_4v_5\}$$

Figure 2.1 Exemple d'un graphe connecté avec cinq nœuds et un arbre qui s'étend [Beineke, 2004]

les entrées diagonales sont égales à zéro et la matrice est symétrique . Dans l'exemple précedant on peut noté A la matrice ajacente

$$A = A^T = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix} \tag{2.1}$$

2.3 Les Algorithmes de Consensus

les stratégies de contrôle de formation basées sur le consensus peut garantir un maintien précis de la formation. Les informations circulent entre les véhicules tant que certaines conditions sont satisfaites.

la plupart des techniques de contrôle des formations (leader/suiveur, comportementales et virtuelles/formation de leader virtuel) peuvent être unifiées dans le cadre général de la recherche d'un consensus. car le partage d'informations est une condition préalable nécessaire au contrôle coopératif et ne puisse être atteinte sauf grâce à des algorithmes des consensus.

Consensus signifie converger asymptotiquement via des communications locales communication à une compréhension commune d'une quantité d'intérêt [Ren, 2007a] l'idée de base du consensus alors est qu'un groupe d'agents parviennent à un accord sur leurs états communs via une interaction

2.3.1 Algorithmes de Consensus pour les systèmes de second ordre

Afin de se concentrer sur l'étude de la loi de contrôle de la formation, un drone individuel est modélisé comme un système de masse ponctuelle. Le problème du consensus pour les systèmes multi-agents de second ordre dans le plan $2D$ est considéré. La dynamique de chaque drone peut être décrite comme un double intégrateur comme :

$$\begin{aligned} \dot{\xi}_i(t) &= \zeta_i(t) \\ \dot{\zeta_i}(t) &= u_i(t) \end{aligned} \tag{2.2}$$

ou ξ_i et $\zeta_i \in \mathrm{R}^m$ les états d'information de $i_{éme}$ drone par exmple ξ_i peut prendre la position ou altitude $\xi_i(t) = x = [x_i, y_i]^T$ et ζ_i soit la vitesse, taux de montée ou la vitesse angulaire. $\zeta_i(t) = v = [v_{xi}\cos\theta_i, v_{yi}\sin\theta_i]^T$ nous avons présenté le protocole de consensus linéaire suivant d'apris ([Ren, 2007a]) :

$$u_i(t) = \sum_{v_j \in N_i(t)} a_{ij}[K_1[\dot{\xi}_J(t) - \dot{\xi}_i(t) - r_{ij}] + k_2[\dot{\zeta_j}(t) - \dot{\zeta_i}(t)] - k_3(\dot{\zeta_i}(t) - \dot{\zeta_*}(t)) \tag{2.3}$$

Où a_{ij} désigne l'élément de la matrice d'adjacence pondérée, $N_i(t)$ désigne l'ensemble des voisins du noeud i la valeur des k_1, k_2, k_3 sont tous positifs.

2.3.2 Calcul mathmatique

La dynamique du $i_{éme}$ agent est définie par :

$$\begin{aligned} \dot{x_i} &= Ax_i + Bu_i \\ y_i &= C.x_i \end{aligned} \tag{2.4}$$

Où x_i sont les états de $i_{éme}$ agents et u_i sont lois de contrôle d'entrée, y_i sorties de notre système qui est supposé stabilisable et détectable.
On se base dessus ([Zhongkui Li, 2010]),([Jinhuan Wang und Hu, 2008])

Lemme 2.1. *Considérons les systèmes linéaires (2.4). Soit B et C sont des matrices non singulières et supposent que toutes les valeurs propres de A appartiennent à l'axe imaginaire. Supposons que le graph (arbre) de communication G(t) est uniformément connecté et la matrice Laplacienne correspondante L(t) par continus et délimités morceaux . Ensuite, la loi de contrôle est donner par :*

$$u_i(t) = c.K \sum_{j=1}^{N} a_{ij}(x_i - x_j) \tag{2.5}$$

Remarque :

$$\sum_{j=1}^{N} a_{ij}(x_i - x_j) = \sum_{j=1}^{N} L_{ij}x_j \tag{2.6}$$

exprime les musures d'un agant i

On remplace (2.5) dans (2.4) :

$$\dot{x_i} = Ax_i + CBK \sum_{j=1}^{N} L_{ij}x_j \tag{2.7}$$

pour $\widehat{x_i} = (T \otimes I_n)x$ telque $T.L.T^{-1} = D$ ou T est le vecteur des valeur propre de notre systeme

Finalment on trouve l'equation (2.8)

$$\dot{\widehat{x_i}} = (A + C\lambda_i BK)\widehat{x_i} \tag{2.8}$$

pour que le consensus converge il faut que (2.8) soit stabale [2]

2.3.2.1 Algorithme de calcul des constantes c et K

Le consonsus est atteint si (2.8) est stable $\Rightarrow$ toutes les matrices sont Hurwitz pour.$i = 2, ..., N$ donc selon les notions de stabilité de lyapunov il exite une matrice P>0 qui est la solution de systeme LMI suivant :

$$A.P + P.A^T - 2.B.B^T < 0$$

avec P>0

$$K = -B^T.P^{-1}$$

$$c_{min} = \frac{1}{Min_{2,...,N} Re\,(\lambda_i)}$$

Tel que λ_i sont les valeurs de L.

2. Notion sur la stabilité au sens de lyapunov (C)

2.3.3 Application des algorithmes de consensus sur une formation de quadrirotor

Nous pouvons résumer cette application en utilisant le schéma suivant ([Petit, 2016]) :

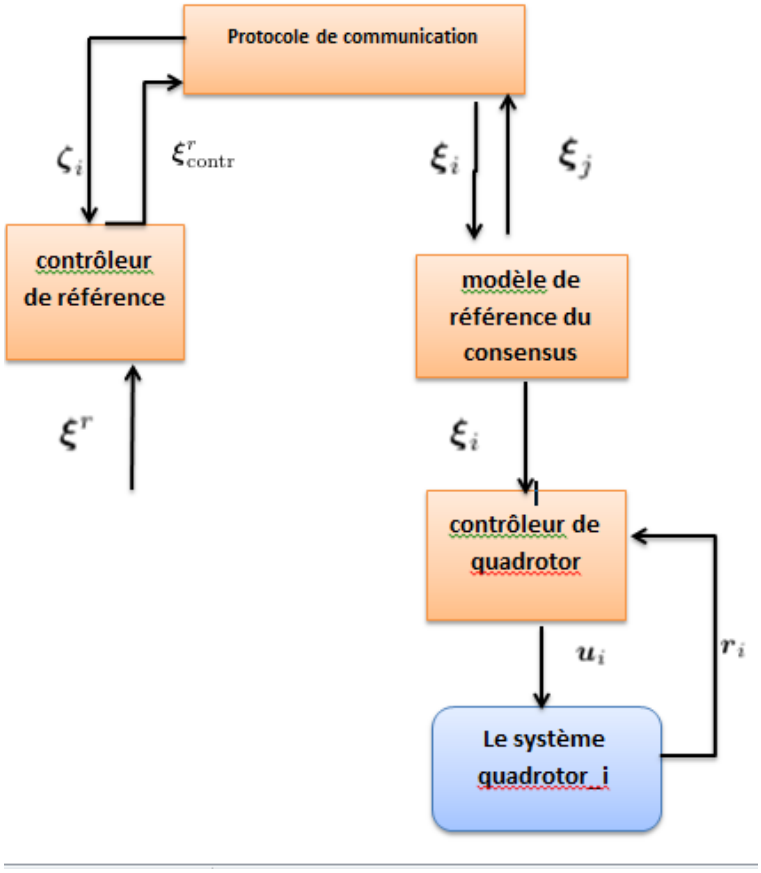


Figure 2.2 Application d'algorithmes de consensus sur une formation d'UAV

2.3.4 Modélisation et commande d'un Quadrotor

Quand on parle d'un quadrotor on parle d'un corps rigide avec quatre rotors aux quatre coins qui forment un carré. Ce corps rigide a six degrés de liberté dans l'espace $3D$. Cependant, sa configuration d'actionneur ne permet de contrôler que quatre degrés de liberté, à savoir la position le long de x, y et z et le lacet. Il s'agit donc d'un véhicule sous-actionné. (le nombre des entrées inférieur au nombre des sorties)

Figure 2.3 Exemple d'un quadrirotor

2.3.5 Modéle Cénimatique

Il existe deux cadres de référence de coordonnées différents associés à la modélisation d'un quadrotor :

— **Le repère terrestre(inertiel ou fixe)** : est noté $O_{FI}(X, Y, Z)$ C'est un repère d'origine O_{FI} et d'axes X , Y et Z lié à la terre supposée immobile

— **Le repère lié au corps de quadrirotor** : Le cadre $O_{ABC}(e_x, e_y, e_z)$ est le cadre qui est fixé au quadrotor de telle manière que son origine coïncide avec le centre de gravité du véhicule. Les quatre rotors se trouvent tous sur le plan $O_x - O_y$. L'axe z positif est orienté vers le haut, à l'opposé du sol, comme le montre (Figure.2.4).

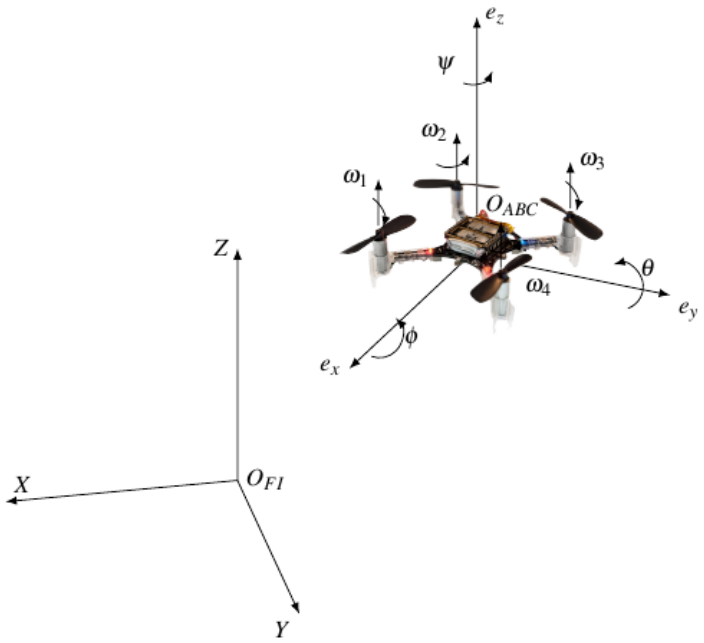


Figure 2.4 Crazyflie dans le système de référence body-frame (OABC) et fixed-frame (OFI). Forces.[De Dinechin und Melquiond, 2015]

La rotation du cadre inertiel par rapport aux axes (x,y,z) , est représentée par la matrice de rotation R utilisant les angles d'Euler.

$$\phi = Phi \quad \theta = Theta \quad \psi = Psi$$

$$R = \begin{bmatrix} \cos\theta\cos\psi & \sin\phi\theta\cos\psi - \cos\phi\sin\psi & \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi \\ \cos\theta\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi \\ -\sin\theta & \sin\phi\cos\theta & \cos\phi\cos\theta \end{bmatrix}. \quad (2.9)$$

Les quadrotors sont généralement conçus dans deux configurations à savoir "X" et "+". La plupart des chercheurs utilisent la configuration (X) pour s'assurer que le cetre de masse est situé au centre du drone.

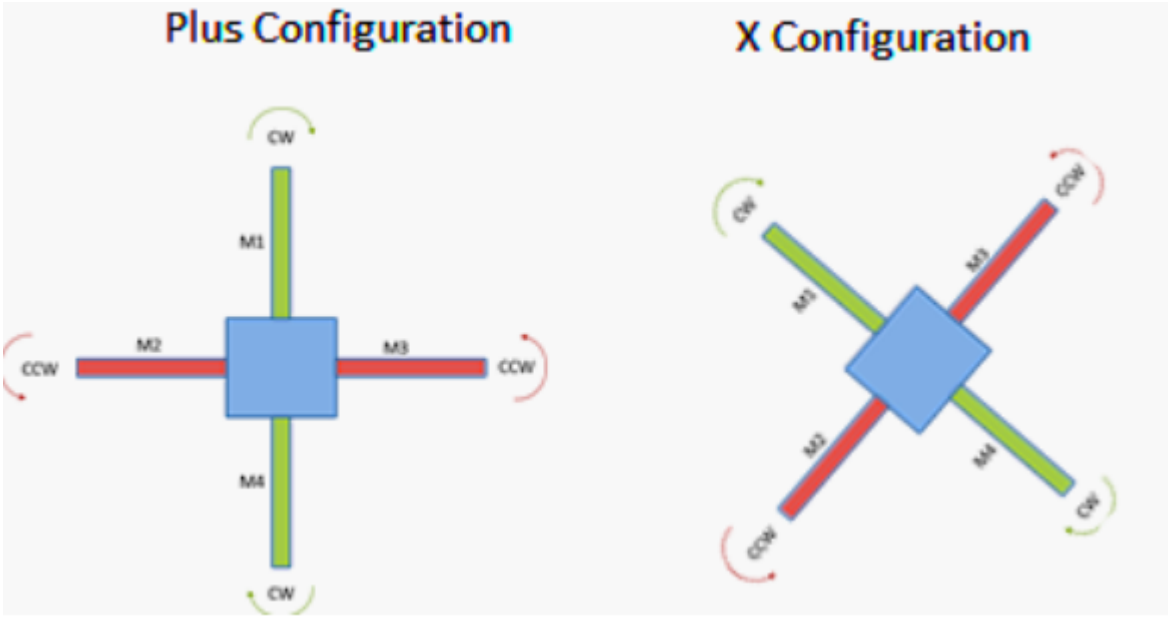


Figure 2.5 les configurations des quadrirotor [Adventures, 2012]

2.3.6 Modilisation des forces Dynamique

2.3.6.1 Hypothèses

Afin de modéliser les forces dynamique de quadrotor,on suppose les hypothéses suivantes :

— La structure du quadrotor est supposée rigide et symétrique.
— La matrice d'inertie J est supposée constante
— Les forces de portance et de trainée sont supposées proportionnelles au carré de la vitesse de rotation des rotors.
— Le centre de gravité . est supposé confondu avec Le repère liée au corps du quadrotor

2.3.6.2 force de poussée

La force de poussée en régime permanent générée par un rotor en vol stationnaire (c'est-à-dire un rotor qui ne se déplace pas horizontalement ou verticalement elle à tendance à faire élever le quadrirotor) peut être modélisé en pratique par (après la simplification des la surface du disque du rotor,..)

$$F_i = C_T \Omega_i^2 \tag{2.10}$$

ou $i^{éme}$ représente le rotor , Ω_i la vitesse angulaire (ou vitesse de rotation) et C_T représente le coefficient de portance déterminée à partir de tests statiques. [M.GUETTACHE, 2019]

2.3.6.3 force de trainéé réaction

C'est la réaction créée par la rotation des quatre hélices :

$$M_i = C_D \Omega_i^2 \tag{2.11}$$

. C_D représente la constante de trainée et peut etre déterminée par des essais statiques de poussée.

La force totale générée dans la direction z est donnée par la somme des poussées individuelles générées par chacun des quatre rotors. La force générée par les rotors (la Figure.2.6) est donnée par :

$$F = \sum_{i=1}^{4} F_i \tag{2.12}$$

$$F = C_T(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \tag{2.13}$$

2.3.6.4 Les couples aérodynamiques actif

Les couples aérodynamiques appliqués au quadrirotor s'écrivent alors (2.22)

$$\tau = \begin{bmatrix} \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} = \begin{bmatrix} l.C_T(F_2 - F_4) \\ l.C_T(F_3 - F_1) \\ C_D.[-(M_1 + M_3) + (M_2 + M_4)] \end{bmatrix} \tag{2.14}$$

Où :

— L $\in R^+$, est la distance entre le centre de gravité du quadrirotor et l'axe de rotation de l'un des rotors

D'après les équations ci-dessus (2.22) et (2.13) on obtient une matrice qui relie la poussée et les couples aérodynamiques.aux vitesses des rotors :

$$\begin{bmatrix} F \\ \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} = \begin{bmatrix} C_T & C_T & C_T & C_T \\ 0 & lC_T & 0 & -lC_T \\ -lC_T & 0 & lC_T & 0 \\ C_D & -C_D & C_D & -C_D \end{bmatrix} \begin{bmatrix} \Omega_1^2 \\ \Omega_2^2 \\ \Omega_3^2 \\ \Omega_4^2 \end{bmatrix} \tag{2.15}$$

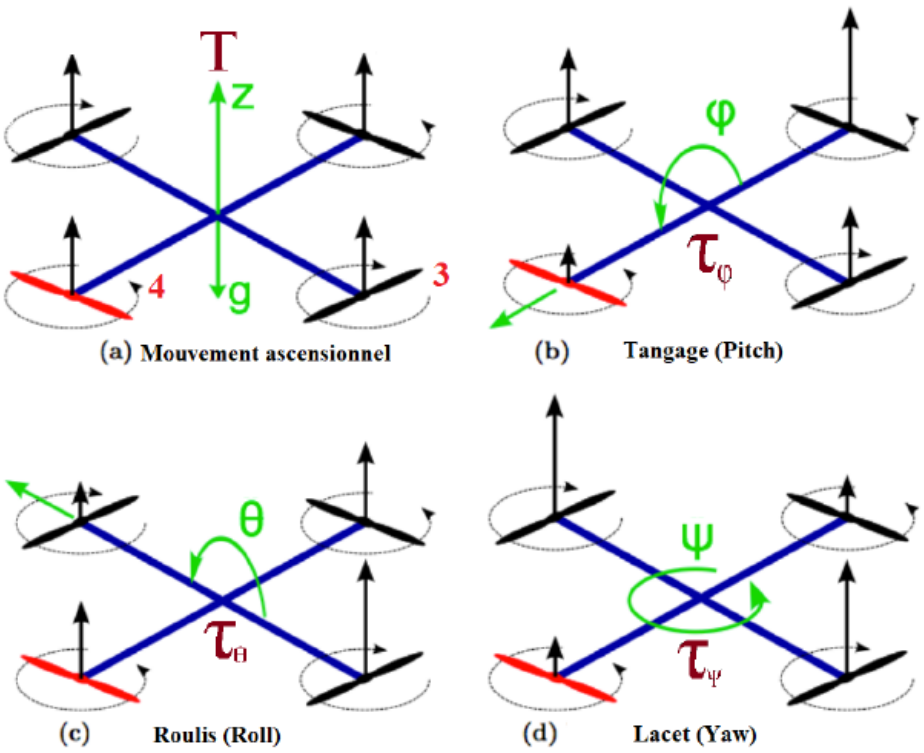


Figure 2.6 Mouvement d'un quadrirotor [M.GUETTACHE, 2019]

2.3.7 Relation entre les vitesses angulaires et les angles d'Euler

Lorsque un solide tourne à une vitesse constante, sa vitesse angulaire est constante, par contre les variations des angles d'Euler seront variables car elles dépendent des angles instantanés entre les axes des deux repères.La séquence des angles d'Euler est obtenue à partir de trois rotations successives : lacet, tangage et roulis.

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = R_\phi R_\theta \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} + R_\phi \begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + \begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} \tag{2.16}$$

Pour des raisons de simplification et comme la plupart des cas étudiés dans la litérature travaillent avec un modèle simplifié où p , q et r représentent les vitesses angulaires

autour des trois axes principaux du quadrotor.donc l'équation devient :

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \tag{2.17}$$

2.3.8 Modèle dynamique non linéaire en utilisant Newton Euler

La dynamique des quadrotors est modélisée à l'aide des équations de mouvement de Newton Euler pour un corps régide dans l'espace libre .
En utilisant les hypothèses de la (section.2.3.6.1) les équations des mouvements sont les suivantes :
Mouvement de translation

$$m\ddot{r} = \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} + \mathbf{R} \begin{bmatrix} 0 \\ 0 \\ F_1 + F_2 + F_3 + F_4 \end{bmatrix} \tag{2.18}$$

Mouvement de rotation

$$I \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \tau - \mathbf{R} \begin{bmatrix} p \\ q \\ r \end{bmatrix} I \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{2.19}$$

où p , q et r représentent les vitesses angulaires autour des trois axes principaux du quadrotor et τ est défini dans l'équation(2.22)
Le modèle dynamique du quadrotor est donc donné par le système d'équations suivant :

$$\begin{cases} \ddot{x} = (\dfrac{F}{m})(\cos\phi \sin\theta \cos\psi + \sin\phi \sin\psi) \\ \ddot{y} = (\dfrac{F}{m})(\cos\phi \sin\theta \cos\psi - \sin\phi \cos\psi) \\ \ddot{z} = -g + \cos\phi \cos\psi (\dfrac{F}{m}) \\ \ddot{\phi} = \left(\dfrac{I_y - I_z}{I_x}\right) \dot{\theta}\dot{\psi} - I_p \Omega \dot{\theta} \left(\dfrac{1}{I_x}\right) + \left(\dfrac{1}{I_x}\right) \tau_\phi \\ \ddot{\theta} = \left(\dfrac{I_z - I_x}{I_y}\right) \dot{\phi}\dot{\psi} - I_p \Omega \dot{\phi} \left(\dfrac{1}{I_y}\right) + \left(\dfrac{1}{I_y}\right) \tau_\theta \\ \ddot{\psi} = \left(\dfrac{I_x - I_y}{I_z}\right) \dot{\phi}\dot{\theta} + \left(\dfrac{1}{I_z}\right) \tau_\psi \end{cases} \tag{2.20}$$

I_x, I_y, I_z représente le moment d'inertie autour des trois axes principaux du quadrotor et I_p représente le moment d'inertie de l'hélice autour de son propre axe.

2.3.8.1 Modèle d'état

Pour construire un modèle d'état, nous choisissons comme vecteur état :

$$[x, y, z, \dot{x}, \dot{y}, \dot{z}, \phi, \theta, \psi, \dot{\phi}, \dot{\theta}, \dot{\psi}]^T = [x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}, x_{12}]^T \tag{2.21}$$

et les commandes :

$$\begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} = \begin{bmatrix} T \\ \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} = \begin{bmatrix} C_T.(\Omega_1^2 + \Omega_3^2 + \Omega_2^2 + \Omega_4^2) \\ l.C_T.(\Omega_2^2 - \Omega_4^2) \\ l.C_T.(\Omega_3^2 - \Omega_1^2) \\ C_D.[(\Omega_1^2 + \Omega_3^2) - (\Omega_2^2 + \Omega_4^2)] \end{bmatrix} \tag{2.22}$$

La représentation d'état obtenue on utilisant (2.21) et (2.22) :

$$\begin{cases} \dot{x_1} = x_4 \\ \dot{x_2} = x_5 \\ \dot{x_3} = x_6 \\ \dot{x_4} = \dfrac{U_1}{m}.(\cos(x_7)\sin(x_8)\cos(x_9) + \sin(x_7)\sin(x_9)). \\ \dot{x_5} = \dfrac{U_1}{m}.(\cos(x_7).\sin(x_8).\sin(x_9) - \sin(x_7).\cos(x_9)) \\ \dot{x_6} = \dfrac{U_1}{m}.(\cos(x_7)\cos(x_8)) - g \\ \dot{x_7} = x_4 \\ \dot{x_8} = x_4 \\ \dot{x_9} = x_4 \\ \dot{x_{10}} = \left(\dfrac{I_y - I_z}{I_x}\right) x_5 x_6 - I_p \Omega x_5 \left(\dfrac{1}{I_x}\right) + \left(\dfrac{1}{I_x}\right) U_1 \\ \dot{x_{11}} = \left(\dfrac{I_z - I_x}{I_y}\right) x_4 x_6 - I_p \Omega x_4 \left(\dfrac{1}{I_y}\right) + \left(\dfrac{1}{I_y}\right) U_2 \\ \dot{x_{12}} = \left(\dfrac{I_x - I_y}{I_z}\right) x_4 x_5 + \left(\dfrac{1}{I_z}\right) U_3 \end{cases} \tag{2.23}$$

2.3.8.2 Stratégies de commande

Dans le cadre de notre travail, il nessaire de concevoir et d'implémenter une loi de commande qui va assurer la stabilités de quadrotor et qu'il atteindre une position

acceptable. [O.MEKKID, 2016],[M.Mokhtari, 2015]

Control PID de la position et d'attitude :

Le régulateur PID (proportionnel-intégral-dérivé) c'est un solution lineaire et un régulateur classique basé sur commande en boucle fermée qui essaie d'obtenir le résultat réel proche du résultat désiré par l'élimination d'erreur de la sortie. Il est basé sur trois opérations :

— coefficient de gain proportionnel : le gain peut stabilisé relativement le systéme sans autres paramètres, mais Plus le coeffcient est élevé, plus le systéme semble plus sensible ,et plus il trop faible le systéme apparaît lent et sera plus difficile de garder stable.

— Coefficient de gain intégral - ce coeffcient peut augmenter la précision de la sortie du systéme .et de méme absorbe les perturbation et les bruit de sortie. Mais quand il plus grand le systéme sera plus lent et oscillatoire.

— coefficient de gain Dérivée - ce coefficient permet le systéme d'atteindre plus rapidement la référence souhaitée.

La dynamique d'altitude dans (2.20) est réduite à un système à double intégrateur linéaire par linéarisation :

$$U_1 = (\widehat{u}_1 + m.g).\frac{1}{\cos\theta.\sin\phi} \tag{2.24}$$

l'erreur de poursuite dans ce cas peux étre definie saus la :

$$\epsilon(t) = \xi_d(t) - \xi(t) \tag{2.25}$$

Donc on peux definir la commande PID comme :

$$U_2 = K_{P1}.\phi_i(t) + K_{I1}.\int_0^1 \phi_i(t)dt + K_{D1}.\frac{\partial\phi_i(t)}{\partial t} \tag{2.26}$$

$$U_3 = K_{P2}.\theta_i(t) + K_{I2}.\int_0^1 \theta_i(t)dt + K_{D2}.\frac{\partial\theta_i(t)}{\partial t} \tag{2.27}$$

$$U_4 = K_{P3}.\psi_i(t) + K_{I3}.\int_0^1 \psi_i(t)dt + K_{D3}.\frac{\partial\psi_i(t)}{\partial t} \tag{2.28}$$

On utilisant une discrétisation :

$$U(k) = K_P.\epsilon(k) + K_I.\Sigma_{j=0}^{k}\epsilon(j)\Delta(t) + K_D.\frac{\epsilon(k) - \epsilon(k-1)}{\Delta(t)} \tag{2.29}$$

Control de position et attitude par mode glissant :

Nous avons choisi aussi une solution non lineaire en utilisant la commande par mode glissant qui assure une stabilité au sens de Lyapunov [Khalil, 2011] grâce à sa simplicité d'implémentation et à sa grande robustesse vis-à-vis les incertitudes paramétriques.

L'erreur de suivi dans ce cas peut être définie sous la forme :

$$\epsilon(t) = \xi_d(t) - \xi(t) \tag{2.30}$$

$\xi_d = [x_d, y_d, z_d, \phi_d, \theta_d, \psi_d]^T$ est le vecteur d'orientation désirés.
Puisque chaque état est d'ordre relatif 2 par rapport à sa commande, la surface de glissant sera :

$$S_i(t) = \dot{\epsilon}_i.(t) + \lambda\epsilon_i(t) \tag{2.31}$$

Remarque
Nous choisissons une surface linéaire pour assurer une convergence exponentielle asymptotique vers le point d'équilibre. et garantit également une commande qui force le système à cette surface de glissment. et aussi l'empêcher de le quitter i.e (si $S_i = 0$ *avec* $i = 1, 2...6$ *nous conclrons que la stabilisation au sens de lyponove de l'erreur est vérifiée)*

2.4 Conclusion

Dans cette section, nous avons élaboré un modèle dynamique complet du quadrotor. Ce modèle montre une nature couplée, complexe, non linéaire, multi variable et sous actionné. Ce qui rend son contrôle relativement difficile. En revanche nous avons également cité l'algorithme de commande utilisé dans cette étude (Algorithme de consensus) .
Le chapitre suivant sera consacré à la simulation de notre modèle en utilisant MATLAB ainsi d'appliqué les commandes. Puis en utilisant la simulation en temps réel par la plateforme ROS .

CHAPITRE 3 Test et résultats de simulation

3.1 Introduction

Ce chapitre décrit les techniques de contrôle introduites dans le monde du quadrotor. Ce sont principalement des commandes classiques et des commandes robustes. le système sera étendu à une coopération des plusieurs véhicules dont un réseau de communication. Nous commencerons par présenter ces méthodes et cadrer notre travail afin d'atteindre les objectifs que nous avons fixés. Nous nous consacrons à exposer et analyser les différents résultats obtenus en utilisant les programmes de simulation réalisée . Ces programmes concernant :

— ***La commande PID d'un drone.***
— ***Contrôle par mode glisant d'un drone.***
— ***Commande d'une coopération des quadrirotors avec les algorithmes de consensus.***
— ***Simulation de plusieurs scénarios de trajectoire en temps réel avec la plateforme ROS***

Simulation d'une commande d'un seul quadrirotor

Dans cette simulation, nous avons choisi 2 types de commandes différents, l'une est linéaire (PID) et l'autre non linéaire (mode glissant) afin de pouvoir faire une conclusion finale.

3.2 Simulation d'une commande PID d'un seul quadrirotor

A base des équations de dynamique du quadrotor mentionnées dans le chapitre précédent, un contrôleur PID linéarisé est utilisé pour contrôler le quadrotor. La linéarisation est effectuée autour du point d'équilibre qui est le point stationnaire. La loi de commande est encore simplifiée en utilisant l'hypothèse du petit angle de :

$$\begin{cases} \sin(\alpha) \approx (\alpha) \\ \cos(\alpha) \approx 1 \end{cases} \tag{3.1}$$

Les angles de roulis et de tangage souhaités sont calculés en fonction de la position souhaitée dans le plan x-y en utilisant les équations :

$$\begin{cases} m(v_x.\cos(\psi) + v_y.\sin(\psi)) = \sin(\theta).U_1 \\ m(v_x.\sin(\psi)v_y.\cos(\psi)) = \sin(\phi)\cos(\theta).U_1 \\ m(v_z + g) = \cos(\phi).\cos(\theta).U_1 \end{cases} \tag{3.2}$$

La commande reçoit la position souhaitée du quadrirotor dans le plan x-y et calcul les angles comme suit :

$$\phi_d = \arctan(\frac{v_x.\sin(\psi_d)v_y.\cos(\psi_d)}{v_z + g}) \tag{3.3}$$

$$\theta_d = \arctan(\frac{v_x.\cos(\psi_d) + v_y.\sin(\psi_d)}{\sqrt{(v_x.\sin(\psi_d)v_y.\cos(\psi_d))^2 + (v_z + g)^2}}) \tag{3.4}$$

3.2.1 Résultat de simulation

Un contrôleur d'attitude et alttitude est introduit pour chaque angle d'orientation, donc une commande d'attitude fixe est testée. Nous choisissons le référence fixe $\phi_d = 20^o$, $\theta_d = 10^o$, $\psi_d = 10^o$ et $Z_d = 4$

Nous pouvons voir les résultats de la simulation dans les figures : (Figure.3.1), (Figure.3.2).

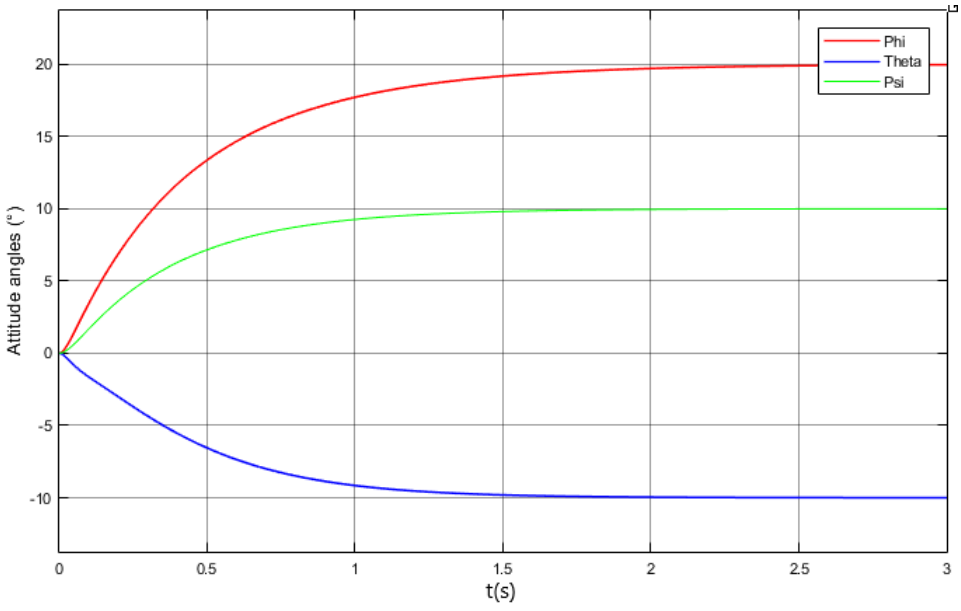


Figure 3.1 Résultat de simulation de vecteur d'orientation

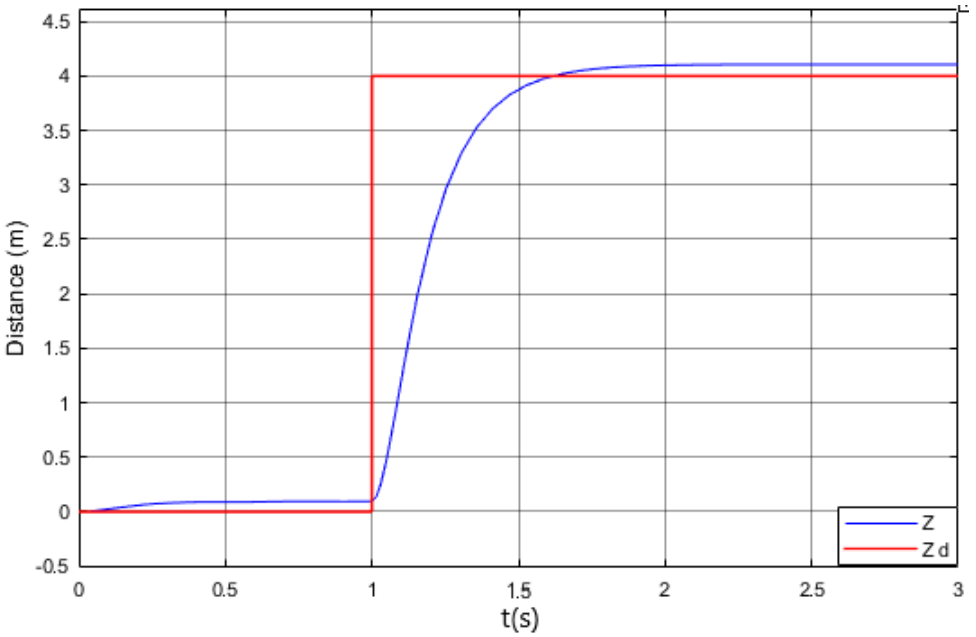


Figure 3.2 Résultat de simulation d'altitude avec PID

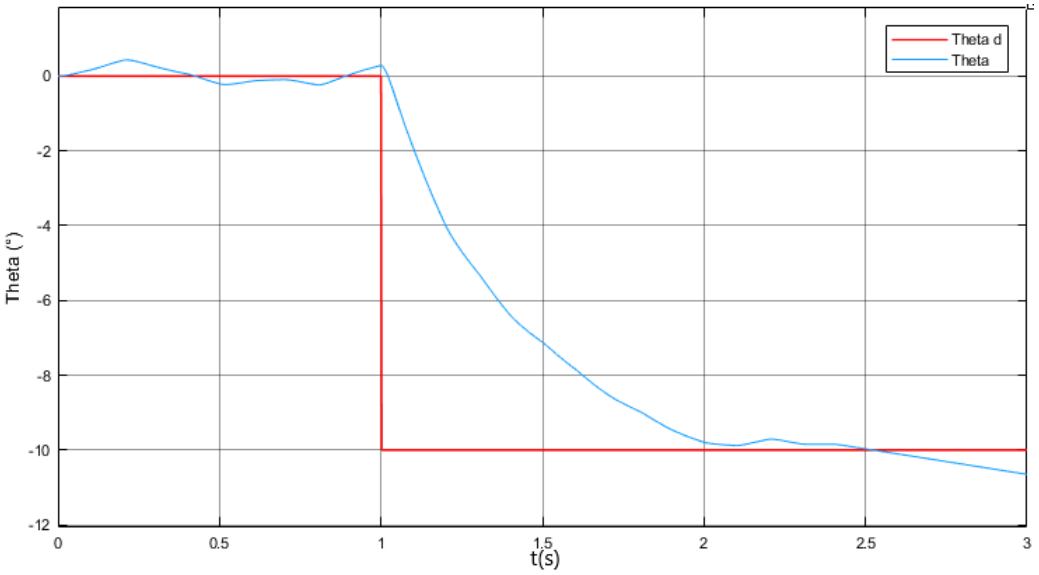


Figure 3.3 Angle θ avec présence de bruit

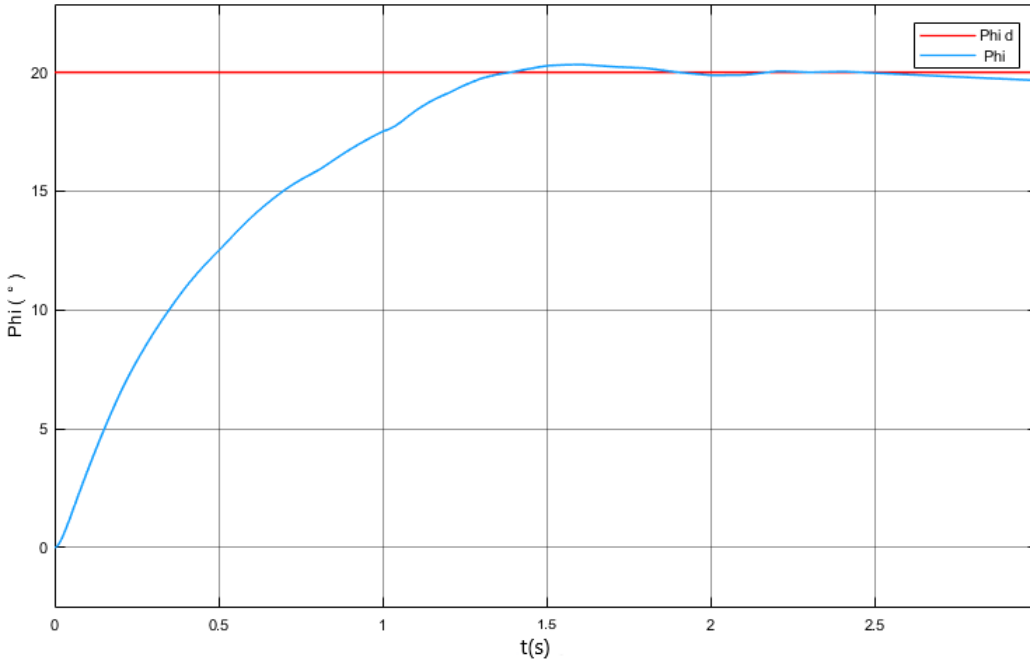


Figure 3.4 Angle ϕ avec présence de bruit

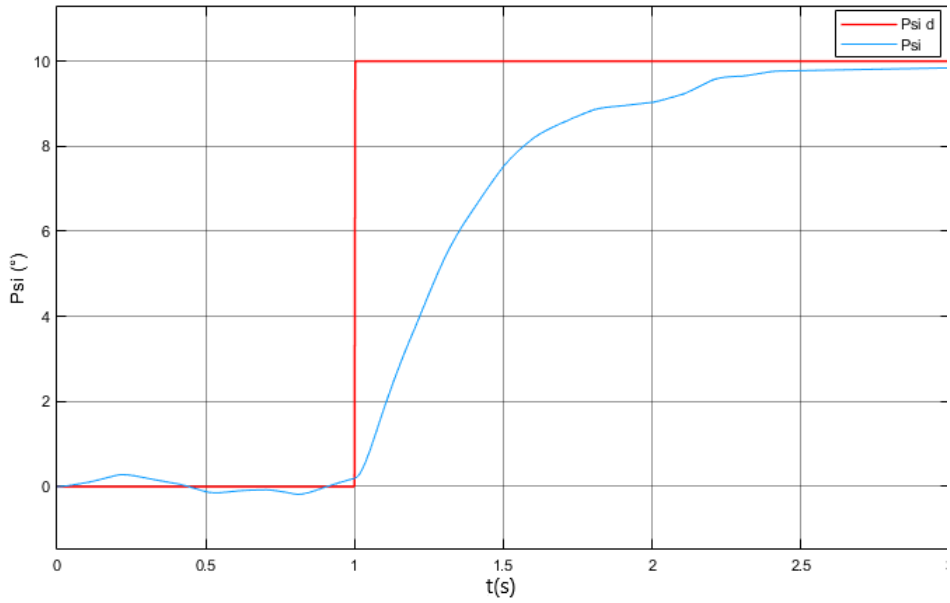


Figure 3.5 Angle ψ avec présence de bruit

Afin de réaliser un controle pour tous les états du systéme (Vecteur d'attitude et vecteur de position), Il faut introduire la notion de commande virtuel.

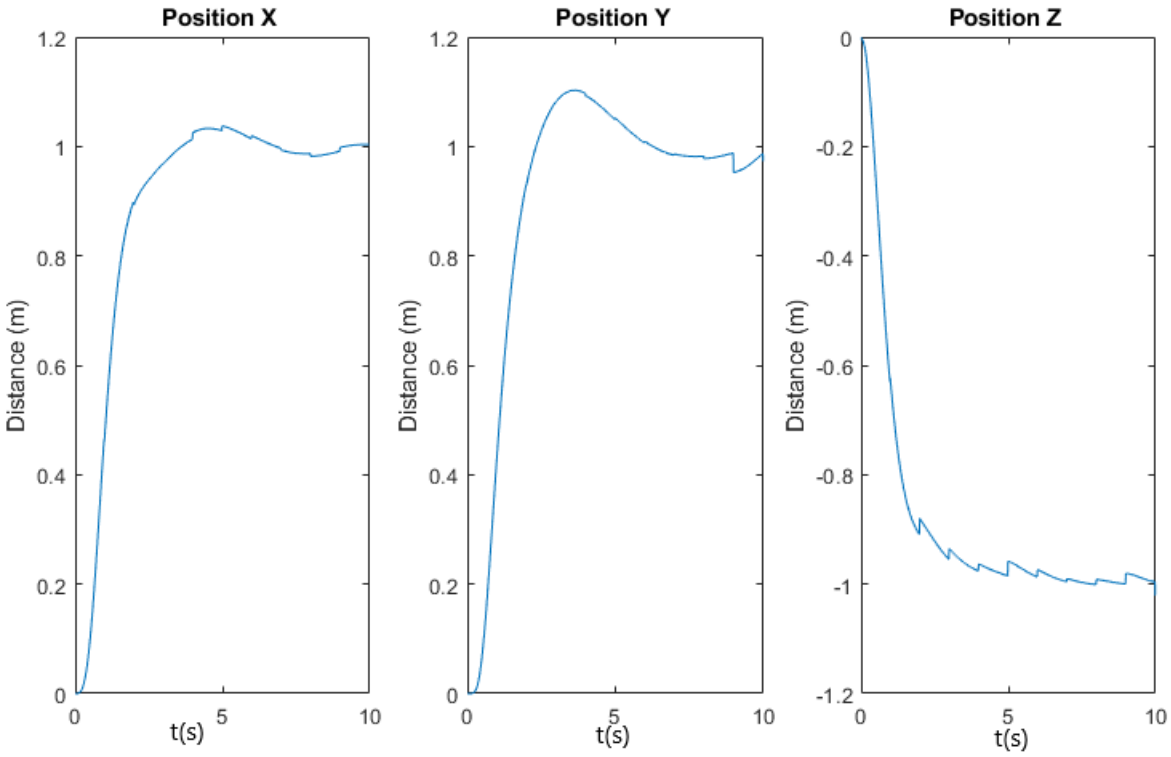


Figure 3.6 vecteur de position

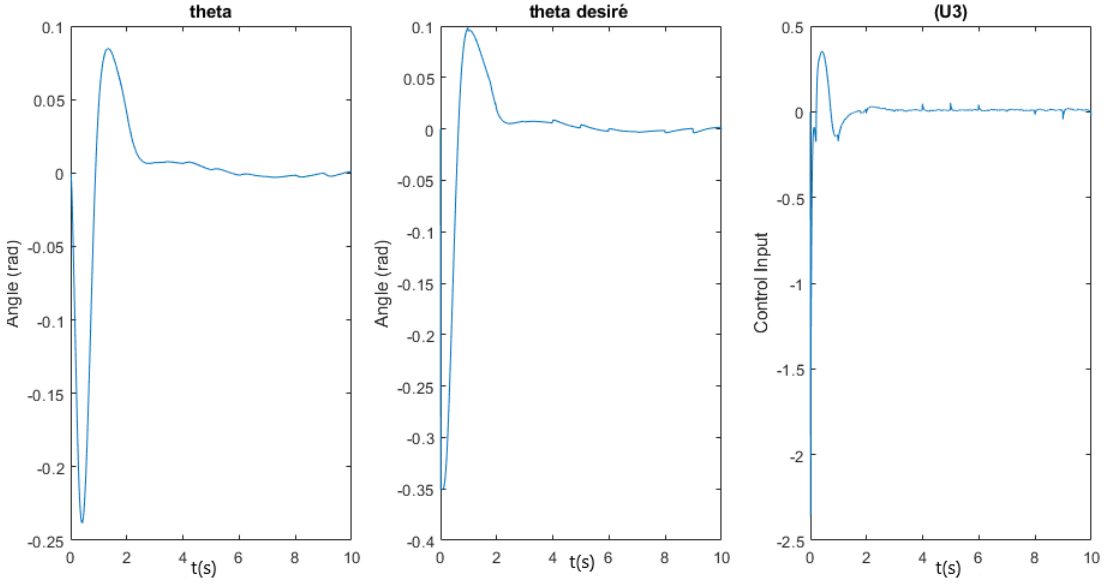


Figure 3.7 Résultat de simulation d'angle θ et la commande U3

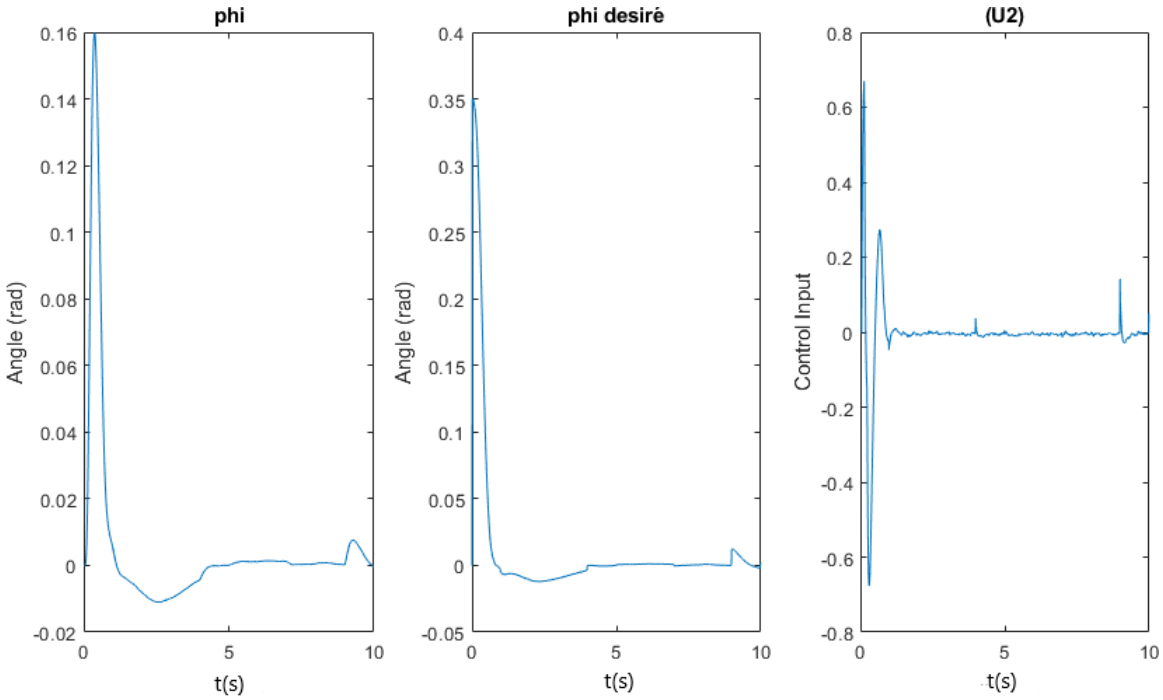


Figure 3.8 Résultat de simulation d'angle ϕ et la commande U2

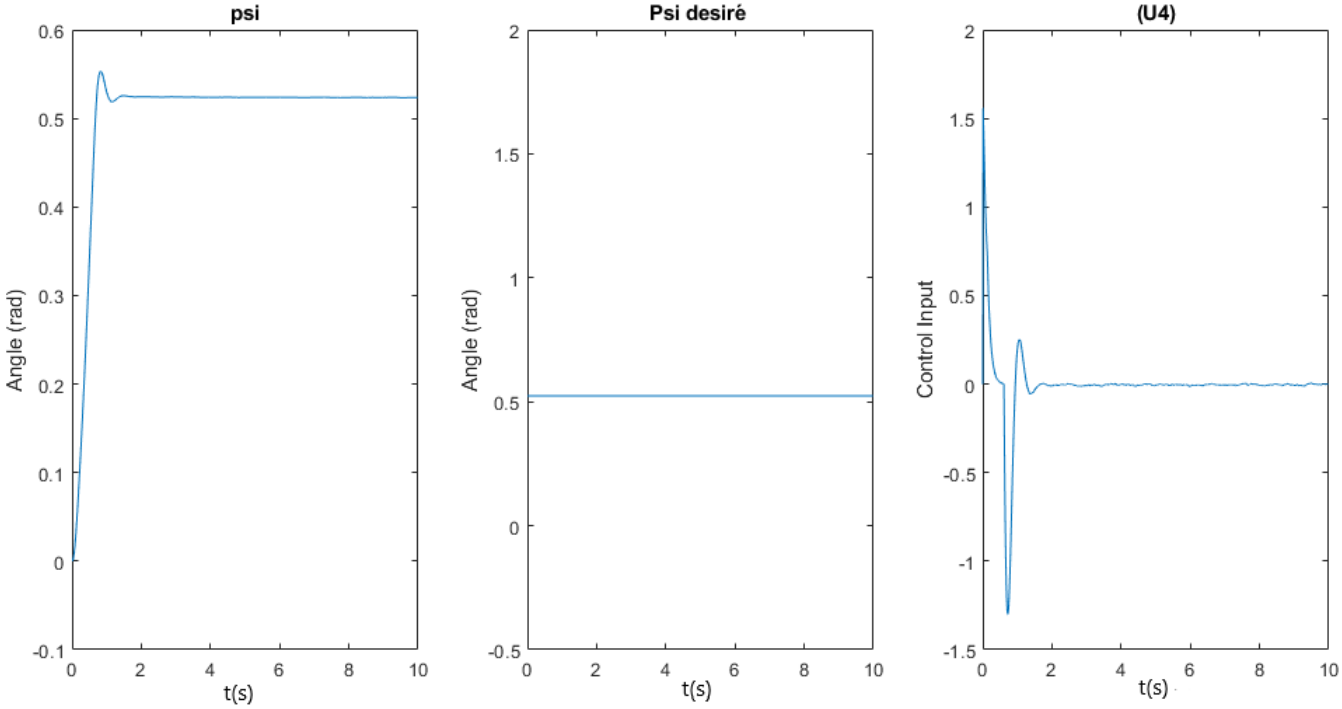


Figure 3.9 Résultat de simulation d'angle ψ et la commande U4

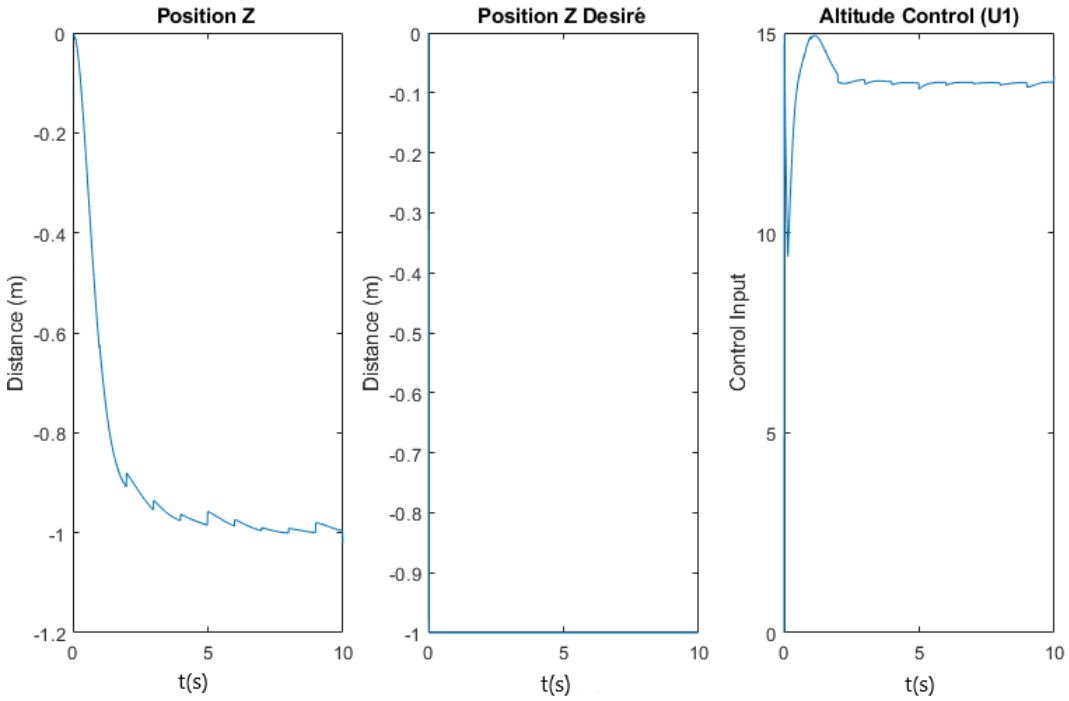


Figure 3.10 Résultat de simulation d'alttitude Z et la commande U1

3.2.2 Interprétation des résultats

On voit que notre commande arrive à stabiliser le quadrotor à la hauteur et angles désirées. Donc la commande PID est très efficace dans le cas des systèmes linéaires à paramètres constants ce qui correspond au vol stationnaire pour le quadrotor. Nous pouvons alors dans ce cas réaliser un vol autonome avec succès mais dans le cas des systèmes non linéaires ayant des variations paramétriques (Quelques perturbations ont été introduites) ces lois peuvent être insuffisantes (côté robustesse) car le régulateur perd son fonctionnement naturel à cause des bruits donc il y a une apparition d'une erreur statique en cas de mauvais choix des gains pour des perturbations assez grand.

3.3 Simulation d'une commande mode glissant

En utilisant les équations (2.30) et (2.31) et (2.23) et se basant sur [Bouabdallah Samir, 2007], [O.MEKKID, 2016], [Xu Rong, 2006], le systéme peut se mettre sous la forme :

$$\ddot{\xi} = f + U + \gamma \tag{3.5}$$

Le vecteur d'etat ou le vecteur d'orientation dans notre cas :

$$\xi = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_7 \\ x_8 \\ x_9 \end{bmatrix} \tag{3.6}$$

$$U = \begin{bmatrix} \frac{U_1}{m}(\cos(x_7)\sin(x_8)\cos(x_9) + \sin(x_7)\sin(x_9)) \\ \frac{U_1}{m}(\cos(x_7)\sin(x_8)\cos(x_9) - \sin(x_7)\sin(x_9)) \\ \frac{U_1}{m}(\cos(x_7)\cos(x_8)) \\ \frac{U_2}{j_x} \\ \frac{U_3}{j_y} \\ \frac{U_3}{j_z} \end{bmatrix} \tag{3.7}$$

$$f = \begin{bmatrix} 0 \\ 0 \\ -g \\ \left(\frac{I_y - I_z}{I_x}\right).x_{11}.x_{12} \\ \left(\frac{I_z - I_x}{I_y}\right)x_{10}.x_{12} \\ \left(\frac{I_x - I_y}{I_z}\right)x_{10}.x_{11} \end{bmatrix} \tag{3.8}$$

$$\gamma = \begin{bmatrix} \gamma_x \\ \gamma_y \\ \gamma_z \\ \gamma_\phi \\ \gamma_\theta \\ \gamma_\psi \end{bmatrix} \tag{3.9}$$

Avec un petit changement

$$\alpha = f + U \tag{3.10}$$

(3.5) devient :

$$\ddot{\xi} = \alpha + \gamma \tag{3.11}$$

La dynamique de surface (2.31) nous donne :

$$\dot{S}_i(t) = \ddot{\epsilon}_i.(t) + \lambda\dot{\epsilon}_i(t) \qquad\qquad i = 1, 2...6 \tag{3.12}$$

En remplaçant par (3.11) :

$$\dot{S}_i(t) = \alpha + \gamma - \ddot{\alpha_{di}} + \lambda.(\dot{\xi} - \dot{\xi}_d) \qquad i = 1, 2...6 \tag{3.13}$$

Dans cette étape il est bien clair que nous devons choisir une dynamique telque :

$$\dot{S}_i(t) = -A_i.sign(S_i(t)) - B_i.S_i(t). \qquad i = 1, 2...6 \tag{3.14}$$

A_i , B_i sont des constant de reglage , et la fonction sign(.) est définie comme :

$$sign(x) = \begin{cases} 1 & si \quad x > 0 \\ 0 & si \quad x = 0 \\ 1 & si \quad x < 0 \end{cases}$$

Par identification (3.13) et (3.14) on obtient enfin les six commandes :

$$\alpha_i(t) = \eta_i - A_i.sign(S_i(t)) - Q_i.S_i(t) \qquad i = 1, 2...6 \tag{3.15}$$

ou η_i :

$$\eta_i = \ddot{\xi}_{d_i}(t) - \lambda.(\dot{\xi} - \dot{\xi}_d) \qquad i = 1, 2...6 \tag{3.16}$$

3.3.1 Résultat de simulation

Pour la première simulation, nous avons donné comme référence un point de coordonnées (1, 2, 4) et sans perturbation .
Nous pouvons voir les résultats de la simulation sur les figures : (Figure.3.11), (Figure.3.12), (Figure.3.13) .

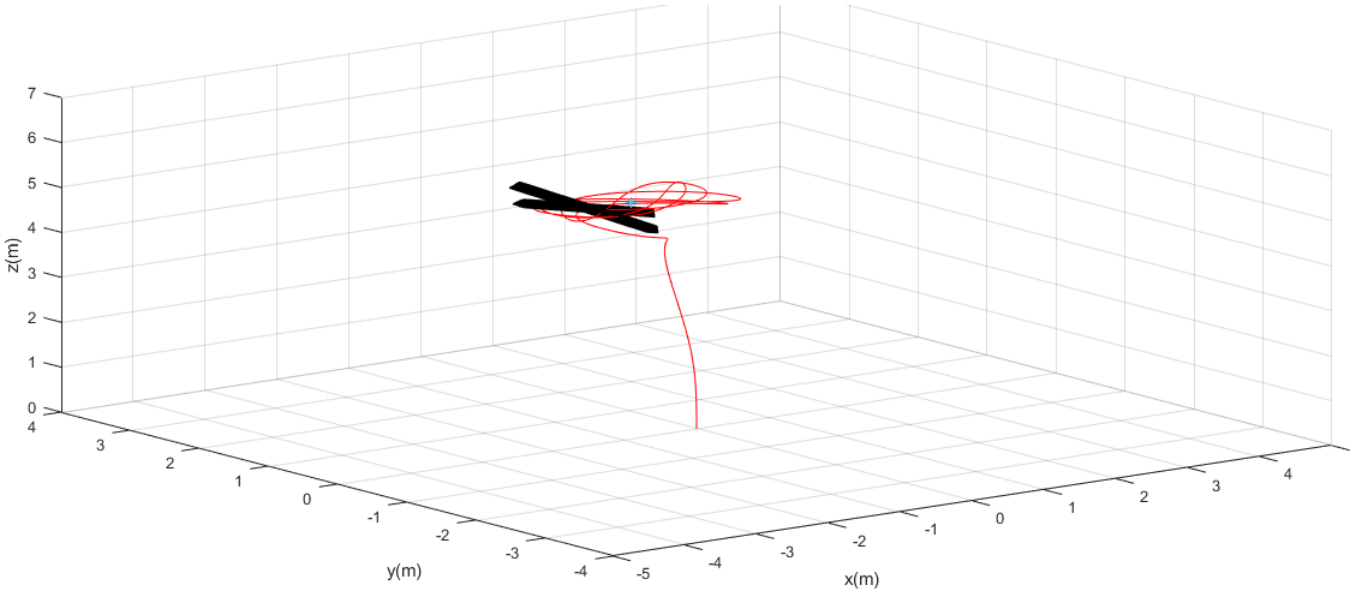


Figure 3.11 Résultat de simulation 3D sans présence de bruit

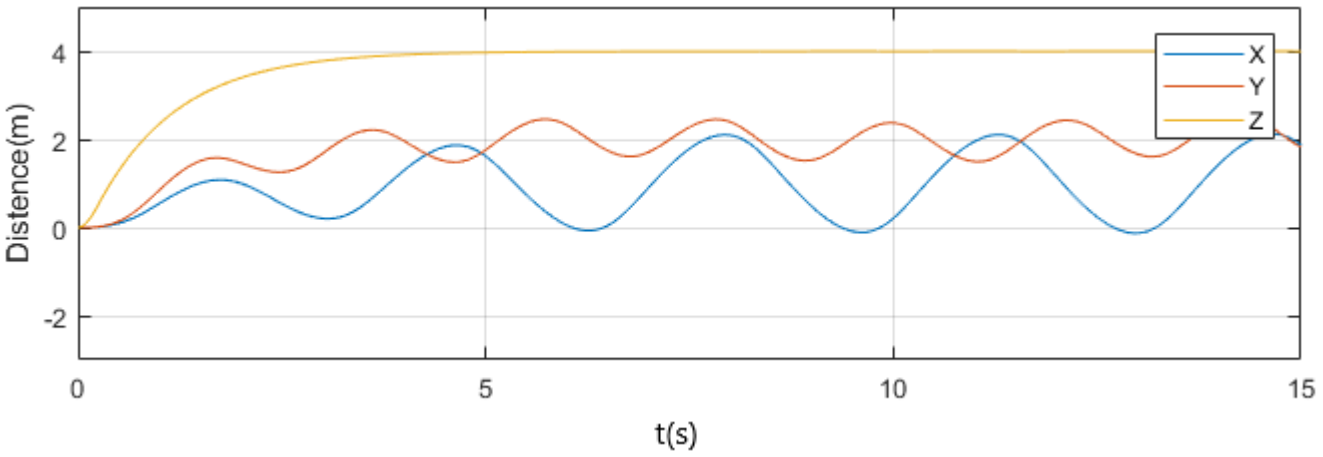


Figure 3.12 Résultat de simulation de vecteur postion

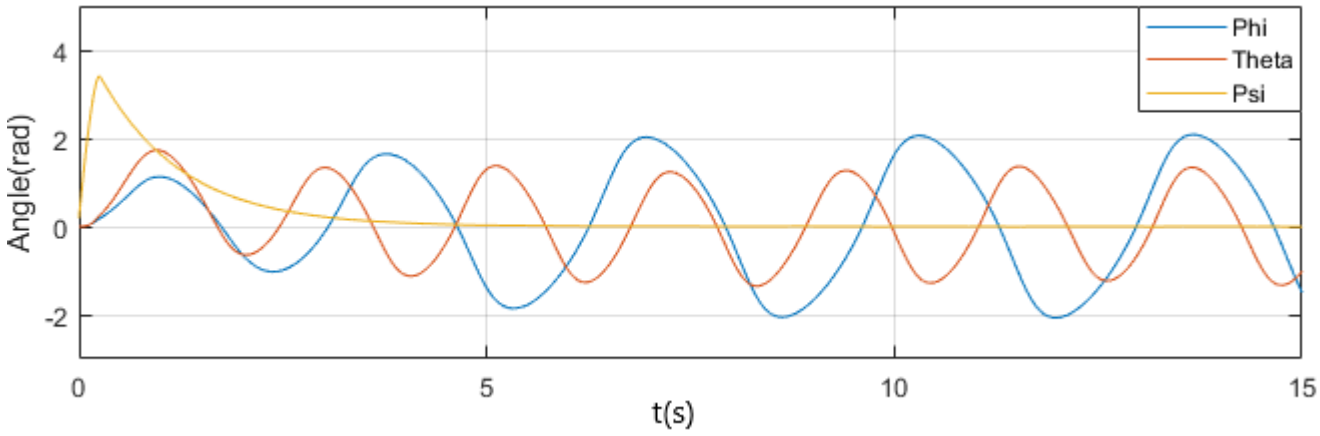


Figure 3.13 Résultat de simulation de vecteur d'orientation

3.3.2 Interprétation des résultats

On constate que le quadrirotor atteint rapidement le point de référence malgré la présence des perturbations. Donc c'est une commande robuste qui rejette les bruits de mesure mais on distingue des ondulations au niveau de signal due au phénomène de chattering causé par la commande discontinue.
Une combinaison convenable consiste à remplacer la fonction discontinue sign(.) de la loi de commande par une fonction continue qui permet d'éliminer le phénomène de broutement. Parmi les fonctions les plus utilisées on trouve :

— La fonction saturation

$$sat(x,\gamma) = \begin{cases} sign(\gamma) & si \quad |x| > \gamma \\ \dfrac{x}{\gamma} & si \quad x < \gamma \end{cases}$$

— La fonction pseudo-signe

$$psigne(x,\gamma) = \frac{x}{|x| + \gamma}$$

— La fonction arctangente

$$sign(x) = \frac{2}{\pi} \arctan(\frac{x}{\gamma})$$

3.3.3 *Résultats de simulation avec une perturbation et en changeant la fonction sign(.)* :

Par conséquent, pour une validation plus précise de notre contrôle, un deuxième scénario a été réalisé où nous avons donné comme référence un point de coordonnées (1, 2, 4) et en présence de perturbation.on a changer aussi le signal sign(.) par un psign(.,.) afin de minimiser le phénomène de chattering
Nous pouvons voir les résultats de la simulation sur les figures :

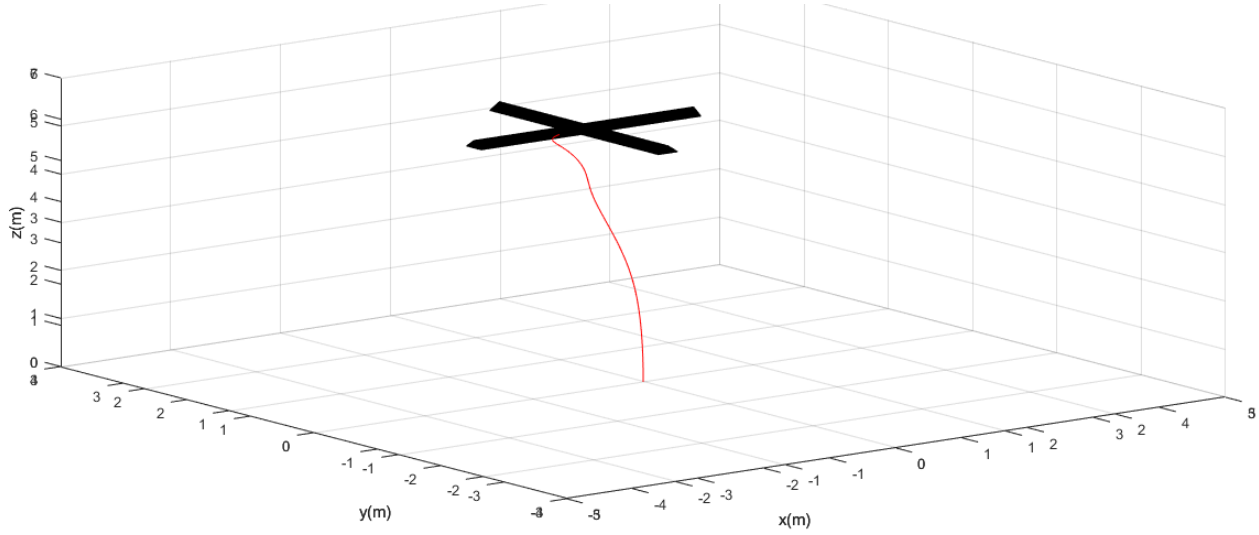


Figure 3.14 Résultat de simulation 3D avec présence de bruit

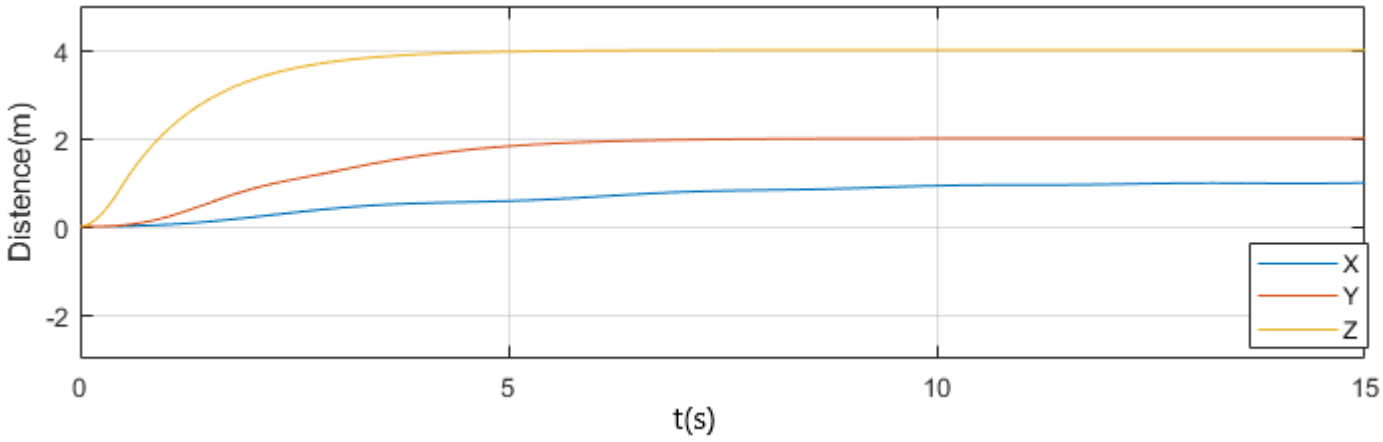


Figure 3.15 Résultat de simulation de vecteur postion

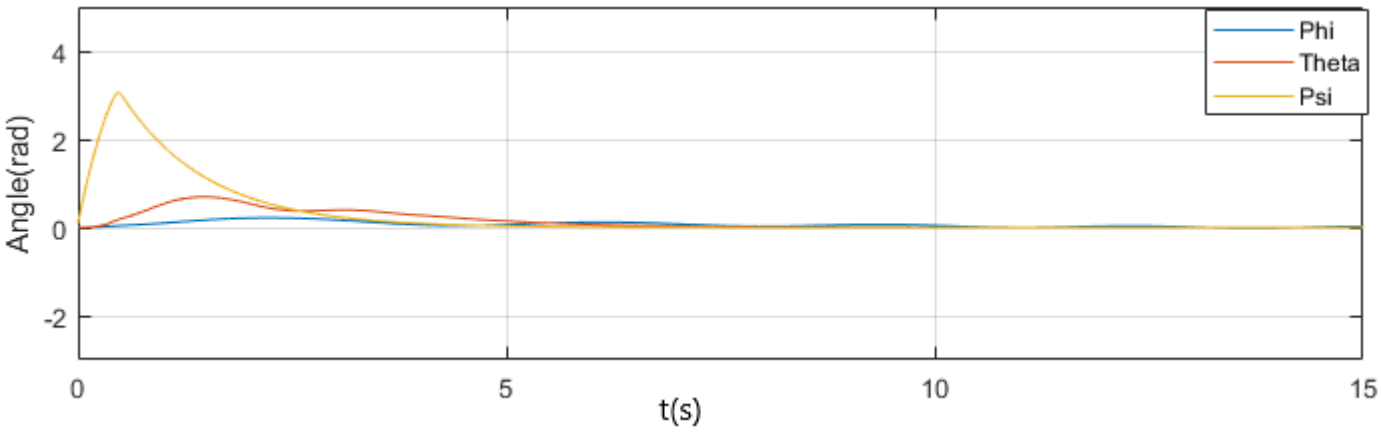


Figure 3.16 Résultat de simulation de vecteur d'orientation

3.3.4 Planification de trajectoire pour un seul quadrotor

Nous représentons dans cette partie la simulation d'une planification de trajectoire du drone en fonction de temps. Le chemin conduisant le drone du point départ au point d'arrivée.

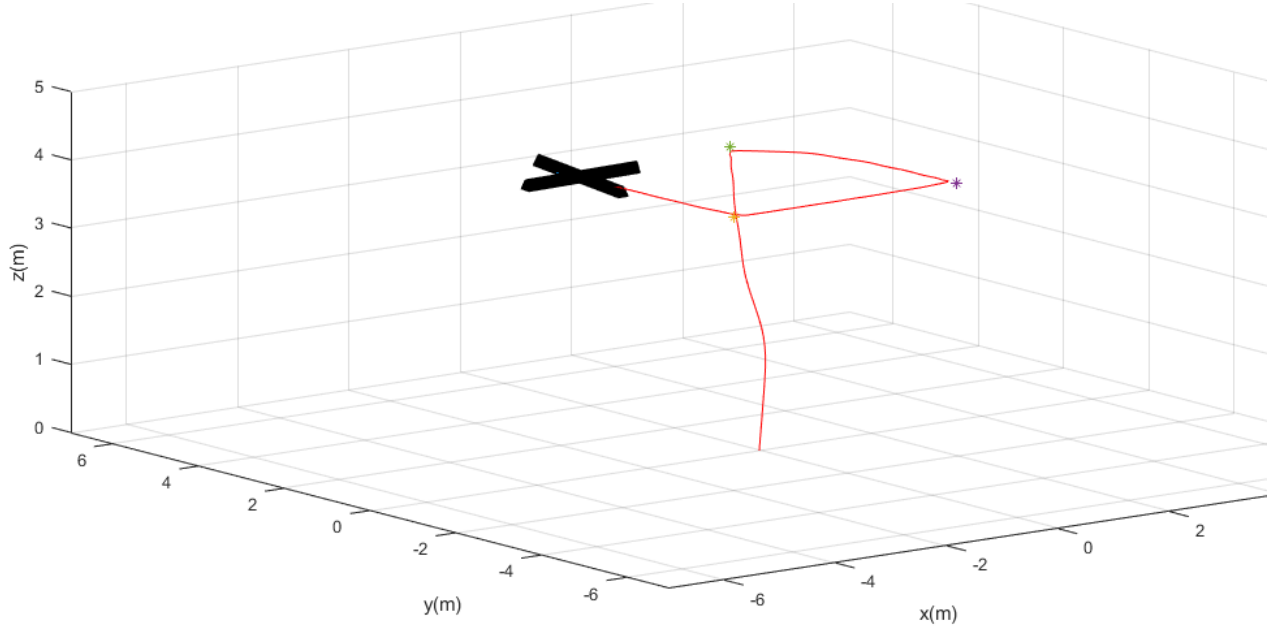


Figure 3.17 Planification de trajectoire du drone

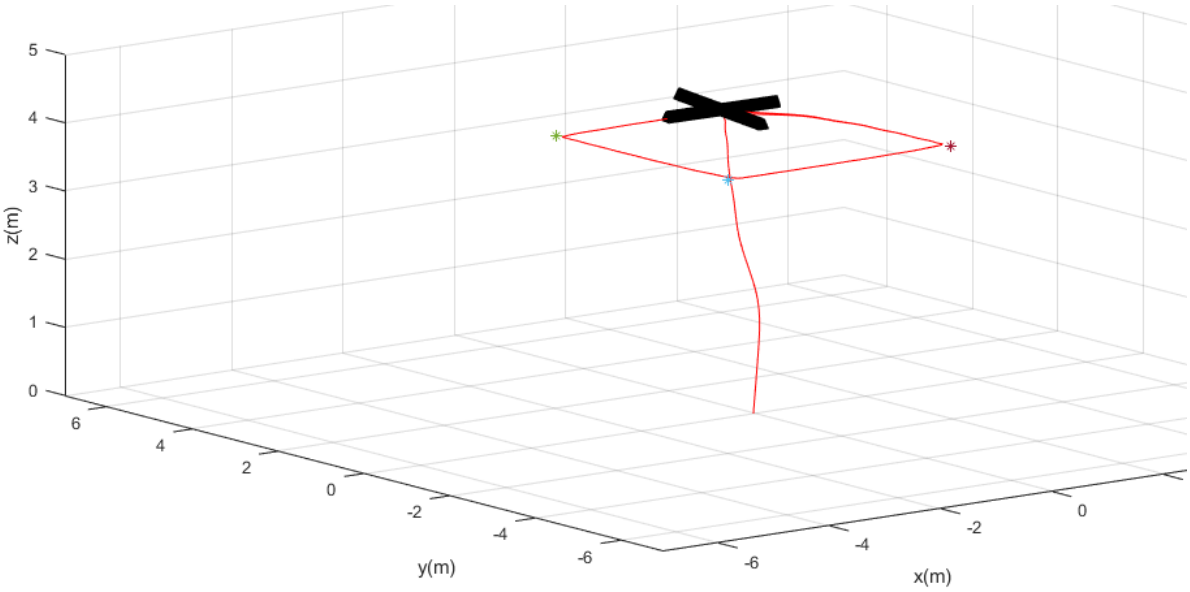


Figure 3.18 Résultat 3D d'une planification de trajectoire

3.4 Simulation d'une commande d'une coopération des quadrirotors

3.4.1 Algorithmes de Consensus pour les systèmes de second ordre

Le drone est modélisé comme une masse ponctuelle, qui résume le problème dans une commande d'un système multi-agents. (2.2).

Creation de graph

Nous devons d'abord générer un graph qui modélise le type de formation. Les types les plus utilisées dans la création d'une coopérative de 3 à 4 drones sont les formes lignes ou les formes en v comme indiqué (3.19)

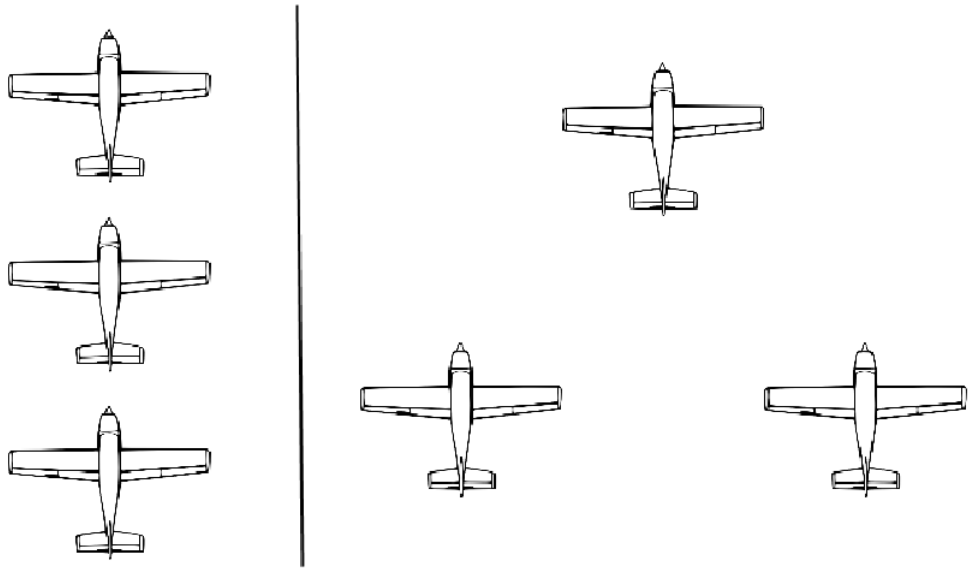


Figure 3.19 Type des formations utilisée

Pour notre cas nous avons choisi $n = 4$ donc nous avons une modélisation de 4 drones. et il y a $2^{\frac{N.(N-1)}{2}}$ graph/matrice possible.
La matrice (3.17) indique les interconnexions de cette formation, où chaque ligne représente une connexion entre les drones. Si une ligne i a une entrée non nulle dans la colonne j, alors le drone i suit le drone j .

$$Adjacency\ matrixes = \begin{bmatrix} 0\,0\,0\,0 \\ 0\,0\,1\,1 \\ 0\,1\,0\,0 \\ 0\,1\,0\,0 \end{bmatrix} \tag{3.17}$$

Cette matrice, il est aussi nommés la matrice d'adjacence.

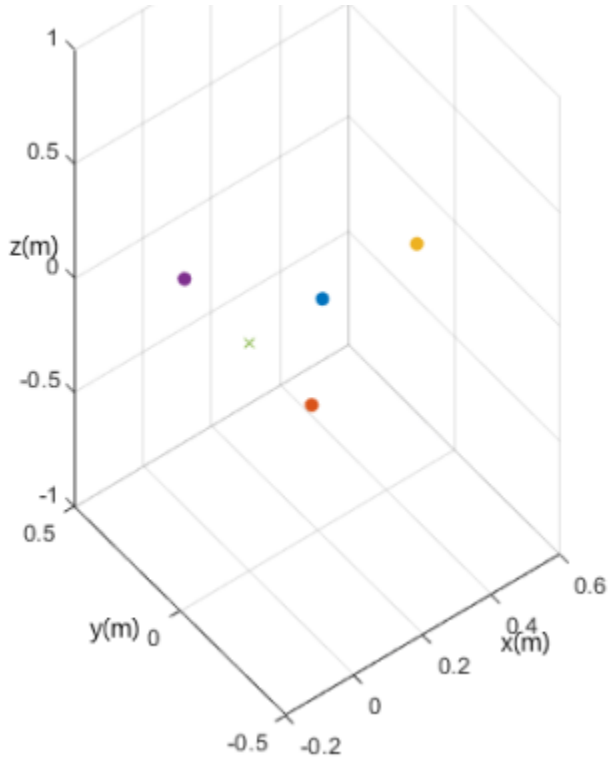


Figure 3.20 La modélisation 3D de la matrice d'adjacence (3.17)

3.4.2 L'implémentation d'algorithme de Consensus sur les états relatifs

Dans un $1^{ére}$ scénario une implémentation d'algorithme de Consensus, permet de converger les états de tous agents à un état final commun avec un simple model d'ordre 2.

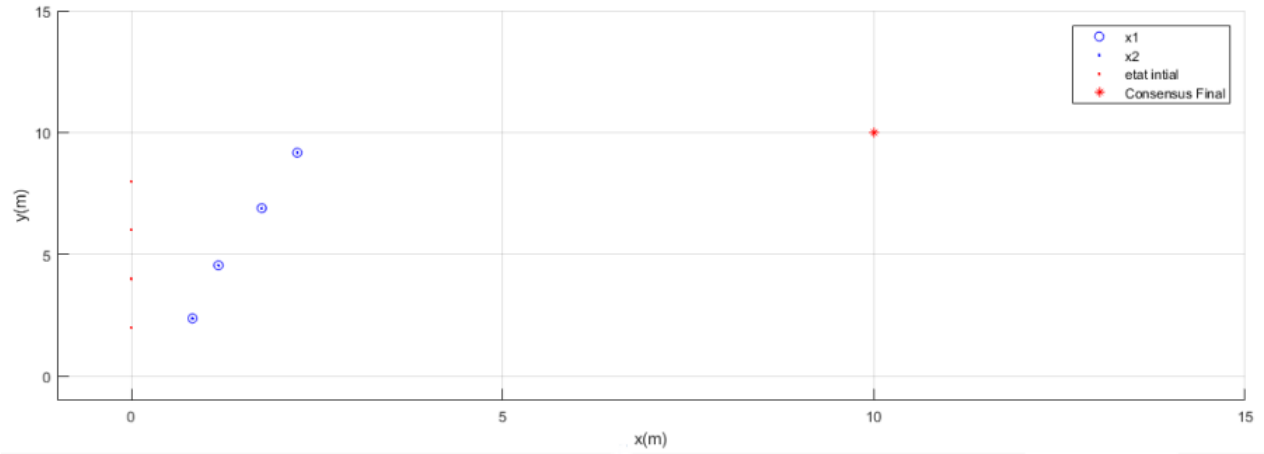


Figure 3.21 L'implémentation d'algorithme de Consensus

On a N agents, ou chaque agent est décrit par sa dynamique donnée par (2.4) $i = 1, \ldots N$. Tel que x_i est le vecteur d'état de l'agent d'index i, d'autre part le vecteur d'état est selon la dynamique de chaque agants defini comme :

$$x_i = \begin{bmatrix} x_{(i,1)} \\ x_{(i,2)} \\ \ldots \\ x_{(i,n)} \end{bmatrix} \tag{3.18}$$

3.4.2.1 Résultat de simulation

Pour un systeme de 4 agents avec la dynamique (3.19) :

$$\begin{cases} \dot{x_i} = \begin{bmatrix} -2 & 2 \\ -1 & 1 \end{bmatrix} x_i + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} u_i \\ y_i = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} x_i \end{cases} \tag{3.19}$$

Nous pouvons voir les résultats d'implémentation d'algorithmes du consensus en 2D ou les états convergents vers un seul point final dans les figures : (3.22),(3.23).

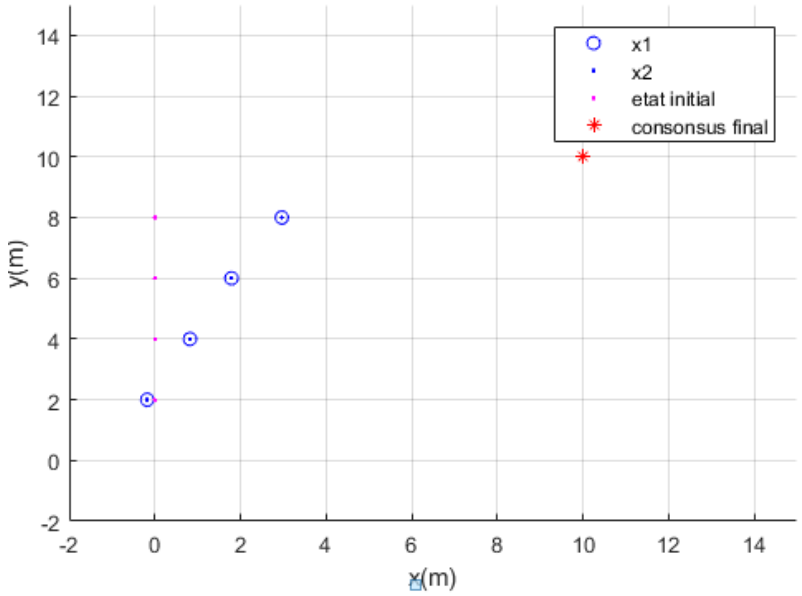


Figure 3.22 (A) Initialisation des vecteurs d'état

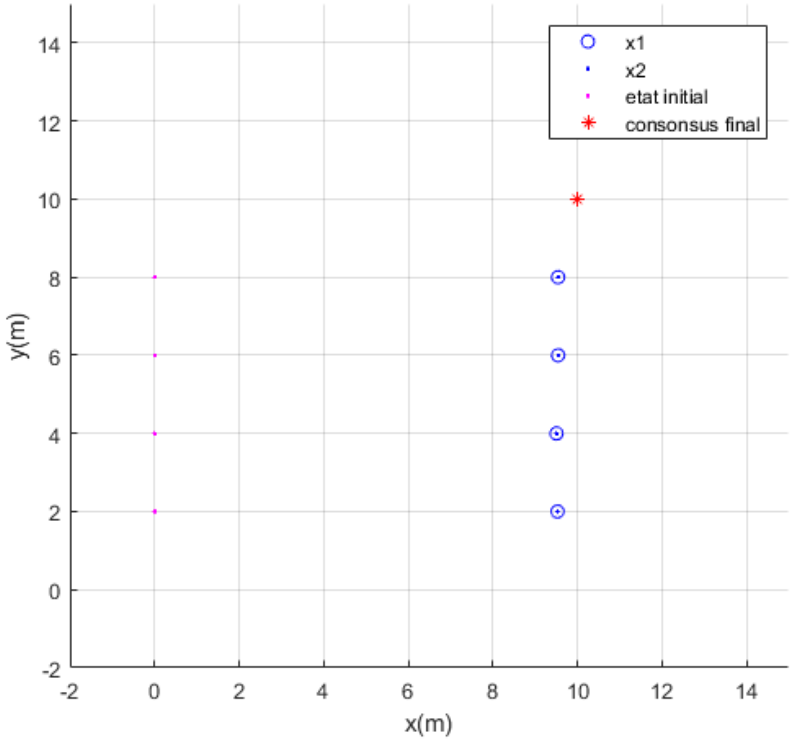


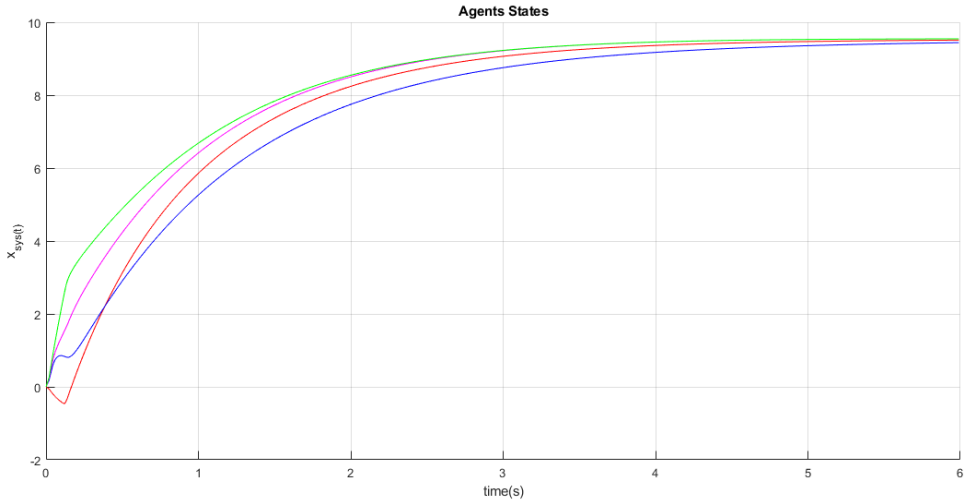
Figure 3.23 (B) Résultat de convergence

Durant cette expérience nous testons la loi de commande utilisée pour une valeur

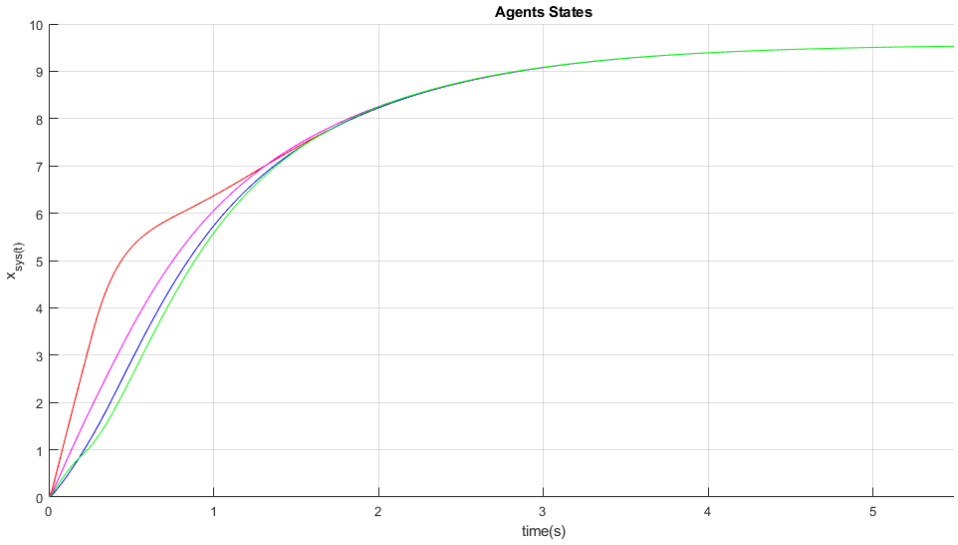
théorique du consensus final selon l'équation :

$$\lim_{t \to \inf} x_i = e^{A.t}. \sum r_i.x_i(0) \tag{3.20}$$

r_i est le vecteur propre de la matrice adjacence..

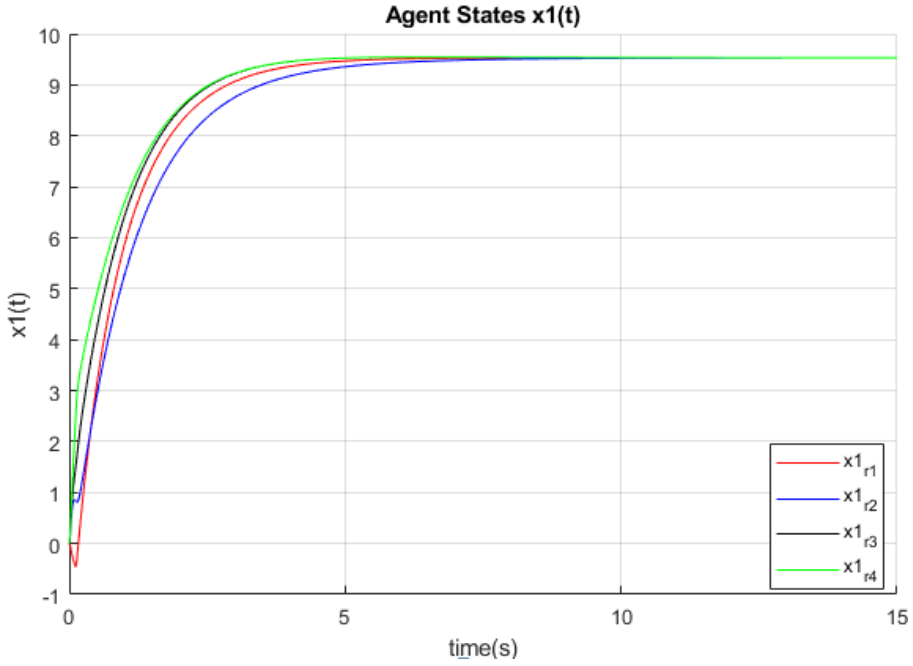


(A)

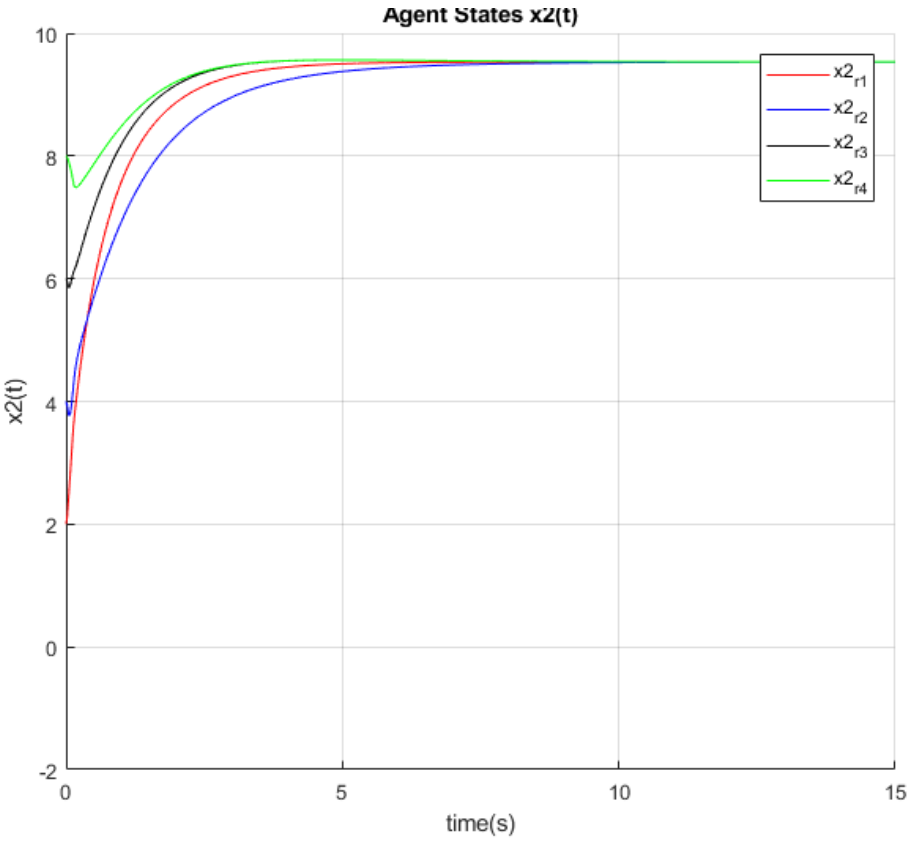


(B)

Figure 3.24 Effet du poids de couplage c sur l'accélération de la convergence



(A)



(B)

Figure 3.25 Signaux de convergence d'états pour chaque agent

3.4.3 Interprétation des résultats

Tous les états convergent vers un point de consensus final qui assure la stabilité de (2.8). Théoriquement, cette valeur peut être calculée par l'équation (3.20) donc elle dépend de valeur initiale des états de chaque agent.
Pour notre cas :

$$x_i(0) = \begin{bmatrix} 0 \\ 2kk \end{bmatrix}$$

kk est le nombre de quadrotors.
La valeur poid du couplage noté c joue le rôle d'accélérateur de convergence et du robustesse de protocole (3.24). Plus que la force de couplage est supérieur à la valeur c_{seuil} qui est donné par l'équation (3.21) le consensus atteint plus rapidement.

$$c_{seuil} = \frac{1}{Min_{2,\ldots,N} Re\,(\lambda_i)} \tag{3.21}$$

3.4.4 Implémentation de l'algorithme consensus avec échange relatif des musures des sorties et un modèle de référence.

Un $2^{éme}$ scénario ou les agents échangents les sorties y_i en utilisant un modèle de référence.
Ils échangent des informations en utilisant l'équation :

$$\widehat{\gamma}_i = c[\sum_{j=1}^{N} a_{ij}(y_i - y_j) + d_i(y_i - y_r)] \tag{3.22}$$

et la commande dans ce cas :

$$u_i = k.v_i + U_r \tag{3.23}$$

U_r un signal de réfrence

3.4.4.1 Résultat de simulation

Dans cette partie nous présentons les résultats recueillis pour les différentes scénario utiliseé en explicitant le protocole expérimental et les outils utilisés.
Pour un $1^{ère}$ essai les agents échangent les sorties y_i en utilisant l'équation (3.24), et la

valeur finale de consensus n'est pas contrôlée.

$$\widehat{\gamma}_i = c[\sum_{j=1}^{N} a_{ij}(y_i - y_j)] \tag{3.24}$$

La commande dans ce cas :

$$u_i = k.v_i \tag{3.25}$$

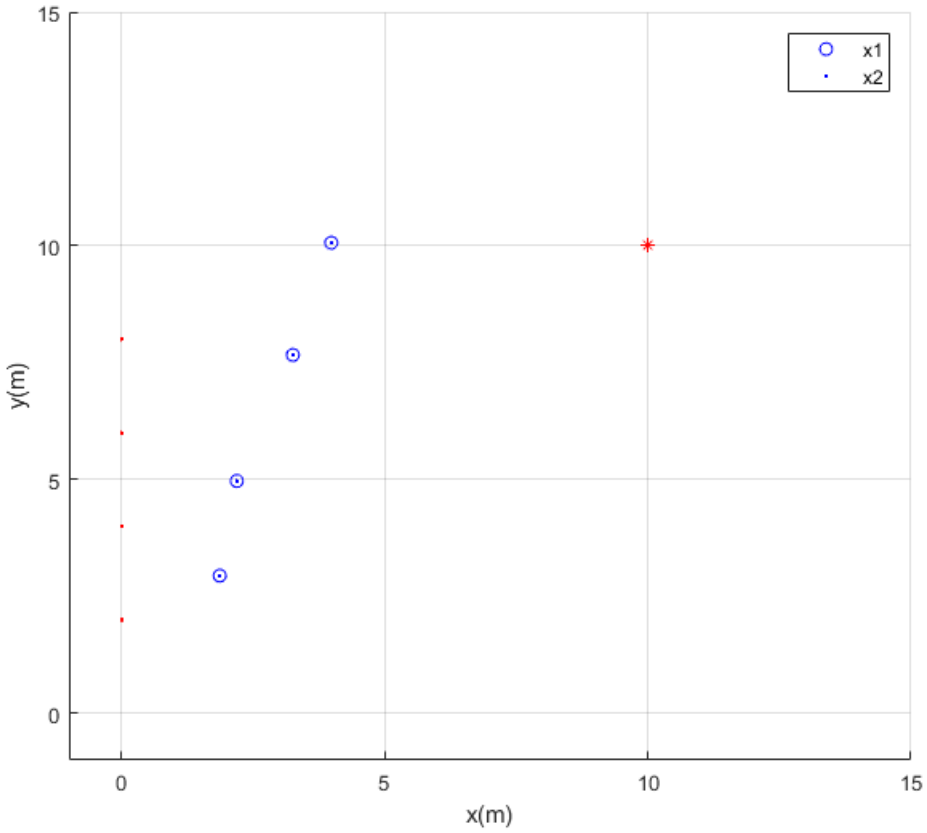


Figure 3.26 Initialisation des états de systéme d'implémentation d'algorithme de Consensus sur les sorties relatifs sans signal de réference

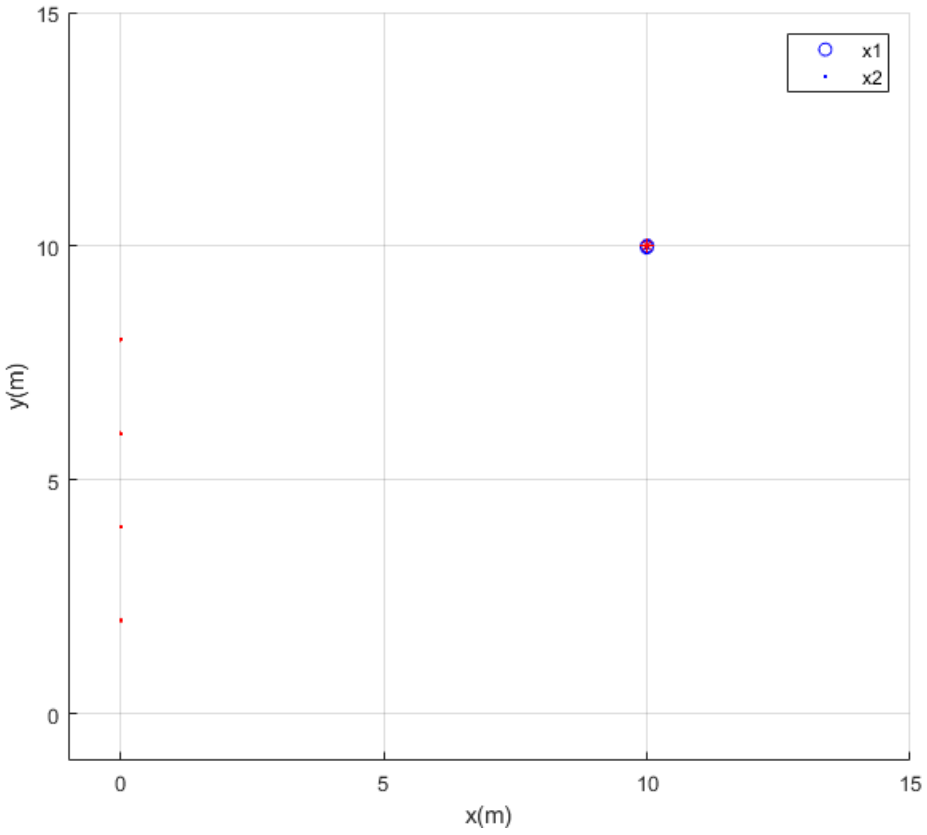


Figure 3.27 Résultat de simulation d'implémentation d'algorithme de Consensus sur les sorties relatifs sans signal de réference

Comme un $2^{éme}$ essai et en utilisant le même graphe d'agents et la même dynamique et en prenant (3.26), (3.27) comme des signaux d'initialisation de commande et d'état de réference :

$$u_{ref} = 0.5.\cos(2.\pi.0.02.t) \tag{3.26}$$

$$x_{ref}(0) == \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{3.27}$$

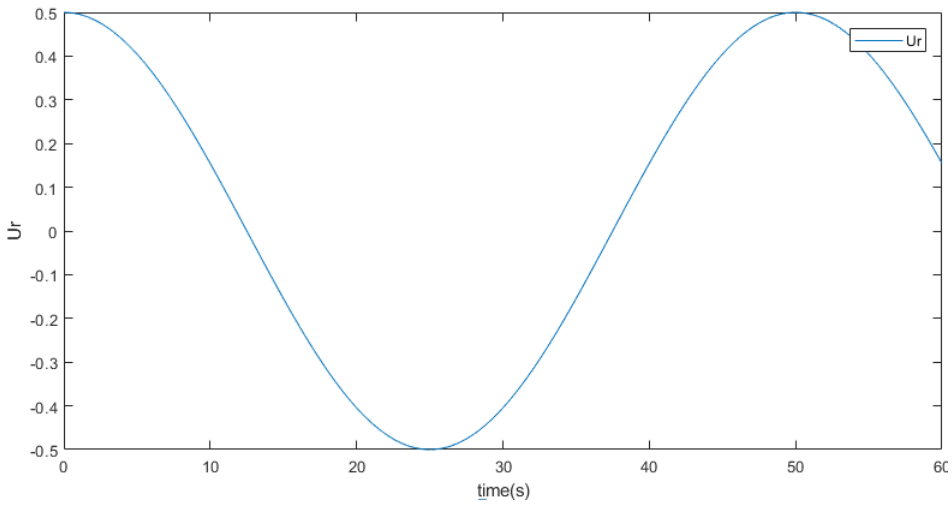


Figure 3.28 Signal de réference Ur.

Nous pouvons voir les résultats de implémentation de l'algorithme avec modéle de réference dans les figures, (Figure.3.31) :

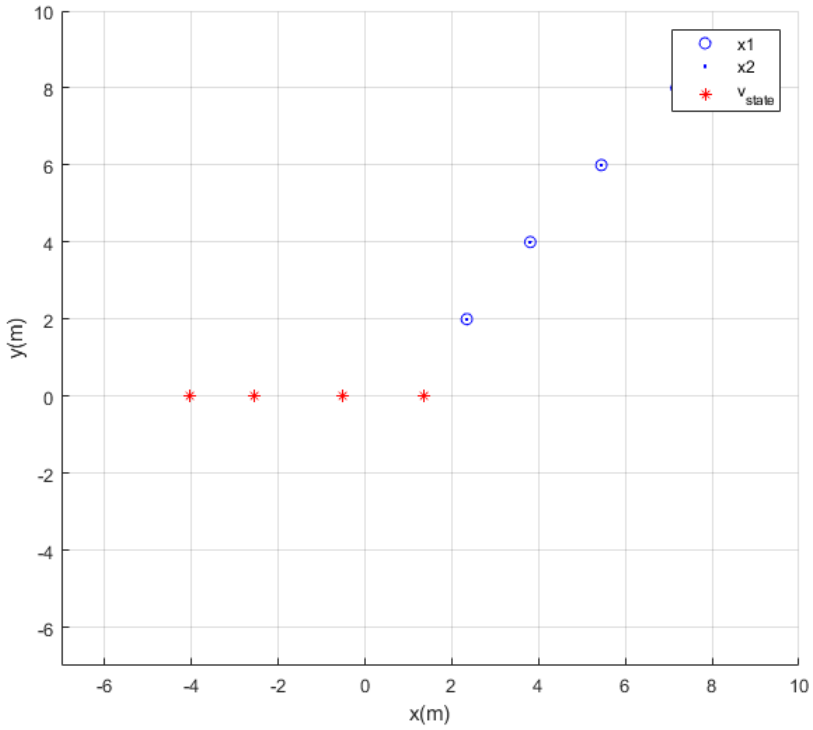


Figure 3.29 Initialisation des états d'implémentation d'algorithme de Consensus sur les sorties relatifs avec modéle de réference

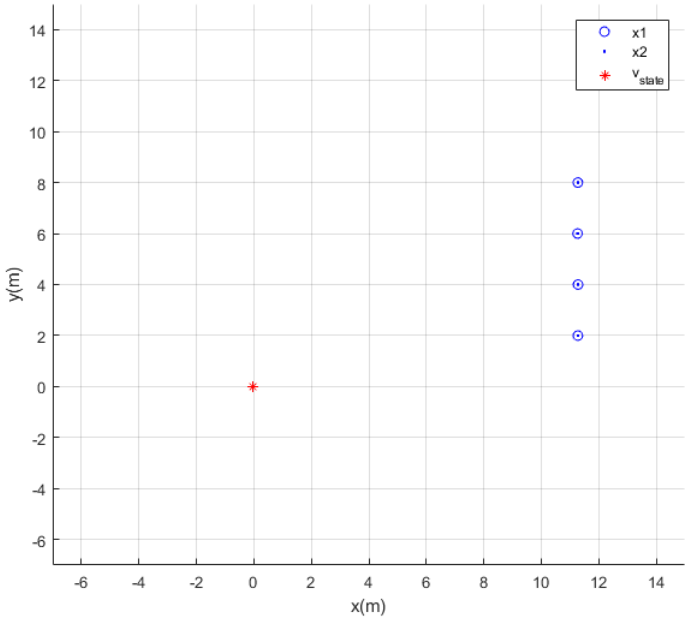


Figure 3.30 Résultat de simulation d'implémentation d'algorithme de Consensus sur les sorties relatifs avec modéle de réference

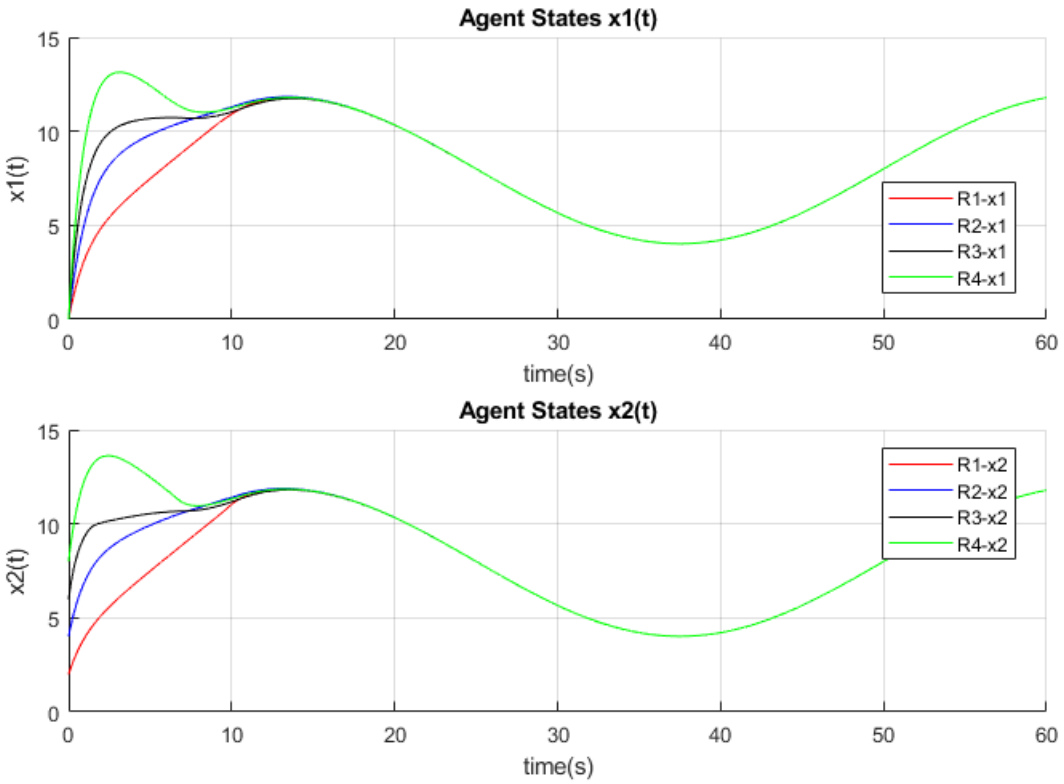


Figure 3.31 Signaux de convergence des états

La figure (3.32) montre également la commande U

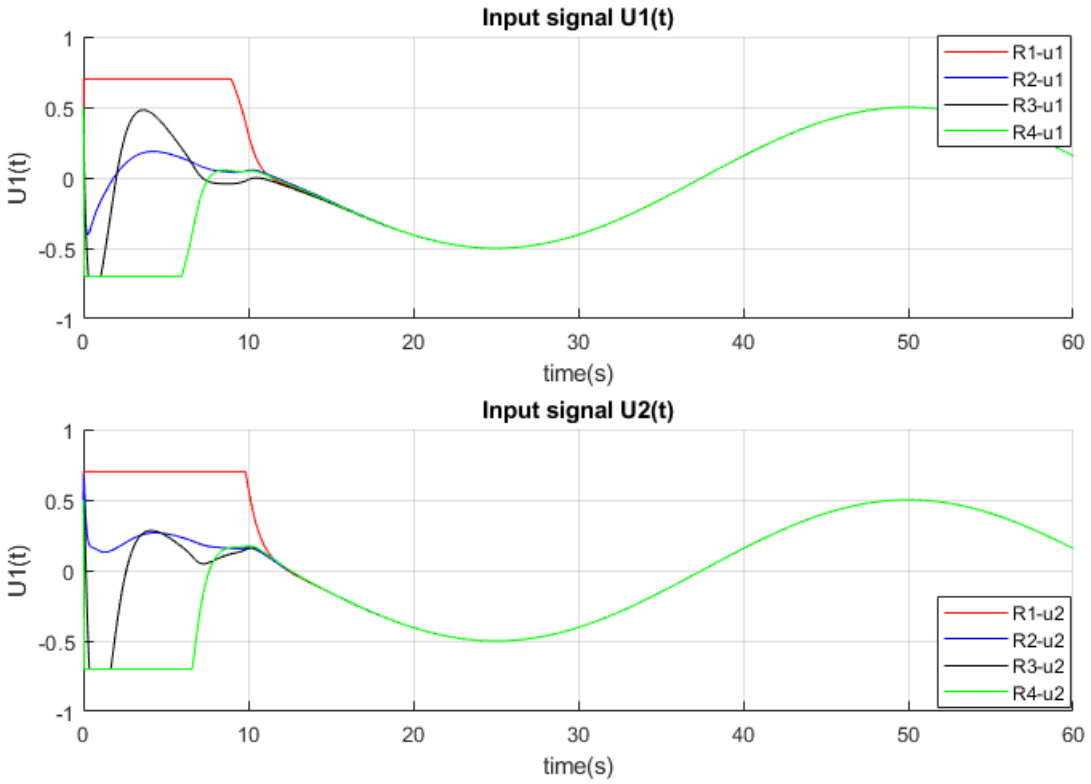


Figure 3.32 commande U

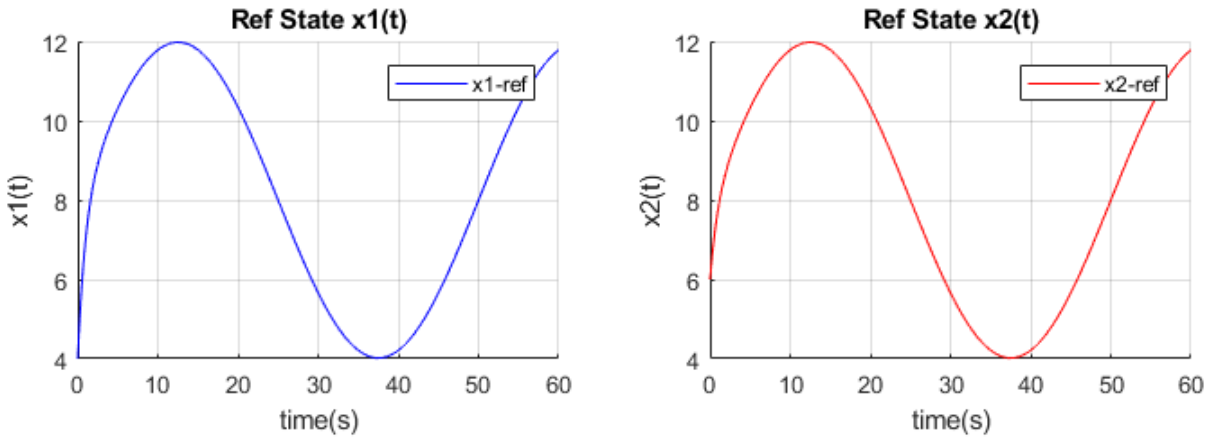


Figure 3.33 Les états de modél de réfrence

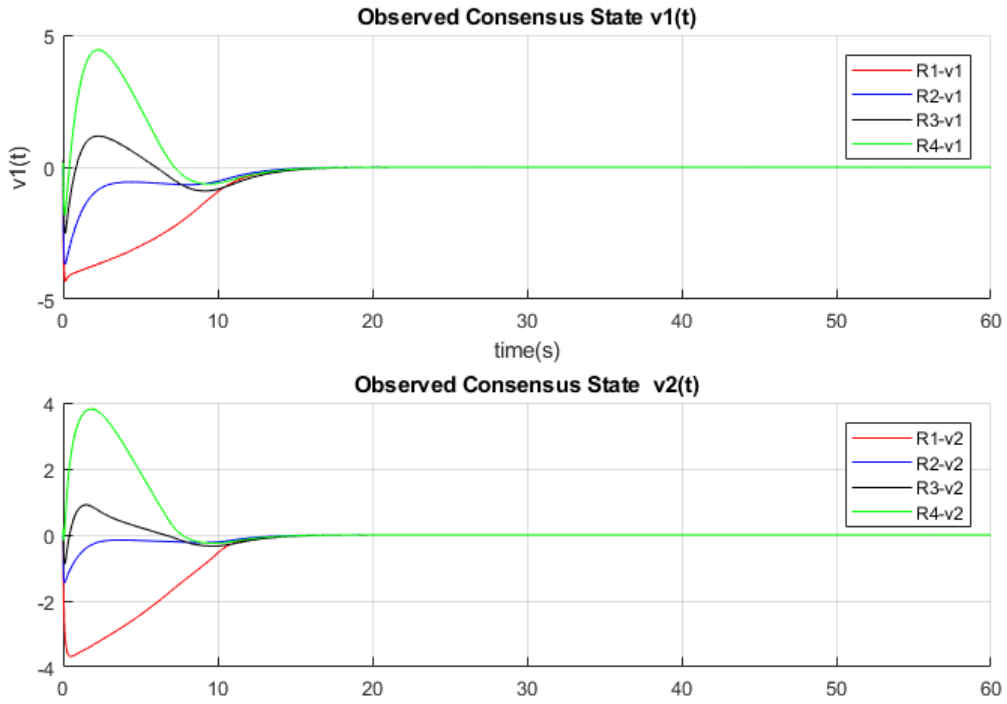


Figure 3.34 Les états d'observé

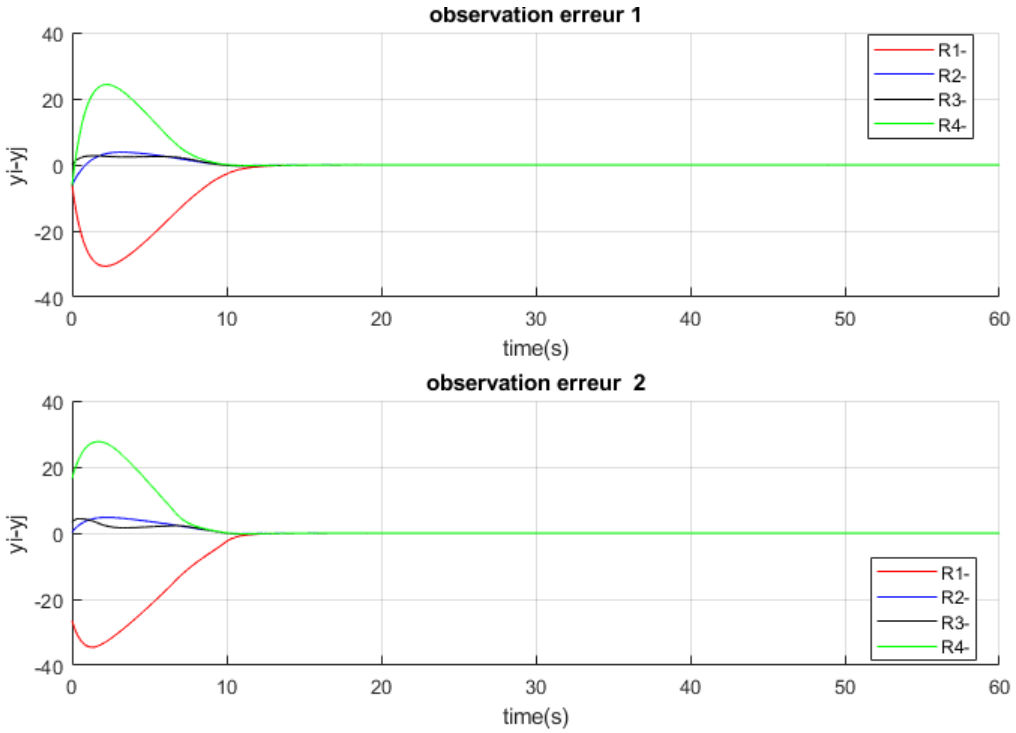


Figure 3.35 erreur d'observation

la figure (3.34) montre le vecteur d'état v_i qui exprime l'erreur d'observation $x_i - x_r$, par contre (3.35) c'est l'erreur d'observation des sorties donc

$$C.v_i{=}y_i - y_r$$

3.4.5 Interprétation des résultats

Nous avons introduit des protocoles de consensus linéaires pour un réseau de dynamique des agents. Le consensus atteint un point final en utilisant une information locale pour une commande de retour d'état tel que :

$$\begin{cases} u_i = k.v_i. & i = 1, ..., N \\ \lim\limits_{t \to \inf} |x_i - x_j| = 0 & i, j = 1, ..., N \end{cases} \tag{3.28}$$

Dans la $1^{ère}$ simulation chaque agent a un mode dynamique linéaire commun qui est contrôlable, le consensus peut être réalisé via des contrôles non distribués ou distribués utilisant le protocole (2.5). Si le consensus est atteint, tout les agents convergeront à leur une valeur de consensus final (3.20). L'un des problèmes les plus difficiles de cette approche est la connexion entre tous les agents. Il convient de noter aussi que la valeur finale du consensus dépend des valeurs initiales et de la dynamique des agents.
La $2^{éme}$ simulation un type observateur basé sur un contrôle par retour d'état est proposé au protocole de consensus :

$$\begin{cases} \dot{v_i} = (A + BK)v_i + F\left(c\sum\limits_{j=1}^{N} a_{ij}C((v_i - v_j) + C.d_i.(v_i - v_r) - \widehat{\gamma_i}\right) \\ y_i = C.v_i \end{cases} \tag{3.29}$$

$(c\sum_{j=1}^{N} a_{ij}C(v_i - v_j))$ exprime l'information partagée entre l'agent et ses voisins .

Notons que tous les protocoles utilisés dans notre cas sont distribués, car ils sont basés uniquement sur les informations relatives des agents voisins.
Donc sous une topologie de communication invariable dans le temps, chaque agent ayant une forme générale de dynamique linéaire. Un observateur avec un système de référence est ajouté afin de résoudre les problèmes pratiques où les états des agents se rapprochent asymptotiquement d'un état de référence, qui peut également varier dans

le temps (Notion de leader virtuel).
Les protocoles de consensus basé sur des mesures d'accords entre agents voisins ont été proposés et analysés.

3.4.6 Application d'algorithmes de Consensus sur les quadrirotors

Finalement, la structure complète de contrôle de la formation est simulée avec un contrôleur de référence actif. Nous avons utilisé une dynamique réelle d'un quadrotor. Les paramètres physiques du quadrotor utilisé sont disponibles en annexe (D).
Dans cette partie, nous utilisons la commande par mode glissant que nous avons développées dans la section précédente . Le consensus s'applique a la position des quadrotors (X,Y,Z), donc nous avons besoin d'utiliseé le modèle de translation (devient 6x6). Le vecteur de commande de modèle de translation d'un quadrotor est $u = [\theta, \phi, \psi, F]$. Ce vecteur serait la sortie de code consensus-contrôle, et en même temps il représente les angles désirées pour le modèle de l'attitude de quadrotor (Figure.2.2).

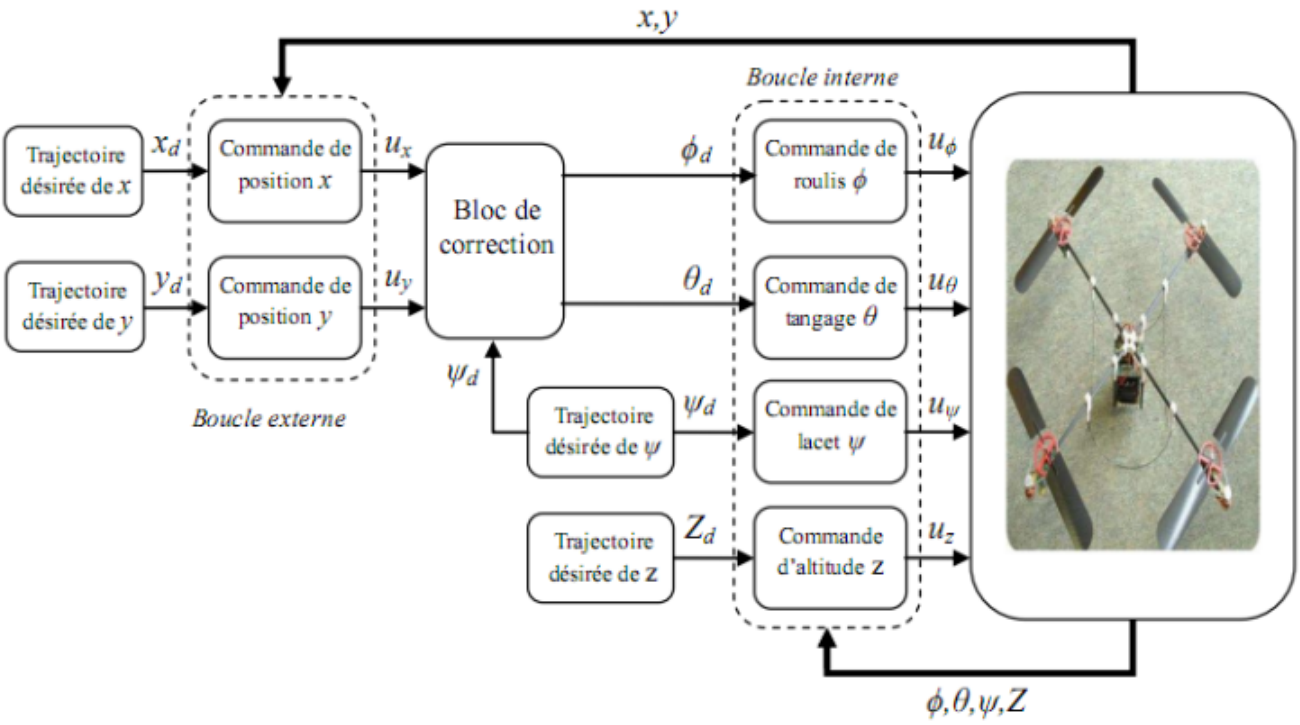


Figure 3.36 Schéma de commande

3.4.6.1 Résultats de simulation

Les figures ci-dessous expriment les résultats de l'application d'algorithme de consensus avec un modèle de référence sur une coopération de 3 quadrotors.

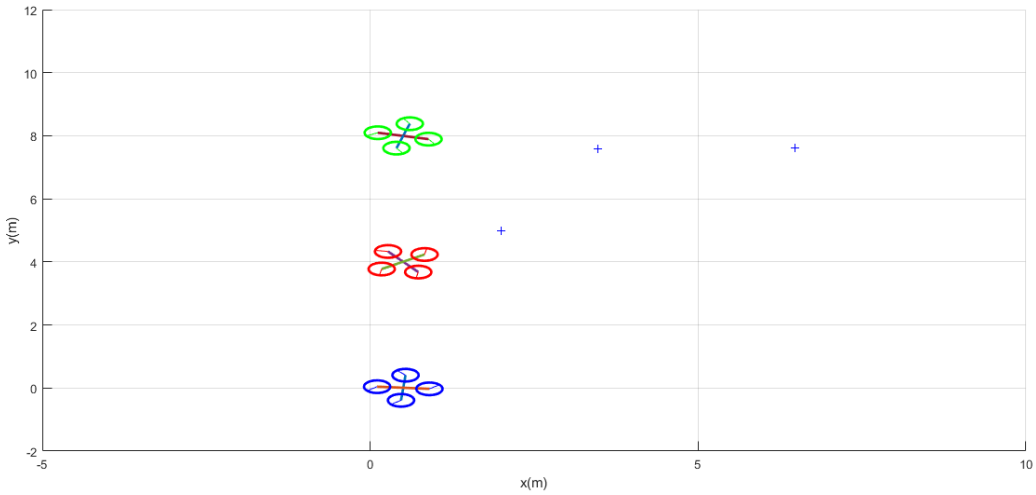
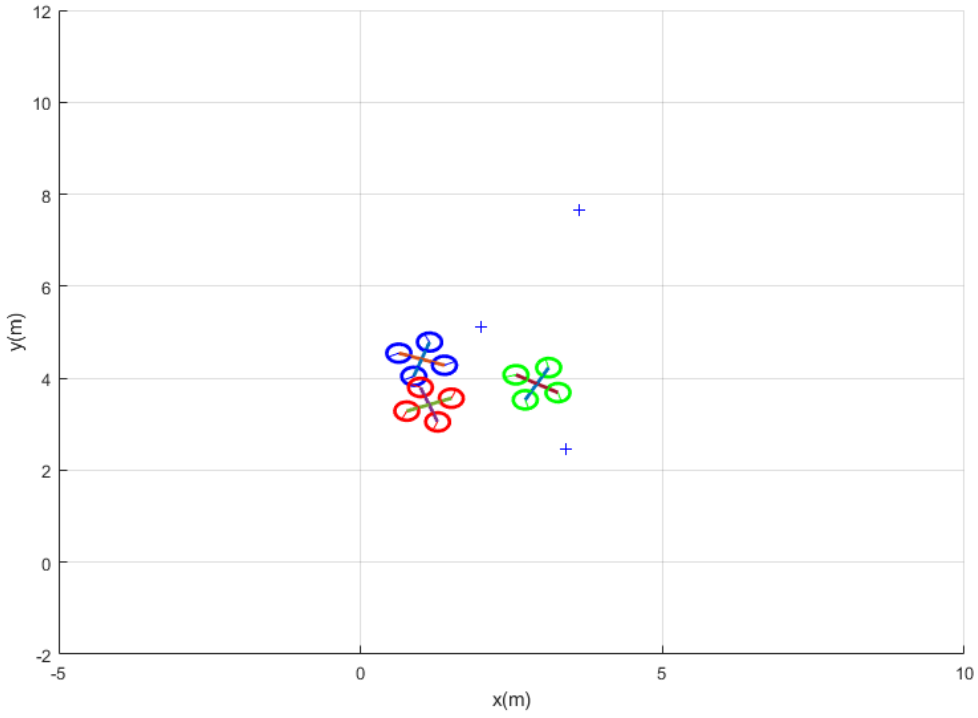


Figure 3.37 Schéma d'initiation de vecteur position des 3 quadrotors



[]

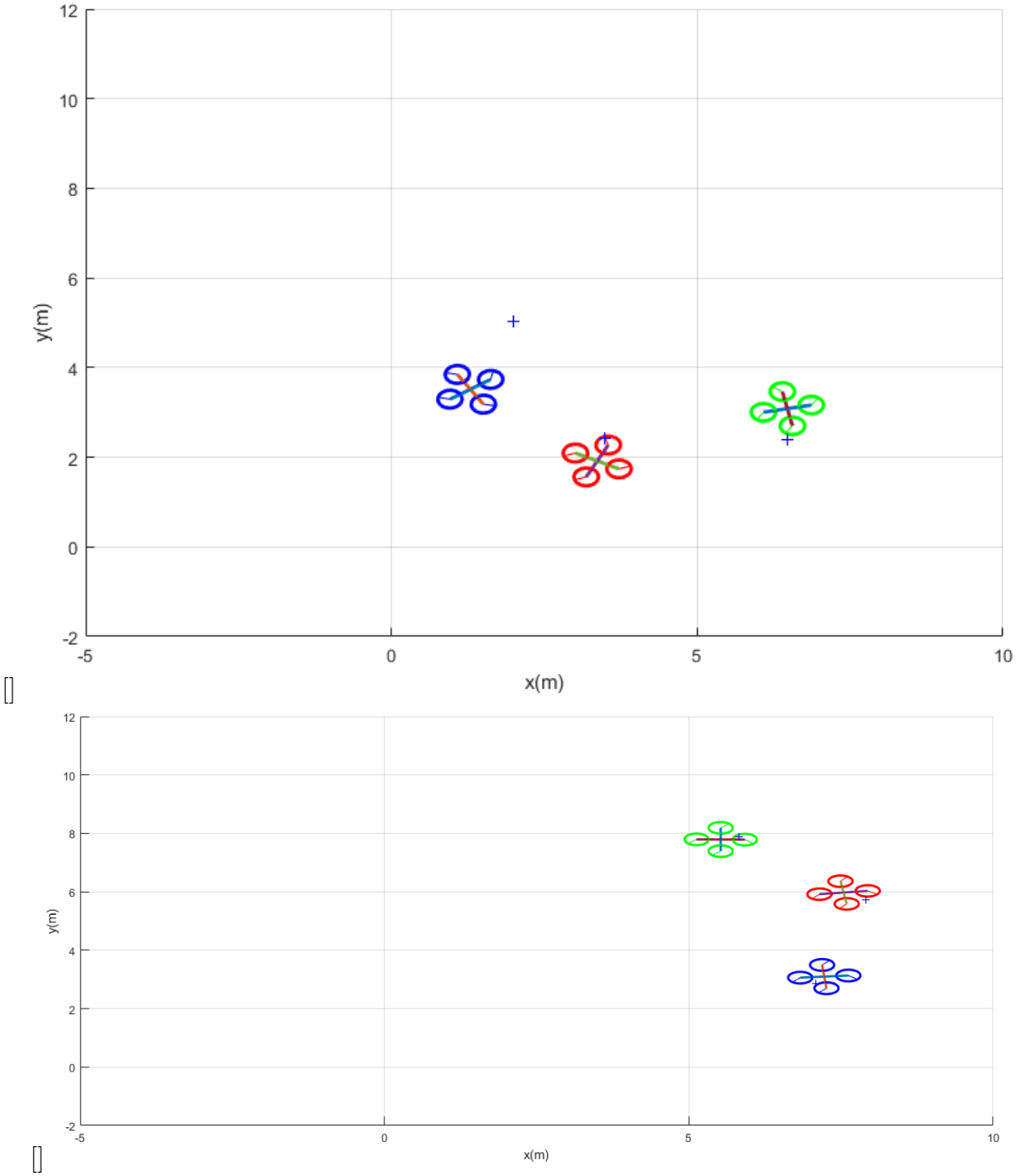


Figure 3.38 Evolution temporelle des robots afin d'atteindre la référence

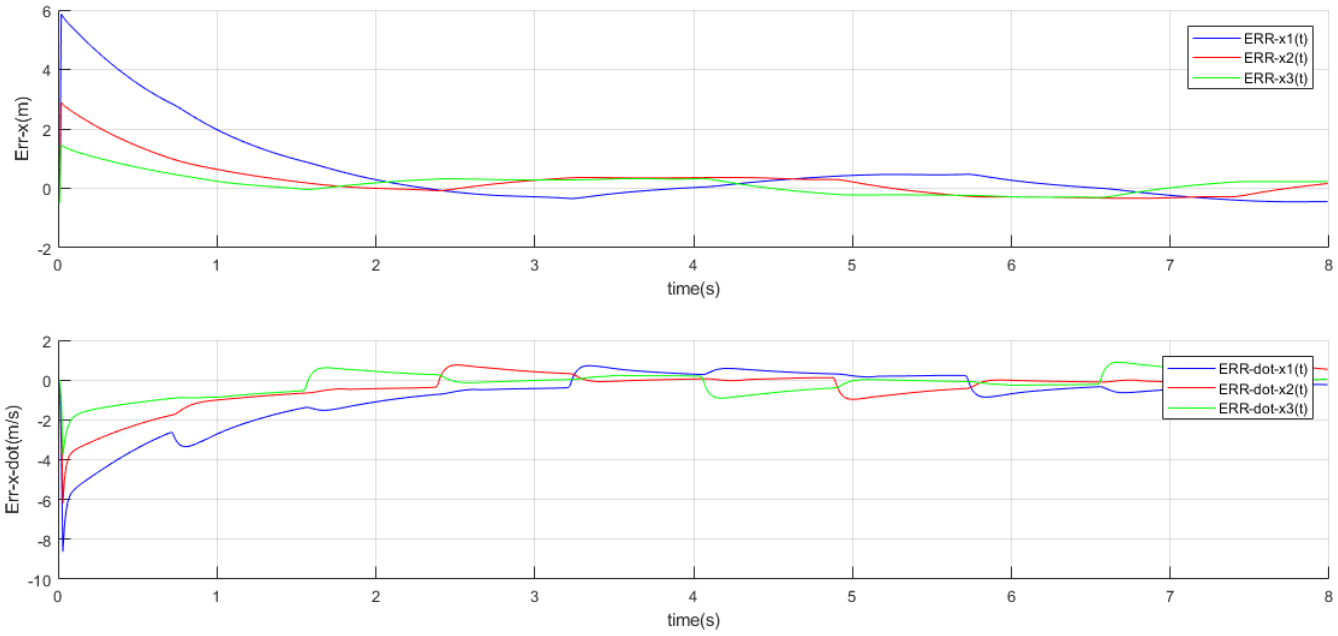


Figure 3.39 Erreur de covergence d'état (X)

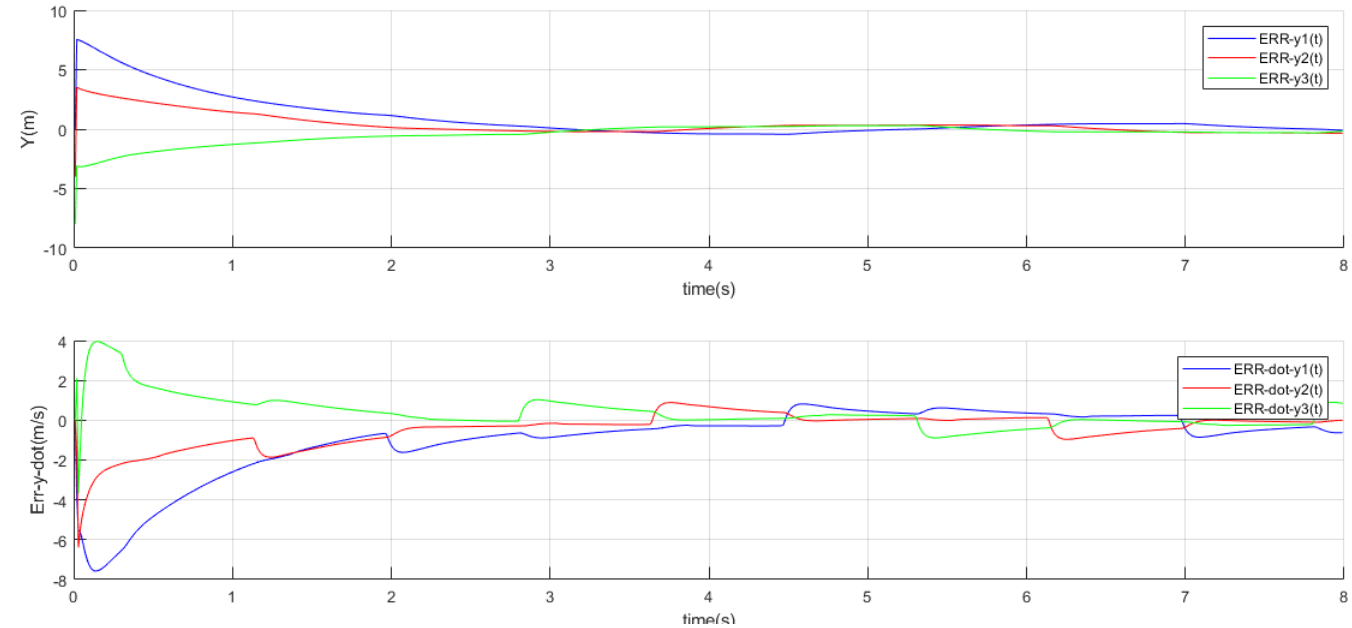


Figure 3.40 Erreur de covergence d'état (Y)

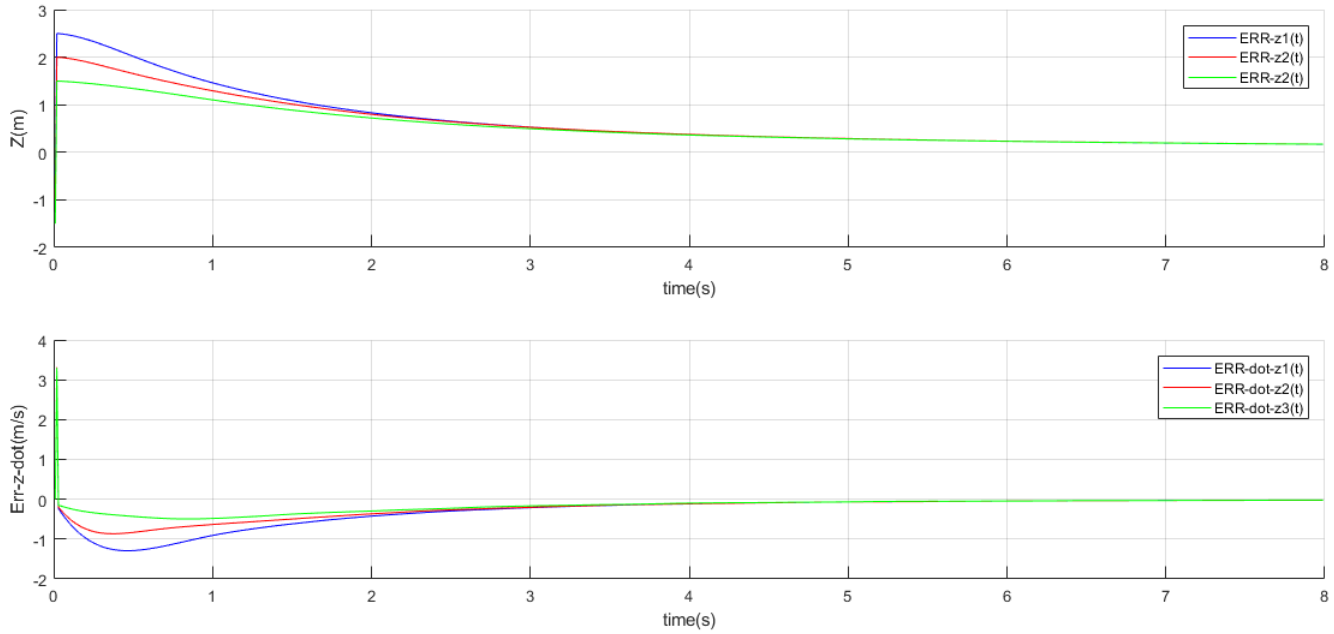


Figure 3.41 Erreur de covergence d'état (Z)

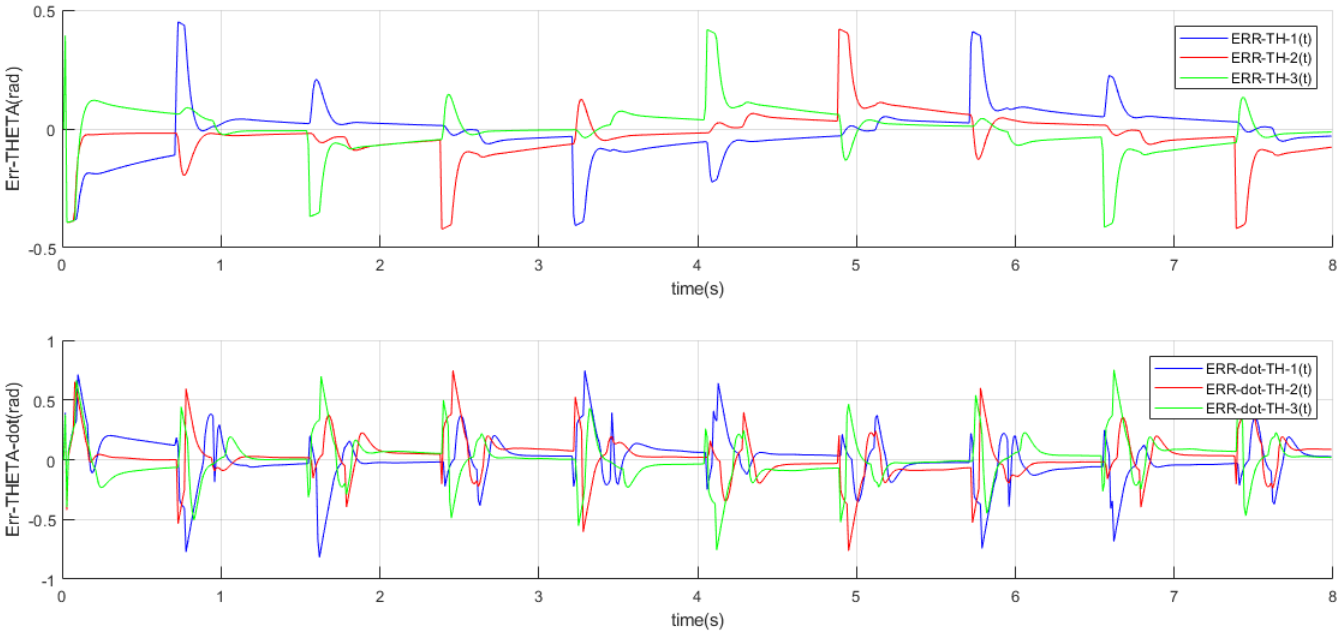


Figure 3.42 Erreur de covergence d'état (θ)

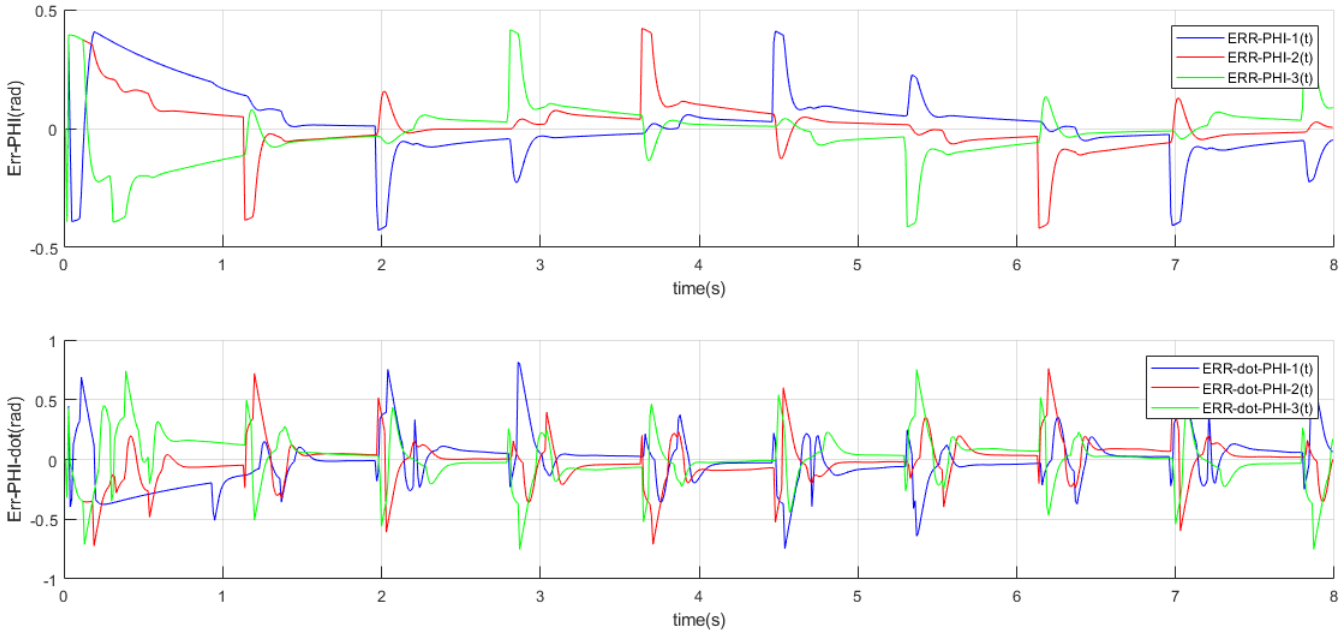


Figure 3.43 Erreur de covergence d'état (ϕ)

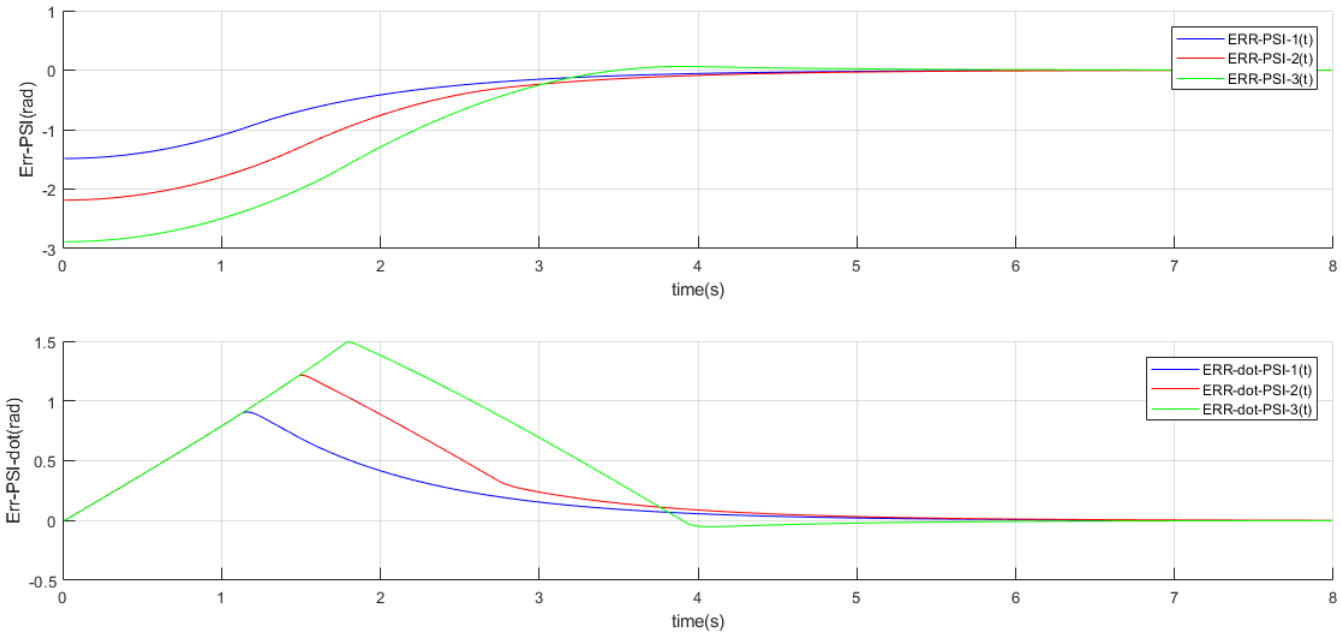


Figure 3.44 Erreur de covergence d'état (ψ)

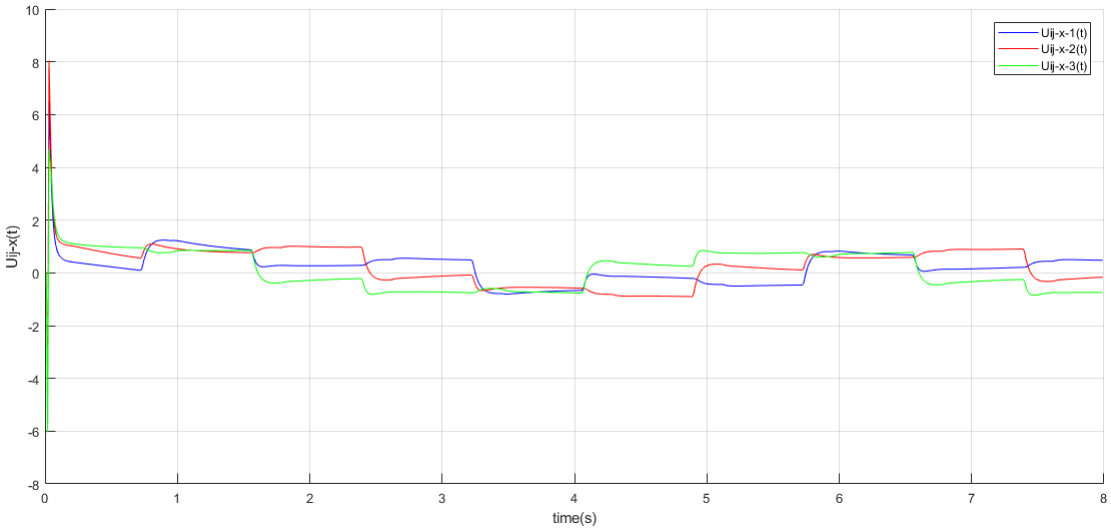


Figure 3.45 Les protocoles utiliseés sur l'etat (X)

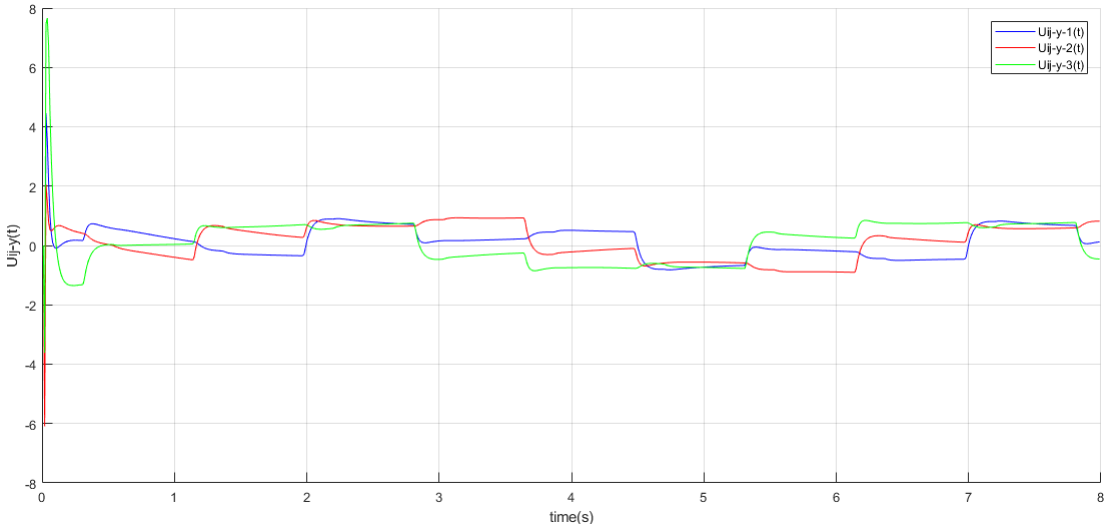


Figure 3.46 Les protocoles utiliseés sur l'etat (Y)

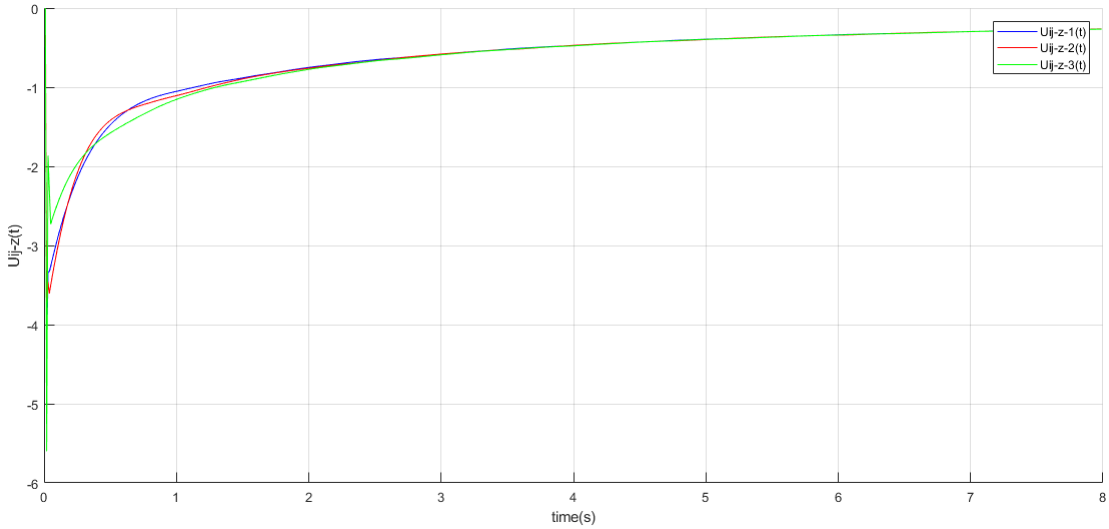


Figure 3.47 Les protocoles utiliseés sur l'etat (Z)

3.4.7 Interprétation des résultats

Cette simulation est une solution au problème de contrôle des formations des drones en 3D. Telque la dynamique de chaque véhicule est modiliséé par le systéme (3.30)

$$\begin{cases} \dot{r_i} = v_i \\ \dot{v_i} = u_i \end{cases} \tag{3.30}$$

$ri \in R^m$ et $vi \in R^m$ expriment la position et la vitesse d'un drone i, et $u_i \in R^m$ la commande definie par :

$$u_i = \ddot{r_i} - \beta(r_i - r_i^*) - \gamma\beta(v_i - \dot{r_i}) - \sum_{j=1}^{N} g_{ij}k_{ij}(r_i - r_i^*) + (r_j - r_j^*) - \sum_{j=1}^{N} g_{ij}k_{ij}\gamma(v_i - \dot{r_i^*}) + (v_j - \dot{r_j^*}) \tag{3.31}$$

Les systèmes multi-drones sont connectés avec une topologie qui peut être modélisée par un graphe. Chaque véhicule atteigne son emplacement souhaité éventuellement tout en préservant la forme souhaitée pendant la durée de transition. Il n'y a pas de leader clair dans l'équipe de formation, par conséquent la formation distribuée basée sur le consensus protocole qui ne requière que des informations sur des voisins locaux des UAV. La topologie de communication et le protocole de contrôle satisfont à la condition de stabilité, qui guidera la formation multi UAV à une convergence asymptotique à la vitesse souhaitée et la même forme de coopération souhaitée.

3.5 Simulation d'un quadrirotor en utilisant ROS

Ces dernières années, le traitement en temps réel s'est imposée comme une discipline importante des sciences et du génie informatiques.
ROS est l'abréviation de « Robot Operating System », comme son nom l'indique ROS est un système d'exploitation pour robot, il a été créé en 2007 sous le nom de «ligne d'interconnexion» par le Laboratoire d'intelligence artificielle de Stanford avec l'appui du projet de l'AI Robot Stanford« STAIR ». En 2008, le développement s'est poursuivi principalement au Willow Garage (créé et financé par les dirigeants de Google). Cet organisme est un institut de recherche en robotique/incubateur ou plus de vingt institutions collaborent dans le sens d'un modèle de développement fédéré et ouvert (aux grandes universités et aux grands industriels de la robotique).

De ce fait, l'outil ROS est classé open source afin d'accélérer les recherches et d'améliorer les robots.
Il existe plusieurs distributions qui ont été améliorées et dont les quatre dernières :

.

Tableau 3.1 les distributions du ROS

Distribution	Date
ROS Noetic Ninjemys	Mai, 2020
ROS Melodic Morenia	Mai 2018
ROS Lunar Loggerhead	Mai, 2017
ROS Kinetic Kame	Mai, 2016

Dans notre cas, nous avons travaillé avec la version *Melodic Morenia* de la distribution ROS sous un système d'exploitation Linux .(ubuntU 18.04).
Vous trouvez Une explication du concept et du système de fichiers du ROS (A)

3.5.1 Simulation d'une commande d'un seul quadrirotor

Comme nous l'avons mentionné précédemment l'outil ROS est classé open source dont il n'est pas nécessaire de refaire tout le travail (création d'environnement, modélisation des moteurs des drones ...). Il suffit juste de choisir un type de drone, ajouter des capteurs de mesure, utiliser un package et de tester la commande qui a été programmée en python. Nous avons validé nos commandes avec un drone IRIS. Le quadcopter IRIS est un tout-en-un autonome, élégant et facile à utiliser est parfait pour toute application d'imagerie aérienne.

Figure 3.48 IRIS quadrirotor

3.5.2 Fusion des données des capteurs

Sauvant dans la robotique les données des capteurs sont bruitées, ce qui crée la nécessité d'utilisation des estimateurs. L'estimation est le processus d'extraction d'informations sur des variables d'intérêt à partir des mesures bruitées des capteurs et qui peuvent être liées à ces variables par des modèles mathématiques complexes.
Les quadrirotors ont besoin de précision et un faible temps de latence dans leur estimation d'état (la position, vitesse) afin d'obtenir un vol stable et robuste. Donc l'obtention de la position et la vitesse d'un qudrirotor est l'un des problèmes majeurs que nous ne pouvons pas traiter avec la simulation théorique. Les unités de mesure inertielle (IMU), les caméras répondent à ces contraintes de puissance et ils sont disponibles dans le commerce depuis un certain temps et ils ont été utilisés pour la détection et la stabilisation dans de nombreuses applications.

Configuration classique d'IMU

(1) 3 axes Accéléromètres pour le calcul d'accélération

(2) 3 axes Gyromètres pour calcule les vitesses angulaires

(3) 3 axes Magnétomètres pour calculer la hauteur

(4) Barométres pour les mesures de pression

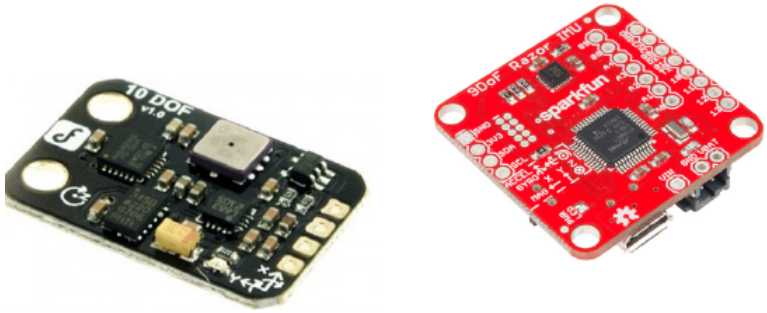


Figure 3.49 IMU

3.5.2.1 Résultat des Commandes

Le schéma (3.50) explique le déroulement du process entre ROS et gazebo

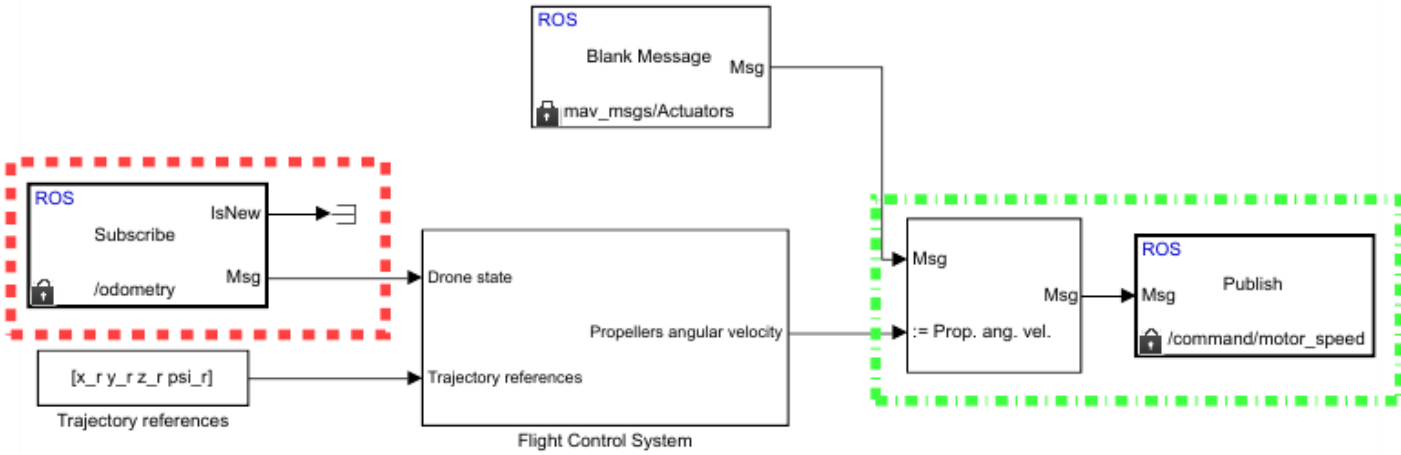


Figure 3.50 Un schéma descriptif de la compilation du programme entre ROS et gazebo

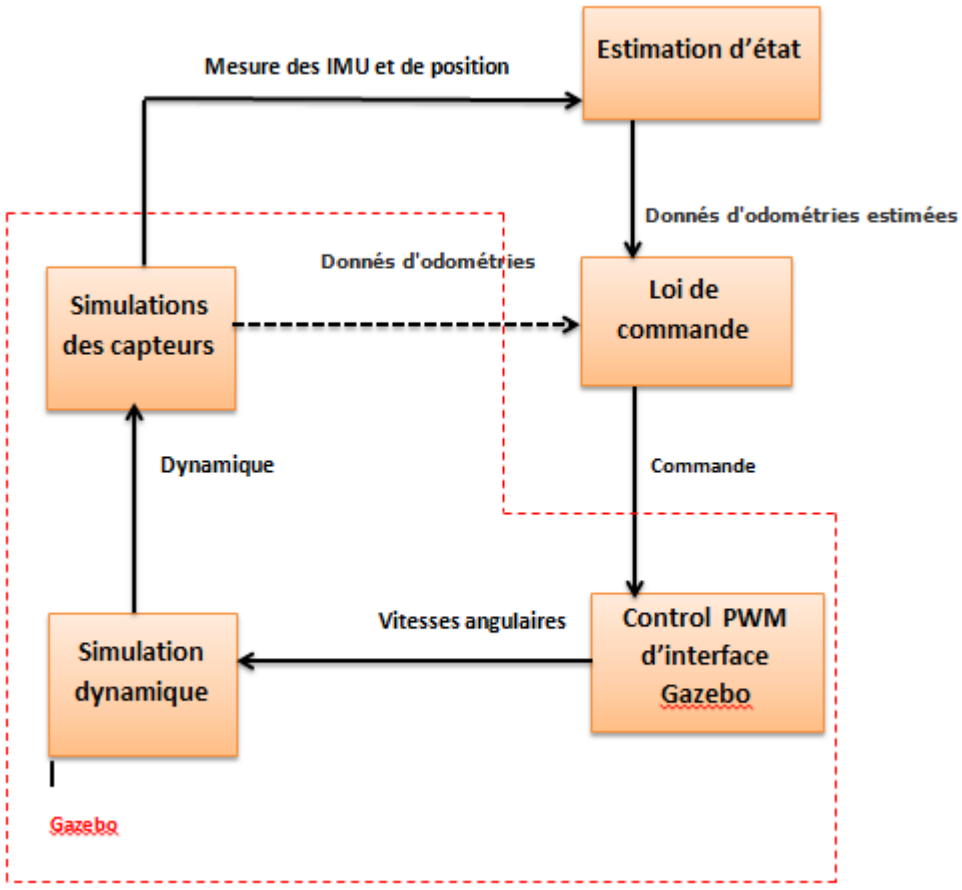


Figure 3.51 Graphe explique le passage des commandes au IRIS

La figure (3.52) exprime l'une des méthodes de modélisation graphique du ROS. Il fournit un plugin GUI pour visualiser le graphe de calcul ROS. Les node (ellipse) et les topics (carrés). Les flèches continues sont les topics actifs qui ent une direction allant du subscriber node au publisher node.

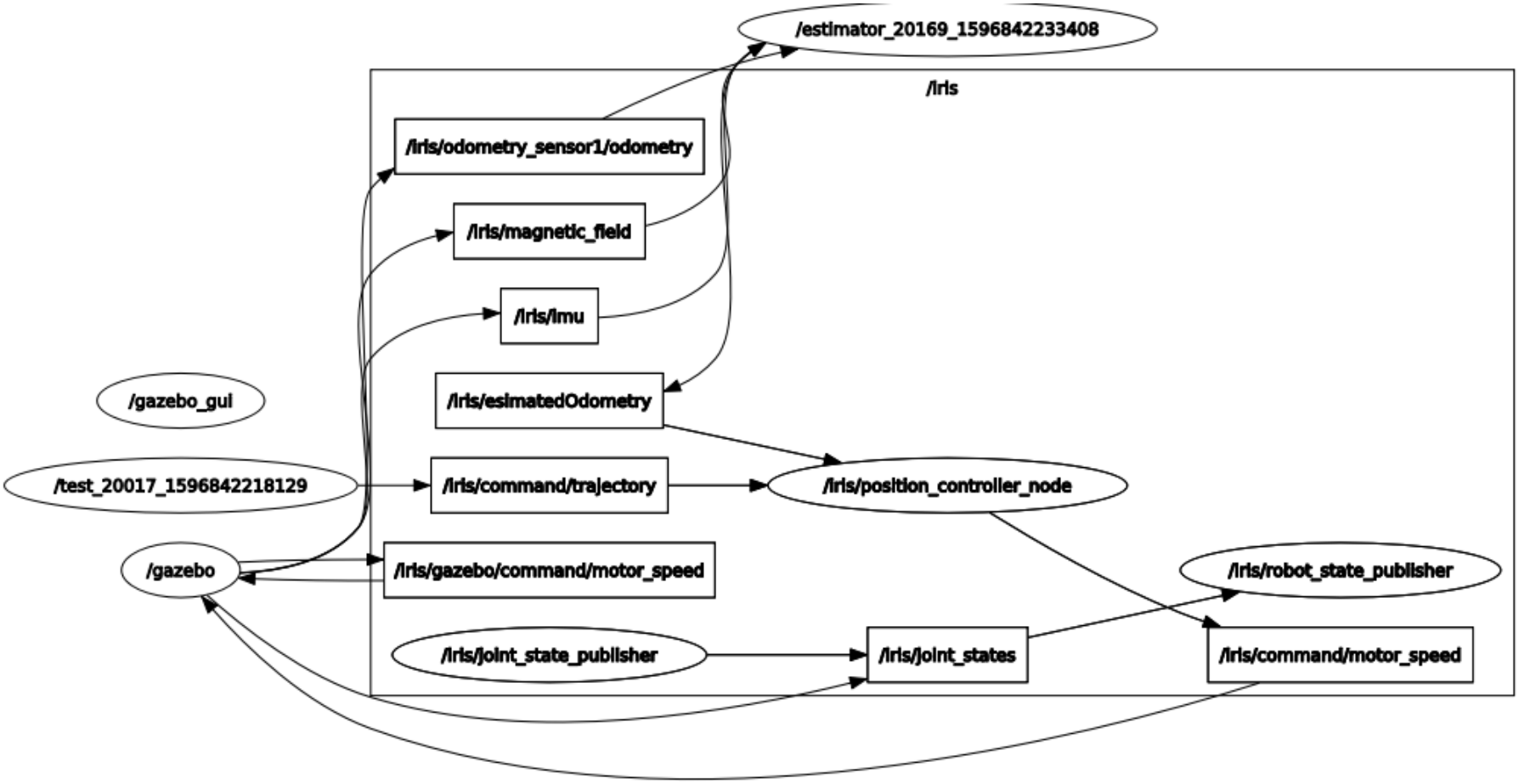


Figure 3.52 RQT graph de la commande d'IRIS

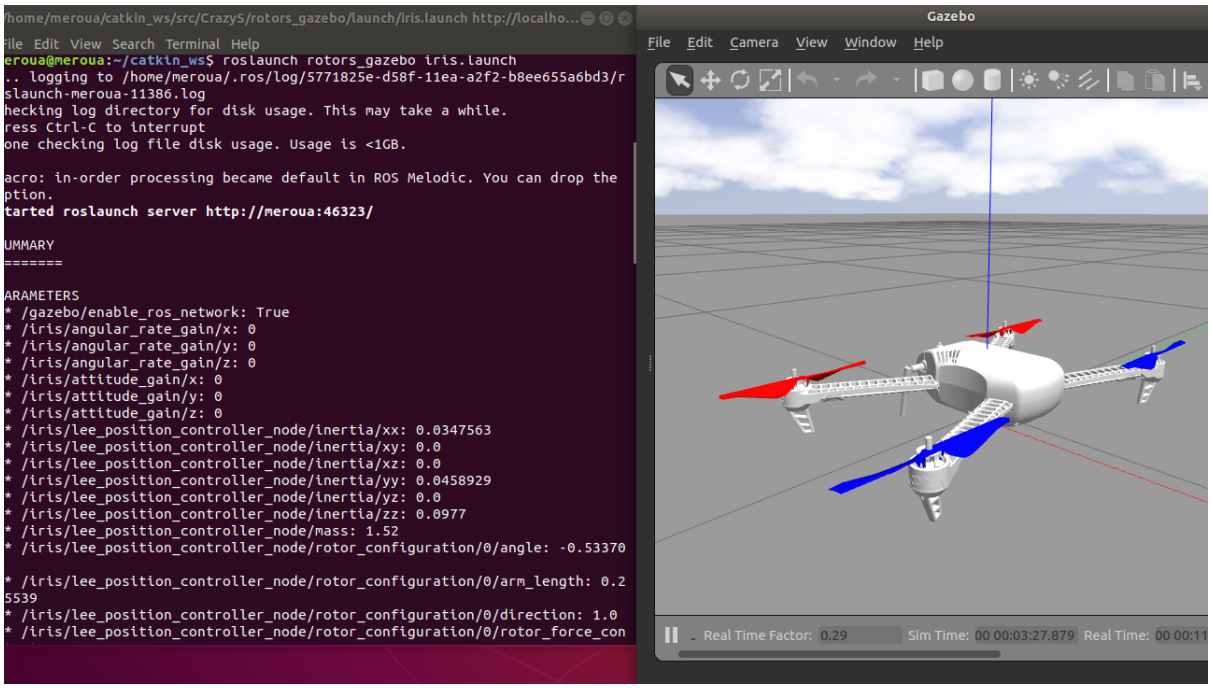


Figure 3.53 L'appel de quadrotor iris avec le ficher launch

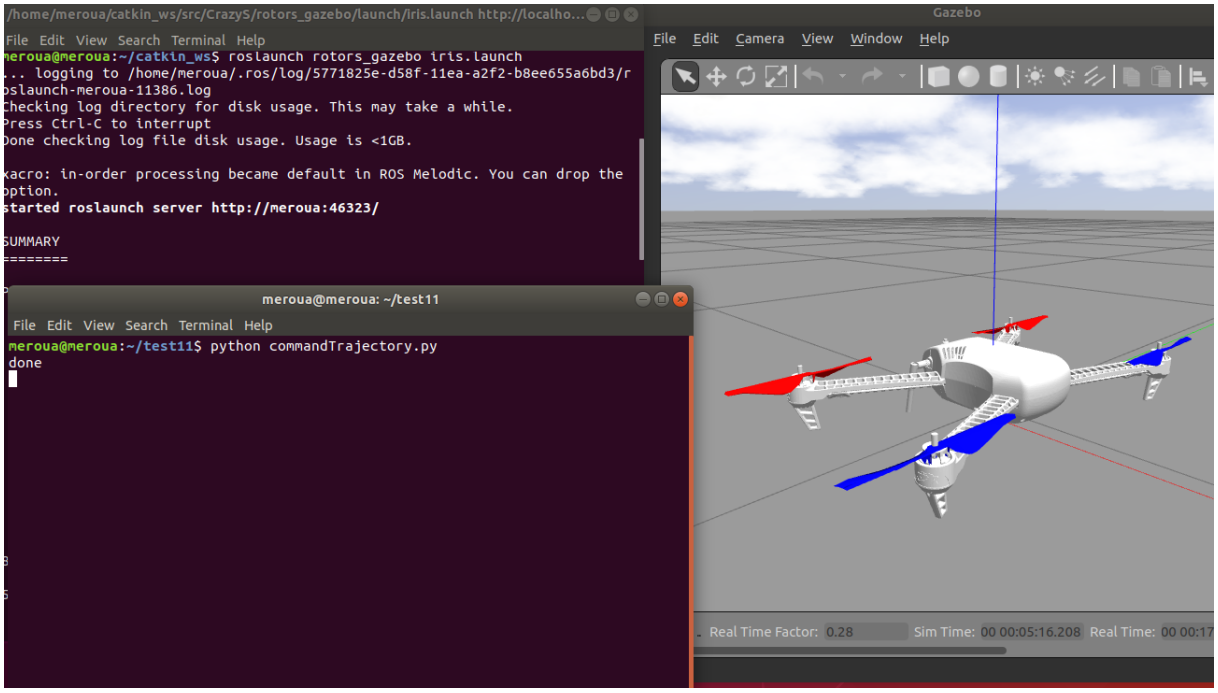


Figure 3.54 Appel de node de commande PID

3.5.3 Simulation d'une commande d'une coopération de quadrirotor

Nous proposons le contrôle de plusieurs quadrotors pour environnement intérieur

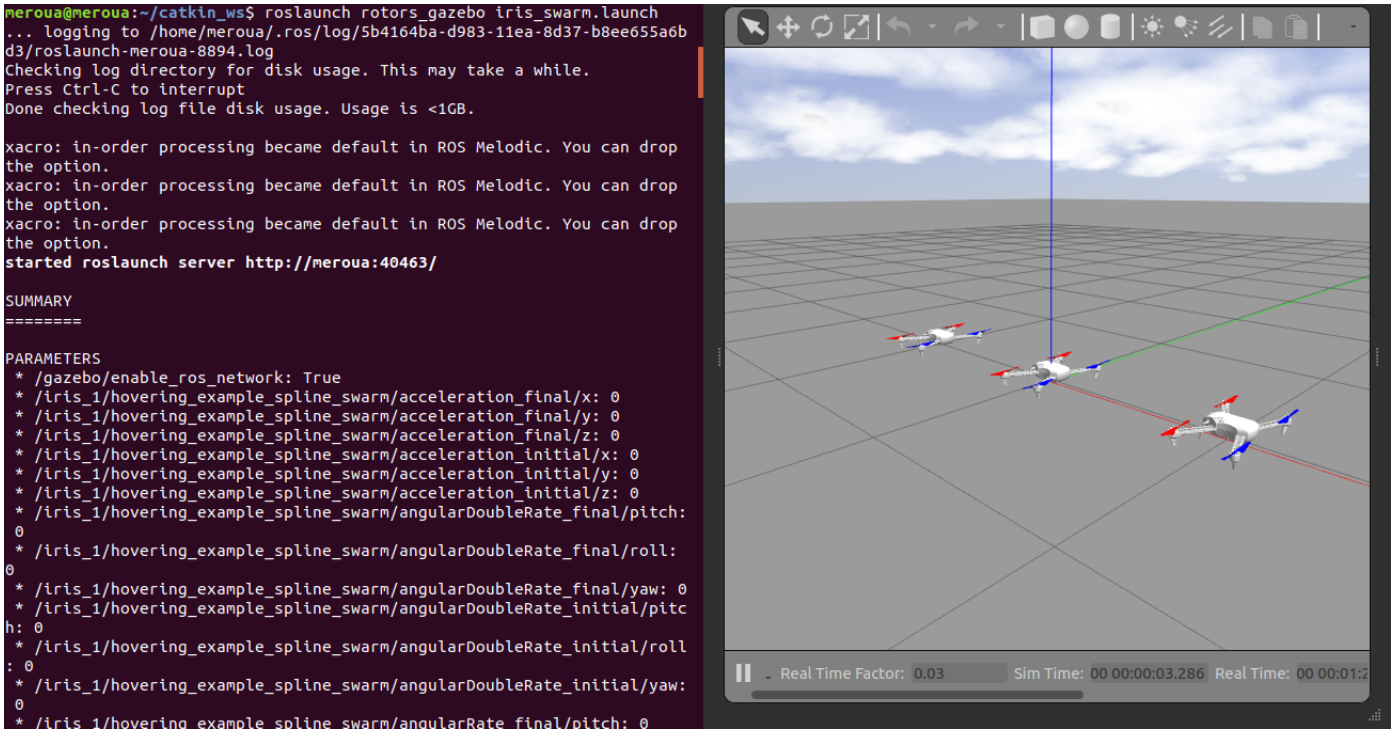


Figure 3.55 Initiation de position des 3 quadrotors

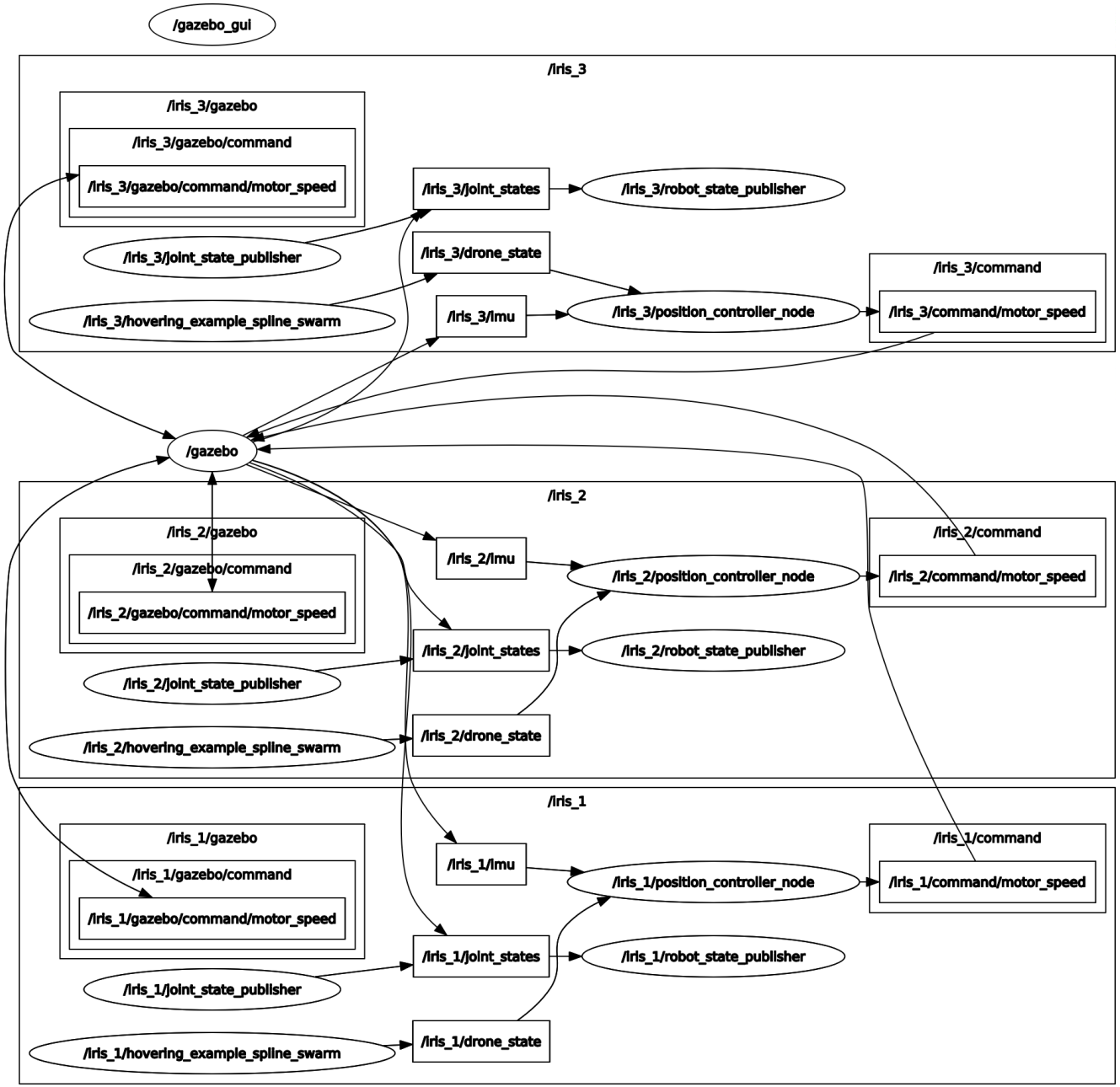


Figure 3.56 RQT graph de la commande d'une coopération de quadrotor IRIS

3.6 Conclusion

Dans cette section, nous avons implémenté les commandes synthétisées au chapitre 2 (pid / mode glissant) et nous avons interprété les résultats qu'elle offrent. Ensuite le contrôle de la formation est évalué par des simulations avec des scénarios spécifiques. Les résultats montrent une amélioration considérable de précision. Enfin nous avons présenté ROS, ses notions de bases. Puis, nous avons implémenté nos algorithmes sous ROS et Gazebo.

CONCLUSION

Au cours des années précédentes, des approches typiques pour le contrôle de la formation pourrait être grossièrement classé comme leader suiveur, comportemental, virtuel leader/structure virtuelle. La plupart des recherches de formation qui sont effectuées sur la base de l'approche leader/suiveur, où certains drones sont conçus comme des chefs de file tandis que d'autres sont conçus comme abonnés. Dans cette approche, les dirigeants suivent la trajectoire prédéfinie et les suiveurs suivre les leaders. C'est facile d'analyser et implémenter le contrôleur leader-suiveur. Cependant, le leader est un point unique pour la formation, et donc cette approche n'est pas robuste face à l'échec du leader.
Dans cet mémoire, un contrôle de formation est présenté avec des protocols qui utilise des mesures entre agents voisine pour influence la trajectoire de référence, ce qui conduit à une meilleure précision dans le maintien de la position. Pour valider l'efficasitéé de l'approche proposée, un modèle de quadrotor est introduit et son contrôle local associé. Les résultats de la simulation avec un groupe de ces véhicules montrent que la solution avec une trajectoire de référence contrôlée améliore considérablement la précision du maintien de la formation. La principale contribution de ce travail réside dans la formulation mathématique du problème qui permet l'optimisation de la trajectoire de référence en temps réel et utilise la rétroaction de la formation. Avec l'approche distribuée, la robustesse aux défaillances des agents est donnée par rapport à une solution centralisée. Seule une communication locale entre voisins est nécessaire, ce qui permet de modifier facilement le nombre de véhicules, la forme de la formation ou la structure de communication.

L'un des problèmes majore de cette solution c'est qu'elle nécessite des mesures exactes de la position du système par rapport à chaque véhicule individuel. Cela pourrait être donné dans un environnement de recherche intérieur fermé (indoor) et en utilisant des capteurs assez puissants afin de garantit la précision des mesures.
Cependant, les situations extérieures réelles posent des défis supplémentaires. En outre, le réseau de communication est supposé être sans retards, sans perte de données et illimité dans le temps. Pour un grand nombre de véhicules, ces hypothèses peuvent ne pas être raisonnables. Aussi un algorithme d'évitement d'obstacle n'est pas assuré à cause de limitation de temps de recherche donné.

Les futures recherches pourraient examiner l'adaptation du cadre de contrôle de la formation pour les scénarios outdoor. En ajoutant des algorithmes d'évitement d'obstacle dans des environnements dynamiques et statiques connus et inconnus.

RÉFÉRENCES

[Adventures 2012] Adventures, Technical : *QuadCopter Stabilization Control System "X Plus Configuration.* 2012. – URL `http://technicaladventure.blogspot.com/2012/09/quadcopter-stabilization-control-system.html`

[Beineke 2004] Beineke, Wilson Robin J. Cameron Peter J. : Topics in Algebraic Graph Theory. In : *Topics in Algebraic Graph Theory*, 2004, S. 1–3

[Bouabdallah Samir 2007] Bouabdallah Samir, Roland : Full control of a quadrotor. In : *IEEE/RSJ International Conference on Intelligent Robots and systems (IROS), 2007.* IEEE (Veranst.), 2007

[Castillo 2004] Castillo, A : Real-time stabilization and tracking of a four-rotor mini rotorcraft. In : *IEEE Trans. Control Systems Technology* (2004), Nr. 4, S. 510–516

[De Dinechin und Melquiond 2015] De Dinechin, Lauter C. ; Melquiond, Guillaume : Mixed reality for robotics,IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). In : *IEEE* (2015)

[Jinhuan Wang und Hu 2008] Jinhuan Wang, Daizhan C. ; Hu, Xiaoming : Consensus of multi-agent linear dynamic systems. In : *Wiley InterScience* 10 (2008), Nr. 1-4, S. 114–155. – ISSN 1934-6093

[Khalil 2011] Khalil, Hassan K. : Nonlinear Systems. In : *Nonlinear Systems.* Upper Saddle River, N.J, 2011, S. 565 – 569

[Makarov 2019] Makarov, M. : Sensor fusion and state estimation, 2019, S. 16

[M.GUETTACHE 2019] M.GUETTACHE, Y.BENAMEUR : *Perception et Navigation visuelle d'un quadrirotor en utilisant des techniques d'apprentissage profond*, École Polytechnique d'Alger, Dissertation, 2019

[M.HAZERCHI 2012] M.HAZERCHI, M.TAZIR : *Planification de la trajectoire d'un robot mobile autonome dans un environnement statique et/ou dynamique* , École Polytechnique d'Alger, Dissertation, Juni 2012

[M.Mokhtari 2015] M.Mokhtari : *Observation et Commande de Drones Miniatures à voilures tournantes*, Université Aboubekr Belkaid Tlemcen, Dissertation, 2015

[Murry 2007] Murry, R.M : Recent Research in Cooperative Control of Multivehicle Systems. Journal of Dynamic Systems. In : *Journal of Dynamic Systems, Measurement, and Control* (2007), Nr. 2, S. 571–583

[O.MEKKID 2016] O.MEKKID : *Analyse de Robustesse des loisde la commande PID et Mode Glissant pour la commande d'attitude d'un Quadrirotor*, École Polytechnique d'Alger, Dissertation, Juni 2016

[Petit 2016] Petit, Jens : *Distributed Consensus-based Formation Control of Quadrotors with Formation Feedback*, University of stuttgart,Keio University Namerikawa Laboratory, Dissertation, 2016

[Ren 2007a] Ren, Atkins E. : Distributed multi-vehicle coordinated control via local information exchange. In : *Wireless Networks* 32 (2007), März, Nr. 2, S. 4

[Ren 2007b] Ren, W. : Consensus strategies for cooperative control of vehicle formations. In : *International Journal of Neural Systems* 1 (2007), Nr. 2, S. 505–512

[Tayebi und McGilvra 2006] Tayebi, A. ; McGilvra, S. : Attitude stabilization of a VTOL quadrotor aircraft. In : *IEEE Trans. Control Systems Technology* (2006), Nr. 3, S. 562–571

[Xu Rong 2006] Xu Rong, Ozguner U. : Sliding mode control of a quadrotor helicopter. In : *Decision and Control, 2006 45th IEEE Conference on* IEEE (Veranst.), 2006, S. 4957–4962

[Zhang u. a. 2005] Zhang, L. J. ; Pierre, S. ; Marchand, L. : Optimization of Handover Performance for FMIPv6. In : *Intelligence in Communicaon Systems*, 2005, S. 169–178

[Zhongkui Li 2010] Zhongkui Li, Guanrong C. : Consensus of Multiagent Systems and Synchronization of Complex Networks :. In : *IEEE Journals Magazine* 57 (2010), Nr. 1-4, S. 213–224. – ISSN 1558-0806

ANNEXE A Robot Operating System/ROS

Comme son nom l'indique, ROS (Robot Operating System) est un système d'exploitation pour robots. De même que les systèmes d'exploitation pour PC, serveurs ou appareils autonomes, ROS est un système d'exploitation complet pour la robotique de service.

ROS en quelques mots Le principe de base d'un OS robotique est de faire fonctionner en parallèle un grand nombre d'exécutables qui doivent pouvoir échanger de l'information de manière synchrone ou asynchrone. Par exemple, un OS robotique doit interroger à une fréquence définie les capteurs du robot (capteur de distance à ultrasons ou infrarouge, capteur de pression, capteur de température, gyroscope, accéléromètre, caméras, microphones...), récupérer ces informations, les traiter (faire ce que l'on appelle la fusion de données), les passer à des algorithmes de traitement (traitement de la parole, vision artificielle, localisation et cartographie simultanée...) et enfin contrôler les moteurs en retour. Tout ce processus s'effectue en continu et en parallèle. D'autre part, l'OS robotique doit assurer la gestion de la concurrence afin d'assurer l'accès efficace aux ressources du robot. Nous décrivons ci-dessous les concepts regroupés dans ROS sous le nom de « ROS Computation Graph » et qui permettent d'atteindre ces objectifs.

— Les paquets :
  sont l'unité principale pour l'organisation de logiciels,ils sont l'élément le plus atomique dans ROS.un paquet peut contenir des processus d'exécution ROS (noeuds - nodes), une bibliothéque dépendant de ROS, des données, des fichiers de configuration, ou tout autre element organise demaniere utile
— Traitement des paquets :
  (1) Le Master :
    Le Master est un service de déclaration et d'enregistrement des noeuds qui permet ainsi à des noeuds de se connaître et d'échanger de l'information.
  (2) Les nœuds :
    Les noeuds (nodes) sont des processus qui effectuent le traitement. par exemple, un noeud controle le capteur laser, un autre nœud controle les moteurs des roues, un nœud eectue la localisation, un nœud effectue la planification du chemin, un nœud fournit une vue graphique du systéme, etc. Un nœud ROS est écrit avec l'utilisation d'une bibliothéque client ROS, comme roscpp ou rospy.

(3) Les topics :
L'échange de l'information s'effectue soit de manière asynchrone via un topic ou de manière synchrone via un service. Un topic est un système de transport de l'information basé sur le système de l'abonnement / publication (subscribe / publish). Un ou plusieurs nuds pourront publier de l'information sur un topic et un ou plusieurs nœuds pourront lire l'information sur ce topic. Le topic est en quelque sorte un bus d'information asynchrone un peu comme un flux RSS. Cette notion de bus many-to-many asynchrone est essentielle dans le cas d'un système distribué. Le topic est typé, c'est-à-dire que le type d'information qui est publiée (le message) est toujours structuré de la même manière. Les nœuds envoient ou reçoivent des messages sur des topics.

(4) Les services :
Le topic est un mode de communication asynchrone permettant une communication many-to- many. Le service en revanche répond à une autre nécessité, celle d'une communication synchrone entre deux nœuds . Cette notion se rapproche de la notion d'appel de procédure distante (remote procedure call).

(5) Les messages :
Un message est une structure de donnée composite. Un message est composé d'une combinaison de types primitifs (chaînes de caractères, booléens, entiers, flottants...) et de message (le message est une structure récursive). Par exemple un noeud représentant un servomoteur du robot, publiera certainement son état sur un topic (selon ce que vous aurez programmé) avec un message contenant par exemple un entier représentant la position du moteur, un flottant représentant sa température, un autre flottant représentant sa vitess .

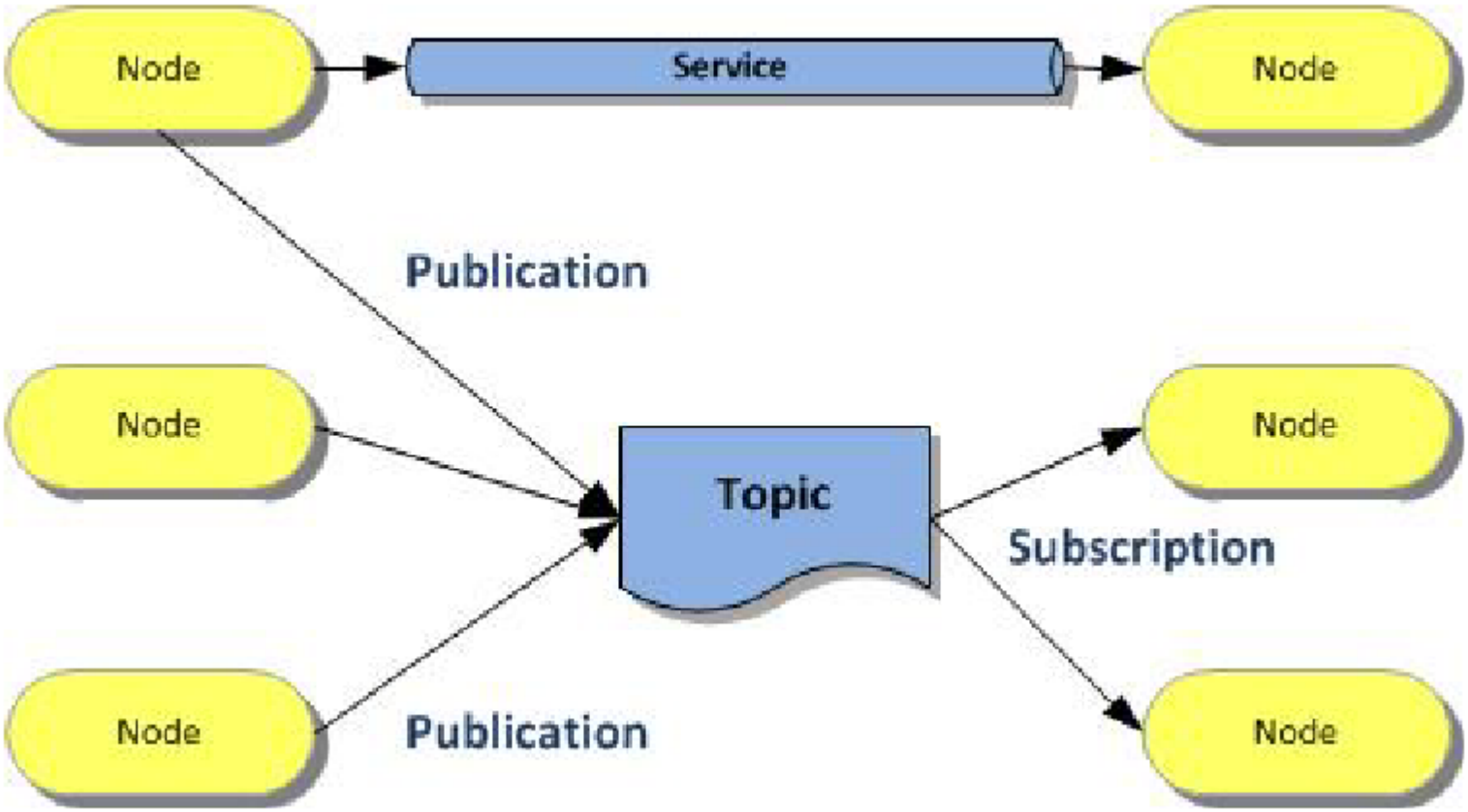


Figure A.1 schéma descriptif de traitement des paquets

ANNEXE B Gazebo

Pourquoi Gazebo ? La simulation de robot est un outil essentiel dans la boîte à outils de chaque roboticien. Un simulateur bien conçu permet de tester rapidement des algorithmes, de concevoir des robots, d'effectuer des tests de régression et de former un système d'IA à l'aide de scénarios réalistes. Gazebo offre la possibilité de simuler avec précision et efficacité des populations de robots dans des environnements intérieurs et extérieurs complexes.
À portée de main est un moteur physique robuste, des graphiques de haute qualité et des interfaces programmatiques et graphiques pratiques. Le meilleur de tous, Gazebo est gratuit avec une communauté dynamique.

Caractéristiques Gazebo est un logiciel libre financé en partie par Willow Garage et peut être reconfiguré, développé et modifié. Il est compatible avec ROS et Player. On peut exécuter Gazebo à partir de ROS et utiliser les API de ces dernières pour contrôler les robots dans les simulations, c'est-à-dire envoyer et recevoir des données de ceux-ci. Ce logiciel permet de faire des simulations réalistes de la physique des corps rigides. Les robots peuvent interagir avec le monde (ils peuvent ramasser et pousser des objets, rouler et glisser sur le sol) et inversement (ils sont affectés par la gravité et peuvent se heurter à des obstacles dans le monde). Pour ce faire, Gazebo utilise de multiples moteurs physiques tels que Open Dynamics Engine (ODE) ou Bullet.

Premiers pas avec Gazebo et ROS Vous devez avoir préalablement installé Gazebo et ROS. Vous êtes maintenant prêt à découvrir le monde de la simulation.

ANNEXE C Notion sur la stabilité au sens de lyapunov

([Khalil, 2011]),([Makarov, 2019])

Lyapunov stability

$\forall \varepsilon > 0, \forall t_0, \exists \delta(\varepsilon, t_0)\ s.t.:$

$\|\boldsymbol{x}(t_0) - \boldsymbol{x}_0\| < \delta(\varepsilon, t_0)$

$\Rightarrow \|\boldsymbol{x}(t) - \boldsymbol{x}_0\| < \varepsilon, \forall t \geq t_0$

Asymptotic stability

i. Lyapunov stability

ii. $\lim_{t\to\infty} \|\boldsymbol{x}(t) - \boldsymbol{x}_0\| = 0$

Exponential stability

i. Asymptotic stability

ii. $\exists M, \alpha > 0\ s.t.:$

$\|\boldsymbol{x}(t) - \boldsymbol{x}_0\| \leq M\|\boldsymbol{x}(t_0) - \boldsymbol{x}_0\|e^{-\alpha(t-t_0)}$

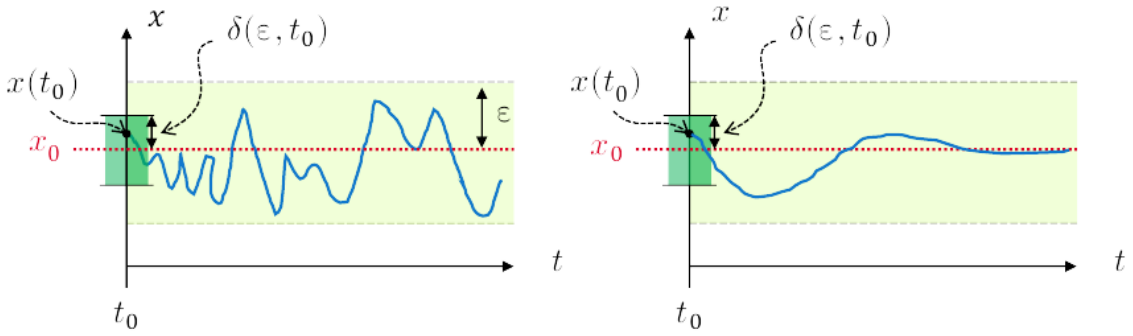


Figure C.1 Stabilité au sens de lyapunov

ANNEXE D Paramètres utilisés

.

Tableau D.1 Tableaux des paramètres physique

Paramètre	Symbole	Valeur	Unité
Masse du quadrirotor	m	0.468	kg
Distance entre le centre d'un moteur et le centre de gravité	l	0.225	m
Moment d'inertie du quadrirotor par rapport à son axe 'X'	J_x	4.856*10^{-}3	kg.m^2
Moment d'inertie du quadrirotor par rapport à son axe 'Y'	J_y	4.856*10^{-}3	kg.m^2
Moment d'inertie du quadrirotor par rapport à son axe 'Z'	J_z	8.801*10^{-}3	kg.m^2
Moment d'inertie du rotor par rapport à son axe 'Z'	J_r	0.00006	kg.m^2
Coefficient de portance	b	1.14*10^{-}7	N.s^2
Coefficient de trainée	k	2.98*10^{-}6	N.m.s^2
Constante de gravité	g	9.81	m.s^{-2}

.

Tableau D.2 Tableaux des paramètres surface de glissment

Constant de reglage	Valeur
λ_1	1
λ_2	1
λ_3	1
λ_4	5
λ_5	5
λ_6	5
λ_7	20
λ_8	20
λ_9	20

Les paramètres physique utilisés pour les simulations du quadrirotor sont résumés dans le tableau suivant :

Les gains de la fonction sign(.) pour la commande par mode glissant synthétisé au premier chapitre ainsi que le paramètre de la fonction pseudosigne(.) sont résumés dans le tableau suivant :

.

Tableau D.3 Tableaux des gains de la fonction sign(.)

Constant	Valeur
A_1	0.08
A_2	0.5
A_3	10
A_4	0.6
A_5	0.1
A_6	0.5
A_7	18
A_8	18
A_9	18

.

Tableau D.4 Tableaux des gains de réglage pour la commande PD

Constant	Valeur
$k_{p\phi}$	1
$K_{p\theta}$	1.5
$k_{p\psi}$	1.8
k_{pz}	120
$k_{d\phi}$	0.5
$k_{d\theta}$	0.5
$k_{d\psi}$	0.5
k_{dz}	25