Diagnostic des systèmes embarqués en réseau. Application à un mini drone hélicoptère

Cédric Berbra

INSTITUT POLYTECHNIQUE DE GRENOBLE

No. attribué par la bibliothèque

THESE

pour obtenir le grade de

DOCTEUR DE L'Institut polytechnique de Grenoble

***Spécialité :* AUTOMATIQUE**

préparée au laboratoire GIPSA-lab, département Automatique

dans le cadre de **l'École Doctorale Électronique, Électrotechnique, Automatique, Traitement du Signal**

présentée et soutenue publiquement

par

Cédric BERBRA

le 12 Novembre 2009

Titre :

Diagnostic des systèmes embarqués en réseau. Application à mini drone hélicoptère

Directeurs de thèse :

Mme Sylviane GENTIL

Mme Suzanne LESECQ

JURY :

Mr.Jean Marc THIRIET	Président
Mme Françoise SIMONOT-LION	Rapporteur
Mr. Dominique SAUTER	Rapporteur
Mme Sylviane GENTIL	Directeur de thèse
Mme Suzanne LESECQ	Co-encadrant
Mr. David HENRY	Examinateur
Mr. Daniel SIMON	Invité

A ma femme Meriem,
à ma famille,
et à Laurine.

Remerciements

Je tiens à remercier chaleureusement toutes les personnes qui m'ont apporté leur aide tout au long de ces trois années. Un sincère et chaleureux remerciement pour madame Sylviane Gentil, Professeur à l'Institut polytechnique de Grenoble ainsi que madame Suzanne Lesecq, Ingénieure de recherche au CEA-LETI de Grenoble, pour la qualité de leur encadrement, leur disponibilité mais aussi la confiance qu'elles m'ont témoigné durant ces 3 années.

Je remercie également Madame Françoise Simonot-Lion ainsi que Monsieur Dominique Sauter pour avoir accepté de rapporter mon travail. Messieurs Davaid Henry, Jean Marc Thiriet et Daniel Simon je vous remercie pour avoir examiné mon travail et pour votre participation au sein de mon jury.

Je désire remercier les partenaires du projet SafeNecs pour leurs collaborations et pour m'avoir fait partagé leurs compétences.

Mes remerciements les plus sincères à Marie-Thérèse, Marielle, Virginie, Marie-Rose et Patricia pour leur gentillesse ainsi qu'à toute l'équipe technique du laboratoire.

Un merci à monsieur Thierry Creuzet pour sa collaboration à la mise en oeuvre du quadrotor.

Je remercie Alma Marouane, Mechraoui Amine, Derbel Haithem, Guerrero Josefermi, Khan Zeeshan, Do Trong Hieu et Nguyen Hoang Van pour l'ambiance au sein du laboratoire.

Un grand merci à mon épouse Meriem ainsi qu'à ma famille et à ma petite soeur Laurine.

Table des matières

Table des figures

Chapitre 1

Introduction

Les systèmes commandés en réseau sont des systèmes où le réseau est utilisé comme moyen de communication dans la boucle de commande. Ce type de système appelé "*Networked Control Systems*" (*NCS*) présente plusieurs avantages. Ceci permet de réduire le coût en poids et en nombre de fils. Cela permet également une plus grande flexibilité, de rendre la maintenance plus facile. Cependant, les performances de commande ou de supervision de ces systèmes sont directement liées à celles du réseau. En effet, celui-ci introduit entre autres inconvénients majeurs des retards dus à la transmission des informations mais aussi à la perte potentielle d'informations. Ces paramètres vont directement influencer voire dégrader les performances du système.

Depuis plusieurs années, ces problèmes ont attiré l'attention de la communauté d'Automatique. Plusieurs travaux ont été menés sur le thème de la commande des systèmes en réseau. Avec ces systèmes, il est important de considérer le réseau comme une partie du système et non pas comme un simple moyen de communication. Pour cela, deux problématiques doivent être considérées. La première consiste à adapter les lois de commande et de diagnostic aux performances des réseaux. La seconde doit permettre de garantir les performances du système en adaptant les performances du réseau.

Ce travail de doctorat est réalisé dans le cadre d'un projet de recherche national intitulé *Conception coordonnée des systèmes tolérants aux défauts contrôlés en réseaux* (Safe-Necs)[1]. Ce projet financé en partie par l'Agence Nationale de la Recherche (ANR) regroupe cinq équipes de cinq laboratoires français :

- CRAN (Centre de Recherche en Automatique de Nancy) ;
- GIPSA-lab (Grenoble Images Parole Signal Automatique) département Automatique ;

[1] http ://safe-necs.cran.uhp-nancy.fr/

- INRIA Rhône-Alpes (Institut National de Recherche en Informatique et en Automatique) ;
- LAAS (Laboratoire d'Analyse et d'Architecture des Systèmes) ;
- LORIA (Laboratoire Lorrain de Recherche en Informatique et ses Applications).

L'objectif de ce projet est double. Le premier est de mettre en œuvre des techniques adaptant le réseau en fonction de l'état du système contrôlé. Ceci est fait en développant des algorithmes d'ordonnancement et des mécanismes de dégradation contrôlée. Le second objectif est d'adapter les algorithmes de commande et de diagnostic aux problèmes posés par le réseau (retards aléatoires, pertes). Ce projet est à l'interaction de trois niveaux de régulation : la boucle fermée de commande, la boucle de supervision et la boucle de contrôle des ordonnancements. Pour mener à bien ce projet, différentes communautés ont collaboré. On peut retrouver la communauté des automaticiens représentée par le CRAN et GIPSA-lab ainsi que celle des informaticiens temps-réel représentée par le LORIA, le LAAS ainsi que l'INRIA Rhône-Alpes.

L'une des tâches de ce projet porte sur l'aspect diagnostic des systèmes en réseau. L'objectif est de tenir compte des possibilités de défaillance du réseau de communication entre le système et son algorithme de diagnostic. Ce thème a été moins étudié que celui de la commande des systèmes en réseau, si on excepte un projet européen (NeCST : Networked Control Systems Tolerant to fault)[2] auquel a participé le CRAN. On voit donc ici s'étendre la notion de diagnostic qui doit considérer le réseau comme un composant susceptible d'être défaillant au même titre que les autres composants du système (capteurs, actionneurs). Il est évident que les pertes de données et leur désynchronisation vont influencer les algorithmes de diagnostic qui, dans les applications conventionnelles, supposent des données régulièrement échantillonnées et synchrones.

L'application choisie pour illustrer notre travail est un mini drone hélicoptère. Celui-ci avait déjà été étudié au département Automatique de GIPSA-lab par une autre équipe de recherche en vue de la réalisation de sa commande et de son observation (dans un schéma classique de boucle fermée). Sur cette application, le caractère embarqué rajoute une difficulté à l'analyse de sûreté. En effet, compte tenu de l'applicatif bas coût et des contraintes en termes de poids embarqué et de taille, on ne peut pas envisager de redondance matérielle. Pour le module de diagnostic, l'approche utilisée est de fusionner les données issues de plusieurs modalités de mesure (accéléromètres, magnétomètres, gyromètres). De plus, le système embarqué incorpore un réseau à travers lequel communiquent les algorithmes de commande, d'observation et de diagnostic avec les capteurs et actionneurs.

Ce travail de doctorat aborde plusieurs problèmes :

[2] http ://www.strep-necst.org/

1. le premier consiste à mettre au point un système de diagnostic performant permettant de détecter et localiser un défaut survenant sur les capteurs le plus tôt possible ;
2. le deuxième problème concerne la mise en œuvre du système en réseau. Ce problème est double. Dans un premier temps il faut tenir compte du réseau et de ses performances dans les algorithmes de commande et de diagnostic. Pour ce faire, il faut garantir le bon fonctionnement du réseau. Dans un second temps, il faut aussi garantir les performances du système. Pour cela, en se basant sur des critères influant la qualité de contrôle, il faut modifier les performances du réseau afin de garantir au mieux les performances souhaitées ;
3. enfin, le dernier problème est la réalisation d'une plateforme expérimentale permettant de valider les solutions proposées.

Ce manuscrit se compose donc de deux grandes parties. Tout d'abord il présente une structure complète de diagnostic afin de pouvoir détecter, localiser et identifier un éventuel défaut survenant sur une centrale d'attitude bas coût constituée par un triaxe d'accéléromètres, un triaxe de magnétomètres et trois gyromètres.

Une centrale d'attitude est utilisée, dans le domaine de l'aéronautique, pour l'estimation de l'orientation des objets. Elle peut aussi être utilisée pour réaliser la capture de mouvement. Celle-ci consiste à déterminer les position et attitude d'un objet en mouvement. Les applications sont très variées et vont du domaine médical (chirurgie assistée par ordinateur) à celui de la réalité virtuelle (jeux vidéo, joysticks, interfaces homme-machine).

Dans cette étude, la centrale d'attitude est utilisée pour estimer l'attitude d'un drone. Ces dernière années, l'intérêt pour les UAV (Unmanned Aerial Vehicle), Véhicules Aériens Autonomes s'est accru. Ces véhicules ont été employés dans un premier temps pour des applications militaires, notamment dans des missions de reconnaissance et de surveillance. Avec le développement de l'informatique et de la micro mais surtout nano-électronique (pour la réalisation des capteurs et actionneurs), les drones se sont perfectionnés. De nos jours, nous pouvons trouver des applications civiles telles que la surveillance urbaine ou routière mais aussi la conservation et la protection de l'environment. Cependant, le faible niveau de fiabilité des drones et l'absence de réglementation constituent un frein à leur circulation dans l'espace aérien.

Il s'agit surtout de garantir la sécurité des biens et des personnes dans le but d'éviter des collisions avec d'autres aéronefs et les écrasements au sol. A cet égard, les conditions de navigation et de contrôle des drones, les opérations de maintenance, la fiabilité des équipements sont autant de facteurs critiques pour lesquels des normes sont à élaborer et des solutions technologiques sont à trouver.
Sur ce point, l'accroissement de la sécurité des vols des drones peut se traduire par la mise en œuvre d'équipements redondants et plus fiables mais aussi par l'implémentation

de systèmes de diagnostic et de lois de commande tolérantes aux défauts.

La première partie est constituée de trois chapitres. Tout d'abord, la notion de représentation d'attitude est introduite suivie de la présentation des différentes mesures constituant une centrale d'attitude.

Dans le chapitre suivant, une étude bibliographique est réalisée sur le diagnostic suivie de la présentation de la contribution proposée dans le domaine du diagnostic des centrales d'attitudes.

Le dernier chapitre est dédié à un exemple. Dans ce chapitre, la centrale d'attitude est utilisée comme capteur pour la stabilisation d'un drone à quatre rotors.

Dans la deuxième partie, ce manuscrit se focalise sur les systèmes appelés "Networked Control Systems", ou "Systèmes commandés en réseaux" (*NCS*).

L'objectif est de contrôler et / ou adapter :

1. soit le système de commande et/ou de diagnostic de l'application pour conserver des performances acceptables (qualité de contrôle) ;
2. soit le système de communication par rapport à des besoins de l'application.

Dans (1), il s'agit d'adapter l'application aux performances du réseau. Cette approche est généralement appelée "control over network". L'objectif consiste à modifier les stratégies de commande et de diagnostic en fonction des retards et des pertes de paquets.

Dans (2), il s'agit d'agir directement sur le réseau. Cette approche est généralement appelée "control of network". L'objectif est de garantir une qualité de service minimale requise par l'application commandée. Pour cela, il faut contrôler l'ordonnancement des messages et limiter la surcharge du support de communication.

La seconde partie du manuscrit est composée de trois chapitres. Nous présentons tout d'abord différents types de réseaux que l'on rencontre aujourd'hui communément dans les "*NCS*".

Dans le chapitre suivant, nous présentons la mise en œuvre d'un tel système à travers l'exemple du quadrotor, à l'aide de simulations. Nous proposons en particulier des solutions pour adapter le diagnostic aux pertes de paquets. Nous avons de même testé quelques unes des propositions des partenaires du projet pour contrôler le réseau, par des politiques de priorité ou d'ordonnancement particulières.

Le manuscrit se termine par un chapitre consacré à l'expérimentation et quelques conclusion et perspectives viennent clorent le document.

Chapitre 2

Représentation d'attitude et centrale d'attitude

Ce chapitre présente tout d'abord quelques rappels sur les outils de représentations d'attitude c'est-à-dire d'orientation d'un objet mobile dans l'espace. On présente ensuite la centrale d'attitude qui sera utilisée dans ce mémoire pour illustrer les algorithmes proposés. Cette centrale est constituée de trois modalités de mesure, à savoir accéléromètre, magnétomètre et gyromètre.

2.1 Représentation de l'attitude

Pour analyser le mouvement d'un objet dans l'espace, il est nécessaire de définir des systèmes de coordonnées comme sur la figure 2.1. N(x_n, y_n, z_n) représente le système de coordonnées inertiel. Le système inertiel le plus utilisé et le plus logique est le système de coordonnées NED correspondant à l'acronyme "North-East-Down" défini à partir d'un plan tangent à la surface de la terre. Dans ce cas, les vecteurs unitaires (x_n, y_n, z_n) coïncident respectivement avec les directions Nord, Est et la pesanteur.
B(x_b, y_b, z_b) représente le système de coordonnées fixé à l'objet. L'origine de ce repère est généralement choisie de telle sorte qu'elle coïncide avec le centre de gravité de l'objet.

Pour exprimer l'attitude d'un objet dans l'espace, différentes représentations peuvent être utilisées, chacune ayant ses avantages et inconvénients. Le choix dépend directement de l'application envisagée. Le lecteur peut se reporter à [80], où l'on retrouve une excellente étude sur les différentes représentations de l'attitude.

Soient $\vec{b}$ et $\vec{r}$ les coordonnées d'un vecteur $\vec{x}$ exprimé dans B et N respectivement.

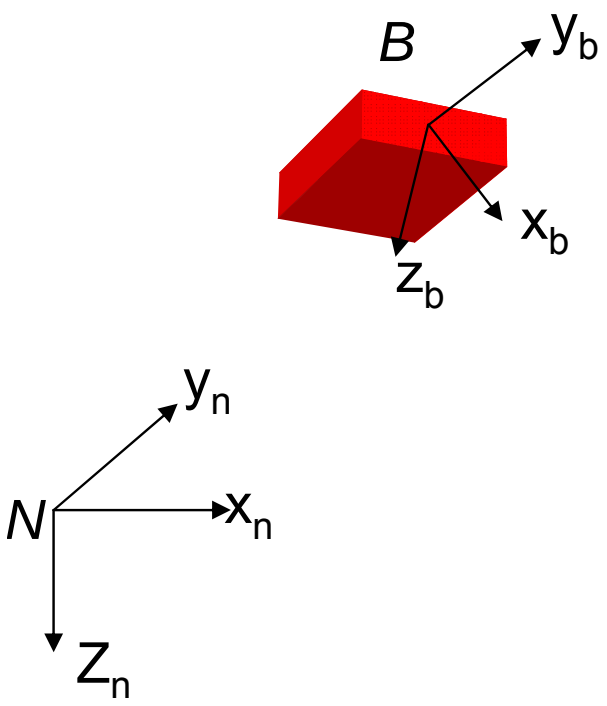


FIG. 2.1 – Repères inertiel et mobile d'un objet

Le vecteur $\vec{b}$ peut être écrit en termes du vecteur $\vec{r}$. Soit $\vec{e} = [e_1 \ e_2 \ e_3]^T$ un vecteur unitaire colinéaire à l'axe de rotation L autour duquel B est tourné d'un angle β afin de coïncider avec N. En conséquence, $\vec{b}$ est obtenu par :

$$\vec{b} = \cos\beta\vec{r} + (1 - \cos\beta)\vec{e}\,\vec{e}^{\,T}\vec{r} - \sin\beta\vec{e} \times \vec{r} \tag{2.1.1}$$

grâce à la définition suivante :

Définition 1 (Rotation simple [29])**.** Le mouvement d'un système de coordonnées B par rapport à un système de coordonnées N est appelé rotation simple de B à N, s'il existe une droite L, appelée axe de rotation, dont l'orientation par rapport à B et N reste inchangée entre le début et la fin du mouvement.

Les coordonnées de $\vec{b}$ et $\vec{r}$ sont liées par la transformation suivante :

$$\vec{b} = C\vec{r} \tag{2.1.2}$$

La matrice C peut être interprétée comme un opérateur qui prend un vecteur fixe $\vec{r}$ exprimé dans N et l'exprime dans B. De l'expression (2.1.1), il découle :

$$C = \cos\beta I_3 + (1 - \cos\beta)\vec{e}\,\vec{e}^{\,T} - \sin\beta[\vec{e}^{\,\times}] \tag{2.1.3}$$

où I_3 représente la matrice identité de dimension trois et $[\overrightarrow{\xi}^{\times}]$ représente un tenseur

anti-symétrique associé au vecteur $\overrightarrow{\xi}$:

$$[\xi^{\times}] = \begin{pmatrix} \xi_1 \\ \xi_2 \\ \xi_3 \end{pmatrix}^{\times} = \begin{pmatrix} 0 & \xi_3 & -\xi_2 \\ -\xi_3 & 0 & \xi_1 \\ \xi_2 & -\xi_1 & 0 \end{pmatrix} \tag{2.1.4}$$

La matrice $C \in \mathbb{R}^{3\times3}$ traduit complètement l'orientation du repère mobile B par rapport au repère inertiel N et elle permet de faire la transformation de coordonnées d'un vecteur d'un système de coordonnées à un autre. Cette matrice est appelée matrice de cosinus directeurs (**DCM**), matrice de rotation, matrice de passage ou encore matrice d'attitude.

2.1.1 Matrice de rotation

La matrice de rotation C n'est pas quelconque, elle appartient au sous-espace des matrices orthogonales de dimension trois, noté $S0(3)$ et défini par :

$$S0(3) = \{C \mid C \in \mathbb{R}^{3\times3}, C^T C = I_3, \det(C) = 1\} \tag{2.1.5}$$

Dans une matrice de rotation C, chaque element c_{ij} est un cosinus directeur, tel que :

$$C = \begin{pmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \end{pmatrix} \tag{2.1.6}$$

Soient :

$$\mathbf{c}_1 = \begin{pmatrix} c_{11} \\ c_{21} \\ c_{31} \end{pmatrix} \quad \mathbf{c}_2 = \begin{pmatrix} c_{12} \\ c_{22} \\ c_{32} \end{pmatrix} \quad \mathbf{c}_3 = \begin{pmatrix} c_{13} \\ c_{23} \\ c_{33} \end{pmatrix} \tag{2.1.7}$$

En conséquence,

$$C = (\mathbf{c}_1 \quad \mathbf{c}_2 \quad \mathbf{c}_3) \tag{2.1.8}$$

où

$$\mathbf{c}_i^T \mathbf{c}_i = 1 \quad \text{et} \quad \mathbf{c}_i^T \mathbf{c}_j = 0 \quad \forall i \neq j \tag{2.1.9}$$

2.1.2 Vitesse de rotation

Dans le cas où le système de coordonnées B tourne par rapport au repère N avec une vitesse angulaire $\vec{\omega} = [\omega_1 \ \omega_2 \ \omega_3]^T$, la matrice C devient variable dans le temps et elle est notée $C(t)$. La relation entre l'attitude du corps et la vitesse angulaire $\vec{\omega}$ s'écrit :

$$\dot{C}(t) = [\vec{\omega}^{\ \times}] C(t) \tag{2.1.10}$$

Notons que $\vec{\omega}$ est la vitesse angulaire du corps par rapport au système N exprimée dans B. L'équation précédente est connue comme l'équation cinématique de l'objet. Cette équation est utilisée dans les systèmes de navigation.

2.1.3 Angles d'Euler et angles de Cardan

La manière la plus simple de décrire une rotation dans l'espace à trois dimensions est d'utiliser les angles d'Euler (ψ, θ, φ). Cependant, il faut faire attention aux confusions car dans certains cas, les angles d'Euler sont confondus avec les angles de Cardan.
En utilisant les angles d'Euler, le passage d'un système de coordonnées $N(x, y, z)$ à un système de coordonnées $B(X', Y', Z')$ s'effectue par 3 rotations successives comme le montre la figure 2.2 :

1. la précession (nom donné au changement graduel d'orientation de l'axe de rotation d'un objet) ψ, autour de l'axe Oz, fait passer de $Oxyz$ au référentiel $Ouvz$;
2. la nutation (la nutation est un balancement périodique de l'axe de rotation) θ, autour de l'axe Ou, fait passer de $Ouvz$ à $OuwZ'$;
3. la rotation propre φ, autour de l'axe OZ', fait passer de $Ouvz$ au référentiel lié au solide $OX'Y'Z'$.

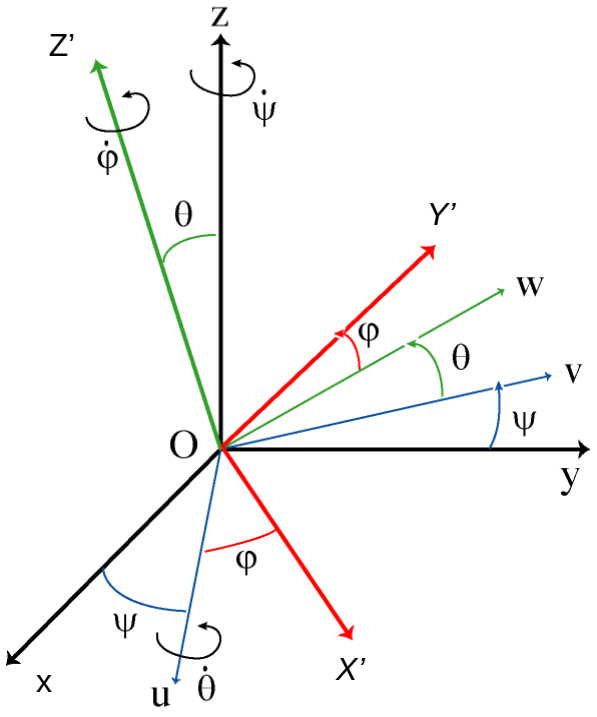


FIG. 2.2 – Angles d'Euler, d'après [98]

Les angles de Cardan décrivent quant à eux la séquence de rotations la plus utilisée dans l'aéronautique et la robotique. En utilisant les angles de Cardan, le passage d'un

système de coordonnées $N(x_n, y_n, z_n)$ à un système de coordonnées $B(x_b, y_b, z_b)$ s'effectue lui aussi en 3 rotations (voir figure 2.3) :

1. Rotation de ψ autour de z_b (angle de lacet avec $-\pi \leq \psi \leq \pi$) (roulis) ;
2. Rotation de θ autour de y_b (angle de tangage avec $-\frac{\pi}{2} \leq \theta \leq \frac{\pi}{2}$) (tangage) ;
3. Rotation de ϕ autour de x_b (angle de roulis avec $-\pi \leq \phi \leq \pi$) (lacet).

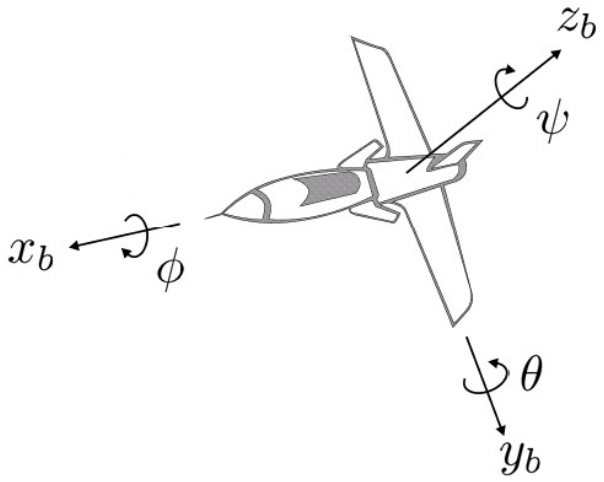


FIG. 2.3 – Angles de Cardan, d'après [39]

Les matrices de rotation qui représentent chaque rotation sont données par [80] :

$$C_{x,\phi} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \sin\phi \end{pmatrix} \tag{2.1.11}$$

$$C_{y,\theta} = \begin{pmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{pmatrix} \tag{2.1.12}$$

$$C_{z,\psi} = \begin{pmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2.1.13}$$

En conséquence, la matrice de rotation C qui décrit le passage du système de coordonnées N au système de coordonnées B devient :

$$C = C_{z,\psi}C_{y,\theta}C_{x,\phi} = \begin{pmatrix} c\psi c\theta & -s\psi c\theta + c\psi s\theta s\phi & s\psi s\phi + c\psi c\phi s\theta \\ s\psi c\theta & c\psi c\phi + s\phi s\theta s\psi & -c\psi s\phi + s\theta s\psi c\phi \\ -s\theta & c\theta s\phi & c\theta c\phi \end{pmatrix} \tag{2.1.14}$$

où $c = \cos(\cdot)$ et $s = \sin(\cdot)$.
Les angles de Cardan sont utilisés plus fréquemment car cette représentation est la plus simple et surtout elle est très intuitive pour décrire une rotation dans l'espace. Néanmoins, elle a le désavantage de présenter une singularité géométrique lorsque $\theta = \pm\frac{\Pi}{2}$, appelée *Gimbal lock*, qui n'est autre qu'une perte de degrés de liberté. De plus, cette singularité géométrique introduit une singularité dans l'équation cinématique associée à cette représentation (voir l'équation (2.1.15)).
La singularité peut être éliminée par la limitation de l'orientation du corps dans l'espace. Par conséquent, cette représentation n'est pas adéquate lorsque le corps évolue dans tout l'espace (toutes les orientations sont possibles).
Soit $\vec{\omega} = [\omega_x \ \omega_y \ \omega_z]^T$ la vitesse angulaire du corps dans le système de coordonnées B, par rapport au système de coordonnées N. Alors, l'équation cinématique s'écrit [29] :

$$\begin{pmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix} = \begin{pmatrix} 1 & \tan\theta\sin\phi & \tan\theta\cos\phi \\ 0 & \cos\phi & -\sin\phi \\ 0 & \frac{\sin\phi}{\cos\theta} & \frac{\cos\phi}{\cos\theta} \end{pmatrix} \begin{pmatrix} \omega_x \\ \omega_y \\ \omega_z \end{pmatrix} \tag{2.1.15}$$

Pour pallier cette singularité, on peut utiliser une autre représentation appelée quaternion.

2.1.4 Quaternions

Une autre représentation de l'attitude est le quaternion unitaire aussi appelé paramètre d'Euler. Le quaternion est une solution alternative au théorème d'Euler, qui énonce qu'une rotation dans l'espace peut être réalisée par une simple rotation β autour d'un axe de rotation $\vec{e}$.
L'algèbre des quaternions est défini dans [20]. Le quaternion unitaire est composé d'un vecteur unitaire $\vec{e}$, nommé axe d'Euler et d'un angle de rotation β autour de cet axe. Il est défini par [80] :

$$q = \begin{pmatrix} \cos\frac{\beta}{2} \\ \vec{e}\sin\frac{\beta}{2} \end{pmatrix} = \begin{pmatrix} q_0 \\ \vec{q} \end{pmatrix} \in \mathbb{H} \tag{2.1.16}$$

où

$$\mathbb{H} = \{q \mid q_0^2 + \vec{q}^T\vec{q} = 1, q = [q_0 \ \ \vec{q}^{\,T}]^T, q_0 \in \mathbb{R}, \ \vec{q} \in \mathbb{R}^3\} \tag{2.1.17}$$

$\vec{q} = [q_1 \quad q_2 \quad q_3]^T$ et q_0 sont la partie vectorielle et la partie scalaire du quaternion respectivement. Le quaternion identité et le quaternion conjugué sont définis par :

$$q_{id} = [1 \ \ 0^T \ ]^T \quad \bar{q} = [q_0 \quad -\vec{q}^{\,T} \ ]^T \tag{2.1.18}$$

Comme le quaternion est unitaire, $q^{-1} = \bar{q}$.
La multiplication de deux quaternions quelconques $q_1 = [q_{1_0} \ \ \vec{q}_1^{\,T} \ ]^T$ et $q_2 = [q_{2_0} \ \ \vec{q}_2^{\,T} \ ]^T$

est définie par :

$$q_1 \otimes q_2 = \begin{pmatrix} q_{1_0} & -\vec{q_1}^T \\ \vec{q_1} & I_3 q_{1_0} + [\vec{q_1}^\times] \end{pmatrix} \begin{pmatrix} q_{2_0} \\ \vec{q_2} \end{pmatrix} \tag{2.1.19}$$

À l'aide de l'algèbre des quaternions [20], des rotations finies dans l'espace peuvent être traitées d'une façon simple et élégante. Soit b_q et r_q les quaternions associés aux vecteurs $\vec{b}$ et $\vec{r}$, et définis par :

$$b_q = [0 \;\; \vec{b}^{\,T}\,]^T \quad r_q = [0 \;\; \vec{r}^{\,T}\,]^T \tag{2.1.20}$$

Ces deux quaternions sont liés par la relation :

$$b_q = q^{-1} \otimes r_q \otimes q = \bar{q} \otimes r_q \otimes q \tag{2.1.21}$$

La matrice de rotation C peut donc être exprimée en termes de quaternion par :

$$C = C(q) = (q_0^2 - \vec{q}^{\,T}\vec{q}\,)I_3 + 2(\vec{q}\,\vec{q}^{\,T} - q_0[\vec{q}^{\,\times}]) \tag{2.1.22}$$

d'où

$$C(q) = \begin{pmatrix} 2(q_0^2+q_1^2)-1 & 2(q_1q_2+q_0q_3) & 2(q_1q_3-q_0q_2) \\ 2(q_1q_2-q_0q_3) & 2(q_0^2+q_2^2)-1 & 2(q_0q_1+q_2q_3) \\ 2(q_0q_2+q_1q_3) & 2(q_2q_3-q_0q_1) & 2(q_0^2+q_3^2)-1 \end{pmatrix} \tag{2.1.23}$$

La relation entre les deux vecteurs $\vec{b}$ et $\vec{r}$ est :

$$\vec{b} = C(q)\vec{r} \tag{2.1.24}$$

Remarque 1. De l'équation (2.1.22) , il est clair qu'une même matrice $C(q)$ est obtenue pour le quaternion q et pour le quaternion $-q$, i.e. $C(q) = C(-q)$. Ceci implique donc que les quaternions q et $-q$ représentent la même attitude physique mais que le sens de rotation est différent. Pour q, la rotation est effectuée avec un angle β autour de $\vec{e}$, tandis que pour $-q$ elle est effectuée avec un angle $2\pi - \beta$, autour du vecteur $\vec{e}$.

Soit $\vec{\omega} = [\omega_x \;\; \omega_y \;\; \omega_z]^T$ la vitesse angulaire du corps rigide dans le système de coordonnées B par rapport à N, exprimée dans B. Alors, l'équation cinématique (2.1.10) exprimée en utilisant la notion de quaternion devient :

$$\begin{pmatrix} \dot{q}_0 \\ \dot{\vec{q}} \end{pmatrix} = \frac{1}{2} \begin{pmatrix} -\vec{q}^{\,T} \\ I_3 q_0 + [\vec{q}^{\,\times}] \end{pmatrix} \vec{\omega}$$
$$\dot{q} = \frac{1}{2}\Xi(q)\vec{\omega} = \frac{1}{2}\Omega(\overrightarrow{\omega})q \tag{2.1.25}$$

avec

$$\Xi(q) = \begin{pmatrix} -\vec{q}^{\,T} \\ I_3 q_0 + [\vec{q}^{\,\times}] \end{pmatrix}, \Omega(\overrightarrow{\omega}) = \begin{pmatrix} 0 & -\overrightarrow{\omega}^T \\ \overrightarrow{\omega} & -\overrightarrow{\omega}^\times \end{pmatrix}, \overrightarrow{\omega}^\times = \begin{pmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{pmatrix} \tag{2.1.26}$$

2.1.5 Paramètres de Rodrigues

Les paramètres de Rodrigues peuvent être utilisés pour s'affranchir de la contrainte de normalité imposée par la représentation de l'attitude à l'aide des quaternions. De cette façon, le nombre d'éléments nécessaires pour décrire la cinématique est réduit de quatre à trois. Cette paramétrisation est donnée par la définition suivante :

$$\vec{\varrho} = \frac{1}{q_0}\,\vec{q} \;=\; \vec{e}\tan\frac{\beta}{2} \tag{2.1.27}$$

La matrice de rotation C peut être exprimée en utilisant les paramètres de Rodrigues par :

$$C(\vec{\varrho}) = I_3 + \frac{2}{1+\vec{\varrho}^{\,T}\vec{\varrho}}\,([\vec{\varrho}^{\,\times}][\vec{\varrho}^{\,\times}] - I_3) \tag{2.1.28}$$

En conséquence, l'équation cinématique peut être écrite :

$$\dot{\varrho}_1 \;=\; \frac{1}{2}(\omega_1 - \omega_2\varrho_3 + \omega_3\varrho_2 + \varrho_1\sum_{i=1}^{3}\varrho_i\omega_i) \tag{2.1.29}$$

$$\dot{\varrho}_2 \;=\; \frac{1}{2}(\omega_2 - \omega_3\varrho_1 + \omega_1\varrho_3 + \varrho_2\sum_{i=1}^{3}\varrho_i\omega_i) \tag{2.1.30}$$

$$\dot{\varrho}_3 \;=\; \frac{1}{2}(\omega_3 - \omega_1\varrho_2 + \omega_2\varrho_1 + \varrho_3\sum_{i=1}^{3}\varrho_i\omega_i) \tag{2.1.31}$$

A partir de l'équation (2.1.27) il est évident que les paramètres de Rodrigues peuvent être utilisés uniquement pour effectuer une rotation autour de l'axe $\vec{e}$ lorsque $\beta \neq \pm\pi$.

2.1.6 Paramètres de Rodrigues modifiés

Dans la section précédente, nous avons observé que les paramètres de Rodrigues permettent d'éliminer la contrainte de normalité associée au quaternion. Cependant, lorsque $q_0 = 0$, les paramètres partent à l'infini, à cause de la présence d'une singularité dans la représentation pour $\beta = \pm\pi$.
Une autre possibilité pour éliminer la contrainte de normalité associée au quaternion est d'utiliser le paramétrage suivant, appelé paramètres de Rodrigues modifiés :

$$\vec{p} = \frac{1}{1+q_0}\,\vec{q} = \vec{e}\tan\frac{\beta}{4} \tag{2.1.32}$$

Il s'en suit que la matrice de rotation C peut être écrite avec les termes des paramètres de Rodrigues modifiés, par :

$$C(\vec{p}) = I_3 + \frac{4(1-\vec{p}^{\,T}\vec{p}\,)}{(1+\vec{p}^{\,T}\vec{p}\,)^2}[\vec{p}^{\,\times}] + \frac{8}{(1+\vec{p}^{\,T}\vec{p}\,)^2}[\vec{p}^{\,\times}]^2 \tag{2.1.33}$$

En conséquence, l'équation cinématique est exprimée par l'équation suivante :

$$\dot{p}_1 = \frac{1}{2}(\frac{\omega_1\tilde{p}}{2} - \omega_2 p_3 + \omega_3 p_2 + p_1 \sum_{i=1}^{3} p_i\omega_i) \quad (2.1.34)$$

$$\dot{p}_2 = \frac{1}{2}(\frac{\omega_2\tilde{p}}{2} - \omega_3 p_1 + \omega_1 p_3 + p_2 \sum_{i=1}^{3} p_i\omega_i) \quad (2.1.35)$$

$$\dot{p}_3 = \frac{1}{2}(\frac{\omega_3\tilde{p}}{2} - \omega_1 p_2 + \omega_2 p_1 + p_3 \sum_{i=1}^{3} p_i\omega_i) \quad (2.1.36)$$

avec $\tilde{p} = 1 - p_1^2 + p_2^2 + p_3^2$. Les paramètres dans l'équation (2.1.32) sont aussi appelés coordonnées stéréographiques du quaternion [90] (projection d'une sphère dans un plan). En effet, ceci permet de faire le lien entre la sphère unité (obtenue par normalité) définie dans un espace de dimension quatre et l'espace euclidien de dimension trois, où le point (-1,0,0,0) est utilisé comme point de base de la projection.
Cependant, l'équation (2.1.32) montre que les rotations autour de l'axe $\vec{e}$ ne peuvent être toutes réalisées. Il y a une singularité pour $\beta = 2\pi$.

2.1.7 Différences d'attitudes exprimées par les quaternions

Considérons deux attitudes d'un corps rigide, paramétrées par les matrices de rotation C_1 et C_2 respectivement. l'écart est calculé grâce à :

$$C_r = C_1 C_2^{-1} = C_1 C_2^T \quad (2.1.37)$$

C_r représente un opérateur d'attitude qui fait tourner $\overrightarrow{b}_2$ en $\overrightarrow{b}_1$. A partir de cette observation, en général, l'écart d'attitude est utilisé dans le cadre de l'estimation et de la commande d'attitude comme une mesure de *l'erreur d'attitude*. Dans ce contexte soit $C_d = C_1$ l'attitude désirée du corps rigide et $C = C_2$ l'attitude réelle de ce même corps. Alors, l'écart d'attitude est obtenue par le calcul de :

$$C_e = C_d C^{-1} \quad (2.1.38)$$

Si l'écart d'attitude vaut zéro, alors, $C_e = I_3$.
Lorsque la représentation de l'attitude du corps est faite par le quaternion unitaire alors le quaternion caractérisant l'écart est obtenu par :

$$q_e = q_1^{-1} \otimes q_2 = \begin{pmatrix} q_{1_0} & \vec{q}_1^T \\ -\vec{q}_1 & I_3 q_{1_0} - [\vec{q}_1^{\times}] \end{pmatrix} \begin{pmatrix} q_{2_0} \\ \vec{q}_2 \end{pmatrix} = \bar{q}_1 \otimes q_2 \quad (2.1.39)$$

Le plus petit des deux angles de rotation entre les attitudes caractérisées par les quaternions q_1 et q_2 respectivement est calculé par l'expression suivante :

$$\beta_r = 2|\arccos(q_{e_0})| \quad (2.1.40)$$

L'erreur d'attitude est utilisée dans le cadre de la commande d'attitude. Elle sera également exploitée dans le cadre du diagnostic d'une centrale d'attitude comme on le verra ultérieurement. Dans le cas où l'erreur d'attitude vaut zéro, le quaternion d'erreur a deux valeurs possibles :

$$q_e = [\pm 1 \quad 0]^T \tag{2.1.41}$$

2.2 Capteurs utilisés pour l'estimation d'attitude

Les composants intégrés sur silicium, les MEMS (Micro Electro Mechanical System) permettent de réaliser en grande quantité des capteurs performants à des dimensions micrométriques. Parmi la grande diversité de capteurs disponibles en technologie MEMS, nous pouvons citer les capteurs inertiels constitués par les accéléromètres et les gyromètres. Nous pouvons aussi trouver des capteurs qui permettent la mesure du champ magnétique terrestre, les magnétomètres. Le point fort de ce type de capteurs est leur petite taille, leur faible consommation énergétique et leur coût. Par conséquent, ils sont parfaitement adaptés pour les applications embarquées grand public.
L'ensemble de ces trois types de capteurs permet la conception de systèmes capables de fournir l'information nécessaire sur l'attitude d'un solide dans l'espace. Néanmoins, la fusion de toutes ces données est indispensable. Les méthodes de fusion des données issues de ces capteurs utilisent les modèles mathématiques correspondant à la sortie de chaque capteur.
Dans les paragraphes suivants, nous allons décrire brièvement la technologie et les modèles de mesure des trois capteurs mentionnés précédemment, afin d'exploiter cette information pour diagnostiquer ces capteurs.

2.2.1 Gyromètre

Un gyromètre permet d'effectuer une mesure de la vitesse de rotation. Les phénomènes physiques utilisés pour réaliser ce type de capteurs sont essentiellement les propriétés inertielles de la matière [95].

Les gyromètres utilisent plusieurs éléments vibrants qui produisent des signaux sinusoïdaux identiques et d'amplitude constante lorsque le capteur est au repos. Lorsque celui-ci est soumis à un mouvement de rotation, les forces de Coriolis induisent une variation d'amplitude de ces signaux. Les principaux paramètres qui spécifient la performance d'un gyromètre sont la résolution, le biais, la sortie à vitesse nulle ZRO (Zero-Rate Output) et le facteur d'échelle. En l'absence de rotation, le signal de sortie d'un gyromètre est modélisé par un bruit blanc et une fonction lentement variable. Le bruit blanc influence la résolution du capteur et celle-ci est exprimée en termes d'écart type ou de vitesse

de rotation divisée par la racine carrée de la bande passante $[(^\circ/sec)/\sqrt{Hz}]$. Le facteur d'échelle représente le gain entre le signal de sortie et la vitesse de rotation, exprimé en $[V/(^\circ/s)]$. Le tableau 2.1 donne les caractéristiques standards des gyromètres utilisés dans la détermination de l'attitude et la navigation inertielle pour le capteur *MAG3* [47] de chez Memsense qui sera implanté dans notre expérience.

Paramètres	Spécifications
Etendue de mesure	± 150 $(^\circ/sec)$
Dérive	< 0.01 $(^\circ/h)$
Précision du facteur d'échelle	0.0125
Bande passante	0-50 (Hz)
Densité de Bruit	0.1 $(^\circ/sec/\sqrt{Hz})$
Gamme de température	$0^\circ C$ à 70 $^\circ C$

TAB. 2.1 – Caractéristiques du gyromètre implanté dans la centrale *MAG3* de Memsense [47]

Modèle de gyromètre : Le signal de sortie du gyromètre est constitué de trois paramètres définis par :

$$\omega_G = \omega + \nu + \eta_G \tag{2.2.1}$$

où ω est la vitesse réelle de rotation, ν est une fonction lentement variable dans le temps et η_G est un bruit blanc lié à la résolution du capteur.

De l'équation (2.2.1), il est clair que lorsqu'il n'y a pas de rotation, le signal de sortie du gyromètre est composé d'un bruit blanc et d'une fonction lentement variable dans le temps. La dynamique de cette fonction est traditionnellement modélisée par un processus de Gauss-Markov [39] décrit par l'équation :

$$\dot{\nu} = -\frac{1}{\tau}\nu + \eta_\nu \tag{2.2.2}$$

Si trois gyromètres sont montés en triaxe orthogonal dans un solide, tels que leurs axes sensibles coïncident avec les principaux axes d'inertie du solide, la sortie des gyromètres et l'évolution du biais deviennent :

$$\vec{\omega}_G = \vec{\omega} + \vec{\nu} + \vec{\eta}_G \tag{2.2.3}$$

$$\dot{\vec{\nu}} = -T^{-1}\vec{\nu} + \vec{\eta}_\nu \tag{2.2.4}$$

où les composantes des vecteur $\vec{\eta}_G$ et $\vec{\eta}_\nu \in \mathbb{R}^3$ sont supposées être des bruits blancs gaussiens et $T = \tau I_3$ est une matrice diagonale de constantes de temps.

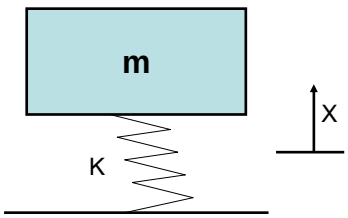
Remarque 2. Afin de déterminer l'attitude d'un solide, les gyromètres sont utilisés pour mesurer les vitesses de rotation du corps. De cette manière, en intégrant l'équation cinématique (2.2.1) à partir d'une condition initiale, il serait possible d'obtenir la variation

de l'attitude. Cependant, les sorties des gyromètres étant entachées de biais ν inconnu , celui-ci doit être estimé au cours du temps.

Remarque 3. Notons que dans certains cas, la dérive ν peut être négligée. C'est en particulier le cas pour les capteurs récents, en particulier le capteur retenu dans notre étude.

2.2.2 Accéléromètre

Un accéléromètre [94] est un capteur qui, fixé à un mobile, permet de mesurer l'accélération de ce dernier. Dans son principe physique, un accéléromètre peut être vu comme une masse m attachée à un ressort qui est lui-même attaché à un repère fixe comme le montre la figure 2.4.
Le principe de tous les accéléromètres est basé sur le principe fondamental de la dynamique $\overrightarrow{F} = m\overrightarrow{a}$ où $\overrightarrow{F}$ est la force, m la masse d'épreuve et $\overrightarrow{a}$ l'accélération. Une accélération $\overrightarrow{a}$ entraînera un déplacement $\overrightarrow{x}$. Donc, si nous observons le déplacement x, il est possible d'en déduire l'accélération. De cette façon, mesurer l'accélération consiste en réalité à mesurer le déplacement de la masse sismique attachée au ressort.



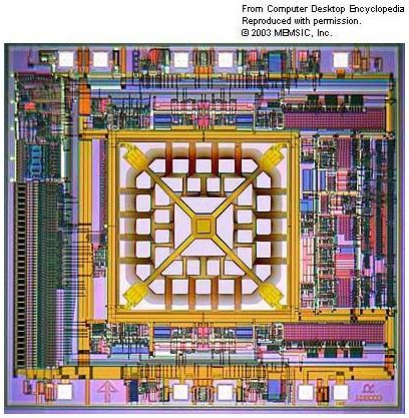


FIG. 2.4 – (gauche) Principe de l'accéléromètre, (droite) Exemple d'accéléromètre MEMS (MEMSIC, Inc)

Les principaux paramètres d'un accéléromètre sont tout d'abord la plage de mesure généralement normalisée en g (accélération de la pesanteur au niveau de la mer pour une latitude de 45°, $g = 9,80665\ m/s^2$), la bande passante (en Hz), la précision, la sensibilité (mV/g), la gamme de température d'utilisation (en $°C$), la masse du capteur, le nombre d'axes (1 à 3 axes), la construction mécanique, l'électronique intégrée. Le tableau 2.2 présente un exemple de caractéristiques disponibles pour le capteur *MAG3* de chez Memsense implanté dans notre application. Toutes les caractéristiques de l'accéléromètre dépendent de la technologie et du procédé de fabrication. Différents types

de transducteurs peuvent être utilisés pour la construction des accéléromètres. Les techniques les plus utilisées sont à détection piézoélectrique, à détection capacitive, à jauge de contrainte/extensométrie, à poutre vibrante, à ondes de surface et à détection optique [5].

Paramètres	Spécifications
Étendue de mesure	$\pm$ 2 g
Bande passante	0-50Hz
Précision	$< 4\mu g$
Bruit (X et Z)	$92\mu g/\sqrt{Hz}$
Bruit (Y)	$110\mu g/\sqrt{Hz}$
Gamme de température	0°C à 70 °C

TAB. 2.2 – Caractéristiques de l'accéléromètre implanté dans la centrale *MAG3* de Memsense [47]

Modèle pour l'accéléromètre : le signal de sortie d'un accéléromètre est décrit par la projection sur l'axe de l'accéléromètre de la somme d'une accélération a et de la gravité g avec un terme de bruit η_A, considéré blanc et gaussien, de moyenne nulle :

$$b_A = (a - g) + \eta_A \tag{2.2.5}$$

Où les trois accéléromètres sont montés en triaxe orthogonal dans un solide, avec leurs axes sensibles qui coïncident avec les principaux axes d'inertie du solide, la sortie des accéléromètres peut être écrite comme le vecteur de mesures :

$$\vec{b}_A = C(q)\vec{r}_A + \vec{\eta}_A \tag{2.2.6}$$

avec $\vec{r}_A = (\vec{a} - \vec{g})$, où $\vec{g} = [0 \;\; 0 \;\; g]^T$ et $\vec{a} \in \mathbb{R}^3$ représentent respectivement les vecteurs de gravité et d'accélération propre du corps, donnés dans le système de coordonnées inertiel N. g est la constante gravitationelle et $\vec{\eta}_A \in \mathbb{R}^3$ est le vecteur de bruits supposés indépendants, blancs et gaussiens, à moyenne nulle.
La matrice $C(q)$ est la matrice de rotation [20] associée au quaternion q, ce dernier caractérisant l'attitude du solide par rapport au repère inertiel N.

Remarque 4. Si l'accélération $\vec{a}$ est faible par rapport à la gravité, $\|\vec{a}\| \ll \|\vec{g}\|$, les mesures des accéléromètres donnent une projection du vecteur de gravité dans le système de coordonnées inertiel B. En conséquence $\vec{r}_A \approx -\vec{g}$ et :

$$\vec{b}_A = -C(q)\vec{g} + \vec{\eta}_A \tag{2.2.7}$$

Dans ce cas, $\vec{\eta}_A$ représente la somme du vecteur de bruit et des perturbations induites par des vibrations ou des accélérations parasites. Le vecteur $\vec{\eta}_A$ ne peut plus alors être considéré comme un vecteur de bruits blancs et gaussiens. En l'absence d'accélération, les vecteurs de mesures sont normalisés. Par conséquent, on prendra $\vec{g} = [0 \;\; 0 \;\; g]^T/||\vec{g}|| = [0 \;\; 0 \;\; 1]^T$.
En présence d'accélération $\vec{a}$, le vecteur de mesure sera normalisé par g.

2.2.3 Magnétomètre

Parmi les instruments de navigation qui permettent de s'orienter par rapport à un système de référence absolu, nous trouvons le compas magnétique. Cet instrument utilise le champ magnétique terrestre comme référence, en conséquence il permet de suivre un cap [96].

Le Champ Géomagnétique International de Référence (IGRF) et le Modèle Magnétique Mondial (WMM) sont les deux modèles les plus utilisés, notamment pour la navigation.
Dans des applications comme celles traitées dans ce document, la zone d'action représentée par un système inertiel NED est extrêmement petite par rapport à la surface de la Terre. Par conséquent, dans cette zone le champ magnétique peut être considéré constant en absence de perturbations magnétiques. Dans le système inertiel NED, le champ magnétique est représenté par un vecteur $\vec{r}_M = [r_{M_x} \;\; 0 \;\; r_{M_z}]^T$ (voir figure 2.5). La variation d'orientation du champ magnétique terrestre dans un système de coordonnées mobile peut être déterminée à partir de l'évolution des 3 composantes cartésiennes du champ dans ce système de coordonnées.

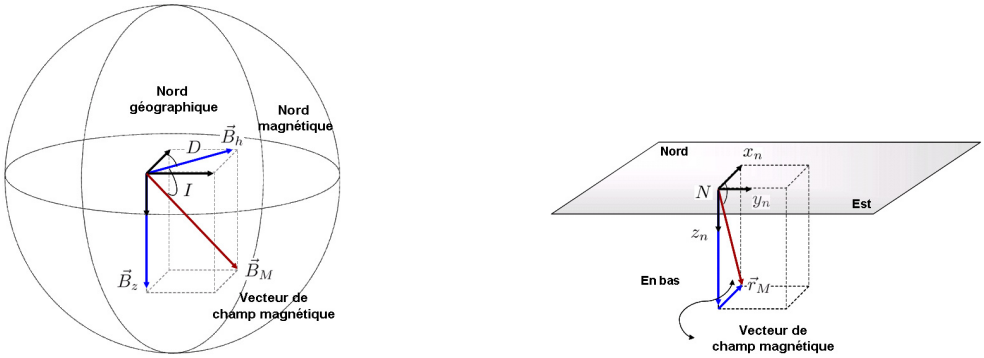


FIG. 2.5 – (gauche) Champ magnétique terrestre, (droite) Champ magnétique *NED*

La figure 2.5 a) montre les composantes du champs magnétique terrestre. Les composantes sont :

- $\overrightarrow{B}_M$ qui correspond à l'intensité totale du vecteur champ magnétique ;
- $\overrightarrow{B}_h$, la composante horizontale du vecteur champ magnétique ;
- $\overrightarrow{B}_Z$, la composante verticale du vecteur champ magnétique. Par convention, Z est positif vers le bas.

D correspond à la déclinaison magnétique, définie comme étant l'angle entre le nord vrai et la direction de la composante horizontale du champ magnétique, mesurée vers l'est à partir du nord vrai. I fournit l'inclinaison magnétique, soit l'angle que fait le vecteur champ magnétique par rapport au plan horizontal et dont la valeur est positive vers le bas.

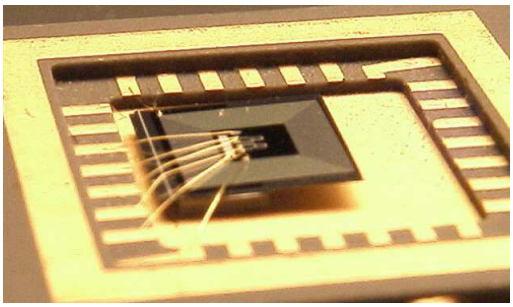


FIG. 2.6 – Exemple d'un magnétomètre à technologie MEMS

Les méthodes de mesure des champs magnétiques sont assez nombreuses et il existe plusieurs principes de capteurs magnétiques. Le tableau 2.3 résume les caractéristiques du magnétomètre implanté dans la centrale d'attitude $MAG3$ de Memsense et qui sera utilisée par la suite. En pratique, les capteurs à effets magnétorésistifs répondent au cahier des charges des applications embarquées en raison de leur performance, coût, taille et facilité de mise en œuvre. L'unité du champ magnétique est le Gauss. La figure 2.7 montre la table de conversion entre le Tesla et le Gauss.

Remarque 5. De même que pour les accéléromètres, la mesure des magnétomètres est normalisée.

T = Tesla	G = Gauss
1 T	10 000 G
100 mT	1 000 G
10 mT	100 G
1 mT	10 G
100 µT	1 G
10 µT	100 mG
1 µT	10 mG
100 nT	1 mG
10 nT	100 µG
1 nT	10 µG

FIG. 2.7 – Table de conversion : Tesla en Gauss, d'après [33]

Paramètres	Spécifications
Étendue de mesure	± 1.9 Gauss
Bande passante	0-50Hz
Densité de bruit	0.14 $mGauss/\sqrt{Hz}$
Gamme de température	0°C à 70 °C

TAB. 2.3 – Caractéristiques du magnétomètre implanté dans la centrale $MAG3$ de Memsense [47]

Modèle de mesure pour le magnétomètre : Le signal de sortie d'un magnétomètre est décrit par la somme d'une projection du champ magnétique terrestre $\vec{r}_M$ dans

B et un terme de bruit η_M considéré blanc et gaussien à moyenne nulle. Si trois magnétomètres sont montés en triaxe orthogonal dans un solide, tels que leurs axes coïncident avec les principaux axes d'inertie du solide, la sortie du magnétomètre triaxe dans le système de coordonnées mobile peut être écrit sous la forme d'un vecteur de mesures :

$$\vec{b}_M = C(q)\vec{r}_M + \vec{\eta}_M \tag{2.2.8}$$

Dans notre cas, un modèle théorique du champ magnétique, consiste à considérer le vecteur de champ magnétique unitaire avec un angle d'inclination $I = 60°$. Nous avons donc : $\vec{r}_M = [r_{M_x} \;\; 0 \;\; r_{M_z}]^T = [\frac{1}{2} \;\; 0 \;\; \frac{\sqrt{3}}{2}]^T$.

Remarque 6. À titre d'information l'intensité approximative du champ magnétique à Grenoble est $\|\overrightarrow{r}_M\| = 40000\ nT = 400mG$.

2.3 Conclusion

Dans ce chapitre, différentes représentations de l'attitude ont été abordées. Il faut remarquer que toutes les représentations à trois paramètres présentent des non linéarités associées à leurs équations cinématiques et dans tous les cas, il existe des singularités géométriques. Les deux représentations qui ne présentent pas de singularité sont le *quaternion unitaire* et la *matrice de rotation*. Néanmoins, ces représentations sont redondantes. La matrice de rotation doit satisfaire les contraintes propres aux matrices de rotation (orthogonalité, $C^T = C^{-1}$). Le quaternion doit être de norme unité. Notons que l'équation cinématique associée au quaternion est linéaire si la vitesse de rotation est supposée constante ; elle est donc plus facilement manipulée.
Dans la suite de ce manuscrit, la représentation de l'attitude avec le quaternion unitaire a été retenue. Tous les algorithmes développés sont basés sur ce formalisme. Le choix de cette représentation est justifié par les points mentionnés précédemment. De plus, comme les applications envisagées sont en temps réel et embarquées, la représentation de l'attitude à partir du quaternion possède un grand avantage par rapport aux autres sur le plan de la stabilité numérique et du coût calcul.
Les angles d'Euler sont cependant plus intuitifs et largement utilisés dans la littérature (voir paragraphe 2.1.3). En conséquence, nous ferons aussi référence à des positions angulaires notées (ϕ, θ, ψ) = (lacet, roulis, tangage) et nous présenterons les résultats sous cette forme. De cette façon, le lecteur qui n'est pas familiarisé avec le quaternion pourra interpréter physiquement les résultats obtenus.

Chapitre 3

Drones et quadrotor expérimental

Un drone ou UAV pour *Unmanned Aerial Vehicule* est un objet aérien de reconnaissance sans pilote, télécommandé ou programmé. Les drones sans pilote et les drones de combat sans pilote représentent une étape importante dans le développement de l'aéronautique, tant militaire que civile [87].

Dans ce chapitre, les différentes catégories de Véhicules Aériens Autonomes (UAV) vont être décrites de manière générale. Dans un second temps, nous nous focaliserons sur le drone à quatre rotors, véhicule pouvant atterrir et décoller à la verticale. Dans un troisième temps, le prototype du département Automatique de GIPSA-lab sera décrit. Il est réalisé dans le cadre du projet ANR Safe-Necs. Enfin, nous verrons les stratégies de commande et d'observation du quadrotor ainsi que son comportement sous l'effet de défauts du capteur "centrale d'attitude".

3.1 Catégories de drones

Les drones peuvent être classés selon trois critères principaux comme la taille ou les performances. Sous le terme performance sont regroupés l'endurance qui représente le temps de vol du drone (l'autonomie) et l'altitude. Celle-ci correspond à l'altitude de croisière. La famille des drones comprend plusieurs catégories :

- les drones miniatures ;
- les drones tactiques, à voilure fixe ou tournante appelés TUAV (*Tactical Unmanned Air Vehicule*) ;
- les drones volant à moyenne altitude et de grande autonomie appelés MALE (*Medium altitude Long Endurance*) ;
- les drones volant à haute altitude et de grande autonomie appelés HALE (High Altitude Long Endurance) ;
- les drones de combat, encore appelés UCAV (*Unmanned Combat Air Vehicule*).

3.1.0.1 Les drones miniatures

Rentrent dans cette catégorie tous les drones dont l'envergure est inférieure à 50 cm. Les *mini-drones* inférieurs à 15 cm, pèsent environs 50 g et possèdent une vitesse de croisière de l'ordre de 50 km/h. L'autonomie est d'environ une vingtaine de minutes pour un rayon d'action d'une dizaine de kilomètres. Ces mini drones sont généralement dédiés à la transmission d'images. Les missions sont la reconnaissance d'un itinéraire, l'évaluation de dommages ou bien encore l'observation d'une cible. Dans le civil, ces machines peuvent être utilisées en milieu urbain afin de survoler des quartiers voire rentrer dans des immeubles. On peut imaginer des applications civiles pour les pompiers ou la police. Même si le marché est quasi inexistant aujourd'hui, c'est sûrement dans le domaine civil que les drones peuvent jouer un grand rôle grâce à leur souplesse et polyvalence. Les applications sont presque infinies, avec des missions qui peuvent être considérées dangereuses, pénibles pour l'équipage ou bien ennuyeuses. La figure 3.1 montre quelques exemples d'applications dans le domaine civil, comme par exemple la surveillance des réseaux routiers, avalanches, côtes maritimes, feux de forêts [48].

FIG. 3.1 – Exemples de missions civiles pour un drone miniature [48]

3.2 Drone 4 rotors

Parmi les véhicules capables de décoller et d'atterrir à la verticale, le plus connu est certainement l'hélicoptère standard qui est composé d'un rotor principal et d'un rotor de queue. Cependant, depuis quelques années, de nombreuses études sont menées sur d'autres véhicules aériens autonomes, tels que par exemple, le drone à quatre rotors souvent appelé X4-Flyer. Le drone à quatre rotors ou quadrotor possède certains avantages par rapport aux hélicoptères conventionnels. En raison de sa symétrie, ce véhicule est dynamiquement élégant, simple à construire. En fait, il est plus aisé de réaliser un

vol stationnaire avec quatre forces de poussées opérant à une même distance du centre de masse qu'avec une seule force de poussée agissant sur le centre de masse. De plus, les pales peuvent être protégées par un carénage, ce qui rend leur utilisation plus sûre que celle des hélicoptères conventionnels, qui sont dangereux à utiliser dans des espaces réduits à cause des pales tournantes non protégées.

La miniaturisation des capteurs, des composants électroniques, l'augmentation des capacités de traitement des calculateurs et le progrès réalisé dans le développement des batteries à polymères de nouvelle génération rendent aujourd'hui possible l'implantation de systèmes embarqués, miniaturisés et autonomes.

FIG. 3.2 – Quadrotor de Draganfly innovation, Inc [51]

Le Draganfly (voir figure 3.2) est un drone à quatre rotors, radio-commandé, disponible chez "Draganfly innovation, Inc" [51]. Cette plateforme est idéale pour comprendre certains concepts liés au vol de ce type d'aéronefs. Bien que le constructeur mentionne que cet aéronef soit beaucoup plus facile à piloter qu'un hélicoptère conventionnel (sur le site web on voit des personnes faisant voler le Draganfly), nous savons qu'il faut être un pilote expert pour réaliser un vol stationnaire avec cet engin.

Le travail présenté dans ce manuscrit a, parmi ses objectifs, le développement d'algorithmes de diagnostic qui permettent de détecter des défauts survenant sur un drone à quatre rotors.

3.3 Architecture du prototype du projet *Safe-NECS*

Avant de présenter le prototype, présentons, tout d'abord rapidement, le projet *Safe-NECS* [45]. Ce projet est un projet de recherche de l'ANR (Agence Nationale de la

Recherche) dans le cadre de l'action de recherche Sécurité des Systèmes embarqués et Intelligence Ambiante (SSIA), projet intitulé *Safe-NECS* pour *Conception coordonnée des systèmes tolérants aux fautes contrôlés en réseaux*. Ce projet réunit des équipes du Département Automatique de GIPSA-lab Grenoble, du CRAN et du LORIA de Nancy du LAAS de Toulouse et de l'INRIA Rhône-Alpes. Un des objectif de ce travail consiste à développer un banc d'essai pour tester les algorithmes proposés par les différents partenaires. Celui-ci est un mini drone à quatre rotors initialement acheté chez Draganfly innovation [51].

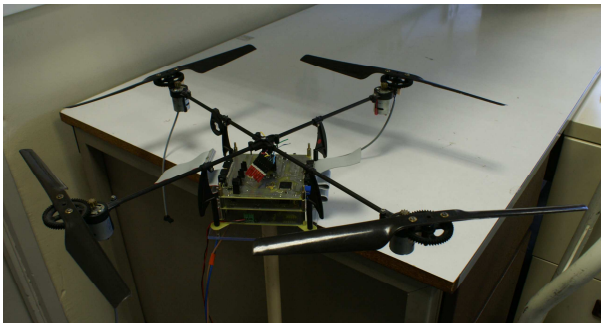


FIG. 3.3 – Quadrotor du projet *Safe-NECS*

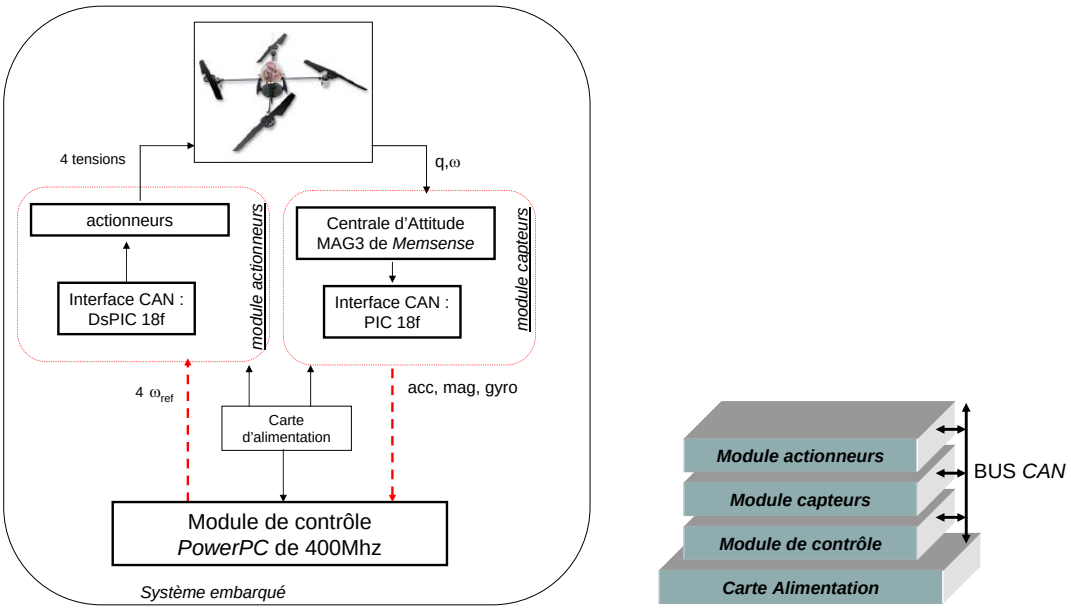


FIG. 3.4 – (gauche) Architecture électronique du prototype du projet *Safe-NECS*, (droite) Liaison des différentes cartes par le bus CAN

Dans le cadre de ce projet, et afin de pouvoir atteindre nos objectifs, toute l'architecture électronique du drone a été réalisée au sein du GIPSA-lab. Sur la figure 3.3, on peut remarquer plusieurs étages de cartes électroniques. Chaque étage communique avec

les autres via le bus de terrain CAN. Dans cette architecture, nous pouvons identifier cinq modules principaux (voir figure 3.4) :

- Module de capteur d'attitude 3.5(a) : ce module est constitué d'une centrale d'attitude de Memsense [47], composée de neuf capteurs de technologie MEMS, à savoir, trois gyromètres, trois accéléromètres et trois magnétomètres. Ce module permet d'obtenir les mesures afin d'estimer l'attitude dans le power PC. Le module est piloté par un micro-contrôleur $PIC18F$ de Microchip [68].
- Module actionneurs 3.5(b) : il implante une boucle interne locale pour commander les actionneurs du drone qui sont constitués d'une hélice entrainée par un moteur à courant continu. Les consignes sont traitées dans ce module par un micro-contrôleur $dsPIC33F$ toujours chez Microchip [67].
- Module de contôle 3.5(c) : ce module est constitué d'un PowerPc de $400Mhz$ intitulé Tiny MPC5200B de chez Phytec [49]. Dans ce module seront réalisés le calcul de l'observateur ainsi que celui de la commande et enfin celui du diagnostic.
- Communication : pour cette architecture, la communication est assurée par un bus de terrain CAN. Ce bus est décrit par la suite.
- Carte d'alimentation : Cette carte électronique permet d'alimenter les différentes cartes décrites ci-dessus.

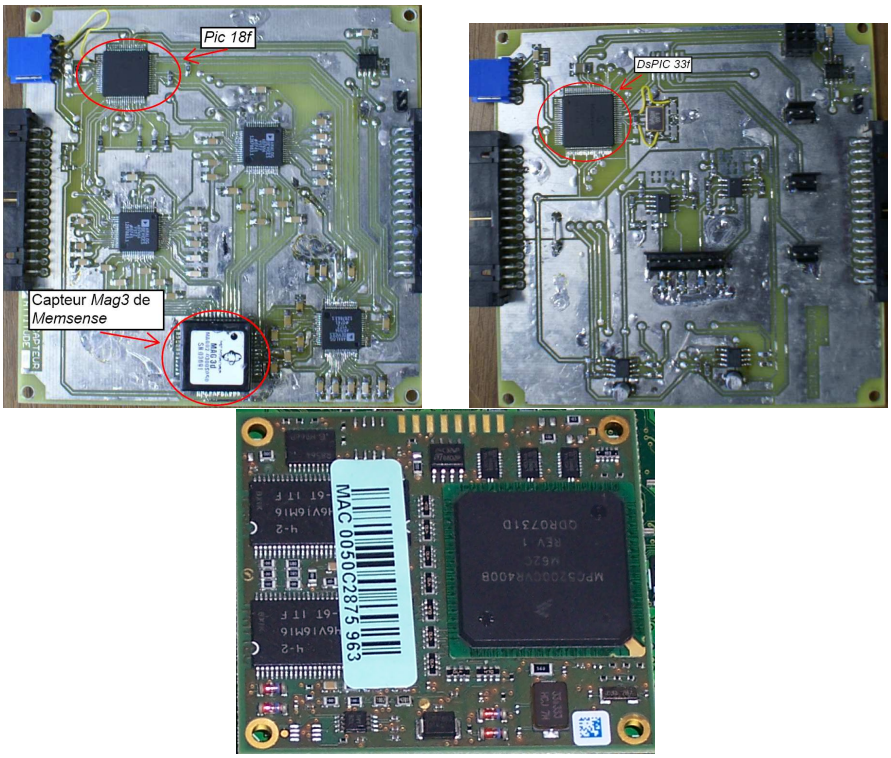


FIG. 3.5 – (haut gauche) Carte capteur, (haut droite) Carte moteur, (bas) Carte PowerPC

3.4 Fonctionnement du drone à quatre rotors

Comme montré sur la figure 3.6, les moteurs avant et arrière (M_1, M_3) tournent dans le sens contraire des aiguilles d'une montre alors que les moteurs droit et gauche (M_2, M_4) tournent dans le sens des aiguilles d'une montre. Chaque actionneur produit une force f_i parallèle à son axe de rotation, ainsi qu'un couple résistant Q_i opposé au sens de rotation. La force totale ou poussée totale exercée sur l'hélicoptère (parallèle à l'axe z_b) est la somme des quatre forces générées par chaque actionneur ($F_T = f_1 + f_2 + f_3 + f_4$). La combinaison des forces f_i et des couples résistants Q_i donne origine aux mouvements angulaires autour des axes principaux du drone :

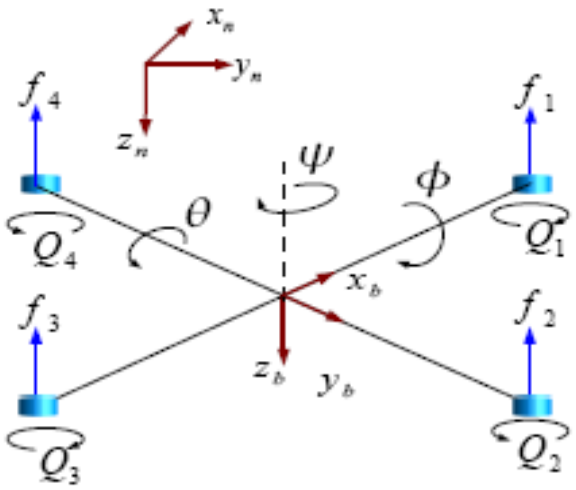


FIG. 3.6 – Fonctionnement du quadrotor

mouvement de tangage (θ) : ce mouvement est assuré par la différence des forces (f_1, f_3) produites par les actionneurs avant et arrière. Cette différence de forces produit un couple τ_θ autour de l'axe y_b ;

mouvement de roulis (ϕ) : ce mouvement est assuré par la différence des forces (f_2, f_4) produites par les actionneurs droit et gauche. Cette différence de forces produit un couple τ_ϕ autour de l'axe x_b ;

mouvement de lacet (ψ) : ce mouvement est assuré par la somme des couples de traînée (Q_i) produits par les quatre actionneurs. Étant donné que les sens de rotation des actionneurs (M_1, M_3) et (M_2, M_4) sont opposés, nous pouvons régler la somme des quatre couples résistants. Quand les quatre rotors tournent à la même vitesse, ils sont soumis au même couple résistant, donc la somme est nulle. Par conséquent, il n'y a pas de rotation autour de l'axe z_b. Par contre, si nous provoquons une différence de vitesse entre les moteurs tournant en sens opposé, les couples résistants provoquent un couple τ_ψ autour de l'axe z_b, provoquant ainsi la rotation de l'engin.

De manière générale et d'après la description des mouvements angulaires, nous pouvons classer trois principaux types de vols :

le vol stationnaire : en montée verticale et après avoir franchi le seuil définissant l'effet du sol (HES)[1], l'hélicoptère peut rester en vol stationnaire à une certaine hauteur constante par rapport au sol en ayant une vitesse de translation nulle. La force de sustentation doit alors équilibrer le poids mg de l'hélicoptère. Cette force de sustentation correspond à une force de poussée F_T qui est orientée (en l'absence de perturbation) dans la direction de l'axe z_b. Dans ce mode de vol, l'hélicoptère a la liberté de faire des rotations autour de l'axe z_b qui, dans ce cas, coïncide avec l'axe z_n du système de coordonnées inertiel ;

les translations verticales : elles sont définies lorsque l'hélicoptère se déplace suivant l'axe z_n. En l'absence de perturbations, la force de poussé F_T est toujours verticale et en montée elle est toujours supérieure au poids de l'hélicoptère ($F_T > mg$) tandis qu'en descente elle est inférieure ($F_T < mg$). La combinaison de l'inclinaison ($\theta \neq 0$ ou $\phi \neq 0$) de l'hélicoptère et de la force de poussée F_T produit une composante de la force suivant l'axe x_n et/ou y_n. Cette force est connue comme la force de traction et celle-ci assure la translation du système dans la direction du vol souhaitée. Par conséquent, les translations verticales sont aussi définies quand l'hélicoptère se déplace dans deux directions simultanément, par exemple dans les plans $x_n z_n$ ou $y_n z_n$;

les translations horizontales : elles sont définies de façon similaire aux translations verticales mais cette fois-ci dans le plan $x_n y_n$. Lorsqu'une translation est effectuée suivant la direction x_n et la force de poussée maintient le système à une hauteur constante par rapport au sol. Le système effectue un vol connu dans la littérature sous le nom de "vol en palier".

D'après la description des mouvements angulaires et des déplacements verticaux et horizontaux, nous pouvons remarquer que le déplacement de l'engin dépend directement de son orientation (attitude). De cette façon, le déplacement peut être contrôlé en commandant de façon performante l'orientation, donnant origine à une commande semi-automatique de la position. Ceci explique pourquoi nous nous focaliserons dans ce document sur la commande des trois angles ϕ θ ψ, bien plus facile à expérimenter en laboratoire. Maintenant, nous allons nous intéresser au modèle de ce drone.

[1] Lorsque l'hélicoptère est près du sol, l'air aspiré par les actionneurs (moteur-hélice-réducteur) forme une couche d'air comprimé entre le véhicule et le sol. Cet effet est connu sous le nom d'effet de sol et en général, il est considéré nul quand l'hélicoptère se trouve à une hauteur égale ou supérieure à son diamètre.

3.5 Modèle mécanique du quadrotor

Le drone 4 rotors utilisé est modélisé par la composition de deux PVTOL (Planar Vertical Take Off and Landing) possédant 6 degrés de liberté provenant de l'orthogonalité des deux axes. Chaque moteur électrique produit une force et un couple notés f_i et Q_i respectivement comme indiqué sur la figure 3.6. Pour établir les équations du mouvement du quadrotor, certaines hypothèses ont été considérées :

1. la structure en forme de croix est supposée rigide ;
2. le drone possède une structure parfaitement symétrique, ce qui permet d'avoir une matrice d'inertie diagonale ;
3. l'évolution du vol en palier et du vol stationnaire est supposée être effectuée hors de l'effet du sol ;
4. la force de poussée f_i et le couple résistant Q_i générés par chaque actionneur sont supposés proportionnels au carré de la vitesse de rotation de la pâle noté ω_{M_i} ;
5. les perturbations induites par le vent sont initialement négligées.

Pour décrire les équations dynamiques du système, plusieurs approches peuvent être utilisées. L'approche de Newton-Euler exploite les concepts de forces et couples. Il existe aussi l'approche de Lagrange qui exploite les concepts d'énergie potentielle et cinétique. Pour représenter l'attitude, le choix s'est porté sur le quaternion unitaire (pour les raisons évoquées dans le paragraphe 2.1.4). De ce fait, l'approche de Newton-Euler sera utilisée pour exprimer les équations dynamiques du quadrotor. En effet, le fait d'utiliser le quaternion unitaire pour représenter l'attitude empêche d'utiliser l'approche de type Lagrange car le quaternion ne peut pas être considéré comme un système de coordonnées généralisées [39].

La partie dynamique du système peut être décrite par l'ensemble des équations suivantes, en considérant les repères N et B :

$$\dot{\overrightarrow{p}} = \overrightarrow{\nu} \tag{3.5.1}$$

$$\dot{\overrightarrow{\nu}} = \overrightarrow{g} - \frac{1}{m} C^T(q) \overrightarrow{F} \tag{3.5.2}$$

$$\dot{q} = \frac{1}{2} q \otimes \Omega \tag{3.5.3}$$

$$I_f \dot{\overrightarrow{\omega}} = -[\overrightarrow{\omega}^{\times}] I_f \overrightarrow{\omega} - G_a + \tau_a \tag{3.5.4}$$

$$I_r \dot{\omega}_{M_i} = \tau_{Mi} - Q_i, \quad i \in \{1:4\} \tag{3.5.5}$$

où m est la masse du quadrotor, g correspond à l'accélération de la gravité. Le vecteur $\overrightarrow{p} = [x, y, z]^T$ correspond à la position de l'origine du repère B par rapport au repère N. Le vecteur $\overrightarrow{\nu} = [\nu_x, \nu_y, \nu_z]^T$ est la vitesse linéaire de l'origine du repère B mesurée dans le repère N. q est le quaternion d'attitude. $\Omega = [0, \overrightarrow{\omega}^T]^T$ est un quaternion, avec $\overrightarrow{\omega} \in \Re^3$ la vitesse angulaire du quadrotor mesurée par les gyromètres dans le repère mobile B. $I_f \in \Re^{3\times3}$ correspond à la matrice d'inertie du drone. $\overrightarrow{F}$=[0 0 T]^T représente la somme des forces engendrées par les actionneurs et est définie dans le repère B. La vitesse de rotation de la pâle et le moment d'inertie du rotor i sont notés respectivement ω_{M_i} et I_r. Le rotor est ici considéré être constitué par le moteur et la pâle.

Les couples s'exerçant sur la structure du drone sont notés G_a et τ_a^i avec i=1,2,3. G_a représente les couples gyroscopiques et τ_a^i les couples de commande induits par la combinaison des forces et couples résistants générés par les actionneurs. Le couple résistant Q_i généré par le rotor i est :

$$Q_i = k\omega_{M_i}^2 \tag{3.5.6}$$

La poussé totale générée par les quatres rotors est :

$$T = \sum_{i=1}^{4} f_i = b\sum_{i=1}^{4} \omega_{M_i}^2 \tag{3.5.7}$$

avec $f_i = b_i\omega_{M_i}^2$ la portance générée par chaque rotor i. Le vecteur G_a contient les couples gyroscopiques :

$$G_a = \sum_{i=1}^{4} I_r(\omega \times e_Z)(-1)^{i+1}\omega_{M_i} \tag{3.5.8}$$

Les couples auxquels est soumis le quadrotor, générés par les quatres rotors sont :

$$\begin{aligned} \tau_a^1 &= db(\omega_{M_2}^2 - \omega_{M_4}^2) \\ \tau_a^2 &= db(\omega_{M_1}^2 - \omega_{M_3}^2) \\ \tau_a^3 &= k(\omega_{M_1}^2 - \omega_{M_2}^2 + \omega_{M_3}^2 - \omega_{M_4}^2) \end{aligned} \tag{3.5.9}$$

$\tau_a = [\tau_a^1, \tau_a^3, \tau_a^3]^T$, avec d la distance entre le rotor et le centre de gravité du quadrotor.

Remarque 7. Pour le quadrotor considéré, le vecteur de mesures (fourni par la centrale d'attitude) est constitué par :

- les trois mesures provenant du tri-axe d'accéléromètres, notées $\overrightarrow{b}_a = [acc_X, acc_Y, acc_Z\ ]^T$;
- les trois mesures provenant du tri-axe de magnétomètres, notées $\overrightarrow{b}_m = [mag_X, mag_Y, mag_Z\ ]^T$;
- les trois mesures provenant des gyromètres, notées $\overrightarrow{\omega}_g = [\omega_X, \omega_Y, \omega_Z\ ]^T$.

TAB. 3.1 – Définition des paramètres et notations du quadrotor

f_i	force produite par chaque moteur électrique
Q_i	couple produit par chaque moteur électrique
ω_{Mi}	vitesse de rotation de chaque pale
ω_{Mi}^{ref}	vitesse de rotation de référence
$\overrightarrow{\omega}_g$=$[gyro_X\ gyro_Y\ gyro_Z]^T$	vecteur de mesure des gyromètres
$\overrightarrow{b}_a$=$[acc_X\ acc_Y\ acc_Z]^T$	vecteur de mesure des accéléromètres
$\overrightarrow{b}_m$=$[mag_X\ mag_Y\ mag_Z]^T$	vecteur de mesure des magnétomètres
τ_a	vecteur de commande (couple)
q_{est}	attitude estimée sous le formalisme des quaternions
I_r	moment d'inertie d'un rotor
T	la poussé générée par les quatres rotors
d	distance entre le rotor et le centre de gravité de l'engin
b	paramètre de la force de poussée
k	paramètre de trainée
I_f	matrice d'inertie du quadrotor
m	masse du quadrotor
$\overrightarrow{\eta}_a$	bruit blanc gaussien, centré sur les accéléromètres
$\overrightarrow{\eta}_M$	bruit blanc gaussien, centré sur les magnétomètres
$\overrightarrow{\eta}_G$	bruit blanc gaussien, centré sur les gyromètres

Par conséquent, dans la suite, le vecteur de mesure sera noté par :

$$y_M = \begin{bmatrix} \overrightarrow{b}_a \\ \overrightarrow{b}_m \\ \overrightarrow{\omega}_g \end{bmatrix} = \begin{bmatrix} acc_X \\ acc_Y \\ acc_Z \\ mag_X \\ mag_Y \\ mag_Z \\ gyro_X \\ gyro_Y \\ gyro_Z \end{bmatrix} \tag{3.5.10}$$

Rappelons que $\overrightarrow{B}_a$ est normalisé par le champ de gravité et $\overrightarrow{B}_M$ est normalisé par le champ magnétique.

Le tableau 3.1 résume les paramètres et notations utilisés dans ce rapport.

3.6 Commande en attitude

L'élaboration de la loi de commande ainsi que celle d'un observateur ont été réalisées dans le cadre du travail de doctorat de J.F. Guerrero Castellanos intitulé : *Estimation de l'attitude et commande bornée en attitude d'un corps rigide : Application à un mini hélicoptère à quatre rotors*, soutenu le 11 Janvier 2008 [39]. Nous allons ici résumer l'objectif général d'une commande en attitude et la loi de commande retenus, telle que proposé dans [39].

3.6.1 Problème

L'objectif est la conception d'une loi de commande qui amène le corps rigide à une attitude constante et le maintienne dans cette attitude par la suite. Soit q_d le quaternion qui représente l'attitude désirée. En conséquence, l'objectif de la loi de commande est représentée par la condition asymptotique suivante :

$$q \to q_d, \quad \vec{\omega} \to 0, \quad t \to \infty \tag{3.6.1}$$

Dans le cas où l'attitude désirée est représentée par le quaternion $q_d = [\pm 1 \quad 0 \quad 0 \quad 0]^T$ (nous voulons que le repère mobile B coïncide avec le repère inertiel N), le quaternion d'erreur défini au paragraphe (2.1.39) devient $q_e = q$ et l'objectif de la loi de commande est :

$$q \to [\pm 1 \quad 0 \quad 0 \quad 0]^T, \quad \vec{\omega} \to 0, \quad t \to \infty \tag{3.6.2}$$

Rappelons que q et $-q$ représentent la même attitude physique.
Dans l'analyse de stabilité seul le cas $q_d = [1 \quad 0 \quad 0 \quad 0]^T$ a été considéré. Néanmoins, les résultats peuvent être appliqués à n'importe quelle orientation désirée. En plus de la condition de stabilité asymptotique, la loi de commande doit prendre en compte les contraintes physiques des actionneurs, afin de fournir un signal de commande compatible avec les actionneurs (pas de saturation en particulier).

De plus, il faut une loi de commande simple, de sorte qu'elle puisse être implantée en temps réel sur le système où la capacité de calcul est réduite. Pour résoudre ce problème, différentes approches sont possibles telles que :

1. une approche de type commande optimale ;
2. une approche non linéaire.

D'après [39], l'approche non linéaire est plus intéressante par sa simplicité et notons que la stabilisation en attitude d'un corps rigide est un problème qui a attiré l'attention ces dernières années de la communauté automaticienne. Cet intérêt est dû à l'analogie existante entre le modèle dynamique d'un corps rigide et celui de nombreux systèmes tels que les satellites, les véhicules marins ou les hélicoptères. Pour résoudre le problème de la stabilisation, une loi de commande bornée a été étudiée [39].

3.6.2 Loi de commande

3.6.2.1 Principe général

Dans ce paragraphe, la loi de commande qui stabilise en attitude un corps rigide est résumée [39]. Le but est la conception d'une commande en attitude qui soit bornée, afin d'utiliser toute l'entendue de la gamme de travail des actionneurs, sans saturation des actionneurs en garantissant la stabilité.

Avant d'énoncer le résultat nous commençons par une définition de la fonction de saturation.

Définition 2 (Fonction de saturation)**.** Soit M une constante positive. La fonction $\sigma_M : \mathbb{R} \to \mathbb{R}$ est une fonction continue et non décroissante telle que :

$$\sigma_M(x) = \begin{cases} x & \text{si} \quad |x| < M; \\ \text{sign}(x)M & \text{dans les autres cas}; \end{cases} \tag{3.6.3}$$

où $sign(x)$ est le signe du réel x.

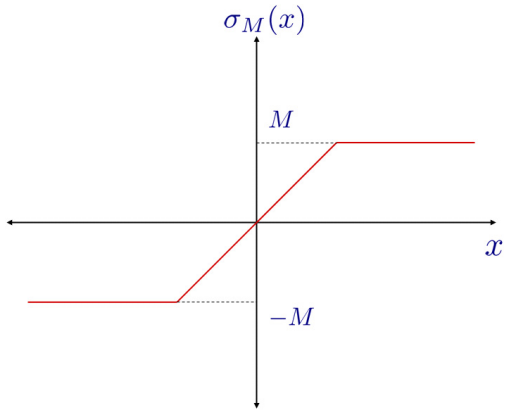


FIG. 3.7 – Fonction de saturation

Théorème 1. Considérons la dynamique du corps rigide (3.5.3) et (3.5.4). La loi de commande bornée $\tau_a^i = [\tau_a^1, \tau_a^2, \tau_a^3]^T$:

$$\begin{aligned} \tau_a^1 &= -\alpha_1 \sigma_{M_1}(\lambda_1[\omega_1 + \rho_1 q_1]) \\ \tau_a^2 &= -\alpha_2 \sigma_{M_2}(\lambda_2[\omega_2 + \rho_2 q_2]) \\ \tau_a^3 &= -\alpha_3 \sigma_{M_3}(\lambda_3[\omega_3 + \rho_3 q_3]) \end{aligned} \tag{3.6.4}$$

stabilise globalement et asymptotiquement le corps rigide à l'origine ($q_0 = 1, \vec{q} = 0$, $\vec{\omega} = 0$) où σ_{M_i} sont des fonctions de saturation, avec $M_i \geq 3\lambda_i\rho_i$ et α_i, λ_i, ρ_i sont des paramètres positifs, $i = \{1, 2, 3\}$.

De façon générale, le théorème précédant signifie que peu importe la condition initiale du corps rigide (attitude et vitesse angulaire), la loi de commande (3.6.4) est normalement capable d'amener les variables d'état du système (l'attitude et la vitesse angulaire) à l'origine de façon asymptotique. De plus, l'amplitude du couple qui s'exerce autour des axes principaux du corps rigide est bornée par M_i. La borne M_i peut être choisie de façon indépendante pour chaque axe et elle dépend des caractéristiques des actionneurs utilisés. La loi de commande (3.6.4) est simple. Elle peut être appliquée à la stabilisation de systèmes en temps réel avec des capacités de calcul contraignantes.
La stabilité de cette loi de commande est démontrée dans [39]. Cette loi de commande a quelques propriétés telles que :

1. la robustesse par rapport à des mesures de vitesse angulaires bornées. La loi de commande (3.6.4) stabilise globalement et asymptotiquement le corps rigide à l'origine en dépit d'une saturation dans les gyromètres ;
2. la robustesse par rapport aux erreurs de modèle. Cette loi est robuste par rapport aux erreurs dans le modèle ;
3. la robustesse par rapport aux perturbations externes.

Toutes ces propriétés sont démontrées dans [39].

3.6.2.2 Application

L'objectif est la stabilisation de l'attitude en boucle fermée du drone à quatre rotors. L'idée consiste en l'élaboration d'une loi de commande τ_a qui permette de stabiliser le drone à quatre rotors vers une attitude désirée, représentée par un quaternion q_d. Pour la conception de la loi de commande, l'auteur de [39] suppose la connaissance de tous les états du système, c'est-à-dire le quaternion d'attitude et le vecteur de vitesse angulaire[2] est possible. La figure 3.8 montre le schéma fonctionnel de la stratégie implémentée aujourd'hui au sein du drone du projet *Safe-NECS*.

La loi de commande fournit un signal de commande τ_a ainsi qu'une commande T qui doit s'exercer autour de chaque axe de l'hélicoptère. Par conséquent, le couple de commande τ_a doit être transformée en quatre signaux de vitesse de rotation désirée ω_{Mi}^{ref} avec $i \in \{1, 2, 3, 4\}$, où i représente le numéro de l'actionneur. La vitesse de rotation

[2]En réalité, les informations sur l'état du système sont obtenues via une centrale de mesure, composé par trois gyromètres, trois accéléromètres et trois magnétomètres. Un observateur d'attitude, présenté dans la prochaine section sera utilisé afin d'estimer le quaternion d'attitude.

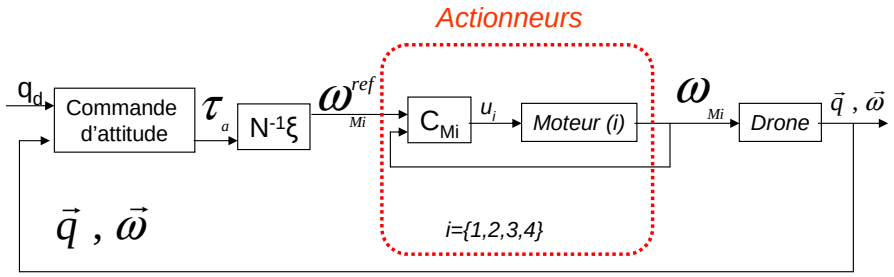


FIG. 3.8 – Commande du quadrotor

désirée, pour chaque actionneur, est obtenue par :

$$\begin{pmatrix} (\omega^{ref}_{M1})^2 \\ (\omega^{ref}_{M2})^2 \\ (\omega^{ref}_{M3})^2 \\ (\omega^{ref}_{M4})^2 \end{pmatrix} = \begin{pmatrix} 0 & db & 0 & -db \\ db & 0 & -db & 0 \\ k & -k & k & -k \\ b & b & b & b \end{pmatrix}^{-1} \begin{pmatrix} \tau_a \\ T \end{pmatrix} \tag{3.6.5}$$

$$\omega^{ref}_{Mi} = N^{-1}\xi$$

Les régulateurs C_{M_i} avec $i \in \{1, 2, 3, 4\}$ ont pour objectif d'élaborer une tension de sorte que l'actionneur i ait une vitesse de rotation ω_{Mi} égale à la vitesse désirée ω^{ref}_M donnée par l'équation (3.6.5). De plus, les régulateurs C_{M_i} doivent assurer que la dynamique du système propulseur soit largement au delà de la bande passante de la commande d'attitude. La conception du régulateur de vitesse C_{M_i} aurait pu être basée sur des approches simples de type PI comme dans [15]. Cependant, le choix s'est basée sur une approche par modes glissants. Cette approche a été justifiée par sa simplicité et robustesse par rapport à des incertitudes de paramètres des moteurs [39].

Dans cette application, la tension maximale qui peut être appliquée aux bornes des moteurs est de 9 V. Lorsque cette tension est appliquée, l'actionneur atteint une vitesse maximale de $\omega_{M,max} = 260\ rad/sec$. En accord avec les valeurs des paramètres aérodynamiques des pâles, k et b, le couple maximal qui peut s'exercer autour de chaque axe du quadrotor est donc :

$$\tau^1_a = 0.40\ Nm \quad \tau^2_a = 0.40\ Nm \quad \tau^3_a = 0.15\ Nm \tag{3.6.6}$$

3.7 Observation

3.7.1 Problème

Des capteurs inertiels accéléromètres et gyromètres, associés à des magnétomètres, tous en technologie $MEMS$, permettent la conception de systèmes de mesure capables

de fournir l'information nécessaire pour déterminer l'attitude d'un solide dans l'espace. Depuis quelques années, le problème de l'estimation de l'attitude en utilisant des capteurs *GAM* (Gyromètre, Accéléromètre et Magnétomètre) a été abordé afin d'appliquer l'estimation d'attitude à des domaines émergents tels que la robotique aérienne, la capture de mouvement humain et la biomécanique. L'approche la plus utilisée pour résoudre ce problème est le Filtre de Kalman Etendu (FKE) [82], [65], [63], en raison de sa capacité à fusionner des signaux provenant de différentes modalités de mesure. Dans notre application, le quadrotor est commandé en attitude. Il faut donc pouvoir déterminer l'attitude du repère mobile B par rapport au repère fixe N. Pour cela, le choix s'est porté sur l'utilisation d'une centrale d'attitude ou *IMU* (*Inertial Measurement Unit*) présentée dans le chapitre 1, paragraphe 2.2. Cette centrale va fournir des mesures permettant d'exprimer l'attitude du repère mobile B attaché au drone par rapport au repère fixe N. (voir (2.2)). Pour rappel, les mesures sont regroupées en trois catégories :

1. accéléromètres :
   on se place sous l'hypothèse de mouvement quasi-statique. Le triaxe d'accéléromètres mesure alors uniquement le champ gravitationnel dans le repère mobile ;
2. magnétomètres :
   l'utilisation du triaxe orthogonal de magnétomètres permet d'avoir des informations sur l'orientation de ce dernier par rapport au nord magnétique ;
3. gyromètres :
   les gyromètres sont utilisés pour avoir une indication sur la vitesse angulaire. La mesure fournie par les gyromètres est parfois entachée d'une dérive lente (voir 2.1). Cependant, dans notre application, on suppose que cette dérive lente est négligeable (hypothèse conforme aux caractéristiques du capteur $MAG3$ implanté).

Pour estimer l'attitude, dans [39], le choix s'est porté sur la mise en œuvre d'un observateur non linéaire. Le principal avantage avancé par [39] pour un observateur non linéaire, à la différence des approches de type Filtre de Kalman Etendu, est la non linéarisation des équations et la garantie de convergence globale de l'erreur à zéro. Pour cela, on peut considérer deux objectifs distincts. Le premier est d'utiliser les accéléromètres et magnétomètres. Ainsi, l'attitude du solide par rapport au système de référence NED ("North-East-Down") est estimée. Cette estimation reste valable pour des mouvements quasi-statiques et sans perturbations magnétiques. En d'autres termes, cela signifie que les mesures des accéléromètres représentent seulement la projection du vecteur de champ de gravité sur le repère attaché au corps B, et celles des magnétomètres représentent, quant à elles, le champ magnétique terrestre. Dans un deuxième temps, le cas des mesures fournies par les trois gyromètres est considéré. Le but est d'obtenir, avec ces mesures, une propagation de l'attitude dans le temps alors que les accéléromètres et magnétomètres permettent d'avoir une estimation de l'attitude à chaque instant. Cet ajout des gyromètres permet en outre de relâcher les contraintes "mouvement quasi-statique" et "pas de perturbation magnétique".

3.7.2 Observateur non linéaire

Maintenant, l'observateur non linéaire va être décrit brièvement. Pour plus de détail, le lecteur peut se reporter à [39]. La figure 3.9 montre le schéma de ce dernier.[3]

3.7.2.1 Principe d'estimation avec les accéléromètres et magnétomètres

Les méthodes d'estimation de l'attitude proposées dans [39] sont basées sur un modèle de mesures particulier. Ce modèle a été proposé pour première fois dans [21]. La caractéristique principale est que si le modèle décrit par

$$\vec{b}_i = C(q)\vec{r}_i + \vec{\eta}_i \tag{3.7.1}$$

est sans bruit, il peut être ramené à un système linéaire bien structuré. De cette façon, le modèle de mesure peut être considéré linéaire par rapport au quaternion.
Soient b_q et r_q les quaternions associés aux vecteurs $\vec{b}_i$ et $\vec{r}_i$, représentant les coordonnées d'un vecteur donné par rapport aux systèmes de coordonnées B et N respectivement. Soient :

$$b_q = [0 \;\; \vec{b}_i{}^T \;]^T \quad r_q = [0 \;\; \vec{r}_i{}^T \;]^T \tag{3.7.2}$$

Comme mentionné dans le paragraphe 2.1.4, ces deux quaternions sont liés par la relation :

$$b_q = \bar{q} \otimes r_q \otimes q = q^{-1} \otimes r_q \otimes q \tag{3.7.3}$$

A partir de la définition du produit de quaternions donnée par l'équation (2.1.19), nous obtenons en multipliant (3.7.3) par q :

$$q \otimes b_q = r_q \otimes q \tag{3.7.4}$$

avec

$$q \otimes b_q = \begin{pmatrix} 0 & -\vec{b}_i{}^T \\ \vec{b}_i & -[\vec{b}_i{}^\times] \end{pmatrix} q \tag{3.7.5}$$

$$r_q \otimes q = \begin{pmatrix} 0 & -\vec{r}_i{}^T \\ \vec{r}_i & [\vec{r}_i{}^\times] \end{pmatrix} q \tag{3.7.6}$$

En soustrayant les équations (3.7.5) et (3.7.6), nous avons :

[3] Notons que dans [39] le biais sur les gyromètres n'est pas négligé mais estimé en même temps que le quaternion. La centrale d'attitude installée sur la nouvelle manipulation ne nécessite pas l'implantation de cette partie de l'algorithme.

$$\begin{pmatrix} 0 & -(\vec{b}_i - \vec{r}_i)^{\,T} \\ \vec{b}_i - \vec{r}_i & -[(\vec{b}_i + \vec{r}_i)^{\,\times}] \end{pmatrix} q = 0$$
$$Hq = 0 \tag{3.7.7}$$

En fait, l'équation (3.7.7) est un modèle de mesure qui ne tient pas compte des bruits, *i.e.* $\vec{\eta}_i = 0$. Lorsque le bruit est considéré, un terme supplémentaire apparaît dans l'équation (3.7.7) qui devient est un système linéaire de la forme :

$$Ax + b = 0$$

où les matrices A $\in \mathbb{R}^{4\times4}$, et b $\in \mathbb{R}^4$.

3.7.2.2 Énoncé du problème général

Soient $\vec{b}_i$ et $\vec{r}_i$ avec $i = a, m$ les vecteurs de mesures et les vecteurs de référence, exprimés dans B et N respectivement. L'équation de mesure (3.7.7) à un instant donné t_k devient :

$$\bar{H}q = 0 \tag{3.7.8}$$

où la matrice $\bar{H}$ est construite comme suit :

$$\bar{H} = \begin{pmatrix} H_1 \\ \vdots \\ H_2 \end{pmatrix} = \begin{pmatrix} \varphi_1^T \\ \vdots \\ \varphi_8^T \end{pmatrix} \in \mathbb{R}^{8\times4} \tag{3.7.9}$$

avec

$$H_i = \begin{pmatrix} 0 & -(\vec{b}_i - \vec{r}_i)^T \\ \vec{b}_i - \vec{r}_i & -[(\vec{b}_i + \vec{r}_i)^{\,\times}] \end{pmatrix} \in \mathbb{R}^{4\times4}$$
$$\varphi_j \in \mathbb{R}^4 \quad \text{avec} \;\; j = 1:8 \tag{3.7.10}$$

Le problème consiste donc à calculer le vecteur $q \in \mathbb{H}$, solution de ce système d'équations linéaires bruitées. Une façon de résoudre ce problème peut être définit par :

$$\hat{q} = \arg\min_q \left\{ f(q) = \frac{1}{2} \left\| \bar{H}q \right\|_2^2 \right\}$$
$$\text{sous la contrainte :} \quad \|q\|_2^2 = 1 \tag{3.7.11}$$

3.7.2.3 Estimation de l'attitude en utilisant une décomposition SVD

Un outil particulièrement utile pour l'analyse matricielle est la décomposition en valeurs singulières (SVD) On peut citer entre autres [107], ou l'on retrouve une étude sur les quaternions en utilisant la SVD. En effet, la SVD va nous permettre la construction explicite de bases orthogonales pour le noyau de la matrice. Ainsi, nous allons utiliser ces propriétés pour trouver le quaternion appartenant au noyau de $\bar{H}$. Nous pouvons alors énoncer le résultat suivant : Soit $\bar{H} \in \mathbb{R}^{8\times4}$ une matrice formée à partir des paires de vecteurs $(\vec{b}_i, \vec{r}_i)$ avec $i = 1 : 2$. Nous supposons qu'au moins deux paires de vecteurs sont non colinéaires. Par conséquent, il existe une matrice orthogonale $U \in \mathbb{R}^{8\times8}$, une matrice orthogonale $V \in \mathbb{R}^{4\times4}$ et une matrice $S \in \mathbb{R}^{8\times4}$ diagonale

$$S = \begin{pmatrix} \sigma_1 & 0 & 0 & 0 \\ 0 & \sigma_2 & 0 & 0 \\ 0 & 0 & \sigma_3 & 0 \\ 0 & 0 & 0 & \sigma_4 \\ 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{3.7.12}$$

avec $\sigma_1 \geq \sigma_2 \geq \sigma_3 \geq \sigma_4 \geq 0$, telles que :

$$\bar{H} = USV^T \tag{3.7.13}$$

Soit $V = [V_1 \;\; V_2 \;\; V_3 \;\; V_4]$, avec $V_i \in \mathbb{R}^4$ les vecteurs singuliers. Alors, le quaternion de rotation $\hat{q}$, solution de (3.7.11) est donné par le quatrième vecteur singulier, c'est-à-dire :

$$\hat{q} = V_4 \tag{3.7.14}$$

De plus, la condition $\|\hat{q}\|_2 = 1$ est naturellement satisfaite.

L'approche SVD est un outil élégant d'analyse théorique et une technique numériquement stable et robuste pour calculer le quaternion d'attitude à partir des vecteurs de mesures. Pour la détermination de l'attitude, le seul travail (à notre connaissance) utilisant la SVD est celui reporté dans [66]. La SVD est utilisée dans ce travail afin de trouver la matrice d'attitude en résolvant le problème de Whaba, mais les auteurs ne vont pas au bout de l'analyse. Cette technique est cependant très utilisée dans le domaine du traitement de signal comme dans [9].

En conclusion, les équations qui régissent l'observateur proposé sont donc :

1. estimation d'un quaternion q_{ps} à l'aide d'une décomposition par SVD à partir des mesures des accéléromètres et magnétomètres ;

2. intégration des mesures des gyromètres :

$$\dot{\hat{q}}(t) = \frac{1}{2}\Xi(\hat{q}(t))[\overrightarrow{\omega}_g(t) + K_1 \overrightarrow{q}_e] \tag{3.7.15}$$

$\hat{q}$ est le quaternion estimé, $K_1 \in \Re$ et $\overrightarrow{q}_e$ correspond à la partie vectorielle du quaternion d'erreur obtenu par un produit de quaternion.

3. Calcul du quaternion d'erreur :

$$q_e(t) = \hat{q}(t) \otimes \hat{q}_{ps}^{-1}(t) = [q_{e_0}, \overrightarrow{q}_e]^T \tag{3.7.16}$$

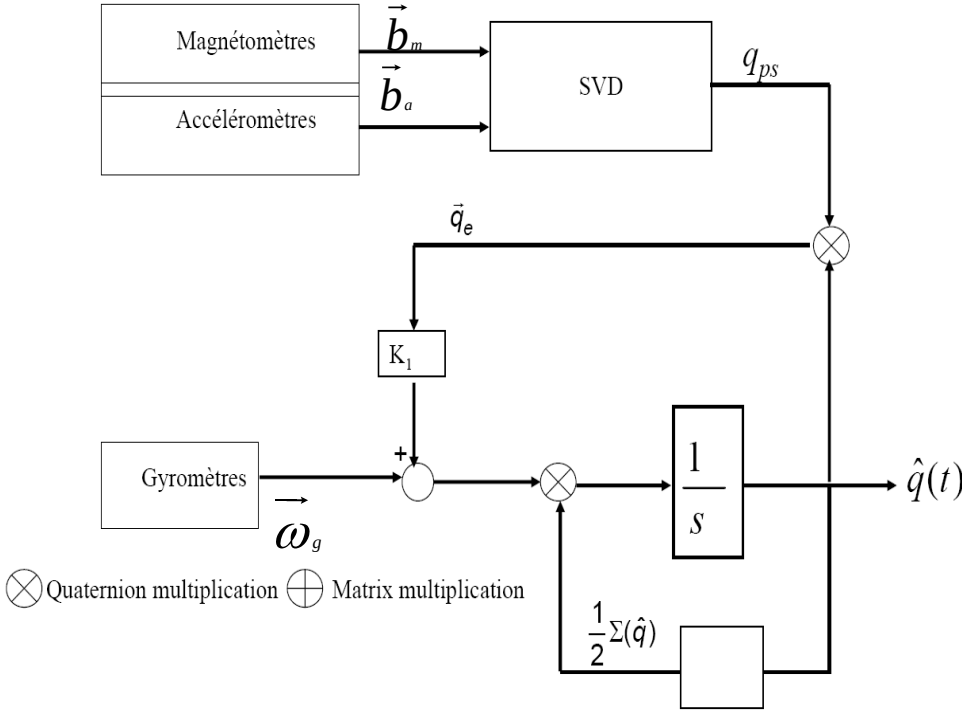


FIG. 3.9 – Observateur non-linéaire

3.8 Résultats de simulation

Dans cette section, plusieurs scénarios de simulation vont être considérés. L'objectif est d'étudier le comportement du système en boucle fermée (figure 3.10) pour une stabilisation en attitude.

Pour cela, à partir des mesures, l'observateur doit estimer une attitude exprimée en quaternion notée q_{est}. Cette attitude est ensuite comparée avec celle de référence notée q^{ref}. Le contrôleur implémenté a quant à lui pour objectif de fournir les quatre vitesses angulaires de référence notées ω_{Mi}^{ref} afin de stabiliser le système dans l'attitude

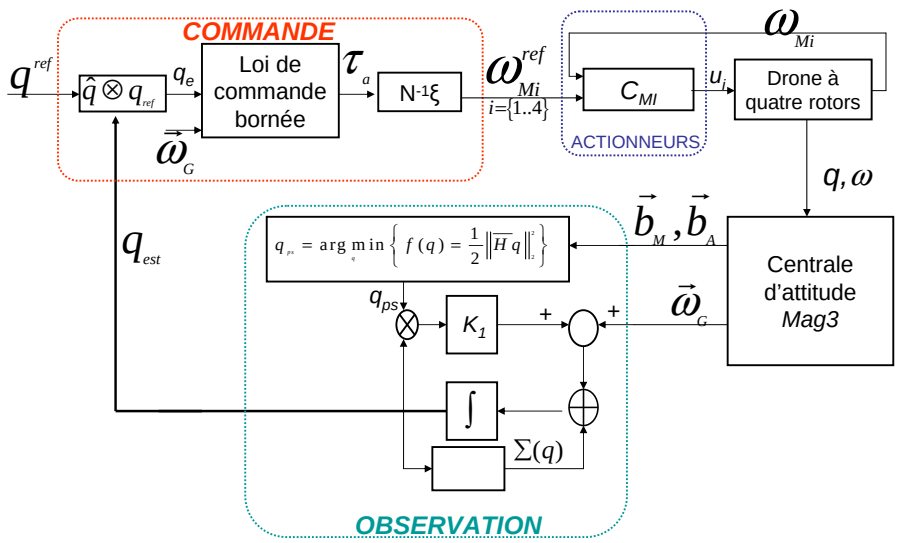


FIG. 3.10 – Système en boucle fermée

TAB. 3.2 – Valeur des paramètres du quadrotor [39]

I_r	moment d'inertie des rotors	3.357 e^{-5} $Jg.m^2$
d	distance entre rotor et centre de gravité	0.225 m
b	paramètre de la force de poussée	29.1 e^{-5}
k	paramètre de trainée	$1.14e^{-6}$
I_f	matrice d'inertie du quadrotor	$diag\{8.28, 8.28, 5.7\}e^{-3}kg.m^2$
m	masse du quadrotor	0.520 kg
$\overrightarrow{\eta}_a$	bruit blanc sur les accéléromètres	0.002 m/s^2
$\overrightarrow{\eta}_M$	bruit blanc sur les magnétomètres	0.0007 mgauss
$\overrightarrow{\eta}_G$	bruit blanc sur les gyromètres	0.01 rad/s

souhaitée. Ces quatre références sont ensuite comparées aux quatres vitesses angulaires à travers quatre boucles locales au niveau des actionneurs.

Un premier simulateur a été développé sous Matlab/Simulink [39]. Cependant, ce simulateur ne permettait que de simuler des stratégies de commande **ou** d'observations "off-line" avec une routine d'intégration à pas fixe. Dans ce travail, nous avons intégré sous Simulink les algorithmes de commande **et** d'observation pour pouvoir simuler le système en boucle fermée "on-line" avec une routine d'intégration à pas variable.

Remarque 8. Rappelons que tous les algorithmes fonctionnement sous le formalisme des quaternions. Cependant, pour une compréhension plus explicite des résultats, les attitudes sont représentées sous le formalisme des angles de Cardan.

Le tableau 3.2 présente les valeurs utilisées pour les simulations réalisées sous Mat-

lab/Simulink dont les résultats sont présentés ci-après.

3.8.1 Fonctionnement normal

Dans ce scénario, le quadrotor est initialisé avec l'attitude $\phi = -25°, \theta = -35°, \psi = -10°$ et doit être stabilisé à l'attitude $\phi = 0°, \theta = 0°, \psi = 0°$. L'observateur est initialisé avec l'attitude $\phi = 0°, \theta = 0°, \psi = 0°$. A l'instant $t = 5\ s$, la consigne en attitude change et est égale à $\phi = 20°, \theta = -10°, \psi = 5°$.
La figure 3.11 montre le résultat de simulation. La figure 3.11 A) montre les quatre

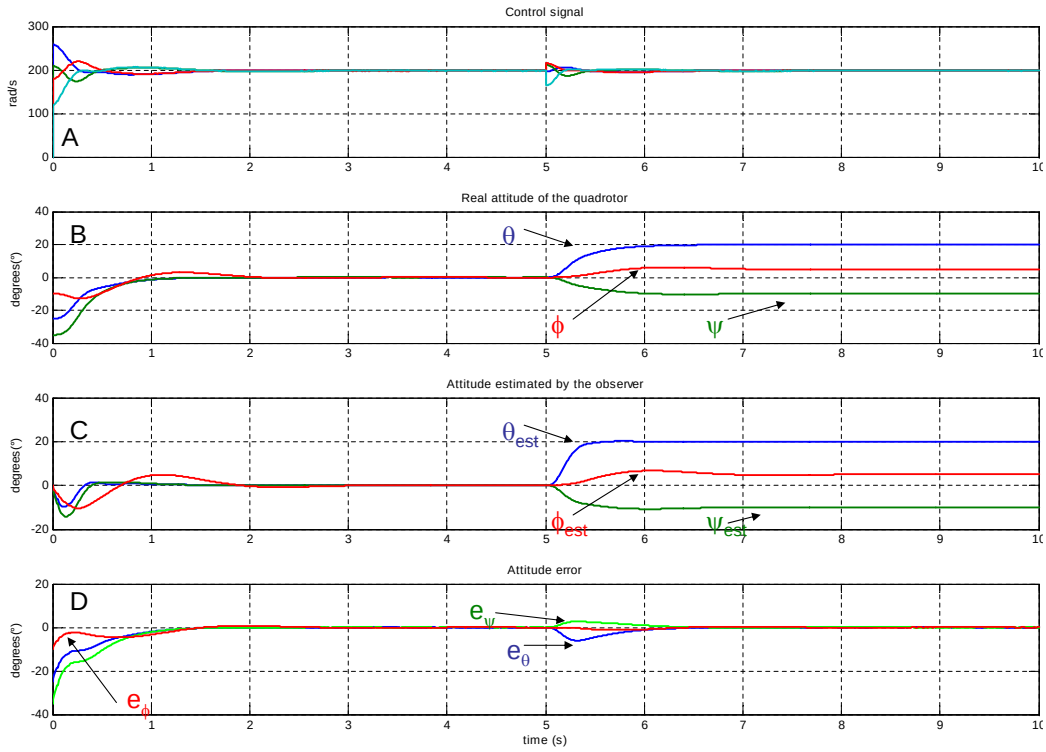


FIG. 3.11 – Fonctionnement normal

consignes de vitesses angulaires. La figure 3.11 B) montre l'attitude réelle[4] du quadrotor en termes d'angles d'Euler. Cette attitude se stabilise au point de fonctionnement souhaité en un temps de réponse d'environ 2 secondes. Après l'instant $t = 5\ s$, il y a un changement de consigne et le quadrotor parvient à son nouveau point de fonctionnement après environ 2 secondes. La figure 3.11 C) montre l'attitude estimée[5] par l'observateur non linéaire. La figure 3.11 D) montre l'erreur entre les attitudes réelle et estimée.

[4]Il est évidement qu'en réalité, les informations sur l'état du système sont pas disponibles mais obtenues par estimation. Cependant, nous sommes en simulation donc nous pouvons avoir accès à toutes les informations que l'on souhaite.

[5]Cette attitude serait la seule disponible avec une implémentation réelle.

On peut remarquer que le comportement est identique à celui de l'attitude réelle. Par conséquent, on peut considérer que l'observateur fonctionne correctement.

3.8.2 Fonctionnement en présence de perturbation

L'objectif ici est de stabiliser le quadrotor en zéro avec une attitude de départ égale à $\phi = -25°, \theta = -35°, \psi = -10°$. Ce scénario montre le comportement du système en

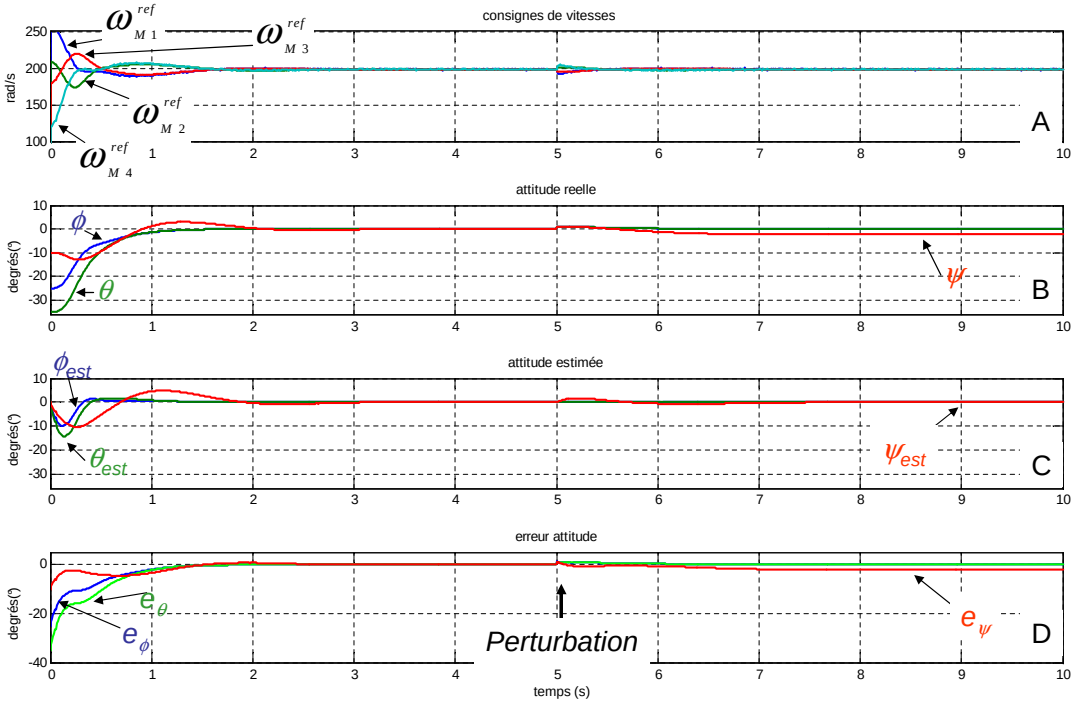


FIG. 3.12 – Fonctionnement avec une perturbation de 2 degrés sur chaque angle à partir de $t = 5\ s$

présence d'une perturbation de 2 degrés sur chaque angle du quadrotor (figure 3.8.2). Cette perturbation est introduite à partir de l'instant $t = 5\ s$. La figure 3.13 est un zoom à partir de l'introduction de la perturbation. On peut remarquer que l'attitude réelle du quadrotor est différente de zéro pour l'angle ψ.

3.8.3 Comportement en présence de défaut capteur

Dans ce scénario, le comportement du système est étudié en présence d'un défaut capteur sur l'accéléromètre d'axe x noté acc_X. Ce défaut est de type additif et apparaît à partir de l'instant $t = 5\ s$. Ce défaut est simulé (dans l'environnement Matlab/Simulink) par l'intermédiaire d'un échelon. Ce défaut est d'amplitude 0.5 g. Le quadrotor est initialisé dans l'attitude $\phi = -25°, \theta = -35°, \psi = -10°$ et doit être stabilisé à l'attitude

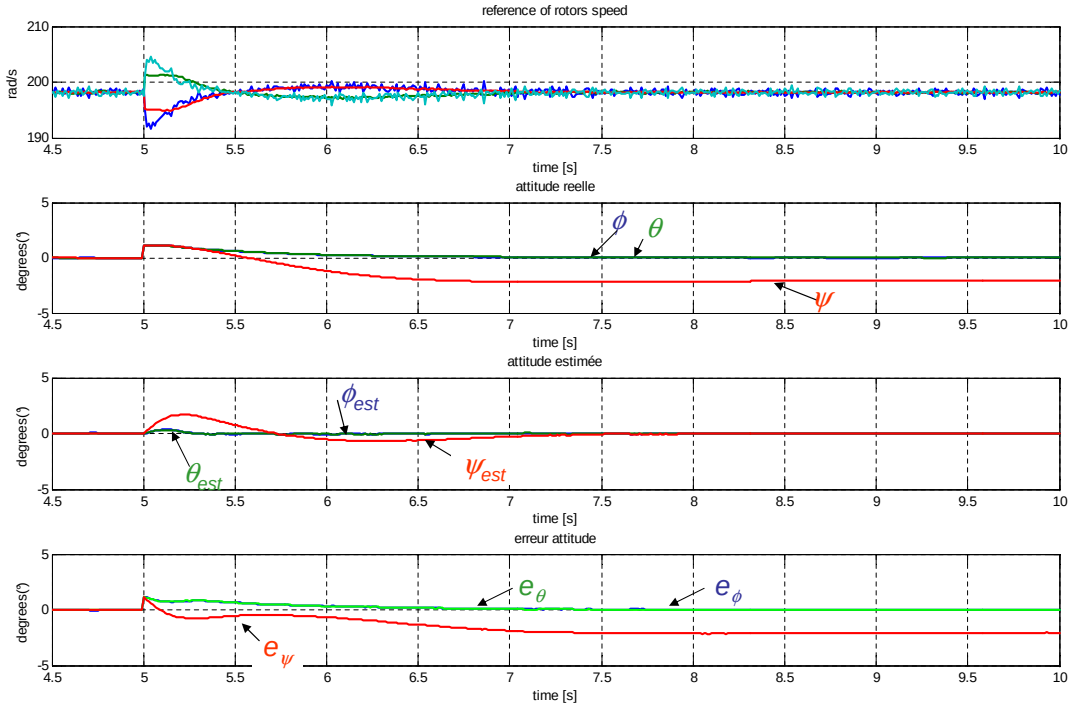


FIG. 3.13 – Zoom à partir de $t = 5\ s$

$\phi = 0°, \theta = 0°, \psi = 0°$. La figure 3.14 montre les résultats obtenus après simulation. Les

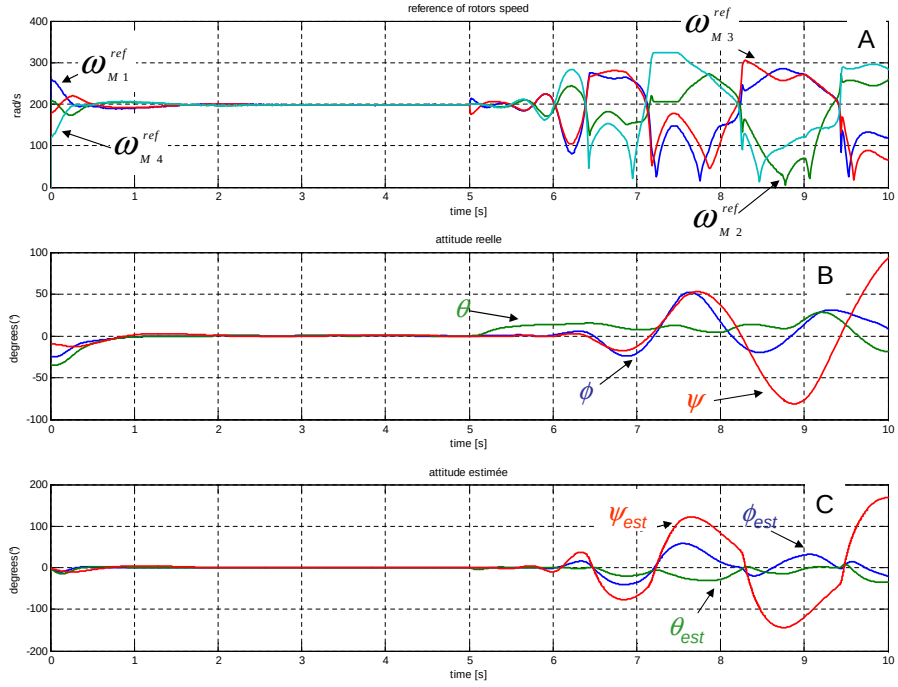


FIG. 3.14 – Fonctionnement avec un défaut capteur sur acc_X

figures 3.14 B) et C) montrent les attitudes réelle et estimée. On remarque que jusqu'à

l'instant $t = 5\ s$ (durant le fonctionnement nominal du système), le quadrotor est stabilisé. Cependant, après l'introduction du défaut, on peut constater que le quadrotor n'est plus stabilisé à l'attitude de référence et que le système devient instable. Dans la réalité, le quadrotor aurait subi un crash.

Cette simulation montre l'importance de pouvoir détecter ce type de défaut le plus tôt possible pour des questions de sécurité. On peut retrouver une étude complète des défauts pouvant survenir sur le quadrotor dans [84].

3.9 Conclusion

Dans ce chapitre, nous avons présenté le quadrotor utilisé dans le cadre de ce travail. Le système en boucle fermée ainsi que son comportement en mode de fonctionnement normal ont été étudiés. Nous avons vu que le quadrotor fonctionne correctement tant que nous ne sommes pas en présence de défaut. Un résultat de simulation a montré que si un défaut est introduit dans le modèle de la centrale d'attitude alors le système peut devenir instable. C'est pourquoi il est nécessaire de pouvoir diagnostiquer un éventuel défaut et le plus tôt possible. Différentes structures de diagnostic vont être étudiées dans le chapitre suivant.

Chapitre 4

Diagnostic

4.1 Rappels théoriques

Dans ce chapitre nous allons rappeler tout d'abord le principe du diagnostic en général dans la littérature pour le diagnostic des centrales d'attitudes présentées dans le chapitre précédent (voir section 2.2, page 32). Nous verrons des techniques existantes puis nous développerons deux algorithmes pour le diagnostic des centrales d'attitude. Le premier, *Algo 1*, nécéssite le modèle dynamique du système sur lequel la centrale est fixé. Le second, *Algo 2*, ne nécéssite pas le modèle du système sur lequel la centrale est fixée : il est donc générique.

4.1.1 Diagnostic : état de l'art

Pour un système en boucle fermée comme décrit par la figure 3.10, il est nécessaire de diagnostiquer les capteurs et actionneurs car l'influence d'un défaut peut être catastrophique comme montré avec la figure 3.14. Dans ce travail de thèse, nous nous sommes principalement focalisés sur les défauts capteurs car le diagnostic des actionneurs est moins difficile (voir section suivante) et s'il y a un souci, c'est tout le bloc comprenant le moteur et les pâles associées qu'il faut changer. Un défaut capteur peut être assimilé à une incohérence entre la valeur réelle y^* et la valeur mesurée y_k. Plusieurs scénarios pour un défaut capteur peuvent être considérés [36] :

- le blocage : dans ce scénario, à partir d'un instant t_f inconnu, la mesure reste bloquée à une certaine mesure :

$$y_k = const, \quad \forall k > t_f. \tag{4.1.1}$$

- la mise à zéro de la mesure : dans ce cas, à un instant donné, le capteur cesse de fonctionner et sa valeur de sortie est zéro après l'apparition du défaut. Ce défaut

est à classer dans la catégorie des défauts catastrophiques, et est généralement dû à des problèmes électriques ou de communication :

$$y_k = 0, \quad \forall k > t_f. \tag{4.1.2}$$

– la dérive ou défaut de type additif : ce type de défaut est le plus courant dans les capteurs analogiques, souvent produit par un changement de température interne ou à des problèmes de calibration. La valeur de sortie du capteur est la valeur théorique plus un terme additif.

$$y_k = y_k^* + \delta, \quad \forall k > t_f. \tag{4.1.3}$$

où δ peut être constant ou variant dans le temps (ex : dérive lente).

– un défaut de type multiplicatif : dans ce cas, un facteur multiplicatif modifie la valeur nominale.

$$y_k = \delta y_k^*, \quad \forall k > t_f. \tag{4.1.4}$$

Le rôle du diagnostic est de surveiller le comportement du système et d'apporter toutes les informations possibles lorsqu'un dysfonctionnement apparaît et le plus tôt possible. Pour arriver à ce résultat, la tâche complète du diagnostic d'un système peut être sous-divisée en trois "sous-tâches" comme présenté sur la figure 4.1 [19] :

– *détection du défaut*. L'objectif de cette tâche est d'avoir une indication sur le fait que quelque chose ne va pas bien dans le système ;
– *localisation du défaut*. Cette tâche a pour objectif de déterminer précisément le composant où est le défaut. Par exemple, dire sur quel capteur le défaut est intervenu ;
– *identification du défaut*. Grâce à cette tâche, la forme et l'amplitude du défaut sont connues.

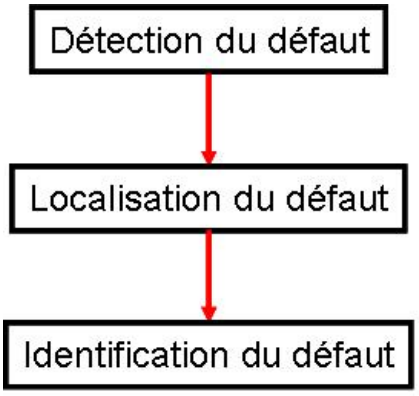


FIG. 4.1 – Les trois étapes de la tâche diagnostic

Il faut considérer les étapes suivantes pour pouvoir faire la détection [30] :

– *génération du vecteur de résidus*. Il faut concevoir des signaux qui reflètent les fautes. Ces signaux sont généralement regroupés dans un vecteur de résidus. La génération du vecteur de résidus se fait en général en temps réel ;

– *évaluation du résidu.* Cette étape est aussi appelée prise de décision. Chaque composante du vecteur du résidus est analysée pour décider s'il y a ou non présence d'un défaut. Cette prise de décision peut être réalisée à l'aide d'un simple test de dépassement de seuil [103] sur les valeurs instantanées ou faire appel à la théorie de la décision statistique [4], [3], ou bien encore de la décision floue. On génère aussi un symptôme. Par exemple, dans le cas de dépassement de seuil, si le résidu est supérieur au seuil alors le symptôme est égal à 1 sinon il est égal à 0.

Une fois ces étapes terminées, la dernière consiste à reconfigurer la tâche de commande. Cette étape consiste à changer la loi de commande afin de continuer à garantir les performances du système ou à les dégrader le moins possible en présence de défaut. Cette étape est appelée Commande tolérante aux défauts *(FTC)*.

Un système est dit tolérant aux défauts si ce système a la capacité de maintenir les objectifs nominaux même en présence de défauts. Il doit permettre également de garantir la stabilité du système et aussi de garantir des performances dégradées acceptables en présence de défauts. La commande tolérante aux défauts est divisée en deux grandes approches :

– l'approche dite passive ;
– l'approche dite active.

La figure 4.2 présente un schéma de système de commande tolérante aux défauts [73]. Pour l'approche passive, les régulateurs sont synthétisés afin d'être robuste à certains

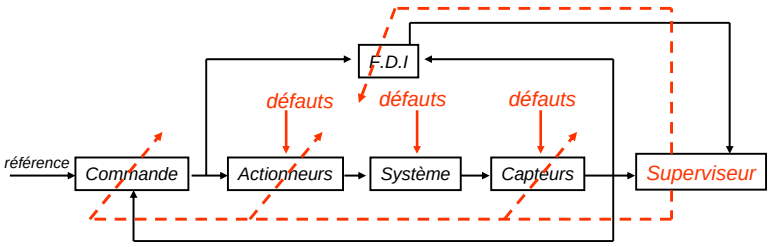


FIG. 4.2 – Principe d'un système de commande tolérante aux défauts

défauts. L'idée principale est de faire en sorte que le système en boucle fermée soit robuste aux incertitudes et à quelques défauts spécifiques. Le point faible de cette approche est sa capacité de tolérance aux défauts qui reste restreinte à quelques défauts uniquement.

L"approche active doit quant à elle réagir aux diverses défaillances du système. Pour cela, l'approche active doit reconfigurer les lois de commande tout en préservant la stabilité et les performances de celui-ci.

La commande tolérante aux défauts est une des plus importantes directions de recherche explorée ces dernières années. On peut citer entre autres [10],[78].

TAB. 4.1 – Table de signature pour l'étape de *localisation*

	f_1	f_2
r_1	1	0
r_2	0	1

Si la tâche de génération du résidu est bien faite, alors la détection devient une tâche aisée. Sans la détection de défaut, il est impossible de faire la tâche localisation et par conséquent, impossible d'identifier le défaut. C'est pourquoi, beaucoup d'efforts se sont concentrés sur la génération du résidu [104].

De nombreuses méthodes, permettant la génération de résidus en utilisant des modèles analytiques, ont été proposées par les automaticiens [19],[60]. Ces approches sont généralement classées en trois catégories [36],[54] :

- observateur [74][92][55] ;
- espace de parité [37],[64] ;
- estimation paramétrique [53].

Toutes ces approches utilisent le modèle mathématique pour générer le résidu. Dans le cas le plus simple, le résidu est obtenu comme la différence entre les sorties du système et celles estimées avec le modèle par l'équation suivante :

$$r_k = y_k - \hat{y}_k \tag{4.1.5}$$

$\hat{y}_k$ peut être obtenu par simulation du modèle à partir des entrées du système mesurées ou par prédiction à partir d'entrés/sorties. En théorie, ce résidu est égal à zéro en présence d'un système sain, et différent de zéro lorsqu'un système est en présence d'un défaut :

$$\begin{aligned} r_k = 0 &\Rightarrow syst\grave{e}me \;\; sain \\ r_k \neq 0 &\Longrightarrow syst\grave{e}me \;\; en \;\; d\acute{e}faut \end{aligned} \tag{4.1.6}$$

Une fois le vecteur de résidus généré, celui-ci est évalué c'est-à-dire transformé en *symptôme*. Dès que l'on a évalué qu'un résidu r est différent de zéro, on a détecté l'apparition d'un défaut. l'étape de *localisation* est réalisée à l'aide de plusieurs résidus. En effet, les symptômes sont ensuite comparés avec une table appelée *table de signature* comme montrée par le tableau 4.1. Les colonnes de cette table représentent les défauts et les lignes représentent les résidus. Le "1" correspond au cas où le résidu est sensible à ce défaut et le "0" correspond au fait que le résidu est insensible à ce défaut.
Par exemple, pour la table 4.1, le résidu r_1 est sensible à la faute f_1 mais ne l'est pas à la faute f_2. Inversement pour le résidu r_2. De plus, si les colonnes de la table sont différentes entre elles, alors le défaut peut être localisé. Inversement, si deux colonnes sont identiques, alors ces deux défauts ne sont pas différentiables.

Avec ces approches, les résidus obtenus sont cependant sensible aux incertitudes de modèle, car le modèle du système ne fournit qu'une approximation du comportement

réel. Les perturbations aussi bien que les incertitudes de modèle sont inévitables dans les systèmes industriels. C'est pourquoi le système de diagnostic doit être robuste. Le problème est donc de développer des résidus qui doivent être insensibles aux erreurs de modèle et surtout aux réelles perturbations mais sensibles aux défauts.

Le nombre de problèmes augmente lorsqu'il est question de la détection des systèmes non-linéaires [104] à cause de la difficulté d'obtenir un modèle précis et d'utiliser certaines techniques comme par exemple les observateurs (erreurs de modèle, etc...).

Le processus de localisation nécessite une table de signature localisante. On trouve dans la littérature deux schémas. Ces schémas sont appelés schéma dédié ou généralisé [19],[54]. Dans le cas du schéma dédié, la génération du résidu est conçue de telle façon que chaque résidu r_i, $i = 1, ..., N$ est sensible à un défaut seulement et reste insensible aux autres. La localisation est alors donnée par :

$$|r_{i,k}| > T_i \Rightarrow \quad f_{i,k} \neq 0, \quad i = 1....N, \tag{4.1.7}$$

où T_i est un seuil prédéfini. Cette procédure de localisation est très restrictive car elle est sensible aux incertitudes du modèle. Les stratégies de localisation basées sur le principe des schémas dédiés sont couramment utilisées dans les schémas de FDI basés sur les réseaux neurones [60]. Pour le second cas, la génération des résidus basée sur l'approche généralisée est telle que chaque résidu r_i, $i = 1, ..., N$ soit sensible à tous les défauts sauf un. Dans ce cas, il faut trouver la valeur de plusieurs seuils [104] :

$$\begin{aligned} |r_{i,k}| < \quad & T_i \\ |r_{j,k}| > \quad & T_j, j = 1, ..i-1, i+1, ..., N \\ & \Rightarrow f_{i,k} \neq 0, i = 1....N, \end{aligned} \tag{4.1.8}$$

Mais l'avantage de cette approche est qu'elle est moins restrictive que le schéma dédié. La figure 4.3 montre ces deux approches à travers un banc de N observateurs surveillant un système à N sorties. Pour les défauts capteurs, dans le cas du schéma généralisé, on prend toutes les entrées et (N-1) sorties pour chaque observateur. Dans le cas du schéma dédié, on ne prend qu'une seule sortie.

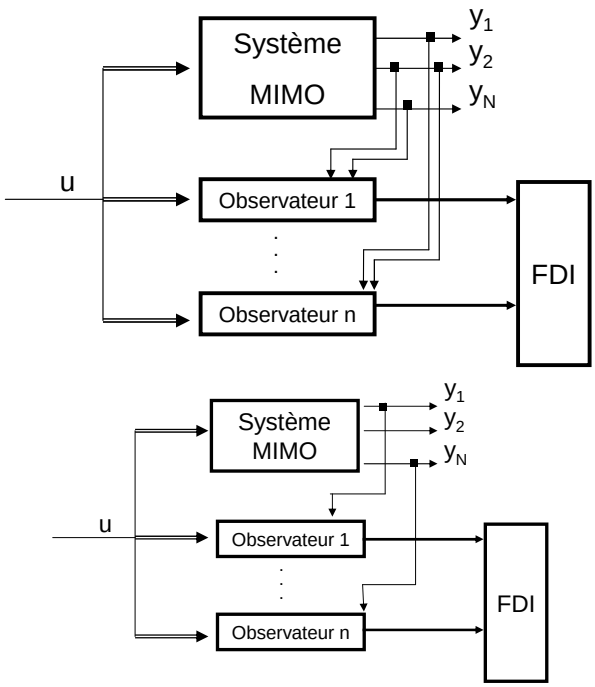


FIG. 4.3 – Génération du vecteur de résidus : en haut schéma généralisé, en bas schéma dédié

4.1.2 Diagnostic des centrales d'attitude

Précédemment, nous avons vu différentes approches pour le diagnostic. Maintenant, nous allons nous concentrer sur le diagnostic des centrales d'attitude.

Dans [42], nous pouvons retrouver une étude sur les différentes approches que l'on peut rencontrer sur le diagnostic et la reconfiguration en cas de défaut dans le domaine de l'aéronautique et du spatial.

4.1.2.1 Approche par espace de parité

La première approche que l'on peut rencontrer est celle de l'espace de parité. Cette approche est très utilisée dans l'aérospatiale pour diagnostiquer les centrales d'attitudes décrites dans 2.2. Par exemple, dans [77] des équations de parité (la conception de la structure de l'espace de parité s'établit à partir des équations du système) sont utilisées afin de détecter et de localiser des défauts sur les gyromètres et accéléromètres. Pour cela, 7 équations ont été générées et utilisées pour la détection et la localisation d'un défaut simple sur un axe uniquement. Cependant, dans cette étude, le composant utilisé est uniquement composé de gyromètres et accéléromètres. De plus, le défaut n'est pas identifié.

4.1.2.2 Approche par réseaux neurones

Dans [69] et [70], une structure de diagnostic par réseaux de neurones est présentée. Dans ces articles, seul le cas des gyromètres est présenté. Les auteurs ont proposés une structure de diagnostic à partir de l'erreur de l'estimation quadratique entre les mesures des gyromètres et leurs estimées. L'idée est que si cette erreur dépasse un certain seuil, alors nous sommes en présence d'un défaut.

4.1.2.3 Approche par observateur

Une structure de banc d'observateurs est utilisée dans [75] pour détecter les défauts d'une centrale d'attitude du vaisseau spatial MARS Express. Dans [43] une structure de banc d'observateurs de type dédié (en prenant en compte toutes les entrées et une seule sortie) est utilisée pour détecter et localiser les défauts capteurs sur un hélicoptère instrumenté par 12 capteurs (3 accéléromètres, 3 magnétomètres et 3 gyromètres, plus un GPS). De cette façon, un résidu est généré pour chaque capteur, grâce à la différence entre la mesure et son estimée. Grâce à cette technique, chaque résidu n'est pas affecté par les défauts sur les autres capteurs. Ceci permet de localiser plus facilement le défaut car chaque résidu généré est sensible à un seul capteur. Les auteurs ont montré la limitation d'une telle structure car il était difficile de détecter un défaut additif de type biais avec une amplitude de 10%. Dans [108] une étude est réalisée autour de la commande du vol avec des défauts sur les gyromètres. Cette étude a été étendue dans [85] à tous les capteurs (accéléromètres, magnétomètres en plus des gyromètres). Cependant, avec toutes ces méthodes, on ne peut pas identifier les défauts car le diagnostic repose sur le modèle mécanique du système.

Deux approches sont maintenant présentées pour détecter les défauts d'une centrale d'attitude. La table 4.2 définit les notations utilisées dans ce chapitre. Rappelons que le vecteur de mesure est défini par :

$$y_M = \begin{bmatrix} \overrightarrow{b}_a \\ \overrightarrow{b}_m \\ \overrightarrow{\omega}_g \end{bmatrix} = \begin{bmatrix} acc_X \\ acc_Y \\ acc_Z \\ mag_X \\ mag_Y \\ mag_Z \\ gyro_X \\ gyro_Y \\ gyro_Z \end{bmatrix} \tag{4.1.9}$$

TAB. 4.2 – Définition des symboles utilisés

FDI	sigle pour définir le processus de détection et de localisation d'un défaut. FDI pour *Fault Detection and Isolation*.
FTC	sigle pour la réalisation de commande tolérante aux défauts. FTC pour *Fault Tolerant Control*.
q_{model}	attitude obtenue grâce au modèle mécanique du système (on peut l'assimiler à une attitude théorique).
$\hat{q}_i$	attitude estimée à partir d'un sous-ensemble de mesures $i = 1..9$.
r_i	résidu sous forme scalaire. $i = 1..9$ pour le cas de *Algo 1* et $i = 1..36$ pour le cas de *Algo 2*.
R_j	vecteur constitué de 6 résidus r_i, avec $j = 1..6$. Symbole utilisé par *Algo 2*.
ω_i	valeur d'une composante de la vitesse angulaire calculée par le modèle mécanique. Utilisé par *Algo 1*.
$\overrightarrow{b}_m, \overrightarrow{b}_a$	mesures issues des magnétomètres et accéléromètres.
$\overrightarrow{\omega}_g$	vitesse angulaire mesurée du quadrotor.
Y_M	vecteur de mesure constitué par $\overrightarrow{b}_a, \overrightarrow{b}_m, \overrightarrow{\omega}_g$.
q_e^i	quaternion d'erreur entre $\hat{q}_i$ et q_{model}, i=1..9. Utilisé par *Algo 1*.
$\hat{q}_{sain}$	quaternion disponible après l'étape d'optimisation des magnétomètres et accéléromètres. Utilisé par *Algo 2*.
$\overrightarrow{\hat{b}}_{acc}, \overrightarrow{\hat{b}}_{mag}$	estimation des mesures des magnétomètres et accéléromètres. Utilisé par *Algo 2*.
Y_{model}	vecteur constitué de $\overrightarrow{\hat{b}}_{acc}, \overrightarrow{\hat{b}}_{mag}$. Utilisé par *Algo 2*.
$r_\omega(i)$	résidus obtenus pour la surveillance des gyromètres. Utilisé par *Algo 2*.
q_{filtre}	quaternion obtenu après filtrage. Utilisé dans l'algo2.
$\hat{\omega}_i$	estimation de la vitesse angulaire avec $i = 1..3$. Utilisé par *Algo 2*.
y_k	valeur réelle de sortie du système.
y_k^*	valeur mesurée de sortie du système.
f_{ax}	défaut survenant sur le capteur accéléromètre X.
f_{ay}	défaut survenant sur le capteur accéléromètre Y.
f_{az}	défaut survenant sur le capteur accéléromètre Z.
f_{mx}	défaut survenant sur le capteur magnétomètre X.
f_{my}	défaut survenant sur le capteur magnétomètre Y.
f_{mz}	défaut survenant sur le capteur magnétomètre Z.
f_{gx}	défaut survenant sur le gyromètre X.
f_{gy}	défaut survenant sur le gyromètre Y.
f_{gz}	défaut survenant sur gyromètre Z.

4.2 Diagnostic du quadrotor : *Algo 1*

Dans le paragraphe 3.8.3, page 60, l'influence d'un défaut capteur sur le système en boucle fermée a été montrée. Cette influence n'étant pas négligeable, il est important de diagnostiquer les défauts. Dans cette section, un premier algorithme noté *Algo 1*, va être présenté.

Comme vu dans la section 4.1, le diagnostic est composé de plusieurs étapes pour

pouvoir détecter et localiser un défaut. L'idée ici est d'implémenter une structure de banc d'observateurs de type généralisé basée sur l'observateur utilisé pour l'estimation de l'attitude (voir section 3.7). La différence entre le signal mesuré et son estimation est utilisée comme résidu $r_k = y_k - \hat{y}_k$.

Pour résoudre ce problème, beaucoup d'observateurs peuvent être employés comme l'observateur de Luenberger pour les système linéaires ou bien le filtre de Kalman [1]. Cette méthode a été utilisée dans [86] et [85] puis reprise et adaptée dans [16] pour tenir compte de l'aspect "Networked Control Systems" (ce qui sera décrit ultérieurement dans le chapitre réseau). La génération de vecteurs va être présentée en deux parties :

- tout d'abord l'aspect génération des résidus pour la surveillance des capteurs gyromètres. Cette partie permet de prendre en compte l'aspect dynamique du système ;
- ensuite, la génération des résidus sur les accéléromètres et magnétomètres sera présentée. Cette partie ne tient pas compte de la dynamique du système, car ces mesures sont supposées être en quasi-statiques.

4.2.1 Génération des résidus pour la surveillance des gyromètres avec *Algo 1*

Pour diagnostiquer d'éventuels défauts pouvant apparaître sur les gyromètres, la technique de banc d'observateurs est utilisée. La centrale fournit les mesures des trois gyromètres. Par conséquent, trois observateurs seront nécessaires pour pouvoir localiser un des trois défauts. La structure pour la génération des résidus gyromètres est schématisée par la figure 4.4. Si l'on se réfère à la figure 4.4, pour chaque observateur, une mesure gyromètre ne sera pas utilisée. Pour palier cette absence, une valeur calculée avec le modèle mécanique du quadrotor (équation 3.5.4) va être utilisée. Notons que cette méthode est forcément sensible aux erreurs de modélisation.

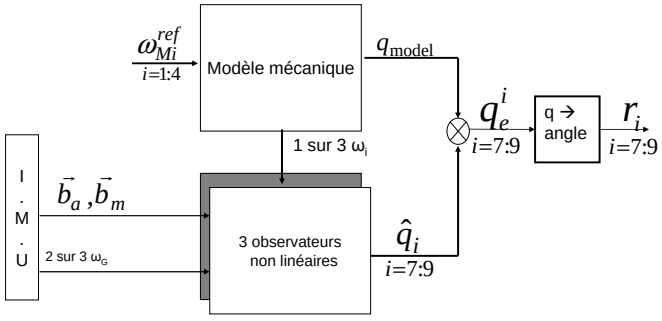


FIG. 4.4 – Génération des résidus pour les trois gyromètres avec *Algo 1*

L'observateur utilisé est le même que pour la commande (voir description de ce dernier dans le paragraphe 3.7.2) mais en l'adaptant pour chaque observateur du banc. Chaque quaternion obtenu, noté $\hat{q}_i$, $i = 7 : 9$, (avec 8/9 mesures + 1 valeur de vitesse angulaire

TAB. 4.3 – Table de signature pour détecter les défauts sur les gyromètres avec *Algo 1.*

	f_{ax}	f_{ay}	f_{az}	f_{mx}	f_{my}	f_{mz}	f_{gx}	f_{gy}	f_{gz}
r_7	1	1	1	1	1	1	0	1	1
r_8	1	1	1	1	1	1	1	0	1
r_9	1	1	1	1	1	1	1	1	0

reconstituée par le modèle) est ensuite comparé avec l'attitude obtenue grâce au modèle mécanique, notée q_{model}. Cette comparaison donne un quaternion d'erreur entre les deux estimations. Ce quaternion est noté q_e^i, $i = 7 : 9$. Chaque observateur est sensible aux défauts pouvant apparaître sur tous les capteurs excepté celui écarté (principe des bancs d'observateurs généralisés). Le résidu est généré à partir du quaternion d'erreur grâce à cette équation initialement définie par (2.1.16) page 28 :

$$r_i = \varphi_e(i) = 2 * arccos(q_e^i(0)), \quad i = 7 : 9 \tag{4.2.1}$$

On obtient finalement la table de signature donnée par la table 4.3. On peut remarquer que les 6 premières colonnes de cette table sont identiques. Par conséquent un défaut sur un accéléromètre ou magnétomètre n'est pas localisable avec ces trois résidus (car les trois accéléromètres et les trois magnétomètres sont utilisés pour la génération des trois résidus). Cependant, les trois dernières colonnes sont différentes. Donc un défaut sur un des gyromètres est localisable.

4.2.2 Génération des résidus pour la surveillance des accéléromètres et magnétomètres avec *Algo 1*

Pour diagnostiquer les défauts sur les accéléromètres et magnétomètres tout en gardant le principe de l'observateur utilisé pour la commande, il faudrait utiliser l'équation (3.7.8) $Hq = 0$; où H est une fonction des mesures des accéléromètres et magnétomètres qui génère q_{ps}. En fait il est impossible de reconstruire la matrice H si l'on écarte une mesure d'accéléromètre ou bien de magnétomètre car nous perdrions plusieurs lignes de la matrice et par conséquent il n'y aurait plus de solution. On peut noter que cette relation est statique.

L'idée pour pouvoir générer le quaternion q_{ps} serait d'utiliser des techniques d'optimisation non linéaire ([44], [61]). Pour le diagnostic, seulement 5 sur 6 mesures produites par les magnétomètres et accéléromètres sont utilisées pour obtenir un quaternion $\hat{q}_i$ par optimisation. La figure 4.5 montre le principe pour générer les résidus afin de détecter les défauts sur les accéléromètres et magnétomètres. Le principe ici est de développer six estimateurs pour détecter et localiser le défaut. Chaque résidu est sensible à tous les défauts sauf ceux apparaissant sur la mesure écartée. Par exemple, le résidu r_1 est calculé en prenant en compte toutes les mesures des accéléromètres et magnétomètres

TAB. 4.4 – Table de signature pour détecter les défauts sur les accéléromètres et magnétomètres avec *Algo 1*

	f_{ax}	f_{ay}	f_{az}	f_{mx}	f_{my}	f_{mz}	f_{gx}	f_{gy}	f_{gz}
r_1	0	1	1	1	1	1	0	0	0
r_2	1	0	1	1	1	1	0	0	0
r_3	1	1	0	1	1	1	0	0	0
r_4	1	1	1	0	1	1	0	0	0
r_5	1	1	1	1	0	1	0	0	0
r_6	1	1	1	1	1	0	0	0	0

exceptée celle produite par l'accéléromètre d'axe X. Par conséquent, l'estimateur r_1 est insensible uniquement aux défauts survenus sur acc_X. La table de signature est donnée par la table 4.4 Le problème d'optimisation consiste à trouver un quaternion $q =$

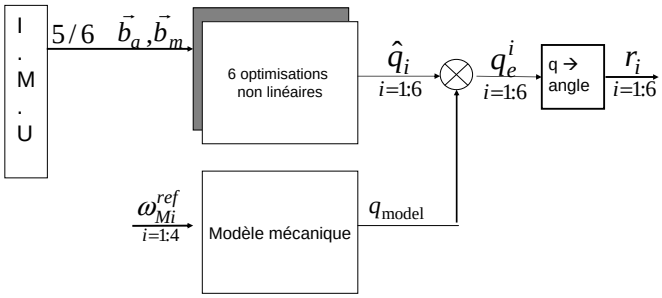


FIG. 4.5 – Génération des résidus pour les accéléromètres et magnétomètres avec *Algo 1*

$[q_0, q_1, q_2, q_3]^T$ minimisant la fonction *f(q̂)* suivante :

$$f(\hat{q}_i) = \frac{1}{2}\sum_{j=1}^{6}(q^T A_j q - Y_M(j))^2, \quad j \neq i \tag{4.2.2}$$

où $Y_M(j)$, $j = 1:6$, sont les mesures des accéléromètres et magnétomètres, $j \neq i$ pour l'estimateur i (cet estimateur n'utilise pas la mesure i). $q^T A_j q$ modélise la mesure. Dans le repère inertiel, les mesures des accéléromètres et magnétomètres sont définies par :

$$g_0 = [g_0(1), g_0(2), g_0(3)]^T \quad b_0 = [b_0(1), b_0(2), b_0(3)]^T \tag{4.2.3}$$

Dans notre application, on prendra :

$$g_0 = [0, 0, 1]^T \quad b_0 = [\frac{1}{2}, 0, \frac{\sqrt{3}}{2}]^T \tag{4.2.4}$$

Les matrices A_j s'écrivent :

$$A_1 = \begin{pmatrix} g_0(1) & 0 & -g_0(3) & g_0(2) \\ 0 & g_0(1) & g_0(2) & g_0(3) \\ -g_0(3) & g_0(2) & -g_0(1) & 0 \\ g_0(2) & g_0(3) & 0 & -g_0(1) \end{pmatrix} \tag{4.2.5}$$

$$A_2 = \begin{pmatrix} g_0(2) & g_0(3) & 0 & -g_0(1) \\ g_0(3) & -g_0(2) & g_0(1) & 0 \\ 0 & g_0(1) & g_0(2) & g_0(3) \\ -g_0(1) & 0 & g_0(3) & -g_0(2) \end{pmatrix} \tag{4.2.6}$$

$$A_3 = \begin{pmatrix} g_0(3) & -g_0(2) & g_0(1) & 0 \\ -g_0(2) & -g_0(3) & 0 & g_0(1) \\ g_0(1) & 0 & -g_0(3) & g_0(2) \\ 0 & g_0(1) & g_0(2) & g_0(3) \end{pmatrix} \tag{4.2.7}$$

$$A_4 = \begin{pmatrix} b_0(1) & 0 & -b_0(3) & b_0(2) \\ 0 & b_0(1) & b_0(2) & b_0(3) \\ -b_0(3) & b_0(2) & -b_0(1) & 0 \\ b_0(2) & b_0(3) & 0 & -b_0(1) \end{pmatrix} \tag{4.2.8}$$

$$A_5 = \begin{pmatrix} b_0(2) & b_0(3) & 0 & -b_0(1) \\ b_0(3) & -b_0(2) & b_0(1) & 0 \\ 0 & b_0(1) & b_0(2) & b_0(3) \\ -b_0(1) & 0 & b_0(3) & -b_0(2) \end{pmatrix} \tag{4.2.9}$$

$$A_6 = \begin{pmatrix} b_0(3) & -b_0(2) & b_0(1) & 0 \\ -b_0(2) & -b_0(3) & 0 & b_0(1) \\ b_0(1) & 0 & -b_0(3) & b_0(2) \\ 0 & b_0(1) & b_0(2) & b_0(3) \end{pmatrix} \tag{4.2.10}$$

On peut estimer le quaternion à partir de cinq mesures car nous travaillons sous l'hypothèse que le mouvement est quasi statique[1]. Comme le montre la figure 4.5, ce

[1]En effet, si le problème n'était pas quasi-statique, il faudrait prendre en compte l'accélération propre du système (fournie par les accéléromètres). Dans ce cas, le problème d'optimisation consisterait à estimer le quaternion mais aussi les trois accélérations (en x, y, z).

quaternion, noté $\hat{q}_i$, i=1 :6, est ensuite comparé avec celui obtenu par le modèle mécanique (q_{model}). Le quaternion d'erreur q_e^i, $i = 1 : 6$, (entre le quaternion estimé par optimisation et le quaternion issu de l'équation mécanique) est lié à la variable φ_e^i qui sera utilisée comme le résidu, $i = 1 : 6$:

$$\begin{aligned} q_e^i &= (q_{model}^{-1} \otimes \hat{q}_i) \quad i = 1:6 \\ q_e^i &= (cos(\tfrac{\varphi_e^i}{2}), sin(\tfrac{\varphi_e^i}{2}\overrightarrow{u}_e^T(i))^T \\ r_i &= 2arccos(q_e^i(0)), \quad i = 1:6 \end{aligned} \tag{4.2.11}$$

avec $\overrightarrow{u}_e$ l'axe de rotation du quaternion d'erreur.

En conclusion, afin de pouvoir diagnostiquer d'éventuels défauts sur les neuf mesures produites par la centrale d'attitude, trois observateurs non linéaires et six estimateurs sur la base d'optimisation non linéaire fonctionnent en parallèle comme l'indique la figure 4.6. La table 4.5 montre la table de signature complète pour la localisation des défauts capteurs.

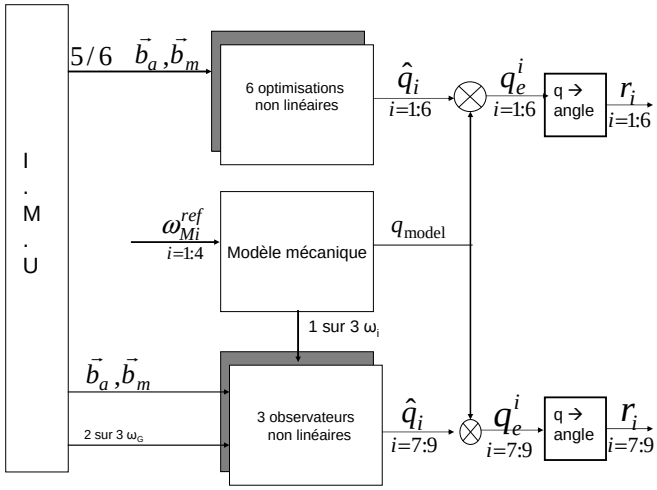


FIG. 4.6 – Génération des résidus pour la centrale d'attitude

Une fois le défaut détecté et localisé, la tâche de commande pourrait être reconfigurée afin de continuer à garantir les spécifications souhaitées. Dans [14], on peut trouver une proposition de commande tolérante aux défauts (FTC) pour ce système.

4.3 Diagnostic du quadrotor : *Algo 2*

Dans les sections précédentes, le principe du diagnostic a été introduit et nous avons vu une technique pour diagnostiquer les défauts capteurs. Cependant, avec cette technique, nous sommes dépendant du modèle mécanique et l'étape d'identification ne peut

TAB. 4.5 – Table de signature complète pour la localisation des défauts sur la centrale d'attitude avec *Algo 1*.

	f_{ax}	f_{ay}	f_{az}	f_{mx}	f_{my}	f_{mz}	f_{gx}	f_{gy}	f_{gz}
r_1	0	1	1	1	1	1	0	0	0
r_2	1	0	1	1	1	1	0	0	0
r_3	1	1	0	1	1	1	0	0	0
r_4	1	1	1	0	1	1	0	0	0
r_5	1	1	1	1	0	1	0	0	0
r_6	1	1	1	1	1	0	0	0	0
r_7	1	1	1	1	1	1	0	1	1
r_8	1	1	1	1	1	1	1	0	1
r_9	1	1	1	1	1	1	1	1	0

pas être faite. Pour ces raisons, nous allons maintenant présenter une structure de diagnostic indépendante du modèle mécanique du drone. Cette structure devra aussi avoir la capacité d'identifier les défauts.

4.3.1 Diagnostic des accéléromètres et des magnétomètres avec *Algo 2*

Le diagnostic de ces capteurs est basé sur l'approche d'optimisation définie dans l'annexe. La figure 4.7 montre le schéma de cette approche pour la surveillance d'un accéléromètre ou d'un magnétomètre. L'idée de base est reprise de [86] et [85]. La premier étape consiste à estimer le quaternion comme pour l'algo 1. Pour cela, il faut résoudre le problème de minimisation du critère défini par l'équation (4.2.2). Cette optimisation est réalisée sous Matlab par la fonction fmincon [105].

A partir du résultat de l'optimisation $\widehat{q}_j$, les mesures des accéléromètres et magnétomètres sont reconstruites à travers le modele suivant :

$$Y_{model_j} = q^T A_j q \tag{4.3.1}$$

où A_j ont été définies par les équations (4.2.5) à (4.2.10). Le résidu est généré comme la différence entre les mesures et leur reconstruction. Pour chaque mesure écartée, un vecteur de résidu, noté R_j, de dimension 6 est donc généré. Comme la centrale d'attitude dispose de 6 mesures, un ensemble de 6 vecteurs sera crée. Au final, un ensemble de 36 résidus sous forme scalaire répartis en 6 vecteurs (noté R_i, $i = 1 : 6$) de 6 résidus est créé, chacun étant sensible différemment aux défauts. La figure 4.7 montre la structure globale pour la génération des résidus pour la surveillance des accéléromètres et magnétomètres.

La table de signature est donnée par la table 4.6. Concernant les notations, R_i, $i =$

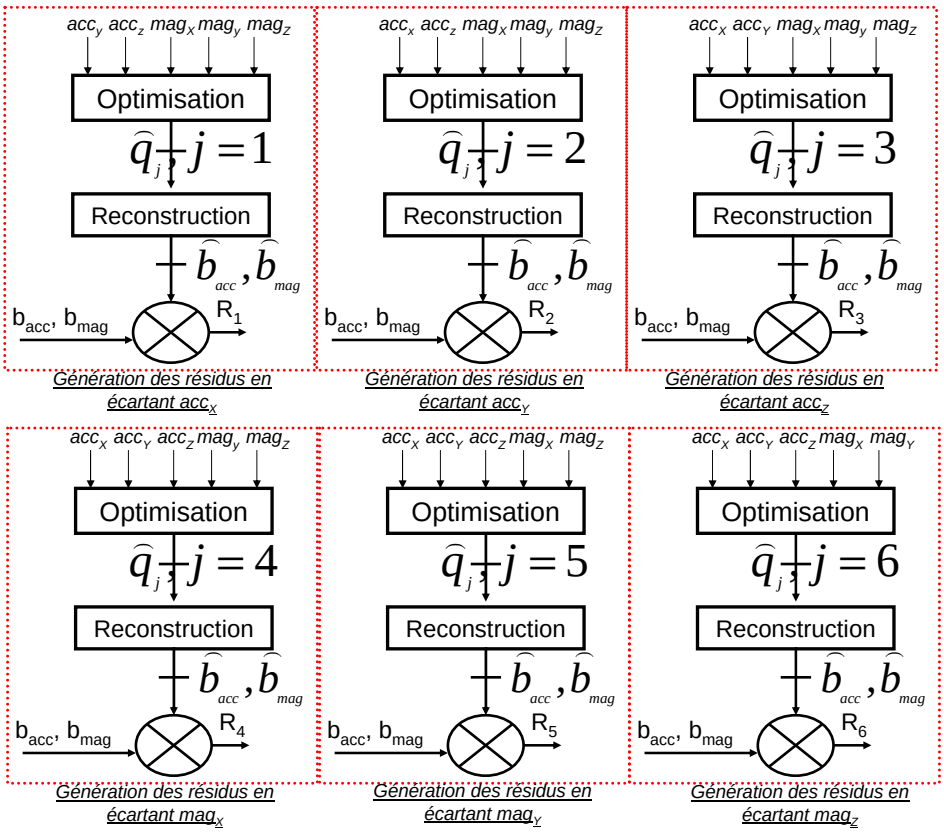


FIG. 4.7 – Génération des résidus pour la surveillance des accéléromètres et magnétomètres avec *Algo 2*

TAB. 4.6 – Table de signature pour les accéléromètres et magnétomètres avec *Algo 2*

	f_{ax}	f_{ay}	f_{az}	f_{mx}	f_{my}	f_{mz}
R_1	**[100000]**	[010000]	[001000]	[000100]	[000010]	[000001]
R_2	[100000]	**[010000]**	[001000]	[000100]	[000010]	[000001]
R_3	[100000]	[010000]	**[001000]**	[000100]	[000010]	[000001]
R_4	[100000]	[010000]	[001000]	**[000100]**	[000010]	[000001]
R_5	[100000]	[010000]	[001000]	[000100]	**[000010]**	[000001]
R_6	[100000]	[010000]	[001000]	[000100]	[000010]	**[000001]**

1 : 6, est le vecteur de résidus obtenu en écartant acc_X, acc_Y, acc_Z, mag_X, mag_Y, mag_Z respectivement. Le vecteur [100000] correspond au cas où il est certain que des résidus sont insensibles à la faute (0), i.e ce vecteur est obtenu en écartant la mesure en défaut. En effet, les cinq mesures introduites dans le bloc optimisation de la figure 4.7 sont alors sans défaut. Le quaternion obtenu correspond donc au quaternion réel (au bruit près bien évidemment). Lors de la reconstruction des mesures, on retrouve forcément les mêmes valeurs toujours au bruit près. Donc lorsque la différence est faite, il est normal

d'obtenir une valeur proche de zéro pour toutes les mesures sans défaut. Par contre, le 1 est quant à lui généré par le fait que le capteur écarté est en défaut. Donc la différence entre cette mesure et son estimée (qui est reconstruite avec le quaternion sans défaut) est forcement différente de zéro. En fait, cette différence correspond à l'amplitude du défaut. Par conséquent, l'étape *d'identification* est réalisée en même temps que l'étape de *localisation*.
[1∅∅∅∅∅] correspond au cas où le capteur fautif est pris en compte dans l'élaboration du vecteur, i.e, ce vecteur est sensible au défaut. Le ∅ correspond au fait que le résidu n'est sensible au défaut que si l'amplitude de ce dernier est supérieure au seuil de détection. Pour la localisation, le but du jeu est de rechercher R_i contenant cinq 0 et un 1.

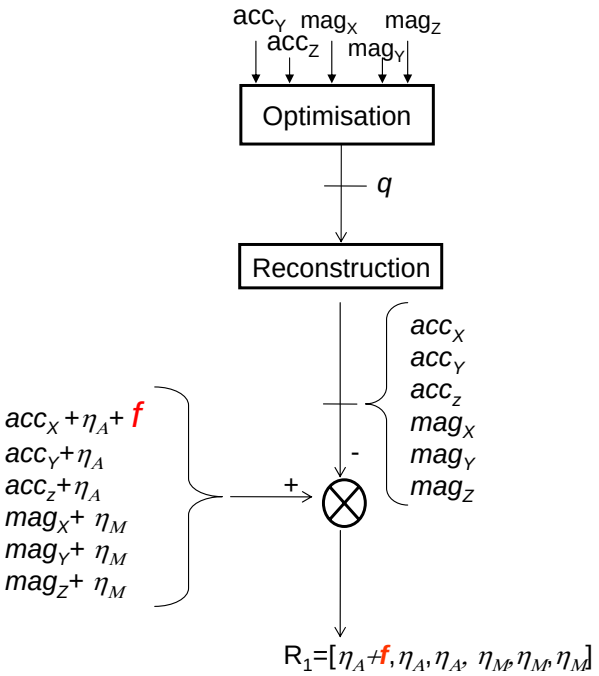


FIG. 4.8 – Schéma pour le calcul de R_1 avec *Algo 2*

Les figures 4.8 et 4.9 montrent comment un défaut est identifiable. Prenons par exemple le cas où le défaut est sur acc_X. La génération du vecteur de résidu R_1 (sans tenir compte de ce capteur) est donnée par la figure 4.8. On peut constater que chaque composante de ce vecteur est différente de zéro. Cependant, les composantes $R_1(2)$ à $R_1(6)$ sont égales aux bruits de mesure notés $\overrightarrow{\eta}_A$ et $\overrightarrow{\eta}_M$. Seule la composante $R_1(1)$ est différente du bruit de mesure. Celle-ci comporte en plus la valeur du défaut. La figure 4.9 montre le cas où la mesure fautive est prise en compte dans la génération du résidu. On peut remarquer que chaque composante de ces vecteurs sera entachée d'une valeur supplémentaire introduite par le défaut.

Remarque 9. Avec cette structure, des défauts multiples peuvent être envisageables.

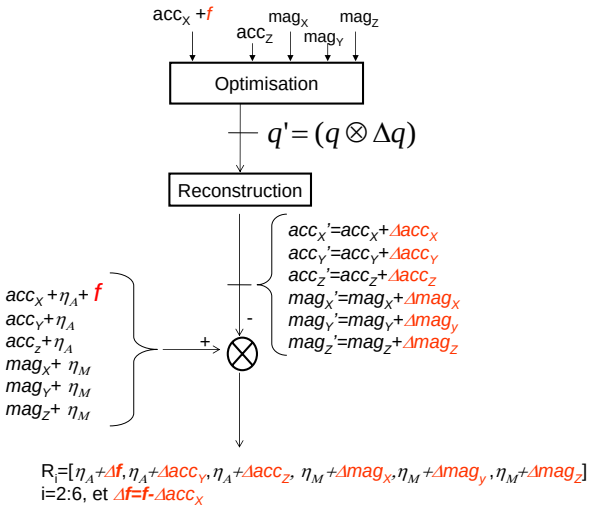


FIG. 4.9 – Schéma pour la génération de R_i, $i = 2 : 6$, qui permet l'identification du défaut, avec *Algo 2*

Dans tous les cas, ils seront détectables. L'étage de *localisation* peut être réalisé sous la condition que les défauts n'apparaissent pas simultanément. En effet, une fois le défaut détecté et localisé, la génération des résidus peut être faite non plus avec 5/6 mesures mais 4/6 (la mesure écartée + la mesure contenant un défaut). Le problème a encore un sens et peut être résolu puisque nous aurons 4 équations à 4 inconnues. Malheureusement cela dépend de l'attitude ! En effet, dans certains cas, on peut avoir un système d'équation sans solution ou avec une infinité de solution !

4.3.2 Diagnostic des gyromètres avec *Algo 2*

Pour les gyromètres, l'approche est plus ou moins similaire. La figure 4.10 montre la proposition pour la génération des résidus pour la détection des défauts sur les gyromètres. Dans cette approche, il y a deux étapes à considérer.

1. Recherche des défauts sur les accéléromètres et magnétomètres. Recherche décrite dans la section (4.3.1).

2. Estimation des gyromètres à partir du quaternion "jugé correct" par le module de diagnostic des accéléromètres et magnétomètres.

Pour diagnostiquer les défauts accéléromètres et magnétomètres, six états estimés sont générés notés $\hat{q}_i$ sur la figure 4.10, chacun d'eux étant généré en écartant une mesure. Dans un premier temps, il faut d'abord détecter et localiser l'éventuel défaut sur les accéléromètres et magnétomètres en utilisant la structure présentée précédemment. Cette étape permet d'obtenir un quaternion supposé "sain".

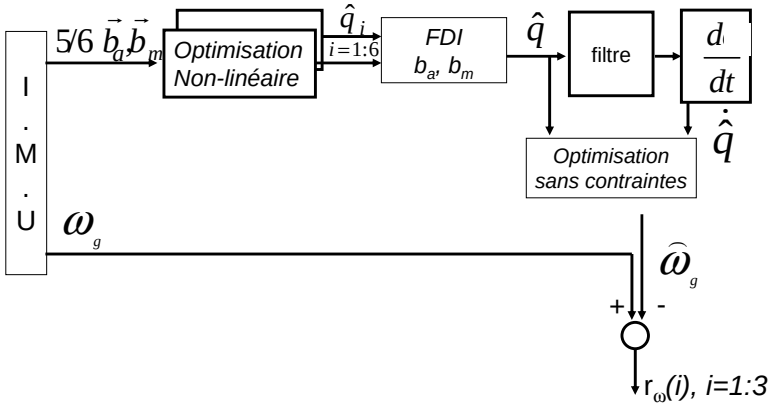


FIG. 4.10 – Génération des résidus pour les gyromètres avec *Algo 2*

Ce quaternion noté $\hat{q}$ est pris comme point de départ pour la génération des résidus pour les gyromètres. Ce quaternion est d'abord filtré afin d'éliminer l'influence du bruit. Le filtre choisi est un filtre passe bas du 1^{er} ordre dont l'équation est donnée par :

$$H(z) = \frac{(1-\alpha)z^{-1}}{1-\alpha z^{-1}} \tag{4.3.2}$$

avec $\alpha = e^{\frac{-T_e}{\tau}}$, τ la constante de temps du système et T_e correspond à la période d'échantillonnage du système considéré.
Une fois filtré, ce quaternion est dérivé. Cette dérivée est approchée par :

$$\frac{dq(t)}{dt} \simeq \frac{q_{filtre}(k) - q_{filtre}(k-1)}{T_e} \tag{4.3.3}$$

où T_e correspond à la période d'échantillonnage. Une fois cette dérivée calculée, les estimations des gyromètres sont calculées à l'aide de l'équation (2.1.25). Ces estimations sont obtenues grâce à une optimisation sans contrainte. L'implémentation de l'optimisation a été faite avec *fminunc* sous Matlab [106]. Le problème d'optimisation à résoudre consiste à minimiser le critère *f(ω)* suivant défini à partir de l'équation (3.5.3) :

$$\dot{q} = \frac{1}{2} q \bigotimes \Omega(\overrightarrow{\omega}) \begin{pmatrix} \frac{\partial q_0}{\partial t} \\ \frac{\partial q_1}{\partial t} \\ \frac{\partial q_2}{\partial t} \\ \frac{\partial q_3}{\partial t} \end{pmatrix} = \begin{pmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{pmatrix} * \begin{pmatrix} 0 & -\omega^T \\ \omega & -\omega^{\times} \end{pmatrix} \tag{4.3.4}$$

Par conséquent, on obtient :

$$min_{\omega} \;\; f(\overrightarrow{\omega}) = f_1(\overrightarrow{\omega}) + f_2(\overrightarrow{\omega}) + f_3(\overrightarrow{\omega}) + f_4(\overrightarrow{\omega}) \tag{4.3.5}$$

avec

$$f_1(\omega) = (\omega_1 * q_1 + \omega_2 * q_2 + \omega_3 * q_3 + 2 * \frac{\partial q_0}{\partial t})^2 \tag{4.3.6}$$

TAB. 4.7 – Table de signature pour les gyromètres avec *Algo 2*

	f_{gx}	f_{gy}	f_{gz}
$r_{\omega(1)}$	1	0	0
$r_{\omega(2)}$	0	1	0
$r_{\omega(3)}$	0	0	1

$$f_2(\omega) = (\omega_1 * q_0 + \omega_2 * q_3 - \omega_3 * q_2 - 2 * \frac{\partial q_1}{\partial t})^2 \tag{4.3.7}$$

$$f_3(\omega) = (\omega_1 * q_3 + \omega_2 * q_0 + \omega_3 * q_1 - 2 * \frac{\partial q_2}{\partial t})^2 \tag{4.3.8}$$

$$f_4(\omega) = (\omega_1 * q_2 - \omega_2 * q_1 + \omega_3 * q_0 - 2 * \frac{\partial q_4}{\partial t})^2 \tag{4.3.9}$$

Une fois l'estimation du vecteur $\hat{\overrightarrow{\omega}}$ réalisée, les résidus sont obtenus en effectuant la différence entre les mesures et leurs estimées par l'équation suivante :

$$r_\omega(i) = \omega_g(i) - \hat{\omega}(i), \quad i = 1:3 \tag{4.3.10}$$

La table de signature est donnée par la table 4.7. On remarque que chaque colonne de cette table est indépendante des autres, par conséquent, chaque défaut est localisable. Avec cette proposition, la détection des défauts multiples sur les gyromètres est réalisable car chaque colonne de la table ne comporte qu'un seul "1". De plus, l'amplitude du résidu en cas de défaut, correspond à l'amplitude du défaut additif. Ce point permet d'effectuer l'étape d'*identification*.

Remarque 10. Avec cette structure, les défauts multiples et simultanés peuvent être détectés mais aussi localisés. Par exemple, un défaut sur un des accéléromètres ou magnétomètres et un défaut sur un gyromètre peuvent être localisés. De même des défauts multiples sur les gyromètres peuvent être localisés.

4.4 Diagnostic du quadrotor : le cas des actionneurs

Dans cette section, une structure de diagnostic pour la détection des défauts sur les actionneurs va être présentée. Dans le cas de notre étude, l'actionneur correspond au moteur à courant continu associé à ses pâles. Le quadrotor est constitué de quatre rotors, par conséquent, nous sommes en présence de quatre actionneurs. La figure 3.8 page 52 montre la stratégie de commande pour le quadrotor. Nous sommes en présence de quatre boucles locales. Nous allons maintenant étudier le cas d'un seul actionneur constitué d'un moteur avec un régulateur noté C_{Mi} le tout implémenté en boucle fermée comme le montre la figure 4.11.

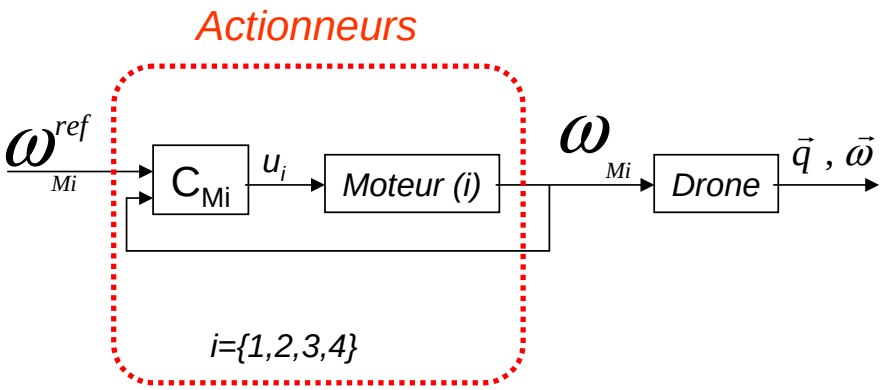


FIG. 4.11 – Structure d'un actionneur du quadrotor

La figure 4.12 montre la structure complète du moteur avec sa régulation et son module de diagnostic.

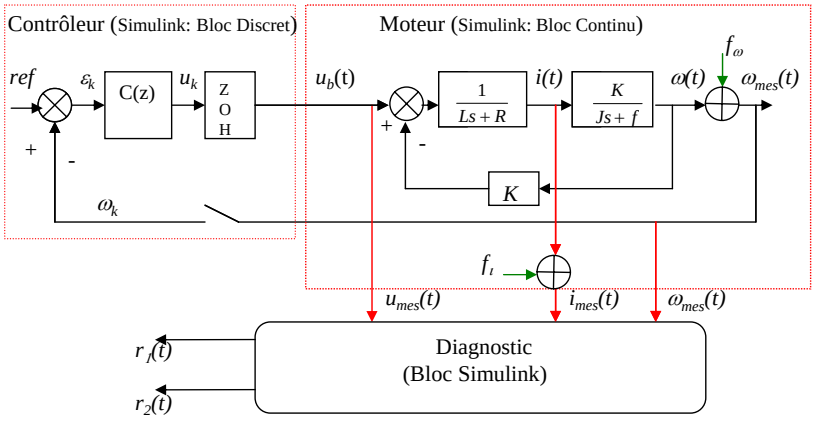


FIG. 4.12 – Actionneur du quadrotor avec la boucle de régulation locale et le module de diagnostic associé

Les équations représentant la fonction de transfert du moteur sont :

$$\omega_M(s) = \frac{K}{Js + f} * I(s) \tag{4.4.1}$$

$$I(s) = (U(s) - K * \omega_M(s))\frac{1}{Ls + R} \tag{4.4.2}$$

On aboutit à la fonction de transfert suivante :

$$\frac{\omega_M(s)}{U(s)} = \frac{\frac{K}{LJ}}{s^2 + [\frac{RJ+LF}{LJ}]s + \frac{RF+K^2}{LJ}} = H(s) \tag{4.4.3}$$

Nous obtenons par conséquent une fonction de transfert du second ordre du type :

$$H(s) = \frac{A}{(1 + \tau_1 s)(1 + \tau_2 s)} \tag{4.4.4}$$

avec $\tau_1 = 502\ ms$, $\tau_2 = 1.4\ ms$ et A =107 $rad/s.volt$. Dans la suite de cette section, la régulation C_{Mi} est un régulateur de type Proportionnel Intégral (PI) dont les coefficients ont été choisis afin d'obtenir un temps de réponse t_r en boucle fermée de 300 ms, correspondant à une constante de temps en boucle fermée de 100 ms. Pour trouver les coefficients, la synthèse d'un correcteur PI en continu a été réalisée puis le correcteur obtenu a été numérisé. La synthèse du correcteur PI en continu nous a donné les valeurs de $Ki = 0.093$ et $Kp = 0.046$. Pour plus d'information voir [15].

4.4.1 Génération des résidus

Pour la génération des résidus, nous supposons que l'on a des informations sur la vitesse de rotation du moteur mais aussi sur l'intensité. L'objectif ici est de construire deux résidus permettant de détecter des défauts à travers les valeurs du courant et de la vitesse de rotation. Pour ce faire, il faut considérer deux étapes.

- génération, à partir du modèle mécanique du système, des deux valeurs du courant et de la vitesse de rotation. Ces deux valeurs seront par la suite nommées valeurs du modèle ;
- génération de deux résidus en effectuant la différence entre les mesures et les valeurs du modèle. En l'absence de défaut, ces deux résidus devraient être inférieurs à des seuils préfixés et inversement si un défaut est introduit.

Les résidus sont générés à partir des équations du modèle par :

$$J\frac{d\omega_{M_{mod}}(t)}{dt} + f\omega_{M_{mod}}(t) = Ki_{mes}(t) \tag{4.4.5}$$

$$L\frac{di_{mod}(t)}{dt} + Ri_{mod}(t) = u_{mes} - K\omega_{M_{mes}}(t) \tag{4.4.6}$$

TAB. 4.8 – Paramètres utilisés pour le moteur et son diagnostic(figure 4.12)

$\omega_{M_{mes}}(t)$	Vitesse angulaire mesurée
$\omega_{M_{mod}}(t)$	Vitesse angulaire calculée par l'algorithme de diagnostic
$i(t)$	courant dans le moteur
$i_{mes}(t)$	courant mesuré dans le moteur
$i_{mod}(t)$	courant calculé par l'algorithme de diagnostic
u_k	Sortie du contrôleur
$u_{mes}(t)$	Tension mesurée aux bornes du moteur
f_ω, f_i	Défauts capteurs
$r_1(t), r_2(t)$	Résidus (sensibles à tous les défauts)
$R = 0.67\ \Omega$	Résistance
$L = 1\ mH$	Inductance
$J = 3.3\ \ 10^{-5} kg.m^2$	Moment d'inertie de l'actionneur
$f = 2.29\ \ 10^{-5} N.m.s$	Coefficient de frottement de l'actionneur
$K = \ \ 10^{-3}$	Constante du couple du moteur

TAB. 4.9 – Table de signature pour détecter les défauts moteur

	f_ω	f_i
$r_1(7.2.3)$	1	1
$r_2(7.2.4)$	1	1

TAB. 4.10 – Table de signature en utilisant les signes pour détecter les défauts moteur

	f_ω	f_i
$r_1(7.2.3)$	+	-
$r_2(7.2.4)$	+	+

$$r_1(t) = \omega_{M_{mes}}(t) - \omega_{M_{mod}}(t) \tag{4.4.7}$$

$$r_2(t) = i_{mes}(t) - i_{mod}(t) \tag{4.4.8}$$

La table de signature est donnée par 7.1. Pour rappel, les lignes représentent les résidus et les colonnes représentent l'influence d'un défaut. Dans cet exemple, on peut constater que les deux résidus sont sensibles aux deux défauts. On peut donc en conclure que les défauts sont détectables mais on ne pourra pas les localiser. Une technique a été proposée par [23], [24] pour pouvoir localiser le défaut. Cette dernière consiste à comparer les signes des résidus. Grâce aux signes des deux résidus, nous pouvons maintenant localiser les défauts.

4.4.2 Résultats en simulation

Maintenant, nous allons présenter des résultats obtenus en simulation. Nous allons tout d'abord montrer le cas sans défaut, puis nous introduirons un défaut capteur sur la vitesse de rotation et enfin, nous verrons l'influence d'un défaut introduit sur le capteur de courant.

4.4.2.1 Fonctionnement nominal

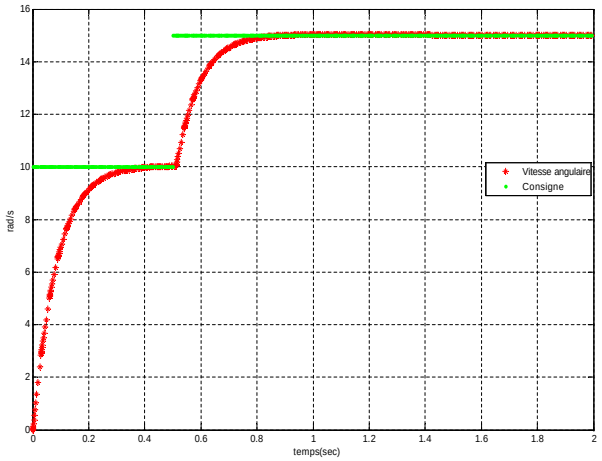


FIG. 4.13 – Réponse indicielle du système en boucle fermée

La figure 4.13 montre la réponse en boucle fermée du système avec une vitesse de référence de 10 rad/s puis d'un changement de consigne à $t = 0.5\ s$. La nouvelle référence est égale à 15 rad/s. On peut remarquer que le temps de réponse du système est de 300 ms ce qui correspond à une constante de temps τ en boucle fermée de 100 ms.

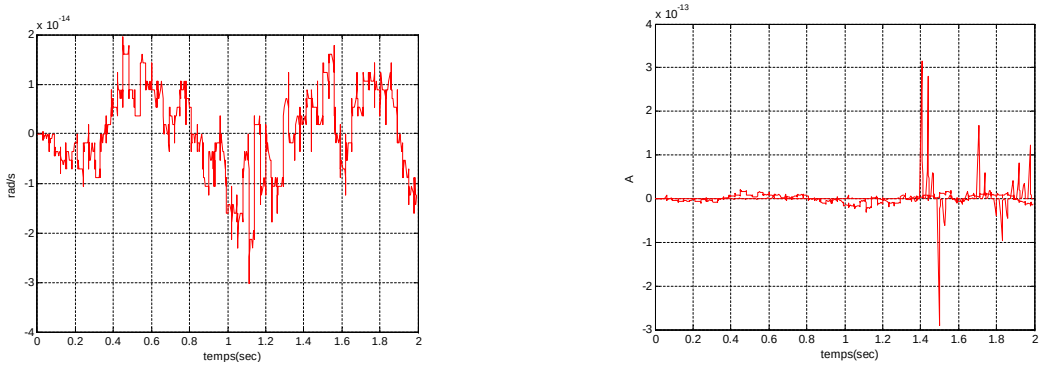


FIG. 4.14 – (gauche) Résidu r_1, (droite) Résidu r_2 (cas sans défaut)

La figure 4.14 montre les résidus r_1 et r_2 obtenus dans le cas nominal. Le lecteur est invité à remarquer que les amplitudes des résidus sont de l'ordre de grandeur des bruits.

Par conséquent, nous pouvons conclure que les deux résidus sont "nuls".

4.4.2.2 Fonctionnement avec un défaut sur la vitesse angulaire

Maintenant, un défaut capteur sur la vitesse angulaire mesurée $\omega_{M_{mes}}$ est introduit à l'instant $t = 1.2\ s$. Ce défaut est d'amplitude 5 rad/s soit 3% d'erreur par rapport à la vitesse de rotation réelle.

La figure 4.15 montre l'influence du défaut sur la réponse du système en boucle fermée. L'allure s'explique par le fait que le défaut est appliqué sur la vitesse de rotation qui est utilisée pour le calcul de la nouvelle consigne.

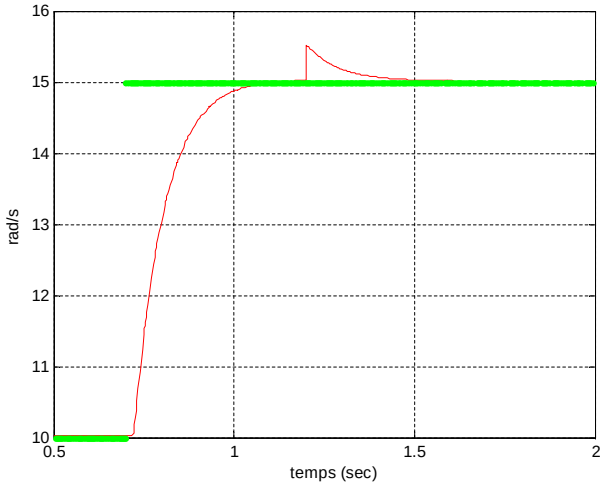


FIG. 4.15 – Réponse indicielle du système en boucle fermée avec un défaut sur la mesure de ω à $t = 1.2\ s$

La figure 4.16 montre les résidus obtenus avec un défaut sur la vitesse angulaire. On peut constater que ces deux résidus sont positifs. Si l'on compare avec la table de signature donnée par le tableau 4.7, on en déduit que le défaut est sur le capteur de vitesse : on a donc bien localisé le défaut.

4.4.2.3 Fonctionnement avec un défaut sur l'intensité

Dans cette étude, nous allons regarder l'influence que peut avoir un défaut sur l'intensité. Ce défaut est introduit à $t = 1.5\ s$ et est d'amplitude 2 mA soit 2% d'erreur. Ce défaut est de type échelon additif. La figure 4.17 montre que ce type de défaut n'a pas d'influence sur la réponse en boucle fermée du système. En effet, ceci s'explique par le fait que le retour se fait par la vitesse de rotation et non par l'intensité.

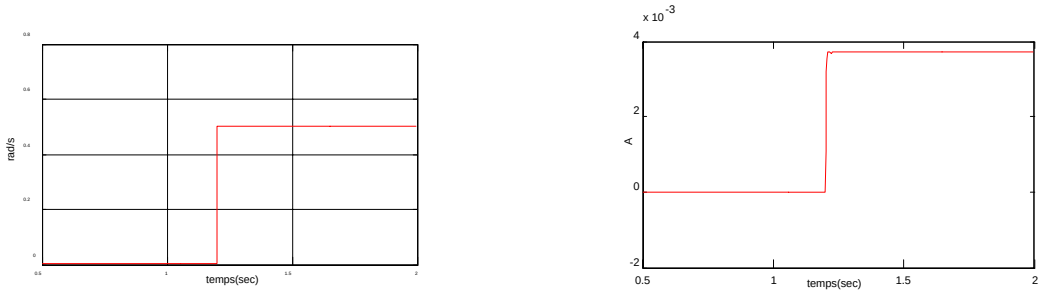


FIG. 4.16 – (gauche) : Résidu r_1 avec un défaut sur la vitesse angulaire, (droite) : Résidu r_2 avec un défaut sur la vitesse angulaire avec un défaut sur la mesure de ω à $t = 1.2\ s$

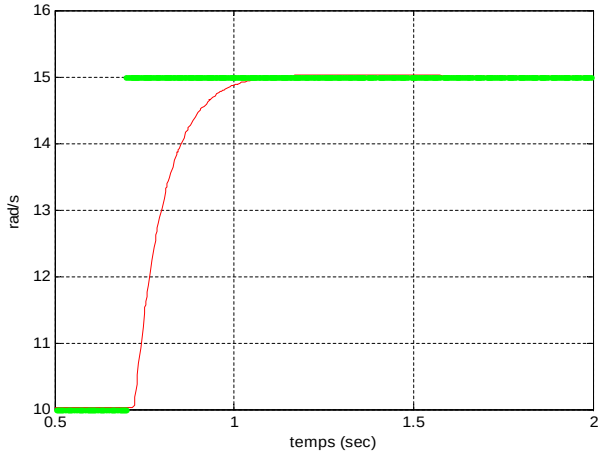


FIG. 4.17 – Réponse indicielle du système en boucle fermée avec un défaut sur la mesure de i à $t = 1.5\ s$

La figure 4.18 montre les résidus r_1 et r_2 en présence du défaut sur le courant. On peut remarquer que les deux résidus sont différents de zéro quand le défaut est apparu. Contrairement au cas précédant (défaut sur $\omega_{M_{mes}}$), les résidus sont de signes contraires. Si l'on compare avec la table de signature donnée par la table 4.7, on en déduit que le défaut est sur le capteur de courant car le résidu r_1 est négatif et que r_2 est positif : on a donc bien localisé le défaut.

4.4.3 Conclusion

Dans ce chapitre, après avoir introduit l'objectif du diagnostic, deux techniques pour détecter des défauts survenant sur la centrale d'attitude ont été proposés. La première technique, nommée *Algo 1*, utilise le principe des bancs d'observateurs généralisés. La génération des résidus est donc basée sur la reconstruction de l'état. L'inconvénient majeur, avec ce type de structure, est que les résidus sont directement liés au modèle du système. Plus le modèle est correct (proche du système), plus les résidus seront

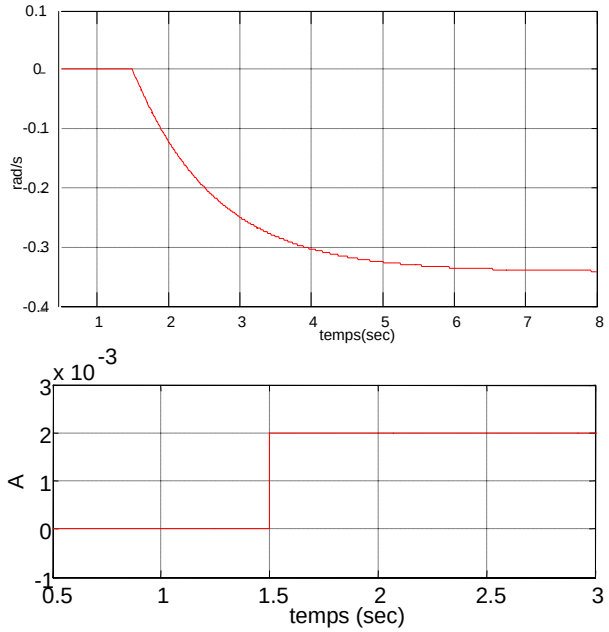


FIG. 4.18 – (haut) Résidu r_1 avec un défaut sur l'intensité, (bas) Résidu r_2 avec un défaut sur la mesure de i à $t = 1.5\ s$

robustes et inversement. En effet, les résidus générés sont sensibles aux paramètres tel que l'inertie. Un autre inconvénient est que ce genre de structure reste sensible aux perturbations. Pour les applications telle que celle du drone, cet inconvénient pose des problèmes si celui-ci est utilisé dans un environnement extérieur. En effet, le moindre coup de vent pourrait générer des fausses alarmes.

Pour palier ces inconvénients, une deuxième structure, nommée *Algo 2*, a été présentée. Cette structure, basée cette fois-ci sur le modèle de la centrale et non plus sur le modèle mécanique du système, permet d'obtenir des résidus basés sur les mesures directement et non plus sur l'état du système. L'avantage de cette structure est de rendre les résidus indépendants du modèle mécanique du système, de l'inertie mais aussi des perturbations liées à l'environnement extérieur (tel que le vent).
Pour illustrer cela, différents scénarios seront présentés dans le chapitre suivant afin de montrer les limitations de *Algo 1* et de voir la robustesse de *Algo 2* en présence de perturbation.

On a présenté dans ce chapitre, mais très brièvement, le diagnostic des actionneurs du quadrotor. En fait, ceci est moins difficile et aussi moins intéressant que pour le cas d'un défaut capteur. En effet, si un défaut actionneur apparaît sur le quadrotor, c'est tout le bloc actionneur qu'il faudra remplacer.

Remarque 11. L'intérêt d'une étude plus approfondie sur les défauts actionneurs serait

plutôt d'un point de vue commande des système. En effet, on pourrait imaginer, une fois le défaut apparu et localisé, d'essayer de passer à une loi de commande à trois rotors au lieu de quatre. Ceci est plutôt du FTC. Cependant, dans ce travail de thèse, on s'est focalisé sur les défaut capteurs et sur un autre thème qui est la commande des systèmes distribués (expliquée par la suite).

Chapitre 5

Résultats en simulation

Dans ce chapitre, des résultats en simulation seront présentés. Le but est ici de comparer les deux approches de diagnostic présentées dans ce mémoire. Sur toutes les courbes de ce chapitre, les unités sont :

- le degré pour la représentation de l'attitude du quadrotor par les angles ϕ, θ, ψ ;
- le degré pour les résidus obtenus par *Algo 1*. Ceci correspond à une différence de 2 positions paramétrées par 2 quaternions. Cette différence est ensuite convertie en angle. Cet angle est utilisé comme résidu pour cet algorithme (voir section 4.2) ;
- le gauss (unité des magnétomètres), le g (unité pour les accéléromètres) et rad/s (unité pour les gyromètres) pour les résidus obtenus par *Algo 2*. Ici, on reconstruit 6 mesures à partir de 5 puis on effectue la différence entre la mesure reconstruite et la mesure réelle (voir section 4.3).

Remarque 12. Pour la présentation des résultats, les angles d'Euler ont été choisis. Cependant, pour les raisons évoquées dans 2.1.4, tous les calculs sont basés sur le quaternion unitaire. Comme la représentation en quaternion n'est pas très physique, nous avons choisi de les convertir en angles afin de faciliter l'interprétation de nos résultats.

Remarque 13. Concernant les résultats obtenus par *Algo 2*, on a fait apparaître les deux bornes fixées par les seuils de décision sur les courbes représentant l'évolution des résidus. Ces seuils ont été déterminés par simulation. L'objectif est de réaliser l'étape de décision. Si l'amplitude du résidu est inférieure au seuil alors 0 lui sera associé et inversement, si l'amplitude du résidu est supérieure au seuil, alors 1 lui sera associé. De cette façon, l'étape *localisation* pourra être réalisée en comparant ces vecteurs à la table de signature fournie par le tableau 4.6.

Dans ce chapitre, seront présentés cinq scénarios différents.

Dans le premier scénario, le cas sans défaut sera considéré. Seront présentés les résultats obtenus pour l'attitude du quadrotor ainsi que les résultats obtenus pour les deux algorithmes de diagnostic.

Dans le deuxième scénario, un défaut sur acc_X sera considéré. De même, les résultats présentés seront l'attitude du quadrotor ainsi que les résidus obtenus avec les deux algorithmes de diagnostic.

Dans le troisième scénario, c'est cette fois-ci un défaut sur le $gyro_Z$ qui sera considéré.

Dans le quatrième scénario, ce sera également un défaut sur le $gyro_Z$ qui sera considéré mais cette fois-ci avec une amplitude très faible.

Enfin, dans le cinquième et dernier scénario, une perturbation est simulée.

5.1 Scénario 1 : pas de défaut

Comme l'indique le titre de ce paragraphe, le cas "nominal", c'est-à-dire le cas où l'ensemble du système fonctionne correctement est présenté. L'objectif ici est de stabiliser le quadrotor dans la position de référence égale à $\varphi = \phi = \psi = 0^{\circ}$ avec une attitude initiale égale à $\varphi = -25^{\circ}, \phi = -35^{\circ}, \psi = -10^{\circ}$. La figure 5.1 montre le résultat de la

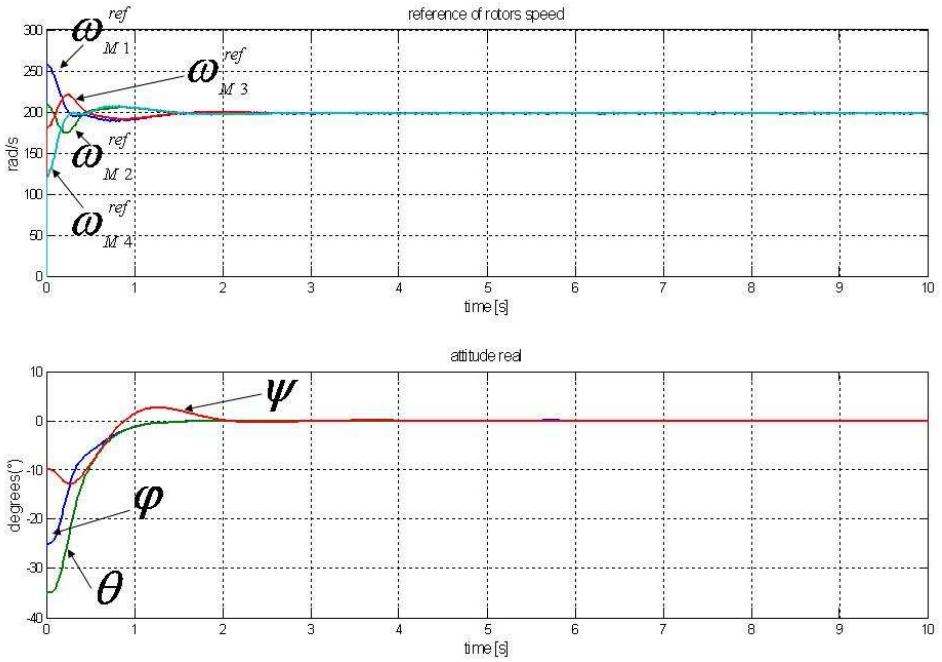


FIG. 5.1 – Attitude du drone sans défaut

stabilisation du quadrotor et on peut remarquer que le quadrotor se stabilise bien dans l'attitude souhaitée. Tous les résidus devraient être nuls dans ce scénario.

5.1.1 Génération des résidus par *Algo 1*

Le module de diagnostic utilisé est celui présenté dans [85] et décrit dans le paragraphe 4.2. La figure 5.2 montre les neuf résidus générés par le module de diagnostic nommé *Algo 1*. Les amplitudes instantanées des résidus R_2 et R_5 (sur les axes y) sont beaucoup plus importantes que les autres même si en moyenne les résidus sont bien nuls. Ceci est explicable par le fait qu'il n'y a aucune information sur cet axe dans le repère inertiel.

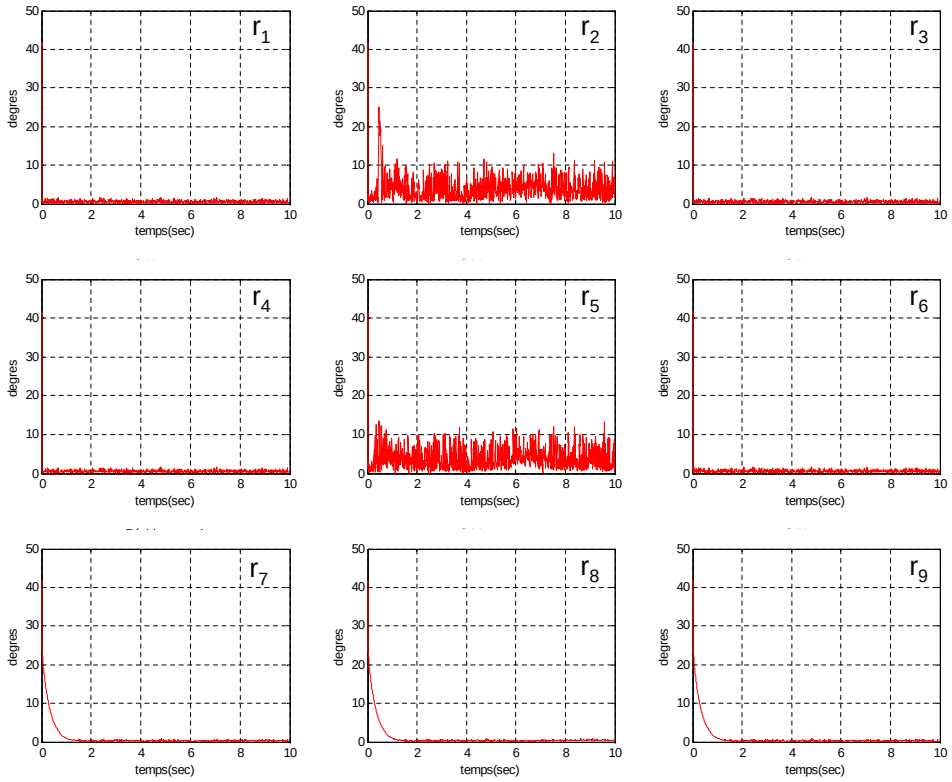


FIG. 5.2 – Les 9 résidus générés à partir de *Algo 1* (scénario 1)

5.1.2 Génération des résidus par *Algo 2*

Le module utilisé ici est celui décrit dans le paragraphe 4.3. Les figures 5.3, 5.4, 5.5, 5.6, 5.7, 5.8 montrent les six vecteurs R_i de résidus. Pour chacun d'eux, les trois courbes du haut concernent les résidus des accéléromètres. Les trois courbes du bas montrent les résidus des magnétomètres.

De ces résidus, plusieurs points peuvent être soulignés :

1. Dans l'ensemble, l'amplitude des résidus correspond au bruit de mesure.

2. Pour les vecteurs R_2 et R_5 voir les figures (5.4 et 5.7) le résidu obtenu $R_2(2)$ ou $R_5(5)$ est plus sensible que les autres (voir l'amplitude). Ceci est dû au fait que la reconstruction des mesures autour de l'axe Y est difficile car les références dans le repère inertiel sont 0 pour cet axe. Par conséquent, il n'y a pas assez d'information pour reconstruire correctement la mesure.
3. Des seuils ont été choisis à partir de cette expérience. Le tableau 5.1 fournit les valeurs des seuils pour chacun des 36 résidus. Par la suite si l'amplitude du défaut est supérieure à ce seuil alors le symptôme associé aura pour valeur binaire 1.

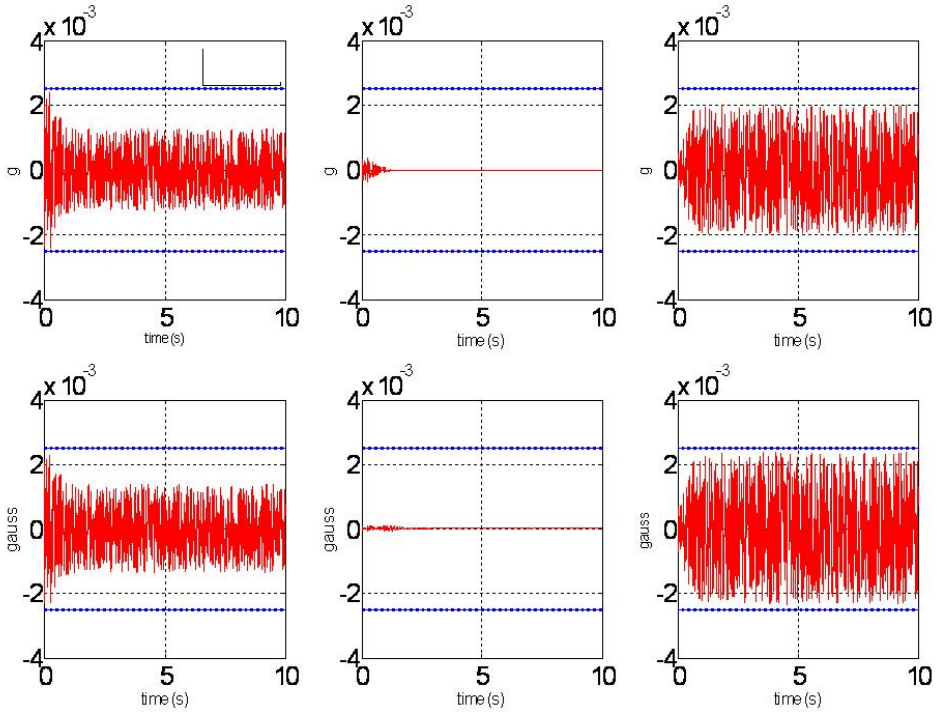


FIG. 5.3 – Les six résidus R_1 (en écartant acc_X) (scénario 1), R_1=[0,0,0,0,0,0]

TAB. 5.1 – Valeur des seuils pour chaque résidu obtenu par *Algo 2* afin de réaliser la décision binaire

	R_1	R_2	R_3	R_4	R_5	R_6
Seuil pour $R_i(1)$	$\pm 0.0025g$	$\pm 0.002g$	$\pm 0.0015g$	$\pm 0.0025g$	$\pm 0.0015g$	$\pm 0.0015g$
Seuil pour $R_i(2)$	$\pm 0.0025g$	$-0.12, +0.06g$	$\pm 0.0015g$	$\pm 0.0025g$	$\pm 0.001g$	$\pm 0.0015g$
Seuil pour $R_i(3)$	$\pm 0.0025g$	$\pm 0.0025\ g$	$\pm 0.0015g$	$\pm 0.0025g$	$\pm 0.0025g$	$\pm 0.0025g$
Seuil pour $R_i(4)$	$\pm 0.0025gauss$	$\pm 0.002gauss$	$\pm 0.002gauss$	$\pm 0.0025gauss$	$\pm 0.0025gauss$	$\pm 0.0015gauss$
Seuil pour $R_i(5)$	$\pm 0.0025gauss$	$\pm 0.002gauss$	$\pm 0.0025gauss$	$\pm 0.0025gauss$	$\pm 0.05gauss$	$\pm 0.0025gauss$
Seuil pour $R_i(6)$	$\pm 0.0025gauss$	$\pm 0.003gauss$	$\pm 0.003gauss$	$\pm 0.0025gauss$	$\pm 0.003gauss$	$\pm 0.0035gauss$

Les figures 5.9, 5.10, 5.11, 5.12, 5.13, 5.14 montrent les valeurs des résidus après la décision binaire, c'est-à-dire les symptômes de la détection.

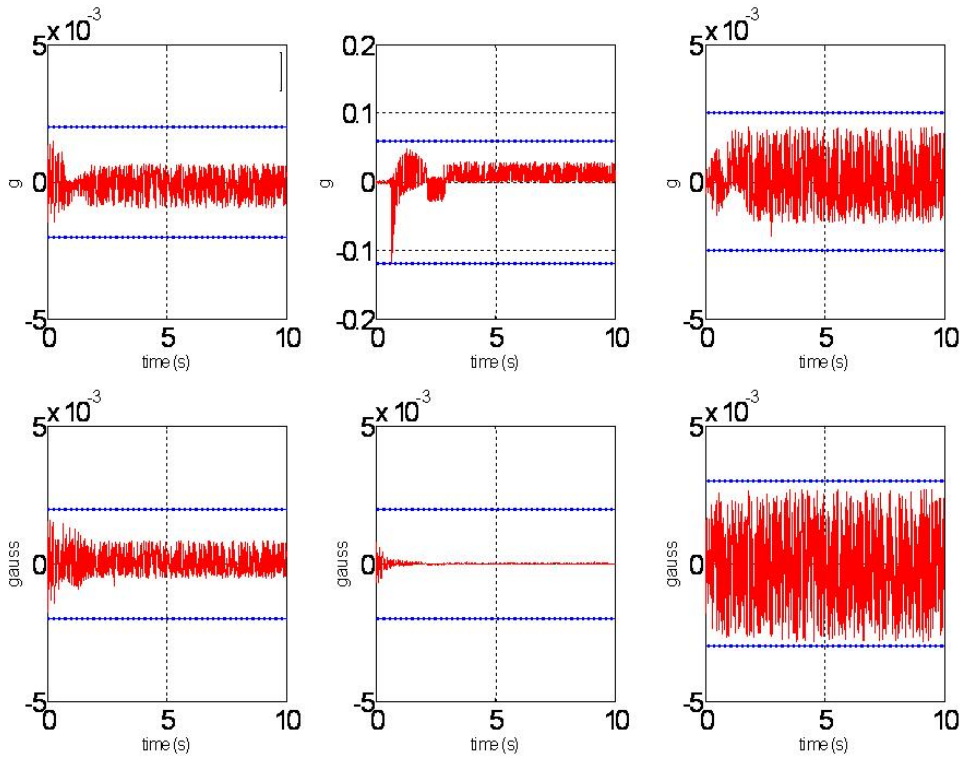


FIG. 5.4 – Les six résidus R_2 (en écartant acc_Y) (scénario 1), R_2=[0,0,0,0,0,0]

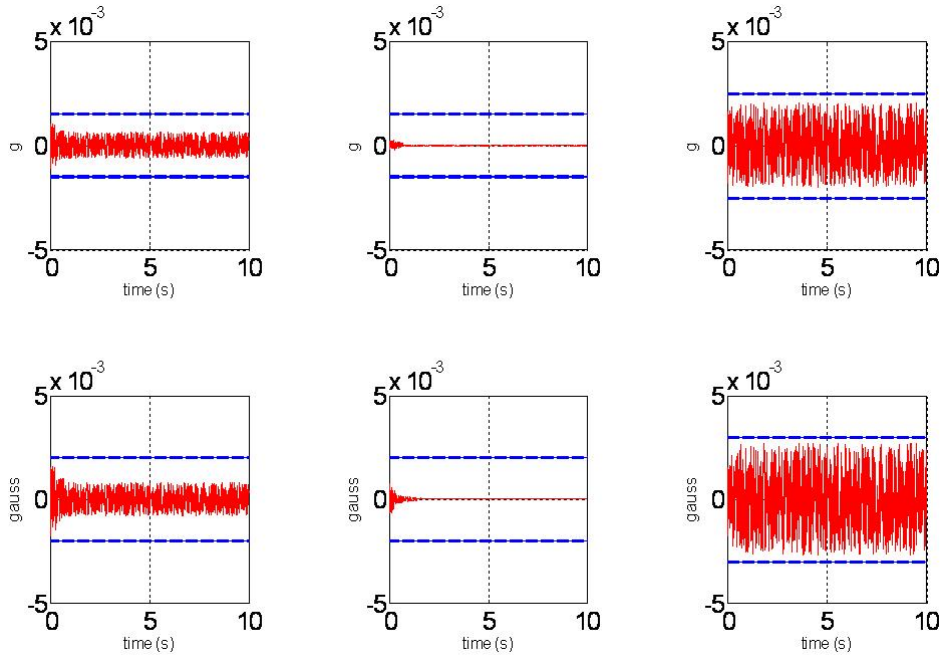


FIG. 5.5 – Les six résidus R_3 (en écartant acc_Z) (scénario 1), R_3=[0,0,0,0,0,0]

5.2 Scénario 2 : défaut sur acc_X

Maintenant, un défaut sur le capteur acc_X est considéré à partir de l'instant $t = 5$ s. Ce défaut est de type additif et est un biais d'amplitude 0.1 g soit l'équivalent de

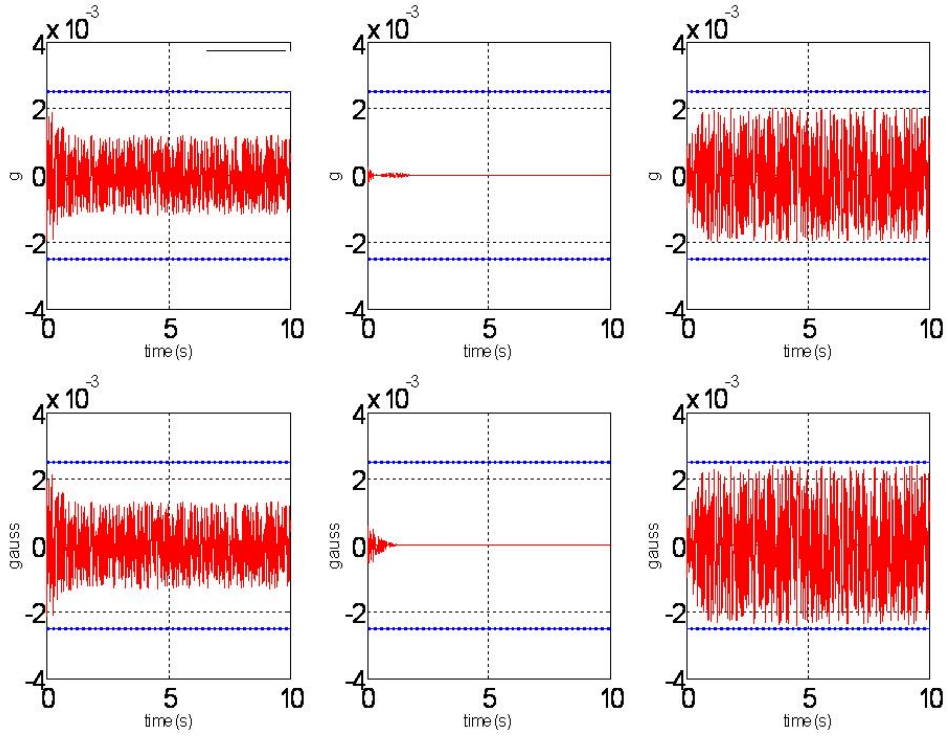


FIG. 5.6 – Les six résidus R_4 (en écartant mag_X) (scénario 1), R_4=[0,0,0,0,0,0]

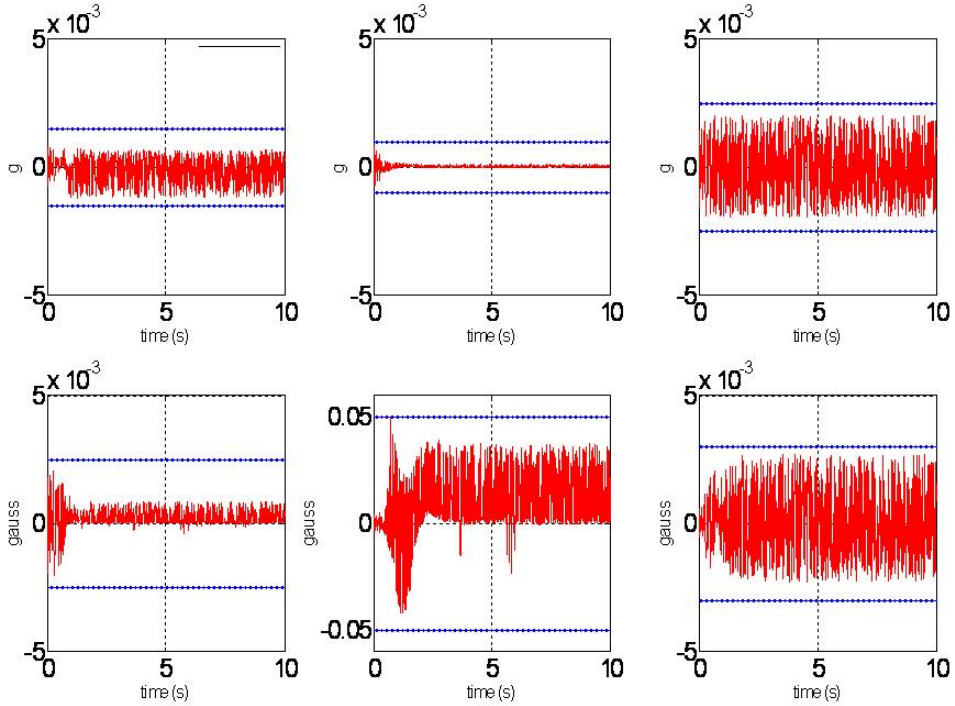


FIG. 5.7 – Les six résidus R_5 (en écartant mag_Y) (scénario 1), R_5=[0,0,0,0,0,0]

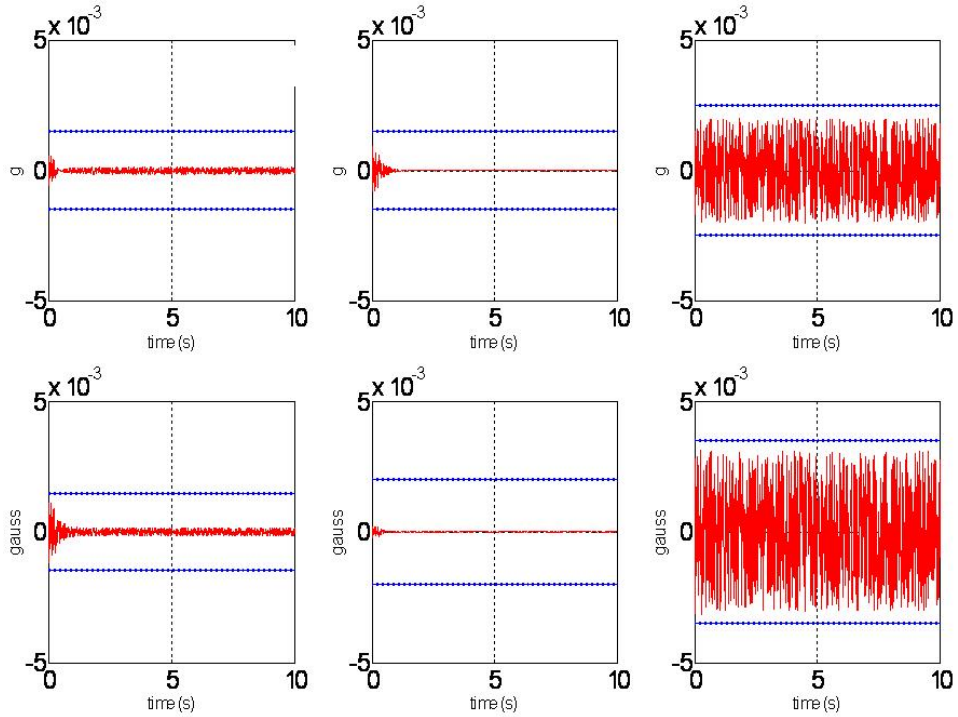


FIG. 5.8 – Les six résidus R_6 (en écartant mag_Z) (scénario 1), R_6=[0,0,0,0,0,0]

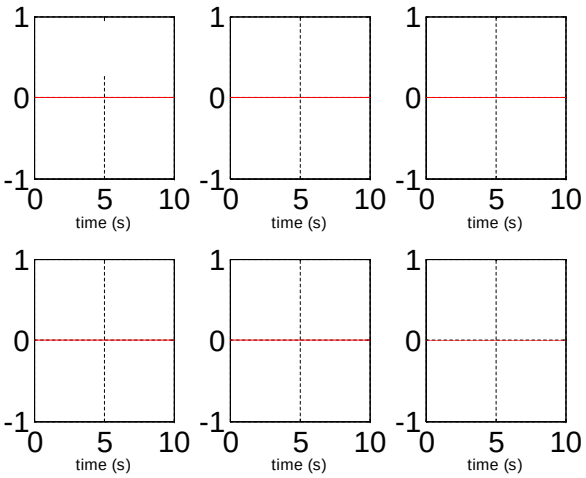


FIG. 5.9 – Les symptômes booléens des six résidus de R_1 (scénario 1)

10% de la valeur nominale. La figure 5.15 montre l'attitude du drone si le défaut n'est pas détecté. Ce défaut introduit une erreur sur l'attitude du drone de 3° sur le tangage (angle θ), et donc un déplacement du quadrotor.

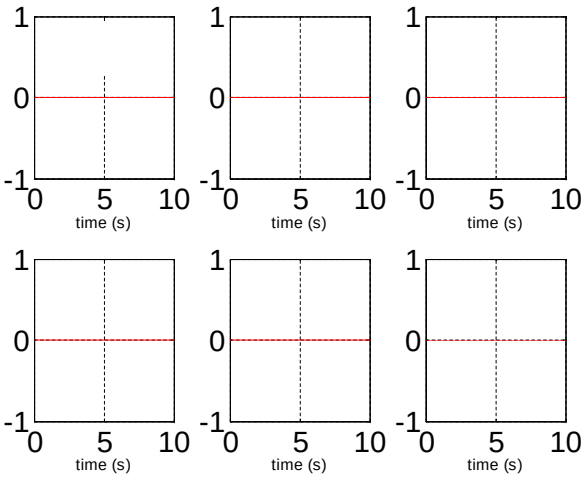


FIG. 5.10 – Les symptômes booléens des six résidus de R_2 (scénario 1)

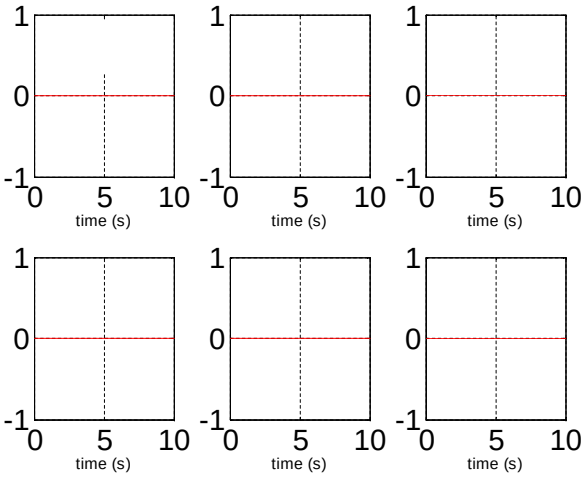


FIG. 5.11 – Les symptômes booléens des six résidus de R_3 (scénario 1)

5.2.1 Génération des résidus par *Algo 1*

La figure 5.16 montre le résultat de ce diagnostic. Les résidus sont tous sensibles à ce défaut excepté le résidu généré sans le capteur en défaut. Pour la localisation, il faut comparer ces résidus à la table de signature donnée par la table 4.5.
Cependant, la dernière étape du diagnostic, qui consiste à identifier le défaut, n'est pas évidente. Les trois résidus R_7, R_8, R_9 ont des allures différentes par rapport aux 6 premiers car nous sommes en présence d'une dynamique. Ces résidus sont générés, pour la surveillance des gyromètres, par le banc des observateurs non-linéaires. On peut remarquer qu'il faut un temps d'environ 1 s pour que les trois observateurs puissent converger.

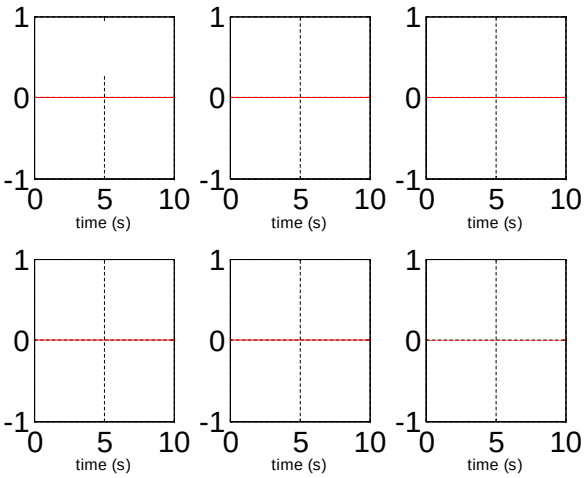


FIG. 5.12 – Les symptômes booléens des six résidus de R_4 (scénario 1)

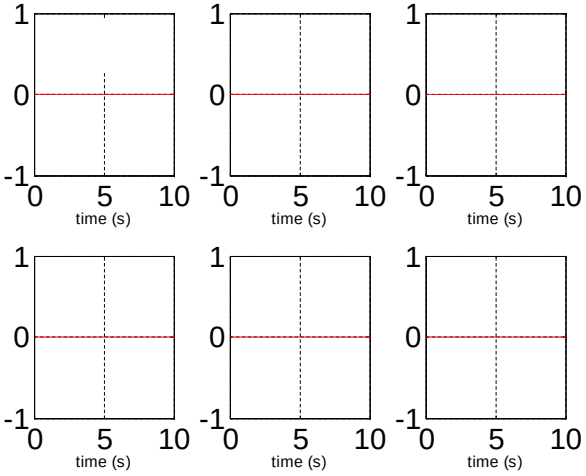


FIG. 5.13 – Les symptômes booléens des six résidus de R_5 (scénario 1)

5.2.2 Génération des résidus par *Algo 2*

Maintenant, la deuxième approche est utilisée. Les figures 5.17, 5.18, 5.19, 5.20, 5.21, 5.22 montrent les vecteurs avec le défaut additif sur acc_X. Cinq résidus sur six de R_1 sont insensibles à ce défaut et leur amplitude est égale aux bruits présents sur les mesures ($\overrightarrow{\eta}_a$ et $\overrightarrow{\eta}_m$, respectivement). Seul le résidu $R_1(1)$ (résidu entre acc_X et son estimé) est sensible à ce défaut. La figure 4.8 synthétise le résultat obtenu. De plus on peut remarquer que l'amplitude de ce résidu est égale à l'amplitude du défaut. Il est parfaitement identifié.

Concernant les 5 autres vecteurs de résidus (résidus générés en prenant en compte le capteur en défaut), on remarque que quelques composantes de chaque vecteur sont sensibles à ce défaut. La figure 4.9 page 79 analyse les résultats observés.

Preuve C :

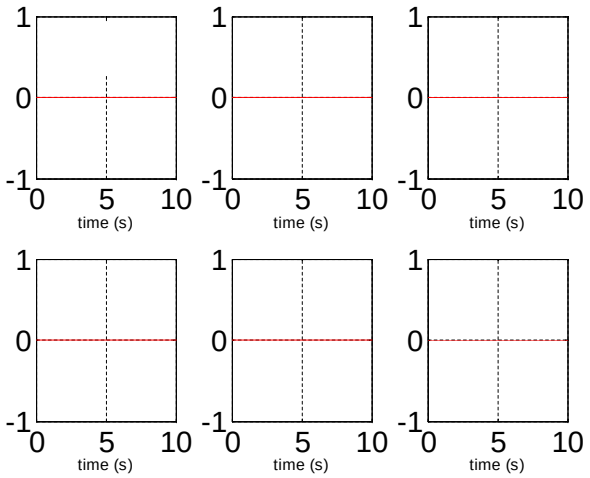


FIG. 5.14 – Les symptômes booléens des six résidus de R_6 (scénario 1)

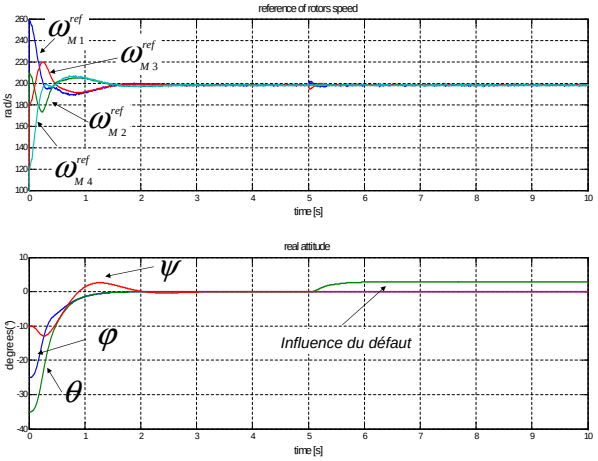


FIG. 5.15 – Attitude réelle du drone avec un défaut additif de 0.1 g sur le capteur acc_X à partir de $t = 5\ s$ (scénario 2)

onsidérons les expressions des accéléromètres et magnétomètres en terme de quaternion obtenues à partir des équations (4.2.5) à (4.2.10) :

$$acc_X = 2 * (q_1 * q_3 - q_0 * q_2) \tag{5.2.1}$$

$$acc_Y = 2 * (q_2 * q_3 + q_0 * q_1) \tag{5.2.2}$$

$$acc_Z = (q_0^2 - q_1^2 - q_2^2 + q_3^2) \tag{5.2.3}$$

$$\begin{gathered} mag_X = 0.5 * (q_0^2 + q_1^2 - q_2^2 - q_3^2) + \frac{\sqrt{3}}{2} * 2 * (q_1 * q_3 - q_0 * q_2) \\ \Rightarrow mag_X = 0.5 * (q_0^2 + q_1^2 - q_2^2 - q_3^2) + \frac{\sqrt{3}}{2} * acc_X \end{gathered} \tag{5.2.4}$$

$$\begin{gathered} mag_Y = (q_1 * q_2 - q_0 * q_3) + \frac{\sqrt{3}}{2} * 2 * (q_2 * q_3 + q_0 * q_1) \\ \Rightarrow mag_Y = (q_1 * q_2 - q_0 * q_3) + \frac{\sqrt{3}}{2} * acc_Y \end{gathered} \tag{5.2.5}$$

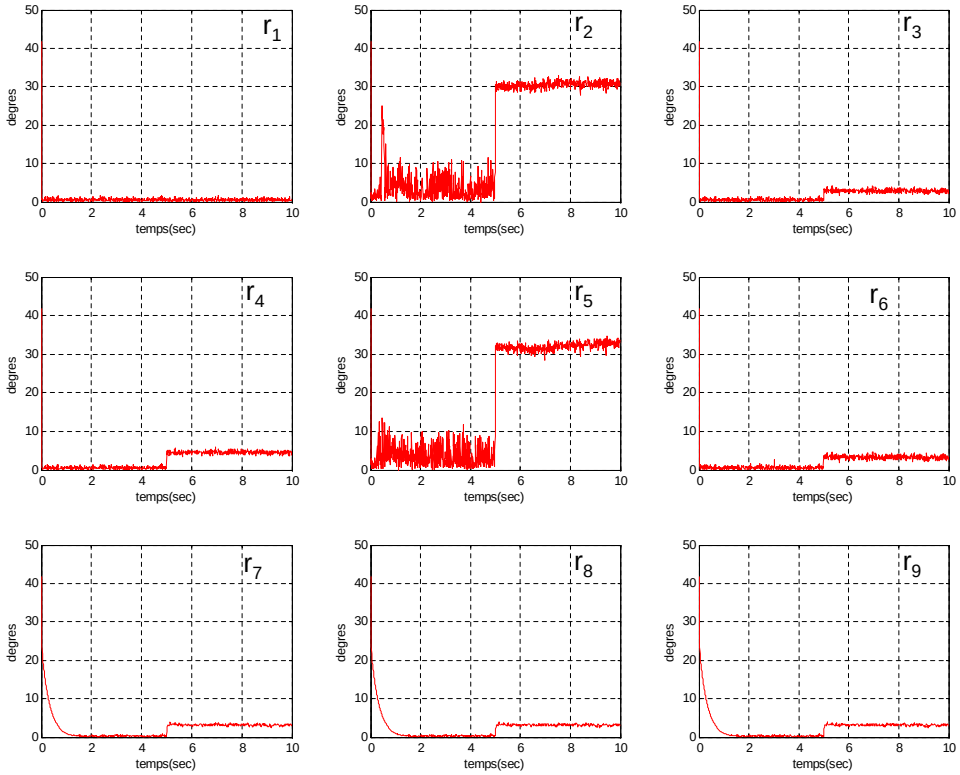


FIG. 5.16 – Les 9 résidus obtenus par *Algo 1* avec un défaut sur acc_X à partir de $t = 5\ s$ (scénario 2)

$$\begin{aligned} mag_Z &= (q_1 * q_3 + q_0 * q_2) + \frac{\sqrt{3}}{2} * (q_0^2 - q_1^2 - q_2^2 + q_3^2) \\ \Rightarrow mag_Z &= (q_1 * q_3 + q_0 * q_2) + \frac{\sqrt{3}}{2} * acc_Z \end{aligned} \tag{5.2.6}$$

Avec la contrainte suivante :

$$\|q\|_2 = 1 \tag{5.2.7}$$

5.2.2.1 Raisonnement

Avant l'apparition du défaut sur acc_X, les hypothèses à considérer sont les suivantes :

- l'attitude du drone est stabilisée à la valeur $\phi = \theta = \psi = 0^\circ$, c'est-à-dire q=[1 0 0 0]^T ;
- en négligant les bruits de mesure, dans cette attitude, les mesures des accéléromètres et magnétomètres sont égales aux valeurs de référence c'est-à-dire $acc_X = acc_Y = 0$ et $acc_Z = 1$, $mag_X = 0.5$, $mag_Y = 0$ et $mag_Z = 0.833$
- ces valeurs satisfont les équations (5.2.1) à (5.2.7).

A $t = 5\ s$, le défaut sur acc_X apparaît. L'équation (5.2.1) devient :

$$acc_X + f = 2 * (\tilde{q}_1 * \tilde{q}_3 - \tilde{q}_0 * \tilde{q}_2) \tag{5.2.8}$$

tandis que les autres mesures sont :

$$acc_Y = 2 * (q_2 * q_3 + q_0 * q_1) \tag{5.2.9}$$

$$acc_Z = (q_0^2 - q_1^2 - q_2^2 + q_3^2) \tag{5.2.10}$$

$$\begin{gathered} mag_X = 0.5 * (q_0^2 + q_1^2 - q_2^2 - q_3^2) + \frac{\sqrt{3}}{2} * 2 * (q_1 * q_3 - q_0 * q_2) \\ \Rightarrow mag_X = 0.5 * (q_0^2 + q_1^2 - q_2^2 - q_3^2) + \frac{\sqrt{3}}{2} * acc_X \end{gathered} \tag{5.2.11}$$

$$\begin{gathered} mag_Y = (q_1 * q_2 - q_0 * q_3) + \frac{\sqrt{3}}{2} * 2 * (q_2 * q_3 + q_0 * q_1) \\ \Rightarrow mag_Y = (q_1 * q_2 - q_0 * q_3) + \frac{\sqrt{3}}{2} * acc_Y \end{gathered} \tag{5.2.12}$$

$$\begin{gathered} mag_z = (q_1 * q_3 + q_0 * q_2) + \frac{\sqrt{3}}{2} * (q_0^2 - q_1^2 - q_2^2 + q_3^2) \\ \Rightarrow mag_z = (q_1 * q_3 + q_0 * q_2) + \frac{\sqrt{3}}{2} * acc_z \end{gathered} \tag{5.2.13}$$

Avec les conditions précédentes, c'est à dire avec un défaut d'amplitude 0.1, on obtient alors

$$\tilde{q} = [q_0 + \Delta q_0, q_1 + \Delta q_1, q_2 + \Delta q_2, q_3 + \Delta q_3] \tag{5.2.14}$$

ce qui peut s'exprimer par :

$$\tilde{q} = q + \Delta q \tag{5.2.15}$$

□

Pour localiser le défaut, il faut chercher le vecteur qui contient 5 zéros et un 1. Pour cet exemple, c'est le vecteur R_1. On s'aperçoit également que les résidus $R_i(1)$ avec $i = 2 : 6$ sont toujours égaux à 1 en présence du défaut. Certains des autres résidus sont sensibles au défaut, d'autres non. Ceci peut s'expliquer par le choix de la valeur des seuils et par le fait que le défaut n'a pas la même influence suivant l'attitude du quadrotor. Le choix devrait se porter sur des seuils adaptatifs et on devrait implanter des tests statistiques.

5.3 Scénario 3 : défaut sur $\omega_g(z)$

Dans ce scénario, un défaut sur le gyromètre Z est introduit. Ce défaut est de type additif. Entre $t = 1\ s$ et $t = 2\ s$, un biais d'amplitude 0.5 rad/s est introduit. Puis entre $t = 3\ s$ et $t = 6\ s$, un défaut en rampe de pente 1 rad/s est introduit comme le montre la figure 5.23.
Dans ce scénario, l'attitude de référence est $\varphi = \theta = \psi = 0^\circ$ puis à $t = 5\ s$, l'attitude de référence change et est égale à $\varphi = \theta = \psi = 10^\circ$. La figure 5.24 montre l'attitude réelle du drone en cas de non détection du défaut. Comme on pouvait s'y attendre, le quadrotor n'est pas stabilisé dans son attitude de référence.

5.3.1 Génération des résidus par *Algo 1*

La figure 5.25 illustre le résultat obtenu pour la phase de génération des résidus. On s'aperçoit, comme attendu, que les résidus mis en place pour la surveillance des accéléromètres et magnétomètres (résidus numérotés de R_1 à R_6) sont proches de zéro

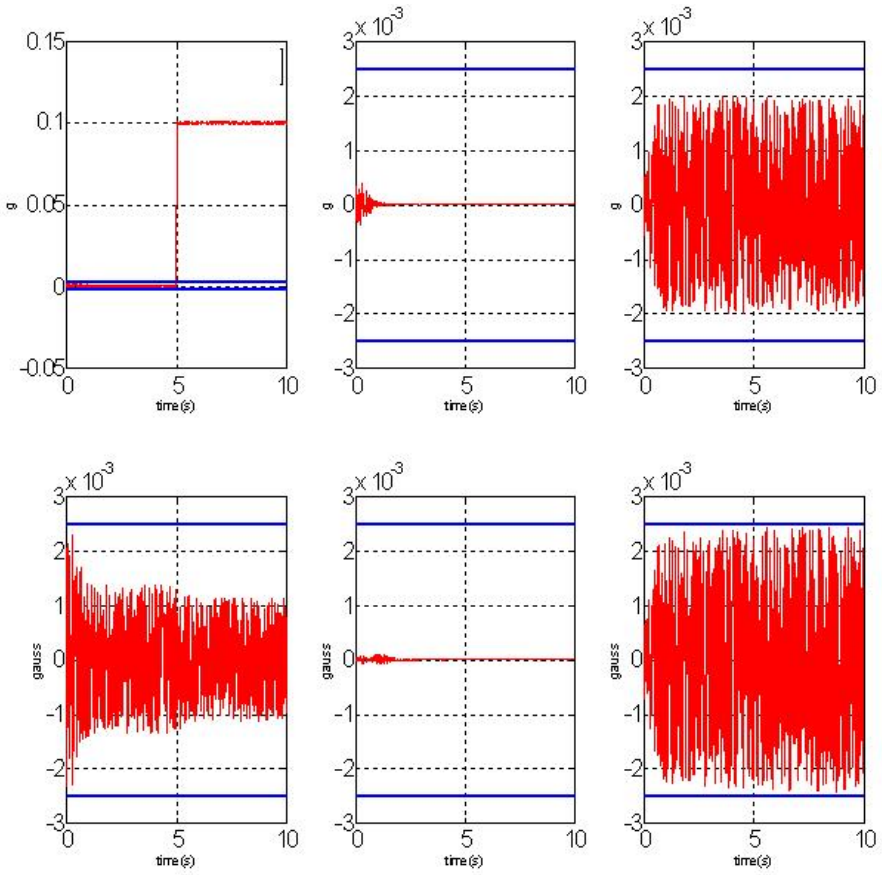


FIG. 5.17 – Les six résidus R_1 (en écartant acc_X) (scénario 2), R_1=[1 0 0 0 0 0]

donc insensibles à ce défaut. Ceci s'explique par le fait que pour leur génération, les mesures des gyromètres ne sont pas utilisées.
Pour les trois derniers résidus (r_7, r_8, r_9), on s'aperçoit que deux des trois résidus sont sensibles à ce défaut. En effet, pour les deux résidus sensibles, le capteur en défaut est utilisé pour la génération du résidu. La phase de localisation peut être réalisée à l'aide de la table de signature correspondante (voir table 4.5). Une fois de plus, soulignons que ce défaut n'est pas identifiable.

5.3.2 Génération des résidus par *Algo 2*

La figure 5.26 montre les résidus obtenus à partir de *Algo 2*. Entre $t = 1$ s et $t = 2$ s, le résidu dédié à la surveillance du gyromètre d'axe z est différent de zéro et de même pour les instants compris entre $t = 3$ s et $t = 6$ s. Le défaut est donc détecté. Pour la localisation, il faut se rapporter à la table de signature (table 4.7). Cette comparaison informe que le défaut est apparu sur le gyromètre d'axe z. Remarquons que l'amplitude du résidu $r_\omega(3)$ correspond à l'amplitude du défaut introduit donc l'étape d'identification

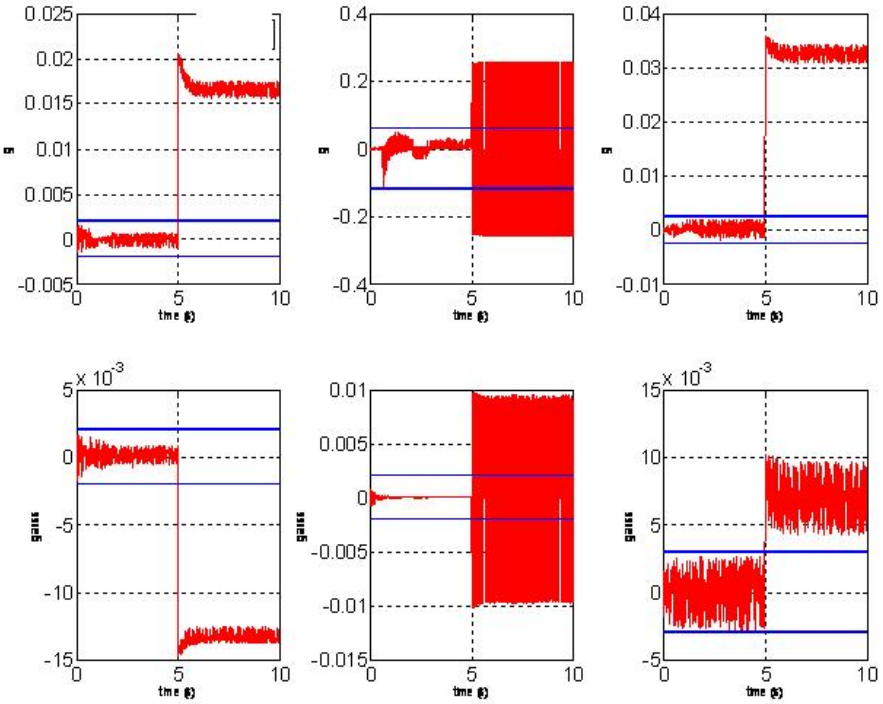


FIG. 5.18 – Les six résidus de R_2 (en écartant acc_Y) (scénario 2), R_2=[1 1 1 1 1 1]

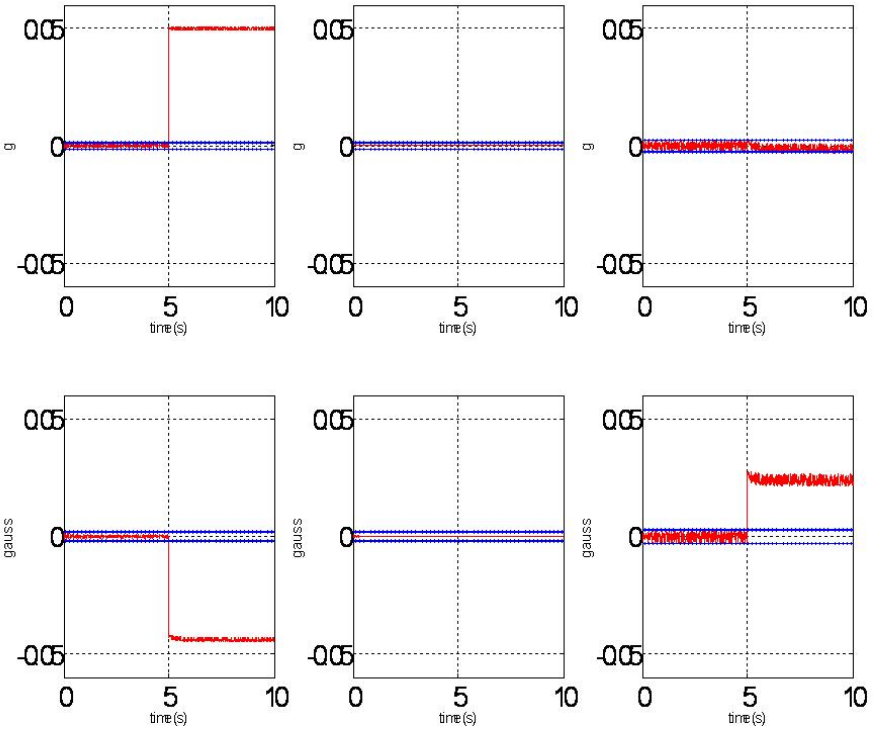


FIG. 5.19 – Les six résidus de R_3 (en écartant acc_Z) (scénario 2), R_3=[1 0 0 1 0 1]

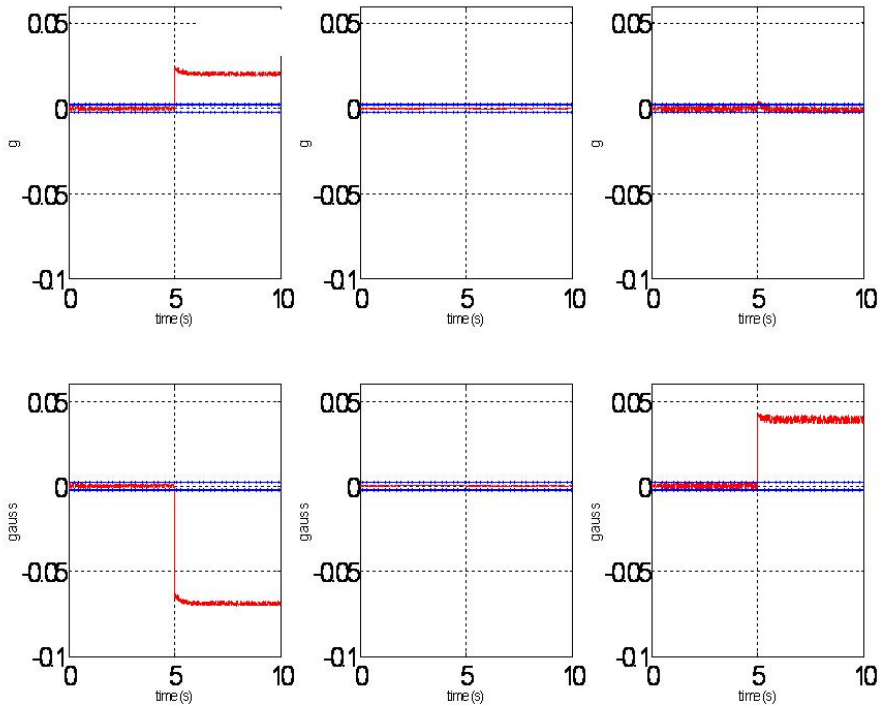


FIG. 5.20 – Les six résidus de R_4 (en écartant mag_X) (scénario 2), R_4=[1 0 0 1 0 1]

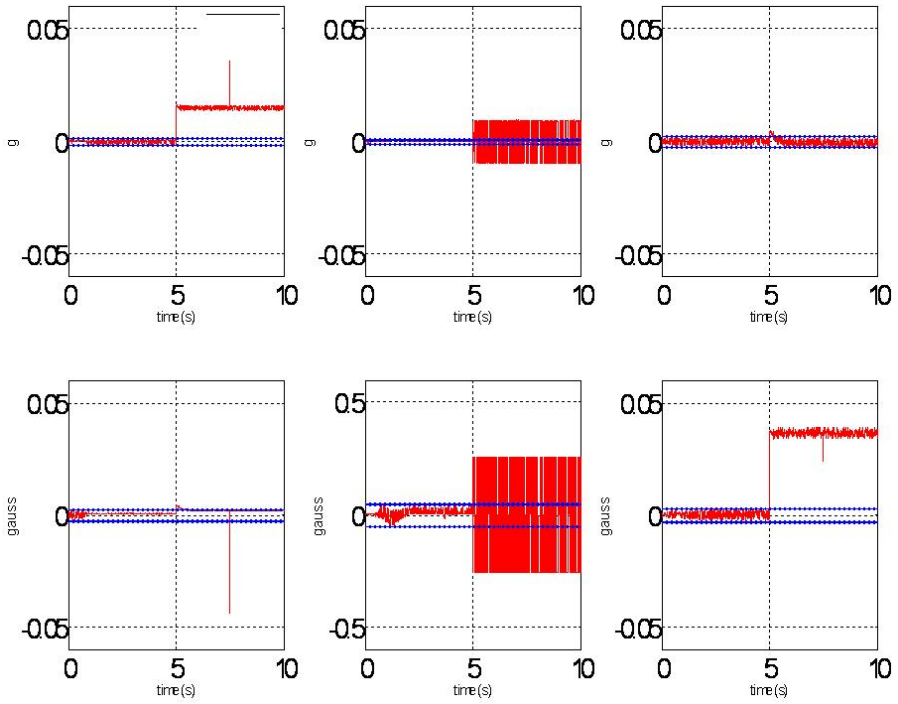


FIG. 5.21 – Les six résidus de R_5 (en écartant mag_Y) (scénario 2), R_5=[1 0 0 0 1 1]

du défaut est accomplie. Par conséquent, la phase de reconfiguration en cas de défaut devient relativement facile à faire.

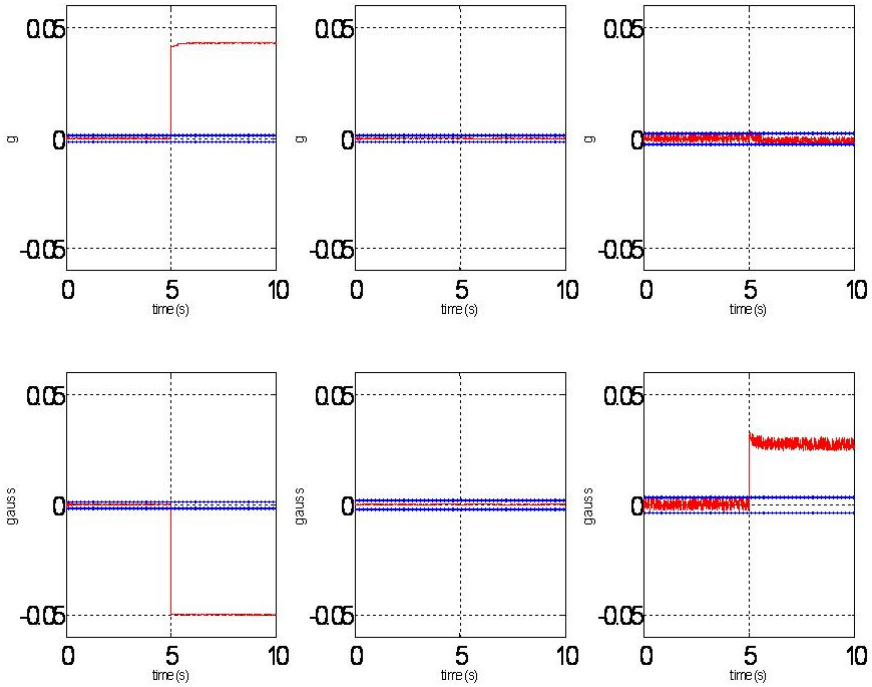


FIG. 5.22 – Les six résidus de R_6 (en écartant mag_Z) (scénario 2), R_6=[1 0 0 0 0 1]

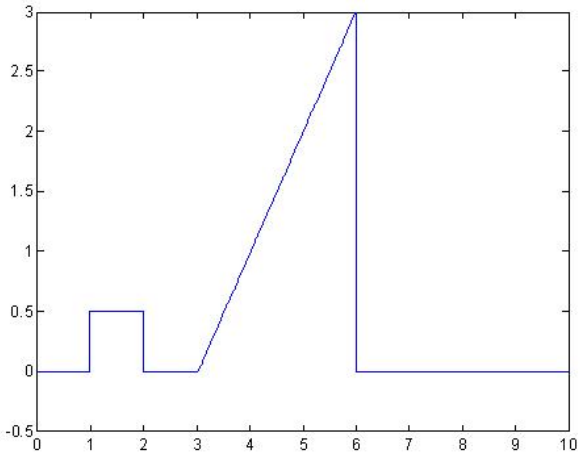


FIG. 5.23 – Forme du défaut additif introduit sur le gyromètre d'axe z ($\omega_g(z)$) (scénario 3).

5.4 Scénario 4 : défaut sur $\omega_g(z)$ d'amplitude très faible

Dans [43], les auteurs ont montré qu'il était difficile de diagnostiquer un éventuel défaut additif de 10% sur un axe par la structure de bancs d'observateurs. Dans ce scénario, un biais d'amplitude 0.1 rad/sec est introduit sur le gyromètre d'axe z à l'instant $t = 5\ s$.

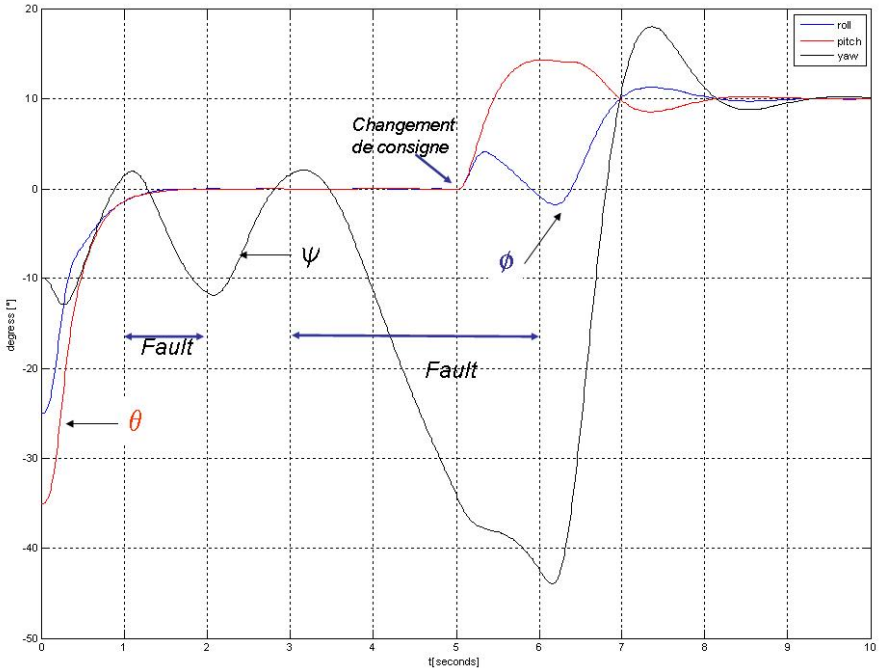


FIG. 5.24 – Attitude réelle du quadrotor avec un défaut sur $gyro_Z$ (scénario 3)

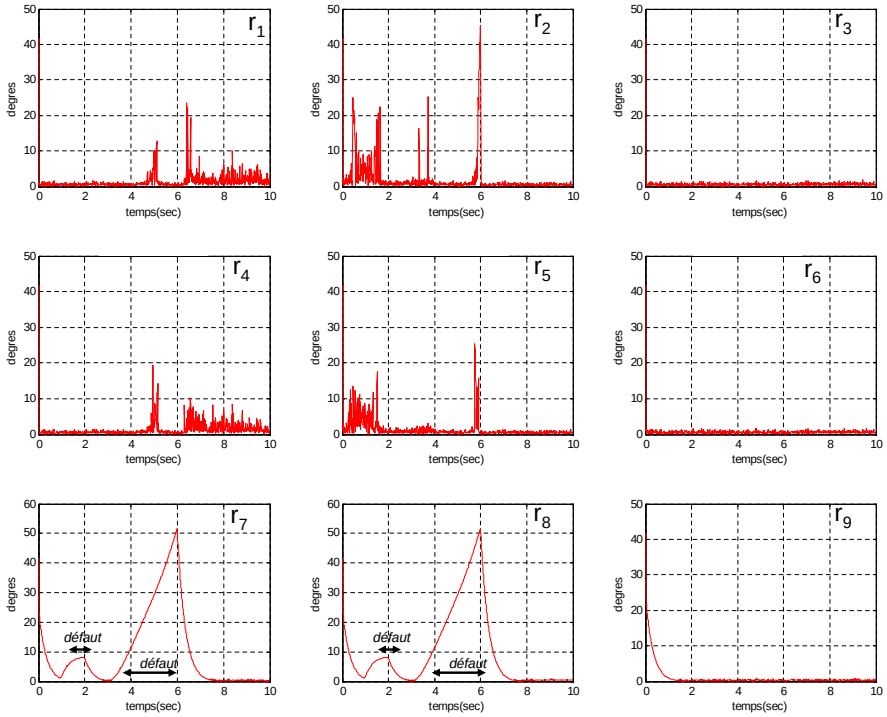


FIG. 5.25 – Les 9 résidus obtenus par *Algo 1* avec un défaut sur $gyro_Z$ (scénario 3)

La figure 5.27 montre l'influence de ce défaut sur l'attitude du drone. Bien évidemment ce défaut introduit une erreur sur le lacet du drone.

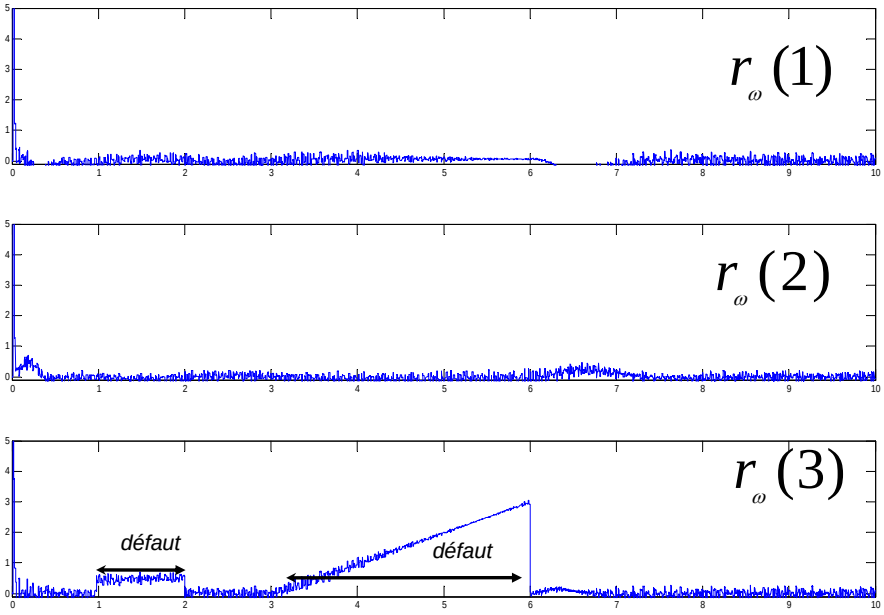


FIG. 5.26 – Résidus générés par *Algo 2* avec un défaut sur $gyro_Z$ (scénario 3)

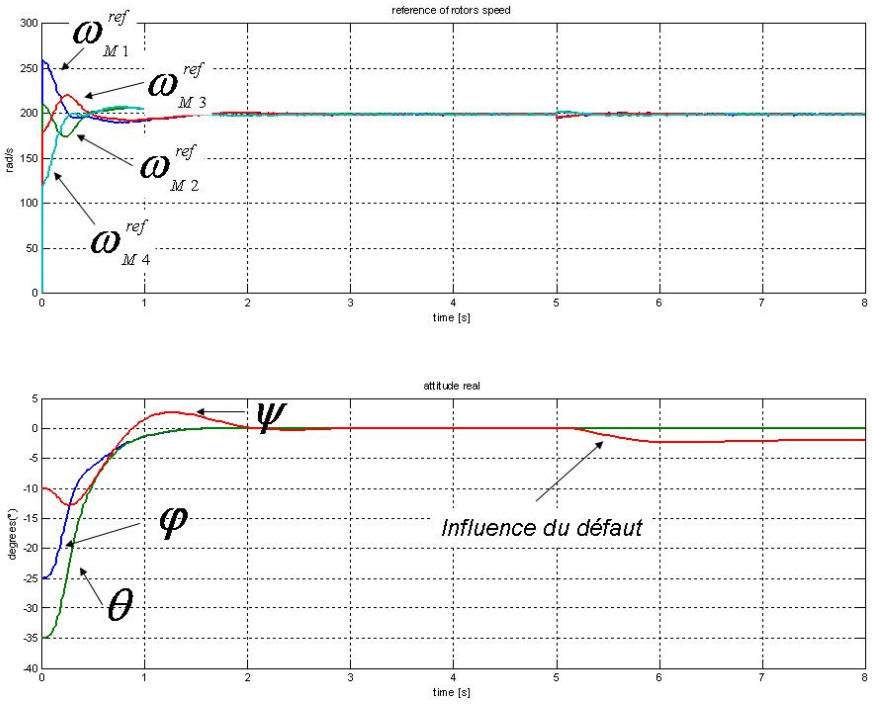


FIG. 5.27 – Attitude réelle du quadrotor avec un défaut sur $gyro_Z$ (scénario 4)

5.4.1 Génération des résidus par *Algo 1*

La figure 5.28 montre les résidus obtenus à partir de *Algo 1*. Avec cette structure, on constate que le défaut sur le gyromètre n'est pas détectable car tous les résidus sont

considérés comme nul. Pour rappel, pour que ce défaut soit détectable, il faudrait que les résidus r_7 et r_8 soient tous deux différents de 0. Donc nous sommes en présence d'un manque à la détection.

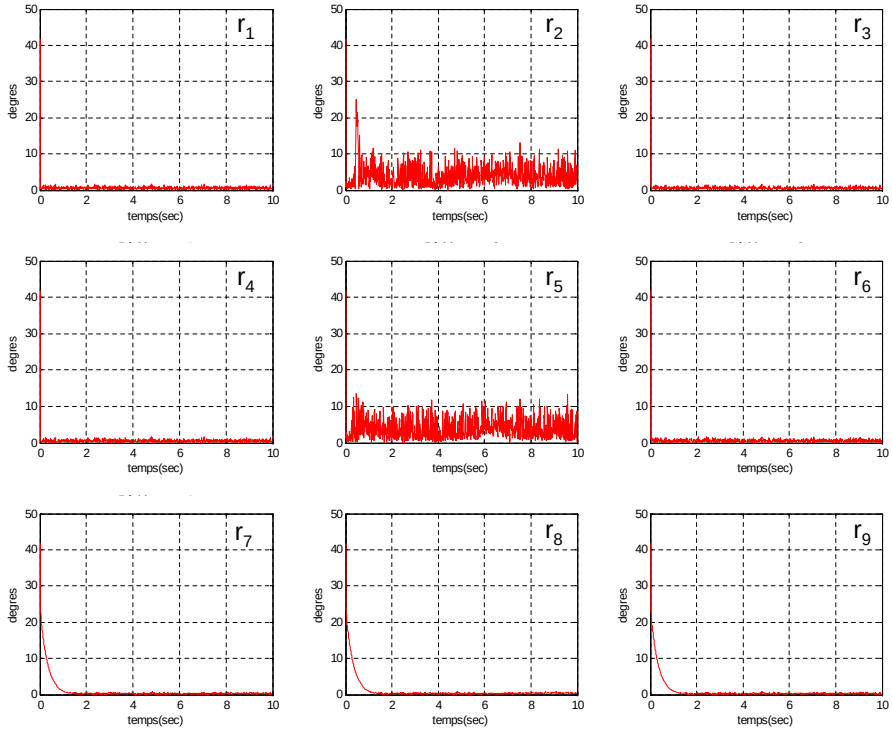


FIG. 5.28 – Les 9 résidus obtenus par *Algo 1* : défaut sur $gyro_Z$ non détecté (scénario 4)

5.4.2 Génération des résidus par *Algo 2*

La figure 5.29 montre les résidus $r_\omega(i)$ $(i = 1..3)$ pour la surveillance des gyromètres. Après $t = 5\ s$, seuls deux des trois résidus sont nuls car insensibles au défaut et le dernier résidu est différant de zéro. L'étape de localisation est donc réalisable (grâce à la table de signature (table 4.7) définie page 81). De plus, on remarque que le résidu est identique aussi bien en amplitude qu'en forme au défaut introduit donc l'étape d'identification du défaut est réalisée. Ce défaut est donc détecté, localisé et identifié.

5.5 Scénario 5 : avec perturbation à $t = 5\ s$ et sans défaut

Dans la section 3.8.2, le comportement du système a été étudié en présence d'une perturbation, la figure 3.12 montrant le résultat de simulation obtenu. Ici, nous allons

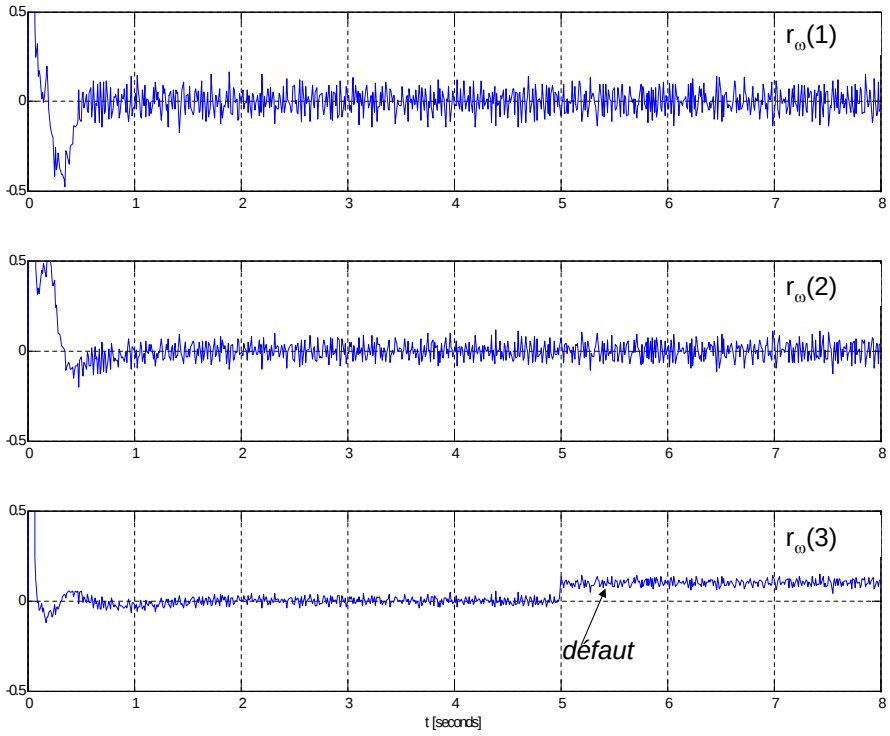


FIG. 5.29 – Résidus générés par *Algo 2* : défaut sur $gyro_Z$ détecté (scénario 4)

étudier le comportement de chaque module de diagnostic pour cette même perturbation (perturbation de 2 degrés sur chaque angle du quadrotor introduite à l'instant $t = 5\ s$, voir paragraphe 3.8.2), en l'absence de défaut.

5.5.1 Génération des résidus par *Algo 1*

La figure 5.30 montre le résultat de simulation avec cette perturbation. On peut constater de ce module de diagnostic est sensible à la perturbation. Ceci s'explique par le fait que les résidus sont générés avec le modèle du quadrotor. Vu que la perturbation est appliquée au système et non au modèle, les deux attitudes obtenues sont différentes. Par conséquent, il est normal que tous les résidus soient différents de zéros.

5.5.2 Génération des résidus par *Algo 2*

Maintenant, le module de diagnostic basé sur le modèle de la centrale d'attitude est testé. Les figures 5.31 à 5.36 montrent les résultats de simulation pour chaque vecteur de résidus. On peut s'apercevoir que chaque vecteur est proche de zéro. Par conséquent, nous pouvons dire que chaque vecteur est insensible à la perturbation. Ceci s'explique par le fait que pour générer les résidus, uniquement les mesures des capteurs et le modèle

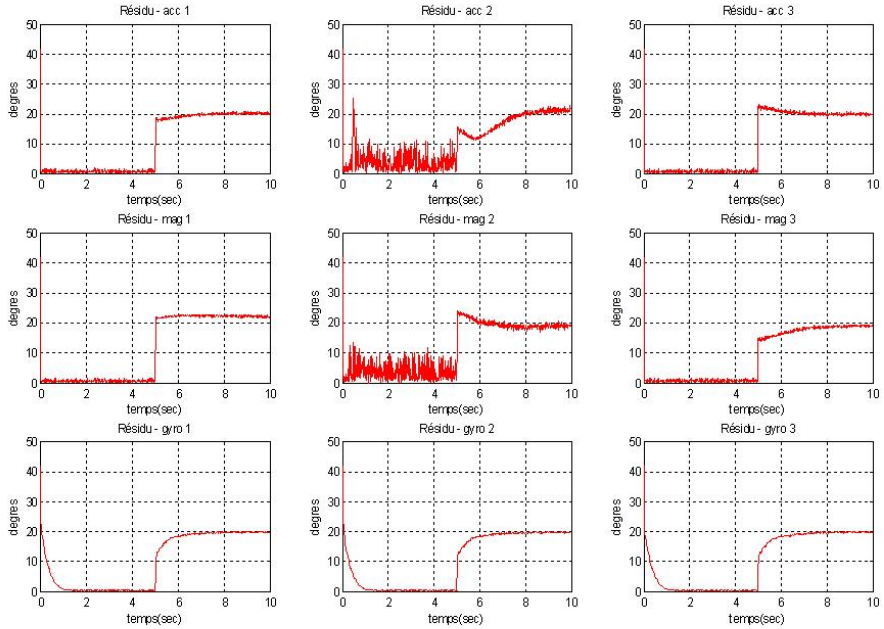


FIG. 5.30 – Les 9 résidus obtenus par *Algo 1* avec une perturbation à partir de $t = 5\ s$ (scénario 5)

des mesures sont utilisés.

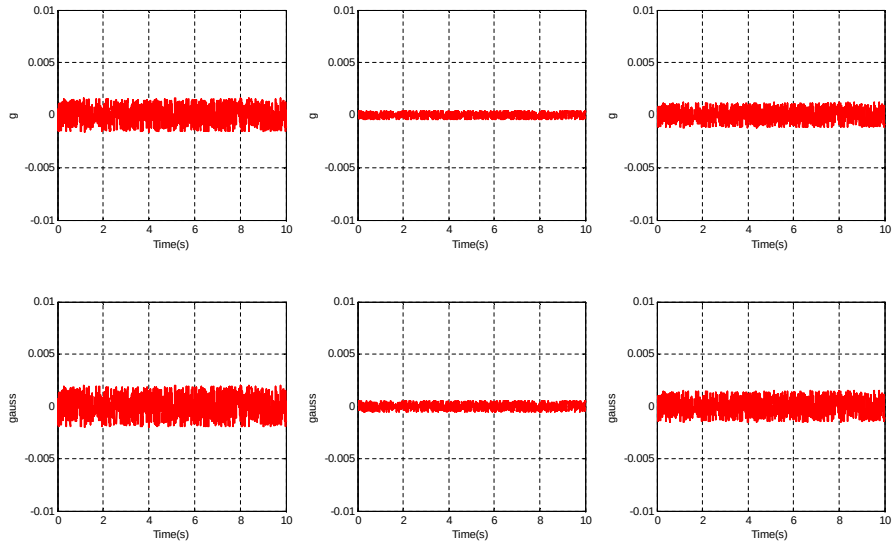


FIG. 5.31 – Les six résidus R_1 (en écartant acc_X) (scénario 5), R_1=[0 0 0 0 0 0]

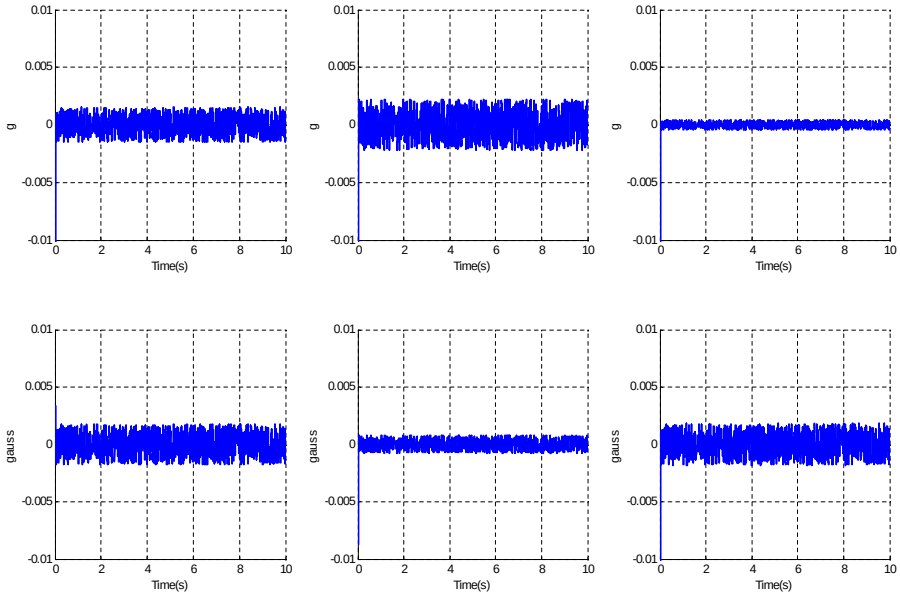


FIG. 5.32 – Les six résidus R_2 (en écartant acc_Y) (scénario 5), R_2=[0 0 0 0 0 0]

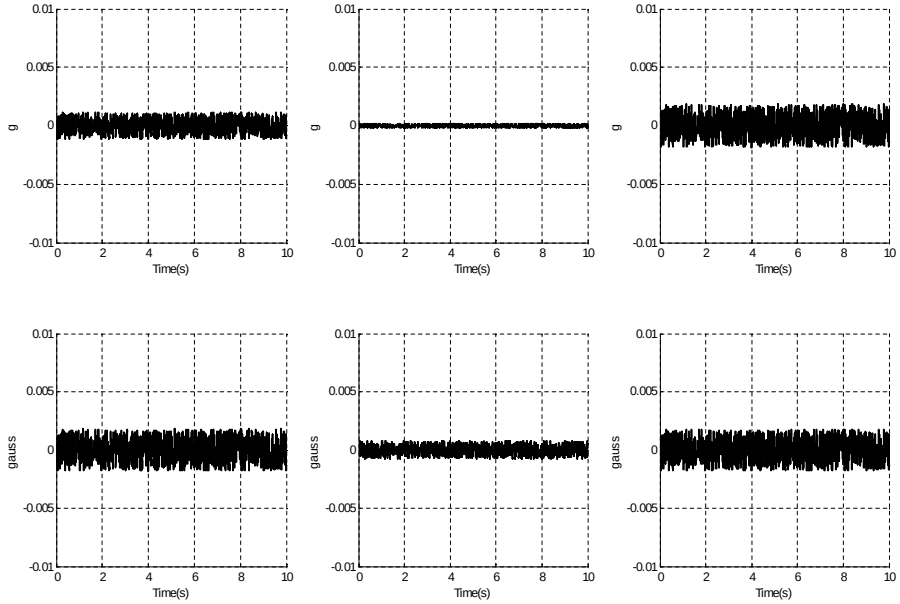


FIG. 5.33 – Les six résidus R_3 (en écartant acc_Z) (scénario 5), R_3=[0 0 0 0 0 0]

5.6 Conclusion

Dans ce chapitre, des résultats (obtenus par simulation avec Matlab/Simulink) ont été présentés pour illustrer les deux algorithmes développés dans le chapitre précédent de ce manuscrit. L'approche de diagnostic par *Algo 2* donne de meilleurs résultats sur plusieurs points :

– les défauts sont détectables, localisables mais aussi identifiables contrairement à

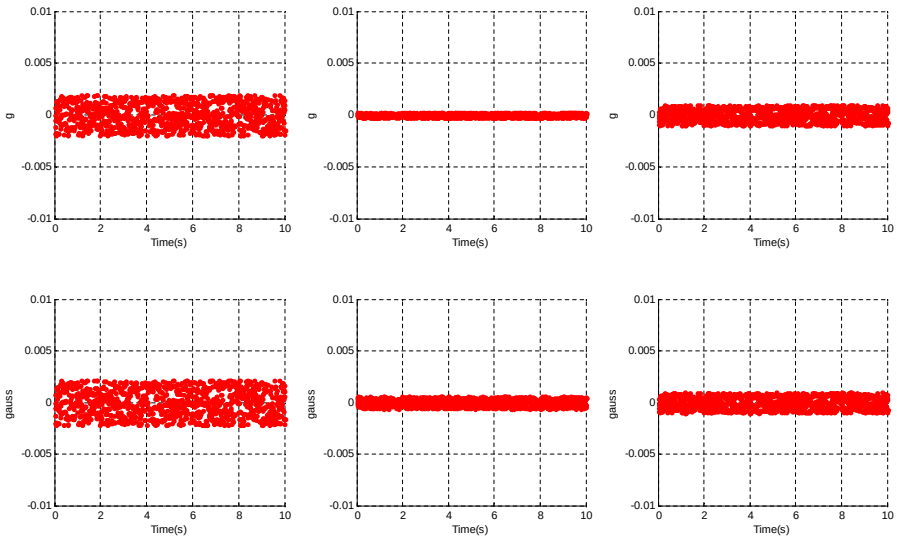


FIG. 5.34 – Les six résidus R_4 (en écartant mag_X) (scénario 5), R_4=[0 0 0 0 0 0]

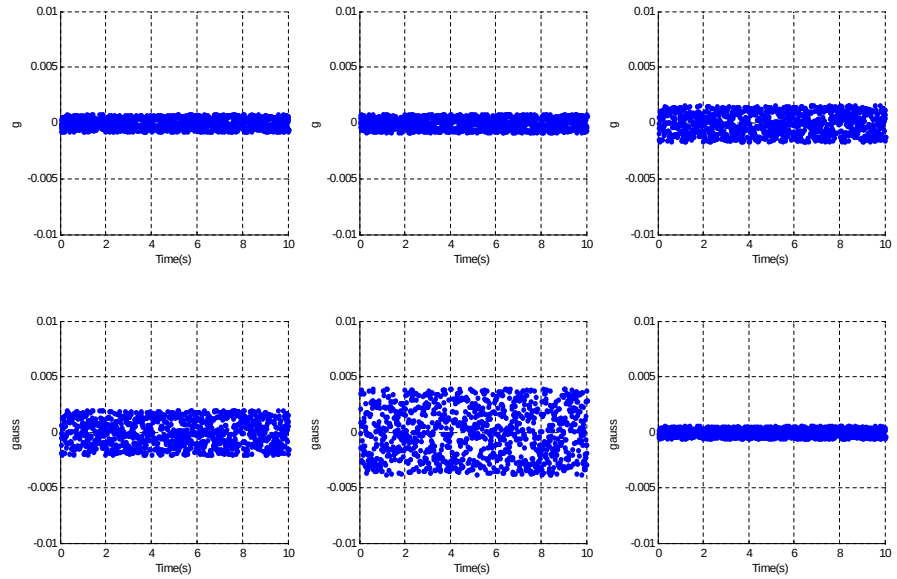


FIG. 5.35 – Les six résidus R_5 (en écartant mag_Y) (scénario 5), R_5=[0 0 0 0 0 0]

Algo 1 ;

- avec *Algo 2*, les défauts multiples voire simultanés sont détectables, localisables mais aussi identifiables ;
- *Algo 2* est un algorithme de diagnostic générique. En effet, celui-ci peut être utilisé sur n'importe quel système car il ne dépend pas de celui-ci. Il est donc insensible à l'inertie, aux erreurs engendrées par l'utilisation d'un modèle ;
- enfin *Algo 2* est insensible aux perturbations comme le montre le scénario 5 (voir section 5.5). Ce résultat est très intéressant en vue d'une utilisation sur une structure réelle.

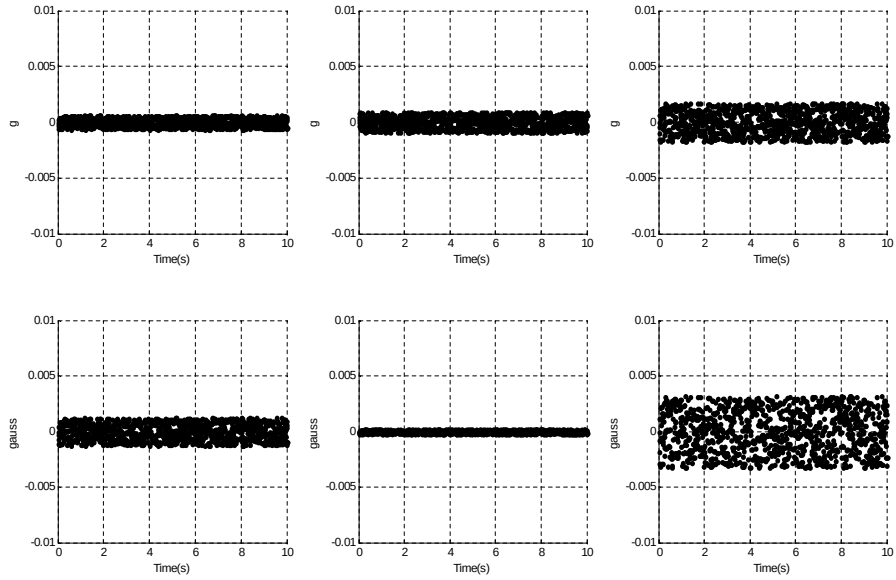


FIG. 5.36 – Les six résidus R_6 (en écartant mag_Z) (scénario 5), R_6=[0 0 0 0 0 0]

Il nous revient maintenant de montrer comment faire évoluer les algorithmes de commande et de diagnostic lorsque le système est contrôlé en réseau. Les solutions étudiées dans les chapitres suivant sont générales. Cependant, elles seront toutes illustrées sur l'exemple du quadrotor pour montrer leur faisabilité sur un système non trivial.

Chapitre 6

Réseaux

Ce chapitre présente différents réseaux étudiés à ce jour en vue de leur implémentation sur le quadrotor. L'intérêt de la mise en réseau des systèmes par rapport aux systèmes centralisés est de disposer d'une plus grande flexibilité, mais aussi de rendre des services distants. La distribution introduit un nouveau composant, le système de communication, qui n'est pas transparent [15] et conditionne le fonctionnement des applications distribuées. L'introduction du réseau a une incidence sur la conception du contrôle-commande ou du diagnostic. En effet, cette conception doit prendre en compte la qualité de service (QoS) que peut fournir le réseau telle que la disponibilité, les délais ou bien les pertes d'informations.

Si depuis quelques années nous voyons de nouveaux termes apparaître tels que "*systèmes contrôlés en réseaux*", la problématique trouve ses fondements dans les années 1980. Quelques précurseurs sont à l'origine de la communauté française "*réseaux locaux industriels*" tel que Guy Juanole, ou internationale comme Karl Johan Aström [76]. Ils rapprochent "*qualité de contrôle* " (QoC) et "*qualité de service*" (QoS). Cette nouvelle problématique induit l'émergence de nouveaux groupes de travail permettant aux différentes communautés de se rencontrer (systèmes à retards, systèmes temps-réel, réseau, diagnostic).

Dans un premier temps, nous allons présenter le réseau CAN (Controller Area Network). Ce réseau sera comparé avec le réseau Ethernet et enfin avec le réseau Ethernet Commuté. Notons que le réseau CAN est implanté sur le benchmark quadrotor. Les trois réseaux (CAN, Ethernet, Ethernet commuté) ont été implantés dans la manipulation "hardware in the loop" que nous verrons ultérieurement.

6.1 Définitions

Dans cette section, le vocabulaire va être défini. On appelle :
- nœud un sous-ensemble relié à un réseau de communication et capable de communiquer sur le réseau selon un protocole de communication ;
- valeurs des bits : les bits peuvent avoir deux valeurs logiques complémentaires définies comme dominante pour le 0 ou récessive pour le 1 ;
- message : chaque information est transmise sur le bus à l'aide d'un message de format défini, de longueur variable mais limitée. Dès que le bus est libre (on dit aussi bus "idle"), n'importe quel nœud relié peut émettre un nouveau message ;
- identifiant : l'identifiant d'un message comporte les informations permettant au nœud de s'identifier auprès d'un système ;
- routage : chaque identifiant (identifier) qui possède un certain nombre des bits indique non pas la destination du message mais la provenance des données de celui-ci. Ainsi tous les nœuds reçoivent le message, et chacun est capable de savoir si ce message lui est destiné ou pas ;
- trame de données ou de requête : une trame de données (data frame) est une trame qui comporte des données. Une trame de requête est émise par un nœud désirant recevoir une trame de données ;
- priorités : les identifiants de chaque message permettent aussi de définir quel message est prioritaire sur les autres.

6.2 Le réseau CAN (Controller Area Network)

L'objectif est de présenter les caractéristiques du bus de terrain CAN (Controller Area Network). Celui-ci a été initialement créé par Bosch pour les applications automobiles avec pour principaux objectifs la fiabilité et un faible coût.

Pour l'histoire, Mercedes-Benz a été le premier constructeur automobile à équiper de CAN ses véhicules et plus particulièrement ses modèles haut de gamme tels que la Classe S. Depuis, de nombreux constructeurs automobiles utilisent le réseau CAN. L'association des constructeurs de véhicules utilitaires en Amérique du Nord l'a adopté comme standard de bus de terrain [22].

Plusieurs associations d'utilisateurs se sont créées, cependant, la plus importante est certainement l'association CiA [50] qui est composée de sociétés et d'institutions (environ 500 aujourd'hui).

6.2.1 Le protocole CAN

Le protocole CAN est un protocole de communication série qui supporte des systèmes temps réel avec un haut niveau de fiabilité. Il existe pour le moment 2 normes couvrant la couche 2 du modèle OSI [99] :

- le CAN standard ou CAN 2.0 A : avec un identifiant d'objet codé sur 11 bits (utilisé comme priorité), il permet d'accepter théoriquement jusqu'à 2 048 types de messages (limité à 2 031 pour des raisons historiques) ;
- le CAN étendu ou CAN 2.0 B : avec un identifiant d'objet codé sur 29 bits, qui permet d'accepter théoriquement jusqu'à 536 870 912 types de messages.

Ces 2 normes sont compatibles, c'est-à-dire qu'il peut circuler sur un même réseau des messages suivant la norme 2.0A et des messages suivant la norme 2.0B. La structure même du protocole CAN possède implicitement les principales propriétés suivantes [11] :

- hiérarchisation des messages ;
- garantie des temps de latence ;
- souplesse de configuration ;
- réception de multiples sources avec synchronisation temporelle ;
- fonctionnement multimaître ;
- détection et signalisation d'erreurs ;
- retransmission des messages altérés dès que le bus est de nouveau au repos ;
- déconnection automatique des nœuds défectueux ;

Remarque 14. On s'aperçoit à travers l'étude de la norme *BOSCH* que le protocole CAN ne couvre que deux des sept couches du modèle des systèmes ouverts OSI [11].

6.2.2 Les couches OSI

Pour le fonctionnement des couches OSI on se reportera à l'annexe C. Le protocole CAN est constitué de la *couche liaison de données* (couche 2) et de la *couche physique* (couche 1). La *couche liaison* de données est constituée de deux sous-couches LLC (*Logic Link Control*), et MAC (*Medium Access Control*), alors que la *couche physique* est divisée en trois sous-couches : PLS (*Physical Signalling*), PMA (*Physical Medium Access*), et MDI (*Medium Dependent Interface*). La sous-couche MAC représente le noyau du protocole CAN. Elle a pour fonction de présenter les messages reçus en provenance de la sous-couche LLC et d'accepter les messages devant être transmis vers la sous-couche LLC. La couche MAC est responsable de :

- la mise en trame du message ;
- l'arbitrage ;
- l'acquittement ;
- la détection des erreurs ;
- la signalisation des erreurs.

La sous-couche LLC s'occupe quant à elle :
- du filtrage des messages ;
- de la notification de surcharge (*Overload*) ;
- de la procédure de recouvrement des erreurs.

La couche physique définit comment le signal est transmis et a par conséquent pour rôle d'assurer le transfert physique des bits entre les différents nœuds en accord avec toutes les propriétés du système (électriques, électroniques...). Bien évidemment, à l'intérieur d'un même réseau, la couche physique doit être la même pour chaque nœud. Cette couche s'occupe donc :
- de gérer la représentation du bit (codage,timing...) ;
- de gérer la synchronisation de bits ;
- de définir les niveaux électriques des signaux ;
- de définir le support de transmission.

6.2.3 La trame

Le format d'une trame CAN est montré sur la Fig.6.1 [28]. Dans le format de CAN, une trame de données est constituée de 47 bits d'entête et de 0 à 64 bits de données regroupé en sept champs différents :
- le début de trame SOF *(Start Of Frame)* ;
- l'arbitrage (*Arbitration Field*) de 12 bits (11 bits pour l'adressage et 1 bit pour différencier une trame de donnée (bit dominant) et une de requête (bit récessif)) ;
- le contrôle (*Control*) : 2 bits réservés pour permettre des extensions et 4 bits pour la taille des données ;
- les données (*Data Field*) : de 0 à 64 bits (0-8 octets) ;
- le CRC (*Cycle Redundancy Code*) : 16 bits ;
- l'acquittement (*ACK*) : 1 bit d'acquittement et 1 bit délimiteur. Le bit d'acquittement est émis au niveau récessif puis chaque station ou nœud ayant reçu la trame sans erreur superposera au bit récessif initial un bit dominant ;
- la fin de trame EOF (*End Of Frame*) : 7 bits récessifs ;
- l'inter-émission (*Int*) : 3 bits récessifs.

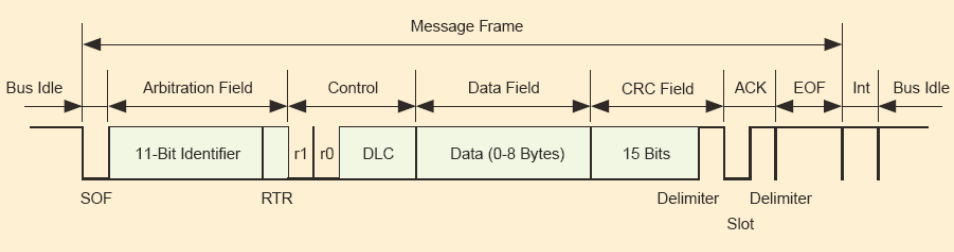


FIG. 6.1 – Format d'une trame CAN [28]

6.2.3.1 L'arbitrage

Une trame CAN est initialement composée des 11 bits pour l'identification et d'un bit pour différencier une trame de données d'une trame de requête (respectivement dominant ou récessif). Ce bit est appelé RTR *(Remote Transmission Request)*. Son rôle est de résoudre un conflit possible lorsque le bus est libre et que deux nœuds souhaitent émettre en même temps. En effet, si plusieurs nœuds désirent utiliser le bus en même temps, on effectue alors un arbitrage bit à bit tout au long du contenu de l'identifiant. Ce mécanisme garantit qu'il n'y aura pas de perte d'information. Lorsqu'un bit récessif est envoyé et qu'un bit dominant est observé sur le bus, l'unité considérée perd l'arbitrage et ne doit plus envoyer aucun bit. L'arbitrage est qualifié de CSMA/CA *(Carrier Sense Multiple Access-Collision Avoidance*).

6.2.3.2 Le contrôle

Le contrôle est composé de 6 bits (Fig.6.2) [11]. Les deux premiers bits (r1 et r0) sont utilisés pour assurer la compatibilité avec des trames étendues.

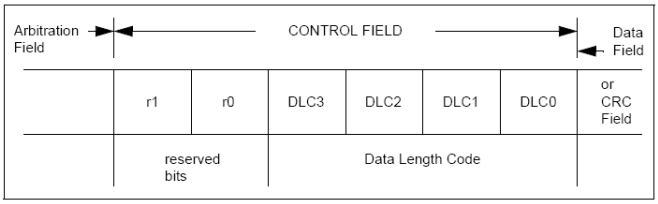


Fig. 6.2 – Format du champ Contrôle [11]

Les quatres derniers bits sont regroupés sous l'appellation DLC *(Data Length Code)*, voir la table. 6.1. Ils permettent de déterminer le nombre d'octets de données contenus dans le champ de données pour une trame de données ou bien le nombre de données nécessaires pour un nœud contenu dans une trame de requête.

6.2.3.3 Les données

Les données ont une longueur qui peut varier de 0 à 64 bits, soit de 0 à 8 octets. Cette longueur est codée en créant le champ DLC. Dans le cas d'une trame de requête, ce champ est nul.

Taille des données (en octets)	DLC3	DLC2	DLC1	DLC0
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0

TAB. 6.1 – Codage des bits DLC en fonction de la taille des données

6.2.3.4 CRC

Le CRC est composé de 15 bits et d'un bit délimiteur récessif de fin de CRC. Il permet de s'assurer de la validité du message transmis. Les champs utilisés pour le calcul du CRC sont : SOF, Arbitrage, Contrôle et enfin des Données. D'après la norme Bosch [11], le calcul du CRC est le suivant :

- le flot de bits, constitué des bits depuis le début de la trame jusqu'à la fin du champs de données est interprété comme un polynôme $f(x)$ avec des coefficients 0 ou 1 affectant la présence ou non de chaque bit.
- le polynôme obtenu est divisé par le polynôme générateur :

$$g(x) = x^{15} + x^{14} + x^{10} + x^7 + x^4 + x^3 + 1 \tag{6.2.1}$$

- le reste de la division entre $f(x)$ et $g(x)$ forme le CRC de 15 bits.

6.2.3.5 L'acquittement

L'acquittement est constitué de 2 bits. Le premier bit est l'acquittement par l'ensemble des nœuds ayant reçu le message. Si aucune erreur n'a été détectée par un nœud (grâce au CRC), ce dernier émet un bit dominant. Le second bit est un délimiteur d'acquittement qui doit toujours être récessif.

6.2.3.6 La fin de trame

La fin de trame (EOF) est constituée de 7 bits récessifs.

6.2.4 La topologie

CAN utilise une topologie de type bus. Le médium de transmission le plus utilisé est la paire torsadée avec des niveaux électriques conformes au standard ISO. Par conséquent, la longueur maximale est d'environ 40 m pour un débit de transmission de 1 $Mbits/s$. Pour atteindre la distance de 1000 m, on peut réduire le débit à 50 $kbits/s$. Des solutions existent en fibre optique mais l'utilisation de celle-ci ne permet pas d'augmenter la longueur de réseau.

6.2.5 Principe de fonctionnement

Le réseau CAN est un réseau au sein duquel des informations de priorités différentes sont transmises selon le principe de diffusion avec un système d'arbitrage de type CSMA/CR (écoute de chaque station avant de parler mais pas de tour de parole, résolution des collisions par priorité).

Quand le bus est disponible, toutes les stations peuvent commencer leur transmission grâce à l'entête de trame. Celle-ci contient l'identifieur qui lui est spécifiquement associé. L'arbitrage n'a lieu que sur cet identifieur. Quand plusieurs nœuds veulent utiliser le médium afin d'émettre leurs informations en même temps, les bits récessifs de l'identifieur d'un nœud sont masqués par les bits dominants de l'identifieur le plus prioritaire d'un autre nœud.

Chaque émetteur écoute le bus et bascule en mode réception dès qu'il détecte un bit dominant. L'arbitrage est gagné par le nœud dont l'identifieur avait la plus haute priorité : il utilise le médium. Les nœuds qui ont perdus l'arbitrage tentent d'émettre à nouveau une fois le bus disponible. On peut constater que l'arbitrage n'est pas destructif.

Remarque 15. Ceci implique que l'information transmise avec la plus haute priorité au sein d'un réseau est retardée dans le pire des cas jusqu'à ce que l'information occupant le bus au moment du souhait de la transmission soit transmise. Par conséquent, le temps écoulé sera donc le temps dit de réaction minimale ou temps de latence maximale. En revanche, les informations de faible priorité risquent de ne pas pouvoir être transmises dans le cas d'un réseau fortement chargé.

Remarque 16. Nous verrons plus loin dans ce manuscrit, comment assurer une qualité de contrôle en "jouant" sur cette notion d'arbitrage par la modification des priorités lorsque cela sera nécessaire.

6.2.6 Avantages

CAN utilise un protocole optimisé pour les messages courts. La priorité des messages est spécifiée dans l'arbitrage. Les messages de haute priorité gagnent toujours l'accès au bus durant l'arbitrage. De plus, le délai de transmission pour les messages de haute priorité peut être garanti.

6.2.7 Inconvénients

Le principal problème pour ce réseau en comparaison avec d'autres tel que Ethernet par exemple, est sa vitesse de transmission (maximale de 1 $Mbits/s$). De plus, un autre désavantage est la longueur des données. Celle-ci est au maximum égale à 8 octets.

6.3 Le réseau Ethernet

Cette partie a pour objectif de présenter rapidement le réseau Ethernet avec son principe de fonctionnement.

6.3.1 Le protocole Ethernet

Le protocole Ethernet est un protocole de réseau local à commutation de paquets. La technologie Ethernet se décline dans de nombreuses variantes [100] telles que :

- deux topologies différentes qui sont bus ou étoile ;
- de multi supports permettant de faire usage de câbles coaxiaux, de fils en cuivre à paires torsadées ou de fibres optiques. Une large gamme de débit avec 10 $Mbits/s$, 100 $Mbits/s$, 1 $Gbits/s$ et 10 $Gbits/s$.

Bien que ce réseau implémente la couche physique (PHY) et la sous-couche Media Access Control (MAC) du modèle OSI, le protocole Ethernet est classé dans la couche de liaison, car les formats de trames que le standard définit sont normalisés et peuvent être encapsulés. Ces couches physiques font l'objet de normes séparées en fonction des débits, du support de transmission, de la longueur des liaisons et des conditions environnementales [100].

Depuis les années 1990, on utilise très fréquemment Ethernet sur paires torsadées pour la connexion des postes clients, et des versions sur fibre optique pour le cœur du réseau. Cette configuration a largement supplanté d'autres standards comme le Token Ring, FDDI et ARCNET. Depuis quelques années, les variantes sans-fil d'Ethernet (normes IEEE 802.11, dites « Wi-Fi ») ont connu un fort succès, aussi bien sur les installations personnelles que professionnelles.

Le service sans connexion d'Ethernet est également non-fiable, ce qui signifie qu'aucun acquittement n'est émis lorsqu'une trame passe le contrôle CRC avec succès ou lorsque celle-ci échoue. Cette absence de fiabilité constitue une clé de la simplicité et des coûts modérés des systèmes Ethernet [46]. L'arbitrage utilisé par Ethernet est qualifié de CSMA/CD *(Carrier Sense Multiple Access with Collision Detection)* :

Lorsqu'un ordinateur veut envoyer de l'information, il obéit à l'algorithme suivant [100] :

1. si le média n'est pas utilisé, commencer la transmission, sinon aller à l'étape 4 ;
2. *[transmission de l'information]* si une collision est détectée, on continue à transmettre jusqu'à ce que le temps minimal pour un paquet soit dépassé (pour s'assurer que tous les postes détectent la collision), puis aller à l'étape 4 ;
3. *[fin d'une transmission réussie]* indiquer la réussite au protocole du niveau supérieur et sortir du mode de transfert ;
4. *[câble occupé]* attendre jusqu'à ce que le réseau soit inutilisé ;
5. *[le câble est redevenu libre]* attendre pendant un temps aléatoire, puis retourner à l'étape 1, sauf si le nombre maximal d'essais de transmission a été dépassé ;
6. *[nombre maximal d'essais de transmission dépassé]* annoncer l'échec au protocole de niveau supérieur et sortir du mode de transmission.

En pratique, ceci fonctionne comme une discussion ordinaire, où les gens utilisent tous un médium commun (l'air) pour parler à quelqu'un d'autre. Avant de parler, chaque personne attend poliment que plus personne ne parle. Si deux personnes commencent à parler en même temps, les deux s'arrêtent et attendent un court temps aléatoire. Il y a de bonnes chances que les deux personnes attendent un délai différent, évitant donc une autre collision. Des temps d'attente exponentiels sont utilisés lorsque plusieurs collisions surviennent à la suite les unes des autres.

6.3.2 Les couches OSI

Ethernet est constitué de la *couche physique* (couche 1) et de la sous-couche Media Acess Control (MAC).

6.3.3 Trame

Le format d'une trame de type Ethernet est montré sur la figure.6.3.

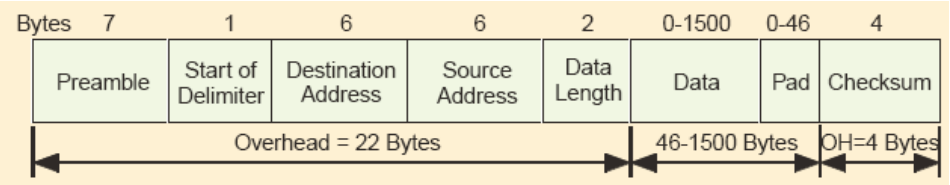


FIG. 6.3 – Format d'une trame Ethernet

Dans le format Ethernet, une trame est constituée de 208 bits (soit 26 octets) d'entêtes et de 368 à 12000 bits (46 et 1500 octets) de données. Une trame est décomposée en huit champs différents :

- préambule (*Preamble*) ;
- début de trame *(Start of Delimiter)* ;
- adresse de destination (*Destination Address*) ;
- adresse de départ *(Source Address)* ;
- longueur des données *(Data Length)* ;
- données (Data) ;
- contrôle *(Checksum)*.

6.3.3.1 Préambule et Début de trame

Le champ préambule est constitué de 7 octets (chacun des 7 octets vaut 10101010). Il a pour fonction de permettre à l'horloge du nœud récepteur de se synchroniser. L'octet représentant le Start of Delimiter vaut 10101011 et indique au nœud récepteur que le début de la trame va commencer.

6.3.3.2 Adresse de destination et Adresse de départ

Chaque champ est codé sur 6 octets et représente respectivement l'adresse des nœuds de destination et de départ.

6.3.3.3 Longueur des données

Ce champ (2 octets) permet d'indiquer quel type de protocole est utilisé (ex : IPv4,IPv6). Pour plus de détails on verra l'annexe D pour l'IPv4 et l'annexe E pour l'IPv6.

6.3.3.4 Données

Ce champ (46 à 1500 octets) contient les données. Si la taille des données est inférieure à 46 octets, alors elles devront être complétées avec des octets de bourrage (padding) et la couche réseau sera chargée de les éliminer.

6.3.3.5 Contrôle

Sur quatre octets, il permet au nœud destinataire de détecter toute erreur pouvant s'être glissée au sein d'une trame. Ce champ est le résultat d'un calcul polynomial entre les champs adresse de destination, adresse de départ, longueur des données et données.

6.3.4 Avantages

Le principal avantage d'Ethernet comme moyen de communication dans les systèmes commandés en réseaux est sa vitesse. En effet, Ethernet possède l'avantage d'avoir un débit allant jusqu'à 10 $Gbits/s$.

6.3.5 Inconvénients

Le protocole Ethernet ne supporte pas les messages à priorité. Lorsque le réseau est chargé, les collisions entre les messages deviennent un problème majeur car cela affecte grandement les retards. Quelques solutions sont proposées pour utiliser Ethernet dans les applications de commande. Par exemple, guider les envois par le temps [56]. Une autre solution est d'utiliser Ethernet commuté en divisant le réseau en sous réseau. Le réseau Ethernet commuté est maintenant présenté.

6.4 Le réseau Ethernet Commuté

6.4.1 Présentation

Pour résoudre les problèmes liés à Ethernet, on peut utiliser Ethernet commuté comme moyen de communication. Des premières études s'intéressant aux retards induits par ce réseau ont été réalisées au sein du CRAN de Nancy, notamment dans [35] et [34].

Dans [12], une étude a été réalisée afin de montrer qu'Ethernet commuté est un bon candidat pour les systèmes commandés en réseaux. Dans un réseau Ethernet com-

muté, les améliorations apportées ont pour but d'augmenter la vitesse du réseau, l'isolation de traffic et de réduire l'impact de l'indéterminisme provenant essentiellement du *CSMA/CD*. La configuration adoptée est celle dite d'une topologie en étoile avec un unique commutateur dont les ports sont reliés à un seul équipement. L'interconnexion avec les autres éléments est réalisée en reliant les commutateurs entre eux. Cette topologie a l'avantage de réduire par segmentation les domaines de collision au seul lien point à point entre un équipement et son commutateur, ou bien entre deux commutateurs [12].

Bien évidemment, dans le réseau Ethernet commuté, l'équipement le plus important est le commutateur qui est régi par trois fonctions :

1. le relai et le filtrage des trames ;
2. la mise à jour des informations permettant de remplir le rôle précédent ;
3. une surveillance de son fonctionnement interne.

6.4.2 Principe de fonctionnement

Le principe de fonctionnement du réseau Ethernet commuté est le suivant :
- quand un message arrive à un port du commutateur, il est mis en mémoire dans un buffer, puis analysé et classé ;
- ensuite, le message est mis dans un buffer qui correspond à son port de destination (figure 6.4) ;
- le bloc "manipulation de paquets" transfère les messages du port d'entrée au port de sortie. Quand le taux d'arrivée des messages de chaque port est supérieur au taux de départ, les messages sont mis dans une file d'attente ;
- les messages mis dans une file d'attente sont transmis d'une façon séquentielle suivant la politique FIFO (First In First Out). La file d'attente peut mener à un retard induit par le réseau puisque les messages temps réel peuvent être bloqués durant la transmission de messages non prioritaires. Pour contrer ce problème, l'utilisation de files d'attente parallèles en sortie de chaque port du commutateur a été proposée.

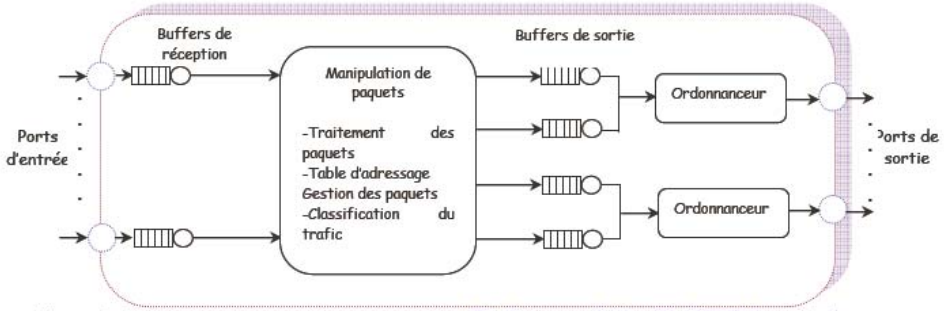


FIG. 6.4 – Architecture d'un commutateur [12]

Le nombre de priorités est limité à huit, permettant l'utilisation de politiques d'ordonnancement. Plusieurs opérations sont accomplies par le commutateur quand une trame est reçue [12] :

- réception de la trame ;
- destruction des trames corrompues, ou de celles qui ne sont pas du bon type ;
- filtrage de la trame suivant les informations de filtrage : il y a destruction si nécessaire ;
- aiguillage vers les autres ports du commutateur ;
- choix de la classe de la trame, puis mise en place dans la file d'attente correspondante ;
- destruction des trames ayant attendu trop longtemps ;
- sélection de la trame à transmettre ;
- émission de la trame.

Au sein du commutateur, on peut introduire une politique d'ordonnancement afin de limiter les collisions. Lors des collaborations entre les partenaires du projet *Safe-Necs*, la politique choisie pour l'implantation sur le quadrotor est celle dite de " *Weighted Round Robin* (WRR)" [25].

6.5 Conclusion

Avec ce chapitre, quelques réseaux intéressants pour les applications embarquées ont été présentés, chacun d'eux possédant leurs avantages et inconvénients. Les inconvénient du réseau Ethernet ne sont pas négligeables pour les applications de commande et de diagnostic. En effet, le fait que Ethernet ne possède pas de mécanisme basé sur les priorités, rend très difficile, en présence d'urgence, de satisfaire les demandes du système à commander. De ce fait, le réseau CAN semble, quant à lui, mieux adapté pour répondre à ce type d'application. Grâce à son mécanisme de priorité, il est possible à tout moment, d'agir sur le système en cas de problème. Nous illustrerons ceci dans le prochain chapitre grâce à l'application du quadrotor.

Chapitre 7

Commande et diagnostic des systèmes commandés en réseau

Dans ce chapitre, une description générale des systèmes commandés en réseau sera présentée. Dans une deuxième temps, nous mettrons en avant les problèmes liés à l'utilisation d'un réseau dans un système en boucle fermée à travers un exemple simple que ce soit pour la commande ou pour le diagnostic. Dans un troisième temps, nous verrons le cas où le quadrotor présenté dans le chapitre 3 est commandé en réseau. Dans cette partie, nous verrons tout d'abord l'aspect "*control over the network*", les algorithmes de commande et de diagnostic (présentés dans les paragraphes 3.6 et 4.2) seront implantés. Ensuite, l'aspect "*control of network*" sera mis en avant. Dans cette dernière partie, nous verrons comment gérer la communication afin de garantir une bonne qualité de contrôle du système.

7.1 Systèmes commandés en réseau : introduction

Un système est appelé "*système commandé en réseau*" lorsque celui-ci communique avec ses algorithmes de commande et de diagnostic à travers un médium de communication en temps réel. Avec ce type de système, de nouvelles problématiques doivent être considérées. En particulier, il faut contrôler et adapter le système de communication et/ou l'application ([110], [93]). Deux approches doivent donc être considérées :

- ***contrôle-commande du réseau pour répondre aux caractéristiques requises par l'application*** : l'objectif est de gérer les communications au mieux à partir d'une qualité de service *QoS* souhaitée (i.e. la bande passante, la gigue[1], le retard et la perte d'informations). Cette approche a le nom de "*control of the network*" ;

[1] variations des instants par rapport aux positions qu'ils devraient occuper initialement.

– ***adaptation de l'application aux performances du réseau :*** dans cette approche, l'objectif est de garantir les performances de l'application malgré l'insuffisance du médium de communication (i.e. stabilité, robustesse et tolérance aux fautes). On peut aussi intégrer des modes "dégradés" pour tolérer et surtout être robuste face aux problèmes induits par le réseau. Cette approche a le nom de "*control over the network*".

Ces deux approches sont complémentaires et aujourd'hui on s'intéresse à la combinaison des deux. Dans ce cas, on parle de ***co-design*** ou *co-conception*. Alors, la qualité de service est considérée simultanément avec la performance souhaitée pour le système.

7.1.1 Implantation

Les systèmes commandés en réseau se décomposent non pas en deux parties mais en trois :

1. le procédé : le terme procédé regroupant le système lui-même mais aussi les capteurs et actionneurs ;
2. le contrôleur : le terme contrôleur regroupe aussi bien la loi de commande que le diagnostic ;
3. le réseau : utilisé comme médium de communication entre le procédé et le contrôleur.

Plusieurs structures peuvent être considérées :

– structure 1 : cette structure est représentée sur la figure 7.1A). Le réseau est utilisé uniquement pour faire communiquer les capteurs et le contrôleur. Dans ce cas-ci, le retard induit par le réseau influence uniquement le passage des informations des capteurs au contrôleur ;
– structure 2 : cette structure est représentée par la figure 7.1B). Le réseau est utilisé uniquement pour faire communiquer les actionneurs et le contrôleur. Dans ce cas-ci, le retard induit par le réseau influence uniquement le passage des informations du contrôleur aux actionneurs ;
– structure 3 : cette structure est représentée par la figure 7.1C). Le réseau est utilisé dans la boucle fermée entre le procédé et le contrôleur ou le module de diagnostic. Cette structure est la plus utilisée et l'influence du réseau est dans ce cas maximale. En effet, le retard induit par le réseau intervient sur l'ensemble du système.

Dans la suite de l'étude, la structure 3 est prise en compte. Dans ce cas, le système fonctionne de la façon suivante avec deux types d'architectures [89] (voir figures 7.2 et 7.3) :

1. à chaque période d'échantillonnage, les informations mesurées par les capteurs sont numérisées à l'aide d'une conversion analogique numérique (A/D) pour être encapsulées dans des trames afin d'être transmises au contrôleur à travers le réseau ;

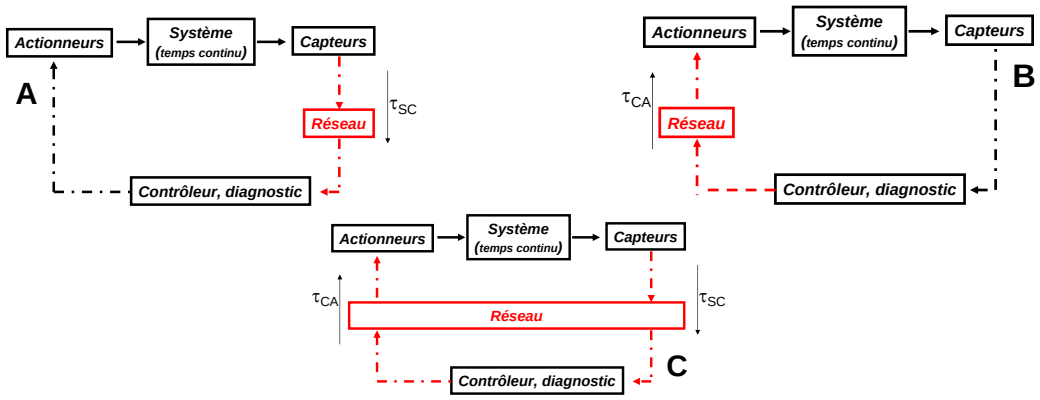


FIG. 7.1 – (haut gauche) Structure 1, (haut droite) Structure 2, (bas) Structure 3

2. le contrôleur lit les valeurs de mesures, et calcule en fonction des valeurs désirées, l'entrée de commande (commande directe) ou consigne d'une boucle locale (commande hiérarchique) à appliquer sur les actionneurs ou encore l'état de bon/mauvais fonctionnement du procédé. Cette étape est bien entendu dans le domaine discret. Les entrées sont ensuite encapsulées à leur tour afin d'être envoyées aux actionneurs (dans le cas de la commande directe) ou aux boucles locales (pour la commande hiérarchique) toujours à travers le réseau ;
3. enfin, les actionneurs récupèrent les consignes et les convertissent en signal analogique grâce à un convertiseur numérique analogique (D/A) et celles ci sont ensuite appliquées à l'entrée de commande du système.

On peut remarquer que les retards peuvent être important. En effet, il y a le retard induit par le médium de communication mais aussi les retards dus aux temps de conversion A/D et D/A, les temps de traitement de la commande et les temps d'encapsulation/désencapsulation des trames. Une des contraintes pour pouvoir contrôler un système est que la somme de ces retards soit inférieure à la période d'échantillonnage du système.

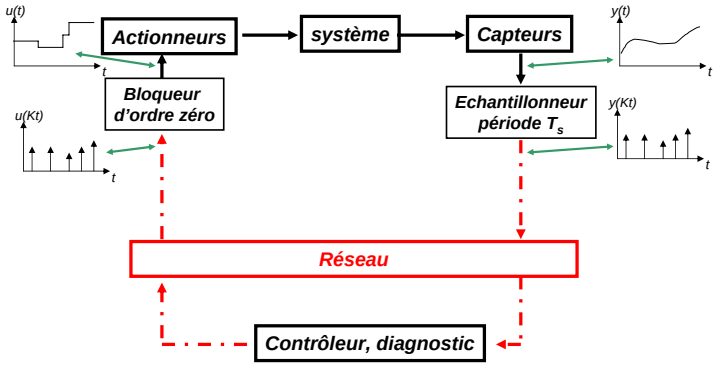


FIG. 7.2 – Structure directe d'un *NCS*

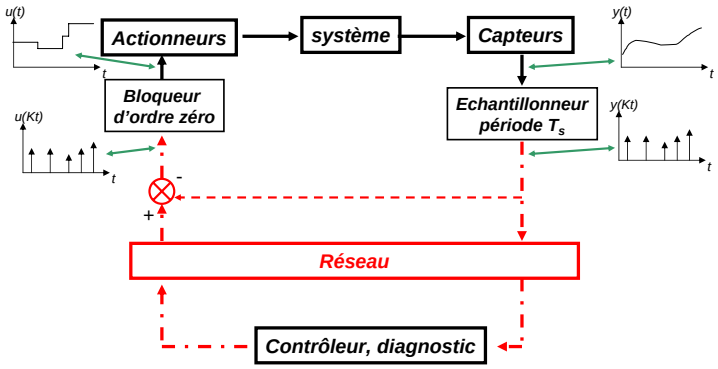


FIG. 7.3 – Structure hiérarchique d'un *NCS*

7.1.1.1 Structure directe

Cette structure est représentée sur la figure 7.2. Dans cette structure, la commande et le procédé sont distants et sont reliés grâce au médium de communication. Un des exemples le plus couramment utilisé est celui de la commande d'un moteur à travers un réseau [88]. Dans [15], ce même exemple est utilisé afin de montrer l'influence du réseau non pas sur la commande mais sur le diagnostic.

7.1.1.2 Structure hiérarchique

La figure 7.3 montre ce type de structure. Périodiquement, le contrôleur principal calcule et envoie le signal de consigne à travers le réseau. Le système attend le signal de consigne avant d'exécuter la boucle de commande locale et envoie les mesures du capteur au contrôleur principal du système de commande en réseau. La boucle de commande en réseau doit en général avoir une période d'échantillonnage plus grande que la boucle de commande locale puisque la commande distante est supposée fournir le signal de référence de celle-ci, qui est mis à jour moins souvent que la commande.

L'utilisation d'une structure dépend de l'application et de sa conception. Par exemple, commander un robot exige l'utilisation de beaucoup de moteurs qui doivent fonctionner ensemble et simultanément. C'est pourquoi, la structure hiérarchique semble plus adaptée [89]. A l'inverse, lorsque le système est plus simple tel qu'un moteur à courant continu, la structure directe peut être choisie.

Remarque 17. Sur le benchmark du quadrotor, la structure hiérarchique est utilisée comme le montre la figure 7.4. Le quadrotor est composé de quatres boucles locales pour la vitesse de rotation de chaque moteur et une boucle globale permettant de fournir le signal de référence des quatres moteurs.

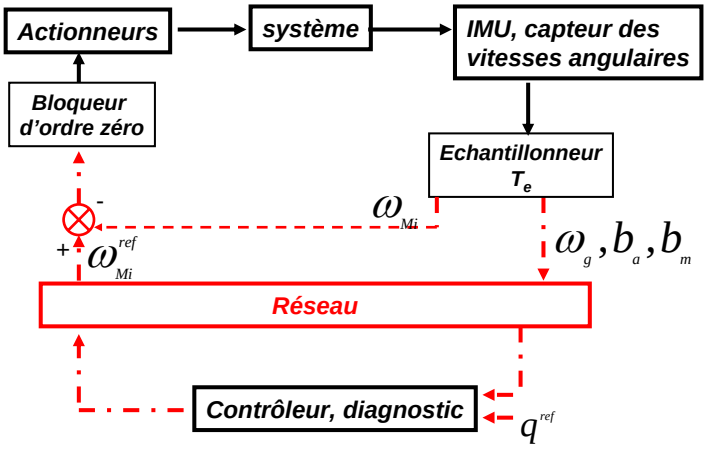


FIG. 7.4 – Structure hiérarchique du quadrotor

7.1.2 Outils de simulation

Plusieurs simulateurs de réseaux existent en particulier *TrueTime* et Simevent qui sont compatibles avec Matlab/Simulink.

Le choix s'est porté sur la boite à outils *TrueTime* car celle-ci est gratuite et disponible via l'adresse internet : *http* : *//www.control.lth.se/truetime/*. Les partenaires du projet *SafeNecs* ont apporté une contribution certaine pour la faire évoluer. En particulier, l'équipe du CRAN a implanté la politique d'ordonnancement du WRR pour le réseau Ethernet Commuté.

7.1.2.1 *TrueTime*

Pour simuler un réseau dans l'environnement Matlab/Simulink, on peut utiliser la boite à outils *TrueTime* développée par des chercheurs Suédois de Lund. *TrueTime* est développé depuis 1999, et est toujours en constante amélioration. Ce simulateur fonctionne sous la forme d'une bibliothèque utilisable sous Simulink comme le montre la figure 7.5. Contrairement à d'autres outils de co-simulation comme Ptolemy [8], *TrueTime* n'est pas basé sur un modèle mathématique [17]. L'application est écrite en langage C++ ou en code Matlab. La différence principale par rapport aux programmes temps réel est que les temps d'exécution ou de transmission doivent être spécifiés par l'utilisateur. Dans la suite, nous présentons quelques concepts de base de cette bibliothèque qui a été utilisée pour la simulation des réseaux lors de notre travail.

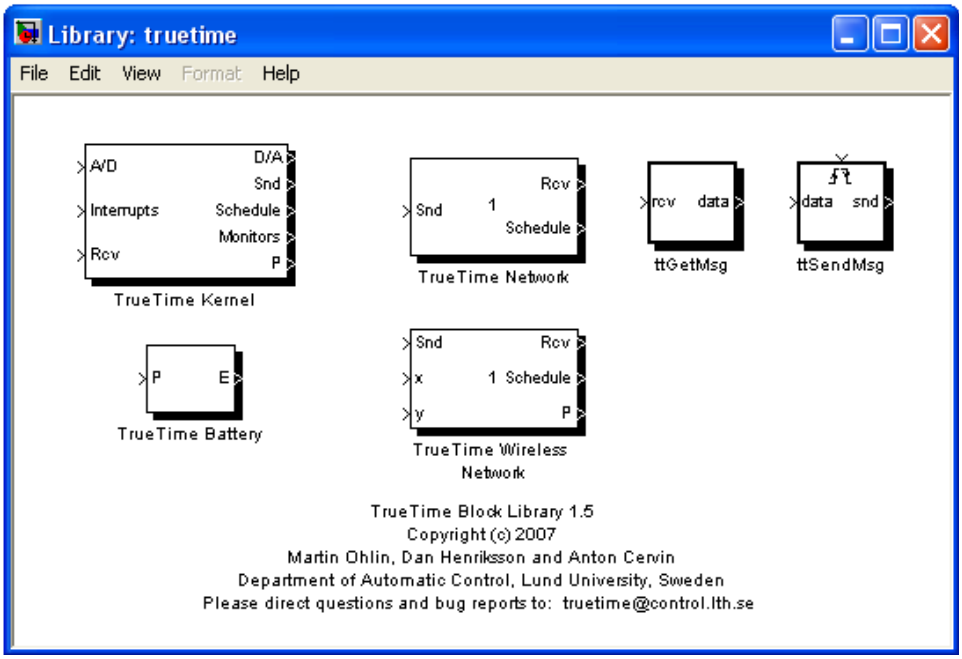


FIG. 7.5 – Bibliothèque TrueTime sous Simulink

7.1.2.2 Le bloc "Kernel Block"

Les "*TrueTime* Kernel Blocks" simulent un nœud avec un noyau temps réel sous une forme générique, des convertisseurs analogique/numérique (A/D) et numérique/analogique (DA) et une interface réseau [72]. Le bloc est configurable via un programme d'initialisation (figure 7.6) qui peut être paramétré. Dans le programme d'initialisation, le programmeur peut créer des objets comme des tâches, des interruptions ou des sémaphores, etc, représentant l'exécution du logiciel dans le nœud.

Durant la simulation, le Kernel appelle les fonctions codes (figure 7.7) utilisant les tâches et les interruptions. Le programme d'initialisation et les fonctions codes peuvent être écrites soit en Matlab soit en C++. Dans ce cas, le programme d'initialisation et les fonctions codes sont compilés en utilisant la commande "MEX" de Matlab.

Les blocs utilisent des algorithmes d'ordonnancement tels que la priorité statique ou l'ordonnancement à partir des durées de vie comme (earliest-deadline-first). Il est possible de spécifier une autre politique d'ordonnancement.

7.1.2.3 Les blocs "Network Block"

Les blocs "*TrueTime* Network" et "*TrueTime* Wireless Network" simulent la couche physique et sous-couche MAC de plusieurs réseaux locaux [72].

```
16/07/09 11:12  C:\Program Files\MATLAB\Toolbox\truetime\exa...\actuator_init.m  1 of 1

function actuator_init

% Distributed control system: actuator node
%
% Receives messages from the controller and actuates
% the plant.

% Initialize TrueTime kernel
ttInitKernel(0, 1, 'prioFP'); % nbrOfInputs, nbrOfOutputs, fixed priority

% Create actuator task
deadline = 100;
prio = 1;
ttCreateTask('act_task', deadline, prio, 'actcode');

% Initialize network
ttCreateInterruptHandler('nw_handler', prio, 'msgRcvActuator');
ttInitNetwork(2, 'nw_handler'); % node #2 in the network
```

FIG. 7.6 – Programme d'initialisation d'un Kernel

```
16/07/09 11:12  C:\Program Files\MATLAB\Toolbox\truetime\examples\...\actcode.m  1 of 1

function [exectime, data] = actcode(seg, data)

switch seg,
 case 1,
  data.u = ttGetMsg;
  exectime = 0.0005;
 case 2,
  ttAnalogOut(1, data.u)
  exectime = -1; % finished
end
```

FIG. 7.7 – Exemple de code

Ces types de réseaux sont CMSA/CD (Ethernet), CSMA/AMP (CAN), Round Robin (Token Bus), FDMA, TDMA (TTP), Ethernet commuté, WLAN (802.11b) et ZigBee (802.15.4). Les blocs simulent seulement l'accès au médium, les possibles collisions ou interférences, et les transmissions point à point ou broadcast. Les couches hautes des protocoles telle que TCP/IP ne sont pas simulées (mais peuvent être implémentées comme application dans les nœuds). Les blocs réseaux sont configurables via une boite de dialogue (figure 7.8). Les paramètres communs à tous les réseaux sont : la vitesse (Data rate), la taille minimale des trames. Pour chaque type de réseau, il y a un nombre de paramètres qui peuvent être spécifiés comme la puissance de transmission pour les réseaux sans fil ou bien les seuils de réception du signal.

De plus, un modèle de simulation avec *TrueTime* peut contenir plusieurs blocs réseau, et chaque Kernel peut être connecté à un ou plusieurs réseaux. Chaque réseau est identifié par un nombre, et chaque nœud connecté au réseau est adressé par un nombre qui est unique pour chaque réseau. Les blocs Réseau peuvent être utilisés de différentes façons. La plus courante est d'avoir un Kernel pour chaque nœud du réseau. Les tâches dans chaque Kernel peuvent émettre ou recevoir des structures de tableaux Matlab à travers

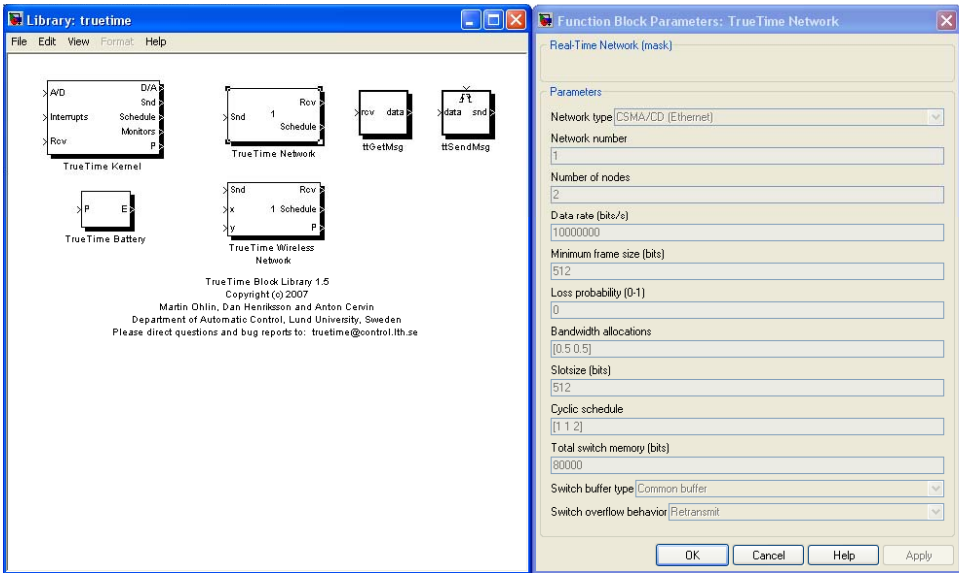


FIG. 7.8 – Paramètres de configuration pour les réseaux

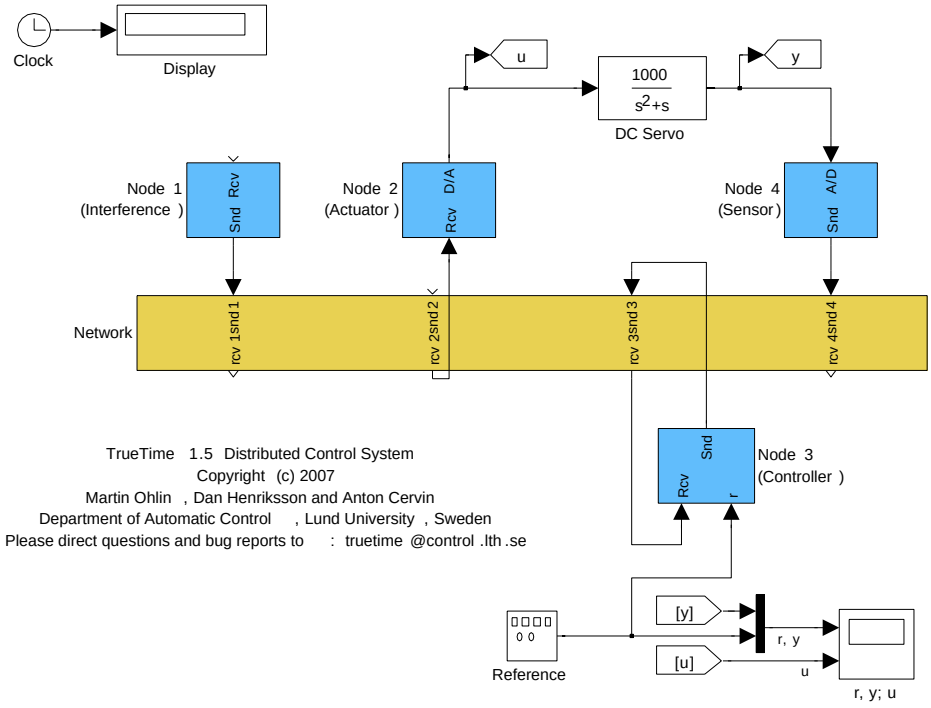


FIG. 7.9 – Exemple de Networked Control Systems avec 4 nœuds et 1 réseau

le réseau en utilisant certaines primitives. Cette approche est très flexible et requiert peu de connaissance en programmation pour configurer le système [18]. Notons que la programmation simule le fonctionnement macroscopique du réseau sans entrer dans le détail des échanges de bits. Notons également que les réseaux simulés ne sont pas en

parfaite cohérence avec la norme les définissant (c'est le cas de Zigbee qui n'implémente pas la phase inactive).

7.2 Influence du réseau : étude de cas avec un moteur à courant continu

L'objectif ici est de mettre en évidence les nouvelles problématiques liées à l'utilisation d'un réseau dans la commande et le diagnostic d'un système en boucle fermée.

Pour cela, considérons un cas d'étude simple à travers l'exemple d'un moteur à courant continu (vu dans le chapitre sur le cas des défauts actionneurs). Cet exemple a été utilisé afin de permettre de comprendre l'influence des caractéristiques du réseau implanté.

7.2.1 Rappel

La figure 7.10 montre le schéma bloc du moteur à courant continu avec sa commande et son module de diagnostic.

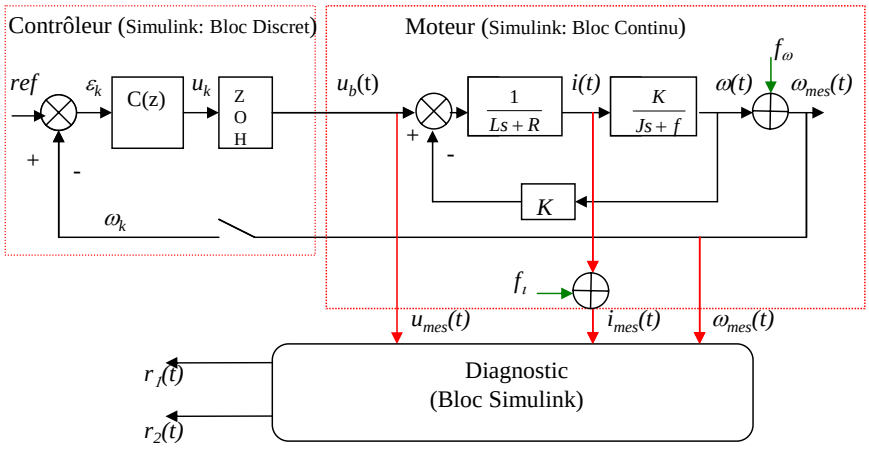


FIG. 7.10 – Moteur en boucle fermée

7.2.1.1 Diagnostic

L'objectif du module de diagnostic est de détecter si un capteur est en défaut. On note f_i un défaut sur le capteur de courant et f_ω un défaut sur le capteur de vitesse.

Pour cela, le module de diagnostic utilise les trois variables mesurées sur le moteur : la tension, le courant ainsi que la vitesse angulaire (voir section 4.4). Les équations des résidus sont :

$$J\frac{d\omega_{mod}(t)}{dt} + f\omega_{mod}(t) = Ki_{mes} \tag{7.2.1}$$

$$L\frac{di_{mod}(t)}{dt} + Ri_{mod}(t) = (u_{mes}(t) - K\omega_{mes}(t) \tag{7.2.2}$$

ou nous rappelons que les indices "mod" sont pour le modèle et "mes" pour les mesures. Les résidus sont définis par :

$$r_1(t) = \omega_{mes}(t) - \omega_{mod}(t) \tag{7.2.3}$$

$$r_2(t) = i_{mes}(t) - i_{mod}(t) \tag{7.2.4}$$

Comme on l'a vu au chapitre 4.4, les deux résidus sont sensibles aux deux défauts. Pour pouvoir localiser, il faut regarder les signes des résidus [24]. La table de signature est

TAB. 7.1 – Table de signature pour détecter les défauts sur les capteurs de vitesse et de courant.

	f_ω	f_i
r_1(7.2.3)	+	-
r_2(7.2.4)	+	+

donnée par le tableau 7.1.

Considérons maintenant le cas où un défaut sur le capteur de vitesse de rotation apparaît à l'instant $t = 1.2s$. Ce défaut est simulé par un échelon d'amplitude 0.5 rad/s, ce qui correspond à 3 % de la valeur nominale.

La figure 7.11 montre le résultat pour les deux résidus.

Ces deux résultats servent de références pour la comparaison avec ceux obtenus dans le cas où le réseau est utilisé.

7.2.2 Influence du réseau sur la commande du système

La figure 7.12 montre le système en boucle fermée. Dans cette structure, le retard induit par le réseau est divisé en deux parties : tout d'abord le retard induit par la communication entre les capteurs et le contrôleur, noté τ_{SC}, et celui correspondant à la communication entre le contrôleur et les actionneurs, noté τ_{CA}. Généralement, afin de faciliter les calculs, ces deux retards sont représentés par un seul retard noté τ et qui est égal à τ=$\tau_{SC} + \tau_{CA}$, [110] [109]. BOZ représente le bloqueur d'ordre zéro. Les constantes de temps du système sont $\tau_1 = 502\ ms$, $\tau_2 = 1.4\ ms$.

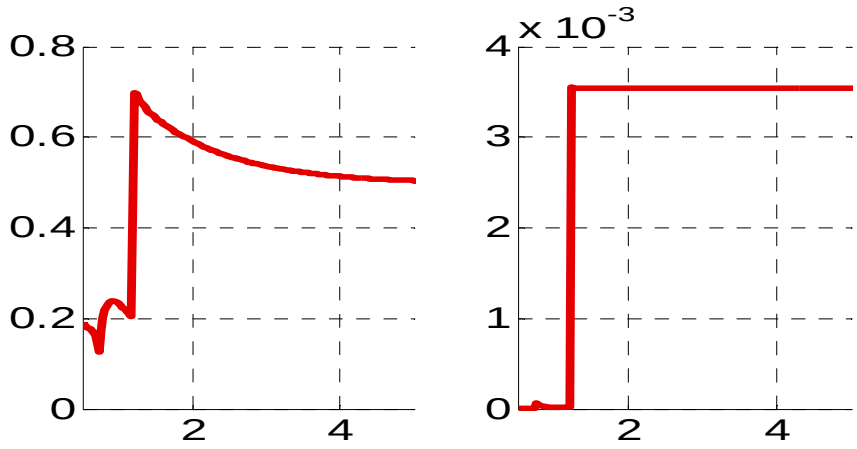


FIG. 7.11 – r_1 et r_2 avec un défaut sur la vitesse angulaire

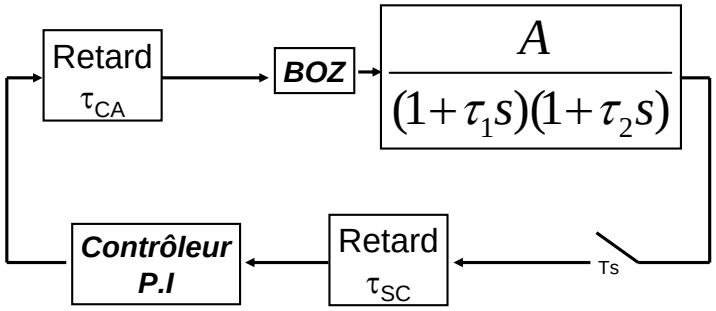


FIG. 7.12 – Moteur en boucle fermée avec l'influence des retards

La figure 7.14 montre la structure du système implantée sous Matlab/Simulink et Truetime.

Les retards induits par l'utilisation d'un réseau peuvent dégrader la réponse du système en boucle fermée. La figure 7.13 montre l'influence du retard sur les performances du système. On sait que plus le retard est important, plus les dégradations des performances du système sont importantes. Les retards dans la boucle de commande déstabilisent le système.

7.2.3 Influence du réseau sur le diagnostic

L'objectif de ce paragraphe est d'évaluer le comportement du module de diagnostic vis-à-vis des performances du réseau. Les résultats de référence sont ceux obtenus dans la section (4.4) auxquels on comparera les résultats lors de l'utilisation du réseau, toutes les autres conditions de simulation étant identiques. Présentons maintenant les caractéristiques du réseau utilisé. Le but dans cette application est de réaliser un réseau dédié. Le réseau choisi est un réseau de type CAN (Controller Area Network) car les performances

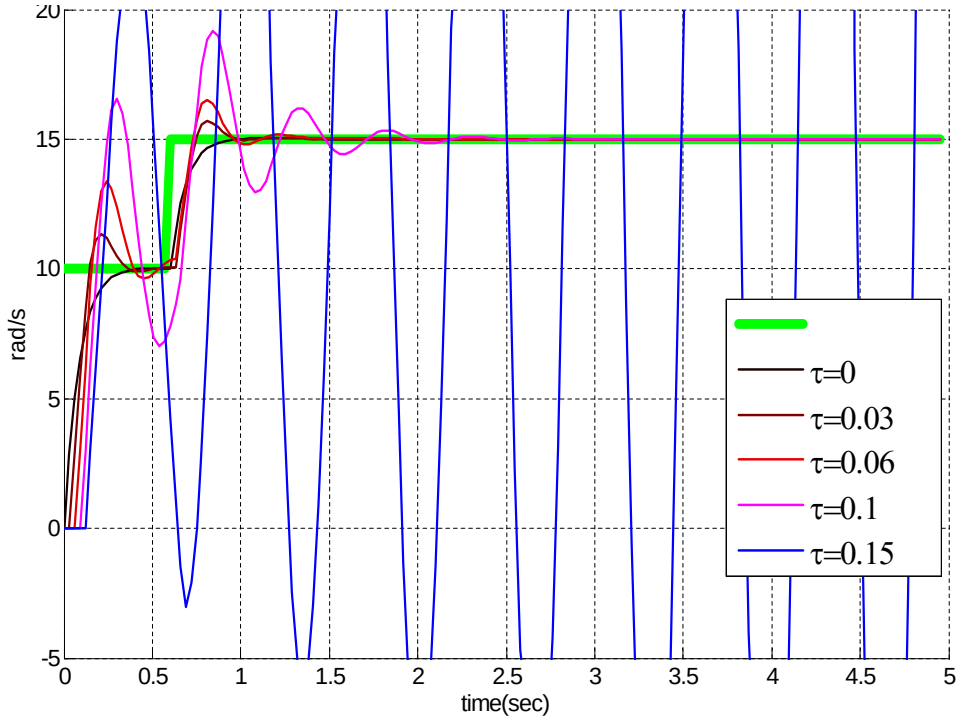


FIG. 7.13 – Réponse du système en boucle fermée avec différents retards induits par le réseau (en seconde)

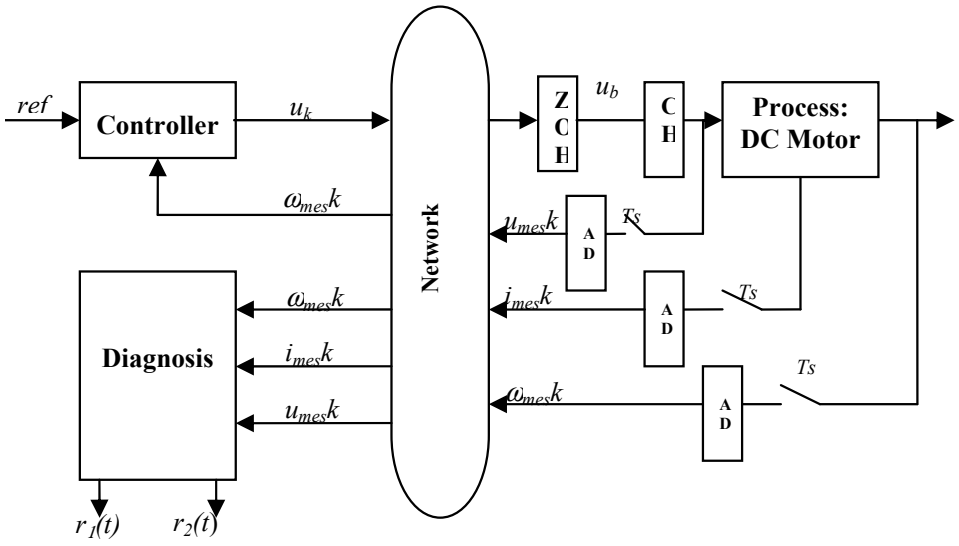


FIG. 7.14 – Architecture du système avec le réseau

qu'il présente sont intéressantes pour notre application par exemple une faible probabilité de perte de paquets. Pour pallier les problèmes de collisions (que l'on rencontre par exemple avec le réseau Ethernet), le réseau CAN utilise la notion de priorité statique lors des envois de données.

Le réseau CAN possède, pour notre application, les caractéristiques suivantes :

1. un débit de 1 $Mbits/s$;
2. 4 flux (3 flux provenant du moteur qui sont guidés par le temps, 1 flux provenant du contrôleur qui est guidé par les événements) :
   - le flux provenant du capteur qui envoie la mesure de la vitesse de rotation ω_{mes} de priorité 11 ;
   - le flux provenant du capteur de courant i_{mes} de priorité 12 ;
   - le flux provenant du capteur de tension u_{mes} de priorité 13 ;
   - le flux provenant du contrôleur de priorité 10.
3. la période d'horloge est de 30 ms ;
4. les données du contrôleur et des capteurs du moteur sont encapsulées dans une trame de longueur 16 $bits$ dont la transmission prend 2 ms. Par conséquent, chaque trame utilise 6.6 % de la capacité du réseau ;
5. UFR (Use Request Factor) du réseau est $UFR = 4 * 6.6\ \% = 24.6\ \%$;
6. toutes les données moteur (ω_{mes}, I_{mes}, U_{mes}) sont cohérentes entre elles car elles sont prises en même temps. Seul l'instant de l'envoi est différent car elles sont envoyées les unes après les autres dans l'ordre de priorité défini.

Ces résultats ont été obtenus par simulation en utilisant le logiciel Matlab/Simulink et la boite à outils *TrueTime*.

7.2.3.1 Fonctionnement normal

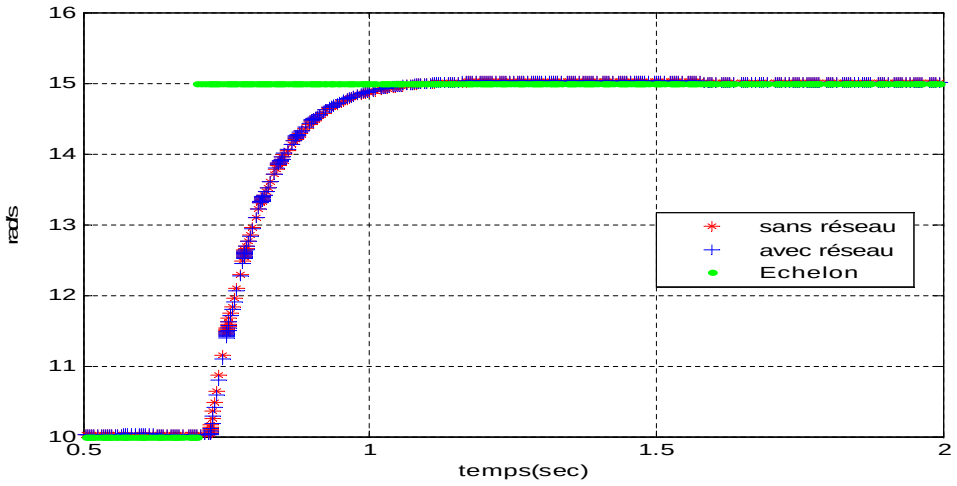


FIG. 7.15 – Réponse indicielle du système avec (en bleu) et sans (en rouge) réseau

La figure 7.15 montre les réponses indicielles obtenues avec (en bleu) et sans (en rouge) réseau. Nous pouvons constater que le réseau qui engendre des retards, n'influence pas la

réponse du système vu que les deux courbes sont confondues. Nous obtenons un temps de réponse d'environ 300 ms et une constante de temps de 100 ms. Ceci est conforme aux spécifications souhaitées. Ce résultat est encourageant pour la suite puisque l'on peut constater que le réseau dédié n'a pas d'influence sur la réponse du système.

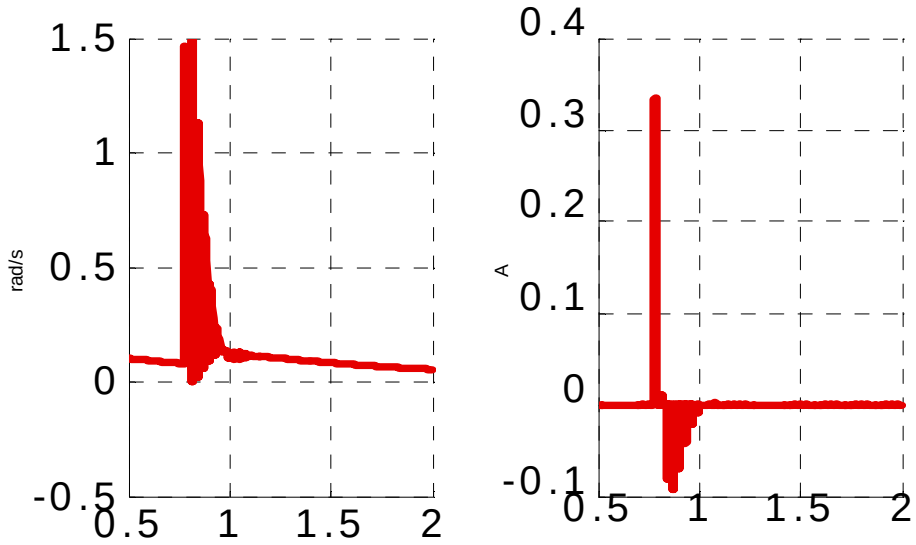


FIG. 7.16 – r_1 et r_2 sans défaut et avec le réseau

La figure 7.16 montre l'influence du réseau sur les résidus si on le considère comme un simple moyen de communication. Il est important de noter que durant le temps de transition, les résidus ont une forte amplitude ce qui correspond à de fausses alarmes puisque les capteurs ne sont pas en défaut. Le problème ici, est que les données ne sont pas synchronisées comme le montre la figure 7.17.

En fonctionnement idéal, les données sont transmises au même instant (figure 7.17.a). Or physiquement, elles ne sont pas transmises simultanément sur le médium de communication (figure 7.17.b). A chaque fois qu'une donnée est reçue, le module de diagnostic calcule les résidus en utilisant cette donnée (figures 7.17.c à .f). Le module de diagnostic est guidé par les événements "*réception de données*".

La première valeur reçue est celle de la vitesse angulaire (7.17.b). Cette valeur est utilisée pour générer r_1 (figure 7.17.d) et i_{mod}. Avec i_{mod}, r_2 est généré. Or les résidus dépendent aussi des autres mesures. C'est pourquoi l'algorithme utilise les dernières valeurs reçues, c'est à dire, celles de la période précédente. Par exemple, nous avons besoin de u_{mes} pour déterminer i_{mod} (7.17.e). Les mesures utilisées pour ce calcul ne sont pas synchrones et par conséquence, le calcul est faux. Le même phénomène est rencontré quand la mesure du courant arrive au module de diagnostic (7.17.c, .d et .f). Les calculs avec les mesures correctes ont lieu uniquement après avoir reçu la dernière

mesure qui est u_{mes} (figure 7.17.f) : l'algorithme doit attendre d'avoir reçu toutes les mesures nécessaire à la génération d'un résidu. Durant les "mauvais" calculs, des fausses alarmes sont générées et leur amplitude n'est pas négligeable. Par exemple la figure 7.18 montre les résidus obtenus sans (à gauche) et avec (à droite) réseau. Ces fausses alarmes apparaissent plus particulièrement pendant les transitions et leur amplitude est importante.

7.2.3.2 Fonctionnement en tenant compte du réseau

On vient de voir que l'introduction du réseau n'est pas neutre sur le module de diagnostic qui est guidé par les événements. Pour cela, il faut modifier l'algorithme de diagnostic : il doit attendre toutes les mesures nécessaires à la génération d'un résidu. Maintenant, la génération des résidus va tenir compte de ce point. Pour cela, les résidus ne seront générés qu'une fois par période. Les figures 7.19 et 7.20 montrent les résultats obtenus.

Il reste quelques instants d'échantillonnage où les résidus ne sont pas complètements nuls malgré l'absence de défauts. Cependant, leur amplitude est faible et la détection du défaut est possible. Les résultats obtenus maintenant sont conformes à ceux obtenus sans réseau. La table de signature nous permet de localiser les défauts (Tableau 7.1).

7.2.3.3 Défaillance du réseau

Comme on vient de le voir, il est nécessaire de considerer le réseau comme un composant du système. Comme tout composant, celui-ci peut avoir des défaillances. Pour cette étude, le terme défaillance est assimilé à la perte de paquets. C'est pourquoi, le réseau doit être diagnostiqué. Nous allons maintenant étudier l'influence d'une perte de paquet sur la réponse du système en boucle fermée et sur les résidus générés pour les défauts capteurs lorsque les algorithmes de commande et de diagnostic ne prennent pas en compte cette perte et affectuent leur calcule avec des données non remises à jour.

La figure 7.21 montre la dégradation de plus en plus importante de la réponse du système en boucle fermée lorsque les paquets contenant la mesure de la vitesse angulaire sont perdus. Par exemple, un dépassement de 3% apparaît dans le cas où il y a 50% de pertes. C'est pourquoi la perte de données doit être considérée comme un défaut et doit être détectée. De plus, il est intéressant de voir l'influence de cette perte sur les résidus qui se comportent comme s'il y avait un défaut sur les capteurs.

Les figures 7.22 et 7.23 montrent l'influence des pertes produites par le réseau sur les résidus capteurs. Ce résultat est important car le résidu généré pour détecter des défauts capteurs est influencé par un défaut réseau, alors il n'est plus possible de localiser les

défauts capteurs. Pour cette raison, il est nécessaire de différencier un défaut réseau d'un défaut capteur :

$$defaut \;\; capteur = \overline{r_{network}} \wedge (r_1 \vee r_2) \tag{7.2.5}$$

$$defaut \;\; reseau = r_{network} \tag{7.2.6}$$

Les équations (7.2.5) et (7.2.6) permettent de localiser les défauts capteurs si le nouvel indicateur appelé $r_{network}$, est sensible uniquement à un défaut réseau. Cet indicateur est booléen et égal à 1 uniquement en présence d'un défaut réseau. Ceci est considéré comme vrai quand ω_{mes} n'est pas parvenu au module de diagnostic à la fin des 20 premières millisecondes de la période d'échantillonnage soit à la fin de 75 % de la période d'échantillonnage. A chaque nouvelle période d'échantillonnage, cet indicateur est remis à 0.

TAB. 7.2 – Table de signature pour détecter les défauts capteur et réseau

	f_ω	f_i	f_{reseau}
r_1(7.2.3)	+	-	ϕ
r_2(7.2.4)	+	+	ϕ
$r_{network}$(7.2.4)	0	0	1

La table 7.2 montre la nouvelle table de signature pour localiser les défauts capteur et réseau. Maintenant, r_1 et r_2 sont tous les deux sensibles à tous les défauts et $r_{network}$ est sensible uniquement au "défaut réseau" qui correspond à la non délivrance des données dans une fenêtre de temps pré-fixée.

7.2.3.4 Proposition de reconfiguration

La figure 7.21 montre l'influence des pertes de paquets sur la réponse du système en boucle fermée[2]. Maintenant que le défaut réseau peut être détecté, l'idée est de réaliser une reconfiguration de la loi de commande [91]. Pour cela, la valeur de ω_{mod} calculée par le module de diagnostic[3] est envoyée au contrôleur pour remplacer la mesure manquante. La figure 7.25 montre le résultat avec un défaut réseau, avec et sans reconfiguration (50% de perte sur le capteur de vitesse angulaire). Avec cette reconfiguration, on tend vers la réponse d'origine. Notons tout de même que cette reconfiguration est très simple car la reconstruction est réalisée par le modèle. D'une manière générale, la reconstruction pourrait se faire par observateur et plus précisément par différent observateurs selon les mesures perdues comme dans [14].

[2]Dans cette étude, nous nous somme focalisé uniquement sur la perte de la mesure ω_{mes}

[3]Cette mesure peut être calculée car elle a besoin de i_{mes} et u_{mes}.

7.2.4 Conclusion

Avec cette étude, nous avons montré l'influence du réseau sur la commande et sur le diagnostic et l'importance de bien gérer les tâches en temps réel en fonction de l'arrivée (ou non) des données. Dans la suite du manuscrit, ces résultats notamment l'implantation de l'indicateur $r_{network}$ seront utilisés pour la mise en réseau du quadrotor.

7.3 "Control over network" : application au quadrotor

On se focalise dans cette partie sur le phénomène qui a retenu notre attention, à savoir la perte de paquets qui peut être induite par la surcharge du réseau. Les résultats obtenus en simulation avec la boite à outils TrueTime sont donnés. Le système utilisé comme exemple est le quadrotor présenté précédemment.

Parmi les questions à résoudre dans le cadre de ce travail de thèse (et du projet *SafeNecs*), on trouve le problème des stratégies de distribution des tâches de diagnostic entre un système embarqué et un système central et la prise en compte des possibilités de défaillance du réseau de communication entre ces deux entités. On voit donc ici s'étendre la notion de diagnostic qui doit considérer le réseau comme un composant, susceptible d'être défaillant au même titre que les autres composants du système (capteurs, actionneurs, système).

Les pertes de données et leur désynchronisation vont influencer les algorithmes de diagnostic qui, jusqu'à présent, supposent des données régulièrement échantillonnées et synchrones. On pourrait envisager des solutions à caractère plus empirique : re-synchronisation artificielle des données par exemple, mais alors la confiance dans une donnée re-synchronisée doit être diminuée et les seuils utilisés pour engendrer un symptôme doivent être augmentés.

Dans notre application, le réseau est implanté dans la boucle de commande comme le montre la figure 7.26. De cette façon, le réseau a une influence maximale car il intervient entre :

- la tâche capteur et la tâche contrôleur : la tâche capteur envoie les flux de données à la tâche contrôleur périodiquement ;
- la tâche contrôleur et la tâche actionneur : la tâche contrôleur calcule la loi de commande après avoir reçu les données capteurs et envoie à la tâche actionneur les quatre consignes de vitesse des quatre moteurs.

La tâche capteur est guidée par le temps, c'est-à-dire que l'envoi est guidé par une horloge alors que les tâches contrôleur et actionneur sont guidées par les événements. Le contrôleur attend la réception des échantillons avant de calculer et d'envoyer un message à la tâche actionneur.

L'application comporte dix-sept données (les quatre références de vitesse de rotation des moteurs, les neuf mesures de la centrale d'attitude et enfin les mesures de la vitesse angulaire des quatre moteurs). Le choix s'est porté sur l'utilisation d'un flux (ou paquet) pour chaque donnée. Il y aura donc dix-sept flux qui transiteront à travers le réseau.

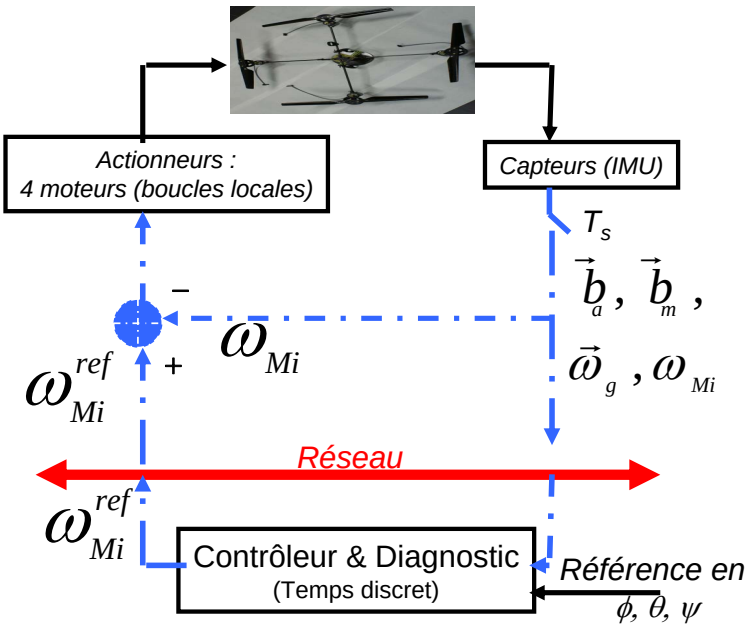


FIG. 7.26 – Quadrotor commandé en réseau

7.3.1 Perte de paquets

Lorsqu'un réseau est utilisé pour commander un système, certains phénomènes propres au réseau lui-même doivent être pris en compte dans l'étude du système commandé via le réseau. Dans [16], on a vu que les retards dus à la communication sont négligeables par rapport à la période d'échantillonnage du système. Uniquement 10 % de cette dernière est utilisée pour la transmission des données, comme cela va être montré ci-après. Cependant, si le retard n'était pas négligeable, nous pourrions l'estimer avec des approches basées sur les observateurs comme dans [62]. Maintenant, nous allons nous intéresser à un autre phénomène induit par le réseau, la perte de paquets. En effet, une trame (ou paquet) peut être émise mais il n'est pas toujours garanti qu'elle soit reçue par le point de destination ou bien encore elle peut comporter des erreurs . Ces phénomènes peuvent être dus à un réseau surchargé ou à des conditions environnementales dégradées comme par exemple la présence de perturbations électromagnétiques.

En fait, l'objectif de cette étude est de pouvoir disposer d'informations sur cette perte de paquets, et d'être capable de différencier une perte de paquet d'un défaut. Dans le paragraphe (7.2), nous avons montré que si le module de diagnostic n'est pas implanté

de manière ad-hoc, une perte de paquet peut induire une fausse alarme. C'est pourquoi, nous avons implanté l'indicateur $r_{network}$ (voir le paragraphe (7.2)). Cet indicateur n'est sensible qu'aux pertes de paquets.

L'idée ici est la suivante (figure 7.27). A chaque instant d'échantillonnage, les données capteurs sont émises par la centrale d'attitude. Nous avons décidé de fixer la durée de vie des paquets à 5 ms, soit l'équivalent d'une demi-période d'échantillonnage. Après ces 5 ms, toutes les données non transmises sont supprimées par le réseau et l'indicateur $r_{network}$ se verra affecter la valeur 1. Les 5 ms restantes permettent en fait d'exécuter la tâche "diagnostic", de reconfigurer si nécessaire la commande, de calculer les références à renvoyer aux boucles locales des actionneurs et d'envoyer ces références.

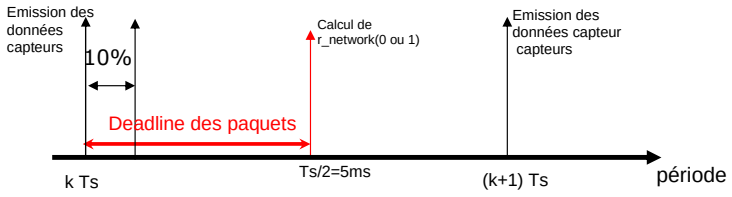


FIG. 7.27 – Chronogramme de fonctionnement de l'indicateur $r_{network}$

7.3.2 Quadrotor avec réseau CAN

Dans ce paragraphe nous allons étudier le cas où le réseau est strictement dédié à l'application c'est-à-dire le cas où il n'est utilisé que par l'application. Il est primordial lorsqu'on souhaite utiliser un réseau CAN, de spécifier :

- le taux de requêtes utilisateur (URL). Ce taux correspond au taux de demandes de transfert de trames faites au niveau de la couche MAC par les applications réalisées à travers le réseau. Ce taux dépend évidemment des caractéristiques de toutes les applications qui vont utiliser la ressource. Ce taux est défini pour n flux par :

$$URL = \sum_{i=1}^{n} \frac{D_i}{T_{ei}} \tag{7.3.1}$$

  avec D_i la durée de la trame et T_{ei} la période d'émission. Ce taux peut être inférieur à 1, cela correspond au cas où le réseau est utilisé correctement ou bien supérieur à 1, ce dernier cas signifie que les demandes utilisateur sont supérieures aux possibilités de transfert des trames par le réseau ;
- la longueur des trames, données par [71] ;
- le débit sur le bus.

Nous considérons dans cette étude, les données suivantes [16] :

- un débit de 1Mbits/secondes ;

– les trames sont composées de 64 bits. Compte tenu que l'on a 17 flux utilisant la ressource, on obtient un URL de 10%.

L'arbitrage de CAN fonctionnant sur le principe de priorité, la distribution des priorités est définie comme suit :

– tout d'abord, priorité est donnée à la commande devant les mesures. Par conséquent, les quatre flux les plus prioritaires sont les quatre consignes de vitesses de rotation des moteurs. Le flux avec la consigne du moteur 1 est plus prioritaire, ensuite vient la consigne du moteur 2 puis celle du moteur 3 et enfin celle du moteur 4. Notons que cette numérotation est arbitraire ;
– ensuite les neuf flux de la tâche capteur. L'ordre de priorité est le suivant : tout d'abord les trois gyromètres dans l'ordre des axes x, y, z, ensuite les trois accéléromètres (dans l'ordre des axes x, y, z) et enfin les trois magnétomètres (dans l'ordre des axes x, y, z) ;
– pour finir, les quatre vitesses moteur sont envoyées avec les priorités les plus faibles car ces quatre mesures ne sont utilisées que pour le diagnostic et pas pour la commande. Le flux avec la mesure de rotation du moteur 1 est le plus prioritaire, ensuite vient la mesure de rotation du moteur 2 puis celle du moteur 3 et enfin celle du moteur 4.

7.3.2.1 Résultats de simulation

Dans ce paragraphe, nous allons présenter les résultats obtenus en simulation. Comme indiqué précédemment, la tâche capteur envoie les neuf mesures à la tâche commande et au module de diagnostic, elle est guidée par le temps. Elle va donc envoyer ses informations toutes les T_e secondes. Dans notre application, la période d'émission choisie est $T_e = 10\ ms$. Dans un souci d'interprétation, les résultats obtenus pour l'attitude du quadrotor sont donnés sous la représentation des angles de Cardan (roulis (ϕ) , tangage (θ) , lacet (ψ)) et non sous la représentation du quaternion unitaire car cette dernière est moins intuitive. Cependant, tous les calculs ont été réalisés en utilisant comme représentation de l'attitude le quaternion unitaire. Le scénario est d'amener le quadrotor de la position initiale $[\phi = -35^\circ; \theta = -25^\circ; \psi = -10^\circ]$ à la position d'équilibre $[\phi = 0^\circ; \theta = 0^\circ; \psi = 0^\circ]$.

La tâche contrôleur est guidée par les événements : elle ne calcule la nouvelle loi de commande que lorsqu'elle a reçu les neuf mesures provenant des capteurs. De plus, puisque dans cette configuration, il faut uniquement 10% de la période d'émission pour transmettre toutes les données, on s'autorise un retard maximal d'une demi-période d'échantillonnage dans la réception des mesures. En conséquence, si au bout d'une demi-période on n'a pas reçu les mesures, les informations manquantes sont considérées comme perdues par le réseau et les algorithmes de commande et de diagnostic sont bloqués. Il s'en suit que la tâche actionneur conserve la commande précédemment reçue.

7.3.2.2 Réseau dédié

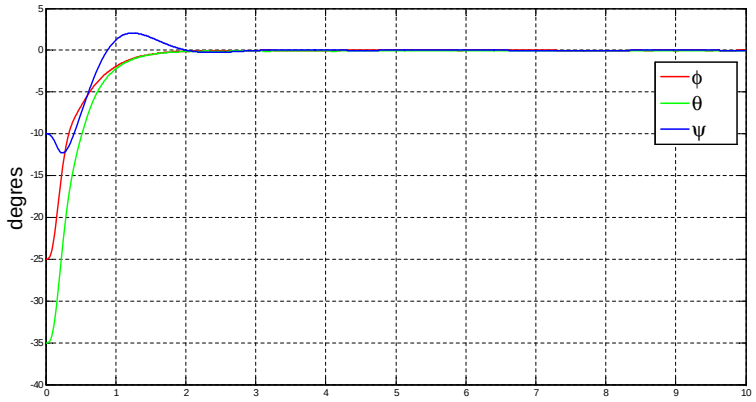


FIG. 7.28 – Attitude réelle du quadrotor avec réseau CAN dédié

Dans [16] (voir figure 7.28), il est montré que les réponses obtenues dans le cas d'un réseau dédié sans perte de données possèdent les mêmes performances que celles sans réseau. Il faut noter ici que l'on obtient des temps de réponse à 5% égaux à 1.22 s pour ϕ, 1.3 s pour θ et enfin 1.85 s pour ψ.

7.3.2.3 Réseau partagé

Pour simuler les situations où le réseau est partagé entre plusieurs applications, nous avons créé une tâche supplémentaire appelée tâche externe. Celle-ci accède au réseau avec une période T_{ef} (tâche guidée par le temps). En modifiant T_{ef}, on modifie la demande de cette tâche et donc la charge du réseau. Dans [38], il est montré que si les trames provenant de la ressource externe à l'application sont moins prioritaires que celles de l'application alors la ressource externe n'a pas d'influence sur le système en boucle fermée. Dans le scénario de simulation que l'on va maintenant commenter, le bus CAN est partagé entre le quadrotor et une ressource externe, dont les trames sont prioritaires devant les trames du quadrotor. Cette ressource envoie toutes les $T_{ef} = 10\ ms$ une trame à travers le réseau d'une longueur maximale soit 111 bits. L'objectif est de stabiliser le quadrotor à l'attitude $\phi = 0^{\circ}$, $\theta = 0^{\circ}$ et $\psi = 0^{\circ}$ avec comme attitude initiale ϕ = -35°, θ = -25° et ψ = -10°. La figure 7.29 montre le résultat obtenu pour ce scénario. Le quadrotor se stabilise dans l'attitude souhaitée. Cependant, on peut remarquer que la réponse est dégradée durant le transitoire. Ceci est dû au fait que de temps en temps les trames provenant du capteur n'arrivent pas dans les 5 ms autorisées et qu'alors la commande converse la valeur précédente. Cela introduit donc un retard aléatoire dans la boucle de commande.

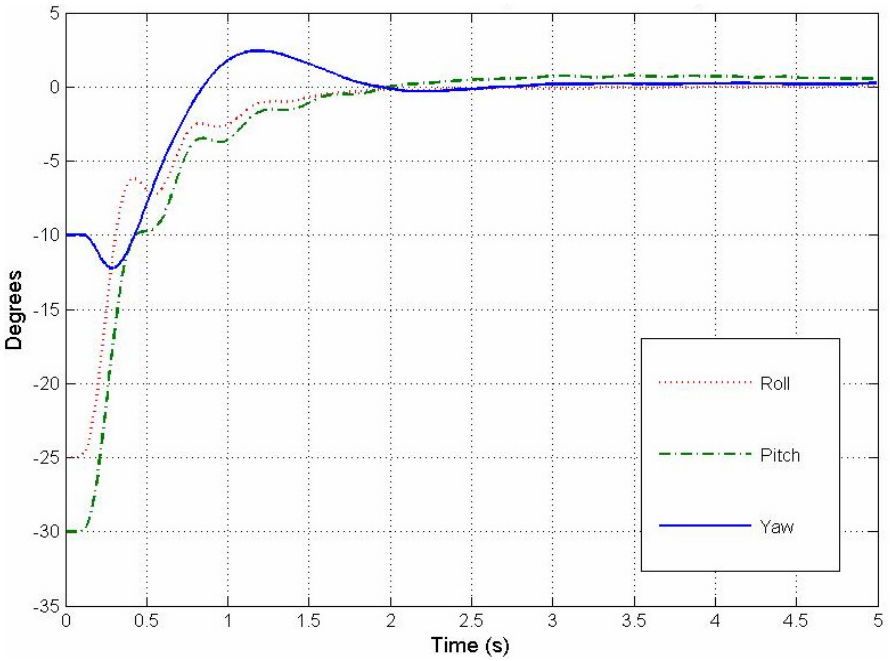


FIG. 7.29 – Attitude du quadrotor avec un réseau CAN partagé

7.3.3 Quadrotor avec Ethernet

7.3.3.1 Réseau dédié

Les conditions de simulation sont identiques à celles du quadrotor commandé au travers du réseau CAN, l'objectif étant de stabiliser le quadrotor à l'attitude $\phi = 0^{\circ}$, $\theta = 0^{\circ}$ et $\psi = 0^{\circ}$ avec comme attitude initiale ϕ = -35°, θ = -25° et ψ = -10°. Cette fois-ci, le réseau utilisé est Ethernet avec les caractéristiques suivantes :

- un débit de 10 $Mbits/s$;
- les trames sont composées de 72 octets (26 octets d'entêtes et 46 octets pour les données).

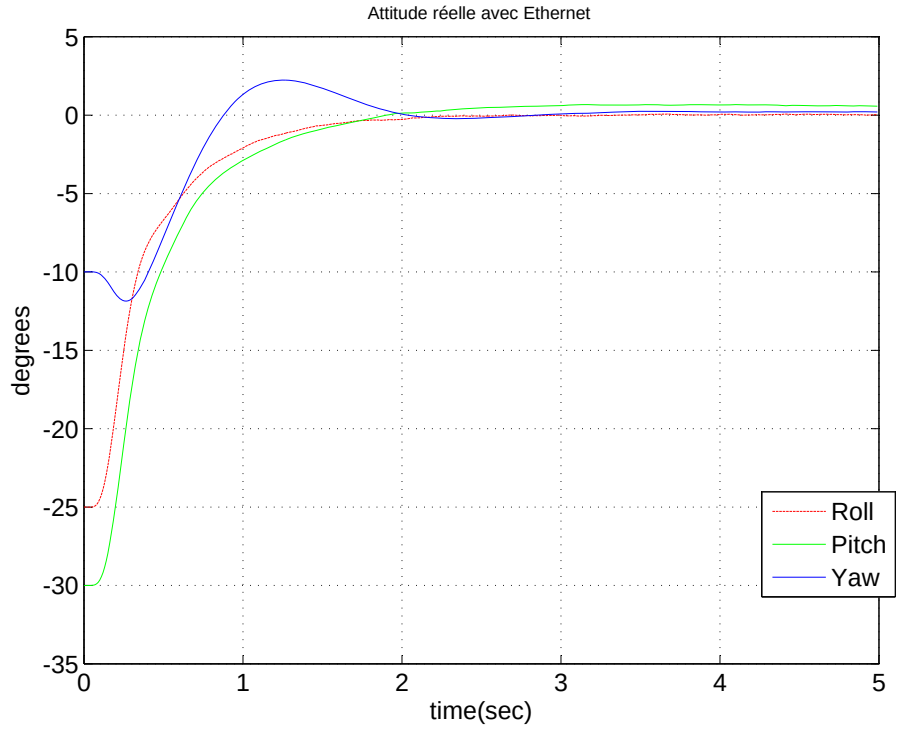


FIG. 7.30 – Attitude réelle du quadrotor avec Ethernet dédié

La figure 7.30 montre le résultat obtenu. Le quadrotor est stabilisé dans l'attitude souhaitée avec un temps de réponse identique à celui obtenu avec le réseau CAN (voir figure 7.28).

7.3.3.2 Réseau partagé

Cette fois-ci, le réseau est partagé avec une ressource externe. La ressource envoie une trame de 12208 bits (longueur maximale) toutes les 10 ms, l'objectif étant de charger le réseau. En conséquence la "Qualité de Service" (QoS) n'est plus garantie et le fonctionnement du système est perturbé. Les conditions de simulation restent inchangées. La figure 7.31 montre l'attitude obtenue dans ce cas. Le quadrotor n'est plus stabilisé. Ceci est dû aux problèmes de collisions introduites par le protocole CSMA/CD.

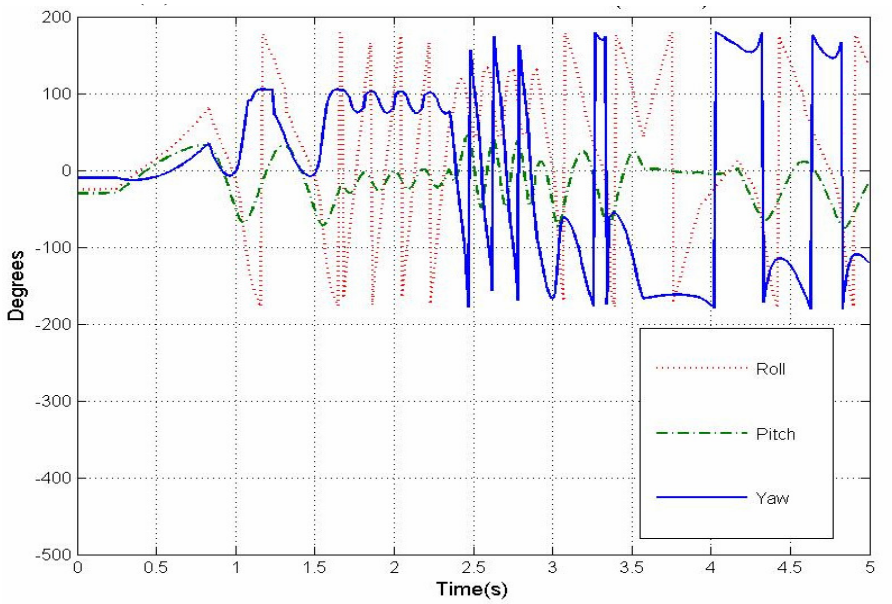


FIG. 7.31 – Attitude réelle du quadrotor avec Ethernet partagé

En conclusion, on s'aperçoit que même si le réseau Ethernet est très rapide, il peut ne pas être adapté à un système commandé en réseau. En effet, plus le nombres de flux est important et plus le problème de collisions est difficile à gérer.

7.3.4 Quadrotor avec Ethernet Commuté

Cette partie du travail a été réalisée en collaboration avec l'équipe du CRAN dans le cadre du projet *Safe-NECS* [25].

7.3.4.1 Réseau dédié

On remplace le réseau par le réseau Ethernet commuté sans politique d'ordonnancement.

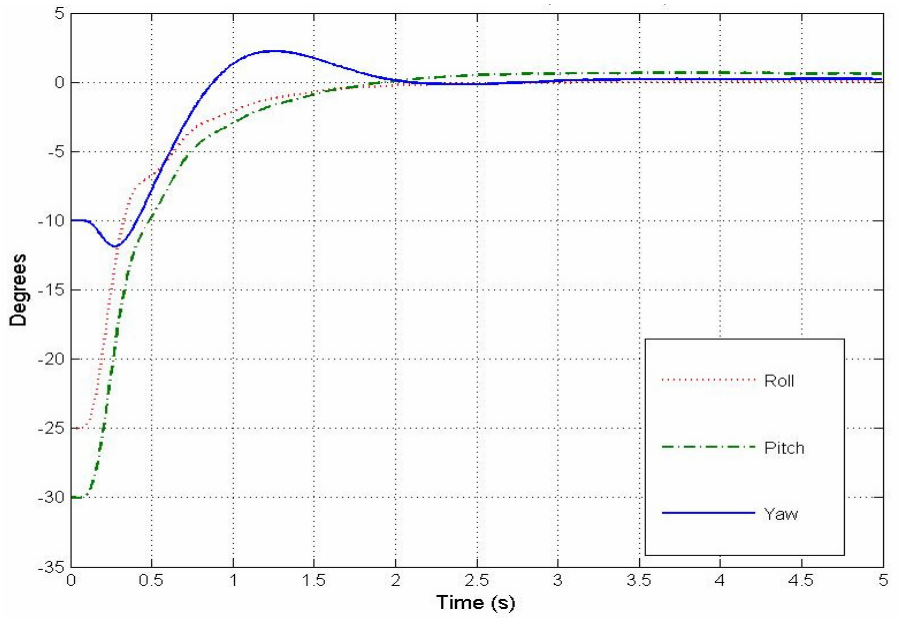


FIG. 7.32 – Attitude réelle du quadrotor avec Ethernet commuté dédié

Le réseau est caractérisé par un débit de 10 $Mbits/s$. Les conditions de simulation restent identiques à celles des cas précédents. La figure 7.32 montre le résultat obtenu. Le quadrotor est stabilisé dans la position désirée.

7.3.4.2 Réseau partagé

Le réseau Ethernet commuté est maintenant partagé. La longueur constituant les paquets du flux externe est maximale et égale à 12208 bits. La figure 7.33 montre le résultat de la simulation.

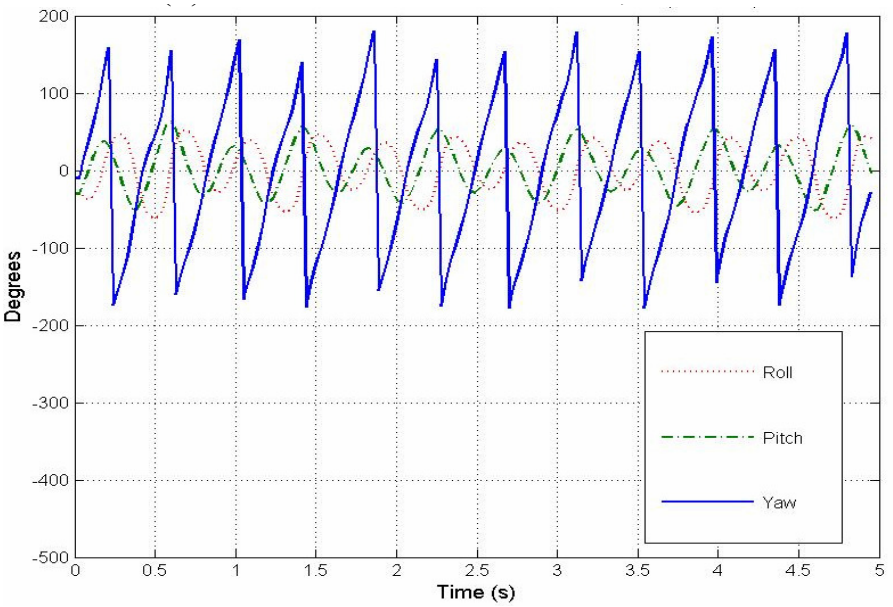


FIG. 7.33 – Attitude réelle du quadrotor avec Ethernet commuté partagé

Tout comme dans le cas précédent, le quadrotor ne se stabilise pas dans l'attitude désirée. Pour palier ce problème, une politique d'ordonnancement de type *WRR* (Weighted Round Robin) est utilisée, politique implantée dans TrueTime par l'équipe du CRAN. Cette stratégie associe un poids à chaque classe de trafic qui correspond à une queue sur la sortie du commutateur. Chaque classe est servie avec un taux proportionnel à son poids. Le port de sortie du commutateur utilisé pour la tâche contrôleur intègre la politique du *WRR* pour permettre de différencier le service entre les mesures produites par la tâche capteur et celle produite par le flux externe. Les poids sont configurés à partir des résultats obtenus dans [26]. Pour garantir que les mesures arrivent dans un délai inférieur au retard maximal de $5ms$ (avant de considérer les paquets perdus), il est nécessaire d'utiliser un poids $w_1 = 9$ pour les mesures et un poids $w_2 = 3$ pour le flux externe. Le résultat de simulation est montré sur la figure 7.34. Avec cet ordonnancement, nous pouvons garantir le service du point de vue de la stabilisation du quadrotor et nous avons pu garantir un service pour le trafic de fond (la tâche externe). De plus, en comparaison avec CAN, l'intérêt principal d'Ethernet commuté avec une politique d'ordonnancement de type *WRR* est la maximisation de la bande passante offerte pour la communication de fond et la minimisation de l'impact de la communication pour l'application. De plus, avec le réseau CAN, nous avions un débit maximal de 1 $Mbits/s$ alors qu'avec ce type de réseau, le débit est de 10 $Mbits/s$.

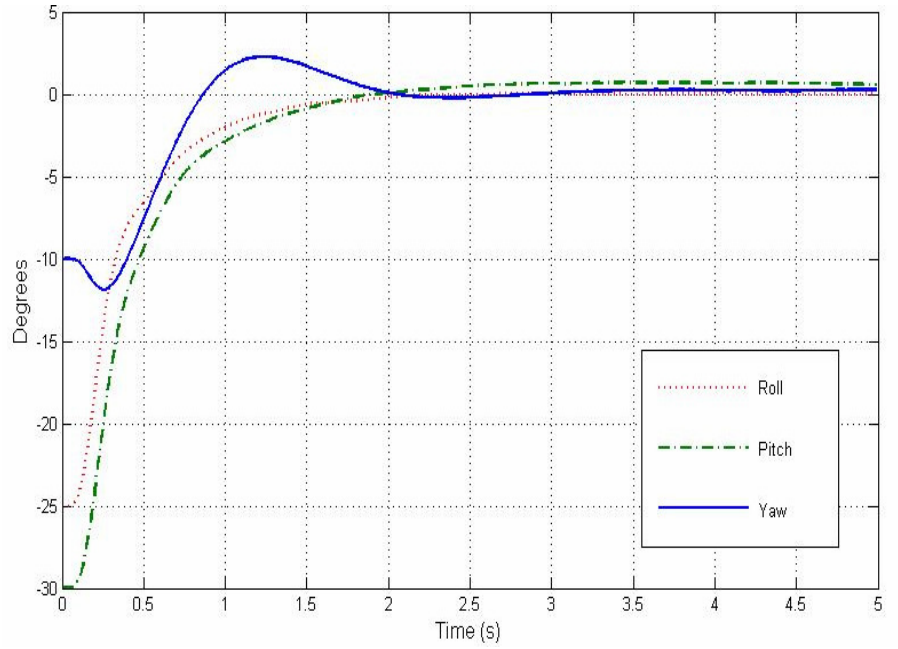


FIG. 7.34 – Attitude réelle du quadrotor avec Ethernet commuté partagé et une politique d'ordonnancement de type WRR

7.3.5 Conclusion sur l'aspect commande à travers un réseau

Dans la première partie de ce chapitre, les différents réseaux introduits dans le chapitre précédent ont été utilisés comme communication entre le quadrotor et la tâche de commande. Les inconvénients, surtout le manque de mécanisme de priorité, du réseau Ethernet influencent vraiment les résultats souhaités pour la stabilisation du quadrotor. En effet, dès que le nombre de paquets devient important, le nombre de collisions devient lui aussi très important et engendre trop de pertes pour pouvoir satisfaire les besoins du système.

Pour contrer cette difficulté, la méthode d'Ethernet commuté a été utilisée. Dès lors que le réseau est partagé, il est nécessaire d'appliquer des politiques de type Weighted Round Robin afin de garantir la disponibilité de la ressource de communication aux besoins du système.

Malgré un débit moindre que pour Ethernet ou bien Ethernet commuté, le réseau CAN (Controller Area Network) possède de bien meilleurs résultats. Grâce à ses mécanismes de priorité, la ressource de communication est en mesure de répondre au besoin de l'application de commande avec un réseau CAN de base c'est à dire sans aucune politique d'ordonnancement autre que celle des priorités statiques.

Remarque 18. Dans la suite de ce manuscrit, le réseau CAN sera choisi et donc utilisé comme ressource de communication. Dans un premier temps, la notion de diagnostic distribué à travers un réseau va être considérée. Dans un second temps, et pour clore ce chapitre, l'aspect "Control of network" (contrôle du réseau) sera étudié.

7.4 Diagnostic du quadrotor en présence du réseau avec CAN

Nous avons vu, dans le cas de l'exemple du moteur, l'influence du réseau sur le diagnostic et plus particulièrement le fait qu'une perte de paquet peut être interprétée comme un défaut capteur. Dans ce qui suit, le module de diagnostic nommé *Algo 1* présenté dans 4.2 est utilisé. Cet algorithme est exécuté après avoir reçu toutes les mesures. Si une mesure se trouve manquante, alors le module de diagnostic ne sera pas exécuté.

7.4.1 Fonctionnement sans défaut

Dans un premier temps, nous allons nous intéresser au cas où il n'y a aucun défaut sur les capteurs. Par conséquent, les résidus calculés par les trois observateurs et les six estimateurs doivent être nuls. La figure 7.35 montre les résultats obtenus dans les deux cas. Cette figure représente les résidus obtenus sans les accéléromètres pour la première ligne, sans les magnétomètres pour la deuxième ligne et sans les gyromètres pour la troisième ligne. Chaque colonne représente l'axe négligé. Par exemple la première courbe représente le résidu obtenu sans l'accéléromètre d'axe X. On peut constater que tous les résidus sont nuls excepté pour les résidus d'axe Y. L'erreur sur Y peut s'expliquer par le fait que lorsque l'on ne prend pas en compte une mesure de cet axe (par exemple lorsque l'on ne prend pas en compte acc_Y ou bien mag_Y) l'unicité de la solution de l'optimisation non linéaire est perdue. Par conséquence, plusieurs attitudes sont solutions d'où ces écarts sur les résidus. Cela veut dire que le seuil de détection sur r_2 et r_5 doit être plus élevé que sur les autres résidus. On peut remarquer que les résultats obtenus avec et sans réseau sont similaires.

7.4.2 Défaut sur mag_X

A $t = 4\ s$, un défaut capteur sur mag_X est considéré. Dans cette simulation, le défaut est du type perte du capteur, c'est-à-dire que celui-ci ne délivre plus que 0 comme valeur. On peut remarquer que dans les cas sans et avec réseau (figure7.36), seul le résidu généré sans le magnétomètre en défaut est insensible à ce défaut, alors que tous les autres résidus le sont. Pour la localisation, il suffit de comparer avec la table de signature (voir table

7.3). La figure 7.36 ne montre que les 6 premiers résidus car les résidus générés pour la surveillance des gyromètres sont toujours supérieurs à zéro.

TAB. 7.3 – Rappel de la table de signature complète pour la localisation des défauts sur la centrale d'attitude avec *Algo 1*

	f_{ax}	f_{ay}	f_{az}	f_{mx}	f_{my}	f_{mz}	f_{gx}	f_{gy}	f_{gz}
r_1	0	1	1	1	1	1	0	0	0
r_2	1	0	1	1	1	1	0	0	0
r_3	1	1	0	1	1	1	0	0	0
r_4	1	1	1	0	1	1	0	0	0
r_5	1	1	1	1	0	1	0	0	0
r_6	1	1	1	1	1	0	0	0	0
r_7	1	1	1	1	1	1	0	1	1
r_8	1	1	1	1	1	1	1	0	1
r_9	1	1	1	1	1	1	1	1	0

On peut conclure donc que le défaut est détecté et localisé avec et sans réseau.

7.4.3 Défaut sur acc_Z

Dans cette simulation, nous avons simulé un défaut de type "perte d'un capteur" (la valeur prise par la mesure est remplacée par 0) sur acc_Z. Ce défaut a été simulé à l'instant $t = 4\ s$. Sur la figure 7.37, on voit que tous les résidus sont sensibles à ce défaut excepté le résidu r_3 car c'est le seul qui est calculé en écartant la mesure de cet accéléromètre. Une fois le défaut détecté, nous comparons le mot binaire obtenu avec la table de signature afin de localiser le défaut. On constate que le mot binaire est celui correspondant à un défaut survenu sur acc_Z. La procédure de détection et de localisation a pu être menée à bien.

Avec ces trois scénarios, on peut constater que le réseau n'a pas d'influence sur le module de diagnostic car les algorithmes sont bien codés et le réseau fonctionne sans perte. Maintenant, étudions le cas où le réseau est défaillant.

7.4.4 Défaut réseau

Nous allons étudier le cas où le réseau induit des pertes de données. Pour cela, 10 % de pertes de paquets sont considérées (pertes sur n'importe quelles mesures provenant des capteurs). Dans [15] et dans la section 7.2, il a été montré que la perte de paquet influence le diagnostic, plus particulièrement. Des fausses alarmes sont introduites, si le module de diagnostic est guidé par les événements, c'est-à-dire si à chaque mesure

reçue, l'algorithme de diagnostic est exécuté. Pour éviter ce problème, le module de diagnostic n'est exécuté que lorsqu'il a reçu toutes les mesures et que l'indicateur de perte de paquets $r_{network}$ est égal à 0.

Si la mesure n'est pas reçue dans le temps imparti alors l'indicateur $r_{network}$ passe à '1' (voir figure 7.38).

7.4.4.1 Réponse en boucle fermée

La figure 7.39 montre le résultat de simulation dans les cas sans perte de données et avec 10 % de perte de paquets. Lorsqu'une perte est détectée, l'indicateur $r_{network}$ prend la valeur 1, la commande n'est pas calculée et les anciennes consignes de vitesse des moteurs sont maintenues. C'est pourquoi on retrouve des différences de comportement, notamment dans le transitoire. Cependant, on peut remarquer que le système est robuste à 10 % de perte de paquets puisqu'il réussi à se stabiliser en 3.3 s.

7.4.4.2 Diagnostic

La figure 7.40 présente le résidu r_1. Celui-ci est calculé uniquement lorsque l'indicateur $r_{network}$ est égal à 0. On peut remarquer avec la figure qu'à chaque fois que l'indicateur est égal à 1, alors le résidu n'est pas calculé.

7.4.5 Conclusion

En conclusion, on peut remarquer que les comportements avec et sans réseau sont identiques si l'on prend en considération le réseau comme composant du système et non pas comme un simple moyen de communication. Notons que l'algorithme de diagnostic n'est exécuté qu'après avoir reçu toutes les mesures afin d'éviter les fausses alarmes.

7.5 "Control of network"

Dans cette section, nous allons nous intéresser au contrôle du système de communication, c'est-à-dire voir comment on peut commander le réseau afin de garantir des bonnes performances pour le système contrôlé à travers celui-ci. Pour simplifier notre étude, nous utiliserons un modèle linéarisé autour du point de fonctionnement $\phi = 0^\circ$; $\theta = 0^\circ$; $\psi = 0^\circ$. Pour charger le réseau, on introduit une tâche supplémentaire dite "tâche externe" de période variable. Le modèle linéarisé est décrit dans l'annexe A.

7.5.1 Description de la simulation

Dans ce qui suit, nous considérons les caractéristiques suivantes :
- le réseau utilisé est CAN (Controller Area Network) ;
- il a un débit de 500 $Kbits/s$;
- les trames de l'application, c'est à dire, la trame capteur et contrôleur sont constituées de $L = 128$ bits ;
- la trame de la tâche externe est composée de $L_c = 128$ bits. L'utilisation de la ressource dépend de la période d'émission de la tâche externe, notée T_{ef} (paramètre que l'on fait varier) ;
- on suppose que toutes les mesures sont envoyées dans un unique paquet de même pour les consignes des boucles locales[4] ;
- D_{sf}, D_{cf}, D_{ef} représentent les durées de transmission des trames capteur, contrôleur et tâche externe respectivement ;
- h représente la période d'échantillonnage du système ;
- le URF (Use Request Factor) du réseau est calculé par $URF = (D_{sf}/T_e + D_{cf}/T_e + D_{ef}/T_{ef})$. Dans le cas d'un réseau dédié, l'application n'utilise que 5 % de la ressource (avec $T_e = 10\ ms$). Pour le cas du système avec la tâche externe, la ressource est utilisée à 10 % ;
- la priorité est dans un premier temps fixe et est donnée à la tâche externe devant l'application. Cette configuration représente le pire cas d'un point de vue commande de l'application [38].
- comme dans le paragraphe précédent, la tâche capteur est guidée par le temps alors que les tâches contrôleur et actionneur sont guidées par les événements.

7.5.2 Quadrotor avec réseau CAN dédié

Les performances de commande sont étudiées quand le quadrotor est implémenté à travers le réseau CAN dédié. Cette étude est l'étude de référence pour tout les exemples qui seront considérés dans cette section.

L'objectif est de stabiliser le quadrotor à la position d'équilibre $[\phi = 0; \theta = 0; \psi = 0]$ avec comme attitude initiale $\phi = -25^\circ; \theta = 30^\circ; \psi = -10^\circ$ [5].

[4] Ceci se justifie par le fait que l'on désire mettre en avant plusieurs techniques d'ordonnancement et par conséquent, le fait d'envoyer tout dans un paquet est plus facile pour l'interprétation.

[5] Contrairement au chapitre précédant, les conditions initiales du quadrotor ont changé afin d'étudier d'autres cas de simulation

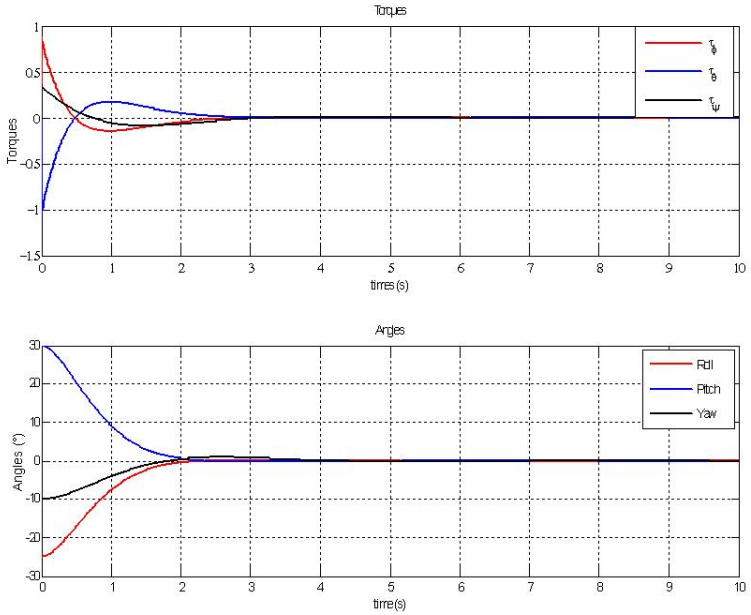


FIG. 7.41 – Réponse du système en boucle fermée avec un réseau dédié ((haut) : les couples, (bas) : l'attitude)

La figure 7.41 montre le résultat de simulation. Le quadrotor atteint son attitude finale en 2 secondes, ce qui est similaire au scénario sans réseau.

7.5.3 Quadrotor avec réseau CAN partagé : priorités fixes

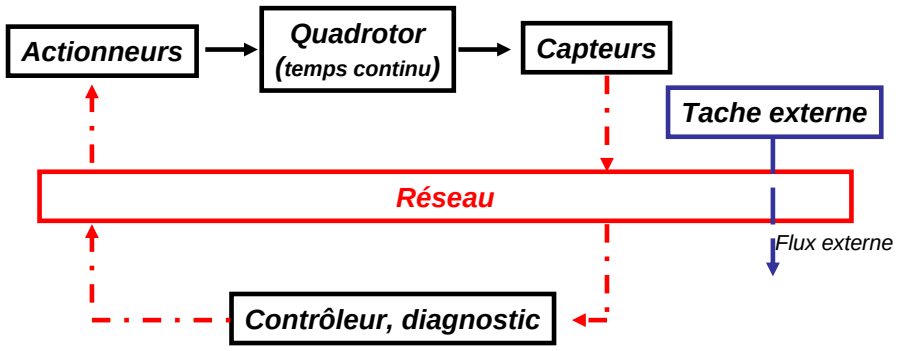


FIG. 7.42 – Architecture du système commandé en réseau avec un réseau partagé

Ici, un flux externe représentant la charge à travers le réseau est utilisé comme le montre la figure 7.42. Dans [38], quelques résultats ont été montrés tels que :

– si la priorité associée au flux externe est plus petite que celles associées aux flux capteur ou contrôleur, alors les performances du système commandé ne sont pas influencées par le flux externe ;

TAB. 7.4 – Fonction de coût pour l'évaluation des performances de commande

URF	T_{ef}	J	$J - J_{min}/J_{min}$
30%	$1ms$	0.033	0%
50%	$568\mu s$	0.0333	0%
75%	$365\mu s$	0.0333	0%
95%	$284\mu s$	0.0333	0%
99%	$272\mu s$	$12 * 10^4$	$3 * 10^8$%
100%	$269\mu s$	$45 * 10^4$	$1.36 * 10^9$%
101%	$266\mu s$	$5.48 * 10^6$	$1.6 * 10^{10}$%
102%	$263\mu s$	$1.57 * 10^7$	$4.75 * 10^{10}$%

– si la priorité associée au flux contrôleur est plus petite que celles associées aux flux capteur ou au flux externe, alors le flux externe influence les performances de commande du système. Plus particulièrement, quand le réseau est chargé, le système ne peut plus être stabilisé (figure 7.43).

Les performances du système sont étudiées en fonction de la charge du réseau. Pour cela, une fonction de coût J est utilisée (voir annexe A, équation A.2.3). Elle représente un critère quadratique pondérant l'écart sur l'état et la commande sur un horizon donnée. J_{min} correspond à la valeur optimale du critère.

Il est clair que plus la fonction de coût est dégradée et plus la performance du système est mauvaise. Le tableau 7.4 montre les résultats. La charge du réseau est augmentée avec la diminution de la période d'émission de la trame de la tâche externe. Plus la tâche externe va émettre et plus la charge sera importante. On remarque que l'on obtient une dégradation qui augmente en fonction de l'augmentation de la charge du réseau. Lorsque l'URF est supérieur à 99%, le système n'est plus stabilisé.

La période de la tâche externe est choisie égale à $T_{ef} = 266\ \mu s$. Dans ces conditions, le réseau est surchargé. La figure 7.43 montre le résultat obtenu dans ce cas. Le quadrotor n'est plus stabilisé. Le problème est que le réseau étant surchargé, la bande passante est toujours occupée par la tâche externe.

Dans la suite du manuscrit, plusieurs politiques d'ordonnancement vont être implantées afin de garantir une meilleure qualité de contrôle.

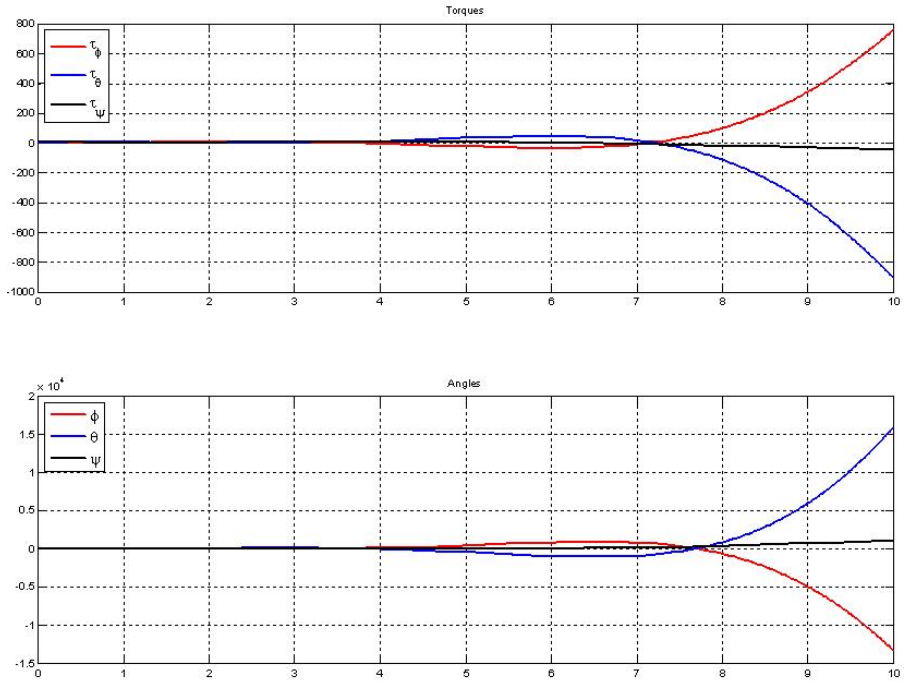


FIG. 7.43 – Attitude réelle du quadrotor avec un réseau CAN partagé ((haut) : les couples, (bas) : l'attitude)

7.5.4 Quadrotor avec réseau CAN partagé : priorités dynamiques

Maintenant, nous allons présenter la politique d'ordonnancement dite "priorité dynamique" développée par des partenaires du projet *Safe-NECS* (équipe du LAAS).

Comme on vient de le voir, lorsque les priorités statiques sont utilisées, que le réseau est surchargé et que les trames de l'application ne sont pas prioritaires, le système ne peut pas être stabilisé. En pratique, il n'est pas toujours possible de donner la priorité maximale à l'application devant les autres tâches. C'est pourquoi, dans [38], [59], une nouvelle politique d'ordonnancement est proposée en faisant apparaître la notion d'urgence. On considère ici que le quadrotor génère deux flux (celui de la tâche capteur et celui de la tâche contrôleur) qui ont un besoin d'urgence. En particulier, il y a "urgence" d'un point de vue contrôle durant les phases transitoires, après un changement de consigne ou bien encore après une perturbation.

7.5.4.1 Fonctionnement

Le principe est repris de [59]. Les priorités dynamiques sont inspirées du modèle "Mixed Traffic Scheduling" [111]. Le champ de priorité peut être divisé en deux parties comme montré sur la figure 7.44.

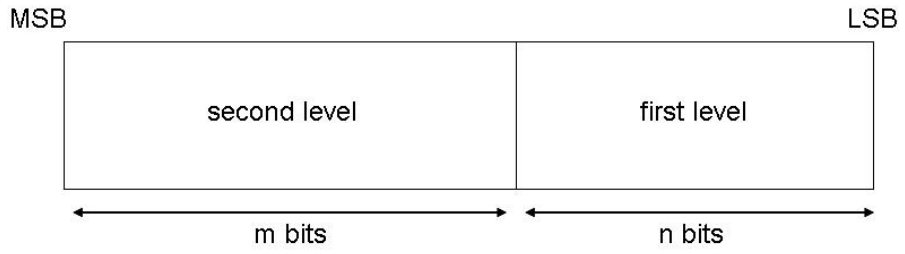


FIG. 7.44 – Structure du champ *identifiant* avec les priorités dynamiques

Le premier niveau représente la priorité statique alors que le second niveau représente la priorité dynamique qui peut être modifiée en fonction de l'urgence. La partie statique est représentée par une combinaison de bits fixés et non modifiables alors que la partie dynamique possède plusieurs combinaisons. L'ordonnancement compare dans un premier temps, la partie dynamique, et si l'urgence est identique, alors la partie statique est comparée.

Pour le quadrotor, il est important d'avoir de très bonnes performance en commande dans les situations de transition et quel que soit l'état du réseau. C'est notamment dans ces situations que la notion d'urgence pour les flux capteur et contrôleur doit être appliquée.

La structure de l'identifiant est coupée en deux parties avec $n = 7$ et $m = 4$. Nous considérons que les flux de l'application ont une notion d'urgence variable, et que la tâche externe possède quant à elle, une notion d'urgence constante. Pour pouvoir comparer avec le cas des priorités fixes (paragraphe 7.5.3), nous considérons que pour le premier niveau (partie statique), la priorité est donnée à la tâche externe (définie par 1111000), suivie par la tâche capteur (définie par 1111100) et enfin celle de la tâche contrôleur (définie par 1111110). Rappelons que toutes les mesures sont envoyées dans une seule trame.

Pour la partie dynamique, quatre valeurs sont considérées : 1111, 0111, 0011, et enfin 0000 représentant respectivement 0%, 50%, 78% et 100% du maximum de priorité disponible dans le second niveau. Rappelons que la priorité la plus grande correspond au mot binaire le plus petit.

7.5.4.2 Choix de l'"urgence"

Pour savoir si l'application est en état d'urgence ou non, nous avons besoin d'information sur le système. Une solution proposée dans la littérature est basée sur une fonction de saturation [38], [59], où les auteurs se sont focalisés sur les systèmes *SISO* (une entrée, une sortie) avec un correcteur proportionnel.

Cependant, le quadrotor est de type *MIMO* (multi-entrées, multi-sorties) et il doit être stabilisé en fonction des trois angles *roulis*, *tangage*, *lacet*. C'est pour cette raison que la stratégie doit tenir compte des trois angles. De plus, dans cette étude, le choix s'est porté sur l'utilisation de l'erreur comme information pour la notion d'urgence et non sur le signal de consigne comme proposé dans [38] car cela permet aussi de prendre en compte les dynamiques dues à des perturbations. En effet, si le quadrotor est dans un comportement transitoire (si les consignes changent ou une perturbation arrive) alors le signal d'erreur sera "grand". Lorsque le quadrotor est stabilisé, le signal d'erreur est proche de zéro. Dans [6], une nouvelle fonction a été proposée, définie par :

$$Prio = \begin{cases} P_{min}, si & \begin{cases} e_\phi < e_{threshold} & et \\ e_\theta < e_{threshold} & et \\ e_\psi < e_{threshold}. \end{cases} \\ P_2, si & \begin{cases} e_\phi \text{ et } e_\theta < e_{threshold} & ou \\ e_\phi \text{ et } e_\psi < e_{threshold} & ou \\ e_\theta \text{ et } e_\psi < e_{threshold}. \end{cases} \\ P_1, si & \begin{cases} e_\phi \text{ et } e_\theta > e_{threshold} & ou \\ e_\phi \text{ et } e_\psi > e_{threshold} & ou \\ e_\theta \text{ et } e_\psi > e_{threshold}. \end{cases} \\ P_{max}, si & \begin{cases} e_\phi > e_{threshold} & et \\ e_\theta > e_{threshold} & et \\ e_\psi > e_{threshold}. \end{cases} \end{cases} \tag{7.5.1}$$

où e_ϕ, e_θ et e_ψ désignent les erreurs par rapport respectivement au roulis, tangage et lacet. Afin de choisir $e_{threshold}$, la notion de qualité de contrôle (QoC) est considérée. En fait, le quadrotor est une application critique, par conséquent, nous considérons qu'une erreur supérieure à $e_{max} = 0.001$ degrés pour chaque angle est importante. Pour la simulation, le seuil est défini par $e_{threshold} = \frac{2}{3} e_{max}$ [59]. Comme l'attitude du quadrotor est représentée par trois angles, quatre niveau d'urgence sont considérés comme le montre l'équation

7.5.1 ainsi que la figure 7.45.

7.5.4.3 Résultat de simulation

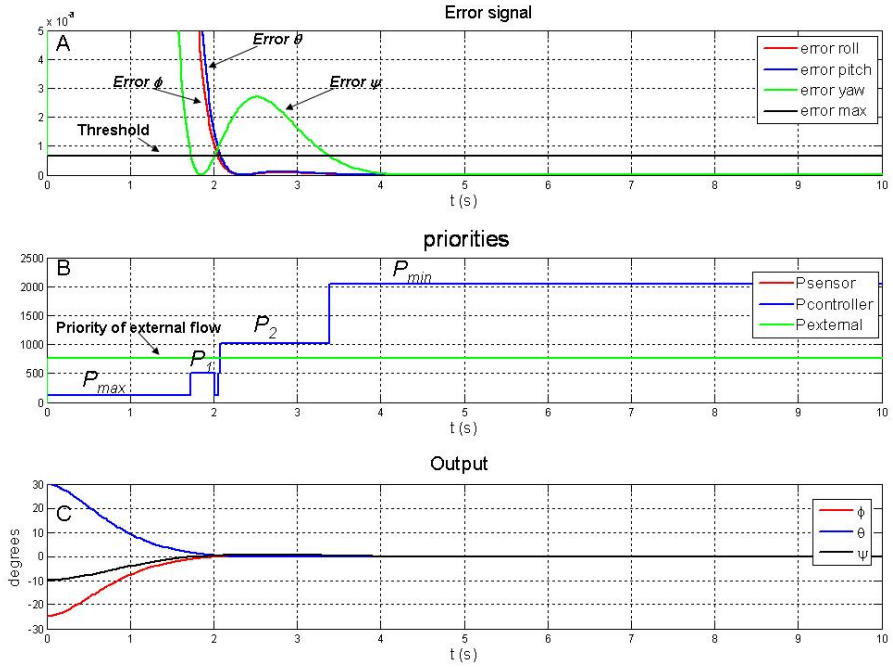


FIG. 7.45 – Attitude réelle du quadrotor avec les priorités dynamiques.

Les conditions de simulation sont inchangées par rapport au cas des priorités fixes.

Pour envoyer la première trame, la tâche capteur n'a aucune information sur l'état du système, c'est pourquoi, nous avons choisi d'utiliser la priorité maximale. Ainsi, la première trame est envoyée au contrôleur le plus rapidement possible. Deux scénarios vont être proposés, l'objectif étant de stabiliser le quadrotor dans la position $\phi = 0^{\circ}; \theta = 0^{\circ}; \psi = 0^{\circ}$.

La figure 7.45 montre le résultat de simulation du premier scénario. La figure 7.45.C montre l'attitude du quadrotor. Avec la notion de priorité dynamique, l'attitude du quadrotor est de nouveau stabilisée même avec un réseau surchargé. On remarque en outre que le quadrotor est stabilisé avec un temps de réponse à 5 % de 2 s, identique au cas avec un réseau dédié. La figure 7.45.A montre la valeur des trois erreurs d'angles. Entre $t = 0$ s et $t = 1.7$ s, la priorité du quadrotor est P_{max} car toutes les erreurs sont importantes. Entre $t = 1.7$ s et $t = 2.1$ s, la priorité du quadrotor est P_1 car seule l'erreur e_ψ est inférieure au seuil fixé. Entre $t = 2.2$ s et $t = 3.3$ s, la priorité du quadrotor est P_2 car cette fois-ci, c'est e_ϕ et e_θ qui sont inférieures au seuil. Enfin, à partir de $t = 3.3$ s, la priorité du quadrotor est P_{min} car toutes les erreurs sont inférieures au seuil.

Dans le second scénario (figure 7.46), une perturbation est introduite à l'instant $t = 5\ s$. Elle consiste à rajouter 2° sur chaque angle du quadrotor.

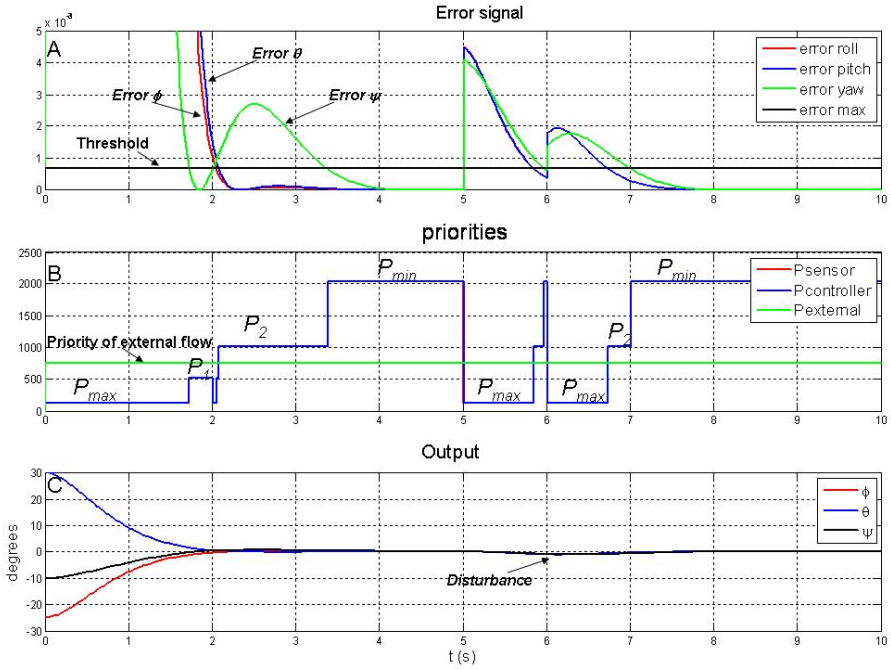


FIG. 7.46 – Attitude réelle du quadrotor avec les priorités dynamiques et une perturbation

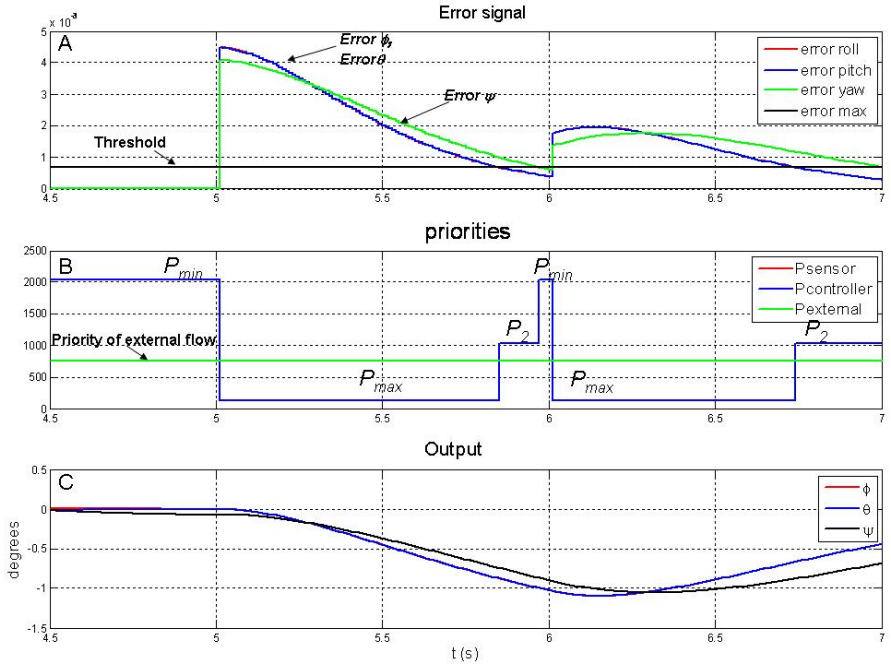


FIG. 7.47 – Zoom de la figure 7.46

La figure 7.46.C montre l'attitude réelle du quadrotor. Entre $t = 0\ s$ et $t = 5\ s$, les résultats sont identiques à ceux de la figure 7.45. Après $t = 5\ s$, c'est-à-dire après l'introduction de la perturbation (voir figure 7.47), toutes les erreurs sont supérieures au seuil. Par conséquent, entre $t = 5.5\ s$ et $t = 5.8\ s$, la priorité du quadrotor est P_{max}. Après $t = 7\ s$, la priorité est donnée à la tâche externe car toutes les erreurs sont de nouveau inférieures au seuil. Finalement, le quadrotor se stabilise dans la position désirée.

7.5.4.4 Conclusion sur l'utilisation des priorités dynamiques

Dans cette première partie sur la notion de commande du réseau, la politique d'ordonnancement basée sur les priorités dynamiques a été présentée. Comme montré précédemment, cette politique est avantageuse puisque très simple à mettre en œuvre donc très peu coûteuse. Cependant, elle devient très vite limitée lorsque le nombre d'applications utilisant la ressource de communication augmente. Lorsque plusieurs ressources ont, à un même instant donné, un besoin d'urgence, cette notion n'est plus valide puisque seule l'une d'entre elles utilisera le bus. En d'autres termes, on retomberait sur les mêmes problèmes que ceux du mécanisme de priorité fixe.

7.5.5 Quadrotor avec réseau CAN partagé : *(m-k)*-firm

Une politique d'ordonnancement basée sur le principe du (m-k)-firm peut être utilisée afin de réduire l'utilisation de la bande passante du réseau et aussi permet de garantir que les messages nécessaires vont être reçus. Cette politique d'ordonnancement a été développée au sein de équipe TRIO du Loria de Nancy [58], [27] partenaire du projet *Safe-NECS*. Cette stratégie est basée sur le principe suivant :

Définition 3. A chaque instant, au moins m paquets, contenant le signal de commande, sur k doivent être transmis [58].

La difficulté principale est la détermination des coefficients m et k, qui garantissent la stabilité du système. A partir de l'étude de la stabilité (voir annexe A), le système en boucle fermée peut tolérer une perte de 55 paquets consécutifs tout en restant stable. En conséquence, les paquets contenant le signal de commande peuvent être perdus sélectivement en accord avec les contraintes de la politique *(m-k)*-firm durant les périodes de surcharge du réseau. Le choix k est donc basé sur cette étude de stabilité et par conséquent, il est tout naturel de choisir $k = 55$. Il est important de remarquer que si $(k-m)$ paquets sur k sont perdus avec $m \in [1, k]$, le système reste stable. Dans [58], une distribution uniforme des *(k-m)* paquets a été proposée afin de garantir une meilleure QoC.

Afin de faciliter la présentation de la séquence obtenue, un mot binaire de longueur k constitué d'un alphabet de 1 et de 0 est utilisé. "1" correspond au cas où le paquet contenant le signal de commande est transmis alors que "0" est utilisé pour indiquer un paquet perdu. La séquence optimale de délivrance des paquets est définie par [58], [57] :

$$\lceil (n+1)\frac{m}{k} \rceil - \lceil n\frac{m}{k} \rceil \tag{7.5.2}$$

avec $n \in \{0, 1, 2, ...\}$. A partir de l'évolution du critère J défini en annexe (A.2.3) (voir figure 7.48), on peut remarquer que pour $l = 19$, le critère J est dix fois plus grand que pour le cas nominal $l = 1$. Cette dégradation est raisonnable pour notre application. C'est pourquoi, $m = 3$ a été choisit pour le nombre de mises à jour de la loi de commande du système et *(k-m)* $= 52$ paquets seront perdus systématiquement tous les $k = 55$ échantillons. Le mot binaire obtenu, appelé $(m-k)$ pattern, pour $m = 3$ et $k = 55$ est :

$$\underbrace{0\ 0\ 0 \ldots 01}_{f_1=18}\underbrace{0\ 0\ 0 \ldots 01}_{f_2=18}\underbrace{0\ 0\ 0 \ldots 01}_{f_3=19} \tag{7.5.3}$$

où f_i représente le nombre de paquets perdus + 1.

La loi de commande optimale minimisant le critère de coût est donnée par :

$$u_i = -L_i x(kh), \quad i = 1..3 \tag{7.5.4}$$

avec

$$L_{1,2} = \begin{pmatrix} 0.0253 & 0.0229 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.0253 & 0.0229 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.0253 & 0.0229 \end{pmatrix} \tag{7.5.5}$$

$$L_3 = \begin{pmatrix} 0.0258 & 0.0231 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.0258 & 0.0231 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.0258 & 0.0231 \end{pmatrix} \tag{7.5.6}$$

7.5.5.1 Résultat de simulation

L'objectif ici est de voir si le quadrotor est stabilisé dans la position souhaitée en présence d'un réseau chargé. Les conditions de simulation restent identiques aux cas précédents.

Le réseau est configuré tel que les capteurs sont guidés par la politique du (m-k)-firm. Les trames sont constituées des variables d'état[6] du système et elles sont envoyées dans un unique paquet. Le contrôleur est quant à lui guidé par les événements, c'est-à-dire qu'il attend d'avoir reçu un paquet venant des capteurs avant de calculer puis d'envoyer le

[6]Comme expliqué en annexe, on suppose que l'état du système est disponible directement. Il est bien évident que normalement l'état n'est disponible qu'après reconstruction de celui-ci par un observateur.

nouveau signal de commande. Enfin, la tâche externe est guidée par le temps. Elle envoie des paquets périodiquement sur le réseau avec une période $T = T_{ef}$ excepté aux instants $T = f_i\ h$. Avec ces conditions de simulation, la figure 7.49 montre les résultats obtenus pour la politique d'ordonnancement (m-k)-firm. Entre deux instant m, la tâche capteur et la tâche contrôleur sont au niveau bas (c'est-à-dire que la tâche est en sommeil) alors que la tâche externe varie entre le niveau haut (les données sont en cours de transmission sur le bus) et le niveau bas. Cependant, à chaque instant m, la tâche externe est au niveau bas alors que la tâche capteur est au niveau haut.

La figure 7.50 montre l'attitude réelle du quadrotor. On constate que le système est là encore stabilisé dans l'attitude désirée.

7.5.5.2 Conclusion sur l'utilisation du (m-k)-firm

Cette seconde technique d'ordonnancement est utilisable sous certaines conditions. Suite à cette étude, où l'ensemble est figé (cela revient à jouer sur la période d'échantillonnage), il faudrait définir le pattern (m-k) en ligne et en fonction des besoins à l'instant donné. Cependant pour réaliser cela, il faudra avoir préalablement calculé et stocké dans une table, toutes les lois de commande pour chaque valeur de m. Ceci peut paraître besogneux et par conséquent limite un peu cette méthode.

Remarque 19. D'un point de vue implémentation réelle, l'approche par priorité dynamique semble plus adéquate pour cette application car moins contraignante que celle du (m-k)-firm, notamment pour le stockage des tables de loi de commande optimale. D'autres politiques d'ordonnancement existent comme celle de l'ordonnancement régulé avec des changements de période d'échantillonnage [32] [79] mais n'ont pas été testées faute de temps.

7.5.6 Conclusion

Dans ce chapitre, plusieurs points ont été étudiés :

– premièrement, les différents réseaux (CAN, Ethernet, Ethernet commuté) ont été utilisés dans le cadre de la commande du quadrotor. Dès lors que le réseau est chargé ou partagé, le réseau Ethernet n'est pas vraiment le plus adapté dû aux problèmes de collisions et cela malgré le débit important. Pour limiter ces collisions, Ethernet Commuté a été utilisé. Malgré cela, les problèmes de collisions ne le rend pas assez flexible. Pour garantir les performances du système, il faut utiliser des politiques d'ordonnancement plus complexes telle que celle du WRR. Par contre, grâce à ses mécanismes de priorité, le réseau CAN correspond mieux à ce genre d'utilisation (malgré son débit moins important que les autres). Pour toutes ces raisons, le réseau utilisé par la suite a été CAN ;

- il a été aussi montré que le réseau doit nécessairement être considéré comme un composant du système complet et non pas comme un simple moyen de communication. D'un point de vue diagnostic, il a été montré que la génération des résidus doit être guidée à partir des événements et non plus supposée être régulièrement échantillonnée comme pour le cas des systèmes traditionnels. De plus, l'implémentation d'un nouvel indicateur permettant de diagnostiquer uniquement le réseau est proposé. Celui-ci est uniquement sensible à la perte d'informations et non pas à d'éventuels défauts capteurs (contrairement aux résidus générés pour les défauts capteurs [7]) ;
- enfin, l'étude sur la commande du réseau a été abordée. Celle-ci permet de garantir, à partir de techniques d'évaluation de la qualité de contrôle, les besoins de l'application. Deux techniques d'ordonnancement ont été utilisées afin de garantir cette qualité de contrôle. La première permet de modifier les priorités en fonction de l'urgence. Cependant, cette technique peut être assez vite limitée en fonction du nombre d'applications à contrôler. La deuxième, basée sur le (m-k)-firm, permet d'utiliser la ressource que lorsque cela est nécessaire. L'inconvénient majeur de cette technique est la mise en mémoire des lois de commandes optimales pour garantir les nécessités de l'application.

[7] Pour les résidus générés pour les capteurs, une perte de paquet est équivalente à un défaut capteur de type valeur bloquée. C'est pour cette raison que ces résidus sont sensibles aux pertes d'informations

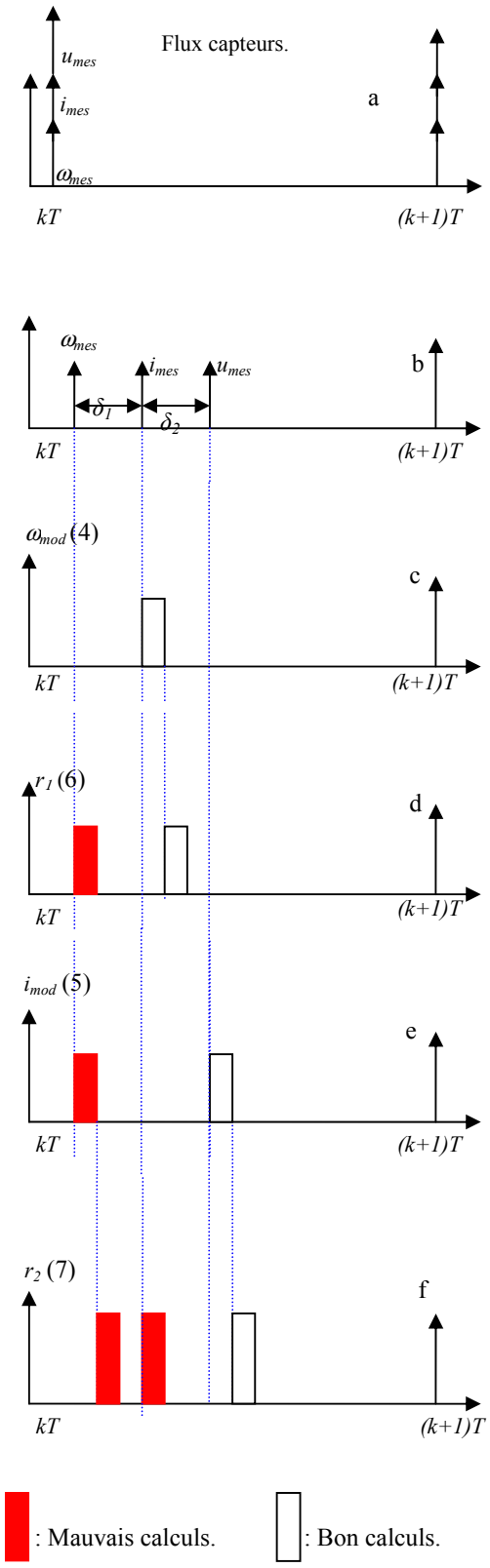


FIG. 7.17 – Chronogramme pour expliquer le problème de fausses alarmes visibles sur la figure7.16

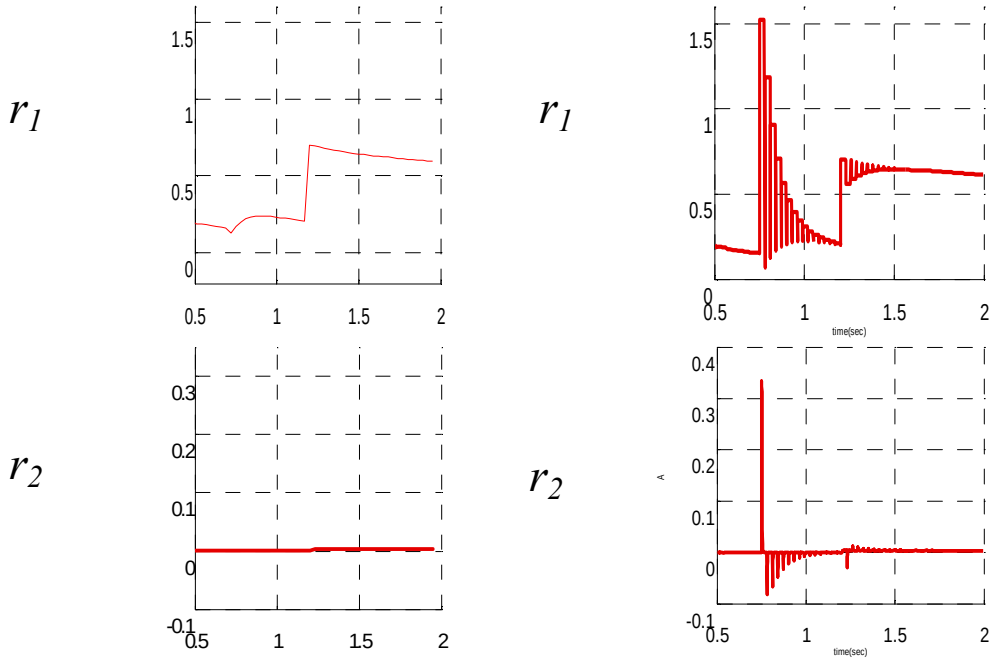


FIG. 7.18 – Génération des résidus r_1 et r_2 avec un défaut sur le capteur de vitesse angulaire introduit à l'instant $t = 1.25\ s$ et d'amplitude $0.5\ rad/s$ (gauche : sans réseau ; droite : avec réseau)

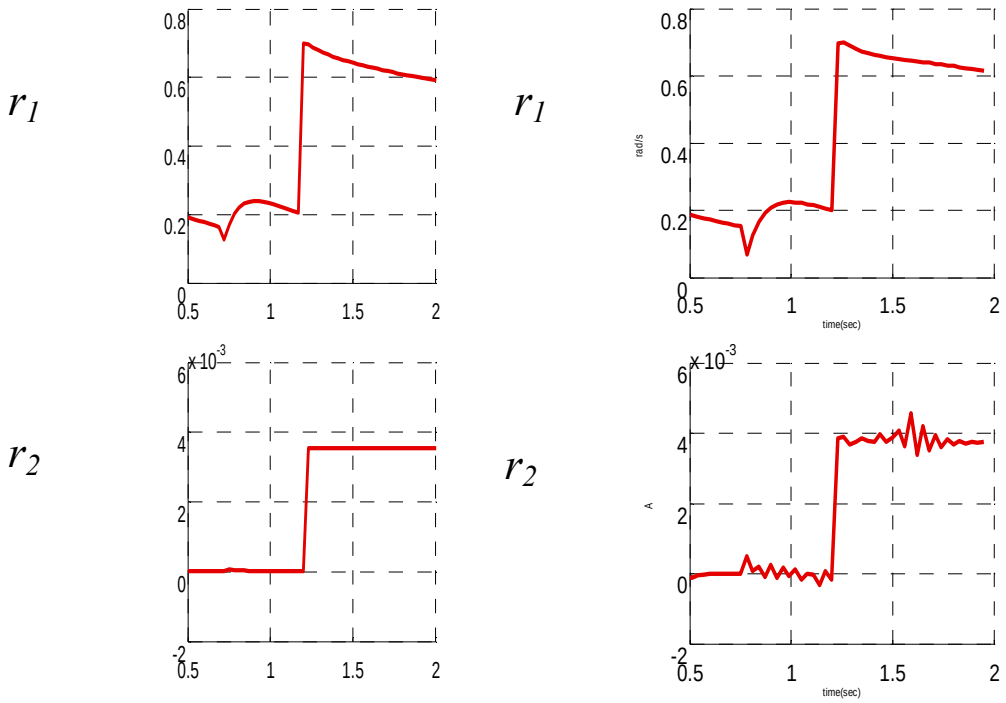


FIG. 7.19 – Génération des résidus r_1 et r_2 avec un défaut sur le capteur de vitesse angulaire introduit à l'instant $t = 1.25\ s$ et d'amplitude $0.5\ rad/s$ (gauche : sans réseau ; droite : avec réseau)

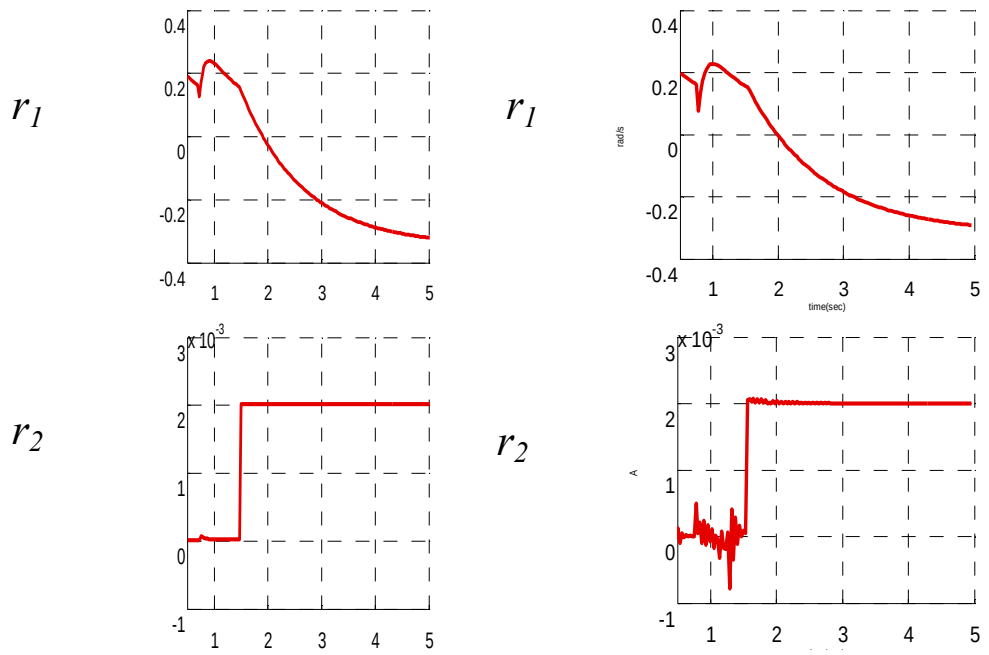


FIG. 7.20 – Génération des résidus r_1 et r_2 avec un défaut sur le capteur de courant introduit à $t = 1.5\ s$ et d'amplitude 2 mA (gauche : sans réseau ; droite : avec réseau)

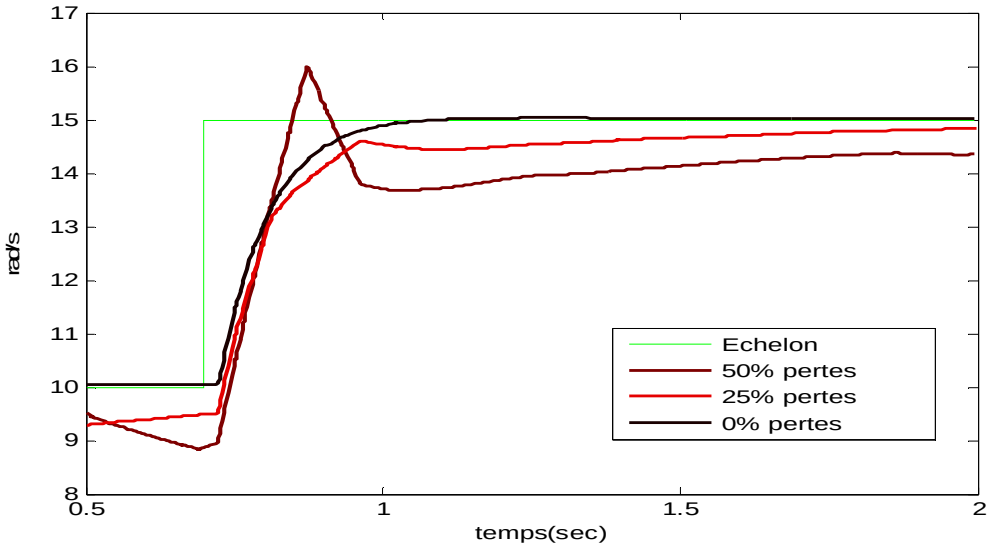


FIG. 7.21 – Réponse du système en boucle fermée en présence de perte de paquets

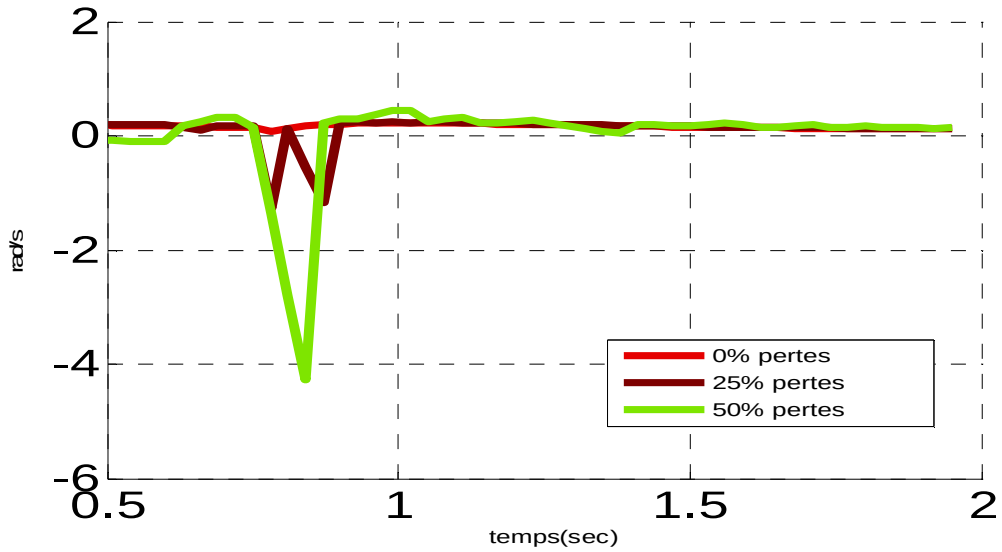


FIG. 7.22 – Résidu r_1 en présence de perte de paquets

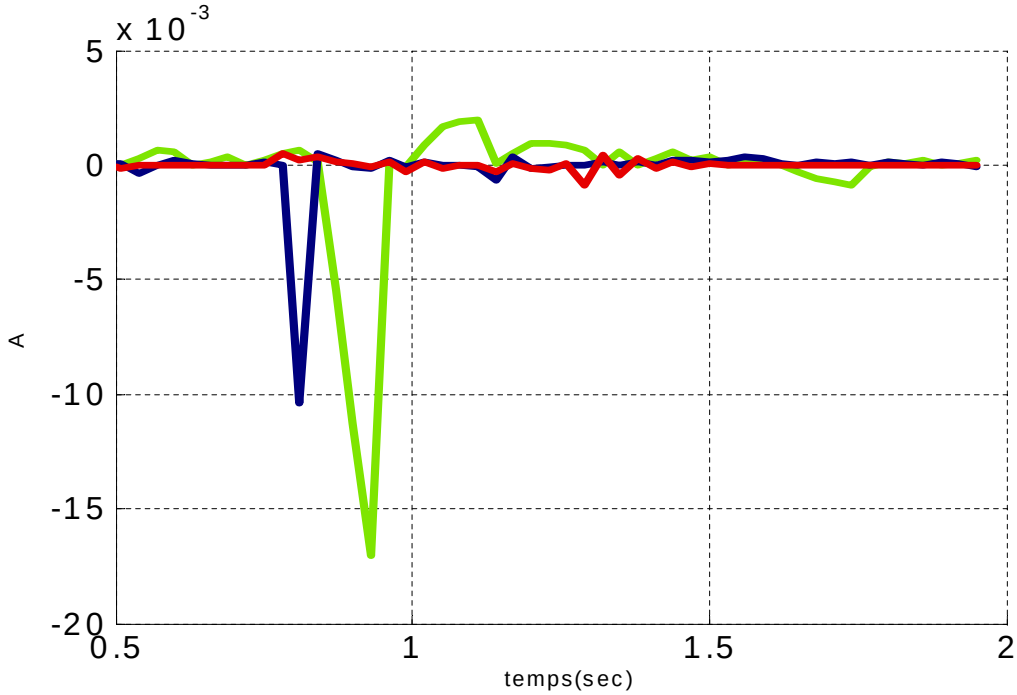


FIG. 7.23 – Résidu r_2 en présence de perte de paquets

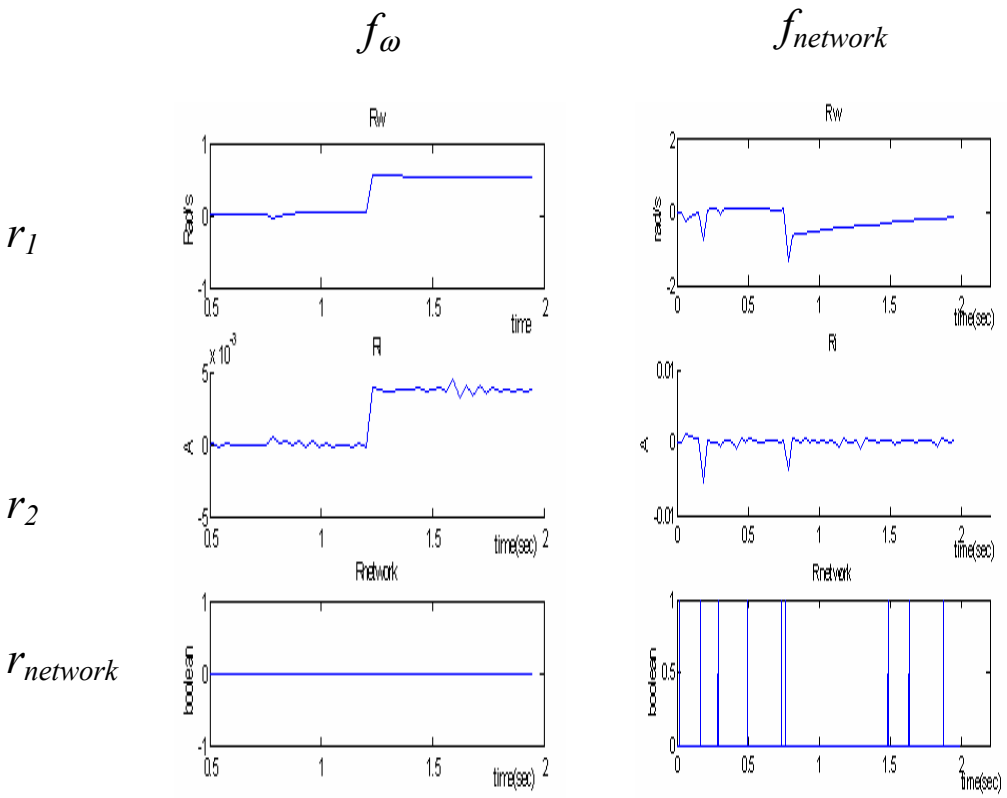


FIG. 7.24 – Défaut réseau et défaut capteur localisé

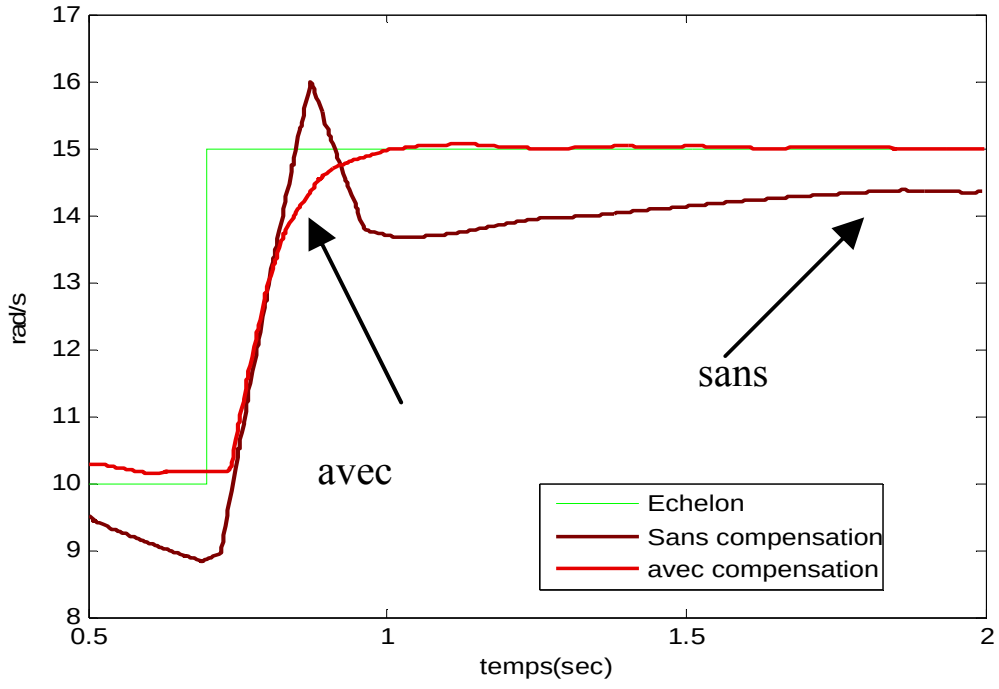


FIG. 7.25 – Défaut réseau et défaut capteur localisé

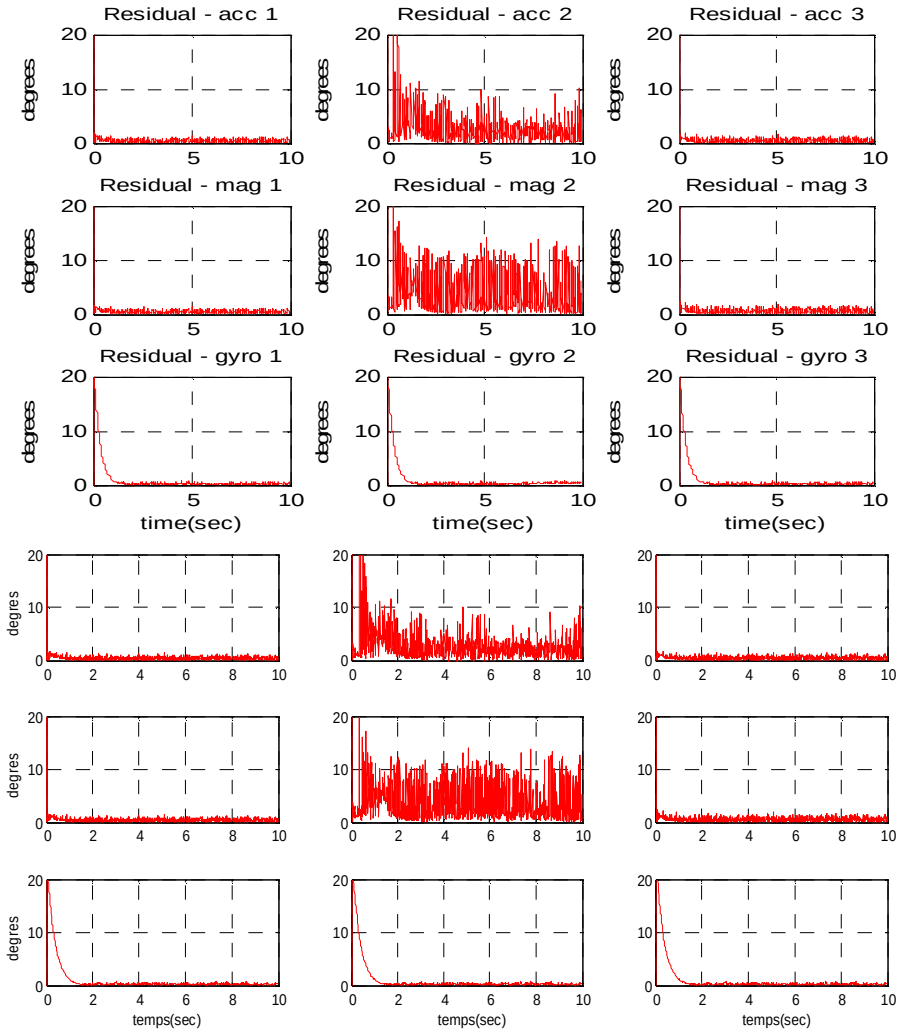


FIG. 7.35 – (haut) Sans réseau, (bas) Avec réseau dédié

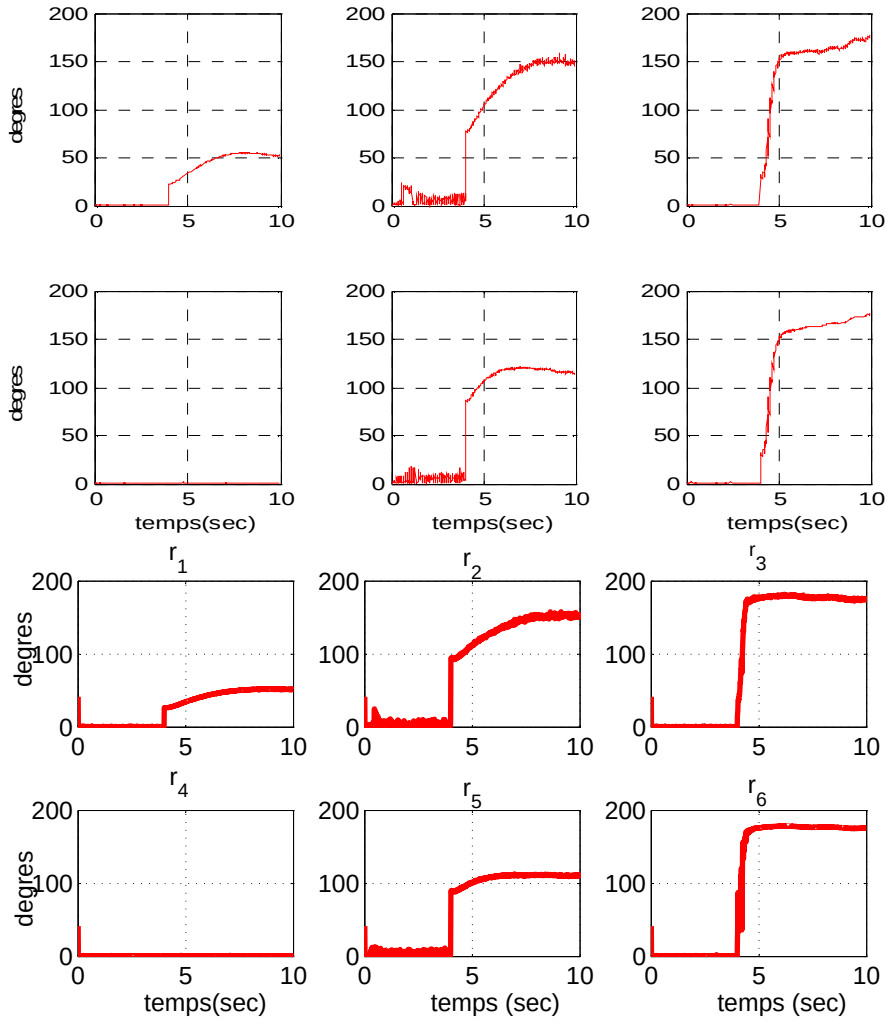


FIG. 7.36 – (haut) Sans réseau, (bas) Avec réseau dédié

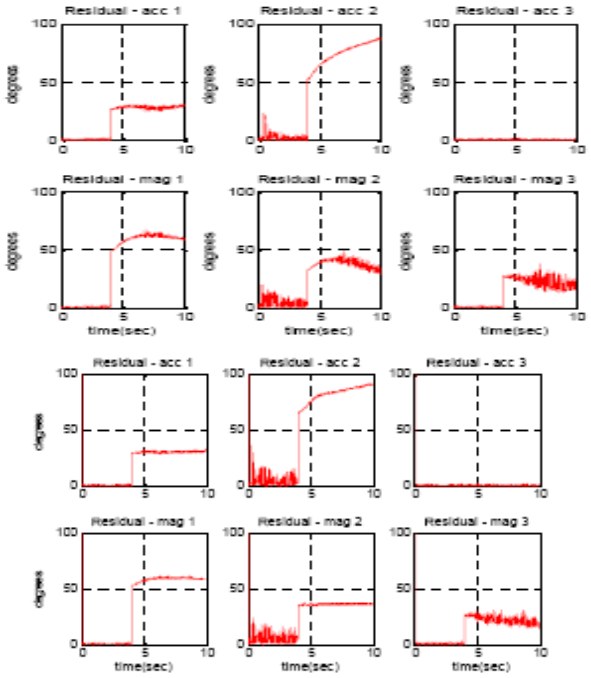


FIG. 7.37 – (haut) Sans réseau, (bas) Avec réseau dédié

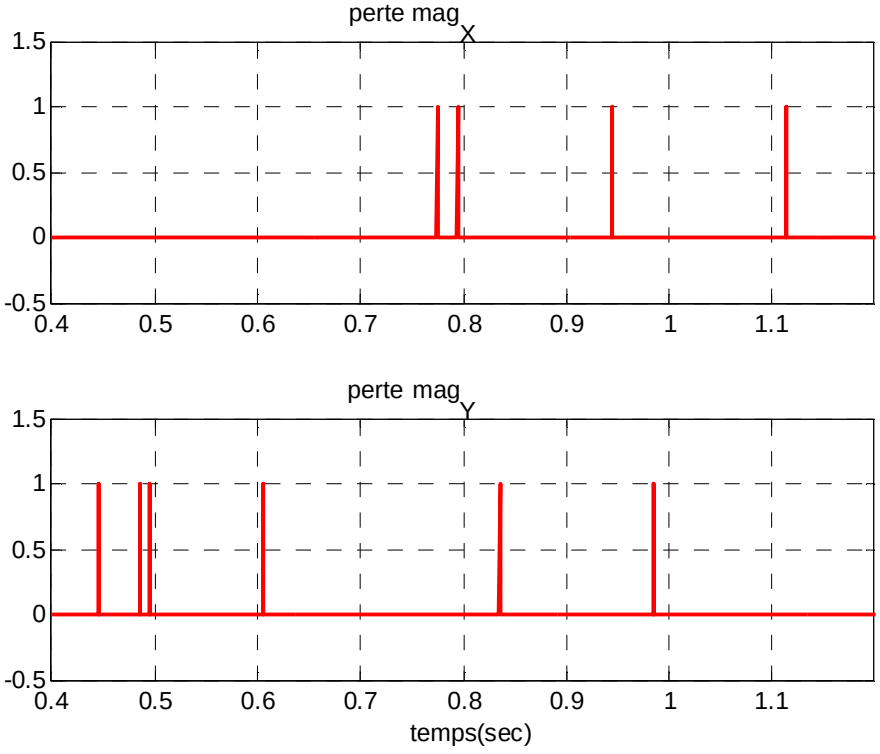


FIG. 7.38 – Indicateur de perte de paquets

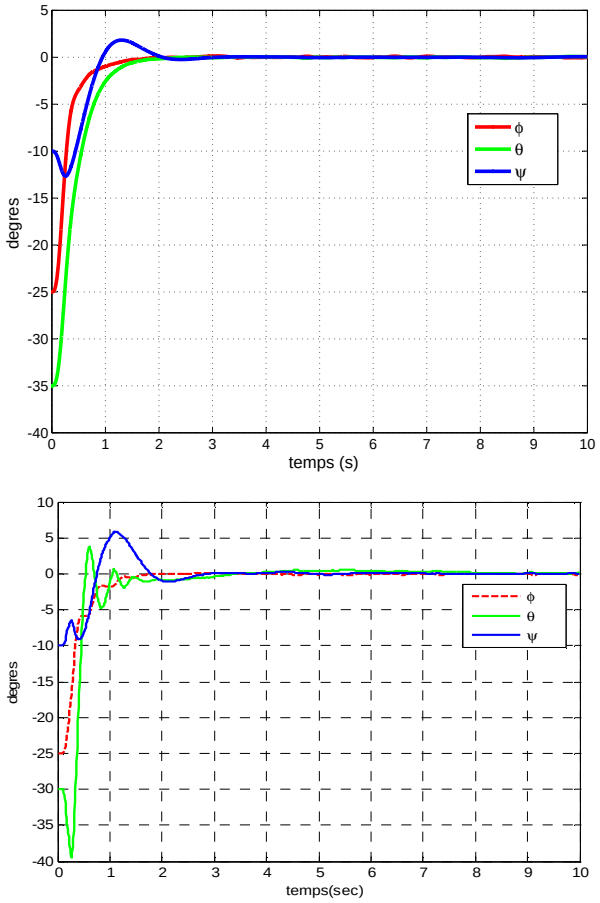


FIG. 7.39 – (haut) Sans réseau, (bas) Avec réseau dédié

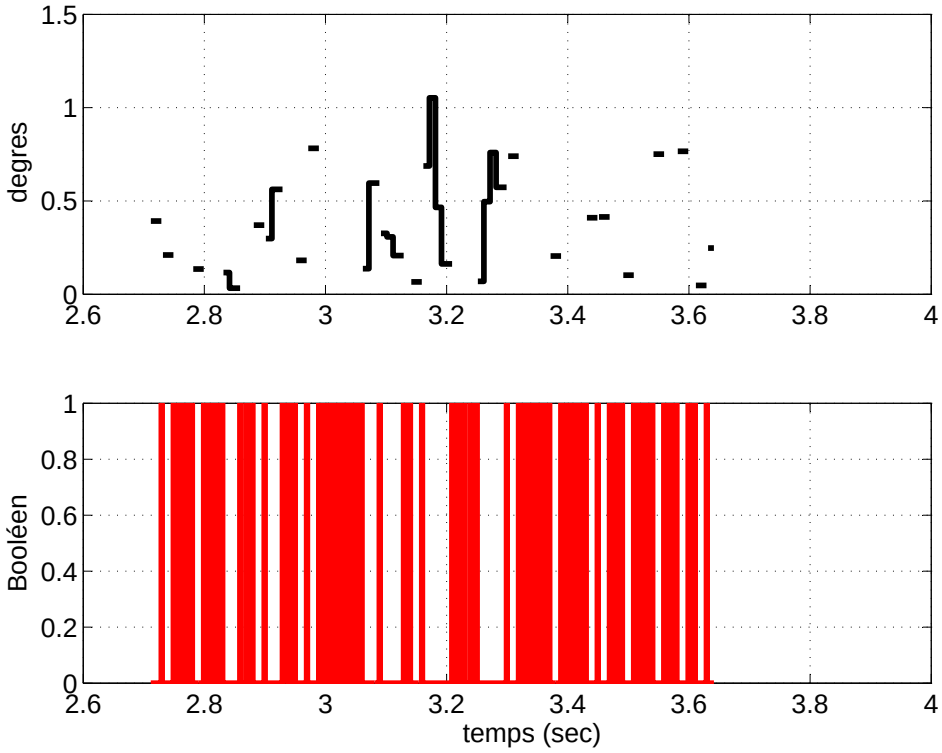


FIG. 7.40 – Résidu r_1 avec perte de paquets

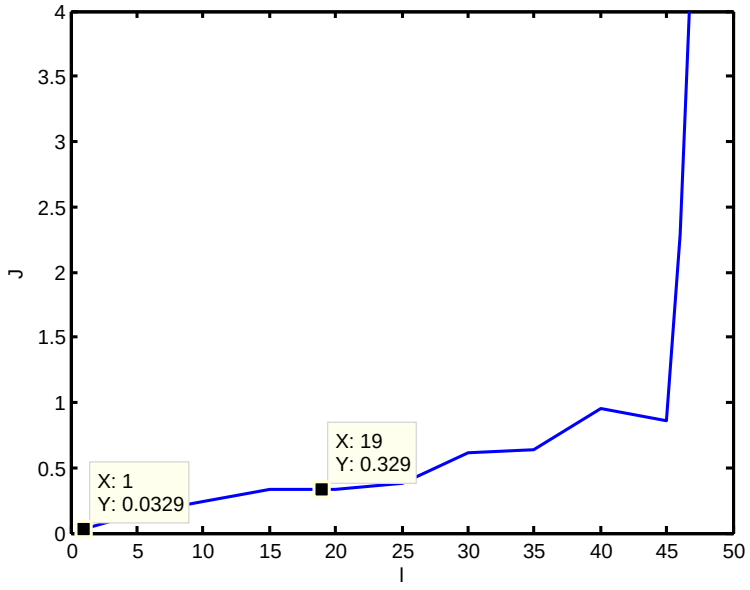


FIG. 7.48 – Fonction coût J en fonction de la mise à jour des consignes aux actionneurs

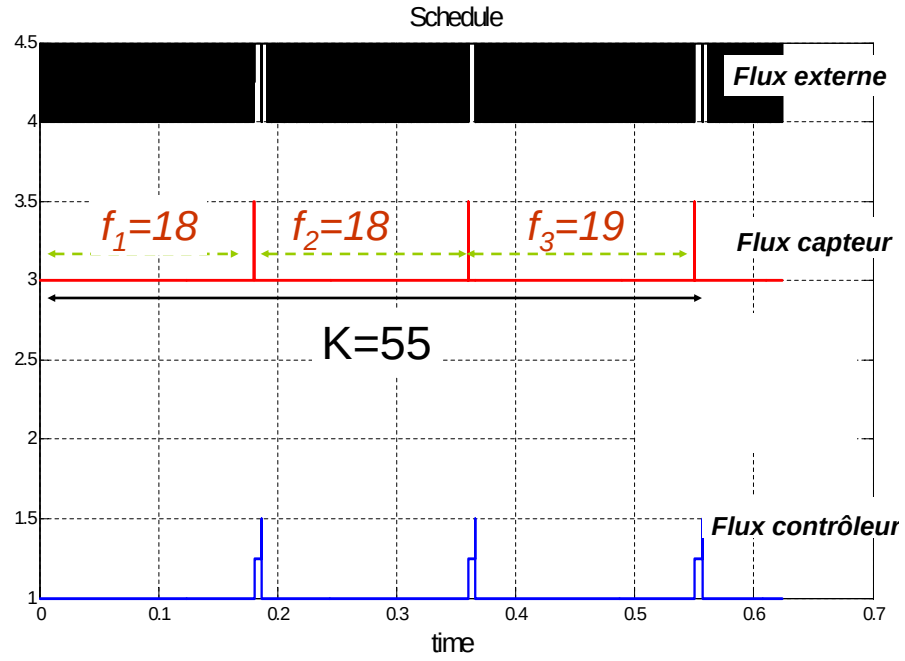


FIG. 7.49 – Ordonnancement des trames avec la politique du *(m-k)*firm

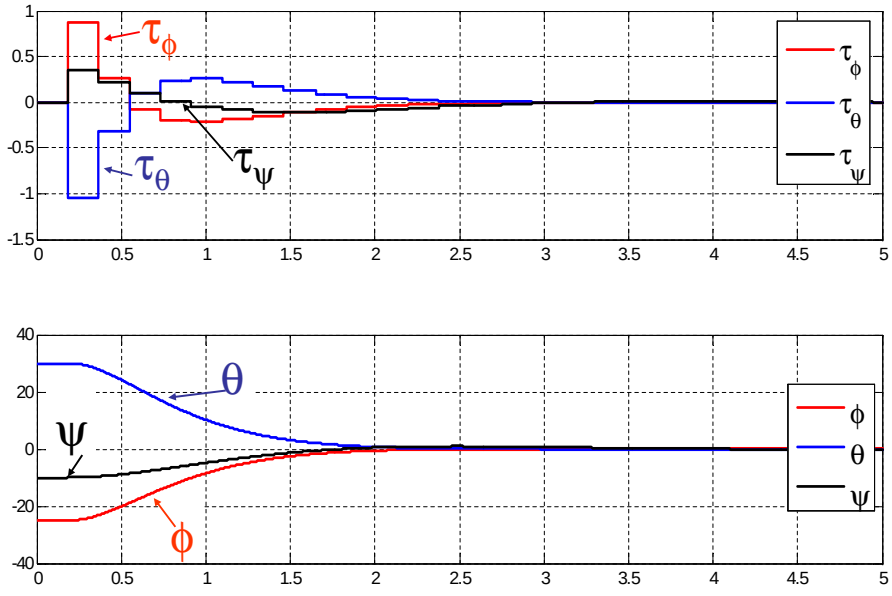


FIG. 7.50 – Attitude réelle du quadrotor avec la politique du *(m-k)*firm

Chapitre 8

Expérimentation

Les résultats présentés dans le chapitre 7 ne sont que des résultats de simulation. Le quadrotor étant un système fragile, son utilisation doit être sécurisée afin de garantir la sécurité des personnes travaillant aux alentours et aussi pour le protéger. C'est pourquoi, nous avons souhaité valider les résultats obtenus en simulation avec une structure dite "Hardware in the loop" [7] avant l'implémentation réelle sur le système.

Cette étape supplémentaire permet :

1. de tester les algorithmes de commande et de diagnostic en temps réel ;

2. de valider la modélisation du réseau faite par TrueTime ;

3. de protéger le système.

L'implantation "Hardware in the loop", représentée sur la figure 8.1, consiste à simuler le modèle mécanique du quadrotor à travers un simulateur implanté sous un PC Linux alors que les algorithmes de commande, observation et diagnostic sont quant à eux implantés dans le PowerPc de Phytec [49]. Les deux calculateurs communiquent à travers un vrai réseau.

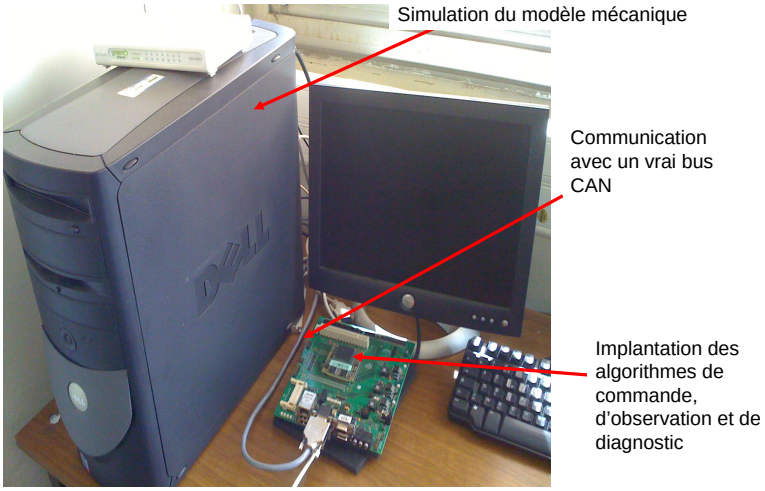


FIG. 8.1 – Implémentation "Hardware in the loop"

Pour réaliser cette expérience, l'INRIA Rhône-Alpes, un des partenaire du projet, a fourni le logiciel intitulé Orccad [81]. ORCCAD est un environnement logiciel permettant de concevoir et de mettre en œuvre le contrôle et la commande d'un système robotique complexe. Il permet également la spécification et la validation des missions à réaliser par ce système. Il est principalement destiné aux applications temps réel critiques (qui exige une garantie a priori du respect de ces diverses contraintes) en robotique, dans lesquelles les aspects relevant de l'automatique (les asservissements, les commandes) sont amenés à interagir étroitement avec ceux manipulant des événements discrets [52].

8.1 Quadrotor sous l'environnement Orccad

Maintenant, nous allons décrire l'implantation de la commande de l'observation et du diagnostic du Quadrotor en boucle fermé sous l'environnement Orccad.

La figure 8.2 montre le bloc diagramme de commande, d'observation et de diagnostic du quadrotor. Sur cette figure, les boites bleues représentent les modules que l'utilisateur peut programmer. Ces modules sont reliés entres eux par leurs entrées/sorties, respectivement (bleu/rouge). La boite intitulée $X4\_GPS\_Phr$ représente le modèle du quadrotor. Celui-ci est implanté dans un PC externe alors que toutes les autres boites sont implantées dans le PowerPC.

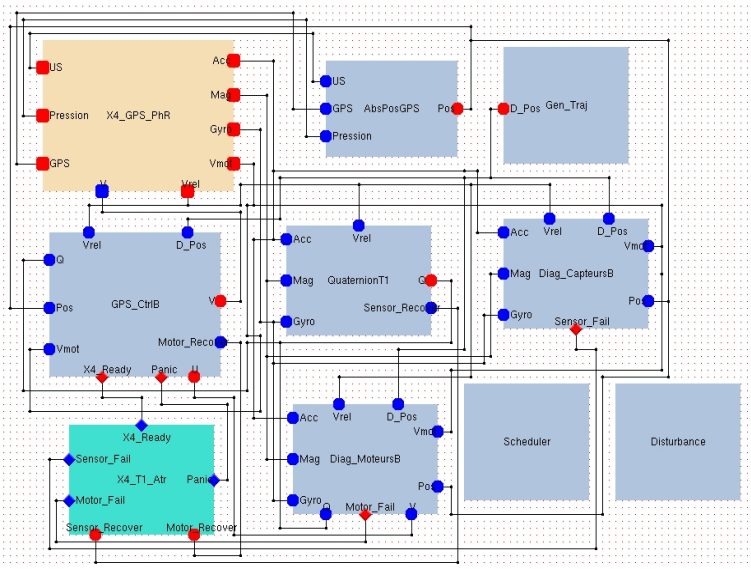


FIG. 8.2 – Bloc diagramme de la commande, de l'observation et du diagnostic sous l'environnement Orccad

Les principales fonctions sont maintenant décrites :

- la commande en attitude débute par la "lecture" des mesures de capteur (accéléromètres, magnétomètres et gyromètres) à la sortie de la boite $X4\_GPS\_PhR$. Les mesures sont utilisées par le module de Quaternion ($QuaternionT1$). Dans ce module est implanté l'observateur non linéaire. Ce module estime l'attitude q_{est} du quadrotor. Cette attitude est ensuite transférée au module GPS_Ctrl_B afin de calculer la nouvelle commande. Les vitesses désirées ω_{Mi}^{ref} sont ensuite envoyées à l'entrée du module XA_GPS_PhR à travers le réseau CAN ;
- le module $Diag\_Capteur\_B$ contient le module de diagnostic. Notons que pour le moment, seul le module de diagnostic *Algo 1* est implanté. Une fois qu'un défaut est détecté et localisé, alors une exception via la sortie $SensorFail$ est générée afin de signaler ce défaut. Cette exception est envoyée à l'automate $X4\_T1\_Atr$ qui va la transférer au module concerné. Pour un défaut capteur, l'exception est renvoyée au module $QuaternionT1$. Dans ce cas, l'observateur peut être modifié afin de prendre en compte ce défaut [63],[14] ;
- le module $Scheduler$ permet d'implémenter un "feedback scheduler". La politique d'ordonnancement du (m-k)-firm vient d'être implantée au sein de ce module. On peut aussi prévoir d'implanter la politique des priorités dynamiques ;
- le module $Disturbance$ permet de générer une charge supplémentaire agissant soit sur le CPU soit sur le réseau. Cette charge permet de tester les deux approches *(m-k)*-firm et priorités dynamiques ;
- le module Gen_Traj est un module non utilisé pour le moment, pouvant servir pour des extensions potentielles de la commande du quadrotor.

8.2 Résultats

Dans cette section, les résultats obtenus avec le "Hardware in the loop" sont présentés et comparés avec ceux obtenus sous Matlab/Simulink et TrueTime. Les résultats sont présentés sous la forme des angles de Cardan car plus représentatifs mais les algorithmes ont été implantés sous le formalisme du quaternion unitaire. Le réseau CAN possède les caractéristiques suivantes :

- une vitesse de 1 $Mbits/s$;
- une période d'échantillonnage de $T_s = 10\ ms$ pour l'envoi des mesures produites par les capteurs ;
- les trames de la tâche capteur et contrôleur sont de longueur égale à 64 bits et chaque donnée est envoyée séparément.

Deux scénarios sont présentés. Le premier est sans défaut et l'autre avec un défaut sur le $gyro_X$. Pour ces deux scénarios, le réseau est dédié à l'application.

8.2.1 Sans perte, sans défaut et réseau dédié

Dans ce scénario, le cas nominal est considéré. Dans ce cas, le quadrotor doit se stabiliser dans la position $\phi = 0^\circ$, $\theta = 0^\circ$ et $\psi = 0^\circ$ avec une initialisation égale à $\phi = 120^\circ$, $\theta = -10^\circ$ et $\psi = 50^\circ$. La figure 8.3 montre le résultat obtenu : à gauche le cas

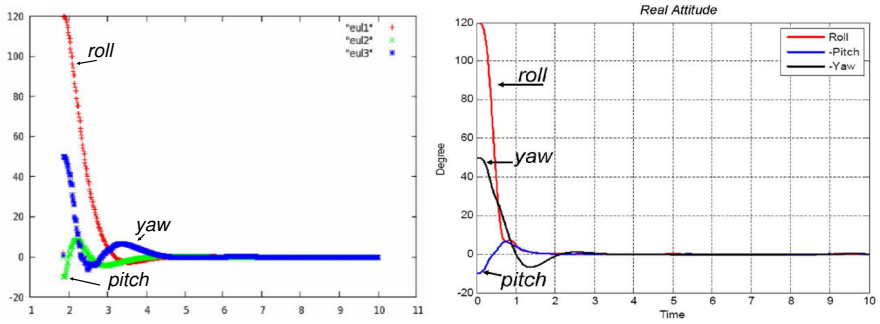


FIG. 8.3 – (gauche) Attitude du quadrotor avec le "Hardware in the loop", (droite) Attitude du quadrotor avec Matlab et TrueTime

"Hardware in the loop" et à droite le cas "Matlab/Simulink". On remarque que pour les deux cas le système se stabilise en 2 s. Ceci nous permet de conclure que le réseau n'influence pas le système : les retards induits par celui-ci et les pertes sont négligeables. Les résultats de simulation via TrueTime sont validés par cette expérimentation plus proche de la réalité.

8.2.2 Sans perte, avec un défaut capteur sur le $gyro_X$ et réseau dédié

Dans cette expérience, nous nous focalisons sur le cas où un défaut sur le $gyro_X$ est introduit à partir de $t = 5\ s$. Ce défaut est introduit au sein du module $X4\_GPS\_PhR$. On constate sur la figure 8.4 qu'avant l'apparition du défaut, les résidus générés pour la surveillance des défauts gyromètre sont négligeables et qu'après celle-ci, deux des trois résidus, r_8 et r_9, sont sensibles à ce défaut. En fait, seul le résidu r_7 généré sans prendre en compte ce capteur est insensible. Après comparaison avec la table de signature (voir table 4.3), on peut conclure que le défaut est localisé.

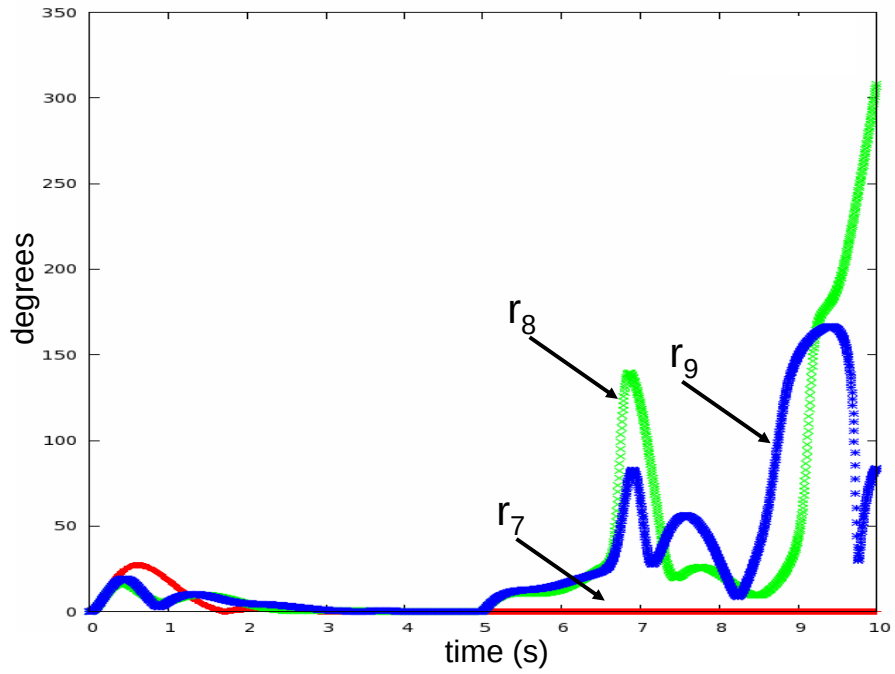


FIG. 8.4 – Résidus pour la surveillance des gyromètres

8.3 Implémentation réelle

Dans cette section, les premiers résultats provenant de l'implémentation réelle sont présentés. Le réseau CAN possède les mêmes caractéristiques qu'en 8.2.

Les données provenant de la centrale d'attitude sont envoyées au PowerPC avec le format suivant :

- Un champ identifiant unique par mesure ;
- Le champ données structuré en deux parties :

1. Les 2 premiers octets contiennent le numéro de l'échantillon k de la donnée y(k) ;
2. Les 2 suivants sont constitués de la mesure elle-même.

Le PowerPC gère un tableau correspondant aux 9 mesures qu'il doit mettre à jour par période d'échantillonnage. Dès qu'une trame est arrivée au PowerPC, celui-ci la traite de la façon suivante :

1. Extraction du numéro de l'échantillon ;
2. Si le numéro de l'échantillon est égal à l'indice du temps courant k alors :
   - identification du paquet grâce au champ Identification de la trame ;
   - conversion de la donnée reçue ;
   - stockage dans le tableau de mesure de la valeur correspondant au numéro de l'identifiant ;
   - incrémentation de 1 du compteur nbrmesures permettant de savoir le nombre de mesures d'indice de temps k reçues ;
3. si le compteur nbrmesures est égal à 9 alors toutes les données de la centrale ont été reçues donc les opérations suivantes peuvent être effectuées :
   - écriture des mesures dans un fichier ;
   - écriture des mesures sur les ports de sorties afin que chaque module de la figure 8.2 puisse les utiliser ;
   - mise à 0 du compteur nbrmesures ;
   - calcul de la commande et du diagostic.
4. Sinon :
   - Une ou plusieurs mesures n'ont pas été reçues donc aucune mesure n'est stockée dans le fichier de mesure ;
   - Mise à 0 du compteur nbrmesures.

Les figures 8.5, 8.6, 8.7 montrent les données réelles. Le quadrotor est dans cet expérience horizontal, ce qui correspond à sa condition d'équilibre. On doit alors trouver que les mesures des gyromètres sont nulles (pas de mouvement), que les mesures des accéléromètres sont égales à [0 0 1] et celles des magnétomètres à $[0.5\ 0\ \frac{\sqrt{3}}{2}]$. Les données ont été envoyées par la centrale d'attitude au PowerPc à travers le réseau CAN. On peut constater que les mesures correspondent à celles utilisées dans le simulateur. Cependant, on peut remarquer que l'axe y des mesures des accéléromètres et magnétomètres comportent un léger biais. On remarque de plus que le niveau de bruit est très faible.

La figure 8.8 montre l'estimation du quaternion obtenue par la tâche QuaternionT1. La valeur théorique au repos est [1 0 0 0]. On peut remarquer la bonne qualité de cette estimation.

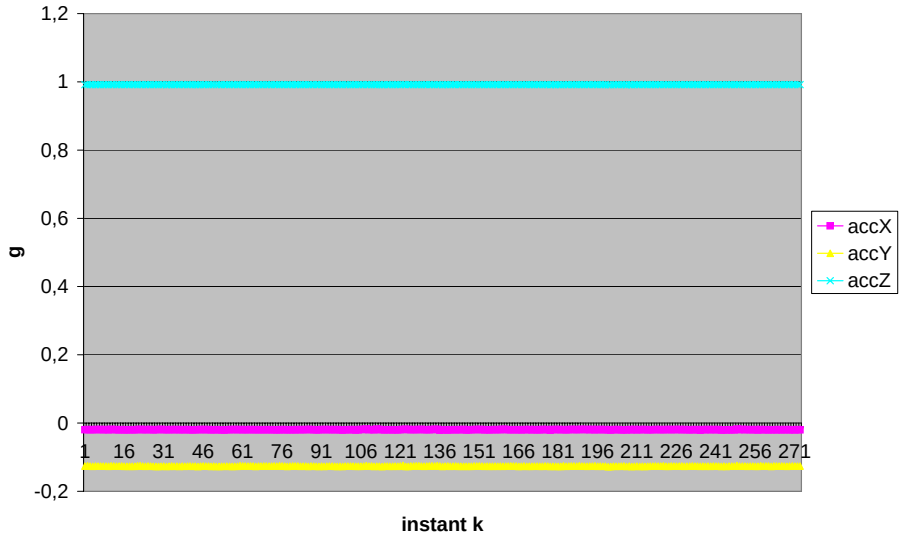


FIG. 8.5 – Mesures des accéléromètres provenant de la centrale d'attitude MAG3

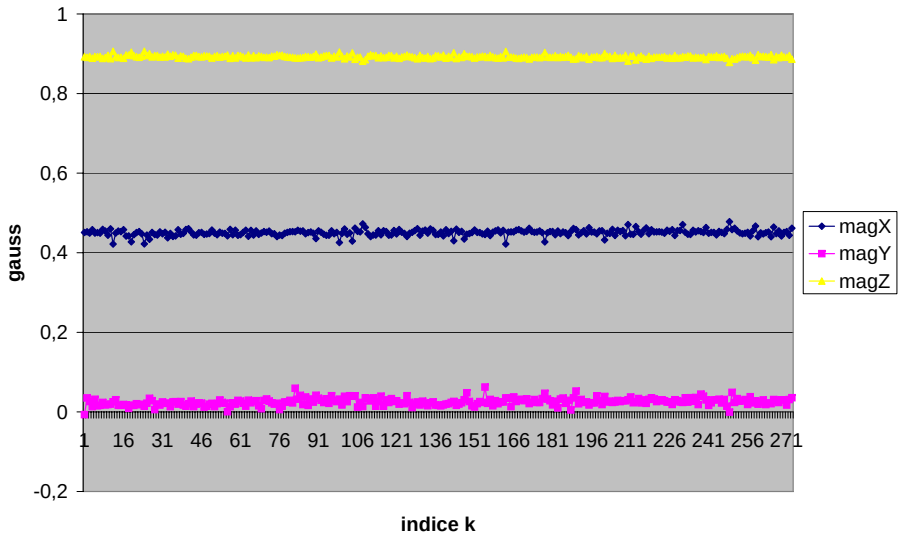


FIG. 8.6 – Mesures des magnétomètres provenant de la centrale d'attitude MAG3

8.4 Module de diagnostic avec les données de l'implémentation réelle

Grâce aux données inscrites dans le fichier, le module *Algo 2* peut être lancé sous l'environnement Matlab en utilisant le fichier obtenu.

La figure 8.9 montre les résultats obtenus pour les six vecteurs de résidus générés pour la surveillance des accéléromètres et magnétomètres. La figure 8.10 montre les résidus obtenus pour la surveillance des gyromètres. Les résidus sont de l'ordre de grandeur du biais de mesure, le seuil de détection est donc facile à fixer. La figure 8.11 montre le

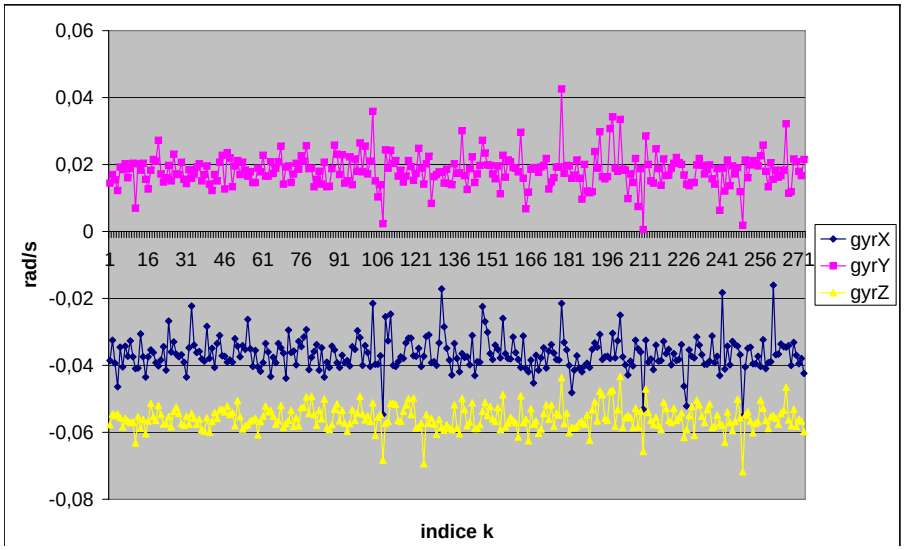


FIG. 8.7 – Mesures des gyromètres provenant de la centrale d'attitude MAG3

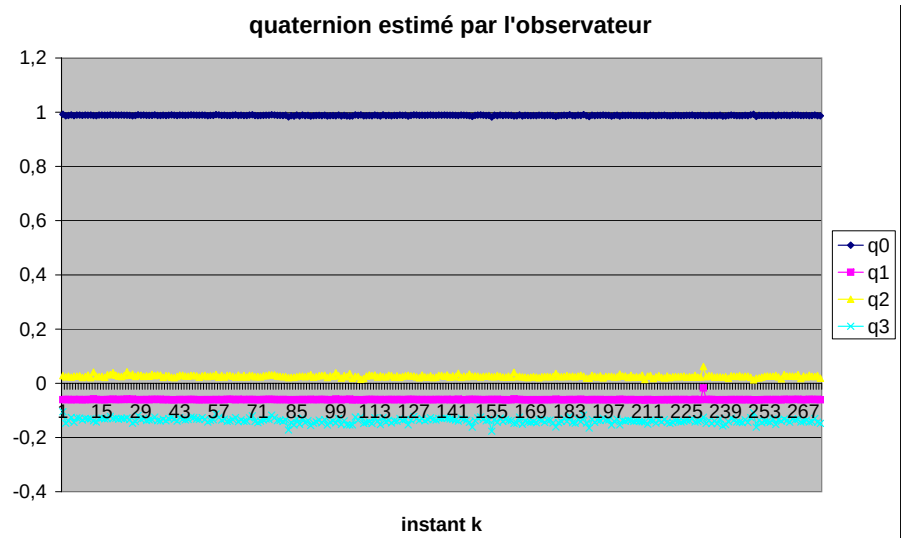


FIG. 8.8 – Estimation du quaternion par l'observateur

résultat obtenu lorsqu'un défaut sur mag_X est introduit. Ce défaut est un biais d'amplitude 0.5 gauss rajouté artificiellement au fichier de mesure. On peut constater que ce défaut est détecté et localisé car les composantes de tous les vecteurs R_i sont toutes différentes de zéro sauf pour R_4 qui quant à lui est composé d'un vecteur contenant cinq zéro et une composante dont l'amplitude est égale à celle du défaut généré.

La figure 8.12 montre le résultat obtenu lorsqu'un défaut sur le $gyro_X$ est introduit. Ce défaut est un biais d'amplitude 0.5 rad/s. On peut constater que seul le résidu $r_\omega(1)$ est sensible à ce défaut et que les deux autre sont nuls. De plus, l'amplitude de $r_\omega(1)$ correspond à l'amplitude du défaut introduit.

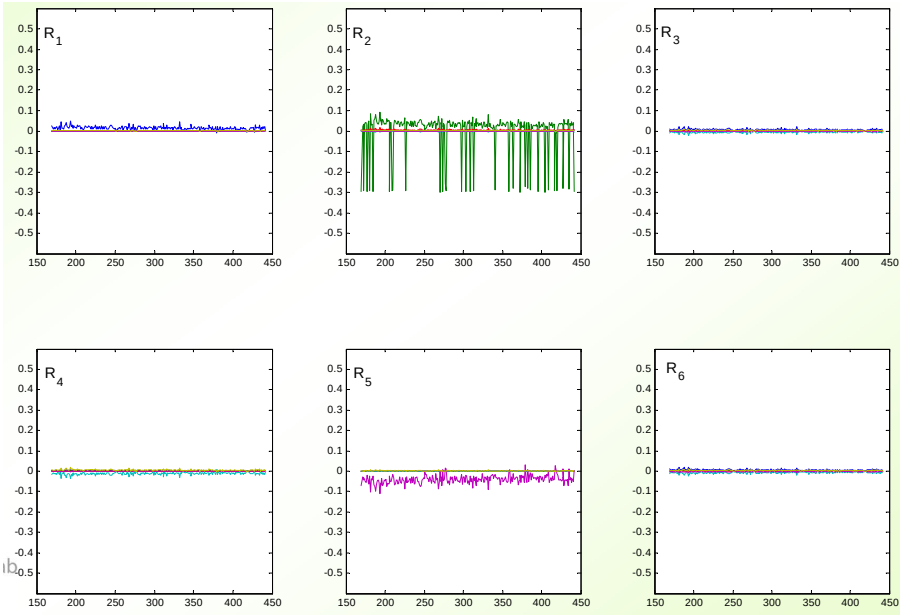


FIG. 8.9 – Générations des résidus par *Algo 2* avec des données réelles (cas sans défaut)

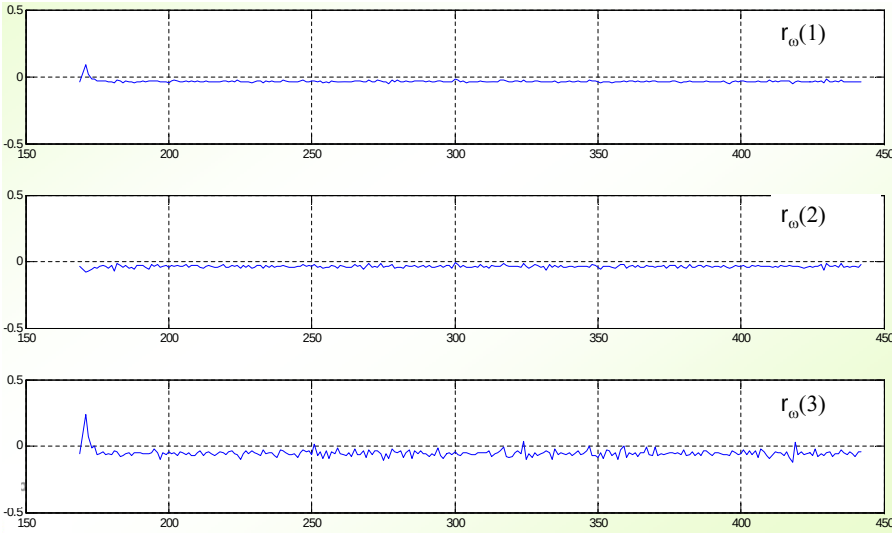


FIG. 8.10 – Générations des résidus pour les gyromètres par *Algo 2* avec des données réelles (cas sans défaut)

8.5 Conclusion

Dans ce chapitre, une expérience "Hardware in the loop" a été réalisée. La communication entre les modules de diagnostic, commande et observation et le modèle du quadrotor a été réalisée avec un vrai réseau CAN dans une véritable architecture temps réel. Ceci nous a permis de valider les résultats de simulation obtenus sous Matlab/Simulink et TrueTime.

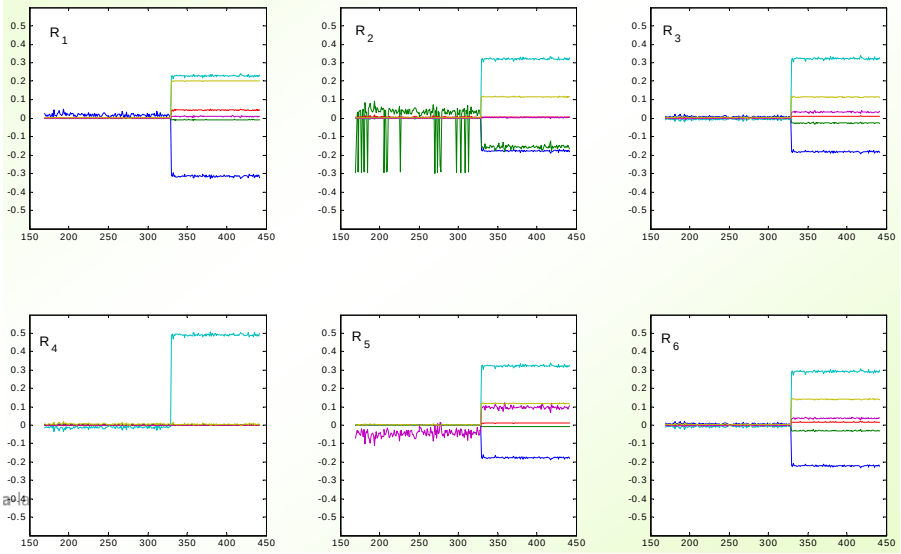


FIG. 8.11 – Générations des résidus R_i, $i = 1 : 6$, (6 composantes par figure) par *Algo 2* avec des données réelles et un défaut sur mag_X

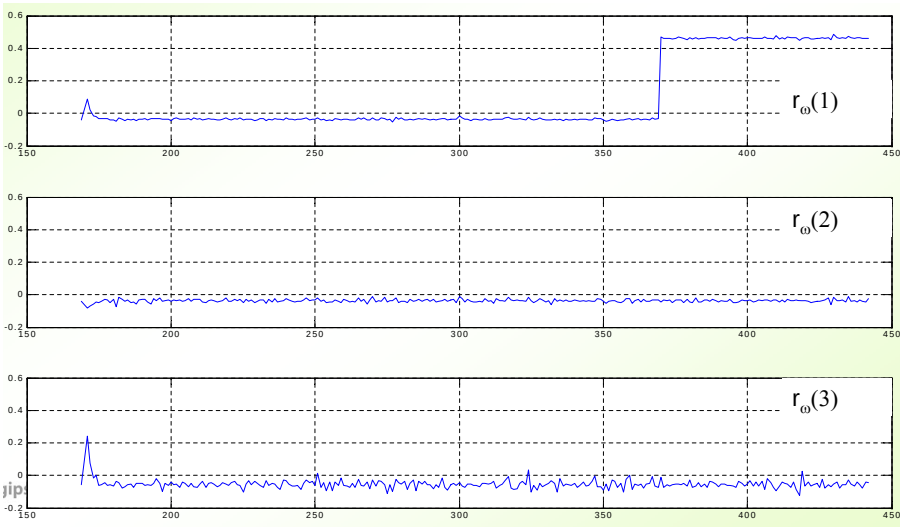


FIG. 8.12 – Générations des résidus pour les gyromètres par *Algo 2* avec des données réelles et un défaut sur le $gyro_X$

La dernière étape, celle où le système n'est plus simulé mais est réel est en cours de test. Les premiers résultats obtenus montrent que l'*Algo 2* fonctionne correctement.

Chapitre 9

Conclusions et perspectives

Dans ce manuscrit, le diagnostic de capteurs appliqué à une centrale d'attitude ainsi que la mise en réseau des systèmes embarqués ont été traités.

Concernant tout d'abord le diagnostic, deux approches ont été proposées pour le diagnostic d'une centrale d'attitude embarquée dans un quadrotor. La première (*Algo 1*), basée sur le principe des bancs d'observateurs généralisés, est assez limitée par le fait qu'elle dépend du modèle mécanique du quadrotor. Avec cette structure, seuls les défauts simples sont envisageables. Envisager des défauts multiples est très difficile. Une fois le défaut détecté et localisé, l'étape d'identification n'est pas réalisable. Un autre inconvénient majeur est la sensibilité des résidus aux perturbations et aux imprécisions du modèle mécanique du quadrotor. Celles-ci influencent la prise de décision en générant de fausses alarmes.

Pour cela, une deuxième structure, nommée *Algo 2*, a été proposée. Contrairement à *Algo 1*, cette structure est insensible aux erreurs de modèle du quadrotor, car il n'est pas pris en compte pour la génération des résidus. Cette structure permet de réaliser les étapes de détection et de localisation. Mais contrairement à *Algo 1*, l'étape d'identification est elle aussi réalisable. Contrairement à l'approche des bancs d'observateurs utilisés par *Algo 1*, le module *Algo 2* utilise uniquement les mesures de la centrale d'attitude pour prendre les décisions. Ceci permet de le rendre *indépendant* du système sur lequel elle est utilisée. Basé uniquement sur la reconstruction des mesures sans passer par l'état du système, l'algorithme est par conséquent indépendant des perturbations pouvant survenir sur le système. En d'autres termes, ce module présenté est générique. On a vu, à travers un exemple, la limitation de la structure par banc d'observateurs. En effet, on peut être amené à avoir des manques à la détection si l'amplitude des défauts est trop petite. Avec des données réelles, grâce à l'étude expérimentale, le module *Algo 2* a été validé.

Une deuxième problématique a été abordée dans ce travail de doctorat. Celle-ci est l'implémentation d'une boucle de commande et de diagnostic à travers un réseau. Dans ce travail, deux approches ont été abordées. La première consiste à mettre au point des techniques pour commander un système à travers le réseau. Dans ce cas, les algorithmes de commande et de diagnostic doivent être modifiés afin de tenir compte des paramètres propres au réseau. Nous avons essayé de répondre aux questions du type : que faire en présence de pertes d'informations ? Pour cela, la méthode utilisée dans ce manuscrit est une méthode pragmatique qui consiste à ne "rien faire" plutôt que de faire des choses qui donneront des résultats faux. Les résultats obtenus avec cette solution ont été par la suite comparés au cas sans réseau. Si le réseau est considéré comme un simple moyen de communication alors il y aura apparition de fausses alarmes surtout dans les transitoires. C'est pourquoi, il est nécessaire de modifier les algorithmes de diagnostic et de les guider par les événements. Lorsque le réseau est considéré comme un composant du système à diagnostiquer lui aussi, les résultats obtenus sont concluants.

Une autre technique consiste à utiliser l'approche par filtre de Kalman en réalisant l'étape de correction de ce dernier uniquement avec les mesures reçues ([63]). Cette technique est utilisée depuis peu au sein de l'équipe. L'idée avec cette approche est de modifier le filtre afin de pouvoir réaliser l'estimation de l'attitude en présence de perte d'informations. Pour ce faire, lorsqu'une mesure est manquante, elle est remplacée par une valeur nulle avec un écart type très grand (permettant de diminuer la confiance dans cette mesure). Cette approche pourra être étendue à la problématique du diagnostic en l'utilisant pour la génération des résidus dans un schéma de bancs d'observateurs.

Dans le cadre du projet Safe-NECS, on a vu que le simulateur TrueTime fournit une bonne base bien que les modèles de réseaux restent à étendre. Des extensions ont été développées pour CAN, Ethernet Commuté et ZigBee par des partenaires du projet et nous les avons testées sur notre application.

D'un point de vue "*Control of network*", nous avons vu deux ordonnancements possibles pour garantir une qualité de contrôle nécessaire au système, l'une basée sur l'approche du *(m-k)*-firm et l'autre basée sur l'approche des priorités dynamiques.

Malheureusement par manque de temps, d'autres méthodes n'ont pas pu être testées comme celle par exemple du changement de la période d'échantillonnage [79], ou bien encore celles basées sur les événements ([40], [41]). Les méthodes présentées dans ce manuscrit sont en cours d'implantation sur le vrai système.

Concernant les perspectives, la première d'entres elles est bien évidemment de tester

toutes les approches développées dans ce manuscrit sur le prototype, une fois que les problèmes liés à la mise en œuvre seront réglés. Ceci permettra de valider les approches en situation réelle.

D'un point de vue diagnostic, il serait intéressant de définir des critères d'évaluation de type critère quadratique basé sur le résidu afin de mettre en œuvre les politiques d'ordonnancement non plus par rapport au besoin de la commande mais plutôt par rapport à celui du diagnostic. On pourrait imaginer un scénario du type diminution de la priorité d'envoi d'une mesure car celle-ci est en défaut, ou bien encore augmenter la priorité d'envoi d'une mesure car à l'instant d'avant, cette dernière a été perdue...

Concernant les perspectives d'un point de vue commande du réseau, il serait intéressant d'étendre cette étude aux réseaux sans fil et d'essayer de mettre en place des techniques d'ordonnancement pour garantir les besoins du système. Avec Ethernet, il serait aussi intéressant d'introduire la notion de priorité en jouant peut-être sur la notion de temps d'attente. Celui-ci pourrait être augmenté pour les tâches non prioritaires afin de faciliter l'envoi des paquets plus prioritaires et de réduire ainsi les collisions.

Comme on le voit, le domaine de la commande et du diagnostic des systèmes en réseau est encore un domaine ouvert dans lequel beaucoup de progrès peuvent être réalisés dans un futur proche, grâce à la collaboration fructueuse de spécialistes du réseau, de l'informatique industrielle et de l'automatique.

Annexe A

Modèle linéarisé du Quadrotor

Dans cette annexe, le modèle linéarisé du quadrotor est présenté. A partir de l'approche Newton-Euler, les équation dynamiques du quadrotor sont :

$$(\dot{\phi}, \dot{\theta}, \dot{\psi})^T = M\omega \tag{A.0.1}$$

$$I_f \dot{\omega} = -\omega \times I_f \omega + \tau_a + G_a \tag{A.0.2}$$

avec la matrice M définie par :

$$M = \begin{pmatrix} 1 & \tan\theta \sin\phi & \tan\theta \cos\phi \\ 0 & \cos\phi & -\sin\phi \\ 0 & \frac{\sin\phi}{\cos\theta} & \frac{\cos\phi}{\cos\theta} \end{pmatrix} \begin{pmatrix} \omega_x \\ \omega_y \\ \omega_z \end{pmatrix} \tag{A.0.3}$$

A.1 Linéarisation et discrétisation du système

L'objectif est de définir une loi de commande qui stabilise le système défini par (A.0.1)-(A.0.2). Pour les conditions de stabilisation $\phi = 0^\circ; \theta = 0^\circ; \psi = 0^\circ$, on peut faire l'approximation suivante :

$$(\dot{\phi}, \dot{\theta}, \dot{\psi})^T = (\omega_1, \omega_2, \omega_3)^T \tag{A.1.1}$$

Le modèle peut, par conséquent, être exprimé en termes de roulis, tangage et lacet ((A.0.1), (A.0.2) et (A.1.1)) :

$$\begin{aligned} \ddot{\phi} &= \dot{\theta}\dot{\psi}\left(\frac{I_{f_y} - I_{f_z}}{I_{f_x}}\right) + \frac{1}{I_{f_x}}\tau_a^\phi \\ \ddot{\theta} &= \dot{\phi}\dot{\psi}\left(\frac{I_{f_z} - I_{f_x}}{I_{f_y}}\right) + \frac{1}{I_{f_y}}\tau_a^\theta \\ \ddot{\psi} &= \dot{\phi}\dot{\theta}\left(\frac{I_{f_x} - I_{f_y}}{I_{f_z}}\right) + \frac{1}{I_{f_z}}\tau_a^\psi \end{aligned} \tag{A.1.2}$$

Les couples gyroscopiques G_a ne sont pas considérés ici pour la mise en œuvre de la loi de commande.

Les variables d'état peuvent être définies par :

$$x^T = (x_1\ x_2\ x_3\ x_4\ x_5\ x_6)^T = (\phi\ \dot{\phi}\ \theta\ \dot{\theta}\ \psi\ \dot{\psi})^T \tag{A.1.3}$$

La linéarisation conduit à

$$\dot{x} = Ax + Bu \tag{A.1.4}$$

où

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \quad B = \begin{pmatrix} 0 & 0 & 0 \\ \frac{1}{I_{fx}} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & \frac{1}{I_{fy}} & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \frac{1}{I_{fz}} \end{pmatrix} \tag{A.1.5}$$

On peut constater que le modèle linéarisé est en fait un système constitué par trois double intégrateurs.

La discrétisation (étape nécessaire avant l'implantation d'un réseau voir annexe B) de (A.1.4), avec $h = 10\ ms$ la période d'échantillonnage nous donne :

$$x(h(k+1)) = \Phi(kh)x(hk) + \Gamma(kh)u(hk) \tag{A.1.6}$$

où

$$\Phi = e^{Ah} \qquad \Gamma = \int_{kh}^{kh+h} e^{As}Bds \tag{A.1.7}$$

Les matrices de (A.1.7) sont calculées avec

$$\begin{pmatrix} \Phi & \Gamma \\ 0 & I \end{pmatrix} = exp\{\begin{pmatrix} A & B \\ 0 & 0 \end{pmatrix} h\} \tag{A.1.8}$$

A.2 Loi de commande linéaire quadratique

Le système linéarisé est utilisé uniquement pour les problèmes de "*Control of Network*", donc certaines hypothèses sont faites afin de faciliter les calculs :

1. l'état du système est disponible directement. en pratique, il y a des capteurs et un observateur ;
2. le signal de commande est constant entre deux périodes d'échantillonnage.

La stabilisation du système permet d'amener et surtout de maintenir le quadrotor dans les condition désirées à partir de conditions initiales quelconques. Les vitesses angulaires sont supposées nulles dans l'attitude désirée $\phi = 0^{\circ}; \theta = 0^{\circ}; \psi = 0^{\circ}$:

$$x \to 0, \quad t \to \infty \tag{A.2.1}$$

Ceci permet de définir la loi de commande suivante

$$u(kh) = -Lx(kh) \tag{A.2.2}$$

Cette loi doit satisfaire les contraintes dynamiques du système (A.1.6) en minimisant la fonction coût suivante :

$$\begin{aligned} J = & \sum_{k=0}^{N-1} [x^T(kh) Q_d x(kh) + u^T(kh) R_d u(kh)] \\ & + x^T(Nh) Q_0 x(Nh) \end{aligned} \tag{A.2.3}$$

avec les matrices de (A.2.3) définies par :

$$Q_d = \int_{kh}^{kh+h} \Phi^T(s) Q \Phi(s) ds \qquad R_d = \int_{kh}^{kh+h} (\Gamma^T(s) Q \Gamma(s) + R) ds \tag{A.2.4}$$

Les matrices Q, R et Q_0 sont définies positives [2]. De plus, les matrices Q et R sont choisies pour obtenir un transitoire acceptable (pas de saturation des actionneurs). Avec $h = 0.01\ s$, la matrice L est donnée par

$$L = \begin{pmatrix} 0.0347 & 0.0280 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.0347 & 0.0280 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.0347 & 0.0280 \end{pmatrix} \tag{A.2.5}$$

A.3 Stabilité du système avec pertes de paquets

Ici, on souhaite étudier le comportement du système en présence de pertes de paquets. On souhaite aussi déterminer la perte maximale de données, avant la perte de stabilité du système. Pour cela, considérons le schéma de la figure A.1. Cette figure permet de simuler sous Matlab l'équation définie par (A.1.6).

On suppose que la loi de commande est émise toutes les kh périodes. De plus, on suppose, afin de calculer la perte maximale tolérable qu'à partir de $t = ih$, $i \in \{k + 1, ..., k + l\}$, les paquets contenant le signal de commande sont perdus par le réseau. Ceci nous permet d'étudier la stabilité du système en considérant le système d'équations

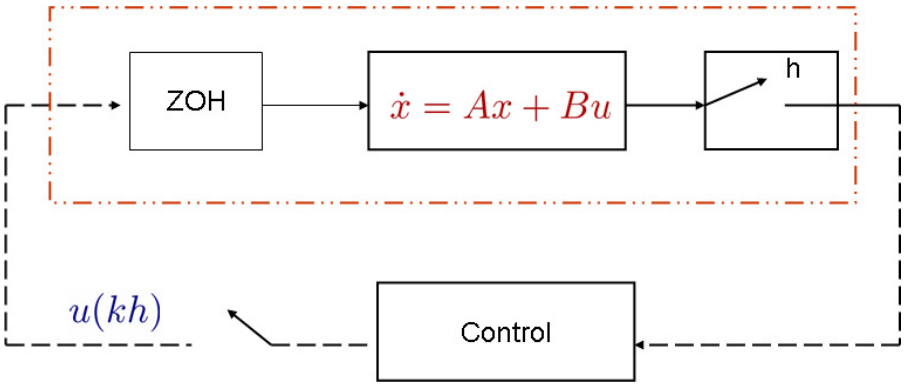


FIG. A.1 – Boucle fermée du système

suivant : [13] :

$$\begin{aligned} x((k+1)h) &= \Phi x(hk) + \Gamma u(hk) \\ x((k+2)h) &= \Phi^2 x(hk) + (\Phi + I)\Gamma u(hk) \\ &\vdots \\ x((k+l)h) &= \Phi^l x(hk) + (\sum_{j=0}^{l-1} \Phi^j \Gamma) u(hk) \end{aligned} \tag{A.3.1}$$

Avec $u(kh) = -Lx(kh)$ l'équation (A.3.1) devient

$$\begin{aligned} x((k+l)h) &= \Phi^l x(hk) - (\sum_{j=0}^{l-1} \Phi^j \Gamma) L x(kh) \\ &= (\Phi^l - \sum_{j=0}^{l-1} \Phi^j \Gamma L) x(kh) = \bar{\Phi} x(kh) \end{aligned} \tag{A.3.2}$$

L'interprétation physique de (A.3.2) est que le système est en boucle ouverte avec l'entrée du système égale à la loi de commande appliquée à l'instant $t = kh$. Par conséquent, l'état $x((k+l)h)$ reste stable si et seulement si toutes les valeurs propres du système défini avec la matrice $\bar{\Phi}$ restent dans le cercle unité.

Afin de trouver la valeur limite l de (A.3.2), une simulation est réalisée. Comme le système linéarisé est constitué de trois double intégrateurs, trois paires de valeurs propres existent. Avec le gain optimal L de (A.2.5), l'évolution d'une paire de valeurs propres en fonction de l est montrée sur la figure A.2. Comme on peut voir, le système peut supporter une remise à jour de la loi de commande toutes les $lh = 55$ $h = 0.55$ s. Grâce au critère de Jury appliqué sur un sous-système du modèle, le résultat de simulation $l = 55$ a été confirmé. En d'autres termes, le système reste stable et est robuste à des pertes de paquets inférieures à 55 pertes consécutives.

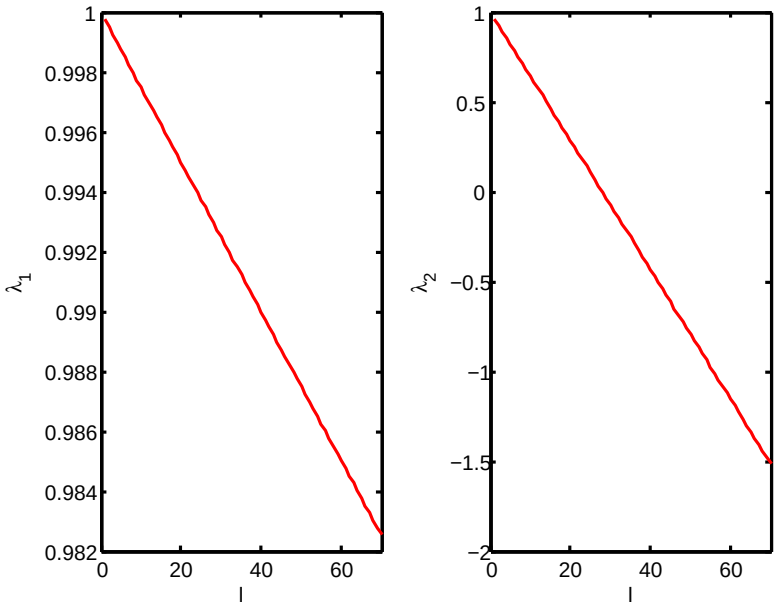


FIG. A.2 – Stabilité du système avec $l \in [0, 70]$

Annexe B

Networked Control System : méthodologie

Dans cette annexe, sont expliqués les principes de base pour réaliser un système commandé en réseau. Toutes les étapes de conception sont réalisées sous l'environnement Matlab/Simulink.

B.0.1 Système continu

Dans un premier temps, il est nécessaire d'élaborer un système continu avec sa commande continue et son diagnostic en temps continu. Une fois cette étape validée, on peut passer à la discrétisation.

B.0.2 Système discret

Pour cette deuxième étape, l'objectif est de discrétiser les algorithmes du contrôleur et ceux du diagnostic. Plusieurs méthodes existent et dans [15], nous avons montré que dans le cas des algorithmes de diagnostic, pour certains cas la méthode dite du bloqueur d'ordre zéro n'est pas forcement adaptée. Lorsque cette étape est réalisée, c'est-à-dire lorsqu'à chaque période d'échantillonnage, les résultats sont identiques à ceux obtenus pour le système à temps continu, nous pouvons regarder l'influence des retards sur le système en boucle fermée.

B.0.3 Introduction des retards

Une fois le système discret réalisé, il est intéressant de regarder l'influence des retards induits par le réseau sur le système en boucle fermée. Pour cela, des retards peuvent être introduits à la place du réseau. Ces retards doivent correspondre au temps de latence maximal induit par le réseau. Une fois cette étape terminée, on peut passer à l'implantation du réseau lui-même.

B.0.4 Implantation du réseau

Maintenant, on peut introduire le réseau dédié dans la boucle fermée à l'aide de la boite à outil Truetime. Les résultats obtenus sans perte de paquets doivent correspondre à ceux obtenus avec l'introduction des retards.

Annexe C

Modèle OSI

Le modèle OSI (de l'anglais *Open Systems Interconnection*, " Interconnexion de systèmes ouverts") d'interconnexion en réseau des systèmes ouverts est un modèle de communications entre ordinateurs proposé par l'ISO (Organisation Internationale de Normalisation). Il décrit les fonctionnalités nécessaires à la communication et l'organisation de ces fonctions [97].

Ce modèle est illustré sur la figure C.1

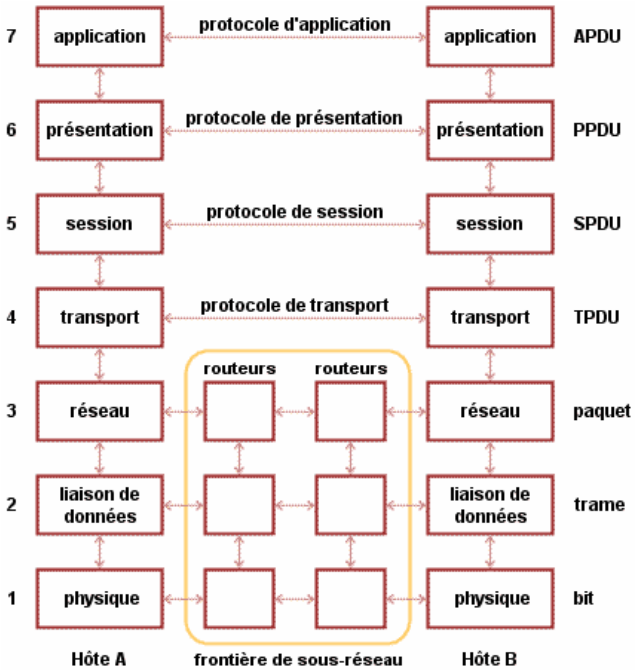


Fig. C.1 – Modèle de référence OSI d'après [83]

Le ***Modèle de Référence OSI*** pour *Open System Interconnection* traite de la

connexion entre systèmes ouverts à la communication avec d'autres systèmes. Ce modèle comporte sept couches. D'après [83], les principes ayant conduit à l'élaboration des sept couches sont les suivants :

1. une couche doit être créée lorsqu'un nouveau niveau d'abstraction est nécessaire ;
2. chaque couche exerce des fonctions bien définies ;
3. les fonctions de chaque couche doivent être choisies en visant la définition de protocoles normalisés ;
4. le choix des frontières entre couches doit minimiser le flux d'information aux interfaces ;
5. le nombre de couches doit être suffisamment grand pour éviter la cohabitation dans une même couche de fonctions très différentes et suffisamment petit pour éviter que l'architecture ne devienne difficile à maîtriser.

La description suivante est un résumé des caractéristiques de chaque couche. La description originelle est donnée dans [83]. Chaque couche de la figure C.1 est décrite en partant de la couche basse.

La couche physique s'occupe de la transmission des bits de façon brute sur un canal de communication. Sa conception doit être telle que l'on doit être sûr que, lorsqu'un côté envoie un 1, on reçoit un 1 de l'autre côté et non pas un 0.

La couche liaison de données La tâche principale de la couche liaison de données est de prendre un moyen de communication "brut" et de le transformer en une liaison qui paraît exempte d'erreurs de transmission à la couche réseau. Elle l'accomplit en fractionnant les données d'entrée de l'émetteur en trame de données, en transmettant les trames en séquence et en gérant les trames d'acquittement renvoyées par le récepteur.

La couche réseau permet de gérer le sous-réseau. La façon dont les paquets sont acheminés de la source au destinataire constitue un élément clé de sa conception. Les routes peuvent être fondées sur des tables statiques "câblées" dans le réseau et rarement changées. Elles peuvent également être déterminées au début de chaque conversation, par exemple lorsqu'on se connecte à un ordinateur depuis un terminal. Enfin, elle peuvent être très dynamiques, recalculées pour chaque paquet de manière à prendre en compte la charge instantanée du réseau utilisé. Si trop de paquets se trouvent simultanément dans le sous-réseau, il va se créer des engorgements. Le contrôle d'une telle congestion est aussi du domaine de la couche réseau.

La couche transport La fonction de base de la couche transport est d'accepter des données de la couche session, de les découper, le cas échéant, en plus petites unités, de les passer à la couche réseau et de s'assurer que tous les morceaux arrivent correctement de l'autre côté.

La couche session permet à des utilisateurs travaillant sur les différentes machines d'établir des sessions entre eux. Une session permet le transport des données, comme la couche transport, mais elle offre également des services évolués utiles à certaines applications. Un des service est la synchronisation.

La couche présentation remplit des fonctions suffisamment courantes et génériques pour n'être pas laissées à la charge de chaque utilisateur. Plus précisément, à la différence des autres couches, qui sont concernées seulement par la transmission fiable des bits d'un point à un autre, la couche présentation s'intéresse à la syntaxe et à la sémantique de l'information transmise.

La couche application La couche application comporte de nombreux protocoles fréquemment utilisés. Par exemple, il y a de par le monde des centaines de terminaux incompatibles. On peut résoudre ce problème en définissant un terminal de réseau virtuel.

Annexe D

Protocole IPv4

L'Internet Protocol version 4 ou IPv4 est la première version d'IP à avoir été largement déployée, et forme encore la base (en 2009) de l'Internet [101].

IPv4 utilise une adresse IP sur 32 bits, ce qui est un facteur limitant à l'expansion d'Internet puisque seulement 4 228 250 626 adresses sont possibles. Cette limitation conduit à la transition d'IPv4 vers IPv6, actuellement en cours de déploiement, qui devrait progressivement le remplacer. Cette limitation est pour l'instant contournée grâce à l'utilisation de techniques de translation d'adresses NAT ainsi que par l'adoption du système CIDR.

D.1 Représentation d'une adresse IPv4

Le plus souvent, une adresse IP est représentée sous la forme de quatre nombres décimaux séparés par des points comme par exemple 193.43.55.67. Chacun des nombres représente un octet. Comme un octet est composé de 8 bits, cela fait 32 bits.

En bits

<table>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>14</td><td>15</td><td>16</td><td>17</td><td>18</td><td>19</td><td>20</td><td>21</td><td>22</td><td>23</td><td>24</td><td>25</td><td>26</td><td>27</td><td>28</td><td>29</td><td>30</td><td>31</td></tr>
<tr><td colspan="4">Version d'IP</td><td colspan="4">Longueur de l'en-tête (en mots de 32 bits)</td><td colspan="8">Type de service</td><td colspan="16">Longueur totale en octets</td></tr>
<tr><td colspan="16">Identification (pour les fragments)</td><td colspan="3">Flags (pour les fragments)</td><td colspan="13">Fragment offset</td></tr>
<tr><td colspan="8">Durée de vie (TTL Time To Live)</td><td colspan="8">Protocole</td><td colspan="16">Somme de contrôle de l'en-tête</td></tr>
<tr><td colspan="32">Adresse source</td></tr>
<tr><td colspan="32">Adresse destination</td></tr>
<tr><td colspan="32">Option(s) + bourrage</td></tr>
</table>

Fig. D.1 – En-tête IPv4

Une trame a une taille maximale, appelée Maximum Transmission Unit ou MTU.

Lorsque la longueur du paquet (datagramme) est supérieure, l'information sera fragmentée. La taille maximum supportée par IPv4 (car codée sur 16 bits) est de 64 ko mais les réseaux ne prennent pas en charge de trames de telles longueurs, typiquement, on a MTU de l'ordre de 1 ko (Arpanet), 1,5 ko (Ethernet) etc.

Pour reconstituer l'information, on utilise les champs suivants dans l'en-tête IPv4 [101] (voir figure D.1) :

- Version (4 bits) : version d'IP utilisée. Ici, 4.
- Longueur de l'en-tête (4 bits) : nombre de mots de 32 bits, soit 4 octets (ou nombre de lignes du schéma). La valeur est comprise entre 5 et 15, car il y a 20 octets minimum et on ne peut dépasser 40 octets d'option (soit en tout, 60 octets).
- Type de service (8 bits) : rarement utilisé. Ce champ permet de distinguer différentes qualité de service différenciant la manière dont les paquets sont traités. Composé de 3 bits de priorité (donc 8 niveaux) et trois indicateurs permettant de différencier le débit, le délai ou la fiabilité.
- Longueur totale en octets (16 bits) : nombre total d'octets du datagramme, en-tête IP comprise. Donc, la valeur maximale est $(2^{16}) - 1$ octets.
- Identification (16 bits) : numéro permettant d'identifier les fragments d'un même paquet.
- Flag (3 bits) :
  1. (Premier bit) actuellement inutilisé.
  2. (Deuxième bit) DF (Don't Fragment) : lorsque ce bit est positionné à 1, il indique que le paquet ne peut pas être fragmenté. Si le routeur ne peut acheminer ce paquet (taille du paquet supérieure à la MTU), il est alors rejeté.
  3. (Troisième bit) MF (More Fragments) : quand ce bit est positionné à 1, on sait que ce paquet est un fragment de données et que d'autres doivent suivre. Quand il est à 0, soit le fragment est le dernier, soit le paquet n'a pas été fragmenté.
- Fragment offset (13 bits) : position du fragment par rapport au paquet de départ, en nombre de mots de 8 octets.
- Durée de vie ou TTL Time To Live (8 bits) : initialisé par l'émetteur, ce champ est décrémenté d'une unité généralement à chaque saut de routeur. Quand TTL = 0, le paquet est abandonné et un message ICMP est envoyé à l'émetteur pour information.
- Protocole (8 bits) : numéro du protocole au-dessus de la couche réseau : TCP = 6, UDP = 17, ICMP = 1.
- Somme de contrôle de l'en-tête ou Checksum ou encore CRC pour Contrôle de Redondance Cyclique (16 bits) : vérification de l'intégrité de l'en-tête seulement. Si le CRC est invalide, le paquet est abandonné sans message d'erreur.
- Adresse source (32 bits) : adresse IP de l'émetteur sur 4 octets ou 32 bits.

- Adresse destination (32 bits) : adresse IP du récepteur sur 4 octets ou 32 bits.
- Options (0 à 40 octets ou 0 à 320 bits par mots de 32 bits ou 4 octets) : facultatif.
- Bourrage : de taille variable comprise entre 0 et 7 bits. Il permet de combler le champ option afin d'obtenir un en-tête IP multiple de 32 bits. La valeur des bits de bourrage est 0.

Annexe E

Protocole IPv6

IPv6 (*Internet Protocol version 6*) est le successeur du protocole IPv4. En effet, le protocole IPv4 ne permet d'utiliser qu'un peu plus de quatre milliards d'adresses différentes pour connecter les ordinateurs et les autres appareils reliés au réseau. Du temps des débuts d'Internet, quand les ordinateurs étaient rares, cela paraissait plus que suffisant. Il était pratiquement inimaginable qu'il y aurait un jour suffisamment de machines sur un unique réseau pour que l'on commence à manquer d'adresses disponibles [102].

Une grande partie des quatre milliards d'adresses IP théoriquement disponibles ne sont pas utilisables, soit parce qu'elles sont destinées à des usages particuliers (par exemple, le multicast), soit parce qu'elles appartiennent déjà à des sous-réseaux importants. En effet, d'immenses plages de 16,8 millions d'adresses, les réseaux dits de classe A, ont été attribuées aux premières grandes organisations connectées à Internet, qui les ont conservées jusqu'à aujourd'hui sans parvenir à les épuiser.

C'est pourquoi il y a aujourd'hui, principalement en Asie, une pénurie d'adresses que l'on doit compenser par des mécanismes comme la Traduction d'adresse et de port réseau (NAPT) et l'attribution dynamique d'adresses, et en assouplissant le découpage en classes des adresses (CIDR).

Au vu de l'importance et de la croissance d'Internet, cette situation pose de plus en plus de problèmes. Il est de plus prévisible que la demande d'adresses Internet va augmenter dans les années à venir, même dans les régions du monde peu connectées jusqu'ici, suite à des innovations comme les téléphones mobiles (et bientôt, sans doute, les automobiles et divers appareils) connectés à Internet.

C'est principalement en raison de cette pénurie, mais également pour résoudre quelques-uns des problèmes révélés par l'utilisation à vaste échelle d'IPv4, qu'a commencé en 1995 la transition vers IPv6. Une adresse IPv6 est un mot de 128 bits. La taille

d'une adresse IPv6 est le quadruple de celle d'une adresse IPv4 [31].

Pour la représentation d'une adresse $IPv6$, on dispose ainsi d'environ $3,4 \times 10^{38}$ adresses, soit 340 282 366 920 938 463 463 374 607 431 768 211 456. On abandonne la notation décimale pointée employée pour les adresses IPv4 (avec 4 groupes de 1 octet séparés par des points, par exemple 172.31.128.1) au profit d'une écriture hexadécimale, où les 8 groupes de 2 octets (= 16 bits) sont séparés par un signe deux-points :

1fff :0000 :0a88 :85a3 :0000 :0000 :ac1f :8001 La notation canonique complète ci-dessus comprend exactement 39 caractères.

Les 64 premiers bits de l'adresse IPv6 (préfixe) servent généralement à l'adresse de sous-réseau, tandis que les 64 bits suivants identifient l'hôte à l'intérieur du sous-réseau : ce découpage joue un rôle un peu similaire aux masques de sous-réseau d'IPv4.

Différentes sortes d'adresses IPv6 jouent des rôles particuliers. Les propriétés de ces blocs sont fixées par un plan d'adressage qui définit des préfixes.

Annexe F

Principe de la SVD

Dans cette annexe, nous allons montrer le principe de la SVD utilisée dans le cadre de l'observateur non linéaire. Nous partons de l'égalité suivante :

$$\bar{H} = USV^T \tag{F.0.1}$$

Or :

$$\frac{1}{2}\left\|\bar{H}\hat{q}\right\|_2 = \frac{1}{2}\left(\hat{q}^T V S^T U^T \cdot USV^T\hat{q}\right) \tag{F.0.2}$$

La propriété d'orthonormalité donne :

$$U^T U = I \in R^{4n\times 4n} \tag{F.0.3}$$

d'où

$$\frac{1}{2}\left\|\bar{H}\hat{q}\right\|_2 = \frac{1}{2}\left\|SV^T\hat{q}\right\|_2 \tag{F.0.4}$$

soit

$$Y = V^T\hat{q} = [y_1 \quad y_2 \quad y_3 \quad y_4]^T \tag{F.0.5}$$

Ainsi le problème devient :

$$\arg\min_Y \left\{ f(Y) = \frac{1}{2}\left\|SY\right\|_2^2 \right\} \text{avec } Y \in D = \left\{Y \text{ tel que } Y^T Y = 1\right\} \tag{F.0.6}$$

Il faut remarquer que la matrice S est formée d'une diagonale et de zéros telle que :

$$S = \begin{pmatrix} \sigma_1 & 0 & 0 & 0 \\ 0 & \sigma_2 & 0 & 0 \\ 0 & 0 & \sigma_3 & 0 \\ 0 & 0 & 0 & \sigma_4 \\ 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{F.0.7}$$

avec $\sigma_1 \geq \sigma_2 \geq \sigma_3 \geq \sigma_4 \geq 0$.
Une propriété très utile d'une décomposition SVD est qu'elle nous donne l'information à propos du rang de la matrice. Ainsi pour une matrice $A \in \mathbb{R}^{m \times n}$ quelconque :

$$\textbf{rang}(A) = r \text{ tel que } \sigma_r > 0 \text{ et } \sigma_{r+i} = 0 \text{ avec } i \in \{1, .., p\}, \text{ où } p = \min(m, n - r).$$

Puisque nous supposons qu'au moins deux paires de vecteurs sont non colinéaires, nous avons :

$$\textbf{rang}(\bar{H}) = 3 \Rightarrow \sigma_3 > 0 \text{ et } \sigma_4 = 0$$

De l'équation (F.0.6), nous devons résoudre :

$$\begin{aligned} f(Y) &= \frac{1}{2}\left[(\sigma_1 y_1)^2 + (\sigma_2 y_2)^2 + (\sigma_3 y_3)^2 + (\sigma_4 y_4)^2\right] = 0 \\ &\text{sous la contrainte :} \quad Y^T Y = 1 \end{aligned} \tag{F.0.8}$$

En conséquence, la solution à l'équation (F.0.8) est :

$$Y = [0\ 0\ 0\ 1]^T \tag{F.0.9}$$

De l'équation (F.0.5), il vient :

$$\begin{aligned} VY &= \hat{q} \\ [V_1\ \ V_2\ \ V_3\ \ V_4]Y &= \hat{q} \\ V_4 &= \hat{q} \end{aligned} \tag{F.0.10}$$

puisque la matrice V est par définition orthogonale $\|V_4\|_2 = 1 \Leftrightarrow \|\hat{q}\|_2 = 1$.

■

Remarque 20. Notons que l'analyse a été faite dans un contexte sans bruit, i.e. les mesures des capteurs sont parfaites. En conséquence, il est possible d'affirmer que le rang de la matrice $\bar{H}$ est de trois. En pratique, à cause du bruit présent dans les mesures, la dernière valeur singulière est différente de zéro mais reste proche de zéro. En fait, la dernière valeur singulière nous donne une mesure de l'erreur. Des équations (F.0.8) et (F.0.9), l'erreur au sens des moindres carrés est donnée par :

$$e = \left\|\bar{H}\hat{q}\right\|_2^2 = \|SY\|_2^2 = \sum_{i=1}^{4}(\sigma_i Y_i)^2 = \sigma_4^2 \tag{F.0.11}$$

Annexe G

Publications

Chapitres de livre :

1. C. Aubrun, C. Berbra, S. Gentil, S. Lesecq, D. Sauter, "***Chapitre 6 : Fault detection and isolation, fault tolerant control*** *dans* ***Co-design approaches for dependable networked control systems***", Wiley, à paraître.
2. C. Berbra, S. Gentil, S. Lesecq, D. Simon, "***Chapitre 7 : Control and Diagnosis for an Unmanned Aerial Vehicle*** *dans* ***Co-design approaches for dependable networked control systems***", Wiley, à paraître.

Conférences :

1. JF. Guerrero-Castellanos, C. Berbra, S. Gentil, S. Lesecq, "***Quadrotor Attitude Control Through a Network with (m-k)-Firm Policy***", Proceedings of 10th European Control Conference (ECC'09), Budapest, Hongrie, 2009.
2. C. Berbra, D. Simon, S. Gentil, S. Lesecq, "***Hardware in the loop networked control and diagnosis of a quadrotor drone***", Proceedings of IFAC SAFEPROCESS 2009, Barcelone, Espagne, 2009.
3. S. Lesecq, S. Gentil, C. Berbra, "***Condition monitoring based on filter bank in the presence of data loss***", Proceedings of the 6th International Conference on Condition Monitoring and Machinery Failure Prevention Technologies, Dublin, Irlande, 2009.
4. C. Berbra, S. Gentil, S. Lesecq, "***Hybrid priority scheme for networked control quadrotor***", Proceedings of the 17th Mediterranean Conference on Control and Automation, MED'09, Thessalonique, Grèce, 2009.
5. H.V. Nguyen, C.Berbra, S. Lesecq, S. Gentil, A. Barraud, C. Godin, "***Diagnosis of an Inertial Measurement Unit Based on Set Membership Estimation***", Proceeding of the 17th Mediterranean Conference on Control and Automation, MED'09, Thessalonique, Grèce, 2009.

6. C. Berbra, S. Lesecq, S. Gentil, J.M. Thiriet, "***Co-design of a safe networked control quadrotor***", Proceedings of Proceedings of the 17th IFAC World Congress,Seoul, Corée, 2008.
7. I. Diouri, C. Berbra, J.P. Georges, S. Gentil, E. Rondeau, "***Evaluation of a switched Ethernet network for the control of a quadrotor***", Proceedings of 16th Mediterranean Conference on Control and Automation, MED'08, Ajaccio, France, 2008.
8. C. Berbra, S. Lesecq, J.J. Martinez Molina, "***A Multi-observer Switching Strategy for Fault-Tolerant Control of a Quadrotor Helicopter***", Proceedings of 16th Mediterranean Conference on Control and Automation, MED'08, Ajaccio, France, 2008.
9. C. Berbra, Z.H. Khan, S. Gentil, S. Lesecq, J.M. Thiriet, "***A diagnosis strategy for FDI in wireless networked control system***", Proceedings of International Conference on Fieldbuses and Networks in Industrial and Embedded Systems, Toulouse, France, 2007.

Workshops :

1. C. Berbra, S. Gentil, S. Lesecq, "***Identification of Multiple faults in an Inertial Measurement Unit***", Proceedings of 7th Workshop on Avanced Control and Diagnosis (ACD'2009), Zielona Gora, Pologne, 2009.
2. JF. Guerrero-Castellanos, C. Berbra, S. Gentil, S. Lesecq, "***Attitude control of a quadrotor tolerant to network faults using (m-k)-firm approach***", Proceedings of 6th Workshop on Avanced Control and Diagnosis (ACD'2008), Coventry, Angleterre, 2008
3. C. Berbra, S. Lesecq, S. Gentil, J.M. Thiriet, "***Diagnosis of a safe network control quadrotor***", Proceedings of 5th Workshop on Avanced Control and Diagnosis (ACD'2007), Grenoble, France, 2007.
4. C. Berbra, S. Lesecq, S. Gentil, J.M. Thiriet, "***Co-design for a safe networked control DC motor***",Proceedings of 3rd International Workshop on Networked Control Systems : Tolerant to Faults, Nancy, France, 2007.

Bibliographie

[1] B.D.O. Anderson and J.B. Moore. *Optimal Filtering.* Prentice-Hall, New Jersey, 1979.

[2] J. Astrom and B. Wittenmark. *Computer-Controlled Systems : Theory and Design.* Prentice-Hall, New Jersey, 1990.

[3] M. Basseville. Detecting changes in signals and systems-a survey. *Automatica*, 24 :309–326, 1988.

[4] M. Basseville and I. Nikiforov. *Detection of abrupt changes- Theory and applications.* Prentice Hall Information and System Sciences Series, 1993.

[5] S. Beeby, G. Ensell, M. Kraft, and N. White. *MEMS Mechanical Sensors.* Artech House, Inc, 2004.

[6] C. Berbra, S. Gentil , and Lesecq S. Hybrid priority scheme for networked control quadrotor. In *17th Mediterranean Conference on Control & Automation*, Thessaloniki, Grèce, 2009.

[7] C. Berbra, D. Simon, S. Gentil, and S. Lesecq. Hardware in the loop networked control and diagnosis of a quadrotor drone. In *the 7th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes*, Barcelone, Espagne, 2009.

[8] UC Berkeley EECS Dept. The ptolemy project (http ://ptolemy.eecs.berkeley.edu/).

[9] N. Le Bihan and J. Mars. Singular value decomposition of quaternion matrices : a new tool for vector-sensor signal processing. *Signal Processing*, 84 :1177–1199, 2004.

[10] M. Blanke, M. Kinnaert, J. Lunze, and M. Staroswiecki. *Diagnosis and fault-tolerant control.* Springer-Verlag, 2003.

[11] BOSCH. *CAN Specification.* http ://www.semiconductors.bosch.de/pdf/can2spec.pdf.

[12] B. Brahimi. *Proposition d'une approche intégrée basée sur les réseaux de Petri de Haut Niveau pour simuler et évaluer les systèmes contrôlés en réseau.* PhD thesis, Université Henri Poincaré, Nancy, 2007.

[13] B. Brahimi, E. Rondeau, and C. Aubrun. Integrated approach based on High Level Petri Nets for Evaluating Networked Control Systems. In *16th Mediterranean Conference on Control and Automation*, Ajaccio, France, 2008.

[14] C.Berbra, S. Lesecq, and J.J. Martinez. A multi-observer switching strategy for fault-tolerant control of a quadrotor helicopter. In *Proceeding of 16th Mediterranean Conference on Control and Automation*, Ajaccio, France, 2008.

[15] C.Berbra, S.Gentil, S. Lesecq, and J.M. Thiriet. Co-design for a safe networked control dc motor. In *3rd International Workshop on Networked Control Systems Tolerant to Faults*, Nancy, France, 2007.

[16] C.Berbra, S.Gentil, S. Lesecq, and J.M. Thiriet. Co-design of a safe network control quadrotor. In *Proceedings of 17th IFAC World Congress'08*, Seoul,Korea, 2008.

[17] A. Cervin, D. Henriksson, B. Lincoln, J. Eker, and K.E. Årzén. How does control timing affect performance ? Analysis and simulation of timing using Jitterbug and TrueTime. *IEEE Control Systems Magazine*, 23(3) :16–30, June 2003.

[18] A. Cervin, M. Ohlin, and D. Henriksson. Simulation of Networked Control Systems Using TrueTime. *3rd International Workshop on Networked Control Systems Tolerant to Faults*, June 2007. Invited talk.

[19] J. Chen and R.J. Patton. *Robust Model Based Fault Diagnosis for Dynamic Systems.* Kluwer Academic Publishers, 1999.

[20] J. C.K. Chou. Quaternion kinematic and dynamic differential equations. *IEEE Transactions on robotics and automation*, 8 :53–64, 1992.

[21] J.C.K. Chou and M. Kamel. Quaternion approach to solve the kinematic equation of rotation, $a_a a_x = a_z a_b$ of a sensor-mounted robotic manipulator. In *IEEE International Conference on Robotics and Automation*, 1988.

[22] CIAME. *Réseaux de terrain : description et critères de choix.* Hermes, 1998.

[23] C. Combastel, S. Gentil, and J.P. Rognon. A symbolic reasoning approach to fault detection and isolation applied to electrical machines. In *IEEE-CCA'98, Conférence on control Applications*, Trieste, Italie, 1998.

[24] C. Combastel, S. Gentil, and J.P. Rognon. Sensitivity of diagnotic residuals to modelling errors. In *IEEE-CCA'99, Conference on Control Applications*, Giron, Espagne, 1999.

[25] I. Diouri, C. Berbra, J.P. Grorges, S. Gentil, and E. Rondeau. Evaluation of a switched ethernet network for the control of a quadrotor. In *16th Mediterranean Conference on Control and Automation*, Ajaccio, France, 2008.

[26] I. Diouri, J.P. Georges, and E. Rondeau. Accommodation of delays for NCS using classification of service. In *IEEE International Conference on Networking, Sensing and Control*, 2007.

[27] F. Felicioni, N. Jia, Y.Q. Song, and F. Simonot-Lion. Impact of a (m-k)-firm data dropouts policy on the quality of control. In *6th IEEE International Workshop on Factory Communication Systems*, 2006.

[28] F.L.Lian, J.R.Moyne, and D.M.Tilbury. Performance Evaluation of Control Networks : Ethernet, Controlnet, and DeviceNet. *IEEE Control Systems Magazine*, 21 :66–83, 2001.

[29] T.I. Fossen. *Guidance and control of ocean vehicles*. New York, 1994.

[30] P.M. Frank and S.X. Ding. Survey of robust residual generation and evaluation methods in observer-based faut detection systems. *Journal of Process Control*, 6 :403–424, 1997.

[31] G6. *IPv6 Théorie et Pratique*. http ://livre.g6.asso.fr/index.php/IPv6.

[32] M. Ben Gaid, D. Simon, and O. Sename. A convex Optimization Approach to Feedback Scheduling. In *16th Mediterranean Conference on Control and Automation*, Ajaccio, France, 2008.

[33] geobiologie (http ://www.geobiologie.fr). Table de conversion : Tesla en gauss.

[34] J.P. Georges, T. Divoux, and E. Rondeau. *Evaluation des majorants des délais de transmission pour les systèmes contrôlés en réseau dans Systèmes commandés en réseau*. Hermès Science, 2007.

[35] J.P. Georges, N. Vatanski, E. Rondeau, C. Aubrun, and S.L. Jamsa-Jounela. Control compensation based on upper bound delay in networked control system. In *17th International Symposium on Mathematical Theory of Networks and Systems*, Kyoto, Juillet 2006.

[36] J. Gertler. Survey of model-based failure detection and isolation in complex plants. *IEEE Control Systems Magazine*, 9 :3–11, 1988.

[37] J. Gertler. Fault detection and isolation using parity relations. *Control Eng. Practice*, 5 :653–661, 1997.

[38] G.Juanole and G.Mouney. Using an hybrid traffic scheduling in networked control systems. *Proceeding of the 10 th European Control Conference*, 2007.

[39] J.F. Guerrero-Castellanos. *Estimation de l'attitude et commande bornée en attitude d'un corps rigide : Application à un mini helicoptère à quatre rotors*. PhD thesis, Université Joseph Fourier-Grenoble I, 2008. (French).

[40] T. Henningsson and A. Cervin. Event-Based Control over Networks : Some Research Questions and Preliminary Results. *Reglermote*, 2006.

[41] T. Henningsson, E. Johannesson, and A. Cervin. Sporadic Event-Based Control of First-Order Linear Stochastic Systems. *Automatica*, 44(11) :2890–2895, November 2008.

[42] D. Henry. From fault diagnosis to recovery actions for aeronautic and aerospace missions : a model-based point of view. In *23rd IAR Workshop on Advanced Control and Diagnosis*, Coventry, UK, 2008.

[43] G. Heredia, A. Oilero, M. Bejar, and R. Mahtani. Sensor and actuator fault detection in small autonomous helicopter. *Mechatronics*, 18 :90–99, 2008.

[44] M.R. Hestenes. *Optimization Theory : The finite dimensional case*. John Wiley & Sons, Inc, 1975.

[45] Safe Necs : Safe-Networked Control Systems. (http ://safe-necs.cran.uhp nancy.fr/).

[46] http ://www.frameip.com/entete ethernet/. Tout sur ethernet.

[47] MEMSense (http//www.memsense.com). Mag3 : Triaxial magnetometer, accelerometer & gyroscope analog inertial sensor.

[48] Onera : (http ://www.onera.fr/conferences/drones/). Mieux connaître les drones.

[49] Phytec (http ://www.phytec.fr/). phycore-mpc.

[50] CiA : CAN in Automation (http ://www.can cia.org/). Cia : Can in automation (http ://www.can-cia.org/).

[51] Dranganfly innovation INC (http ://www.draganfly.com/).

[52] Inria. Orccad (http ://www.inrialpes.fr/sed/orccad/).

[53] R. Isermann. Process fault detection based on modeling and estimation methods : A survey. *Automatica*, 20 :387–404, 1984.

[54] R. Isermann. *Fault Diagnosis Systems : An introduction from fault detection to fault tolerance*. Springer-Verlag, 2006.

[55] R. Isermann and P. Ballé. Trends in the application of model-based fault detection and diagnosis of technical processes. *Control Eng Practice*, 5 :709–719, 1997.

[56] J.Eidson and W.Cole. Ethernet rules closed-loop system. *InTech*, 1998.

[57] N. Jia. *Conception conjointe optimisée de lois de contrôle et d'ordonnancement.* PhD thesis, Institut National Polytechnique de Lorraine, 2009.

[58] N. Jia, Y. Song, and F. Simonot-Lion. Graceful degradation of the quality of control through data drop policy. In *9th European Control Conference, ECC'07*, Kos, Greece, 2007.

[59] G. Juanole, G. Mouney, and C. Calmettes. On different priority schemes for the message scheduling in networked control systems. *Proceeding of 16th Mediterranean Conference on Control and Automation*, 2008.

[60] J. Korbicz, J. Kościelny, Z. Kowalczuk, and W. Cholewa. *Fault diagnosis. Models, Artificial Intelligence, Applications.* Springer-Verlag, 2004.

[61] L.S. Lasdon. *Optimization Theory for Large Systems.* Dover publication Inc., 2002.

[62] S. Lesecq, K. Chabir, S. Gentil, and D. Sauter. Extended kalman filter taking acoount of extra delays and intermittent data in Network Controlled Systems. In *7th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes*, Barcelone, 2009.

[63] S. Lesecq, S. Gentil, and N. Daraoui. Quadrotor attitude estimation with data losses. In *9th European Control Conference 2009 (ECC'09)*, 2009.

[64] D. Maquin and J. Ragot. *Diagnostic des systèmes linéaires.* Hermes, 2000. (French).

[65] J.L. Marins, X.Yun, E.R. Bachmann, D. R.B. McGhee, and M.J. Zyda. An Extended Kalman Filter for Quaternion-Based Orientation Estimation Using MARG Sensors. In *Proceeding of the 2001 IEEE/RSI International Conference on Intelligent Robots and Systems*, Maui, Hawai, 2001.

[66] F.L. Markley. Attitude determination using vector observations and the singular value decomposition. *Journal of Astronautical Sciences*, 38 :245–258, 1988.

[67] Microchip. dspic33f family : Data sheet. Technical report.

[68] Microchip. Pic18f6585/8585/6680/8680 : Data sheet. Technical report, 2004.

[69] M.R. Napolitano, Y. An, and A. Seanor. A fault tolerant flight control system for sensor and actuator failure using neural network. *Aircraft Design*, 3 :103–128, 2000.

[70] M.R. Napolitano, D.A. Windson, and J.L. Casanova. Kalman filters and neural-network schemes for sensor validation in flight control systems. *IEEE Transaction on Control Systems Technology*, 6 :596–611, 1998.

[71] N. Navet. *Evaluation de performances temporelles et optimisation de l'ordonnancement de tâches et messages.* PhD thesis, Institut National Polytechnique de Lorraine, 1999.

[72] M. Ohlin, D. Henriksson, and A. Cervin. TrueTime 1.5—Reference Manual. January 2007.

[73] R.J. Patton. Fault-tolerant control systems : The 1997 situation. In *proceeding of IFAC Symposium Safeprocess*, 1997.

[74] R.J. Patton and J. Chen. Observer-based fault detection and isolation : robustness and applications. *Control eng. Pratice*, 5 :671–682, 1997.

[75] R.J. Patton, F.J. Uppal, S. Simani, and B. Polle. A Monte Carlo analysis and design for FDI of a satellite control system. In *Proceedings of Safeprocess'2006*, Beijing, China, 2006.

[76] J.P. Richard and T. Divoux. *Systèmes commandés en réseau.* Lavoisier, 2007.

[77] A.L. Satin and R.L.Gates. Evaluation of parity equations for gyro failure detection and isolation. *Journal of Guidance, Control and Dynamics*, 1 :14–20, 1978.

[78] D. Sauter and F. Hamelin. *Commande tolérante aux défauts dans Supervision des procédés complexes.* Hermes, 2007.

[79] O. Sename, D. Simon, and M. Ben Gaid. A LPV approach to control and real-time scheduling codesign : application to a robot-arm control. In *47th IEEE Conference on Decision and Control, CDC2008*, 2008.

[80] M.D. Shuster. A survey of attitude representations. *The Journal of the Astronautical Sciences*, 41 :439–517, 1993.

[81] D. Simon, R. Pissard-Gibollet, and S. Arias. Orccad, a framework for safe robot control design and implementation. In *1st National Workshop on Control Architectures of Robots : software approaches and issues*, 2006.

[82] Y.S. Suh. Attitude Estimation by Multiple-Mode Kalman Filters. *IEEE Transactions on industrial electronics*, 53 :155, 2006.

[83] A. Tanenbaum. *Réseaux. Cours et exercices.* Prentice Hall, 1999.

[84] A. Tanwani. Stratégies de diagnostic pour un système embarqué : Application à un mini drone 4 rotors. Master's thesis, INPG, 2006.

[85] A. Tanwani, J. Galdun, J.M. Thiriet, S. Lesecq, and S. Gentil. Experimental Networked Embedded Minidrone - part I. Consideration of faults. In *Proceeding of ECC 2007*, Kos, Greece, 2007.

[86] A. Tanwani, S. Gentil, S. Lesecq, and J.M. Thiriet. Experimental Networked Embedded Minidrone - part II. distributed FDI. In *Proceeding of ECC 2007*, Kos, Greece, 2007.

[87] Commission technique et aérospatiale. Les avions de combat sans pilote et l'avenir de l'aéronautique militaire (document a/1884). Technical report, Assemblée interparlementaire européenne de sécurité et de défense, 2004.

[88] Y. Tipsuwan and M.Y Chow. Network-based Controller Adaptation based on QoS Negotiation and Deterioration. In *IECON'01 : The 27th Annual Conference of the IEEE Industrial Electronics Society*, 2001.

[89] Y. Tipsuwan and M.Y Chow. Control methodologies in networked control systems. *Control Eng Practice*, 11 :1099–1111, 2003.

[90] P. Tsiotras. New control laws for the attitude stabilization of rigid bodies. In *13th IFAC Symposium on Automatic Control in Aerospace*, 1994.

[91] N. Vatanski, J.P. Georges, C. Aubrun, and S.L. Jamsa-Jounela. Control reconfiguration in networked control system. In *IFAC Symposium Safeprocess*, Beijing, PRC, 2006.

[92] V. Venkatasubramanian, R. Rengaswamy, and S. Kavuri. A review of process fault detection and diagnosis. part I : Quantitative model-based methods. *Computer Chem Eng.*, 27 :293–311, 2003.

[93] G.C Walsh and Y. Hong. Scheduling of networked control systems. *IEEE Control Systems Magazine*, 21 :57–65, 2001.

[94] wikipedia (http ://en.wikipedia.org/wiki/Accelerometer). Accelerometer.

[95] wikipedia (http ://en.wikipedia.org/wiki/http ://en.wikipedia.org/wiki/Rate gyro). Rate gyro.

[96] wikipedia (http ://en.wikipedia.org/wiki/Magnetometers). Magnetometers.

[97] wikipedia(http ://fr.wikipedia.org/modele OSI). Modèle OSI.

[98] wikipedia(http ://fr.wikipedia.org/wiki/Angles d'Euler). Angle d' Euler.

[99] wikipedia(http ://fr.wikipedia.org/wiki/Controllerareanetwork). Controller area network.

[100] wikipedia(http ://fr.wikipedia.org/wiki/Ethernet). Ethernet.

[101] wikipedia(http ://fr.wikipedia.org/wiki/IPv4). Ipv4.

[102] wikipedia(http ://fr.wikipedia.org/wiki/IPv6). Ipv6.

[103] A.S. Willsky. A survey of design methods for failure detection in dynamic systems. *Automatica*, 12 :601–611, 1976.

[104] M. Witczak. *Modelling and Estimation Strategies for Fault Diagnosis of Non-Linear Systems : From analytical to soft computing approaches*. Springer, 2007.

[105] MathWorks (www.mathworks.com). Optimization toolbox-fmincon.

[106] MathWorks (www.mathworks.com). Optimization toolbox-fminunc.

[107] F. Zhang. Quaternions and matrices of quaternions. *Linear Algebra Appl.*, 21 :21–57, 1997.

[108] J. Zhang and J. Jiang. Modeling of vertical gyroscopes with consideration of fauts. In *Proceedings of Safeprocess'2006*, Beijing, China, 2006.

[109] W. Zhang. *Stability analysis of networked control system*. PhD thesis, Department of Electrical Engineering and Computer Science, Western Reserve University, Aout 2001.

[110] W. Zhang, M.S. Branicky, and S.M. Phillips. Stability of networked control systems. *IEEE Control Systems Magazine*, 21 :84–99, 2001.

[111] K. Zuberi and K. Shin. Scheduling messages on Controller Area Network for real time CIM applications. *IEEE transactions On Robotics And Automation*, 1997.

Résumé : Les systèmes commandés en réseau sont des systèmes où le réseau est utilisé comme moyen de communication dans la boucle de commande. En conséquence, la qualité de commande (QoC) ou de la supervision de ces systèmes est directemenet liée à celle du réseau. Ce travail de doctorat aborde plusieurs problèmes. Le premier consiste à mettre au point un système de diagnostic performant permettant de détecter et localiser un défaut survenant sur un capteur particulier, une centrale d'attitude, embarqué dans un quadrotor. Le deuxième concerne la mise en oeuvre du système en réseau. Il faut tenir compte du réseau et de sa qualité de service (QoS) dans les algorithmes de commande et de diagnostic. Ensuite pour garantir les performances du système, il faut modifier la QoS du réseau. La réalisation d'une plateforme expérimentale a permis de valider les solutions proposées.

Mots clefs : Supervision, Détection, Localisation, Identification, Commande à travers un réseau, Quadrotor.

Abstract: The systems controlled through a network are the systems where the network is used as communication channel in the control loop. However, the quality of control (QoC) or supervision of theses systems is directly influenced by the QoS of the network. In this work, some problems are considered. The first is to design a diagnostic module to detect and isolate a sensor faults in an Inertial Measurement Unit applied to a quadrotor. The second is to design a networked control system where the network and its QoS must be taking into account in the control and diagnostic algorithms. After that, to guarantee the performance of the system, the QoS of the network must be modified. A benchmark has been realized to valid all solutions proposed.

Keywords : Diagnosis, Detection, Isolation, Identification, Networked Control Systems, Quadrotor.

Laboratoire GIPSA-lab - ENSE^3 - BP 46, 38402 Saint-Martin d'Hèeres, FRANCE.