

Projet Drone Au323

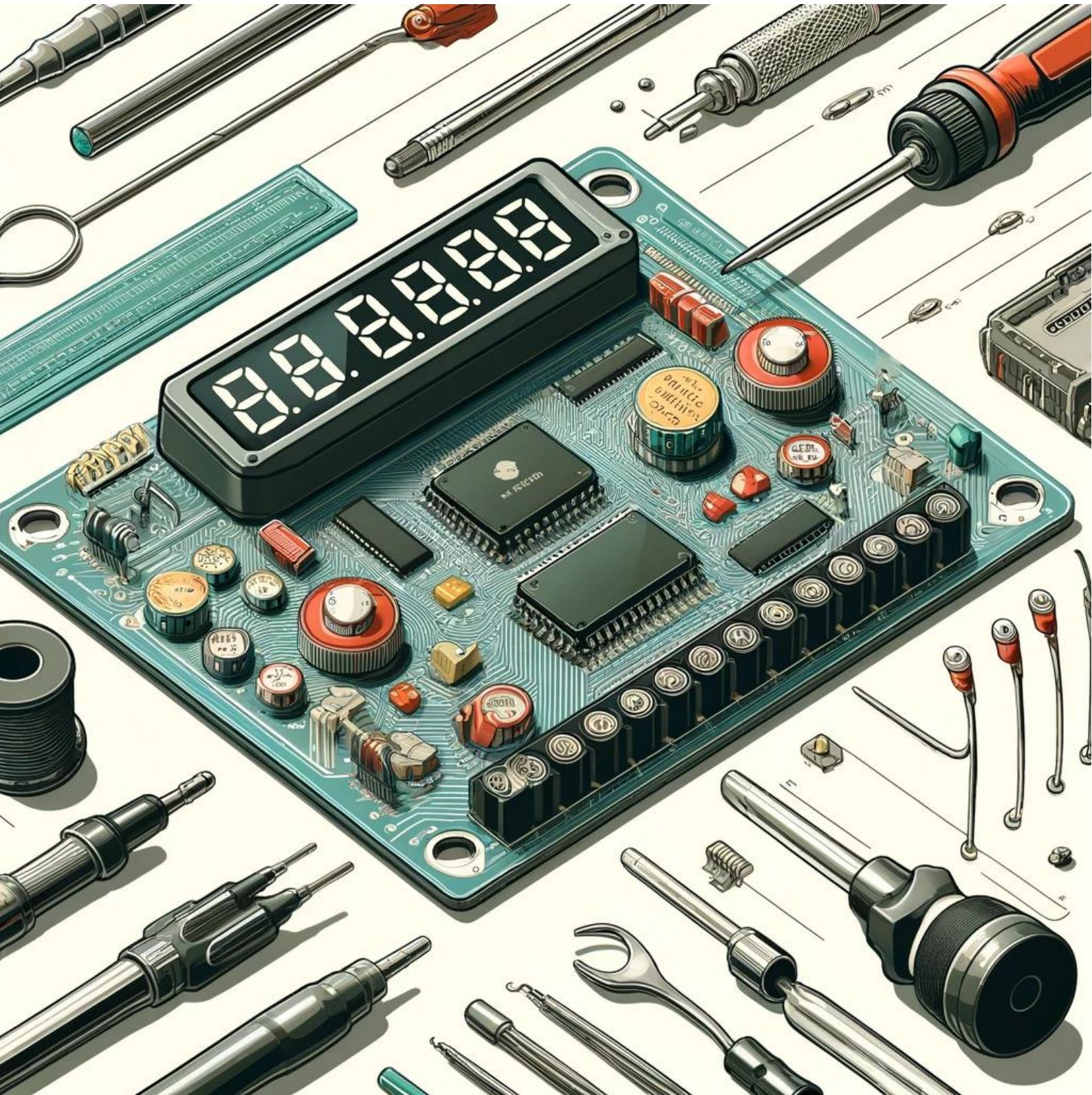


Table des matières

Préambule	3
Introduction	4
Partie 1 : Simulation du drone	5
Partie 2 : Ajout d'un régulateur PID	13
Partie 3 : Réalisation d'un régulateur par Placement de pôles	18
Conclusion	20
Annexes	21

Préambule

Ce TP est basé sur un article de conférence sur la commande d'un micro-drone quadrirotor :

Auteurs: Bouabdallah, S.; Noth, A.; Siegwart, R

Titre : PID vs LQ Control Techniques Applied to an Indoor Micro Quadrotor

Conférence : IROs'2004, Sendai, Japan, 2004

L'article est téléchargeable depuis le site de L'Ecole Polytechnique Fédérale de Lausanne EPFL :

[Article sur la commande d'un micro-drone quadrirotor](#)

Les paramètres nécessaires à la simulation sont donnés en Annexe (faire attention aux unités).

Il est demandé aux étudiants de :

- Lire l'article jusqu'au paragraphe IV.
- Vérifier la cohérence entre les équations des couples (7) et la figure 1.
- Traduire les termes des tableaux de l'article (page 3) (utiliser par exemple [Dictionnaire](#))
- Vérifier le passage des équations du moteur à courant continu de (11) à (12) et de (12) à (13).

Introduction

Dans le cadre de ce projet, nous entreprendrons l'étude et la simulation de la stabilité d'un drone quadrirotor à l'aide des outils Matlab et Simulink. Le travail s'appuiera sur un article de conférence intitulé "PID vs LQ Control Techniques Applied to an Indoor Micro Quadrotor", rédigé par Bouabdallah, Noth et Siegwart, présenté lors de l'IROS 2004.

Le projet sera structuré autour de plusieurs objectifs clés :

1. **Simulation des Dynamiques de Vol** : Nous commencerons par modéliser les dynamiques de roulis, de tangage et de lacet du drone, en créant plusieurs versions de modèles sous Simulink. Ces modèles nous permettront de comprendre et de visualiser le comportement du drone sous différentes conditions initiales et configurations d'hélices.
2. **Implémentation de Contrôleurs PID** : La seconde phase consistera à intégrer des régulateurs PID pour contrôler l'attitude du drone. Cette étape nous aidera à stabiliser le système et à maintenir le drone dans une configuration d'équilibre malgré les perturbations.
3. **Commande par Placement de Pôles** : Pour une approche plus avancée, nous linéariserons le modèle du drone et appliquerons une commande par placement de pôles. Cette technique nous permettra de définir des pôles stables et d'examiner la réponse du système sous différentes configurations.
4. **Régulation PID Filtrée Discrète** : Enfin, nous explorerons l'implémentation d'un régulateur PID filtré dans un contexte discret, renforçant ainsi notre compréhension des systèmes de contrôle dans des environnements numériques.

Ce rapport détaillera les différentes étapes de notre projet, les méthodologies employées, ainsi que les résultats obtenus à chaque phase. Les simulations et analyses présentées offriront une vue d'ensemble des défis liés à la stabilité et au contrôle des drones quadrirotors

Partie 1 : Simulation du drone

On s'intéresse à la simulation des dynamiques de roulis, de tangage et de lacet du drone, et non pas à celle des mouvements de translation (selon les axes x, y et z). Dans ce projet, les équations (9) et (10) sont suffisantes. Celui qui s'intéresse au modèle complet (rotations/translations) peut se référer au rapport de thèse du premier auteur.

$$\ddot{\phi} = \dot{\theta}\dot{\psi} \left(\frac{I_y - I_z}{I_x} \right) - \frac{J}{I_x} \dot{\theta}\dot{\Omega} + \frac{l}{I_x} U_1$$

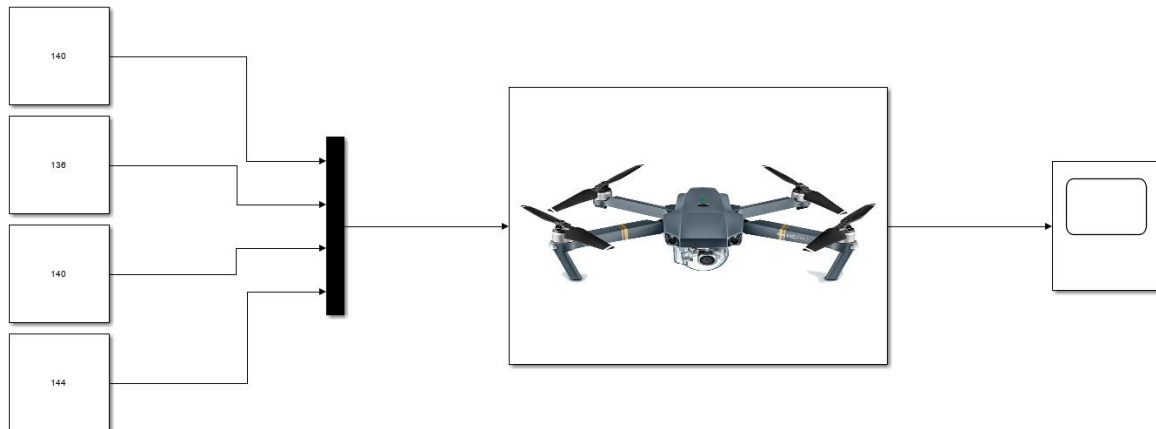
$$\ddot{\theta} = \dot{\phi}\dot{\psi} \left(\frac{I_z - I_x}{I_y} \right) + \frac{J}{I_y} \dot{\phi}\dot{\Omega} + \frac{l}{I_y} U_2$$

$$\ddot{\psi} = \dot{\phi}\dot{\theta} \left(\frac{I_x - I_y}{I_z} \right) + \frac{1}{I_z} U_3$$

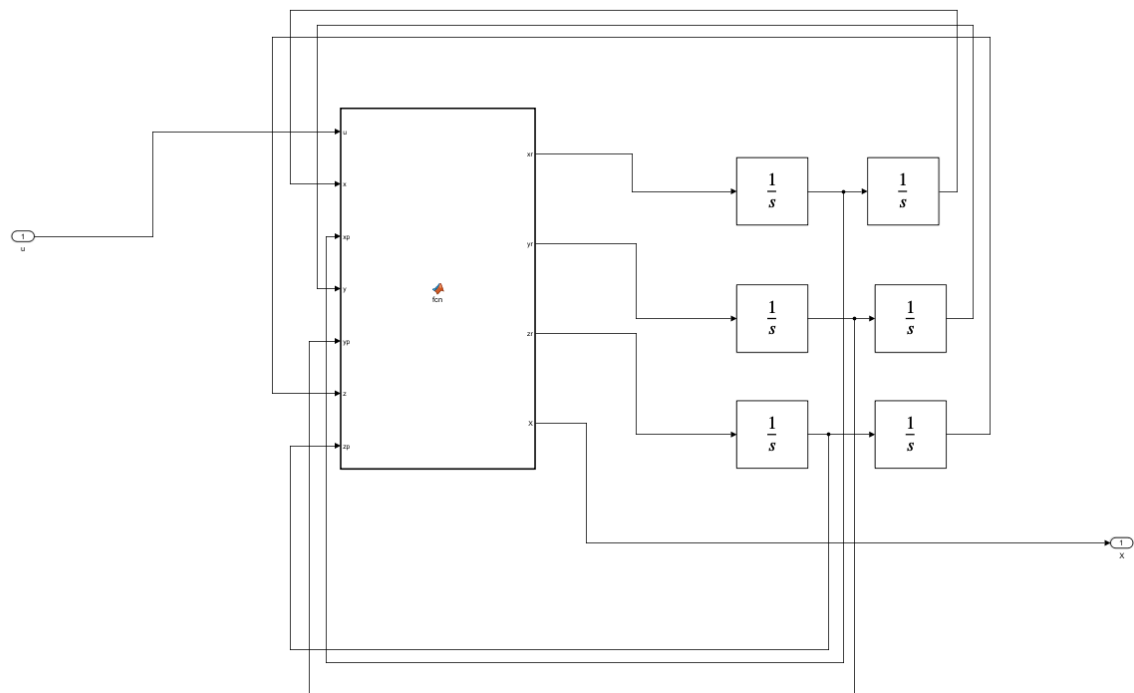
$$\begin{cases} U_1 = b(\Omega_4^2 - \Omega_2^2) \\ U_2 = b(\Omega_3^2 - \Omega_1^2) \\ U_3 = d(-\Omega_1^2 - \Omega_3^2 + \Omega_2^2 + \Omega_4^2) \\ \Omega = \Omega_2 + \Omega_4 - \Omega_1 - \Omega_3 \end{cases}$$

Créer un modèle Simulink, réaliser deux à quatre versions de cette partie mécanique :

1. Une version en utilisant six blocs Integrator et un bloc Matlab Function



Et à l'intérieur du subsysteme on a :



Le code utilisé :

```
function [xr, yr,zr,X] = fcm(u,x,xp,y,yp,z,zp)
Ix = 7.5e-3;
Iy = 7.5e-3;
Iz = 7.5e-3;
J = 6e-5;
l = 0.23;
b= 3.13e-5;
d= 7.5e-7;

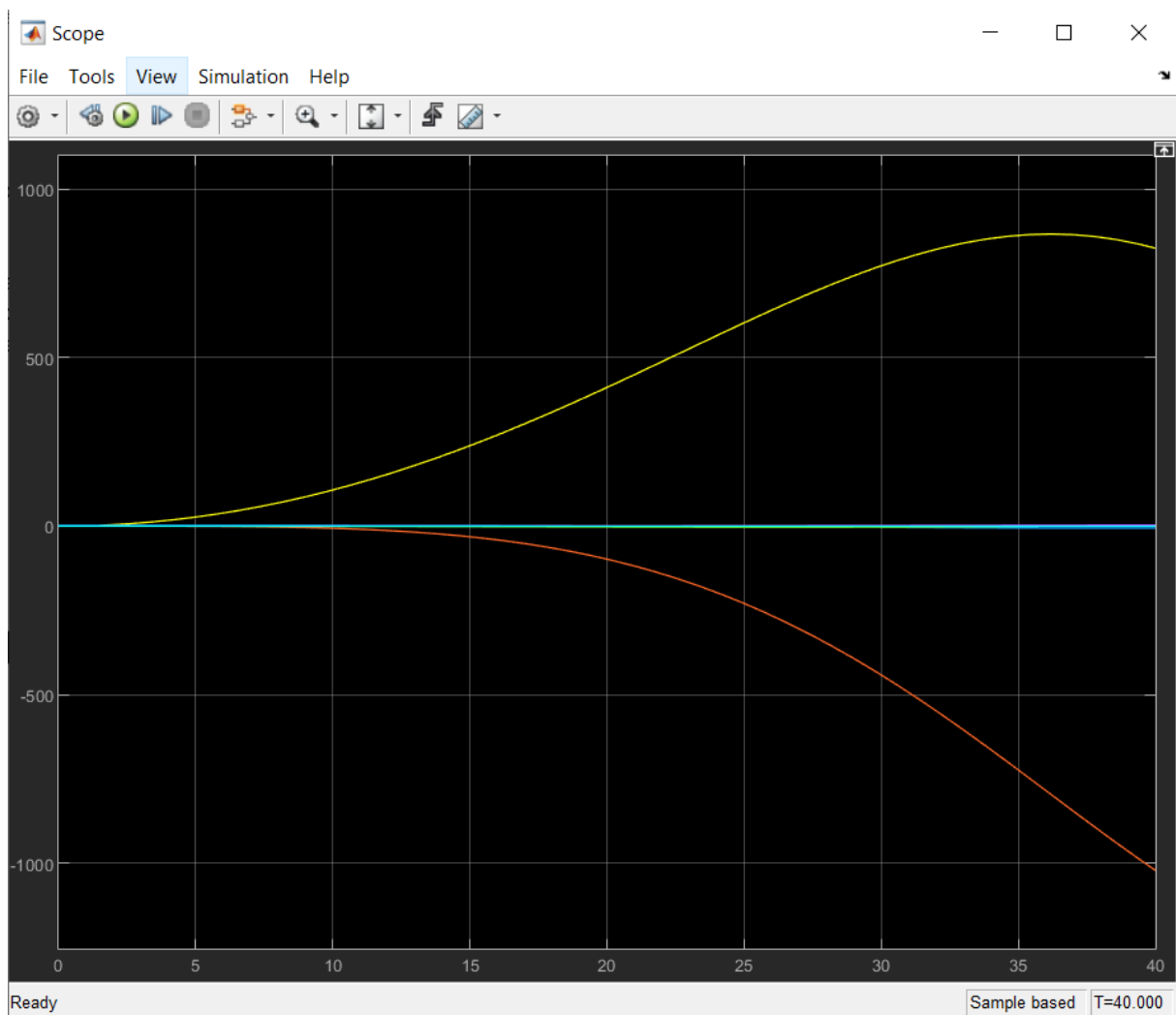
Om1=u(1);
Om2=u(2);
Om3=u(3);
Om4=u(4);

U1=b*(Om4^2-Om2^2);
U2=b*(Om3^2-Om1^2);
U3=d*(Om4^2+Om2^2-Om1^2-Om3^2);
Om = Om2+Om4-Om1-Om3;

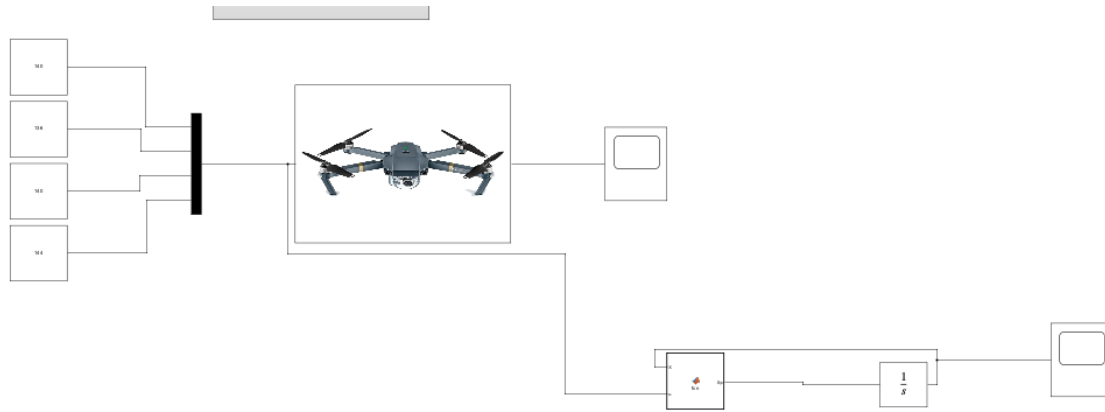
xr = yp*zp*((Iy-Iz)/Ix)-(J/Ix)*yp*Om+(l/Ix)*U1;
yr = xp*zp*((Iz-Ix)/Iy)+(J/Iy)*xp*Om+(l/Iy)*U2;
zr= xp*yp*((Ix-Iy)/Iz)+(l/Iz)*U3;

X = [x,xr,y,yp,z,zr];
```

Quand on lit le scope on observe :



2. Une version représentée par un seul bloc Intégrateur et un bloc Matlab Function (à partir de la représentation d'état). Ne pas oublier de mettre à jour les conditions initiales (vecteur colonne de six éléments).



Voici le code :

```
function Xp= fcn(X, u)
Ix = 7.5e-3;
Iy = 7.5e-3;
Iz = 7.5e-3;
J = 6e-5;
l = 0.23;
b= 3.13e-5;
d= 7.5e-7;

Om1=u(1);
Om2=u(2);
Om3=u(3);
Om4=u(4);

U1=b*(Om4^2-Om2^2);
U2=b*(Om3^2-Om1^2);
U3=d*(Om4^2+Om2^2-Om1^2-Om3^2);
Om = Om2+Om4-Om1-Om3;

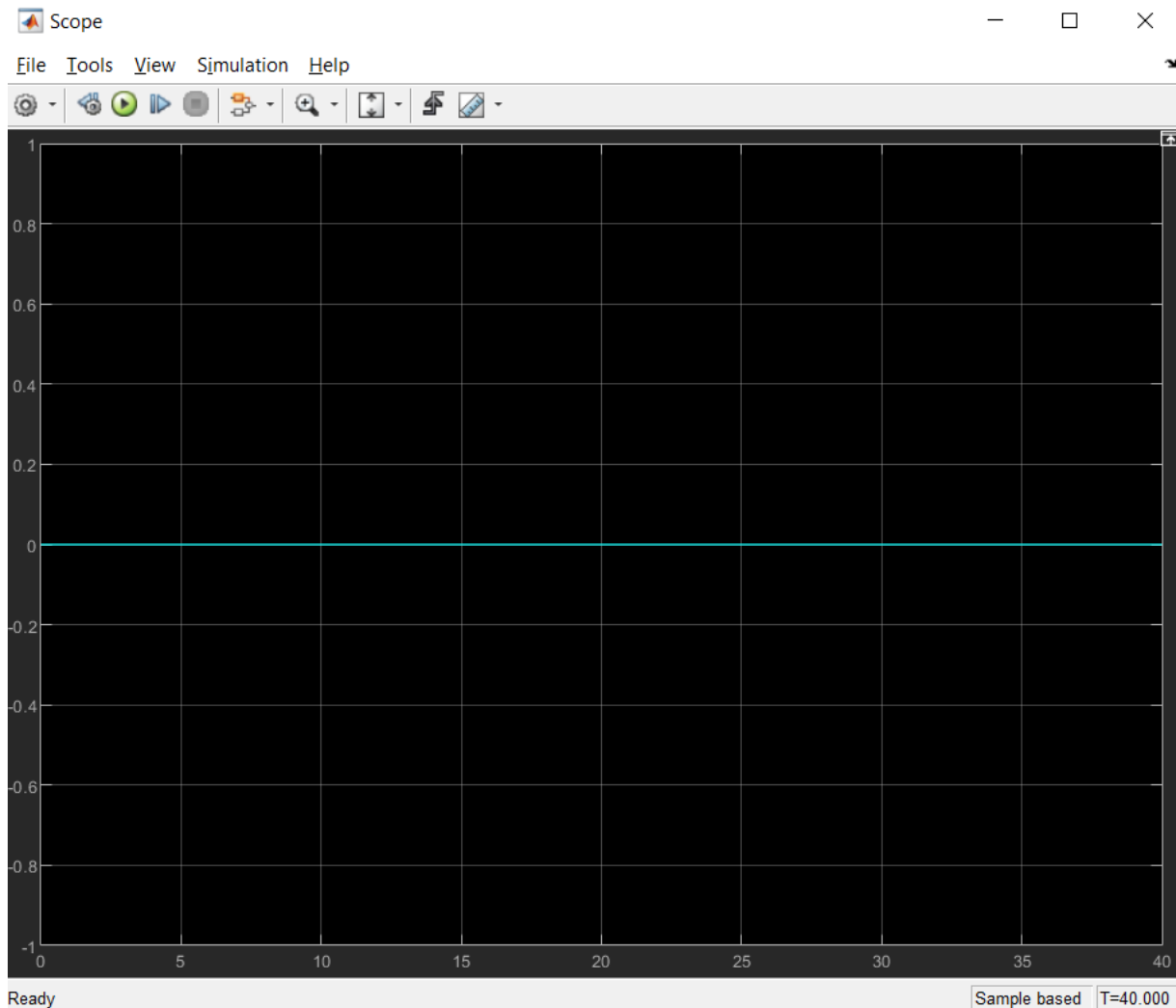
xr = X(4)*X(6)*((Iy-Iz)/Ix)-(J/Ix)*X(4)*Om+(1/Ix)*U1;
yr = X(2)*X(6)*((Iz-Ix)/Iy)+(J/Iy)*X(2)*Om+(1/Iy)*U2;
zr= X(2)*X(4)*((Ix-Iy)/Iz)+(1/Iz)*U3;

Xp = [X(2),xr,X(4),yr,X(6),zr];
```


c) Première simulation : Les hélices ont toutes la même vitesse (140 rad/s). Le drone peut-il voler avec ces paramètres ? Justifier en calculant sa portance, sachant que la portance de chaque hélice est donnée par la relation : $F_1 = b \times \Omega_i^2$.

- Simuler dans ce cas le système avec des conditions initiales nulles. Utiliser le bloc Step pour simuler les entrée Ω_i . Tracer les courbes des angles en fonction du temps.

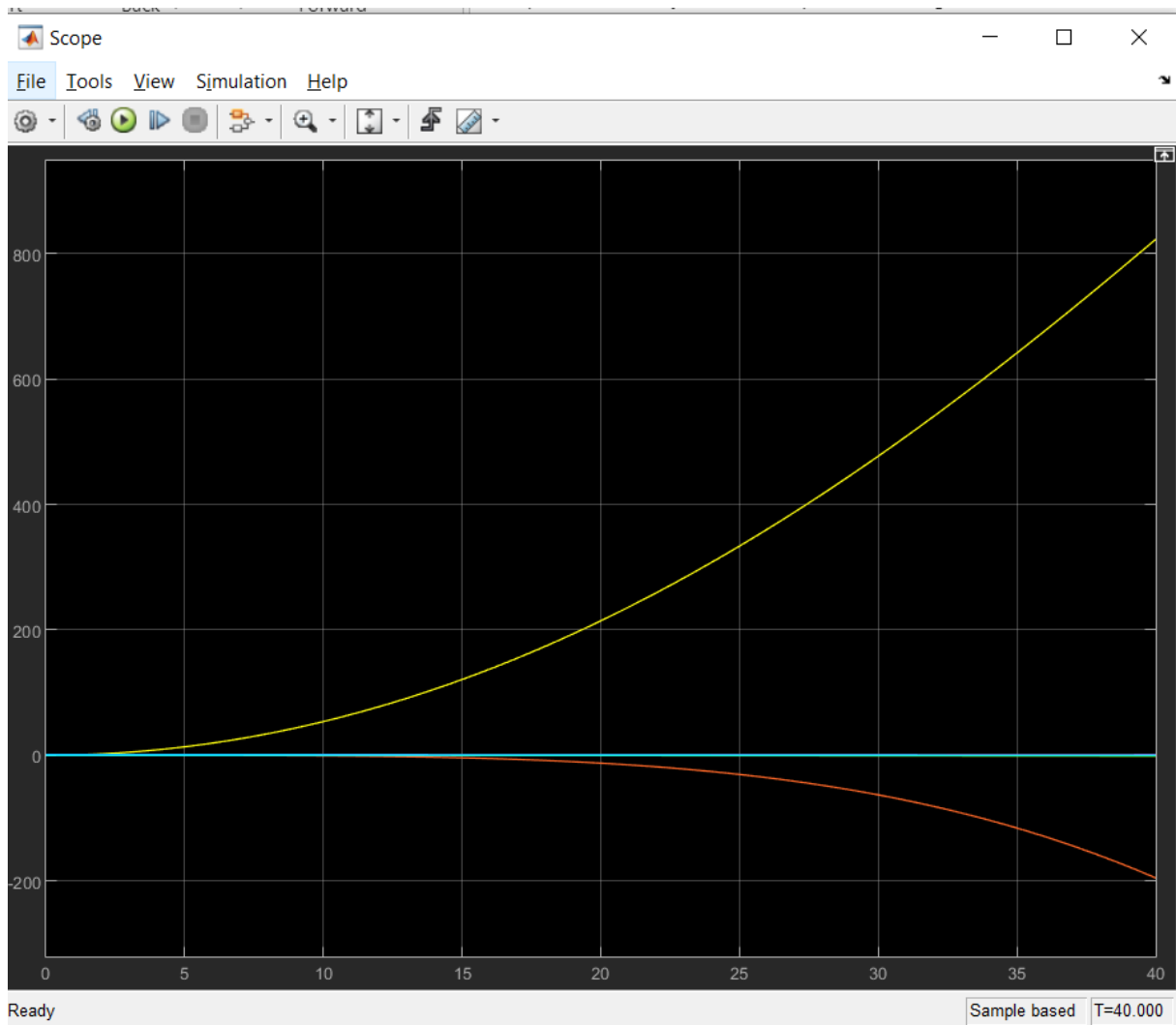
Avec la deuxième version, on obtient ce graphique :



On a le résultat souhaité, le drone est stable, avec les courbes de position et de vitesse des angles à 0.

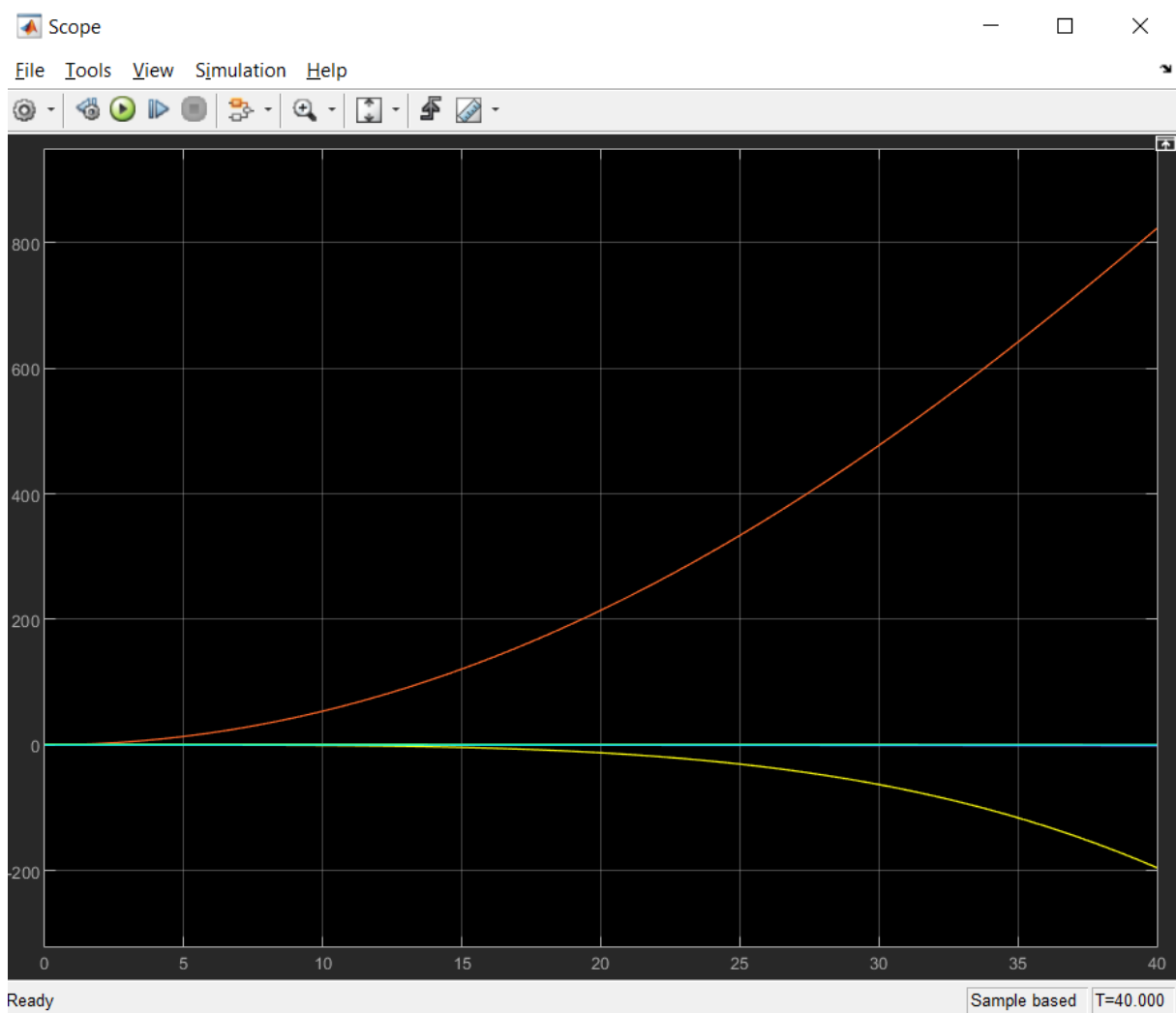
Deuxième simulation : Modifier les vitesses des hélices afin de simuler les trois cas suivants :

- petit roulis initial. $\Omega_{m1}=140$ $\Omega_{m2}=138$ $\Omega_{m3}=140$ $\Omega_{m4}=142$:



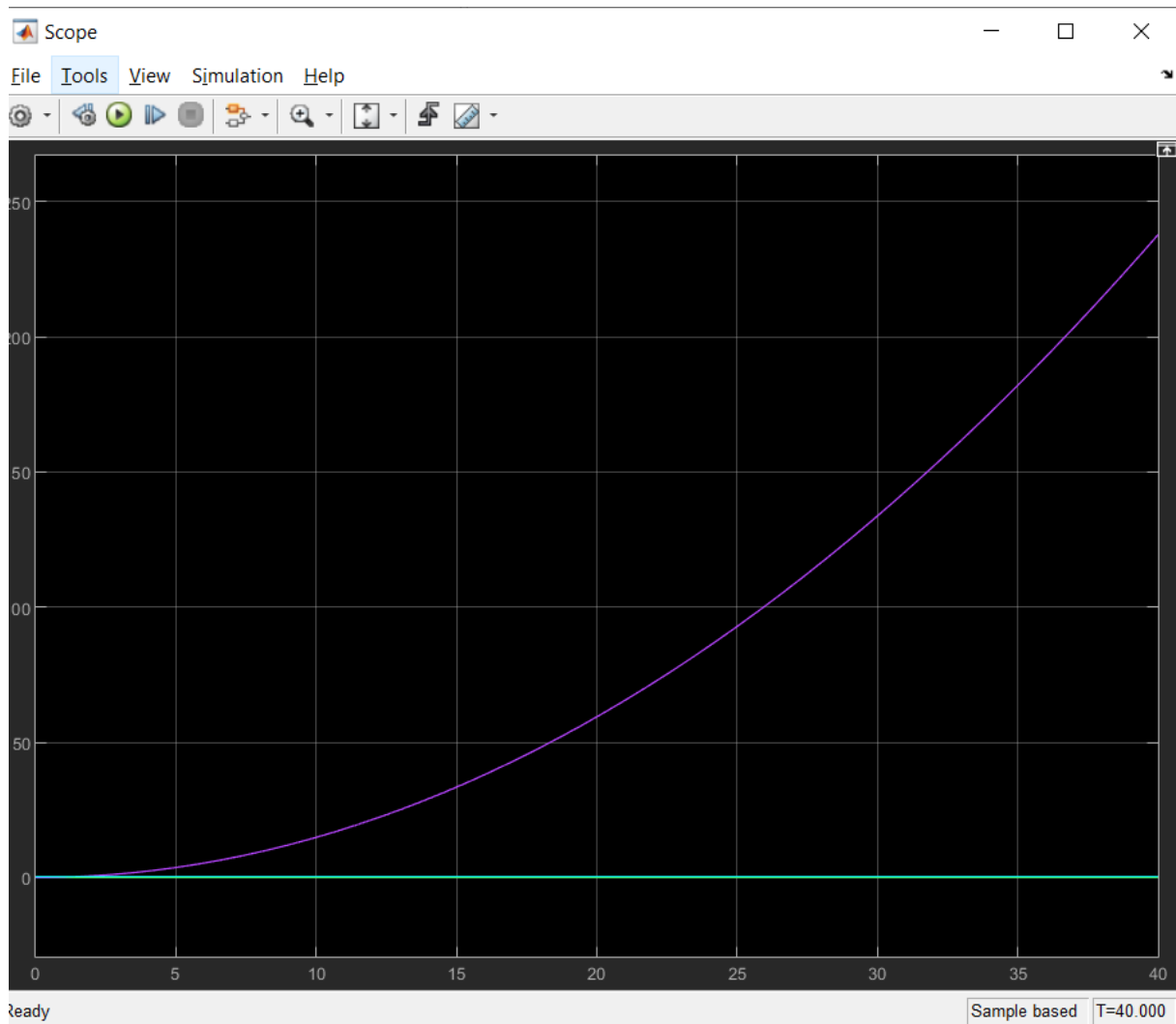
Nous observons le mouvement de roulis attendu

2^{ème} cas : mouvement de tangage. $\Omega_1=138$ $\Omega_2=140$ $\Omega_3=142$ $\Omega_4=140$:



On observe bien le mouvement de tangage attendu.

- 3^{ème} cas : le lacet. $\Omega_{m1}=138$ $\Omega_{m2}=142$ $\Omega_{m3}=138$ $\Omega_{m4}=142$

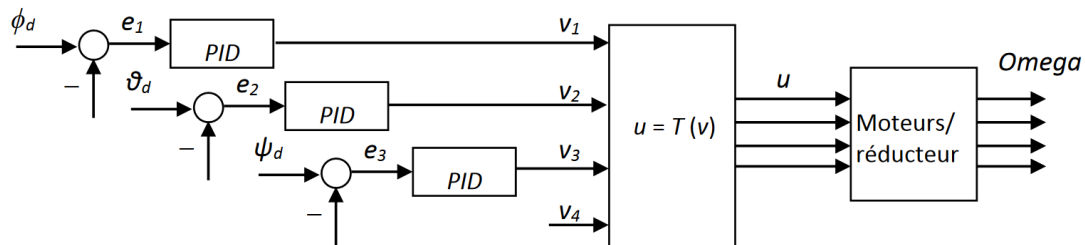


On a bien le mouvement de lacet attendu.

Et pour finir, on observe bien que pour chaque cas, seul l'axe concerné va varier, si l'on respecte la condition : $\sum_1^4 F_i = \text{cste}$

Partie 2 : Ajout d'un régulateur PID

Pour contrôler l'altitude du drone, on implémente trois régulateurs PID. Cependant, le système a quatre entrées (les tensions u_1, u_2, u_3, u_4). En se basant sur les équations (16), la commande du drone peut être réalisée selon le schéma suivant, avec $v_4 = 16$:



Les trois contrôleurs PID interviennent pour ajuster l'altitude $[\phi, \theta, \psi]^T$ autour de la configuration d'équilibre. Ainsi, v_1 agit sur le couple du roulis, v_2 sur le couple de tangage et v_3 sur le couple de lacet.

a) Que représente physiquement les variables $v(i)$?

Pour le contrôle de l'attitude d'un drone quadrirotor avec des régulateurs PID, les variables $v(i)$ jouent un rôle crucial dans le système de commande.

$v(1)$: Commande de Roulis (ϕ) :

- Cette variable est la sortie du régulateur PID responsable du contrôle du roulis (ϕ).
- Le roulis est la rotation autour de l'axe longitudinal (x) du drone.
- Le régulateur PID ajuste $v(1)$ pour maintenir ou atteindre un angle de roulis désiré (ϕ_d).

2. $v(2)$: Commande de Tangage (θ) :

- Cette variable est la sortie du régulateur PID responsable du contrôle du tangage (θ).
- Le tangage est la rotation autour de l'axe transversal (y) du drone.
- Le régulateur PID ajuste $v(2)$ pour maintenir ou atteindre un angle de tangage désiré (θ_d).

3. $v(3)$: Commande de Lacet (ψ) :

- Cette variable est la sortie du régulateur PID responsable du contrôle du lacet (ψ).
- Le lacet est la rotation autour de l'axe vertical (z) du drone.
- Le régulateur PID ajuste $v(3)$ pour maintenir ou atteindre un angle de lacet désiré (ψ_d).

4. **v(4) : Commande de Portance Totale (Force Verticale) :**

- Cette variable est liée à la portance verticale du drone et correspond à la somme des forces générées par les quatre hélices pour contrebalancer la gravité.
- En pratique, v(4) est utilisé pour contrôler la position verticale du drone (altitude).

Schéma de Commande

Dans le schéma de commande du drone quadrirotor, les régulateurs PID interviennent pour ajuster les angles de roulis, de tangage et de lacet autour de leurs configurations d'équilibre. Les sorties des régulateurs PID (v(1), v(2), et v(3)) sont combinées pour générer les commandes des forces de portance des hélices F_i à travers une transformation, pour la stabilité et le contrôle du drone.

- b) Etudier l'équation (16) de l'article et vérifier s'il n'y a pas d'erreur.

$$\begin{aligned}\phi(s) &= \frac{B^2 b l}{s^2 (s + A)^2 I_x} (u_2^2(s) - u_4^2(s)) \\ \theta(s) &= \frac{B^2 b l}{s^2 (s + A)^2 I_y} (u_3^2(s) - u_1^2(s)) \\ \psi(s) &= \frac{B^2 b l}{s^2 (s + A)^2 I_z} (u_1^2(s) + u_3^2(s) - u_2^2(s) - u_4^2(s))\end{aligned}$$

Les équations semblent correctes, car elles respectent les principes de la dynamique des quadrirotors :

- **Roulis et Tangage** : Dépendant de la différence des carrés des vitesses des hélices opposées.
- **Lacet** : Dépendant de la somme et de la différence des carrés des vitesses des hélices, équilibrant les contributions de toutes les hélices.

- c) Comment a-t-on trouvé l'expression de la transformation T (Rectifier son expression si nécessaire) ?

On cherche un système de la forme :

$$u = T(v)$$

On sait que : $v = M \times u$, avec $M = \begin{pmatrix} 0 & -1 & 0 & 1 \\ -1 & 0 & 1 & 0 \\ -1 & 1 & -1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}$

et $M^T = \begin{pmatrix} 0 & -1 & -1 & 1 \\ -1 & 0 & 1 & 1 \\ 0 & 1 & -1 & 1 \\ 1 & 0 & 1 & 1 \end{pmatrix}$

$$v = M \times u \Leftrightarrow \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{pmatrix} = \begin{pmatrix} 0 & -1 & 0 & 1 \\ -1 & 0 & 1 & 0 \\ -1 & 1 & -1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix} \times \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{pmatrix}$$

On a $v = M \times u \Leftrightarrow u = M^{-1}v \Leftrightarrow \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{pmatrix} = \begin{pmatrix} 0 & -1 & 0 & 1 \\ -1 & 0 & 1 & 0 \\ -1 & 1 & -1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}^T \times \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{pmatrix}$

$$\Leftrightarrow \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{pmatrix} = \begin{pmatrix} -0,5 \times v_2 + 0,25 \times (v_4 - v_3) \\ -0,5 \times v_1 + 0,25 \times (v_4 + v_3) \\ 0,5 \times v_2 + 0,25 \times (v_4 - v_3) \\ 0,5 \times v_1 + 0,25 \times (v_4 + v_3) \end{pmatrix}$$

$$\Leftrightarrow \begin{cases} u_1 = -0,5 \times v_2 + 0,25 \times (v_4 - v_3) \\ u_2 = -0,5 \times v_1 + 0,25 \times (v_4 + v_3) \\ u_3 = 0,5 \times v_2 + 0,25 \times (v_4 - v_3) \\ u_4 = 0,5 \times v_1 + 0,25 \times (v_4 + v_3) \end{cases}$$

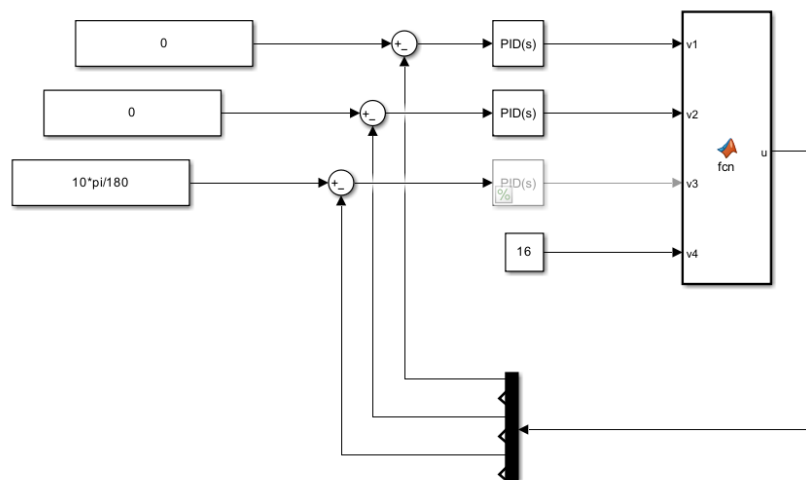
On implémente ces 4 équations afin de pouvoir faire la transformation T.

d) Simuler le système régulé en utilisant trois blocs PID. Modifier si nécessaire les paramètres K_p , K_i , K_d utilisés

pour reproduire les figures Fig. 5 et Fig. 6 (page 4 de l'article). Ici, les angles désirés (les consignes) sont nuls.

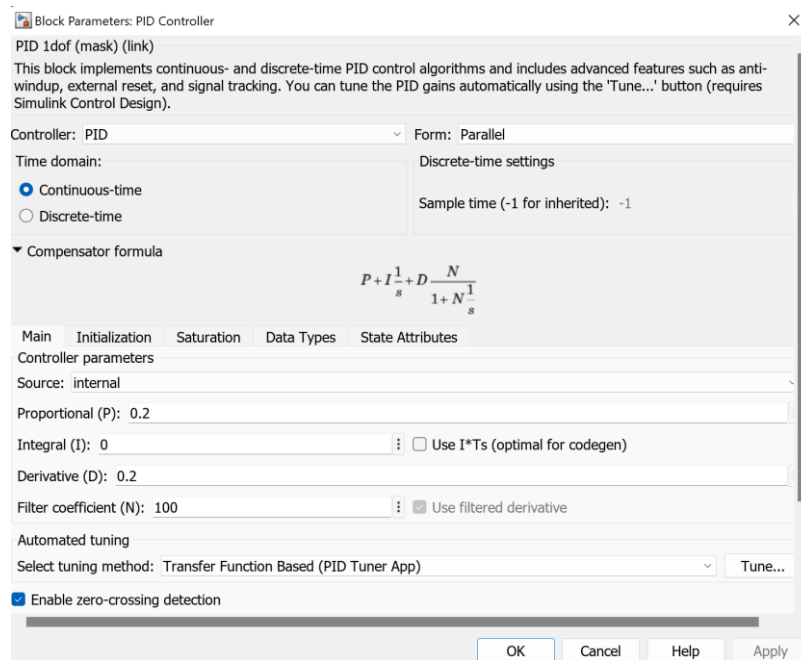
Pour trouver les valeurs du PID (les valeurs d'entrée et sortie respectivement : e_n , v_n), il y a plusieurs étapes :

1. On commence par le PID qui a en entrée e_1 et sortie s_1 : v_2 , v_3 et v_4 sont nuls. On vient mettre $\phi_d = \frac{10\pi}{180}$ et faire varier d'abord le k_p puis k_d afin de reproduire les figures demandées.
2. On vient répéter l'action pour chaque PID.

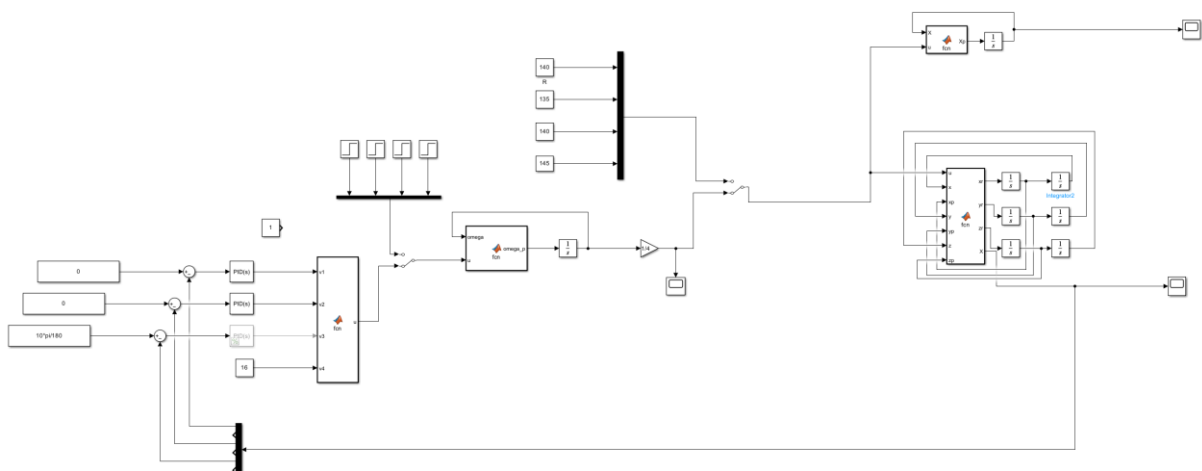


Voici la méthode appliquer pour calculer le dernier PID.

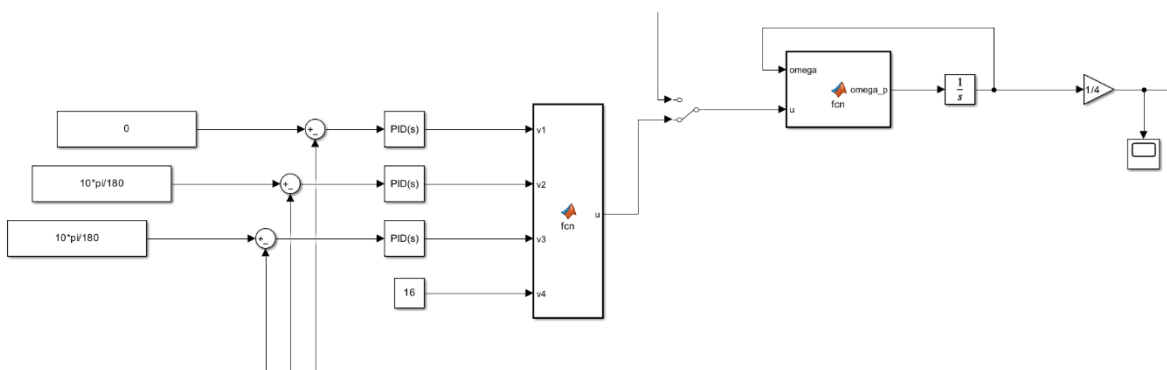
On obtient les paramètres suivants :

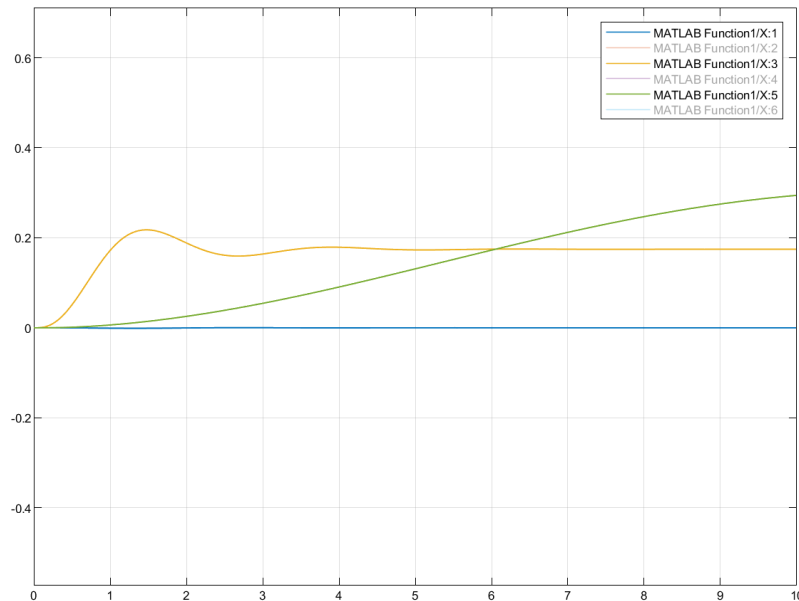


Soit un schéma final qui correspond au suivant :



On fait un essai en faisant varier le tangage et le lacet.





Partie 3 : Réalisation d'un régulateur par Placement de pôles

En se basant sur la représentation d'état du système, on veut contrôler le système autour de la position stationnaire ($X = 0$). Trouver la commande qui permet de maintenir le système à l'équilibre. Linéariser le système autour de cette position.

1. Modélisation Non-Linéaire du Système

Nous partons des équations de la dynamique du quadrirotor :

$$\begin{cases} \ddot{\phi} = \dot{\theta}\dot{\psi} \left(\frac{I_y - I_z}{I_x} \right) - \frac{J}{I_x} \dot{\theta}\Omega + \frac{l}{I_x} U_1 \\ \ddot{\theta} = \dot{\phi}\dot{\psi} \left(\frac{I_z - I_x}{I_y} \right) + \frac{J}{I_y} \dot{\phi}\Omega + \frac{l}{I_y} U_2 \\ \ddot{\psi} = \dot{\phi}\dot{\theta} = \left(\frac{I_x - I_y}{I_z} \right) + \frac{1}{I_z} U_3 \end{cases}$$

2. Linéarisation du Système

Pour linéariser le système, nous considérons un point d'équilibre où les angles et leurs dérivées sont nulles. La représentation d'état non-linéaire est donnée par :

$$\dot{X} = AX + BU$$

Où $X = [\phi, \theta, \psi, \dot{\phi}, \dot{\theta}, \dot{\psi}]^T$

Les matrices A et B du système linéarisé autour du point d'équilibre sont obtenues comme suit :

$$A = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & \frac{I_y - I_z}{I_x} \dot{\psi} & -\frac{J}{I_x} \dot{\theta} \\ 0 & 0 & 0 & \frac{I_z - I_x}{I_y} \dot{\psi} & 0 & \frac{J}{I_y} \dot{\phi} \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$B = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ \frac{l}{I_x} & 0 & 0 \\ 0 & \frac{l}{I_y} & 0 \\ 0 & 0 & \frac{1}{I_z} \end{pmatrix}$$

3. Commande par Placement de Pôles

La commande par placement de pôles consiste à déterminer une matrice de gain K telle que les pôles du système en boucle fermée soient placés à des positions spécifiques désirées. Pour cela, nous utilisons la méthode suivante :

Définir les pôles désirés : Les pôles souhaités doivent être choisis de manière à garantir la stabilité et les performances souhaitées (par exemple, des pôles négatifs réels pour un système fortement amorti).

Calcul de la matrice de gain K : Utiliser la fonction `place` de MATLAB pour déterminer K.

```
1 ``matlab
2 % Définir les matrices A et B du système linéarisé
3 A = [...]; % Matrice A définie ci-dessus
4 B = [...]; % Matrice B définie ci-dessus
5
6 % Définir les pôles désirés
7 desired_poles = [-2 -2.1 -2.2 -2.3 -2.4 -2.5];
8
9 % Calculer la matrice de gain K en utilisant la fonction place
10 K = place(A, B, desired_poles);
11
12 % Afficher la matrice K
13 disp('La matrice de gain K est :');
14 disp(K);
15 ``
```

Application de la Commande

La commande d'entrée U est définie comme :

$$U = -KX$$

Cette commande U permet de maintenir le système à l'équilibre autour de la position stationnaire $X = 0$. En appliquant cette commande, nous pouvons contrôler les angles de roulis, de tangage, et de lacet du drone quadrirotor.

La linéarisation du système et l'application de la commande par placement de pôles permettent de stabiliser le quadrirotor autour de la position stationnaire. La matrice de gain K est déterminée de manière à placer les pôles du système en boucle fermée à des positions spécifiques, assurant ainsi la stabilité et les performances souhaitées du système.

Conclusion

Enfin, Les valeurs obtenues tout au long du projet sont cohérentes avec les équations et les objectifs fixés. Les différentes parties du projet comprennent la simulation des dynamiques du drone, l'ajout du modèle des moteurs à courant continu et l'ajout d'un régulateur PID. Les simulations réalisées avec différentes conditions initiales et paramètres ont permis d'observer le comportement du drone dans différentes situations. Les résultats obtenus correspondent aux attentes et démontrent la validité des modèles et des régulateurs utilisés.

Annexes

Symbol	definition
R	rotation matrix
ϕ	roll angle
θ	pitch angle
ψ	yaw angle
Ω_i	rotor speed
$I_{x,y,z}$	body inertia
J	propeller inertia
b	thrust factor
d	drag factor
l	lever

Figure 1: Définition des symboles utilisées