UNIVERSITÀ
POLITECNICA
DELLE MARCHE

FACOLTÀ DI INGEGNERIA
Corso di Laurea in Ingegneria Informatica e dell'Automazione

TESI DI LAUREA TRIENNALE

Studio e sviluppo di tecniche per il controllo dell'assetto di mini droni

Study and development of techniques for the control of mini drones attitude

Candidato:

Raiola Corrado

Matricola 1081852

Relatore:

Prof. Ippoliti Gianluca

Correlatore:

Prof. Orlando Giuseppe

A.A. 2019/2020

Sommario

Il seguente elaborato ha lo scopo di studiare le leggi matematiche che governano un quadricottero, analizzarne il modello fisico e comprendere la struttura di un sistema a blocchi *Simulink* che ne descriva la dinamica.

Ci si focalizza in particolare sulla dinamica x del sistema implementando un nuovo controllore che permetta al quadricottero di mantenere la posizione di volo stazionario.

La sintesi del controllore è effetuata mediante l'utilizzo della tecnica del luogo delle radici, applicata grazie a *tool* specifici presenti nel software *Matlab*.

Indice

Elenco delle figure

Capitolo 1

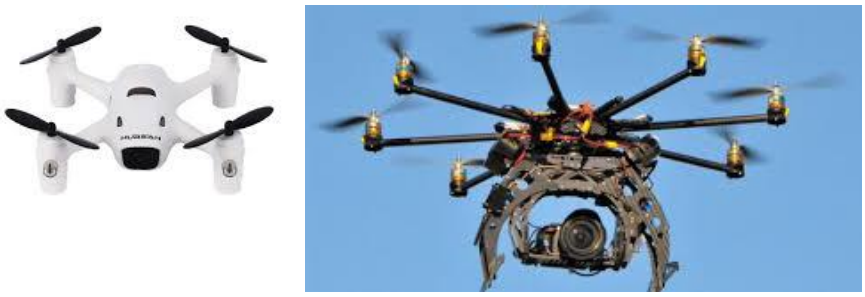
Introduzione

I droni sono **aeromobili a pilotaggio remoto (APR)**, cioè apparecchi che vengono pilotati da un computer di bordo o più semplicemente da un pilota che li guida da remoto con un radiocomando. In linea di massima sono degli oggetti che, mediante l'azione di più motori, hanno la capacità di volare negli spazi aperti e consentono di effettuare diverse tipologie di operazioni sia di tipo professionale che di tipo ricreativo o sportivo, come fare scatti fotografici oppure trasportare piccoli pesi.

Solitamente sono dotati di una struttura realizzata con materiali leggeri, in modo tale che questi possano volare senza alcuna complicazione, e da una batteria che ne alimenta i motori, ovvero che consente alle eliche di potersi muovere e offrire quindi la forza necessaria al drone per spiccare il volo.

Esistono diversi tipi di droni e, grazie al progresso tecnologico, nuove tipologie si svilupperanno nel corso degli anni. A seconda del numero dei motori, e quindi di eliche, i droni si distinguono in tricotteri, quadricotteri, esacotteri, ottocotteri.

Un maggiore numero di motori significa ovviamente più potenza, e permette di portare in volo un peso maggiore, garantendo anche maggiore stabilità in aria, specie in presenza di vento. Non a caso, il quadricottero si è affermato come drone per uso ricreativo, mentre esacotteri e ottocotteri sono droni più utilizzati da professionisti.



(a) Quadricottero (b) Ottocottero

Figura 1.1: Tipologie di droni

Una tipologia che, come i quadricotteri, è utilizzata per scopi puramente ricreativi o di pratica è rappresentata dai mini droni. Questi ultimi hanno una dimensione ridotta rispetto alle altre tipologie, spesso si mantengono su un palmo della mano.

Per l'espletamento del seguente elaborato è stato preso in esame il mini drone dell'azienda francese **Parrot S.A.**, in particolare il modello *Mambo*. Quest'ultima fornisce infatti un pacchetto scaricabile direttamente dal sito della MathWorks (Parrot Minidrones Support from Simulink) contenente un modello Simulink che è possibile caricare direttamene sul dispositivo per poterlo controllare oppure effettuare una simulazione di volo di durata arbitraria.

Grazie a tale modello è stato possibile sostituire al controllore di default della x un nuovo controllore ottenuto mediante una sintesi con il luogo delle radici. Prima di fare ciò, tuttavia, è stato necessario studiare le equazioni fisiche che descrivono la dinamica del quadricottero, analizzare i singoli blocchi che compongono il modello Simulink, isolare la sola dinamica x e ricavarne la funzione di trasferimento.

Figura 1.2: Parrot S.A. mini drone Mambo

1.1 Configurazione di un quadricottero

Un quadricottero è formato da quattro bracci estesi ciascuno dei quali è fornito di un motore elettrico sul quale è attaccato un'elica. Rotori opposti ruotano nello stesso verso. In questo modo quando tutti i rotori ruotano con la stessa velocità angolare, l'accellerazione angolare attorno all'asse perpendicolare al piano immaginario che contiene la struttura del drone è nulla.

Esistono due configurazioni di volo che possono essere adottate da un quadricottero:

- configurazione a più;
- configurazione a croce.

Nella configurazione a più, la variazione dell'assetto del drone si ottiene modificando la velocità rotazionale di due eliche. Diversamente, un quadricottero in configurazione a croce necessita della variazione della velocità angolare di tutti e quattro le eliche, garantendo in questo modo un momento maggiore ed una migliore manovrabilità rispetto alla configurazione precedente.

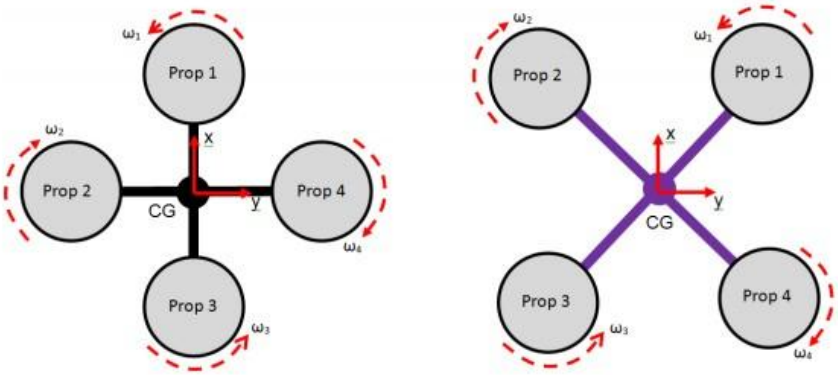


Figura 1.3: Configurazione a più (sinistra) e configurazione a croce (destra)

Il quadricotero è inoltre un corpo rigido sotto-attuato. Esso possiede infatti sei gradi di libertà (6 DoF): tre riguardano la rotazione intorno a tre assi, e sono indicati con *roll* o *rollio* (rotazione intorno all'asse longitudinale del corpo), *pitch* o *beccheggio* (rotazione intorno ad un asse trasversale del corpo) e *yaw* o *imbardata* (rotazione intorno ad un asse verticale passante per il baricentro del corpo). I restanti tre sono dovuti al movimento in tre direzioni: x (movimento lungo la parte frontale del corpo), y (movimento lungo la parte sinistra del corpo) e z (variazione di altitudine).

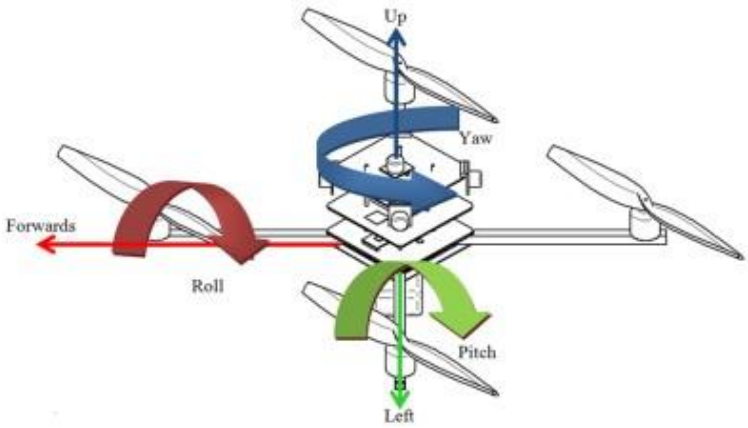


Figura 1.4: Movimenti rispetto agli assi

A seconda del tipo di configurazione adottata dal quadricottero si dovrà agire in modi differenti sui quattro rotori al fine di ottenere i sei movimenti descritti sopra e mostrati in 1.4.

1.1.1 Configurazione a più

In questa configurazione gli assi x e y sono allineati con i bracci del quadricottero (1.3). Cambiando contemporaneamente la velocità rotazionale dei quattro motori dello stesso valore, si ottiene una spinta che genera una **variazione di altitudine.** Per effettuare una **manovra di rollio** bisogna incremenentare la velocità rotazionale del motore 2 e decrementare in contemporanea quella del motore 4. Similmente, incrementando la velocità del motore 1 e decrementando quella del motore 3 si ottiene una **manovra di beccheggio**. Per finire, applicando velocità diverse ad ogni coppia di motori che gira nella stessa direzione, si produce una spinta necessaria per una **manovra di imbardata.** I tipi di manovre sono mostrate in figura 1.5.

1.1.2 Configurazione a croce

In questa configurazione gli assi x e y sono sfasati di $45°$ con i bracci del quadricottero (1.3). Cambiando contemporaneamente la velocità rotazionale dei quattro motori dello stesso valore si ottiene una spinta che genera una **variazione di altitudine**. Per effettuare una **manovra di rollio** bisogna incrementare dello stesso valore la velocità rotazionale dei motori 3 e 4 e decrementare in contemporanea quella dei motori 1 e 2. Similmente, incrementando la velocità rotazionale dei motori 1 e 4 e decrementando quella dei motori 2 e 3 si ottiene una **manovra di beccheggio**. Per finire, applicando velocità diverse alle coppie di motori che girano in versi opposti, si produce una spinta necessaria per una **manovra di imbardata.** I tipi di manovre sono mostrate in figura 1.6.

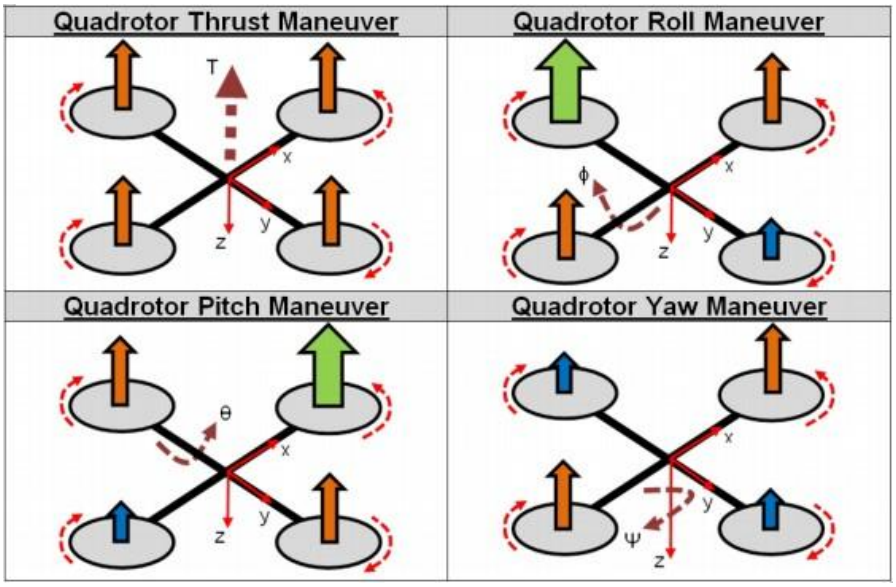


Figura 1.5: Manovre nella configurazione a più

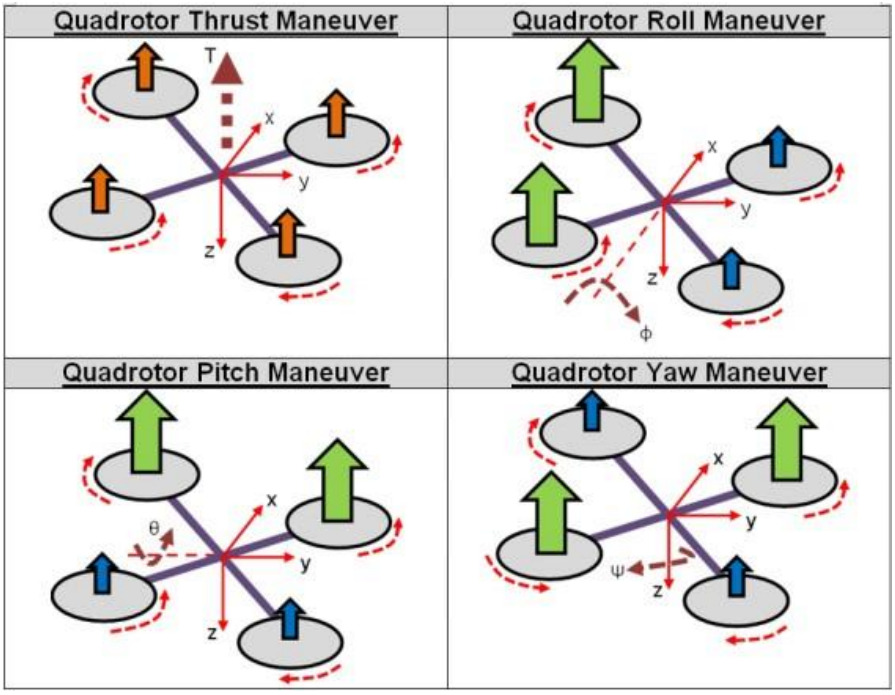


Figura 1.6: Manovre nella configurazione a croce

Capitolo 2

Modello matematico del quadricottero

In questo capitolo viene descritto il processo di derivazione del modello matematico del quadricottero e lo sviluppo delle equazioni del moto con le quali sarà possibile comprendere la struttura del modello a blocchi *Simulink*.

2.1 Sistemi di riferimento di un quadricottero

Prima di mostrare il processo di derivazione del modello matematico del quadricottero è importante definire i sistemi di riferimento necessari per la descrizione del moto rotazionale e traslazionale del drone.

I sistemi di riferimento in questione sono tre e sono mostrati in figura 2.1:

- *Inertial Frame (i)*;
- *Vehicle Frame (v)*;
- *Body Frame (b)*.

2.1.1 Inertial Frame

Il *sistema di riferimento inerziale* è un sistema di coordinate fisso e solidale con l'osservatore. L'asse x del sistema punta a $Nord$, l'asse y punta a $East$ e l'asse z punta verso il centro della Terra.

2.1.2 Vehicle Frame

Il *sistema di coordinate del veicolo* ha la propria origine nel centro di massa del quadricottero. Gli assi di questo sistema di riferimento sono perfettamente allineati con quelli del sistema inerziale e mantengono tale direzione e verso anche durante la rotazione del quadricottero. Infatti questo sistema si occupa di descrivere il moto traslazionale del drone rispetto al sistema di riferimento inerziale.

2.1.3 Body Frame

Questo sistema di riferimento ha la propria origine situata nel centro di massa del quadricottero, esattamente come il *Vehicle Frame*. Tuttavia, a differenza di quest'ultimo, mantiene i tre assi allineati con il corpo del drone, in particolare l'asse x punta verso la parte frontale del corpo, l'asse y punta verso la destra e l'asse z punta verso il basso, in accordo alla regola della mano destra. Tale sistema serve a descrivere il moto rotazionale del quadricottero rispetto al sistema di riferimento del veicolo e mediante gli angoli di yaw, $pitch$ e $roll$.

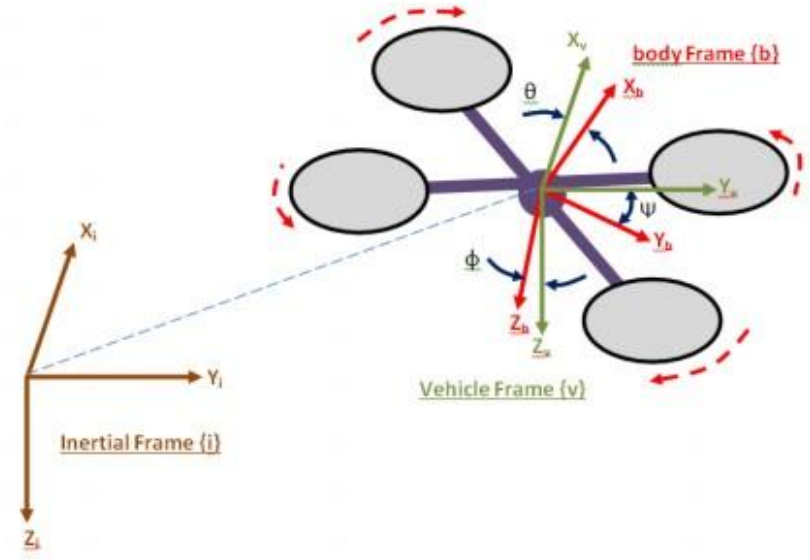


Figura 2.1: Sistemi di riferimento di un quadricottero

2.2 Cinematica di un quadrirotore

Come spiegato nel paragrafo precedente, per poter ricavare il modello matematico di un quadricottero sono necessari tre sistemi di riferimento. Durante il processo di derivazione sarà altresì necessario essere in grado di spostare l'attenzione da un sistema di coordinate all'altro, e per fare ciò occorrerà definire quali relazioni legano l'*Inertial Frame* con il *Body Frame*.

A tal proposito consideriamo i seguenti vettori

$$\mathbf{P}^b = [x^b \;\; y^b \;\; z^b]^T$$
$$\mathbf{P}^i = [x^i \;\; y^i \;\; z^i]^T$$
$$\mathbf{\Lambda}^b = [\phi \;\; \theta \;\; \psi]^T$$

che rappresentano rispettivamente la posizione del quadricottero nel *Body Frame*, la posizione del quadricottero nell'*Inertial Frame* e la rotazione del quadricottero.

Analogamente possiamo definire i vettori relativi alle velocità lineari e angolari nel *Body Frame* come segue

$$\mathbf{V}^b = [u \;\; v \;\; w]^T$$
$$\dot{\mathbf{\Lambda}}^b = [p \;\; q \;\; r]^T.$$

2.2.1 Direct Cosine Matrix (DCM)

Le trasformazioni del moto angolare del quadricottero dal *Body Frame* all'*Inertial Frame* possono essere rappresentate da una matrice di rotazione che prende il nome di *Direct Cosine Matrix (DCM)*. Per determinare la forma della DCM è sufficiente considerare la rotazione totale come composizione di tre rotazioni attorno a tre assi opportuni nel seguente modo. Si esegue una prima rotazione in senso antiorario di un angolo ψ attorno all'asse z. La matrice che esprime tale rotazione è la seguente

$$\mathbf{R}(z,\psi) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.1}$$

La seconda rotazione è di un angolo θ, sempre in senso antiorario, attorno all'asse y, ed è rappresentato dalla matrice $\mathbf{R}(y,\theta)$ data da

$$\mathbf{R}(y,\psi) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}. \tag{2.2}$$

Infine, si esegue una terza rotazione in senso antiorario di un angolo ϕ attorno all'asse x. La matrice che rappresenta tale rotazione è data da

$$\mathbf{R}(x,\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix}. \tag{2.3}$$

La matrice finale che rappresenta la composizione delle tre rotazioni è data da

$$\mathbf{R}_i^b = \mathbf{R}(x,\phi)\mathbf{R}(y,\theta)\mathbf{R}(z,\psi) \tag{2.4}$$

$$\mathbf{R}_i^b = \begin{bmatrix} \cos\theta\cos\psi & \cos\theta\sin\psi & -\sin\theta \\ \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \sin\phi\cos\theta \\ \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi & \cos\phi\sin\theta\sin\psi - sin\phi\cos\psi & \cos\phi\cos\theta \end{bmatrix}. \tag{2.5}$$

Possiamo ora definire l'equazione che lega l'*Inertial Frame* con il *Body Frame* come

$$\mathbf{P}^b = \mathbf{R}_i^b \cdot \mathbf{P}^i \tag{2.6}$$

$$\begin{bmatrix} x^b \\ y^b \\ z^b \end{bmatrix} = \begin{bmatrix} \cos\theta\cos\psi & \cos\theta\sin\psi & -\sin\theta \\ \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \sin\phi\cos\theta \\ \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi & \cos\phi\sin\theta\sin\psi - sin\phi\cos\psi & \cos\phi\cos\theta \end{bmatrix} \begin{bmatrix} x^i \\ y^i \\ z^i \end{bmatrix}. \tag{2.7}$$

Inoltre la DCM è una matrice ortogonale, cioè la sua inversa coincide con la trasposta. Dunque si può esprimere la trasposizione dal **Body Frame** all'**Inertial Frame** calcolando la trasposta della DCM

$$\mathbf{R}_b^i = (\mathbf{R}_i^b)^T = \begin{bmatrix} \cos\theta\cos\psi & \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi \\ \cos\theta\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi \\ -\sin\theta & \sin\phi\cos\theta & \cos\phi\cos\theta \end{bmatrix} \quad (2.8)$$

con la quale si ottiene

$$\mathbf{P}^i = \mathbf{R}_b^i \cdot \mathbf{P}^b \quad (2.9)$$

$$\begin{bmatrix} x^i \\ y^i \\ z^i \end{bmatrix} = \begin{bmatrix} \cos\theta\cos\psi & \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi \\ \cos\theta\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \cos\phi\sin\theta\sin\psi - sin\phi\cos\psi \\ -\sin\theta & \sin\phi\cos\theta & \cos\phi\cos\theta \end{bmatrix} \begin{bmatrix} x^b \\ y^b \\ z^b \end{bmatrix}. \quad (2.10)$$

2.2.2 Matrice di rotazione per le velocità angolari

Le trasformazioni delle velocità angolari dall'*Inertial Frame* al *Body Frame* vengono descritte dalla seguente equazione

$$\boldsymbol{\omega}^b = \mathbf{R}(x,\phi)\mathbf{R}(y,\theta)\mathbf{R}(z,\psi)\begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} + \mathbf{R}(x,\phi)\mathbf{R}(y,\theta)\begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + \mathbf{R}(x,\phi)\begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} \quad (2.11)$$

dove $\boldsymbol{\omega}^b$ rappresenta la velocità angolare nel *Body Frame* e $\mathbf{R}(x,\phi)$, $\mathbf{R}(y,\theta)$ e $\mathbf{R}(z,\psi)$ sono rispettivamente la (2.3), (2.2) e (2.1).

Svolgendo i prodotti fra matrici si ottiene l'esperessione semplificata data da

$$\boldsymbol{\omega}^b = \begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} 1 & 0 & -\sin\theta \\ 0 & \cos\phi & \cos\theta\sin\phi \\ 0 & -\sin\phi & \cos\theta\cos\phi \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \quad (2.12)$$

dalla quale è possibile ricavare l'equazione che descrive le trasformazioni dal sistema di coordinate *Body Frame* all'*Inertial Frame*

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \sin\phi\frac{\sin\theta}{\cos\theta} & \cos\phi\frac{\sin\theta}{\cos\theta} \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi\frac{1}{\cos\theta} & \cos\phi\frac{1}{\cos\theta} \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix}. \quad (2.13)$$

2.3 Dinamica di un quadrirotore

2.3.1 Accelerazione lineare

L'accelerazione lineare, nel sistema di riferimento inerziale, è descritta dalla seconda legge di $Newton$

$$\mathbf{F} = m\dot{\mathbf{V}} \quad (2.14)$$

dove m è la massa del quadricottero e $\mathbf{V}$ è il vettore velocità nel *Body Frame*.

Inoltre osserviamo che le velocità u, v e w sono misurate nel sistema di coordinate *Body Frame* e che il vettore velocità $\mathbf{V}$ può ruotare e cambiare contemporaneamente il suo modulo.

Indichiamo il vettore velocità esplicitandone i versori nel sistema di riferimento del corpo del quadricottero

$$\mathbf{V} = \begin{bmatrix} u \\ v \\ w \end{bmatrix} = u\hat{\mathbf{b}}_1 + v\hat{\mathbf{b}}_2 + w\hat{\mathbf{b}}_3, \tag{2.15}$$

e notiamo che effetuandone la derivata, come richiesto nella (2.14), otteniamo la seguente equazione

$$\frac{d\mathbf{V}}{dt} = \left(\frac{du}{dt}\hat{\mathbf{b}}_1 + \frac{dv}{dt}\hat{\mathbf{b}}_2 + \frac{dw}{dt}\hat{\mathbf{b}}_3\right) + \left(u\frac{d\hat{\mathbf{b}}_1}{dt} + v\frac{d\hat{\mathbf{b}}_2}{dt} + w\frac{d\hat{\mathbf{b}}_3}{dt}\right) \tag{2.16}$$

dove la prima parentesi sta ad indicare la variazione della velocità nel tempo, mentre la seconda parentesi è la variazione della velocità dovuta alla rotazione degli assi, in accordo col *Teorema di Coriolis*. I versori del sistema di coordinate *Body Frame*, infatti, sono dei vettori di modulo unitario che cambiano nel corso del tempo, e pertanto verranno considerati non costanti durante il calcolo della derivata.

Ricordando, inoltre, che per il *Teorema di Poisson* esiste uno e un solo vettore $\boldsymbol{\omega}$ tale per cui vale che

$$\frac{d\hat{\mathbf{b}}_1}{dt} = \boldsymbol{\omega} \times \hat{\mathbf{b}}_1 \tag{2.17}$$

$$\frac{d\hat{\mathbf{b}}_2}{dt} = \boldsymbol{\omega} \times \hat{\mathbf{b}}_2 \tag{2.18}$$

$$\frac{d\hat{\mathbf{b}}_3}{dt} = \boldsymbol{\omega} \times \hat{\mathbf{b}}_3, \tag{2.19}$$

sostituendo la (2.17), (2.18) e (2.19) nella (2.16), possiamo riscrivere la (2.14) nel seguente modo

$$\mathbf{F} = m\dot{\mathbf{V}} + \boldsymbol{\omega} \times m\mathbf{V} \tag{2.20}$$

$$\begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} = m \begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} + m \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times \begin{bmatrix} u \\ v \\ w \end{bmatrix}. \tag{2.21}$$

Svolgendo i prodotti e riorganizzando il tutto si ottiene

$$\begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} = m \begin{bmatrix} \dot{u} + qw - rv \\ \dot{v} + ru - pw \\ \dot{w} + pv - qu \end{bmatrix}. \tag{2.22}$$

Trascurando le forze aerodinamiche, le uniche forze esterne che agiscono sul corpo del quadricottero sono la spinta data dai motori ($\mathbf{T}$) e la forza peso ($\mathbf{W}$). Inoltre, la spinta è sempre attiva lungo l'asse z mentre la forza peso è proiettata secondo l'assetto che assume il quadricottero

$$\begin{bmatrix} W_x \\ W_y \\ W_z - T \end{bmatrix} = m \begin{bmatrix} \dot{u} + qw - rv \\ \dot{v} + ru - pw \\ \dot{w} + pv - qu \end{bmatrix}. \tag{2.23}$$

Effettuando una conversione dal sistema di riferimento inerziale al *Body Frame* tramite la DCM data dalla (2.5) e riorganizzando il tutto si ottiene

$$\mathbf{R}_i^b \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ T \end{bmatrix} = m \begin{bmatrix} \dot{u} + qw - rv \\ \dot{v} + ru - pw \\ \dot{w} + pv - qu \end{bmatrix} \tag{2.24}$$

$$\begin{cases} \dot{u} = rv - qw - g\sin\theta \\ \dot{v} = pw - ru + g\cos\theta\sin\phi \\ \dot{w} = qu - pv + g\cos\phi\cos\theta - \frac{T}{m} \end{cases} . \tag{2.25}$$

Considerando che la spinta data da tutti i motori è proporzionale alla somma dei quadrati delle velocità angolari dei motori

$$T = b(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \tag{2.26}$$

dove b è il coefficiente di spinta e Ω_i è la velocità angolare del motore i-esimo, si ottiene il seguente sistema di equazioni

$$\begin{cases} \dot{u} = rv - qw - g\sin\theta \\ \dot{v} = pw - ru + g\cos\theta\sin\phi \\ \dot{w} = qu - pv + g\cos\phi\cos\theta - \frac{b}{m}(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \end{cases} . \tag{2.27}$$

2.3.2 Accelerazione angolare

Applicando una coppia esterna si andrà a modificare il momento angolare del quadricottero. Considerando poi che il vettore momento angolare può cambiare la propria direzione si ottiene la seguente equazione

$$\mathbf{M} = \dot{\mathbf{H}} + \boldsymbol{\omega} \times \mathbf{H} \tag{2.28}$$

$$\mathbf{H} = I\boldsymbol{\omega} \tag{2.29}$$

dove $\boldsymbol{\omega}$ rappresenta la variazione dell'assetto e I è il momento d'inerzia del quadricottero.

Tenendo in considerazione che un quadricottero è per ipotesi un corpo rigido simmetrico rispetto i suoi piani xz e yz, e che gli assi di rotazione coincidono con gli assi principali, il momento d'inerzia può essere riscritto come segue

$$I = \begin{bmatrix} I_x & 0 & 0 \\ 0 & I_y & 0 \\ 0 & 0 & I_z \end{bmatrix} . \tag{2.30}$$

In base a queste considerazioni si ottiene la seguente equazione

$$\mathbf{M} = I\dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times I\boldsymbol{\omega} \tag{2.31}$$

che, andando ad espandere, fornisce il sistema

$$\begin{cases} M_x = \dot{p}I_x + qr(I_z - I_y) \\ M_y = \dot{q}I_y + pr(I_x - I_z) \\ M_z = \dot{r}I_z + pq(I_y - I_x) \end{cases} . \tag{2.32}$$

Ricordando inoltre la simmetria nei piani xz e yz, si può assumere che $I_x \cong I_y$ e dunque possiamo semplificare la (2.32) come segue

$$\begin{cases} M_x = \dot{p}I_x + qr(I_z - I_y) \\ M_y = \dot{q}I_y + pr(I_x - I_z) \\ M_z = \dot{r}I_z \end{cases} . \tag{2.33}$$

Le coppie esterne sono prodotte dalla spinta e dal trascinamento delle eliche. Trascurando l'inerzia e le coppie aerodinamiche di queste ultime, possiamo riscrivere le coppie esterne nel modo seguente

$$M_x = lb(\Omega_2^2 - \Omega_4^2) \tag{2.34}$$

$$M_y = lb(\Omega_1^2 - \Omega_3^2) \tag{2.35}$$

$$M_z = d(\Omega_2^2 + \Omega_4^2 - \Omega_1^2 - \Omega_3^2) \tag{2.36}$$

dove d è il fattore di trascinamento (*drag factor*) dei motori e l è la distanza tra l'elica e il centro di massa del quadricottero.

Sostituendo la (2.34), (2.35) e (2.36) nella (2.33) e isolando le variabili relative all'accelerazione angolare si ottiene il sistema di equazioni

$$\begin{cases} \dot{p} = \frac{lb}{I_x}(\Omega_2^2 - \Omega_4^2) - qr\frac{I_z - I_y}{I_x} \\ \dot{q} = \frac{lb}{I_y}(\Omega_1^2 - \Omega_3^2) - pr\frac{(I_x - I_z)}{I_y} \\ \dot{r} = \frac{d}{I_z}(\Omega_2^2 + \Omega_4^2 - \Omega_1^2 - \Omega_3^2) \end{cases} . \tag{2.37}$$

2.4 Modello matematico finale

Raccogliamo sotto un unico sistema tutte le equazioni finora ricavate al fine di poter descrivere l'evoluzione temporale del quadricottero attraverso un modello matematico finale.

I sistemi di equazioni (2.27) e (2.37) descrivono rispettivamente le forze e i momenti meccanici applicati al sistema. Ad esse aggiungiamo anche le *coordinate di navigazione*, ricavabili mediante l'ausilio della DCM (2.8) e mostrate qui di seguito

$$\begin{bmatrix} \dot{x}^i \\ \dot{y}^i \\ \dot{z}^i \end{bmatrix} = \mathbf{R}_b^i \begin{bmatrix} u \\ v \\ w \end{bmatrix} \tag{2.38}$$

$$\begin{cases} \dot{x}^i = (\cos\theta\cos\psi)u + (\sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi)v + (\cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi)w \\ \dot{y}^i = (\cos\theta\sin\psi)u + (\sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi)v + (\cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi)w \\ \dot{z}^i = -(\sin\theta)u + (\sin\phi\cos\theta)v + (\cos\phi\cos\theta)w \end{cases} . \tag{2.39}$$

Un ulteriore sistema che mette in relazione le velocità angolari con le derivate degli *angoli di Eulero* è ottenibile dalla relazione (2.13) ed è dato da

$$\begin{cases} \dot{\phi} = p + (q\sin\phi + r\cos\phi)\tan\theta \\ \dot{\theta} = q\cos\phi - r\sin\phi \\ \dot{\psi} = (q\sin\phi + r\cos\phi)\sec\theta \end{cases} . \tag{2.40}$$

Unendo i sistemi (2.27), (2.37), (2.39) e (2.40) si ottiene il modello matematico non lineare finale del quadricottero:

$$\begin{cases} \dot{\phi} = p + (q\sin\phi + r\cos\phi)\tan\theta \\ \dot{\theta} = q\cos\phi - r\sin\phi \\ \dot{\psi} = (q\sin\phi + r\cos\phi)\sec\theta \\ \dot{p} = \frac{lb}{I_x}(\Omega_2^2 - \Omega_4^2) - qr\frac{I_z - I_y}{I_x} \\ \dot{q} = \frac{lb}{I_y}(\Omega_1^2 - \Omega_3^2) - pr\frac{(I_x - I_z)}{I_y} \\ \dot{r} = \frac{d}{I_z}(\Omega_2^2 + \Omega_4^2 - \Omega_1^2 - \Omega_3^2) \\ \dot{x} = (\cos\theta\cos\psi)u + (\sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi)v + (\cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi)w \\ \dot{y} = (\cos\theta\sin\psi)u + (\sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi)v + (\cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi)w \\ \dot{z} = -(\sin\theta)u + (\sin\phi\cos\theta)v + (\cos\phi\cos\theta)w \\ \dot{u} = rv - qw - g\sin\theta \\ \dot{v} = pw - ru + g\cos\theta\sin\phi \\ \dot{w} = qu - pv + g\cos\phi\cos\theta - \frac{b}{m}(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \end{cases} \tag{2.41}$$

Come si può notare il sistema presenta delle componenti non lineari. Sarà necessario procedere con una linearizzazione in un punto di equilibrio al fine di poter applicare delle tecniche di controllo lineare ed effettuare, dunque, la sintesi del controllore.

Capitolo 3

Linearizzazione

Questo capitolo ha lo scopo di fornire i concetti teorici che occorrono per poter linearizzare un sistema non lineare. Attraverso tali strumenti sarà possibile linearizzare il sistema (2.41) e andarne a valutare e analizzare le caratteristiche, soffermandosi in particolar modo sulla dinamica lungo x che è l'oggetto di studio.

3.1 Generalità

Il procedimento di linearizzazione consiste nel descrivere il comportamento di un sistema non lineare attorno ad un punto di equilibrio nominale mediante un particolare sistema lineare, che rappresenterà un approssimazione del sistema originario.

Consideriamo dunque un sistema *multi-ingresso multi-uscita* ($MIMO$) non lineare e stazionario

$$\begin{cases} \dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) \\ \mathbf{y}(t) = \mathbf{g}(\mathbf{x}(t), \mathbf{u}(t)) \end{cases} \tag{3.1}$$

soggetto ad un ingresso costante

$$\mathbf{u}(t) = \bar{\mathbf{u}}, \tag{3.2}$$

e siano $\bar{x}$ e $\bar{y}$ rispettivamente lo stato e l'uscita nominali tali per cui si ha

$$0 = \mathbf{f}(\bar{\mathbf{x}}, \bar{\mathbf{u}}) \tag{3.3}$$

$$\bar{\mathbf{y}} = \mathbf{g}(\bar{\mathbf{x}}, \bar{\mathbf{u}}). \tag{3.4}$$

Possiamo dunque riscrivere l'ingresso, lo stato, l'uscita e lo stato iniziale nel seguente modo

$$\mathbf{u}(t) = \bar{\mathbf{u}} + \delta\mathbf{u}(t) \tag{3.5}$$

$$\mathbf{x}(t) = \bar{\mathbf{x}} + \delta\mathbf{x}(t) \tag{3.6}$$

$$\mathbf{y}(t) = \bar{\mathbf{y}} + \delta\mathbf{y}(t) \tag{3.7}$$

$$\mathbf{x}_{t_0} = \bar{\mathbf{x}} + \delta\mathbf{x}_{t_0} \tag{3.8}$$

dove i primi addendi stanno ad indicare i valori nominali mentre i secondi le variazioni delle variabili d'ingresso, stato, uscita e stato iniziale.

Sostituendo ora i valori delle nuove variabili nella (3.1) si ottiene

$$\begin{cases} \dot{\bar{\mathbf{x}}} + \delta\dot{\mathbf{x}}(t) = \mathbf{f}(\bar{\mathbf{x}} + \delta\mathbf{x}(t), \bar{\mathbf{u}} + \delta\mathbf{u}(t)) \\ \bar{\mathbf{y}} + \delta\mathbf{y}(t) = \mathbf{g}(\bar{\mathbf{x}} + \delta\mathbf{x}(t), \bar{\mathbf{u}} + \delta\mathbf{u}(t)) \end{cases} \tag{3.9}$$

$$\bar{\mathbf{x}} + \delta\mathbf{x}(t_0) = \bar{\mathbf{x}} + \delta\mathbf{x}_{t_0}. \tag{3.10}$$

Supponendo che le funzioni $\mathbf{f}$ e $\mathbf{g}$ siano sufficientemente regolari, si può sviluppare in *serie di Taylor* rispetto a x e u nell'intorno di $\mathbf{x} = \bar{\mathbf{x}}$ e $\mathbf{u} = \bar{\mathbf{u}}$ e troncare al primo ordine. Essendo che $\dot{\bar{\mathbf{x}}} = 0$ si ha

$$\begin{cases} \delta\dot{\mathbf{x}}(t) = \mathbf{f}(\bar{\mathbf{x}}, \bar{\mathbf{u}}) + \left.\frac{\partial \mathbf{f}(\mathbf{x},\mathbf{u})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\bar{\mathbf{x}},\mathbf{u}=\bar{\mathbf{u}}} \delta\mathbf{x}(t) + \left.\frac{\partial \mathbf{f}(\mathbf{x},\mathbf{u})}{\partial \mathbf{u}}\right|_{\mathbf{x}=\bar{\mathbf{x}},\mathbf{u}=\bar{\mathbf{u}}} \delta\mathbf{u}(t) \\ \bar{\mathbf{y}} + \delta\mathbf{y}(t) = \mathbf{g}(\bar{\mathbf{x}}, \bar{\mathbf{u}}) + \left.\frac{\partial \mathbf{g}(\mathbf{x},\mathbf{u})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\bar{\mathbf{x}},\mathbf{u}=\bar{\mathbf{u}}} \delta\mathbf{x}(t) + \left.\frac{\partial \mathbf{g}(\mathbf{x},\mathbf{u})}{\partial \mathbf{u}}\right|_{\mathbf{x}=\bar{\mathbf{x}},\mathbf{u}=\bar{\mathbf{u}}} \delta\mathbf{u}(t) \end{cases} \tag{3.11}$$

Inoltre ricordando la (3.3) e la (3.4) si può semplificare il tutto e ottenere il seguente sistema lineare

$$\begin{cases} \delta\dot{\mathbf{x}}(t) = \left.\frac{\partial \mathbf{f}(\mathbf{x},\mathbf{u})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\bar{\mathbf{x}},\mathbf{u}=\bar{\mathbf{u}}} \delta\mathbf{x}(t) + \left.\frac{\partial \mathbf{f}(\mathbf{x},\mathbf{u})}{\partial \mathbf{u}}\right|_{\mathbf{x}=\bar{\mathbf{x}},\mathbf{u}=\bar{\mathbf{u}}} \delta\mathbf{u}(t) \\ \delta\mathbf{y}(t) = \left.\frac{\partial \mathbf{g}(\mathbf{x},\mathbf{u})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\bar{\mathbf{x}},\mathbf{u}=\bar{\mathbf{u}}} \delta\mathbf{x}(t) + \left.\frac{\partial \mathbf{g}(\mathbf{x},\mathbf{u})}{\partial \mathbf{u}}\right|_{\mathbf{x}=\bar{\mathbf{x}},\mathbf{u}=\bar{\mathbf{u}}} \delta\mathbf{u}(t) \end{cases} \tag{3.12}$$

o più semplicemente

$$\begin{cases} \delta\dot{\mathbf{x}}(t) = \mathbf{A}\,\delta\mathbf{x}(t) + \mathbf{B}\,\delta\mathbf{u}(t) \\ \delta\mathbf{y}(t) = \mathbf{C}\,\delta\mathbf{x}(t) + \mathbf{D}\,\delta\mathbf{u}(t) \end{cases} \tag{3.13}$$

dove lo stato iniziale è stato ricavato direttamente dalla (3.10) ed è pari a

$$\delta\mathbf{x}(t_0) = \delta\mathbf{x}_{t_0}, \tag{3.14}$$

mentre i valori delle matrici $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ e $\mathbf{D}$ sono date dai *Jacobiani* delle funzioni $\mathbf{f}$ e $\mathbf{g}$ rispetto allo stato $\mathbf{x}$ e all'ingresso $\mathbf{u}$

$$\mathbf{A} = \left.\frac{\partial \mathbf{f}(\mathbf{x},\mathbf{u})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\bar{\mathbf{x}},\mathbf{u}=\bar{\mathbf{u}}} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}_{\mathbf{x}=\bar{\mathbf{x}},\mathbf{u}=\bar{\mathbf{u}}} \tag{3.15}$$

$$\mathbf{B} = \left.\frac{\partial \mathbf{f}(\mathbf{x},\mathbf{u})}{\partial \mathbf{u}}\right|_{\mathbf{x}=\bar{\mathbf{x}},\mathbf{u}=\bar{\mathbf{u}}} = \begin{bmatrix} \frac{\partial f_1}{\partial u_1} & \cdots & \frac{\partial f_1}{\partial u_p} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial u_1} & \cdots & \frac{\partial f_n}{\partial u_p} \end{bmatrix}_{\mathbf{x}=\bar{\mathbf{x}},\mathbf{u}=\bar{\mathbf{u}}} \tag{3.16}$$

$$\mathbf{C} = \left.\frac{\partial \mathbf{g}(\mathbf{x},\mathbf{u})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\bar{\mathbf{x}},\mathbf{u}=\bar{\mathbf{u}}} = \begin{bmatrix} \frac{\partial g_1}{\partial x_1} & \cdots & \frac{\partial g_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial g_q}{\partial x_1} & \cdots & \frac{\partial g_q}{\partial x_n} \end{bmatrix}_{\mathbf{x}=\bar{\mathbf{x}},\mathbf{u}=\bar{\mathbf{u}}} \tag{3.17}$$

$$\mathbf{D} = \left.\frac{\partial \mathbf{g}(\mathbf{x},\mathbf{u})}{\partial \mathbf{u}}\right|_{\mathbf{x}=\bar{\mathbf{x}},\mathbf{u}=\bar{\mathbf{u}}} = \begin{bmatrix} \frac{\partial g_1}{\partial u_1} & \cdots & \frac{\partial g_1}{\partial u_p} \\ \vdots & \ddots & \vdots \\ \frac{\partial g_q}{\partial u_1} & \cdots & \frac{\partial g_q}{\partial u_p} \end{bmatrix}_{\mathbf{x}=\bar{\mathbf{x}},\mathbf{u}=\bar{\mathbf{u}}} . \tag{3.18}$$

Questo nuovo sistema descrive in maniera approssimata il comportamento del sistema non lineare attorno al particolare equilibrio considerato nel caso in cui le variazioni delle funzioni di ingresso e dello stato iniziale $\delta\mathbf{u}(t)$ e $\delta\mathbf{x}_{t_0}$, nonché le variazioni $\delta\mathbf{x}(t)$ e $\delta\mathbf{y}(t)$ da esse provocate, siano sufficientemente piccole in norma (valida approssimazione derivante dall'aver trascurato i termini di ordine superiore al primo nello sviluppo in serie di Taylor).

3.2 Linearizzazione del sistema quadricottero

Posto le basi relative ai concetti teorici della linearizzazione, vogliamo adesso ottenere un modello matematico lineare che approssimi quello del quadricottero. Questo è importante in quanto la tecnica di sintesi che si vuole sviluppare in questo progetto è una tecnica di controllo lineare e, in quanto tale, necessita di un sistema linearizzato.

Come prima cosa, consideriamo il sistema (2.41) ed eseguiamone una prima approssimazione per *piccole oscillazioni*. Ipotizzando, infatti, che le variazioni degli angoli di Eulero siano relativamente piccole, le funzioni *seno* e *coseno* possono essere approssimate nel modo seguente

$$\begin{cases} \sin\alpha \cong \alpha \\ \cos\alpha \cong 1 \end{cases} . \tag{3.19}$$

Si ottiene dunque un nuovo sistema dato da

$$\begin{cases} \dot{\phi} \cong p + q\phi\theta + r\theta \\ \dot{\theta} \cong q - r\phi \\ \dot{\psi} \cong q\phi + r \\ \dot{p} \cong \frac{lb}{I_x}(\Omega_2^2 - \Omega_4^2) - qr\frac{I_z - I_y}{I_x} \\ \dot{q} \cong \frac{lb}{I_y}(\Omega_1^2 - \Omega_3^2) - pr\frac{(I_x - I_z)}{I_y} \\ \dot{r} \cong \frac{d}{I_z}(\Omega_2^2 + \Omega_4^2 - \Omega_1^2 - \Omega_3^2) \\ \dot{x} \cong u + (\phi\theta - \psi)v + (\theta + \phi\psi)w \\ \dot{y} \cong \psi u + (\phi\theta\psi + 1)v + (\theta\psi - \phi)w \\ \dot{z} \cong -\theta u + \phi v + w \\ \dot{u} \cong rv - qw - g\theta \\ \dot{v} \cong pw - ru + g\phi \\ \dot{w} \cong qu - pv + g - \frac{b}{m}(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \end{cases} . \tag{3.20}$$

3.2.1 Modello in spazio di stato

Dopo aver effettuato una prima approssimazione, si passa alla costruzione del modello in spazio di stato. Consideriamo dunque il seguente vettore di stato $\mathbf{x} = [\phi \ \ \theta \ \ \psi \ \ p \ \ q \ \ r \ \ x \ \ y \ \ z \ \ u \ \ v \ \ w]^T$, e indichiamone le componenti come segue

$$\begin{aligned} x_1 &= \phi \\ x_2 &= \theta \\ x_3 &= \psi \\ x_4 &= \dot{x_1} \\ x_5 &= \dot{x_2} \\ x_6 &= \dot{x_3} \\ x_7 &= x \\ x_8 &= y \\ x_9 &= z \\ x_{10} &= \dot{x_7} \\ x_{11} &= \dot{x_8} \\ x_{12} &= \dot{x_9} \end{aligned} \tag{3.21}$$

Riscriviamo poi il sistema (3.20) in maniera tale da ottenere la funzione vettoriale di stato

$$\begin{cases} \dot{x_1} = x_4 \\ \dot{x_2} = x_5 \\ \dot{x_3} = x_6 \\ \dot{x_4} = \frac{1}{I_x} U_2 - x_5 x_6 \frac{I_z - I_y}{I_x} \\ \dot{x_5} = \frac{1}{I_y} U_3 - x_4 x_6 \frac{(I_x - I_z)}{I_y} \\ \dot{x_6} = \frac{1}{I_z} U_4 \\ \dot{x_7} = x_{10} \\ \dot{x_8} = x_{11} \\ \dot{x_9} = x_{12} \\ \dot{x_{10}} = x_6 x_{11} - x_5 x_{12} - g x_2 \\ \dot{x_{11}} = x_4 x_{12} - x_6 x_{10} + g x_1 \\ \dot{x_{12}} = x_5 x_{10} - x_4 x_{11} + g - \frac{1}{m} U_1 \end{cases} \tag{3.22}$$

dove si è indicato con

$$\begin{cases} U_1 = b(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \\ U_2 = lb(\Omega_2^2 - \Omega_4^2) \\ U_3 = lb(\Omega_1^2 - \Omega_3^2) \\ U_4 = d(\Omega_2^2 + \Omega_4^2 - \Omega_1^2 - \Omega_3^2) \end{cases} \tag{3.23}$$

gli ingressi forniti al sistema e che rappresentano rispettivamente la spinta data dai motori per poter ottenere una variazione dell'altitudine, la coppia di rollio, la coppia di beccheggio e la coppia di imbardata.

A questo punto consideriamo un punto di equilibrio per il sistema per poter procedere con la linearizzazione. Il punto in questione sarà quello relativo alla posizione in *hover* del drone,

cioè la condizione di volo stazionario, nella quale il quadricottero si troverà fermo in volo ad una certa quota costante da terra. Per ottenere tale condizione basterà applicare un ingresso costante siffatto

$$\bar{\mathbf{u}} = \begin{bmatrix} mg & 0 & 0 & 0 \end{bmatrix}^T . \tag{3.24}$$

Infatti, in tale posizione, la spinta data dai motori dovrà essere tale da reggere il carico verticale dovuto al peso del quadricottero. Da qui si ricava un punto di equilibrio tale per cui sia soddisfatta la relazione (3.3)

$$\bar{\mathbf{x}} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & \bar{x} & \bar{y} & \bar{z} & 0 & 0 & 0 \end{bmatrix}^T . \tag{3.25}$$

Sfruttando le relazioni (3.15), (3.16), (3.17) e (3.18) sarà possibile calcolare la matrice della dinamica $\mathbf{A}$, la matrice degli ingressi $\mathbf{B}$ e le matrici per l'uscita $\mathbf{C}$ e $\mathbf{D}$ che formeranno il sistema linearizzato del quadricottero.

Concentrandosi esclusivamente sulla dinamica lungo x, quindi considerando solo le funzioni di stato relative alla velocità e all'accelerazione di x, si ha

$$\begin{cases} \delta\dot{x} = \delta u \\ \delta\ddot{u} = -g\delta\theta \end{cases} . \tag{3.26}$$

Da questa prima analisi si può osservare come la dinamica lungo x dipenda esclusivamente dal valore che assume l'angolo di beccheggio, dato che il quadricottero è un sistema *sotto-attuato*.

Inoltre si evince che è impossibile controllare in maniera diretta la dinamica x del drone in quanto non si ha nessuna dipendenza dagli ingressi.

Capitolo 4

Analisi del modello *Simulink*

Una parte fondamentale di questo lavoro è sicuramente rappresentata dall'analisi del modello *Simulink* del *mini-drone "Mambo"* dell'azienda Francese *Parrot S.A.*, che è possibile scaricare direttamente dal sito della *MathWorks* al seguente indirizzo `https://it.mathworks.com/hardware-support/parrot-minidrones.html`.

Il pacchetto, infatti, permette di sviluppare e simulare algoritmi per il controllo del volo del minidrone andando a sostituire un nuovo controllore a quello di default.

Una volta ottenuto il pacchetto, sarà sufficiente impartire il comando *asbQuadcopterStart* nella *CommandWindows* di Matlab per poter iniziare un nuovo progetto. All'apertura di quest'ultimo ci si ritrova sulla schermata il modello *Simulink* del minidrone (Figura 4.1).

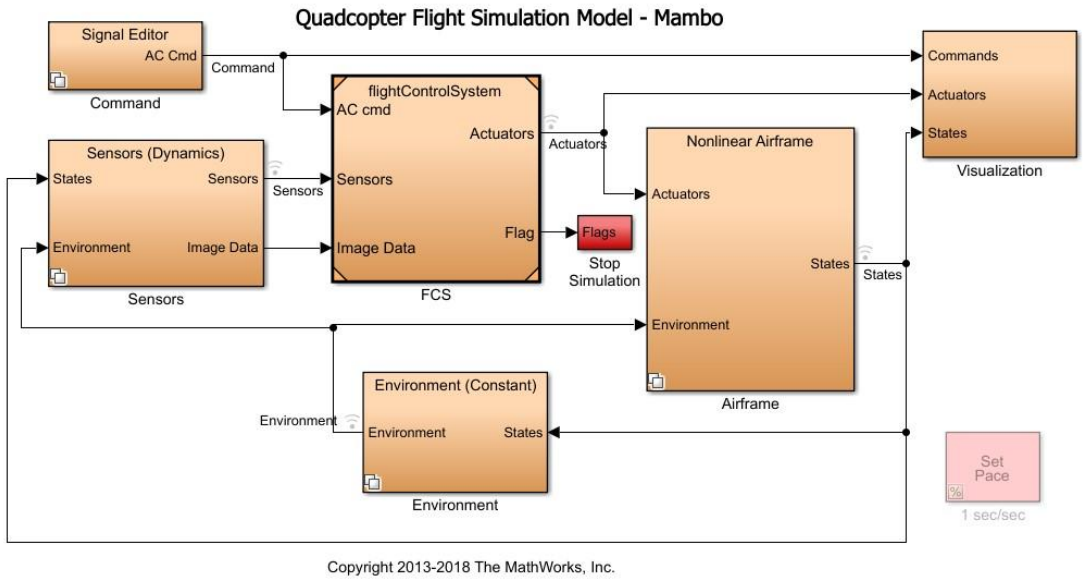


Figura 4.1: Modello *Simulink* del minidrone Mambo

Come si può osservare, il modello presenta la tipica struttura di un sistema a controreazione, ed è costituito da sei blocchi principali:

- ***Command***: il seguente blocco si occupa della costruzione dei segnali di riferimento che verranno utilizzati dal controllore per poter mantenere il drone nella posizione di volo stazionario.
- ***Sensor***: il seguente blocco contiene tutte le costanti di trasduzione e le relazioni tra i diversi sensori presenti nel drone della Parrot. In particolare si occupa di fornire in ingresso al blocco di controllo i dati elaborati dai sensori, tenendo in considerazione le costanti d'ambiente presenti nel blocco Environment.
- ***Environment***: il blocco Environment costruisce un vettore contenente tutte le costanti ambientali che agiscono durante il volo del drone, come ad esempio:
  - la costante di gravità;
  - il campo magnetico;
  - le costanti atmosferiche come pressione, velocità del suono, densità e temperatura dell'aria.

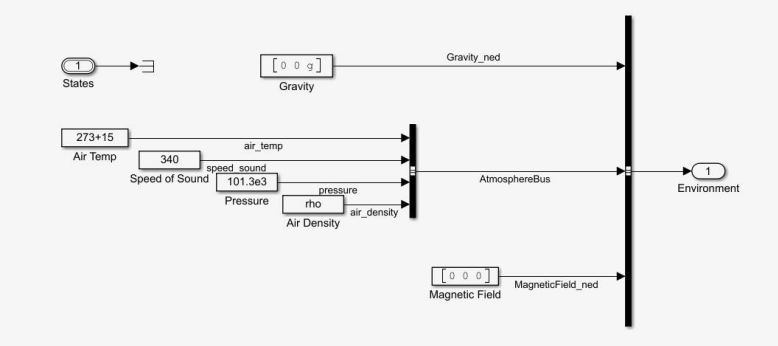


Figura 4.2: Apertura blocco *Environment*

- ***Flight Control System (FCS)***: il blocco FCS (Flight Control System) rappresenta il blocco di controllo all'interno del quale sarà possibile trovare i controllori di default per ogni grado di libertà.
- ***Airframe***: il blocco Airframe contiene il modello che descrive il drone e rappresenta il processo del sistema a blocchi.
- ***Visualization***: Con il seguente blocco si ha la possibilità di visualizzare le variabili di stato del drone e le velocità angolari degli attuatori al fine di monitorare l'andamento della sessione di volo durante la simulazione. Permette inoltre di visualizzare, scegliendo opportunamente il valore della variabile *VSS_VISUALIZATION*, il volo in 3D del drone.

All'apertura del progetto, dopo la fase di creazione, saranno caricate in automatico diverse strutture dati e variabili, le quali potranno essere opportunamente modificate al fine di ottenere un diverso settaggio del modello. Ad esempio sarà possibile scegliere se simulare il volo del quadricottero su di un sistema non lineare oppure su di una approssimazione lineare di quest'ultimo.

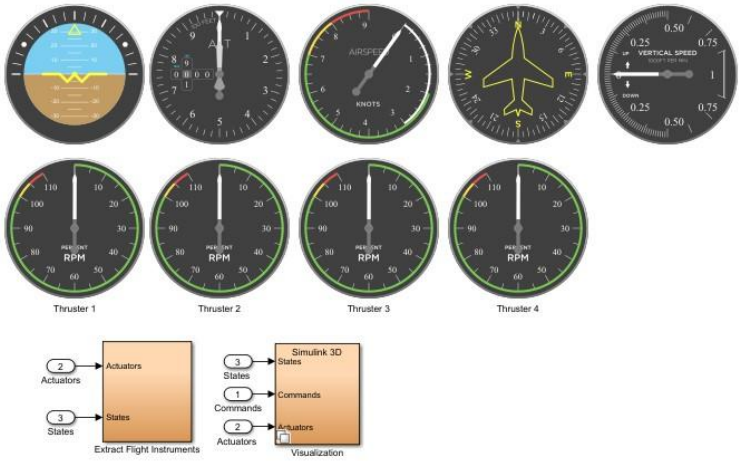


Figura 4.3: Apertura blocco *Visualization*

La selezione del sistema lineare, risulterà necessaria ai fini del ricavo della funzione di trasferimento della variabile x, la quale permetterà l'applicazione di tecniche di controllo per la sintesi di un nuovo controllore.

Nei paragrafi successivi si analizzeranno più nel dettaglio i blocchi più importanti del modello *Simulink*.

4.1 Command

Il blocco *Command* permette la creazione dei segnali che verranno inviati direttamente nella sezione di controllo ed utilizzati come riferimento per l'orientazione e la posizione del drone. Accedendo a tale blocco si potrà osservare la presenza di quattro $sotto - blocchi$, di cui solo uno selezionabile per simulazione. Modificando il valore della variabile *VSS_COMMAND*, sarà possibile scegliere il $sotto - blocco$ di riferimento che si intende utilizzare:

- ***VSS_COMMAND = 0*** selezionerà il *Signal Editor* per la generazione di segnali di default;
- ***VSS_COMMAND = 1*** selezionerà il *Joystick* per il controllo manuale del drone;
- ***VSS_COMMAND = 2*** selezionerà il *Data* per la selezione dei segnali direttamente da un file *.mat*;
- ***VSS_COMMAND = 3*** selezionerà lo *Spreadsheet Data* per la selezione dei segnali da un foglio di calcolo.

Accedendo al blocco *Signal Editor* (Figura 4.4), che è quello selezionato di default, si può osservare che l'uscita *AC Cmd* raccoglie diversi valori tra cui la posizione e l'orientazione di riferimento forniti dal blocco *Position/Attitude Reference*. Da quest'ultimo si osserva anche che i valori del *pitch* e del *roll* vengono forniti in ingresso ad una porta AND, la quale deciderà quale modalità di controllo attivare. Se i valori degli angoli saranno entrambi nulli verrà selezionata la modalità di controllo della posizione. Al contrario, se almeno uno dei valori è diverso da zero sarà selezionata la modalità di controllo dell'orientazione.

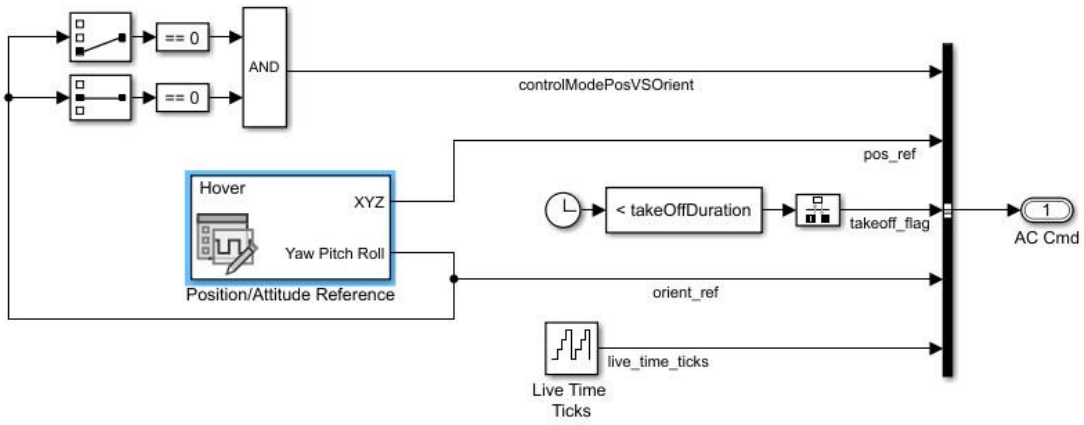


Figura 4.4: Blocco *Signal Editor*

4.2 Sensor

Il blocco *Sensor* riceve in ingresso i valori delle variabili di stato direttamente dall'anello di retroazione che, insieme alle costanti ambientali, gli permettono di elaborare i dati che saranno poi necessari al sistema di controllo durante la fase di comparazione con i segnali riferimento.

Cambiando opportunamente il valore della variabile *VSS_SENSOR* presente nel *Workspace* di *Matlab* si potranno selezionare i seguenti sottoblocchi:

- ***Sensors (Feedthrough)*** $\rightarrow$ $VSS\_SENSOR = 0$;
- ***Sensors (Dynamics)*** $\rightarrow$ $VSS\_SENSOR = 1$.

Selezionando il primo si avrà la possibilità di non considerare i disturbi durante la fase di elaborazione dei dati. Al contrario, scegliendo il secondo, si selezioneranno i sensori dinamici che terranno conto anche dei rumori, corretti successivamente in fase di controllo.

All'interno di quest'ultimo, che rappresenta il blocco di default, si hanno due differenti $sotto-blocchi$ di cui uno si occupa di elaborare le immagini della fotocamera (*Camera*), mentre il secondo, *IMU_Pressure*, elabora i dati provenienti dal sensore *IMU*.

L'IMU, acronimo di *Inertial Measurement Unit*, è un dispositivo di misura che trova utilizzo nelle applicazioni embedded per fornire informazioni riguardanti posizione e direzione di un oggetto nello spazio tridimensionale. Può essere utilizzato negli autoveicoli, ad esempio per il controllo elettronico della velocità, oppure generalmente nel campo dell'aereonautica e nautica. Nel nostro caso è stato montato su un quadrirotore per fornire i dati necessari per il calcolo degli angoli di roll, pitch e yaw necessari per monitorare la dinamica del drone stesso.

Per fornire tali valori l'IMU fa utilizzo di componenti integrati MEMS (Micro Electro-Mechanical System) ovvero sensori che permettono di trasformare un input meccanico, come ad esempio una pressione oppure un'accelerazione, in un impulso elettrico quantificabile e misurabile. Tali sensori sono:

- **Giroscopio:** dispositivo in grado di misurare il moto rotazionale;
- **Accelerometro:** dispositivo in grado di misurare l'accelerazione;
- **Magnetometro:** dispositivo che misura il campo magnetico;

- **Barometro:** dispositivo di pressione, il quale potrebbe essere utilizzato per un'eventuale misura dell'altezza.

Aprendo il blocco *IMU_Pressure* si può notare come le variabili di stato come l'accelerazione lineare e angolare relative al *Body Frame*, vengano utilizzate, insieme al centro di gravità (CG) e al vettore di gravità, opportunamente convertito tramite la DCM, per poter ottenere le relative accelerazioni nel sistema di riferimento inerziale, che saranno utilizzate successivamente (Figura 4.5).

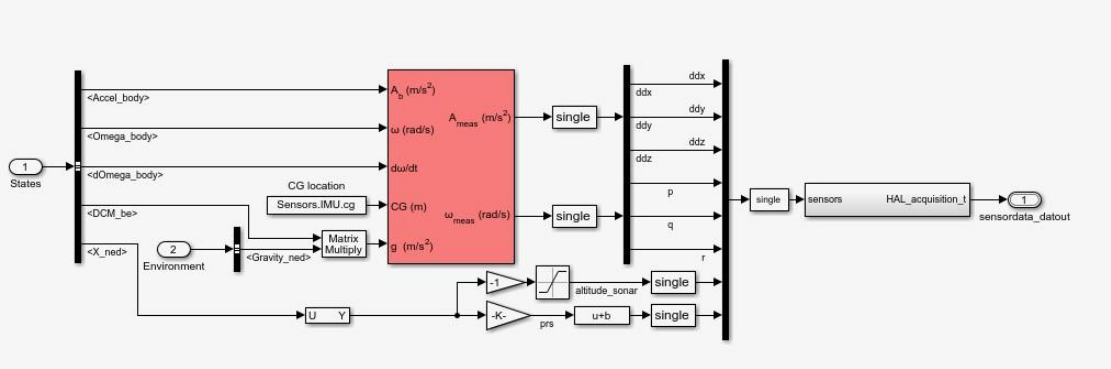


Figura 4.5: Blocco *IMU_Pressure*

4.3 Airframe

Il blocco *Airframe* descrive il modello matematico del quadricottero e rappresenta il processo del sistema a blocchi. Alla sua apertura possiamo notare la presenza di due $sotto-blocchi$ (Figura 4.6):

- **Linear Airframe**;
- **NonLinear Airframe.**

Il primo rappresenta un approssimazione lineare del sistema reale, mentre il secondo descrive il sistema non lineare del drone.

Modificando il valore della variabile *VSS_VEHICLE* a 0 si seleziona il modello lineare, altrimenti, impostandone il valore ad 1, si attiverà il modello non lineare.

Di seguito si analizzerà solo il $sotto-blocco$ inerente al processo non lineare (selezionato di default), mentre il blocco relativo al modello linearizzato, verrà approfondito nel capitolo successivo dove giocherà un ruolo fondamentale per il ricavo della funzione di trasferimento della dinamica lungo x.

Il blocco *NonLinear Airframe* si compone di tre $sotto-blocchi$ fondamentali:

- **AC model:** si occupa della costruzione e del calcolo delle equazioni della dinamica del drone, fornendo in uscita le forze e i momenti meccanici calcolati mediante la spinta fornita dagli attuatori, le variabili d'ambiente e le velocità lineari e angolari del *Body Frame* fornite come ingresso;

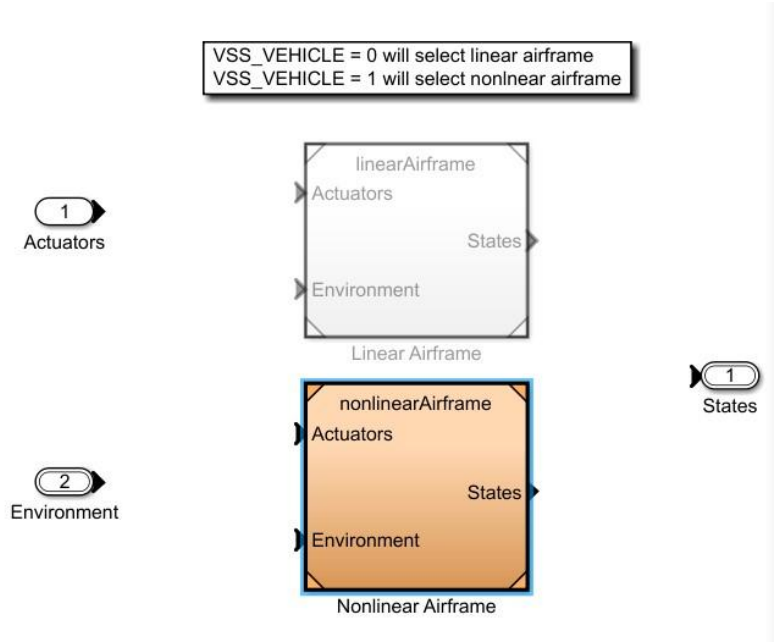


Figura 4.6: Selezione del modello

- **6DOF (Quaternion):** blocco *Matlab* che si occupa di fornire le principali variabili di stato come le velocità, le posizioni e le accelerazioni sia del sistema di riferimento inerziale che di quello solidale al body del drone. Il tutto viene calcolato mediante l'approccio dei quaternioni, che offre una maggiore precisione a costo di una maggiore complessità;
- **Bus setup:** si occupa di riordinare in un unico vettore le seguenti variabili che formeranno il vettore di stato:
  - *LLA* → latitudine, longitudine e altitudine;
  - *V_ned* → velocità lineare nel sistema inerziale;
  - *X_ned* → vettore posizione nel sistema inerziale;
  - *Euler* → angoli di Eulero;
  - *DCM_be* → matrice di rotazione per il passaggio dal *Body Frame* all'*Inertial Frame*;
  - *V_b* → velocità lineare nel *Body Frame*;
  - *Omega_b* → velocità angolare nel *Body Frame*,
  - *dOmega_b* → accelerazione angolare nel *Body Frame*;
  - *Accel_b* → accelerazione lineare nell'*Inertial Frame*.

Il blocco *AC model* è a sua volta descritto da quattro $sotto-blocchi$ (Figura 4.7). Il primo è il ***Gravity Force Calculation*** che si occupa del calcolo della forza di gravità rispetto al sistema solidale al drone. A seguire si ha il ***Drag Calculation*** che calcola la forza di attrito viscoso che esercita l'aria quando impatta sul drone, il ***Motor Forces and Torques*** che permette di calcolare le forze e i momenti meccanici ed infine l'***Applied Force Calculation*** che ha il compito di unire la forza di gravità, la forza aerodinamica e la forza dei motori al fine di ottenere come risultante le forze esterne.

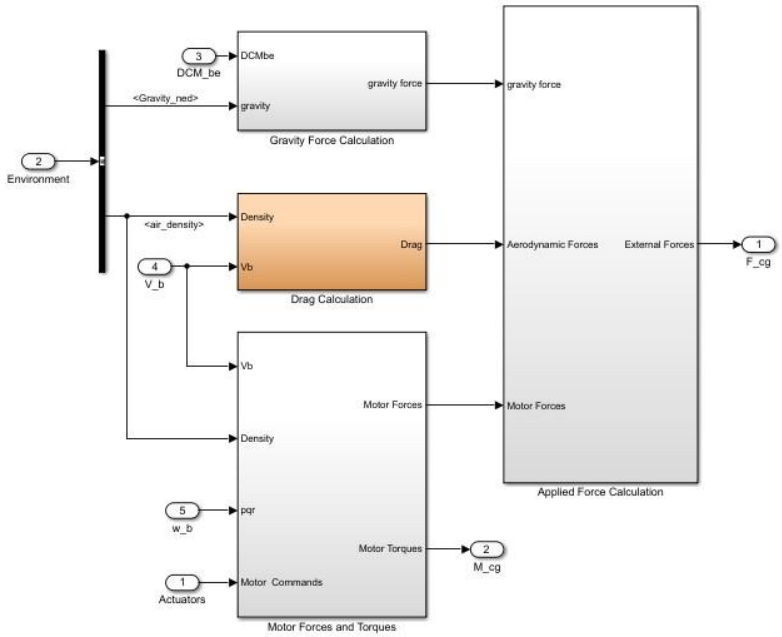


Figura 4.7: Blocco *AC_model*

4.3.1 Forza di gravità

Sappiamo che la forza di gravità punta sempre verso il centro della Terra. Quando però il drone effettua delle manovre di rollio o di beccheggio, il sistema di riferimento solidale al corpo del quadricottero e il sistema di riferimento inerziale perdono la coincidenza lungo l'asse z, cioè gli assi relativi alla variazione di altitudine di entrambi i sistemi di coordinate non sono più paralleli l'uno con l'altro. Dunque si effettua una trasformazione di coordinate dal sistema di riferimento inerziale a quello solidale con il corpo del drone facendo utilizzo della matrice di rotazione DCM mostrata in (2.5) e in accordo alla (2.24).

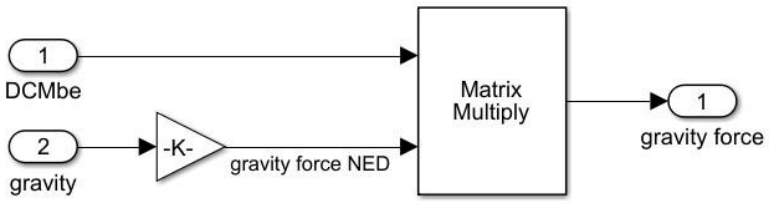


Figura 4.8: Calcolo della forza di gravità

4.3.2 Forza Aerodinamica

Nei capitoli relativi al processo di derivazione delle equazioni fisico-matematiche che compongono il modello del quadricottero, si è voluto trascurare l'effetto della resistenza dell'aria sul drone dato che, per il seguente progetto, si considerano piccoli spostamenti attorno al punto di equilibrio nominale. Tuttavia le forze aerodinamiche sono presenti nel modello *Simulink* e agiscono secondo la seguente relazione

$$\vec{F}_D = \frac{1}{2}\rho C_d A_d \vec{V}_b |\vec{V}_b| \tag{4.1}$$

dove ρ è la densità dell'aria, $\vec{V}_b$ è la velocità nel sistema solidale al corpo del drone, A_d è la superficie che impatta il fluido e C_D è il coefficiente di *drag* (indicato precdentemente con d).

Nel modello *Simulink* si va a calcolare la pressione dell'aria (fornita dal valore della velocità e della densità dell'aria) per poi moltiplicarla alla superficie del drone che impatta il fluido e al coefficiente di drag (notare la corrispondenza tra la (4.1) e la Figura 4.9).

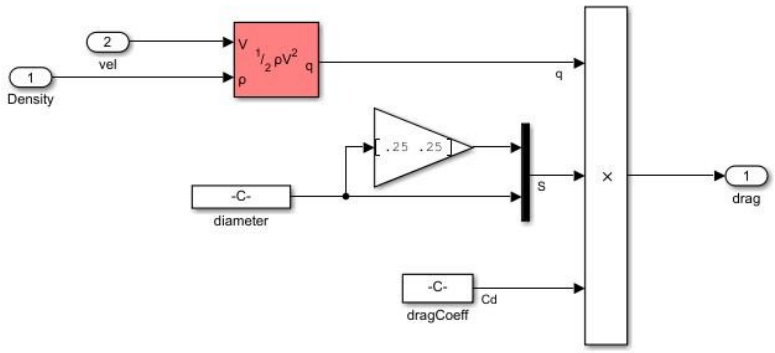


Figura 4.9: Calcolo dell'attrito viscoso

In seguito si moltiplica il risultato per -1 essendo $\vec{F}_D$ una forza di attrito viscoso che si oppone al movimento del drone.

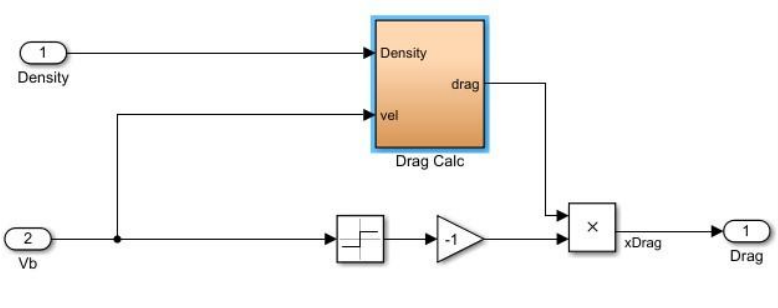


Figura 4.10: Calcolo della forza aerodinamica

4.3.3 Forze e momenti dei motori

Il blocco *Motor Forces and Torques* è presentato in figura 4.11 e mostra al suo interno due blocchi fondamentali:

- *MotorsToW*;
- *For Each Subsystem.*

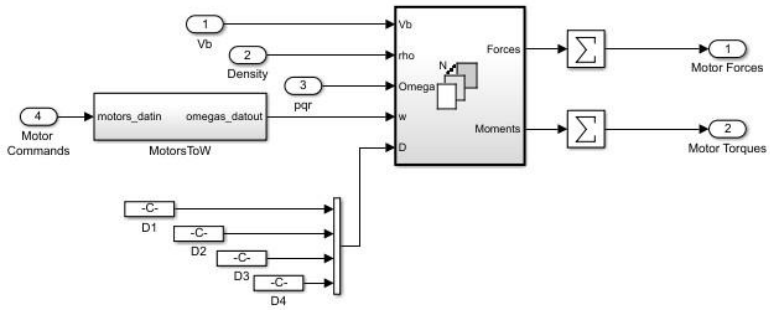


Figura 4.11: Blocco *Motor Forces and Torques*

Il primo si occupa di trasformare il valore degli attuatori fornitogli in ingresso in un segnale interpretabile dai motori.

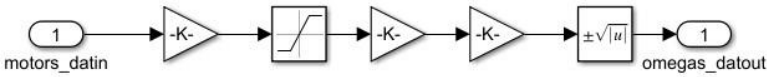


Figura 4.12: Blocco *MotorsToW*

Il secondo, invece, rappresenta un operazione ciclica che riceve in ingresso il valore delle velocità lineari e angolari, della densità dell'aria, l'uscita del primo blocco e il vettore D, che descrive il posizionamento rispetto al centro di massa dei rotori. In uscita, invece, fornisce i valori delle forze e dei momenti.

Esso è mostrato in figura 4.13.

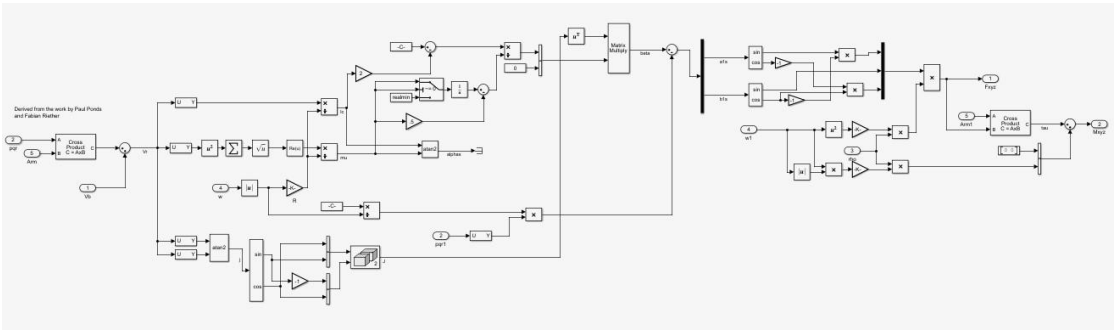


Figura 4.13: Dinamica dei rotori

Diamo ora uno sguardo più attento ai *sotto – blocchi* che compongono l'intero sistema.

Un fenomeno tipico che si verifica durante il volo di un quadricottero è il cosiddetto *Blade Flapping*. Quando i rotori traslano orizzontalmente si viene a generare una differenza tra il sollevamento delle lame anteriori e quelle posteriori, provocando un inclinazione errata dei motori del veicolo che si traduce in una perdita di stabilità. Il $flapping$ del rotore, può essere descritto calcolando l'ampiezza e la direzione della traslazione del rotore e definendo un sistema di riferimento locale, B_i, allineato in quella specifica direzione. Come prima cosa bisogna calcolare l'*advance ratio* e le direzioni azimutali del rotore, che derivano dalle seguenti relazioni:

$$\vec{v} = \vec{v}_b + \vec{\omega}_b \times \vec{d}_i \tag{4.2}$$

$$\mu_i = \frac{\|\vec{v}_{(1,2)}\|}{\Omega_{ir}} \tag{4.3}$$

$$\eta = \arctan\left(\frac{v_2}{v_1}\right) \tag{4.4}$$

dove $\vec{v}$ rappresenta il vettore velocità dei rotori, μ_i è l'*advance ratio* dell'i-esimo rotore e η è la direzione azimutale del moto. Il vettore $\vec{d}_i$ rappresenta, invece, il posizionamento dei rotori rispetto al centro di massa del veicolo, e si può scrivere come:

$$\vec{d}_N = \begin{bmatrix} 0 & d & h \end{bmatrix}^T \tag{4.5}$$

$$\vec{d}_S = \begin{bmatrix} 0 & -d & h \end{bmatrix}^T \tag{4.6}$$

$$\vec{d}_E = \begin{bmatrix} d & 0 & h \end{bmatrix}^T \tag{4.7}$$

$$\vec{d}_W = \begin{bmatrix} -d & 0 & h \end{bmatrix}^T \tag{4.8}$$

dove d è la lunghezza del braccio del veicolo, h è l'altezza dei rotori rispetto al centro di massa e $(NSEW)$ rappresentano le direzioni cardinali dei rotori.

Il calcolo delle componenti del vettore velocità del veicolo e dell'*advance ratio* è mostrato rispettivamente nelle figure 4.14 e 4.15.

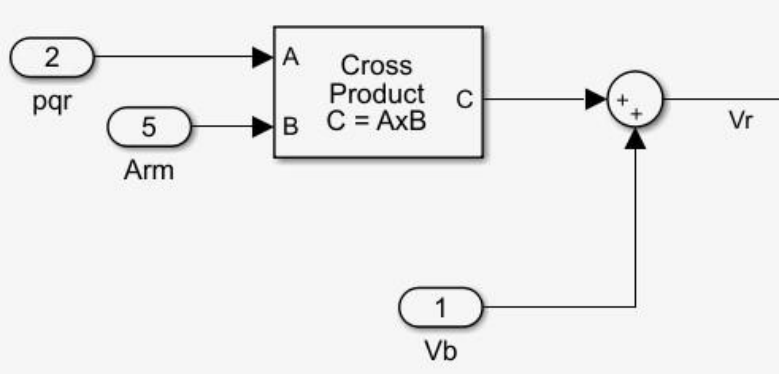


Figura 4.14: Calcolo delle componenti della velocità

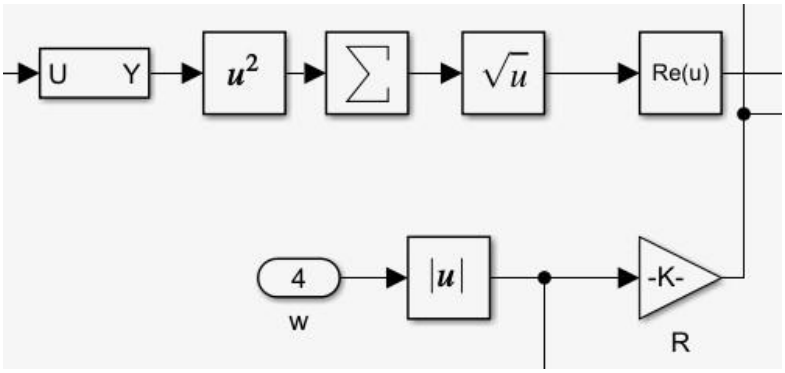


Figura 4.15: Calcolo dell' *advance ratio*

Gli angoli di *flapping* laterale e longitudinale dell'i-esimo rotore calcolato nel sistema di riferimento locale B_i sono:

$$u_{1_s i} = \frac{1}{1 - \frac{\mu_i^2}{2}} \mu_i (4\theta_t - 2\lambda_{hi}^2) \tag{4.9}$$

$$v_{1_s i} = \frac{1}{1 - \frac{\mu_i^2}{2}} \left(\frac{C_T}{\sigma} \frac{8}{9} \frac{\mu_i \gamma}{a} + \frac{C_T}{2\mu_i} 2\lambda_{hi}^2 \right) \tag{4.10}$$

dove λ_{hi} rappresenta l'*inflow ratio* approssimabile come $\lambda_{hi} = \sqrt{\frac{C_T}{2}}$,$\gamma$ è il *Lock Number* il cui valore è $\gamma = \frac{\rho a_0 c r^4}{I_b}$, θ_t rappresenta l'angolo di pitch rispetto alla direzione di traslazione e a descrive l'angolo di salita.

Per ottenere gli angoli di flapping nel sistema di riferimento solidale con il corpo del drone si fa utilizzo della seguente matrice di rotazione

$$\mathbf{J}_{B_i} = \begin{bmatrix} \cos\eta & -\sin\eta \\ \sin\eta & \cos\eta \end{bmatrix} \tag{4.11}$$

$$\begin{bmatrix} a_{1_s i} \\ a_{1_s i} \end{bmatrix} = \mathbf{J}_{B_i} \begin{bmatrix} u_{1_s i} \\ v_{1_s i} \end{bmatrix}. \tag{4.12}$$

Le equazione (4.9) è calcolata come mostrato in figura 4.16.

Nella sezione di destra della figura 4.13 troviamo, infine, il calcolo delle seguenti equazioni:

$$\vec{t_i} = C_T \rho A r^2 \omega_i^2 \begin{pmatrix} -\sin a_{1_s i} \\ \cos a_{1_s i} \sin b_{1_s i} \\ -\cos b_{1_s i} \cos a_{1_s i} \end{pmatrix} \tag{4.13}$$

$$\vec{q_i} = C_Q \rho A r^3 \omega_i |\omega_i| \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{4.14}$$

$$\vec{m}_i = \vec{t_i} \times \vec{d_i} \tag{4.15}$$

dove ρ è la densità dell'aria, r è il raggio del rotore, A è l'area del rotore e C_T e C_Q sono rispettivamente il coefficiente di spinta e il fattore di *drag*.

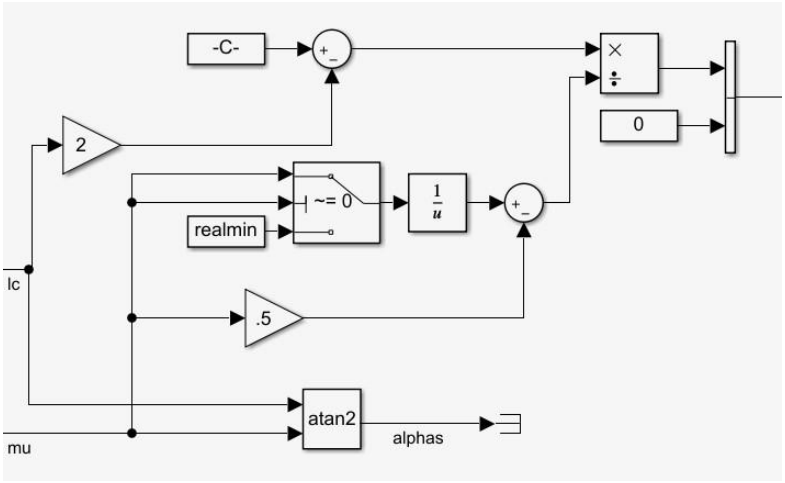


Figura 4.16: Calcolo dell'angolo di orientamento

I vettori $\vec{t}_i$ e $\vec{q}_i$ sono la spinta e la coppia dell'i-esimo rotore, mentre $\vec{m}_i$ è il momento causato dalla spinta dell'i-esimo motore.

Per un rotore in bilico, il momento prodotto dal $flapping$ del rotore è dovuto unicamente al vettore di spinta che agisce intorno allo spostamento dal centro di gravità del veicolo. Le prime armoniche degli angoli di $flapping$ longitudinale e laterale dell'i-esimo rotore sono indicati da $a_{1_s i}$ e $b_{1_s i}$, rispettivamente.

In figura 4.17 è mostrato il calcolo della (4.13), mentre la (4.14) e mostrata in 4.18.

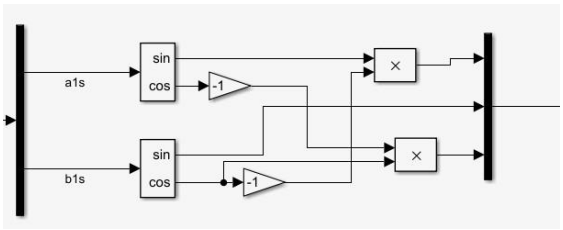


Figura 4.17: Calcolo della spinta del rotore i-esimo

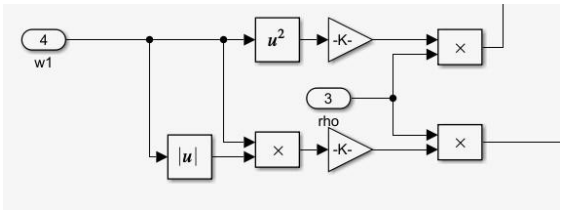


Figura 4.18: Calcolo dell coppia del rotore i-esimo

La corrispondenza tra l'equazione (4.15) e il modello a blocchi relativo, è mostrato in figura 4.19.

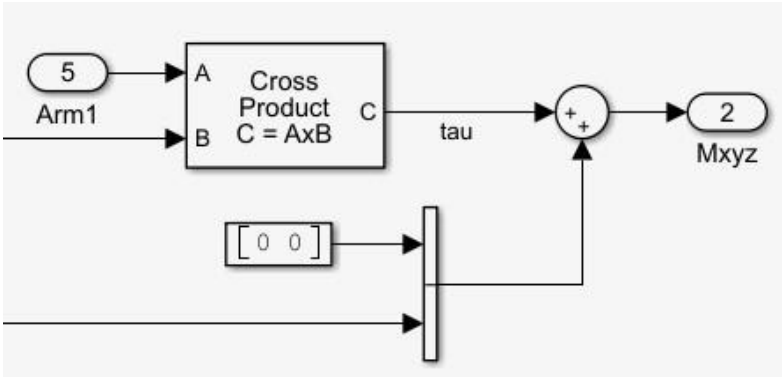


Figura 4.19: Calcolo del momento del rotore i-esimo

4.4 Flight Control System (FCS)

Il blocco FCS rappresenta il sistema di controllo del quadricottero all'interno del quale si trovano i controllori di default di ogni grado di libertà. Analizziamo nel dettaglio, per l'espletamento del progetto, la struttura dell'intero blocco al fine di comprendere come poter sostituire nuovi controllori a quelli già preseti.

Il blocco si presenta come in figura 4.20, e si possono notare gli ingressi e le uscite che lo compongono. Come input abbiamo i segnali di riferimento dell'assetto e la posizione del veicolo provenienti dal blocco *Command*, i dati elaborati dai sensori di bordo che serviranno per poter stimare con precisione le variabili di stato del sistema, ed infine i dati relativi alle immagini della videocamera, che passeranno prima per il blocco *Image Processing System* per essere processati.

Gli output, invece, sono costituiti dal segnale di controllo degli attuatori, che permetterà di regolare la velocità di questi ultimi in base ai controlli eseguiti precedentemente sui gradi di libertà, e il $flag$, che è un segnale binario che permette di fermare la simulazione qualora si presentassero condizioni anomale di volo.

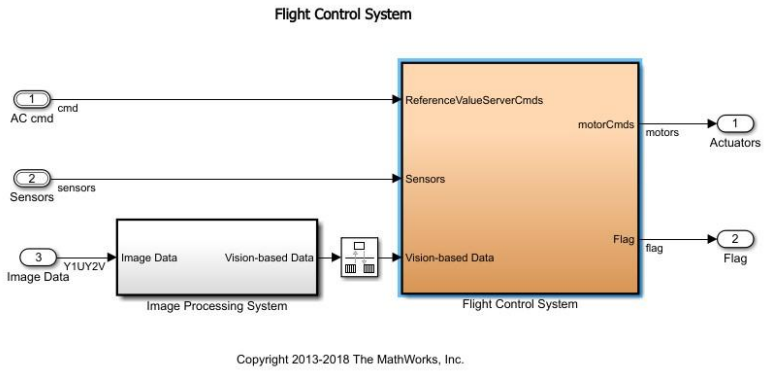


Figura 4.20: Blocco *Flight Control System*

Accedendo al blocco interno ci si ritrova di fronte alla schermata diversi $sotto - blocchi$ che svolgono le seguenti funzionalità:

- ***estimator*:** questo blocco si occupa di ricevere in ingresso i dati relativi ai sensori, dopo esser stati raggruppati nel blocco *sensordata_group*, per poi effettuarne una stima al fine di restituire i valori dello stato del sistema ai blocchi immediatamente successivi;
- ***controller*:** rappresenta il blocco che contiene al suo interno i controllori di default del sistema e che restituisce il segnale di riferimento dei motori e della posizione del veicolo;
- ***Crash Predictor Flags*:** permette di attivare la variabile $flag$ quando il drone si allontana dalle posizioni e/o orientazioni di riferimento;
- ***Logging*:** raccogle tutti i valori relativi alla posizione e all'orientazione nonchè le velocità, le accelerazioni e altre variabili legate alla simulazione al fine di poterne visualizzare gli andamenti.

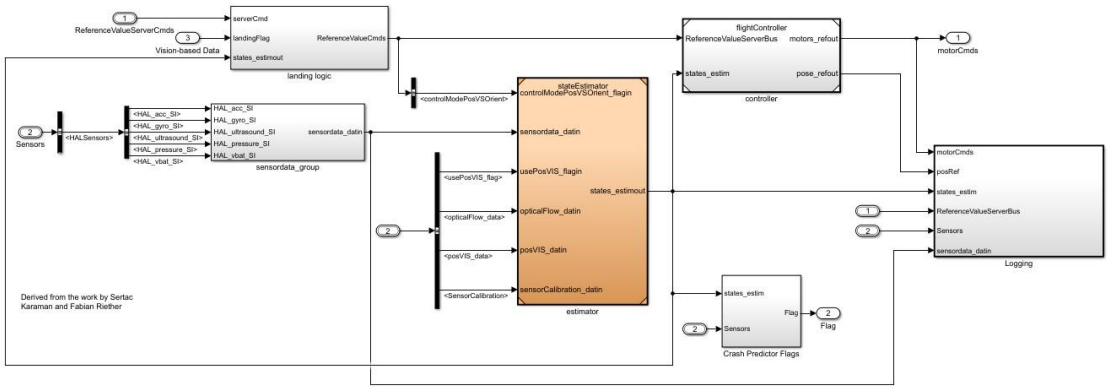


Figura 4.21: Apertura del blocco *Flight Control System*

Il blocco *estimator* è composto da ulteriori sotto-blocchi il cui scopo è quello di fornire una stima precisa delle variabli di stato. In particolare i dati vengono pre-processati all'interno del blocco *SensorPreprocessing*, per poi passare per i blocchi: *Complementary Filter* che fornisce in uscita il valore stimato delle velocità angolari e degli angoli di Eulero, *EstimatorXYPosition* che fornisce i valori di posizione e velocità lungo gli assi longitudinali e latitudinali del veicolo, *EstimatorAltitude* che fornisce i valori stimati della quota e della velocità lungo l'asse z.

I valori così ottenuti vengono poi forniti al blocco *Flight Controller*. Quest'ultimo si presenta come mostrato in figura 4.23 e si compone dei seguenti blocchi:

- ***Yaw***;
- ***XY-to-reference-orientation***;
- ***Attitude***;
- ***gravity feedforward/equilibrium thrust***;
- ***ControlMixer***;
- ***thrustsToMotorCommands***.

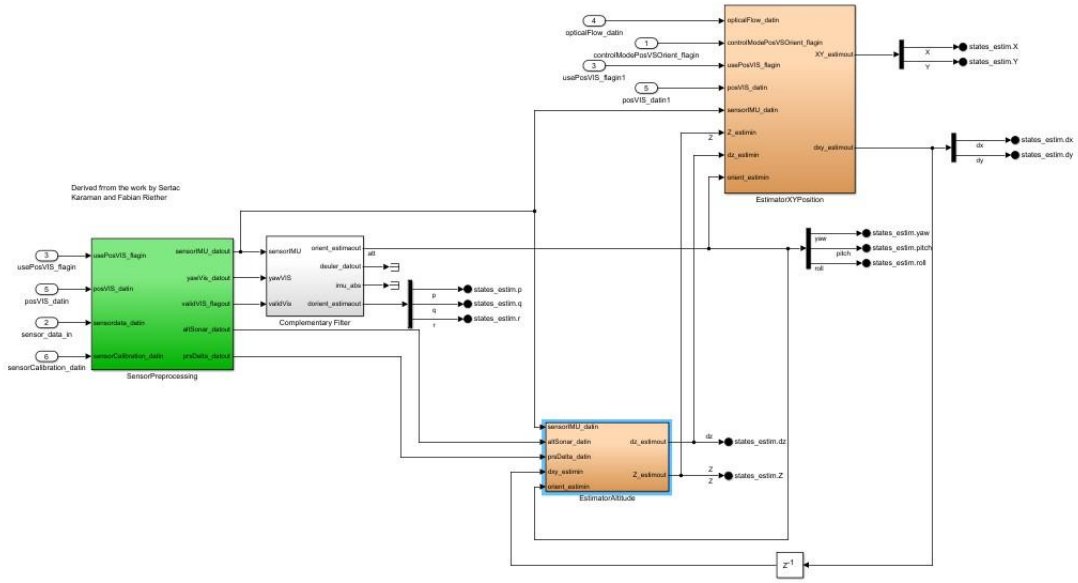


Figura 4.22: Apertura del blocco *State Estimator*

I primi quattro rappresentano i controllori relativi alle variabili di imbardata (yaw), spostamento trasversale (x e y), beccheggio e rollio ($pitch$ e $roll$) ed infine altitudine (z). Gli ultimi due forniscono invece i riferimenti necessari all'azione degli attuatori ottenuti dai blocchi di controllo precedenti.

Notiamo inoltre che viene effettuato un controllo in cascata. Si passa dapprima per il blocco di controllo dello yaw che è necessario per rilevare la corretta posizione del drone. Si arriva dunque al controllo della x e della y che forniscono il segnale di riferimento degli angoli di beccheggio e rollio, per poi concludere con il controllo di questi ultimi.

Ciò è dovuto al fatto che il drone è un sistema sotto attuato dove i gradi di libertà sono accoppiati l'uno con l'altro. L'unico grado di libertà disaccoppiato da tutti e quello relativo alla dinamica verticale.

Analizziamo ora i singoli controllori di ogni grado di libertà.

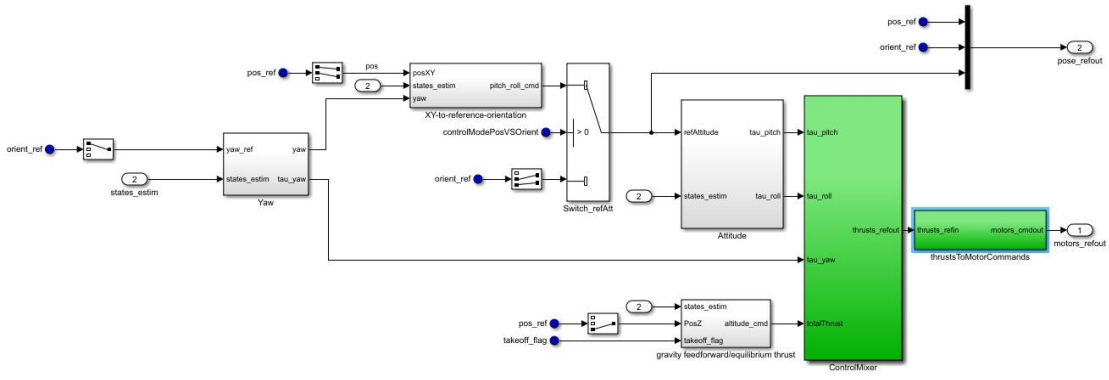


Figura 4.23: Apertura blocco *Flight Control*

4.4.1 Controllore dell'angolo di imbardata

Il seguente controllore, così come tutti i controllori di default, rappresenta un controllore *Proporzionale Integrale Derivativo (PID)*. Nello specifico, come mostra la figura 4.24, sono presenti solo l'azione proporzionale e derivativa. Dall'ingresso *states_ estim*, vengono prelevati solo i valori dello yaw e della sua derivata di cui il primo è confrontato con il valore di riferimento per generare un errore che poi viene moltiplicato per il guadagno proporzionale, mentre il secondo è moltiplicato direttamente per il guadagno derivativo.

I segnali vengono poi sommati insieme per generare il riferimento da fornire al blocco *ControlMixer* per il controllo dei motori.

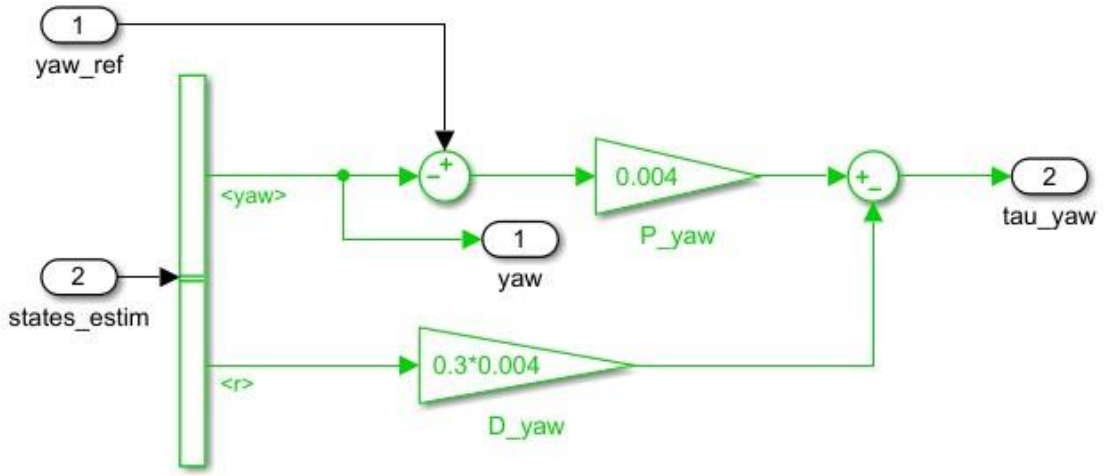


Figura 4.24: Controllore dello yaw

4.4.2 Controllore della posizione logitudinale e latitudinale

Come discusso precedentemente il drone è un sistema sottoattuato dove i gradi di libertà sono accoppiati l'uno con l'altro. In particolare, come mostrato nel capitolo relativo alla linearizzazione, non è possibile effettuare un controllo diretto sulle posizioni longitudinali e latitudinali, ma è tuttavia possibile controllarle indirettamente mediante gli angoli di beccheggio e rollio. Per questo motivo il controllore del $pitch$ e del $roll$ si trova immediatamente dopo quello della x e della y, in quanto, da qui, viene generato un segnale di riferimento che non sarà utilizzato per il controllo dei motori ma sarà fornito al controllore dell'assetto come segnale di riferimento variabile.

La struttura del controllore è analoga a quella dello yaw salvo che per la sezione superiore. Quest'ultima si occupa di calcolare la matrice 2×2 di rotazione utilizzando l'angolo di imbardata fornitogli dal blocco precedente. Infatti una variazione dello yaw, e quindi una rotazione rispetto l'asse z, provoca anche una variazione di posizione nel piano xy.

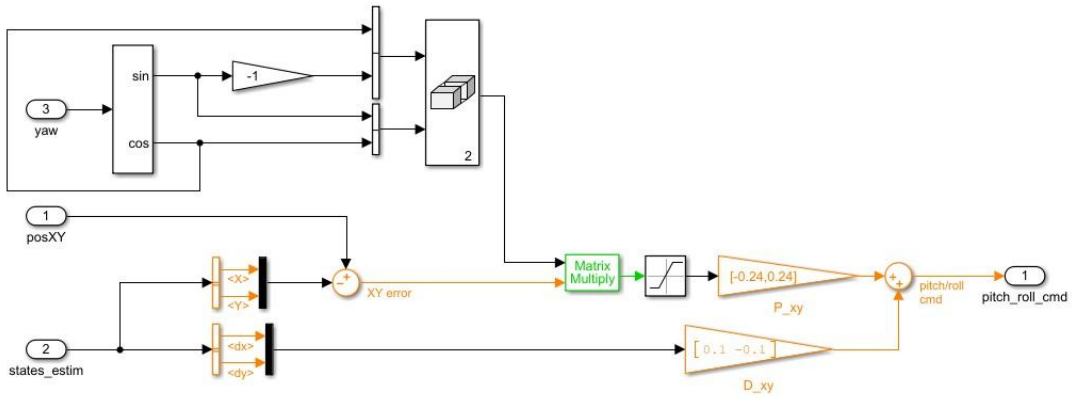


Figura 4.25: Controllore della x e della y

Tale matrice viene opportunamente moltiplicata per il segnale di errore generato dalla differenza delle posizioni longitudinali e latitudinali e il segnale di riferimento proveniente dal blocco di comando.

L'implementazione del nuovo controllore dovrà avvenire in questo blocco, sostituendo il controllore di default con nuovi blocchi *Simulink* e facendo attenzione a lasciare inalterato il controllo sulla variabile y dato che non è oggetto di esame.

4.4.3 Controllore dell'assetto

Il blocco di controllo dell'assetto riceve il segnale di riferimento variabile direttamente dal blocco precedente. Modificando, tuttavia, lo switch presente tra i due blocchi sarà possibile decidere di scegliere anche un segnale costante relativo all'orientazione, oppure il segnale relativo alla modalità di controllo della posizione o dell'orientazione.

Il controllore presenta sia un azione proporzionale che una derivativa e integrativa, le quali sono sommate insieme per ottenere i due segnali di riferimento.

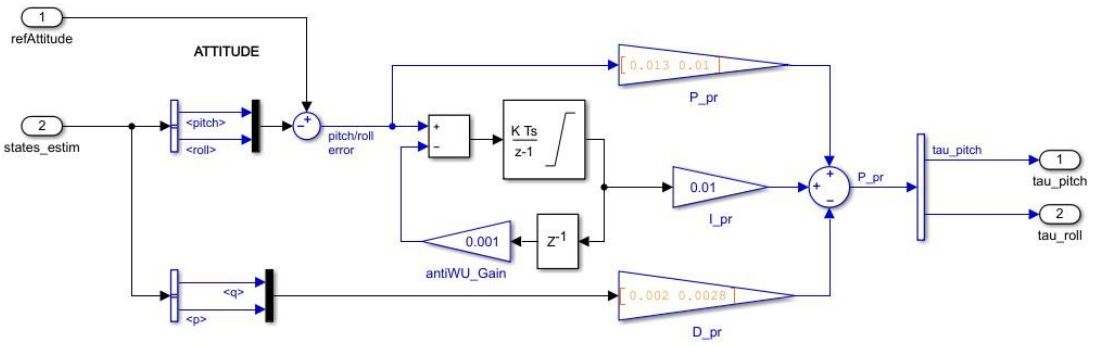


Figura 4.26: Controllore del *pitch* e del *roll*

4.4.4 Controllore della quota

La variabile della quota è l'unica ad essere disaccoppiata dalle altre. Il controllore presenta un azione proporzionale e derivativa. La posizione viene confrontata con il riferimento per produrre un segnale di errore che, dopo esser stato moltiplicato per il guadagno proporzionale, si somma al segnale relativo alla velocità lungo z per il guadagno derivativo.

Uno switch permette di dividere la fase di decollo, che necessita di una spinta maggiore degli attuatori, con la fase di volo stazionario. Ciò viene effettuato dalla variabile $takeoff\_flag$.

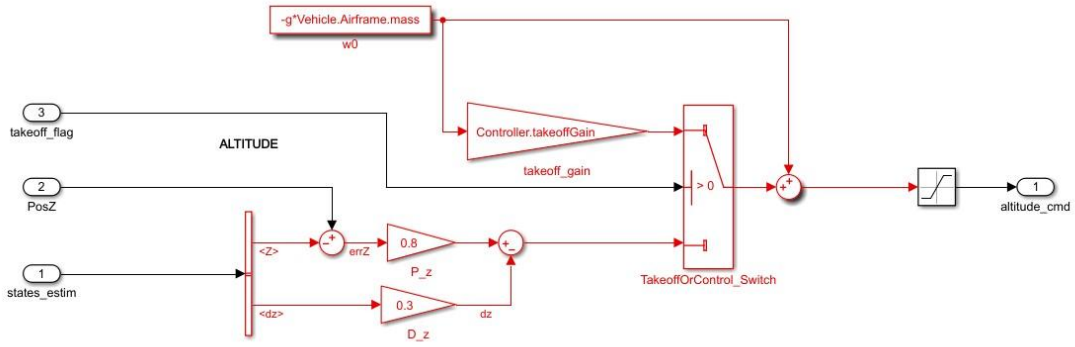


Figura 4.27: Controllore della z

Capitolo 5

Ricavo della funzione di trasferimento

In questo capitolo vederemo come ricavare la funzione di trasferimento della dinamica lungo x necessaria per la sintesi di un nuovo controllore.

La tecnica di sintesi di cui si vuole fare utilizzo e quella del luogo delle radici. Essendo quest'ultima una tecnica di controllo lineare, bisognerà utilizzare un approssimazione lineare del sistema reale che, nel modello $Simulink$ del quadricottero, è possibile ottenere modificando il valore di una singola variabile di comando: la $VSS\_VEHICLE$.

Come già visto nel capitolo precedente, settando il valore $VSS\_VEHICLE = 0$, si imposta per la simulazione il modello lineare mostrato in figura 5.1.

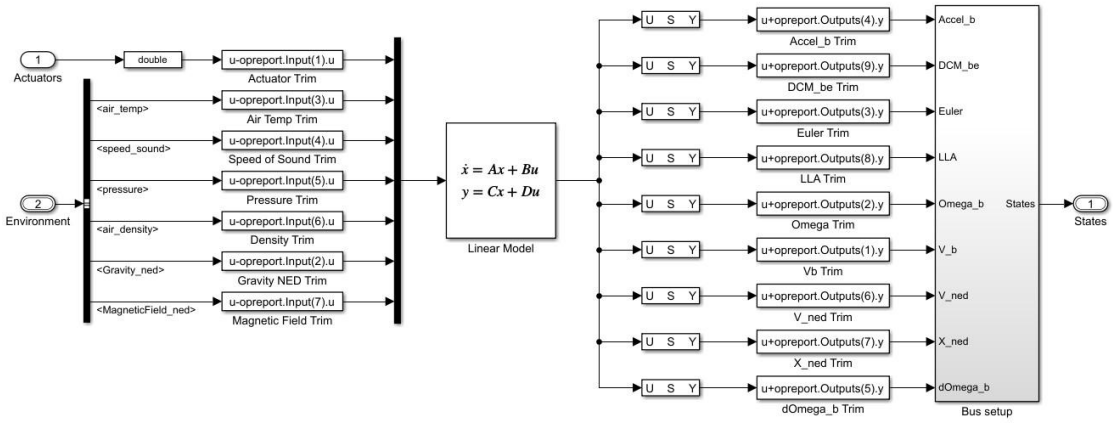


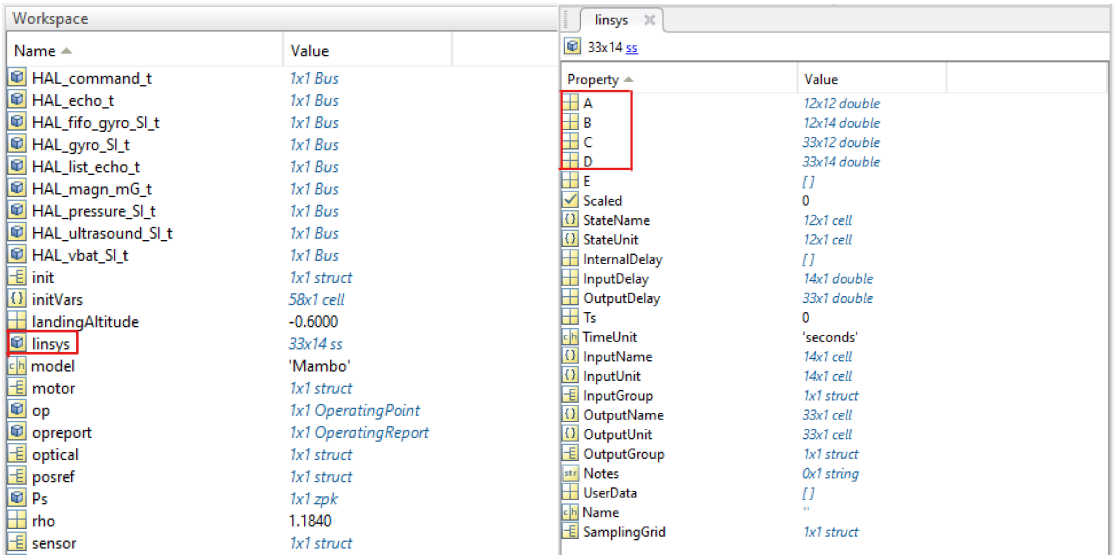
Figura 5.1: Blocco relativo al modello lineare

Tale modello è stato ottenuto utilizzando la funzione $trimLinearizeOpPoint$, la quale utilizza il $tool$ ***Simulink Control Design*** per eseguirne la linearizzazione.

5.1 Funzione di trasferimento della x

Il primo passo da compiere per ottenere la FdT desiderata è quello di aprire la struttura dati $linsys$ presente nel $WorkSpace$ di Matlab. Dopo aver selezionato il modello lineare, tale struttura non sarà immediatamente disponibile. Bisognerà, infatti, avviare la simulazione almeno una volta per dare la possibilità al software di calcolare le matrici che comporranno il sistema lineare.

A questo punto, aperta la struttura dati, si avrà la possibilità di accedere alle matrici A, B, C e D dalle quali si potrà isolare la sola dinamica lungo x.



(a) Struttura dati (b) Matrici del sistema lineare

Figura 5.2: Accesso alla struttura dati $linsys$

All'interno di tale struttura sono presenti anche i vettori di stato, ingresso ed uscita che permetteranno una migliore lettura delle matrici.

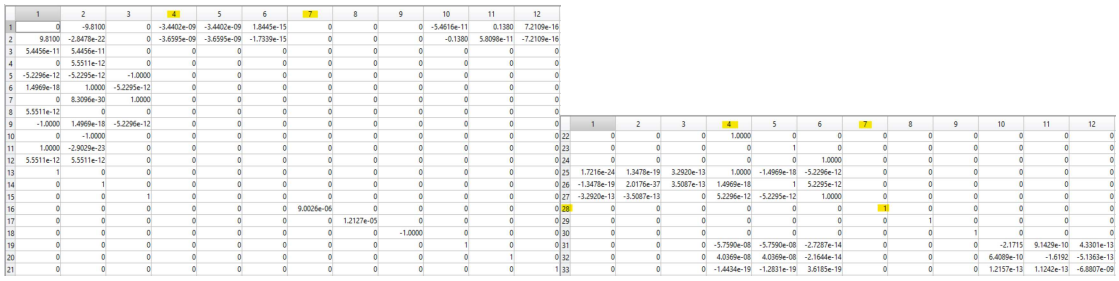
Qui di seguito sono mostrati tutti i dati di cui avremo bisogno.

	1	2	3	4	5	6	7	8	9	10	11	12
1	0	3.6611e-17	0	0	0	0	0	0	0	1	3.0292e-23	-5.2296e-12
2	-3.6611e-17	0	0	0	0	0	0	0	0	0	1	5.2295e-12
3	-6.8347e-13	-2.0323e-28	0	0	0	0	0	0	0	0	-5.2295e-12	1.0000
4	0	-9.8100	0	-3.4402e-09	-3.4402e-09	6.8532e-13	0	0	0	-5.4616e-11	0.1380	-3.2848e-13
5	9.8100	-2.8478e-22	0	-3.6595e-09	-3.6595e-09	-6.8520e-13	0	0	0	-0.1380	5.8098e-11	-3.5159e-13
6	5.4456e-11	5.4456e-11	0	-6.8347e-13	6.8347e-13	0	0	0	0	3.2920e-13	3.5087e-13	0
7	1.7216e-24	1.3478e-19	3.2920e-13	1.0000	-1.4969e-18	-5.2296e-12	0	0	0	0	0	0
8	-1.3478e-19	2.0176e-37	3.5087e-13	1.4969e-18	1	5.2295e-12	0	0	0	0	0	0
9	-3.2920e-13	-3.5087e-13	0	5.2296e-12	-5.2295e-12	1.0000	0	0	0	0	0	0
10	0	0	0	-5.7590e-08	-5.7590e-08	-2.7287e-14	0	0	0	-2.1715	9.1429e-10	4.3301e-13
11	0	0	0	4.0369e-08	4.0369e-08	-2.1644e-14	0	0	0	6.4089e-10	-1.6192	-5.1363e-13
12	0	0	0	-1.4434e-19	-1.2831e-19	3.6185e-19	0	0	0	1.2157e-13	1.1242e-13	-6.8807e-09

Figura 5.3: Matrice A

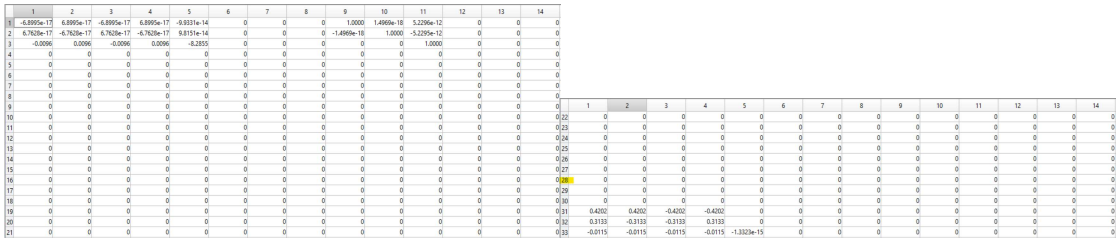
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	-6.8995e-17	6.8995e-17	-6.8995e-17	6.8995e-17	-9.9331e-14	0	0	0	1.0000	1.4969e-18	5.2296e-12	0	0	0
5	6.7628e-17	-6.7628e-17	6.7628e-17	-6.7628e-17	9.8151e-14	0	0	0	-1.4969e-18	1.0000	-5.2295e-12	0	0	0
6	-0.0096	0.0096	-0.0096	0.0096	-8.2855	0	0	0	0	0	1.0000	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0.4202	0.4202	-0.4202	-0.4202	2.5288e-14	0	0	0	0	0	0	0	0	0
11	0.3133	-0.3133	-0.3133	0.3133	1.8856e-14	0	0	0	0	0	0	0	0	0
12	-0.0115	-0.0115	-0.0115	-0.0115	-7.9371e-16	0	0	0	0	0	0	0	0	0

Figura 5.4: Matrice B



(a) Prima parte (b) Seconda parte

Figura 5.5: Matrice C



(a) Prima parte (b) Seconda parte

Figura 5.6: Matrice D

	1
1	phi theta psi(1)
2	phi theta psi(2)
3	phi theta psi(3)
4	ub,vb,wb(1)
5	ub,vb,wb(2)
6	ub,vb,wb(3)
7	xe,ye,ze(1)
8	xe,ye,ze(2)
9	xe,ye,ze(3)
10	p,q,r(1)
11	p,q,r(2)
12	p,q,r(3)

(a) Stato

	1
1	Actuators(1)
2	Actuators(2)
3	Actuators(3)
4	Actuators(4)
5	AtmosphereBus.air_density
6	AtmosphereBus.air_temp
7	AtmosphereBus.pressure
8	AtmosphereBus.speed_sound
9	Gravity_ned(1)
10	Gravity_ned(2)
11	Gravity_ned(3)
12	MagneticField_ned(1)
13	MagneticField_ned(2)
14	MagneticField_ned(3)

(b) Ingresso

1	Accel_body(1)
2	Accel_body(2)
3	Accel_body(3)
4	DCM_be(1)
5	DCM_be(2)
6	DCM_be(3)
7	DCM_be(4)
8	DCM_be(5)
9	DCM_be(6)
10	DCM_be(7)
11	DCM_be(8)
12	DCM_be(9)
13	Euler(1)
14	Euler(2)
15	Euler(3)
16	LLA(1)
17	LLA(2)
18	LLA(3)
19	Omega_body(1)
20	Omega_body(2)
21	Omega_body(3)

(c) Uscita(1)

22	V_body(1)
23	V_body(2)
24	V_body(3)
25	V_ned(1)
26	V_ned(2)
27	V_ned(3)
28	X_ned(1)
29	X_ned(2)
30	X_ned(3)
31	dOmega_body(1)
32	dOmega_body(2)
33	dOmega_body(3)

(d) Uscita(2)

Figura 5.7: Vettori di stato, ingresso e uscita

Come si nota dai valori evidenziati in figura 5.7, gli stati a cui siamo interessati sono quelli relativi alla velocità di x nel sistema di riferimento solidale al corpo del drone e la posizione nel sistema di riferimento inerziale. Analogamente per l'uscita si è interessati alla sola riga 28 relativa alla posizione x.

Ricordando che un sistema lineare e stazionario assume la seguente forma

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) + Bu(t) \end{cases}, \tag{5.1}$$

eseguendo i prodotti tra le matrici A, B, C e D con i corrispondenti vettori di stato, ingresso e uscita e considerando solo le righe evidenziate, trascurando dunque tutti quei valori che assumono ordini di grandezza $<< 0.001$ e ricordando che il vettore $Gravity\_ned$ assume la seguente forma $g = [0 \;\; 0 \;\; 1]$, si può riscrivere il sistema lineare come segue

$$\begin{cases} \ddot{x} = -9.81\ \theta + 0.138\ \dot{\theta} \\ \dot{x} = \dot{x} \\ y = x \end{cases}. \tag{5.2}$$

Come si può notare, le funzioni di stato del sistema (5.2) ricordano quelle del sistema (3.26) mostrato nel capitolo della linearizzazione, con un'unica differenza dovuta alla presenza di un termine che dipende dalla velocità angolare dell'angolo di beccheggio.

Applicando la trasformata di Laplace e ricordando che lo stato iniziale per x e per θ è nullo si ha

$$\begin{aligned} \mathcal{L}[x] &= X(s) \\ \mathcal{L}[y] &= Y(s) \\ \mathcal{L}[\ddot{x}] &= s^2 X(s) \\ \mathcal{L}[\theta] &= \Theta(s) \\ \mathcal{L}[\dot{\theta}] &= s\Theta(s) \end{aligned} \tag{5.3}$$

Dove si sono applicate le seguenti proprietà delle trasformate di Laplace

$$\mathcal{L}[f'(t)] = s \cdot F(s) - f(0) \tag{5.4}$$

$$\mathcal{L}[f''(t)] = s^2 \cdot F(s) - s \cdot f(0) - f'(0). \tag{5.5}$$

Si ottiene in definitiva il seguente sistema

$$\begin{cases} s^2 X(s) = -9.81\Theta(s) + 0.138 s\Theta(s) \\ Y(s) = X(s) \end{cases}, \tag{5.6}$$

da cui si ricava la seguente funzione di trasferimento

$$\frac{X(s)}{\Theta(s)} = \frac{0.138(s - 71.0869)}{s^2} \tag{5.7}$$

che lega la variabile x alla variazione dell'angolo di beccheggio, inteso in questo caso come ingresso del sottosistema che descrive la dinamica lungo x.

Capitolo 6

Sintesi del controllore

6.1 Generalità

Il luogo delle radici è un procedimento grafico mediante il quale è possibile stabilire delle correlazioni tra i valori dei poli, degli zeri, del guadagno K' che caratterizzano la funzione razionale di variabile complessa

$$F(s) = K' \frac{\prod_{i=1}^{m}(s - z_i)}{\prod_{i=1}^{n}(s - p_i)} \tag{6.1}$$

e i valori dell'equazione algebrica

$$\prod_{i=1}^{n}(s - p_i) + K' \prod_{i=1}^{m}(s - z_i) = 0. \tag{6.2}$$

Ricordando che le radici dell'equazione (6.2) coincidono con i poli della funzione razionale

$$W(s) = \frac{F(s)}{1 + F(s)} \tag{6.3}$$

tale processo può essere utilizzato nello studio di un sistema a controreazione. Il procedimento consiste nell'individuare il luogo geometrico dei punti del piano complesso descritto dalle radici dell'equazione (6.2) al variare del parametro K'. Qualora K' assuma valori positivi si parlerà di *luogo positivo* delle radici. Viceversa, se K' assume valori negativi il luogo prende il nome di *luogo negativo*.

Generalmente, per poter individuare il luogo delle radici, positivo o negativo che sia, è conveniente riscrivere la (6.2) nella forma

$$\prod_{i=1}^{n}(s - p_i) = -K' \prod_{i=1}^{m}(s - z_i) \tag{6.4}$$

per poi sostituire a questa la coppia di uguaglianze tra grandezze reali

$$\prod_{i=1}^{n}|s - p_i| = |K'| \prod_{i=1}^{m}|s - z_i| \tag{6.5}$$

$$\sum_{i=1}^{n} \angle(s - p_i) = \pi + \angle(K') + \sum_{i=1}^{m} \angle(s - z_i) + h2\pi \tag{6.6}$$

dove h è un qualsiasi numero intero, positivo o negativo.

La prima è detta *condizione di modulo* e ad essa è associata la proprietà secondo cui in ogni punto del luogo delle radici, il valore assoluto di K' è pari al prodotto del modulo dei vettori che congiungono il punto con i poli di $F(s)$, diviso il prodotto dei moduli dei vettori che uniscono il punto agli zeri di $F(s)$

$$|K'| = \frac{\prod_{i=1}^{n} |s - p_i|}{\prod_{i=1}^{m} |s - z_i|}. \tag{6.7}$$

La seconda, detta *condizione di fase*, assume due diverse forme a seconda del segno di K' e afferma che la differenza tra la somma delle fasi dei vettori orientati dai poli di $F(s)$ ai punti del luogo e la somma delle fasi dei vettori orientati dagli zeri di $F(s)$ ai punti del luogo è pari a π o 0 più un multiplo di 2π a seconda che si tratti del luogo positivo o negativo

$$\sum_{i=1}^{n} \angle(s - p_i) - \sum_{i=1}^{m} \angle(s - z_i) = \begin{cases} \pi + h2\pi & (luogo\ positivo) \\ h2\pi & (luogo\ negativo) \end{cases}. \tag{6.8}$$

La seconda condizione è utilizzata per il tracciamento del luogo, mentre la prima potrebbe essere utilizzata per far corrispondere ai punti del luogo il valore K' in corrispondenza del quale il punto stesso è radice.

Passiamo ora in rassegna alcune regole fondamentali per il tracciamento:

1. I punti singolari del luogo possono essere calcolati mediante il sistema di equazioni:

$$\begin{cases} f(s, K') = 0 \\ \frac{\partial f(s,K')}{\partial s} = 0 \end{cases} \tag{6.9}$$

   con

$$f(s, K') = \prod_{i=1}^{n} (s - p_i) + K' \prod_{i=1}^{m} (s - z_i). \tag{6.10}$$

2. I poli della $F(s)$ sono punti del luogo corrispondenti al valore nullo di K'. Per $K' = 0$ si ha infatti:

$$\prod_{i=1}^{n} (s - p_i) = 0. \tag{6.11}$$

3. Quando K' tende a $+\infty$ (per il luogo positivo) o a $-\infty$ (per il luogo negativo), m rami del luogo convergono sugli zeri della $F(s)$ ed altri $n - m$ rami divergono verso il punto improprio.

4. Al luogo positivo appartengono i punti dell'asse reale che lasciano alla loro destra un numero dispari di zeri e/o poli (contati con la loro molteplicità) della funzione $F(s)$. I restanti punti dell'asse reale appartengono al luogo negativo.

5. I rami del luogo che divergono verso il punto improprio tendono asintoticamente a $n - m$ semirette che formano, con l'asse delle ascisse, angoli pari a

$$\phi_+ = \frac{\pi + 2h\pi}{n - m} \tag{6.12}$$

nel luogo positivo ed a

$$\phi_- = \frac{2h\pi}{n-m} \tag{6.13}$$

nel luogo negativo. Tutte queste semirette hanno in comune il punto dell'asse reale di ascissa

$$s_0 = \frac{\sum_{i=1}^{n} p_i - \sum_{i=1}^{m} z_i}{n-m} \tag{6.14}$$

che viene detto *centro degli asintoti*.

In figura 6.1 sono riportate le configurazioni assunte dagli asintoti in alcuni casi semplici.

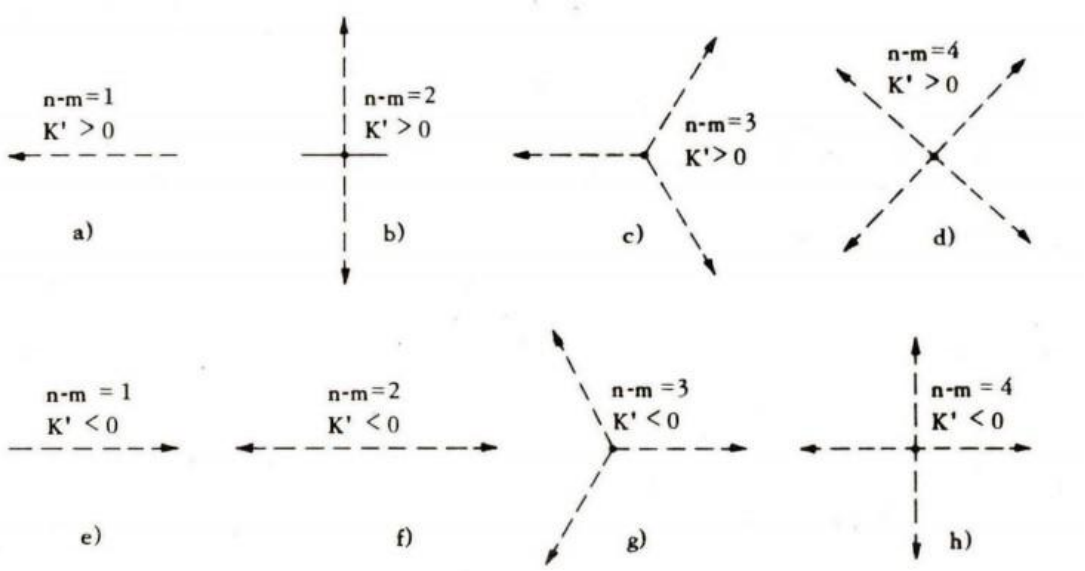


Figura 6.1: Configurazioni assunte dagli asintoti

6. Gli eventuali punti di attraversamento dell'asse immaginario da parte del luogo si possono determinare applicando l'algoritmo di *Routh* all'equazione (6.2).

6.2 Ricavo della funzione di controllo

Per poter effettuare la sintesi del controllore mediante il luogo delle radici si è fatto utilizzo del *tool* ***Control System Designer*** accessibile direttamente dalla *CommandWindow* di *Matlab* digitando il comando *sisotool* e passandogli come parametro la funzione di trasferimento ricavata in precedenza. Dunque è stato necessario riportare la funzione di trasferimento (5.7) in *Matlab*. Ciò è stato realizzato mediante la funzione *zpk* come mostrato in figura 6.2.

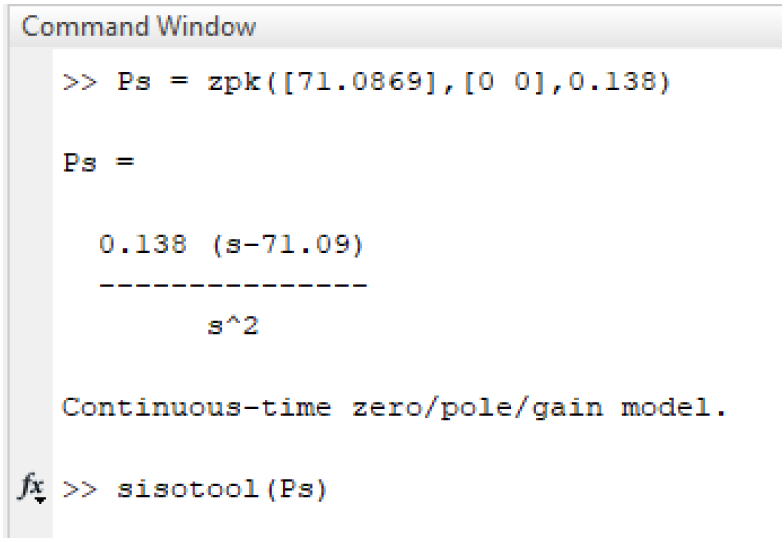


Figura 6.2: Trasfrimento della *FdT* in *Matlab*

All'apertura del *tool*, ci ritroveremo di fronte alla schermata mostrata in figura 6.3, dove vengono rappresentati, oltre al luogo delle radici, anche i diagrammi di *Bode* e la risposta al gradino.

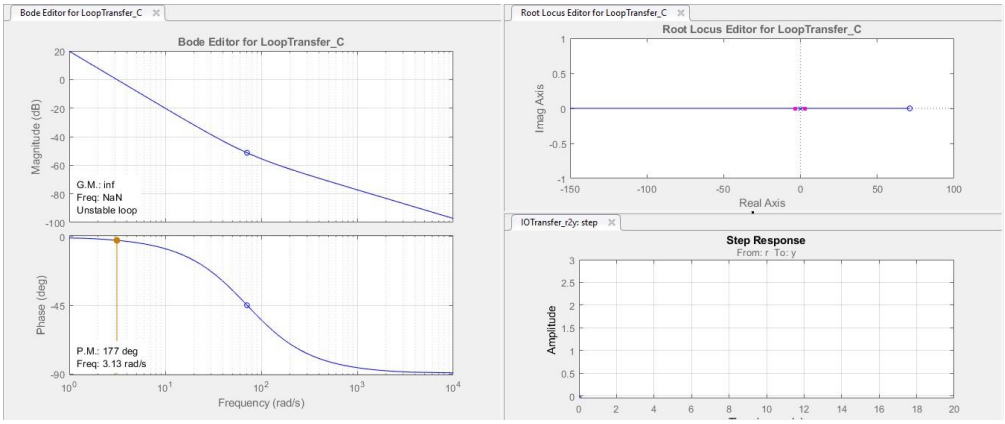


Figura 6.3: Caratteristiche iniziali del sistema

Come prima osservazione possiamo notare che il sistema è di tipo 2, data la presenza di un polo doppio in zero. Il sistema, inoltre, è a **fase non minima.** Infatti, guardando anche solo la

funzione di trasferimento (5.7), si nota la presenza di uno zero positivo. In generale, un sistema può essere di due tipi:

- sistema a *fase minima*;
- sistema a *fase non minima*.

Nel primo caso, il sistema ha tutti gli zeri nel semipiano sinistro del piano complesso. Questa situazione è favorevole in quanto si ha la sicurezza che m rami del luogo positivo, per K' sufficientemente elevato, tenderanno a spostarsi nel semipiano sinistro del piano complesso. Risulta quindi decisiva, ai fini della stabilità, la posizione e l'orientamento degli $n - m$ asintoti del luogo.

Nel secondo caso, per K' sufficientemente elevato si ha instabilità del sistema retroazionato. Non esistono metodi standard per la risoluzione generale del problema di stabilizzazione. Talvolta è necessario studiare il luogo negativo o aggiungere poli e zeri in modo opportuno, con il fine di creare punti singolari che "attraggano" il luogo a sinistra dell'asse immaginario. Il procedimento non risulta essere sempre immediato. Si possono avere, tuttavia, le seguenti situazioni tipiche:

- Se F(s) ha solo poli a parte reale negativa (sistema a ciclo aperto stabile asintoticamente), è necessario e sufficiente, ai fini della stabilità asintotica a ciclo chiuso, scegliere valori di K' non troppo grandi;
- Se F(s) ha solo uno zero ed un polo con parte reale positiva, con il polo posto a destra dello zero, si può porre un altro polo positivo a destra di quello già esistente. In questo modo, si creerà un punto singolare tra questi due poli, ed il luogo può assumere una configurazione tale da stabilizzare il sistema a ciclo chiuso per K' interno ad un opportuno intervallo (relativo ad una porzione di luogo positivo);
- Se F(s) ha solo uno zero ed un polo con parte reale positiva, con il polo posto a sinistra dello zero, si ha sicuramente un punto singolare a destra dello zero positivo. Se non è presente un altro punto singolare nel tratto di luogo negativo tra l'origine ed il polo positivo, allora il luogo può assumere una configurazione tale da stabilizzare il sistema a ciclo chiuso per K' interno ad un opportuno intervallo (in questo caso relativo ad una porzione di luogo negativo).

Essendo dunque il sistema a fase non minima si può decidere di studiare il luogo negativo invece di quello positivo. A tal fine si è modificato il guadagno del controllore con uno negativo, in questo caso -0.1. Aggiungendo, inoltre, la coppia $polo - zero$ $(-100, -1)$ si ottiene il luogo delle radici riportato in figura 6.4. Tali caratteristiche sono ottenibili, dunque applicando il controllore dato da:

$$G(s) = \frac{-10\ (s+1)}{(s+100)}. \tag{6.15}$$

Come si può notare il sistema in catena chiusa è stabile. Inoltre si ha un buon margine di guadagno, osservando i diagrammi di *Bode*, e la risposta al gradino presenta un buon tempo di salita ed una sovraelongazione non troppo pronunciata.

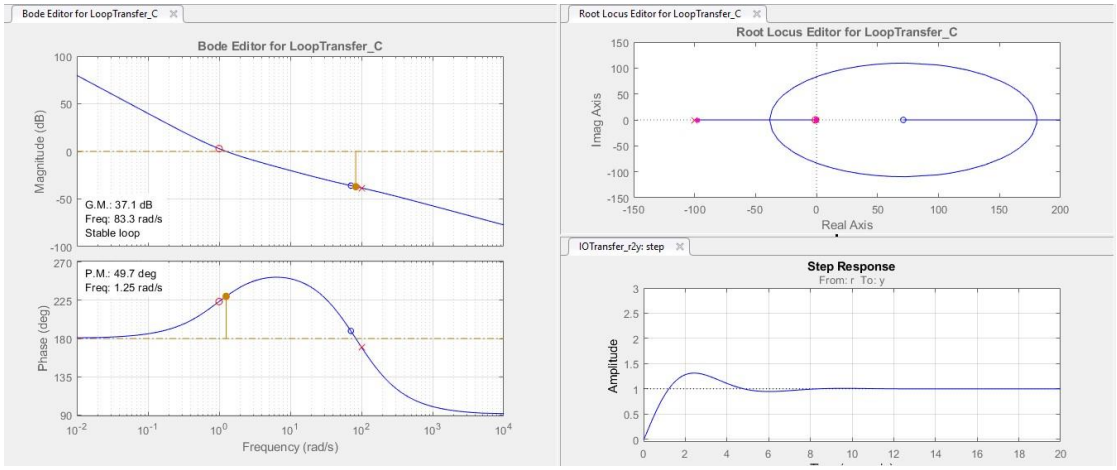


Figura 6.4: Caratteristiche del sistema applicando il nuovo controllore

6.2.1 Implementazione in Matlab

Una volta ottenuto il sistema di controllo, si può passare alla fase di implementazione in $Matlab$. Tale fase è avvenuta modificando in maniera opportuna il controllore di default presentato nel capitolo 4 e mostrato in figura 4.25. Per prima cosa, si è cercato di disaccoppiare i blocchi relativi al controllo dell y con quelli costituenti il controllo della x. Dunque, si è eliminato la presenza della variabile dx, non più necessaria per il controllore PID, e si è diviso il segnale relativo all'errore XY (che rappresenta un segnale doppio, uno per la x e uno per la y) in maniera tale da separare il controllore PID della y da quello inserito per la x. Quest'ultimo, in particolare, è costituito da un semplice blocco $Zero - Pole$, preceduto e anteceduto da blocchi per la conversione dei dati, con il quale si è potuto inserire la FdT del controllore appena trovato (6.15). Infine, si sono riuniti i segnali per formare il riferimento per il $pitch$ e il $roll$ mediante un blocco Mux. il tutto è mostrato in figura 6.5.

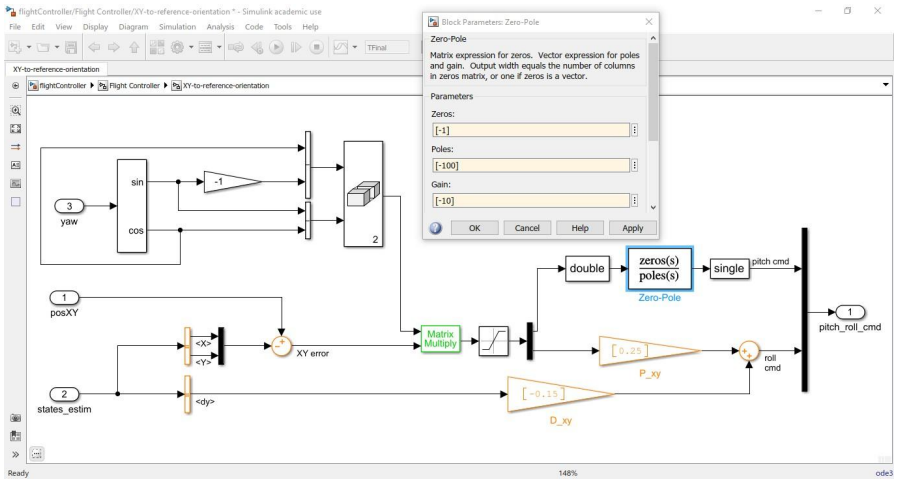


Figura 6.5: Modifica del controllore di default

6.2.2 Simulazione e confronto

Selezionando il sistema lineare impostando $VSS\_VEHICLE = 0$ e avviando la simulazione, si ottengono i tracciati della x e del $pitch$ mostrati in figura. Inoltre, si possono osservare anche i tracciati del controllore di default al fine di comparare i risultati ottenuti:

TRACCIATI X (SISTEMA LINEARE)

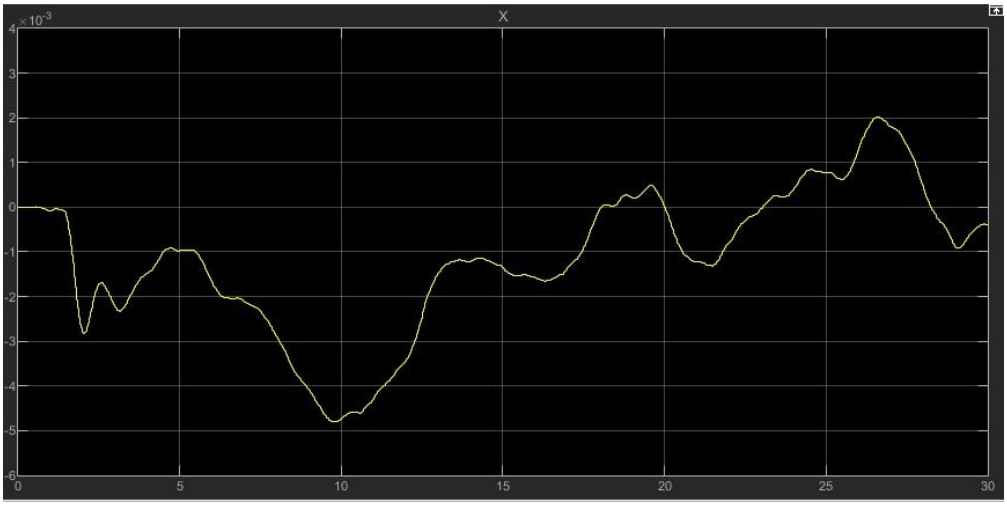


Figura 6.6: Tracciato x del nuovo controllore

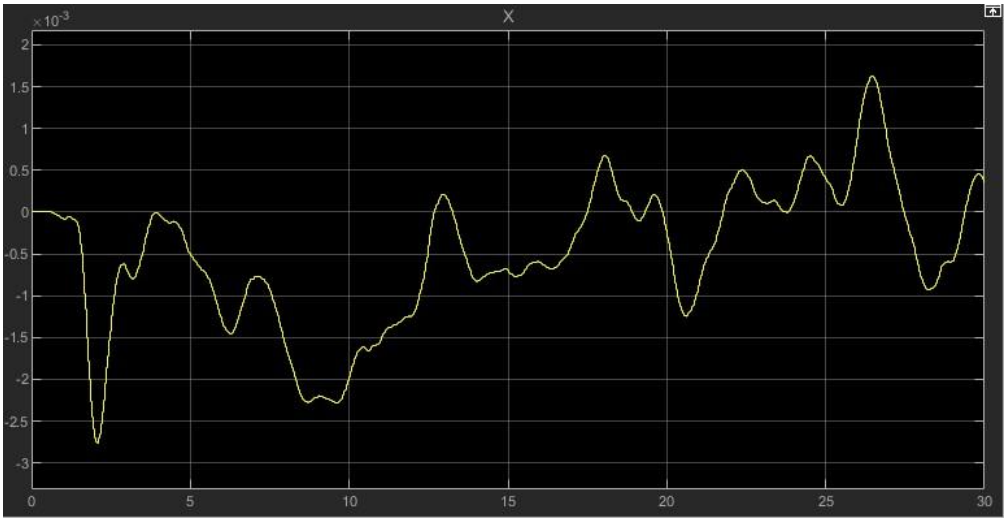


Figura 6.7: Tracciato x del controllore di default

Tracciati pitch (sistema lineare)

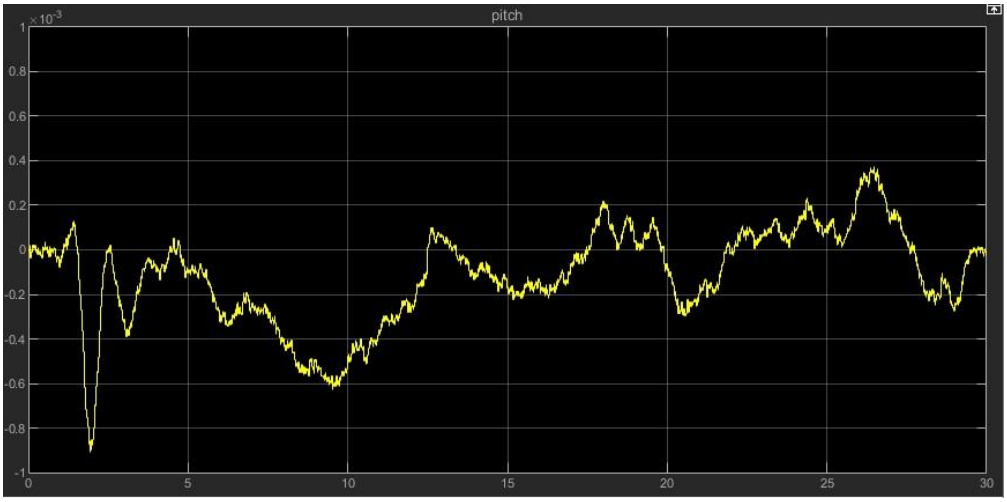


Figura 6.8: Tracciato *pitch* del nuovo controllore

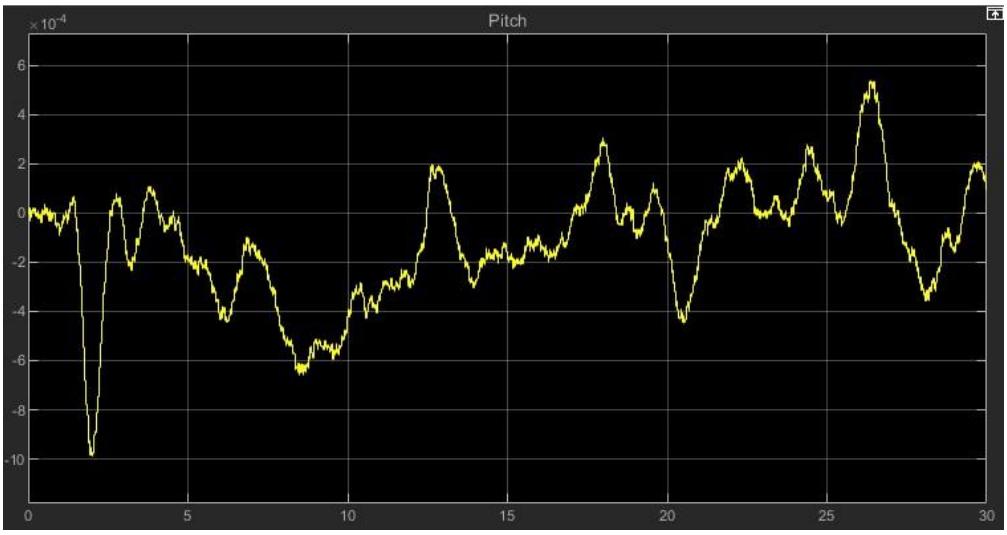


Figura 6.9: Tracciato *pitch* del controllore di default

Analogamente, selezionando il sistema non lineare ($VSS\_VEHICLE = 1$) si ottengono i seguenti risultati:

Tracciati x (sistema non lineare)

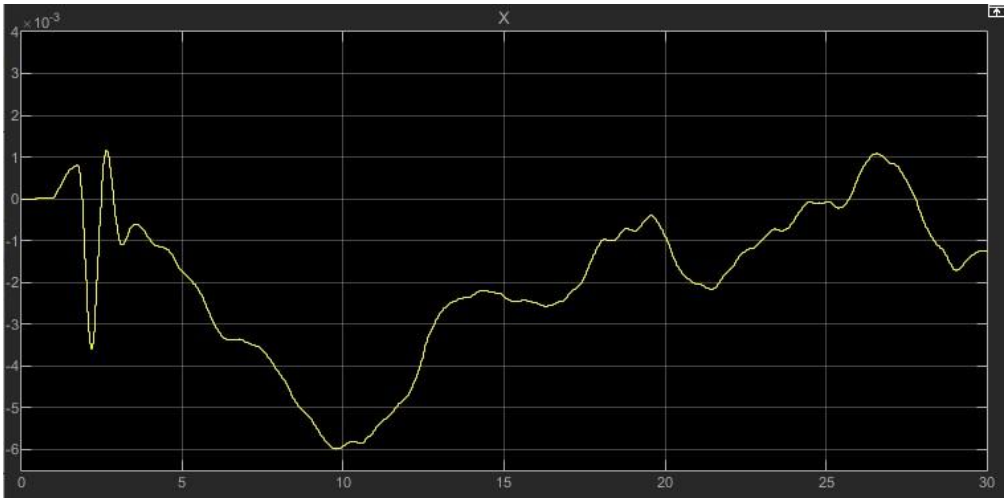


Figura 6.10: Tracciato x del nuovo controllore (N.L.)

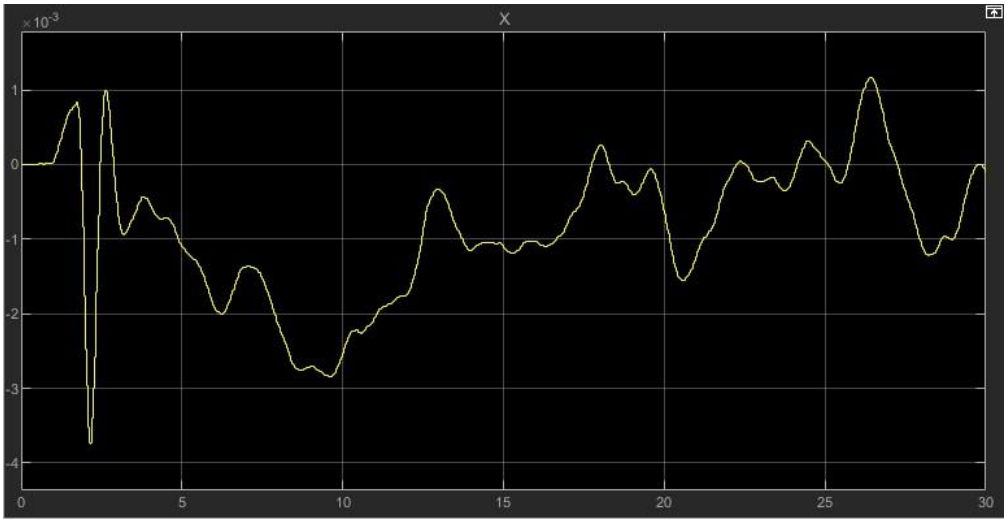


Figura 6.11: Tracciato x del controllore di default (N.L.)

Tracciati pitch (sistema non lineare)

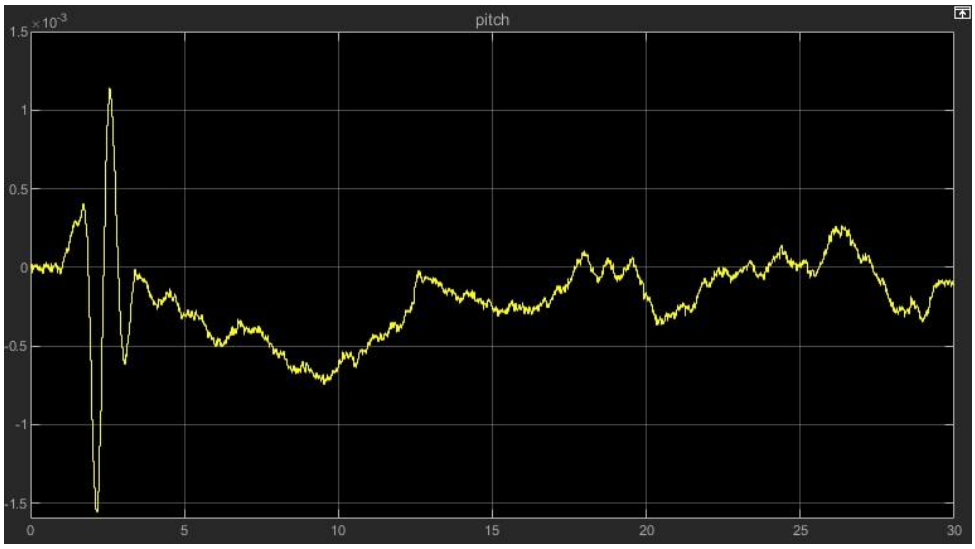


Figura 6.12: Tracciato *pitch* del nuovo controllore (N.L.)

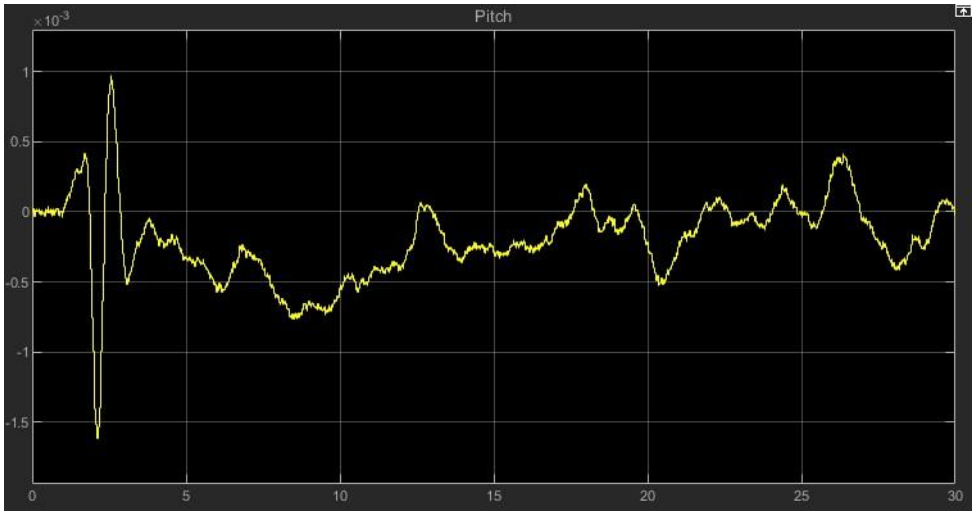


Figura 6.13: Tracciato *pitch* del controllore di default (N.L.)

Capitolo 7

Conclusioni

Lo scopo dell'attività era quello di ottenere un controllore per la variabile x che riuscisse a mantenere il drone fisso nella sua posizione in *hover*.

Dai risultati ottenuti, mostrati nel capitolo precedente, si può osservare come l'andamento nel tempo della variabile x si mantenga all'interno di un intorno molto vicino allo zero (valore di riferimento), in un range di valori compreso tra -5×10^{-3} e 2×10^{-3}, similmente a come accadeva per il controllore di default.

Analogamente, la variabile relativa alla variazione dell'angolo di beccheggio, che come mostrato ampiamente nei capitoli precedenti, risulta essere strettamente legata all'andamento lungo l'asse x del quadricottero, assume, nel corso della simulazione, valori contenuti in un intervallo di estremi $[-10, 4] \times 10^{-3}$, dunque valori molto simili a quelli che assumeva con l'azione del controllore PID.

Le stesse considerazioni possono essere fatte riguardo all'azione di controllo svolta nel sistema non lineare. Risulta, inoltre, interessante notare come la variabile di *pitch*, sia nel caso del controllore (6.15) che in quello originale, presenti un piccolo transitorio di assestamento nella fase iniziale della simulazione.

In conclusione, si può affermare che il controllore costruito riesce a svolgere correttamente il suo compito e che potrebbe essere una valida sostituzione a quello presente originariamente nel modello *Simulink*.

Bibliografia

[1] Pounds, Mahony, Corke, (21 February 2010), *Modelling and control of a large quadrotor robot*, Control Engineering Practice.

[2] Bouabdallah S. (2007), *Design and control of quadrotors with application to autonomous flying*

[3] Isidori A. (1993), *Sistemi di controllo*, Siderea

[4] G. P. Carratelli, M. Del Duca , *Controllo di un quadcopter*

[5] Wei Zhong Fum (September 2015), *Implementation of Simulink controller design on Iris+ quadrotor*

[6] Tomáš Jiřinec (Prague, May 30, 2011), *Stabilization and control of unmanned quadcopter*

[7] Gian Paolo Incremona (6 Giugno 2018), *Modello e controllo di un quadricottero*, Corso di Fondamenti di Automatica