UNIVERSITÀ
POLITECNICA
DELLE MARCHE

Facoltà di Ingegneria
Corso di Laurea in Ingegneria Informatica e dell'automazione

Sintesi del sistema di controllo della traiettoria di un drone: un approccio sliding mode multivariabile

Control system synthesis of a drone's trajectory: a multivariable sliding mode approach

Candidato:
Kevin Javier

Relatore:
Prof. Giuseppe Orlando

Correlatore:
Prof. Gianluca Ippoliti

Anno Accademico 2022-2023

UNIVERSITÀ
POLITECNICA
DELLE MARCHE

Facoltà di Ingegneria
Corso di Laurea in Ingegneria Informatica e dell'automazione

Sintesi del sistema di controllo della traiettoria di un drone: un approccio sliding mode multivariabile

Control system synthesis of a drone's trajectory: a multivariable sliding mode approach

Candidato:
Kevin Javier

Relatore:
Prof. Giuseppe Orlando

Correlatore:
Prof. Gianluca Ippoliti

Anno Accademico 2022-2023

Università Politecnica delle Marche
Facoltà di Ingegneria
Corso di Laurea in Ingegneria Informatica e dell'automazione
Via Brecce Bianche – 60131 Ancona (AN), Italy

Per aspera ad astra

Indice

Elenco delle figure

Elenco delle tabelle

Capitolo 1

Introduzione

I droni sono essenzialmente piccoli aeromobili che possono essere controllati in modo remoto o autonomo, ovvero in assenza di pilota. Molti di essi sono equipaggiati con motori elettrici o a scoppio, eliche o ali fisse, telecamere, sensori e GPS per consentire una vasta gamma di operazioni e funzioni. Per questo motivo questi velivoli sono caratterizzati da un'ampia varietà di forme, dimensioni e funzioni, potendo trovare applicazione in diversi settori, dalla fotgrafia e videografia, all'agricoltura, al monitoraggio ambientale, per non parlare dell'ambito dell'emergenza e soccorso in situazioni di pericolo, oppure nel mondo delle consegne.
Tuttavia, come tutte le tecnologie, un uso irresponsabile e non etico potrebbe condurre a una serie di comportamenti nocivi per la comunità: ad esempio i droni potrebbero rappresentare una minaccia per la sicurezza aerea, potrebbero essere sfruttati in ambito bellico per arrecare danni ad esseri umani, potrebbero catturare dati sensibili e violare la privacy delle persone, senza contare i possibili utilizzi da parte della malavita per scopi illegali.
Dal punto di vista puramente ingegneristico, invece, i droni sono sistemi dinamici di una certà complessità, per via della loro non linearità. Diventa dunque di fondamentale importanza controllarne l'assetto (ovvero la posizione e l'orientazione nello spazio) e la traiettoria (ovvero il movimento nello spazio), con l'obiettivo di garantirne la stabilità.
Oltre alla complessità dovuta alla forte non linearità del sistema, i droni sono soggetti a varie perturbazioni ambientali, come vento, turbolenza e variazioni delle condizioni atmosferiche (disturbi), e ad incertezze che derivano dalla non perfetta conoscenza dei parametri del modello. Per questo motivo, le tecniche di controllo classiche, basate sull'ipotesi di linearità del modello, possono fornire prestazioni non soddisfacenti in molte condizioni operative. Per tener conto di questi aspetti, si deve ricorrere a metodologie di controllo basate su modelli non lineari, che possano inoltre garantire le specifiche di controllo anche in presenza di incertezze (disturbi e/o variazioni parametriche). Fra i possibili approcci di questo tipo, il controllo sliding mode è noto per la sua capacità di mantenere il sistema su una cosiddetta "superficie di scivolamento (sliding surface)", indipendentemente dalle perturbazioni, garantendo una buona stabilità del sistema. Inoltre questa strategia di controllo permette al progettista di tracciare traiettorie complesse, richiedendo al drone di

rispettare particolari specifiche, e garantisce una risposta molto rapida del sistema di controllo, che diviene dunque molto affidabile.

Lo scopo che questo elaborato si propone è presentare il progetto del sistema di controllo per il Parrot Mambo, un drone a quattro eliche, utilizzando il controllo sliding mode, che come è stato appena detto è particolarmente noto per la sua capacità di far fronte a incertezze parametriche e a disturbi non noti, purché limitati.

Presentiamo ora, per sommi capi, i capitoli in cui è strutturata la tesi.
Nel **capitolo 2** si trova una presentazione dei quadricotteri in generale, di cui vengono spiegati la struttura e il comportamento dinamico, per poi arrivare al corrispondente modello matematico.
Nel **capitolo 3** si procede con la descrizione del Parrot Mambo, di cui vengono illustrati i parametri fisici e le componenti.
Nel **capitolo 4** invece si approfondisce l'ambiente Simulink che ci è servito per costruire il sistema di controllo del quadricottero. In particolare è stato utilizzato il progetto asbQuadcopter, contenente la modellazione completa del velivolo in ogni sua componente strutturale e dinamica.
Nel **capitolo 5**, cuore di questo elaborato, viene illustrata la teoria alla base della tecnica sliding mode con la quale è stato configurato il controllo dei gradi di libertà del drone, in particolare degli angoli di Eulero (rollio, beccheggio e imbardata) e della quota.
Nel **capitolo 6**, in seguito, si presentano le modifiche apportate al progetto asbQuadcopter in modo tale da generare i segnali di riferimento e implementare il controllo sliding mode del Parrot Mambo.
Nel **capitolo 7** troviamo i risultati ottenuti dalla simulazione al pc, sui quali sono state effettuate delle valutazioni tramite indici di prestazione, facendo un confronto rispetto al progetto originale che invece sfruttava i regolatori PID.
Nel **capitolo 8**, come ultima cosa, vengono esposte le conclusioni tratte dalle simulazioni sul Parrot Mambo, con uno sguardo verso possibili miglioramenti da attuare in sviluppi futuri.

Capitolo 2

Modellazione matematica del Parrot Mambo

2.1 Struttura e dinamica di un quadricottero

2.1.1 Angoli di Eulero

L'orientamento di un velivolo viene tradizionalmente descritto in aeronautica mediante gli angoli di rollio, beccheggio ed imbardata, comunemente noti come angoli RPY, acronimo dell'inglese roll (rollio), pitch (beccheggio) e yaw (imbardata), come illustrato in figura 2.1.

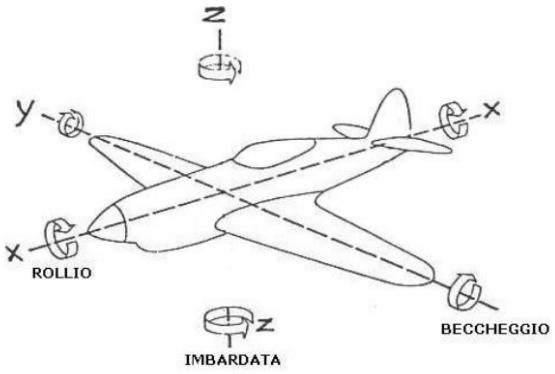
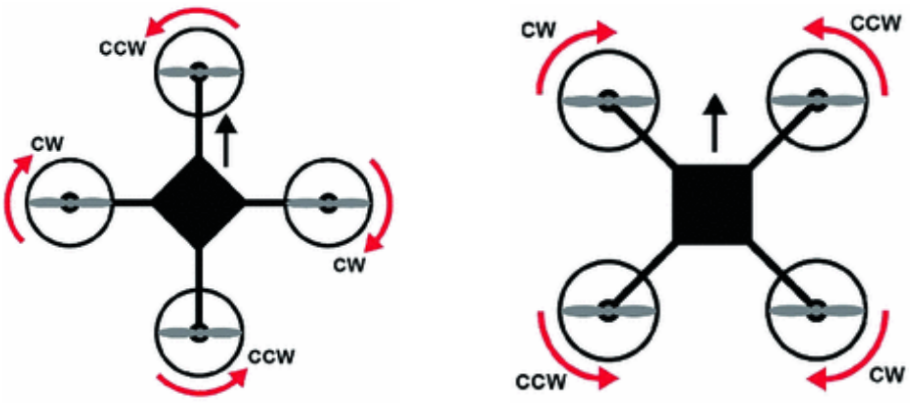


Figura 2.1: Angoli di Eulero

Immaginiamo un sistema di riferimento solidale con il corpo rigido del velivolo, come ad esempio il Parrot Mambo, dove l'asse x è orientato longitudinalmente, l'asse y è orientato trasversalmente e l'asse z è perpendicolare al piano del velivolo. Quando il velivolo compie una rotazione di un angolo ϕ intorno all'asse x, questa è chiamata rollio, mentre una rotazione di un angolo θ intorno all'asse y è chiamata beccheggio. Infine, quando il velivolo ruota di un angolo ψ intorno all'asse z, questo movimento è denominato imbardata. Gli angoli sono definiti positivi quando il drone ruota attorno agli assi x, y e z nel senso positivo, secondo la regola della mano destra.

2.1.2 Dinamica di un quadricottero

Un quadricottero, per definizione, presenta quattro rotori disposti secondo una certa configurazione, in base all'orientamento dei vettori che definiscono il sistema di riferimento solidale con il corpo del drone, noto come BFFR (Body Fixed Frame of Reference); in particolare si possono avere due configurazioni: "a +" o "a croce" (illustrate in figura 2.2). Tuttavia, qualsiasi sia la configurazione utilizzata, essa non va a modificare la perfetta simmetria strutturale del drone né tantomeno le sue prestazioni.



(a) configurazione a più (b) configurazione a croce

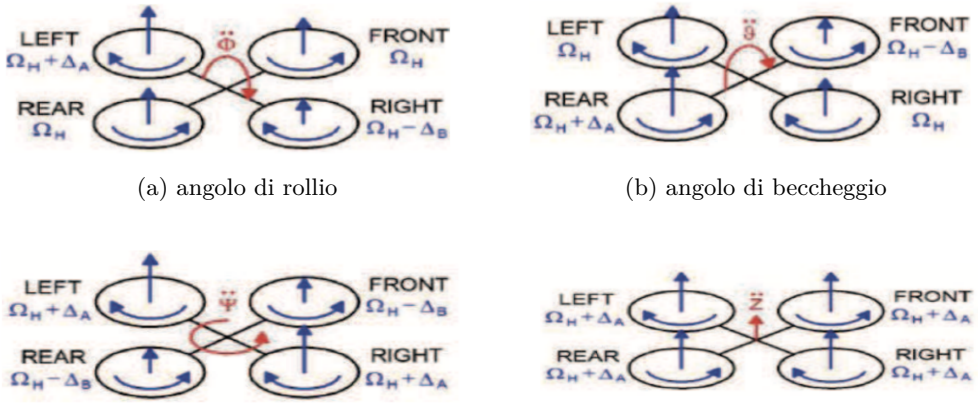
Figura 2.2: Esempio sottofigure.

Il quadricottero rappresenta un classico esempio di sistema sottoattuato: ha infatti 6 gradi di libertà che sono i 3 angoli di Eulero e le 3 coordinate x, y e z, ma è dotato solo di quattro motori. In altre parole, il numero di variabili di controllo è inferiore rispetto al numero di gradi di libertà. Per spostarsi all'interno dello spazio tridimensionale, il drone infatti si basa esclusivamente sulla regolazione della potenza fornita ai quattro motori.
Il concetto è semplice: quando questa potenza è distribuita in modo non uniforme tra i motori, si genera un momento torcente rispetto a un asse che passa attraverso il centro di massa del drone, inducendo una rotazione del drone attorno allo stesso asse.
Grazie a questo meccanismo, è facile far coincidere le nostre variabili di controllo con le spinte che possono essere generate dai motori con l'obiettivo di imporre al drone una certa quota e un certo assetto (il valore assunto dai 3 angoli).
Come mostrato nella figura 2.2, i rotori che si trovano diagonalmente opposti girano in verso contrario, per cui uno gira sempre in senso orario e l'altro in senso antiorario: quando i motori esercitano lo stesso thrust, questa disposizione fa sì che si annullino tutti i momenti torcenti rispetto ai 3 assi. Tuttavia la variazione delle velocità

angolari dei 4 motori induce la generazione di spinte che portano il drone a spostarsi lungo i suoi gradi di libertà:

- generazione di un angolo di yaw: occorre creare una differenza tra le velocità dei rotori che girano in senso orario e quelli che girano in senso antiorario.
- spostamento sull'asse verticale: occorre impostare una velocità di rotazione uniforme tra i quattro motori.
- traslazione sul piano: occorre creare una differenza tra le velocità dei rotori adiacenti; questo andrà a generare un momento torcente che inclinerà il velivolo di un certo angolo di pitch e/o di yaw, con la conseguente traslazione.

Per poter comprendere meglio quanto detto sulla dinamica del velivolo, si consiglia la visione della figura 2.3.



(a) angolo di rollio

(b) angolo di beccheggio

(c) angolo di imbardata

(d) movimento verticale

Figura 2.3: Relazione tra le velocità angolari e i movimenti del drone

2.2 Modellazione matematica di un quadricottero

2.2.1 Ipotesi di modellazione

Dopo aver illustrato la dinamica di un quadricottero, vediamo le assunzioni preliminari di cui abbiamo bisogno per modellare un quadricottero.

- Il drone è un corpo rigido.
- Ogni motore del drone è strutturalmente identico agli altri.
- Coincidenza del centro di massa, del baricentro geometrico e dell'intersezione dei 3 assi del sistema di riferimento solidale con il drone.
- Le spinte e le coppie generate dai rotori sono direttamente proporzionali al quadrato della loro velocità angolare.

- Il drone è un sistema tempo-invariante, cioè i parametri che ne descrivono la struttura (massa, dimensioni) non sono soggetti ad usura nel tempo.
- Perfetta simmetria rispetto ad ognuno dei 3 assi del sistema di riferimento solidale con il drone, che quindi comporta anche avere una matrice d'inerzia diagonale come segue:

$$I = \begin{pmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{pmatrix} \tag{2.1}$$

2.2.2 Sistemi di riferimento

Qualsiasi spostamento o rotazione del quadricottero e qualsiasi forza che influisce sulla sua dinamica possono sempre essere descritti rispetto a due sistemi di riferimento, come illustrato nella Figura 2.4.

- **Sistema di riferimento inerziale (Inertial frame)**: questo sistema di riferimento è inerziale, e i suoi assi tridimensionali (x_i, y_i, z_i) sono definiti rispettivamente in direzione Nord terrestre, Est terrestre e centro della Terra.
- **Sistema di riferimento solidale con il drone (Body frame)**: questo sistema è solidale con il drone stesso. I suoi assi tridimensionali (x_b, y_b, z_b) sono orientati rispettivamente verso la parte anteriore del drone, il lato destro del drone e verso la parte inferiore del drone. L'origine di questo sistema coincide con il centro di massa del drone.

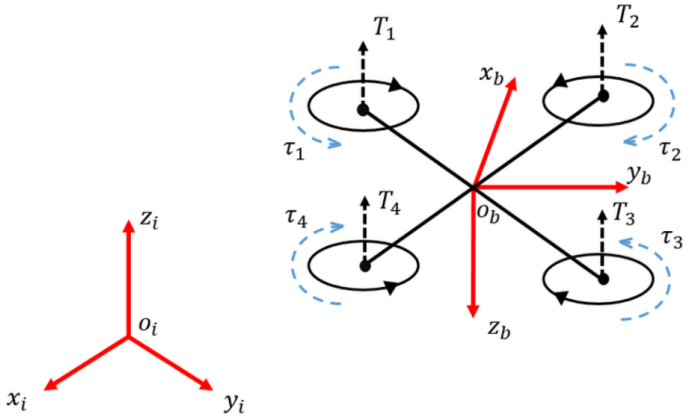


Figura 2.4: Sistemi di riferimento

Rispetto al sistema di riferimento $O_i = (x_i\ y_i\ z_i)$, definiamo $\xi = (x\ y\ z)^T$ come vettore posizione del quadricottero espresso in metri e $\sigma = (\phi\ \theta\ \psi)^T$ come vettore degli angoli di Eulero espresso in radianti, che fornisce l'orientazione del body frame rispetto all'inertial frame.

Rispetto al sistema di riferimento $O_b = (x_b\ y_b\ z_b)$, definiamo $v = (u\ v\ w)^T$ come vettore delle velocità lineari del quadricottero espresso in metri al secondo e $\omega = (p\ q\ r)^T$ come vettore delle velocità angolari espresso in radianti al secondo.
Il passaggio da un sistema di riferimento all'altro si ottiene tramite tre rotazioni consecutive secondo gli angoli di roll, pitch e yaw, esattamente in questo ordine, ottenendo la seguente matrice di rotazione complessiva, la quale consente la conversione dal body frame all'inertial frame rispetto a tutti e tre gli assi di rotazione. La matrice 2.2 rappresenta questa trasformazione di coordinate come $R_b^i : O_b \to O_i$.

$$R_b^i = \begin{pmatrix} c(\psi)c(\theta) & c(\psi)s(\theta)s(\phi) - s(\psi)c(\phi) & c(\psi)s(\theta)c(\phi) + s(\psi)s(\phi) \\ s(\psi)c(\theta) & s(\psi)s(\theta)s(\phi) + c(\psi)c(\phi) & s(\psi)s(\theta)c(\phi) - s(\psi)c(\phi) \\ -s(\theta) & c(\theta)s(\phi) & c(\theta)c(\phi) \end{pmatrix} \tag{2.2}$$

Da notare, inoltre, che R_b^i è una matrice composta da vettori riga linearmente indipendenti, il che implica che sia una matrice ortogonale. In altre parole, soddisfa le condizioni $R_b^i(R_b^i)^T = (R_b^i)^T R_b^i = I$. Questo significa che la matrice R_b^i ha una matrice inversa, che indicheremo come $R_i^b = (R_b^i)^{-1}$, poiché $det(R_b^i) \neq 0$. Inoltre, abbiamo $(R_b^i)^T = (R_b^i)^{-1} = R_i^b$.
Di conseguenza, otteniamo la matrice 2.3, che rappresenta la matrice di rotazione complessiva per la conversione di coordinate dall'inertial frame al body frame, realizzando quindi la trasformazione $R_i^b : O_i \to O_b$.

$$R_i^b = \begin{pmatrix} c(\psi)c(\theta) & s(\psi)c(\theta) & -s(\theta) \\ c(\psi)s(\theta)s(\phi) - s(\psi)c(\phi) & s(\psi)s(\theta)s(\phi) + c(\psi)c(\phi) & c(\theta)s(\phi) \\ c(\psi)s(\theta)c(\phi) + s(\psi)s(\phi) & s(\psi)s(\theta)c(\phi) - c(\psi)s(\phi) & c(\theta)c(\phi) \end{pmatrix} \tag{2.3}$$

Sia ora $\dot{\xi} = (\dot{x}\ \dot{y}\ \dot{z})^T$ il vettore delle velocità lineari del drone rispetto all'inertial frame. La relazione, espressa nella 2.4 tra il vettore delle velocità lineari $\dot{\xi}$ rispetto all'inertial frame ed il vettore delle velocità lineari $\mu = (u\ v\ w)^T$ rispetto al body frame è data dalla matrice R definita sopra:

$$\dot{\xi} = R_b^i\, \mu \tag{2.4}$$

Allo stesso modo possiamo trovare la relazione, espressa nella 2.5, tra il vettore delle velocità angolari $\omega = (p\ q\ r)^T$ rispetto al body frame ed il vettore delle velocità angolari $\dot{\sigma} = (\dot{\phi}\ \dot{\theta}\ \dot{\psi})^T$ rispetto all'inertial frame, data dalla matrice T (che riportiamo nella 2.6) come segue:

$$\dot{\sigma} = T_b^i\, \omega \tag{2.5}$$

dove:

$$T_b^i = \begin{pmatrix} 1 & s(\phi)t(\theta) & c(\phi)t(\theta) \\ 0 & c(\phi) & -s(\phi) \\ 0 & \frac{s(\phi)}{c(\theta)} & \frac{c(\phi)}{c(\theta)} \end{pmatrix} \tag{2.6}$$

La conversione delle velocità angolari dal body frame all'inertial frame viene effettuata invece dalla matrice di rotazione $T_i^b = (T_b^i)^{-1}$ definita nella 2.7).

$$T_i^b = \begin{pmatrix} 1 & 0 & -s(\theta) \\ 0 & c(\phi) & c(\theta)s(\phi) \\ 0 & -s(\phi) & c(\theta)c(\phi) \end{pmatrix} \tag{2.7}$$

2.2.3 Equazioni del modello dinamico

Per arrivare a determinare le equazioni che modellano la dinamica di un quadricottero ci avvaliamo della seconda legge della dinamica di Newton applicata ai corpi rigidi per ricavare le equazioni 2.8 e 2.9, espresse rispetto al body frame, che descrivono l'accelerazione lineare ed angolare del velivolo a causa delle forze rispettivamente traslazionali e rotazionali agenti su di esso.

$$m\dot{\mathrm{v}} = -m\omega \times \mathrm{v} + mgR_i^b e_3 - \frac{1}{2}\rho C_d A \mathrm{v}||\mathrm{v}|| + \sum_{i=1}^{4} T_i \tag{2.8}$$

$$I\dot{\omega} = -\omega \times I\omega - M_g + -M_{td} \tag{2.9}$$

Nella 2.8 possiamo notare le forze che agiscono sul sistema, cioè:

- la forza di attrito viscoso lineare (drag force), in quanto il drone si trova immerso in un fluido che ne ostacola il movimento. Infatti, come si può notare nella 2.10, la forza di drag ha stessa direzione ma verso opposto alla velocità lineare del velivolo, e dipende dalla densità ρ del fluido, dall'area A della superficie impattata dal fluido, dal coefficiente di drag C_d e dal quadrato della velocità lineare del drone.
$$F_d = -\frac{1}{2}\rho C_d A \mathrm{v}||\mathrm{v}|| \tag{2.10}$$
- la spinta (thrust) complessiva generata dai quattro motori, definita dalla sommatoria $\sum_{i=1}^{4} T_i$, dove, trascurando il fenomeno del blade flapping:
$$T_i = b\omega_i^2 \begin{pmatrix} 0 \\ 0 \\ -1 \end{pmatrix} \tag{2.11}$$
con:
$$b = C_T \rho A r^2 \tag{2.12}$$
- la forza di gravità, che agisce solo sulla componente verticale del sistema (si può notare dalla presenza del versore che rappresenta la direzione verticale nell'inertial frame, moltiplicato poi per la matrice di rotazione R_i^b al fine di convertire la forza nel body frame).

Nella 2.9 possiamo notare i momenti torcenti che agiscono sul sistema, cioè:

- M_g, momento torcente giroscopico dovuto all'inerzia J_r dei rotori del quadricottero.

$$M_g = \begin{pmatrix} -J_r\Omega_r q \\ J_r\Omega_r p \\ 0 \end{pmatrix} \tag{2.13}$$

dove Ω_r viene utilizzato per definire il concetto di "residuo" di velocità angolare dei rotori, vale a dire la velocità angolare complessiva che è orientata nel senso positivo di rotazione intorno all'asse di imbardata, tenendo conto delle velocità angolari di tutti e quattro i rotori e delle rispettive direzioni di rotazione.

$$\Omega_r = \omega_1 - \omega_2 + \omega_3 - \omega_4 \tag{2.14}$$

- M_{td}, momento torcente dovuto alla spinta dei motori e alla forza di attrito viscoso.

$$M_{td} = \begin{pmatrix} lb(\omega_1^2 - \omega_2^2 - \omega_3^2 + \omega_4^2) \\ lb(\omega_1^2 + \omega_2^2 - \omega_3^2 - \omega_4^2) \\ d(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2) \end{pmatrix} \tag{2.15}$$

dove l è la distanza tra ogni motore e il centro di massa, b è la costante di spinta e d è la costante di drag.

Ora che tutte le forze e i momenti che agiscono sul sistema sono stati definiti, la complessa dinamica del drone a 6 gradi di libertà può essere pienamente rappresentata dalle seguenti equazioni differenziali non lineari, entrambe espresse rispetto all'inertial frame, dove la prima (2.16) è il sottosistema traslazionale mentre la seconda (2.17) è il sottosistema rotazionale.

$$\begin{pmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{pmatrix} = \begin{pmatrix} -\frac{1}{2}\rho C_d A \mathrm{v}||\mathrm{v}||e_1 + \frac{1}{m}(s(\phi)s(\psi) + c(\psi)s(\theta)c(\phi)) \sum_{i=1}^{4} b\omega_i^2 \\ -\frac{1}{2}\rho C_d A \mathrm{v}||\mathrm{v}||e_2 - \frac{1}{m}(s(\phi)c(\psi) + s(\psi)s(\theta)c(\phi)) \sum_{i=1}^{4} b\omega_i^2 \\ -g - \frac{1}{2}\rho C_d A \mathrm{v}||\mathrm{v}||e_3 + \frac{1}{m}c(\phi)c(\theta) \sum_{i=1}^{4} b\omega_i^2 \end{pmatrix} \tag{2.16}$$

$$\begin{pmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{pmatrix} = \begin{pmatrix} \frac{1}{I_{xx}}[-qr(I_{zz} - I_{yy}) - J_r\Omega_r q + lb(\omega_1^2 - \omega_2^2 - \omega_3^2 + \omega_4^2)] \\ \frac{1}{I_{yy}}[-pr(I_{xx} - I_{zz}) + J_r\Omega_r p + lb(\omega_1^2 + \omega_2^2 - \omega_3^2 - \omega_4^2)] \\ \frac{1}{I_{zz}}[-pq(I_{yy} - I_{xx}) + d(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2)] \end{pmatrix} \tag{2.17}$$

Tuttavia preferiremmo avere la dinamica rotazionale del quadricottero espressa rispetto al body frame, in quanto siamo interessati a controllare i 3 angoli di Eulero, che sono espressi rispetto al sistema di riferimento solidale con il drone.
Inoltre, ipotizzando che il drone lavori in condizioni di hovering e che dunque gli angoli di roll e pitch siano molto piccoli, la matrice T_i^b (2.7)per la conversione delle velocità dall'inertial frame al body frame si può approssimare con una matrice identità, per

cui le 2.16 e 2.17 diventeranno:

$$\begin{pmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{pmatrix} = \begin{pmatrix} -\frac{1}{2}\rho C_d A \mathrm{v}||\mathrm{v}||e_1 + (s(\phi)s(\psi) + c(\psi)s(\theta)c(\phi))\frac{U_z}{m} \\ -\frac{1}{2}\rho C_d A \mathrm{v}||\mathrm{v}||e_2 - (s(\phi)c(\psi) + s(\psi)s(\theta)c(\phi))\frac{U_z}{m} \\ -g - \frac{1}{2}\rho C_d A \mathrm{v}||\mathrm{v}||e_3 + c(\phi)c(\theta)\frac{U_z}{m} \end{pmatrix} \tag{2.18}$$

$$\begin{pmatrix} \ddot{\phi} \\ \ddot{\theta} \\ \ddot{\psi} \end{pmatrix} = \begin{pmatrix} \frac{1}{I_{xx}}[-qr(I_{zz} - I_{yy}) - J_r\Omega_r q + U_\phi] \\ \frac{1}{I_{yy}}[-pr(I_{xx} - I_{zz}) + J_r\Omega_r p + U_\theta] \\ \frac{1}{I_{zz}}[-pq(I_{yy} - I_{xx}) + U_\psi] \end{pmatrix} \tag{2.19}$$

dove sono state effettuate le seguenti sostituzioni, in modo tale da rendere visibili le variabili di controllo (ovvero le spinte) che verranno utilizzate per il controllo del quadricottero:

$$\begin{pmatrix} U_z \\ U_\phi \\ U_\theta \\ U_\psi \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^{4} b\omega_i^2 \\ lb(\omega_1^2 - \omega_2^2 - \omega_3^2 + \omega_4^2) \\ lb(\omega_1^2 + \omega_2^2 - \omega_3^2 - \omega_4^2) \\ d(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2) \end{pmatrix} \tag{2.20}$$

Capitolo 3

Descrizione del Parrot Mambo

Il quaricottero di riferimento in esame all'interno di questo lavoro di tesi è il Parrot Mambo minidrone (Figura 3.1), selezionato per via delle sue piccole dimensioni che lo rendono molto agile e la sua compatibilità con l'ambiente Matlab/Simulink, consentendo anche di configurare prima e caricare poi i controllori dei gradi di libertà via Bluetooth all'interno del Flight Control System.

Figura 3.1: Parrot Mambo minidrone

3.1 Componenti del Parrot Mambo

- **Dimensioni**: 18×18×5 cm (con carene).
- **Peso**: 63g (senza carene e accessori).
- **Energia**: - batteria LiPo 660 mAh.
  - 8 min di autonomia con carene e/o accessori.
  - 10 min di autonomia senza carene e accessori.
  - 30 min di ricarica con caricatore da 2.1 Ampere.
- **Sensoristica**:
  - sensore ad ultrasuoni per misura dell'altitudine (fino a 4 m di quota).

  - sensore di pressione barometrica di ausilio al sensori ad ultrsuoni.
  - IMU (unità di misura inerziale) composta da un accelerometro a 3 assi ed un giroscopio a 3 assi.
  - telecamera verticale da 60 FPS (frames per second).

- **Motori**: 4 motori coreless, ovvero privi di un nucleo di ferro nel rotore
  - 2 motori A (Anti-Clock).
  - 2 motori C (Clock).

3.2 Parametri fisici del Parrot Mambo

Tabella 3.1: Parametri fisici del Parrot Mambo

Parametro	**Simbolo**	**Valore**	**Unità di misura**
Massa	m	0.063	kg
Distanza motore-centro di massa	l	0.0624	m
Altezza dei rotori rispetto al centro di massa	h	0.0159	m
Costante di spinta	b	0.0107	Ns^2
Costante di drag	d	0.78264×10^{-3}	Nms^2
Momento d'inerzia del roll	I_{xx}	0.582857×10^{-4}	kgm^2
Momento d'inerzia del pitch	I_{yy}	0.716914×10^{-4}	kgm^2
Momento d'inerzia dello yaw	I_{zz}	0.1×10^{-3}	kgm^2
Momento d'inerzia del rotore dei motori	J_r	0.1021×10^{-6}	kgm^2

3.3 Immagini descrittive del Parrot Mambo

Nell'immagine 3.2, possiamo notare che il drone presenta due led, che ricordano due occhi, che assumono un colore diverso in base allo stato del drone:

- verde fisso : il drone è pronto
- rosso fisso : è stato rilevato un problema
- arancione fisso : il drone è in avviamento
- rosso lampeggiante : la batteria è scarica

Nell'immagine 3.3, invece, possiamo osservare:

- l'alloggiamento che ospita la batteria di alimentazione
- la porta micro-usb per la ricarica ed il collegamento al computer
- il pulsante di accensione/spegnimento
- la fotocamera verticale

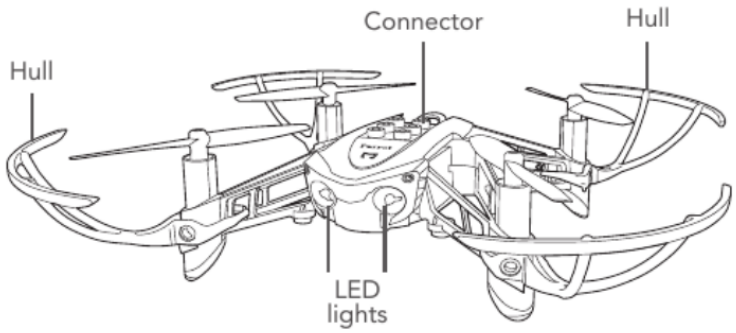


Figura 3.2: Vista anteriore e superiore

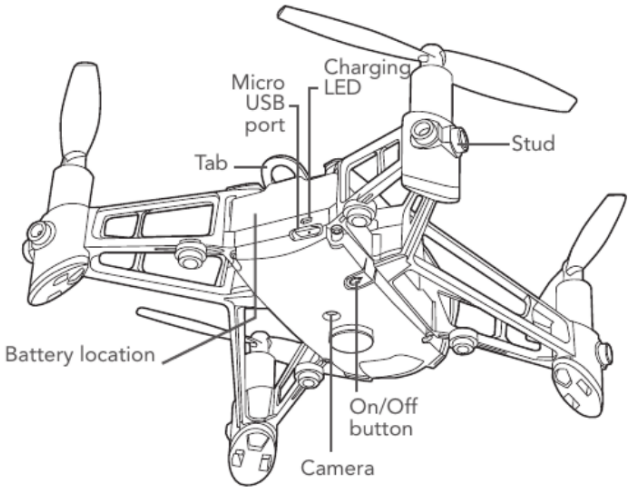


Figura 3.3: Vista posteriore e inferiore

Capitolo 4

Modellazione tramite il progetto asbQuadcopter in Simulink

Ora che abbiamo ricavato il modello matematico che descrive la dinamica del Parrot Mambo tramite un sistema di equazioni non lineari, passiamo a mostrare il progetto asbQuadcopter di Simulink, che consente la modellazione del Parrot Mambo, al cui interno sono state eseguite delle modifiche per implementare la struttura di controllo realizzata in sliding mode (argomento del Capitolo 5).

4.1 Presentazione del progetto asbQuadcopter

Per creare un nuovo progetto asbQuadcopter è sufficiente digitare il comando *asbQuadcopterStart* sulla command window di Matlab: il progetto viene salvato di default all'interno della directory 'MATLAB/projects/examples' del proprio PC, la quale viene creata automaticamente qualora non sia già presente.
Per poter utilizzare il progetto è fondamentale installare i seguenti Toolbox di Matlab:

- Aerospace Blockset
- Aerospace Toolbox
- Computer Vision Toolbox
- Control System Toolbox
- Image Processing Toolbox
- Signal Processing Toolbox
- Simulink 3D Animation
- Simulink Coder

All'avvio del progetto, vengono inizializzate tutte le variabili di ambiente del modello e l'utente può visualizzare le interfacce che raffigurano il progetto Simulink complessivo (Figura 4.1), l'animazione 3D per la visualizzazione delle prestazioni del drone durante la simulazione (Figura 4.3) e l'Instrument Panel per monitorare lo stato del drone

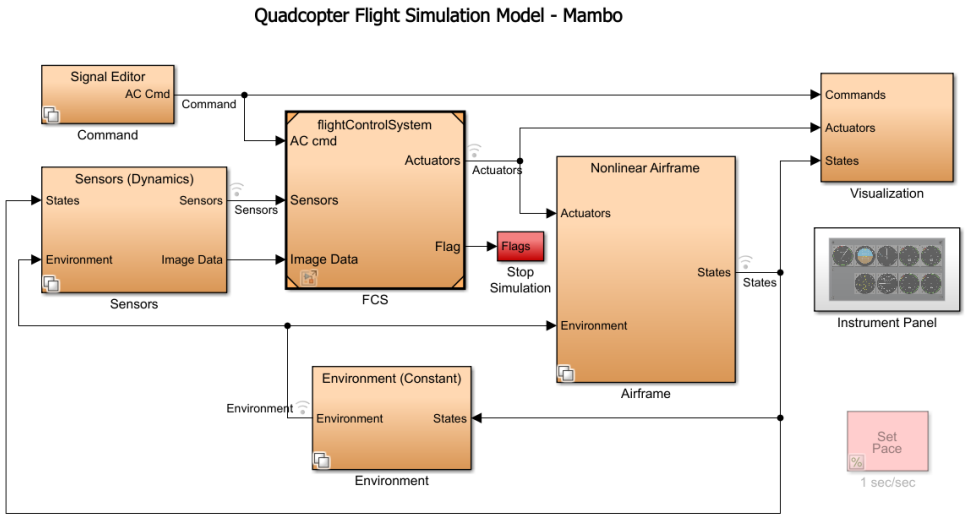


Figura 4.1: Schema complessivo

Figura 4.2: Instrument panel

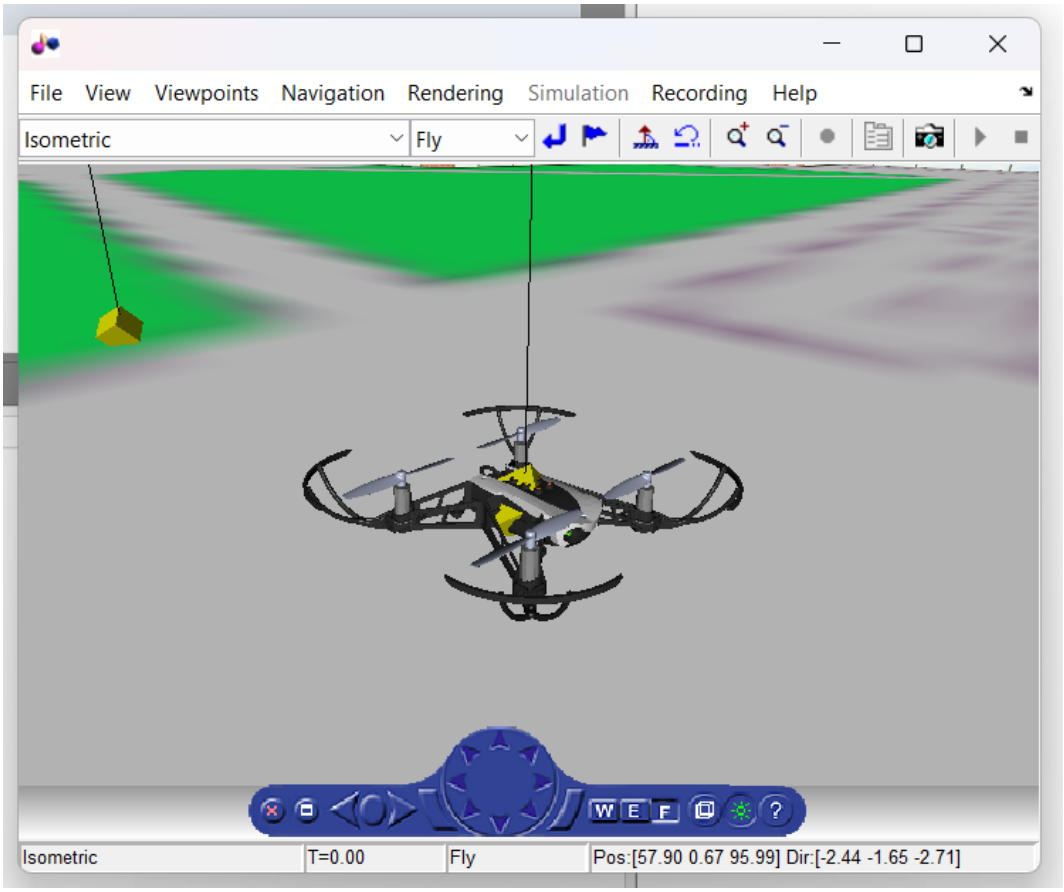


Figura 4.3: Animazione 3D

durante il volo (Figura 4.2).

La Figura 4.1 mostra i blocchi macroscopici che compongono il nostro sistema-quadricottero:

- il blocco **AirFrame**, ovvero il processo da controllare
- il blocco **FCS**, ovvero il sistema di controllo
- il blocco **Sensors**, ovvero l'insieme dei sensori
- il blocco **Command**, ovvero il generatore degli ingressi di riferimento relativi alle variabili controllate
- il blocco **Environment**, che riproduce l'ambiente circostante e contiene le variabili di ambiente
- il blocco **Flags**, il cui scopo è segnalare uno stato pericoloso per il drone, ad esempio un'inclinazione o una velocità eccessive o il contatto con il terreno, per poi forzare l'interruzione del volo.
- il blocco **Visualization**, che permette di visualizzare i valori dei riferimenti assegnati, degli sforzi di controllo degli attuatori e delle variabili di stato del drone.
- il blocco **Instrument Panel**, contenente tutti i dati di volo del quadricottero.

Si individuerà facilmente, tra i primi 4 macro-blocchi, il celebre schema di controllo in controreazione: il controllore, utilizzando gli ingressi di riferimento impartiti, progetta infatti la legge di controllo tramite retroazione dello stato del processo, ottenuto dalle misure dei sensori, tenendo conto dell'ambiente circostante.

4.2 Processo da controllare

Nel blocco Airframe si trovano due sottoblocchi, Linear Airframe e Nonlinear Airframe, rappresentanti rispettivamente il modello linearizzato e non lineare del quadricottero (Figura 4.4). La scelta tra i due modelli viene effettuata impostando la variabile globale VSS_VEHICLE, inizializzata di default ad 1 (modello non lineare) e caricata sul workspace automaticamente durante l'avvio del progetto asbQuadcopter.

- VSS_VEHICLE $= 1 \rightarrow$ modello non lineare
- VSS_VEHICLE $= 0 \rightarrow$ modello lineare

Entrambi i blocchi presentano due ingressi (lo sforzo di controllo degli attuatori generato dal controllore e le variabili di ambiente) e un'uscita (lo stato del quadricottero). Passiamo ora a descrivere il blocco utilizzato per implementare il controllo sliding mode, cioè il NonLinear Airframe.

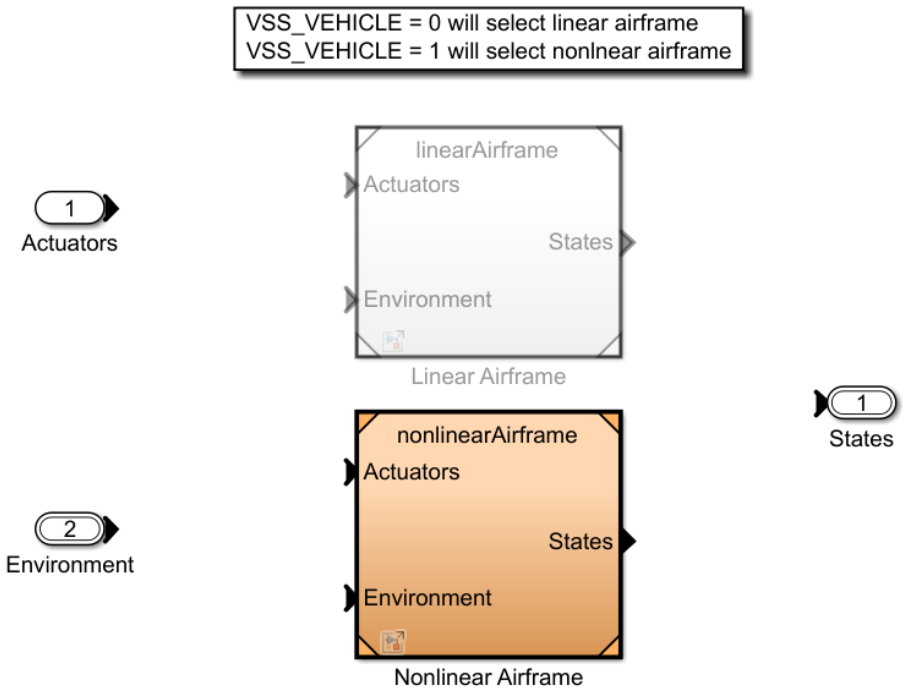


Figura 4.4: Airframe

4.2.1 Processo non lineare

Come si può osservare in figura 4.5, il modello non lineare del drone consiste in 4 blocchi:

AC model, rappresentato in figura 4.6 contiene la modellazione matematica della dinamica del quadricottero. Questo blocco, conoscendo le spinte eseguite dai motori, le variabili d'ambiente, la matrice DCMbe (Direct Cosine Matrix) e le velocità lineari ed angolari rispetto al body frame del drone, restituisce le forze e i momenti torcenti rispetto ai 3 assi principali del drone.

6DOF (Quaternion) è il blocco che, sulla base delle forze e dei momenti che agiscono sul quadricottero, determina tutte le grandezze cinematiche di interesse del drone che costituiscono lo stato complessivo del velivolo (posizioni, angoli, velocità, accelerazioni).

- V_e: velocità traslazionale rispetto all'inertial frame (anche detto Earth Fixed Frame).
- X_e: posizione nello spazio rispetto all'inertial frame.
- ϕ, θ, ψ: angoli di Eulero, che descrivono l'orientamento del body frame del drone rispetto all'inertial frame attraverso una serie di rotazioni effettuate

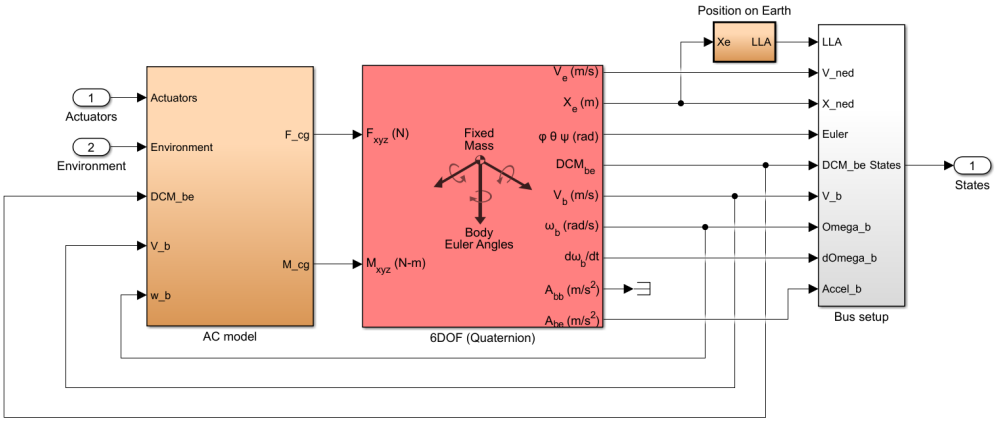


Figura 4.5: Nonlinear airframe

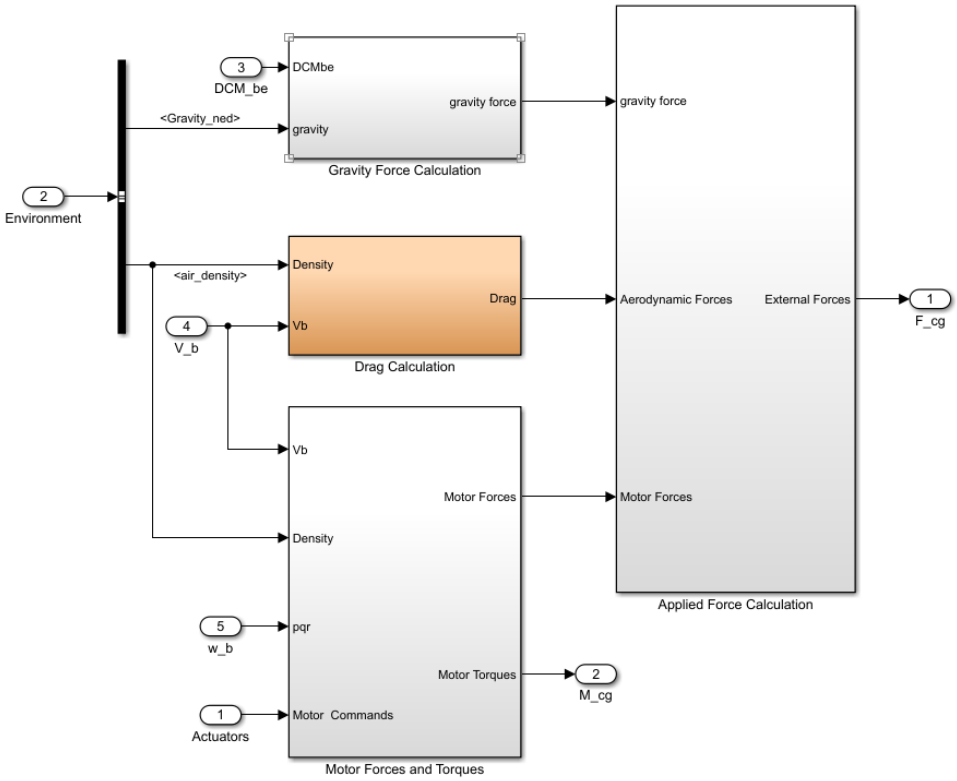


Figura 4.6: AC model

attorno agli assi x, y e z di quest'ultimo.

- DCM_{be}: coefficienti della Direct Cosine Matrix, matrice per la conversione di coordinate dall'inertial frame al body frame.
- V_b: velocità traslazionale rispetto al body frame.
- ω_b : velocità angolare rispetto al body frame.
- $\frac{d\omega_b}{dt}$: accelerazione angolare rispetto al body frame.
- A_{bb}: accelerazione traslazionale lungo gli assi del body frame espressa rispetto al body frame stesso.
- A_{be}: accelerazione traslazionale lungo gli assi del body frame espressa rispetto all'inertial frame.

Il nome 6DOF (Degrees Of Freedom) è associato con il termine Quaternion in quanto utilizza un'estensione dei numeri complessi, i quaternioni, rappresentati da un'espressione del tipo a + bi + cj + dk: l'aggiunta delle altre due componenti è necessaria per evitare problemi legati alla presenza di singolarità nella rappresentazione degli angoli di Eulero.

Bus setup, che troviamo in figura 4.7 è il blocco che, ricevendo le grandezze cinematiche dal blocco 6DOF (Quaternion), le accorpa andando a costruire con un multiplexer un unico vettore di stato del velivolo, il quale verrà poi trasferito al controllore grazie al bus di controreazione.

Position on Earth è il blocco che, partendo dalla posizione X_e del drone rispetto all'inertial frame, produce le coordinate geodetiche del drone (latitudine, longitudine e altitudine), tenendo conto della curvatura terrestre. L'effettiva conversione da Flat Earth Frame a LLA (Latitude Longitude Altitude) avviene all'interno del blocco Flat Earth to LLA (Figura 4.8).

Entriamo ora nello specifico, andando a descrivere la struttura del blocco AC model, a sua volta costituito da 4 blocchi:

Gravity Force Calculation, come si può osservare in figura 4.9, è il blocco che calcola le 3 componenti della forza di gravità rispetto body frame : per attuare la conversione da inertial frame a body frame viene sfruttata la Direct Cosine Matrix DCM_{be}.

Drag Calculation, che troviamo in figura 4.10, è il blocco che calcola le 3 componenti della forza di attrito viscoso (descritto dalla 2.10) dovuto all'aria che si oppone al moto del quadricottero: infatti ha stessa direzione e verso opposto rispetto al

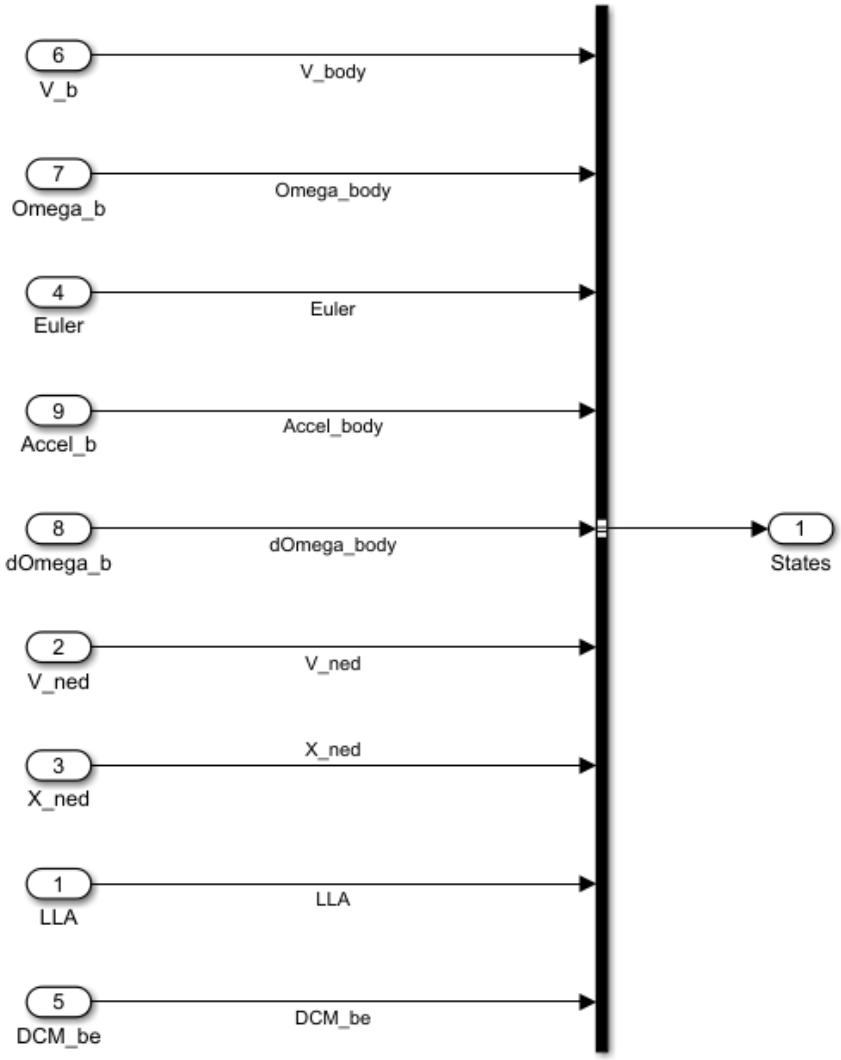


Figura 4.7: Bus setup

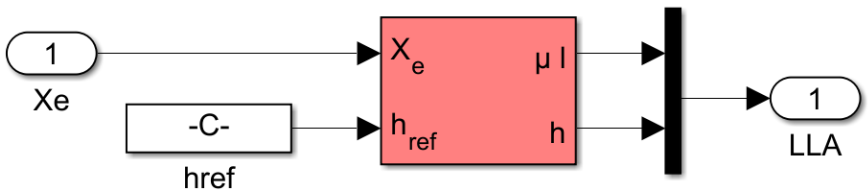


Figura 4.8: Position on earth

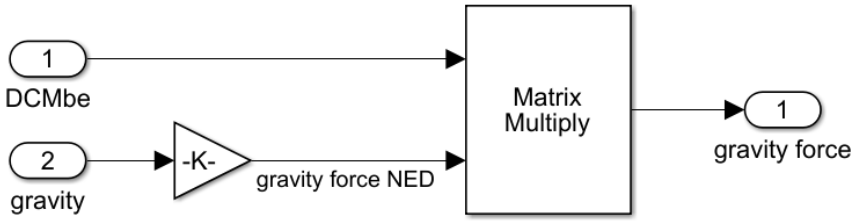


Figura 4.9: Gravity force calculation

vettore di velocità lineare Vb mentre il suo modulo dipende dal coefficiente di drag Cd, dalla densità ρ del fluido che il drone attraversa, dall'area A della superficie del drone che impatta il fluido durante il volo e dalla sua velocità lineare Vb (figura 4.11).
L'area A della superficie del drone che impatta l'aria durante il volo viene calcolata come un vettore a 3 componenti, di cui le prime due rappresentano l'area delle superfici anteriore (e posteriore) e laterali del drone, mentre l'ultima rappresenta l'area della superficie superiore (ed inferiore). Nel calcolo delle 3 componenti, il quadricottero viene considerato come un parallelepipedo in cui le superfici superiore e inferiore siano quattro volte maggiori rispetto le superfici anteriore e posteriore e la superficie laterale.

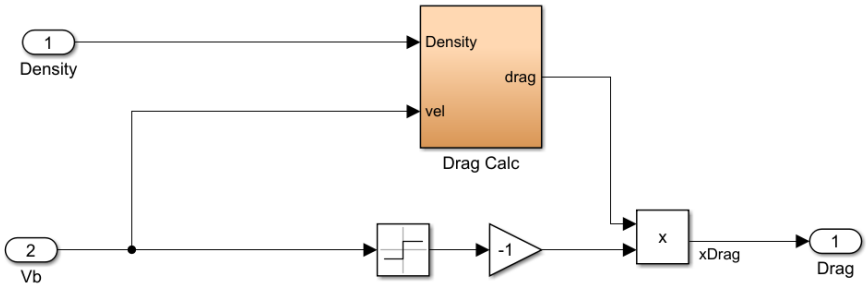


Figura 4.10: Drag calculation

Motor Forces and Torques è il blocco che calcola tutte le forze traslazionali e rotazionali applicate al corpo del drone rispetto al body frame.
In particolare possiamo vedere come esso sia costituito da due blocchi:

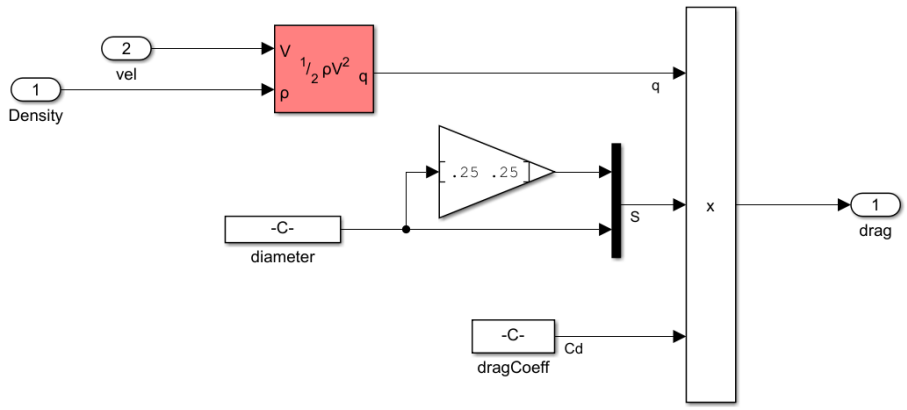


Figura 4.11: Drag calc

- blocco **MotorsToW**: ha il compito di convertire i comandi inviati ai motori (Motor Commands) in segnali interpretabili dagli stessi, cosicché gli attuatori possano generare le velocità di rotazione ω desiderate. Lo troviamo in figura 4.12.

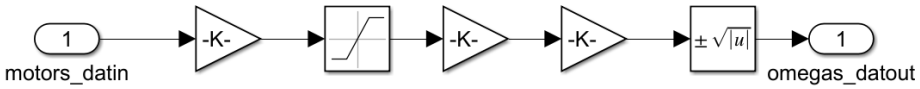


Figura 4.12: MotorsToW

- blocco **For Each Subsystem**: implementa un ciclo 'for each' attraverso il quale vengono calcolate tutte le forze e i momenti esercitati da ciascun motore del drone: per determinare queste grandezze utilizza l'uscita del blocco MotorsToW, ovvero le velocità angolari dei motori, a cui si aggiungono la velocità lineare V_b del drone, la densità ρ dell'aria, le velocità angolari p q r rispetto agli assi x y z del body frame e i vettori delle distanze dei motori dal centro di massa del velivolo. Lo troviamo in figura 4.13.

Applied Force Calculation, come si può osservare in figura 4.14 è il blocco che calcola rispetto al body frame la risultante di tutte le forze lineari che agiscono sul drone, ovvero la gravità, l'attrito viscoso e le spinte dei motori.

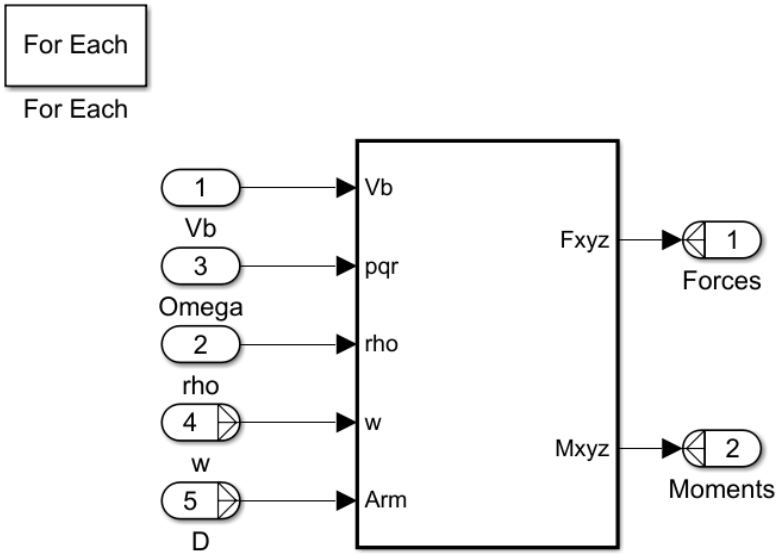


Figura 4.13: ForEachSubsystem

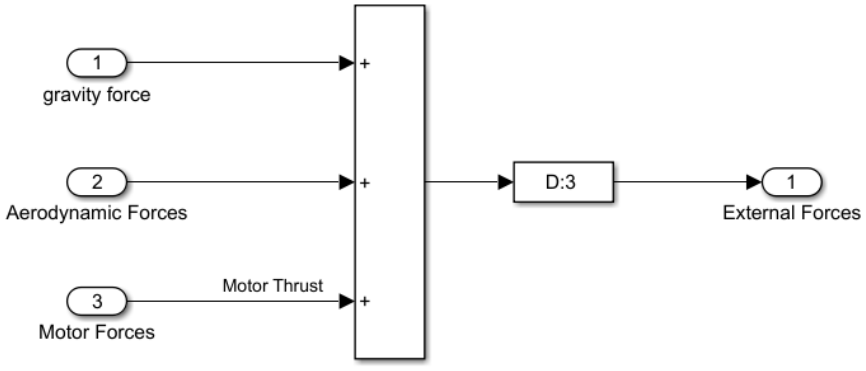


Figura 4.14: Applied Force Calculation

4.3 Generatore dei segnali di riferimento

Il blocco Command è adibito alla generazione degli ingressi di riferimento che vengono inviati al sistema di controllo, per fare sì che il drone segua la traiettroia desiderata. Lo troviamo qui di seguito nella figura 4.15.

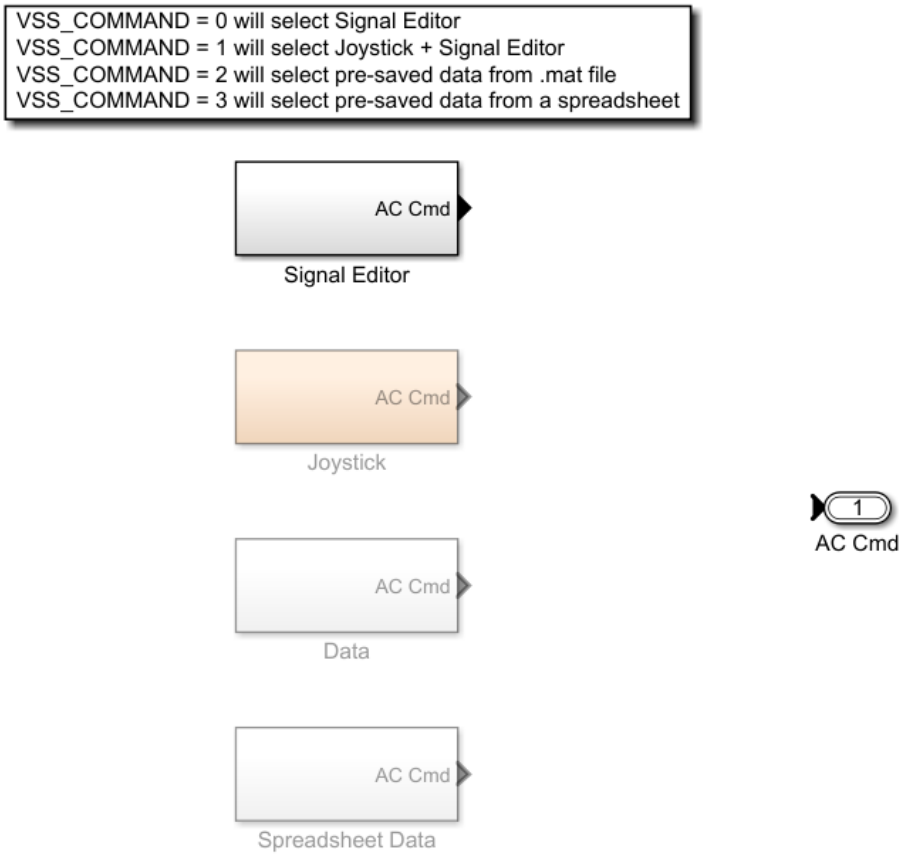


Figura 4.15: Command

Il modo in cui i segnali di riferimento vengono impartiti al velivolo dipende da come viene impostata la variabile VSS_COMMAND, caricata sul Workspace di Matlab dal progetto asbQuadcopter: in base al valore che essa assume, come si può notare nella figura sottostante, viene selezionato il blocco per la generazione dei riferimenti:

- **Signal Editor** permette di fornire i riferimenti tramite l'utilizzo del blocco Signal Editor predefinito di MATLAB.
- **Joystick + Signal Editor** permette di fornire i riferimenti di roll, pitch e yaw per mezzo di un joystick, e inoltre di stabilire riferimenti anche per le coordinate x, y e z con un blocco Signal Editor.

- **MAT Data** permette di fornire i riferimenti memorizzati all'interno di un MAT file (.mat).

- **Spreadsheet Data** permette di fornire i riferimenti memorizzati all'interno di un foglio di calcolo Excel (.xlsx).

In questo elaborato l'assegnazione degli ingressi di riferimento è stata realizzata impostando la variabile VSS_COMMAND a 0, ovvero editando manualmente i segnali. Inizialmente nel Signal Editor (Figura 4.16), era contenuto il blocco Position/Attitude Reference che generava i riferimenti per tutti e 6 i gradi di libertà. In seguito questo blocco è stato sostituito con un altro blocco Signal Builder, che incorpora un'interfaccia utente molto semplice da utilizzare. I riferimenti generati sono mostrati all'interno del capitolo 6.

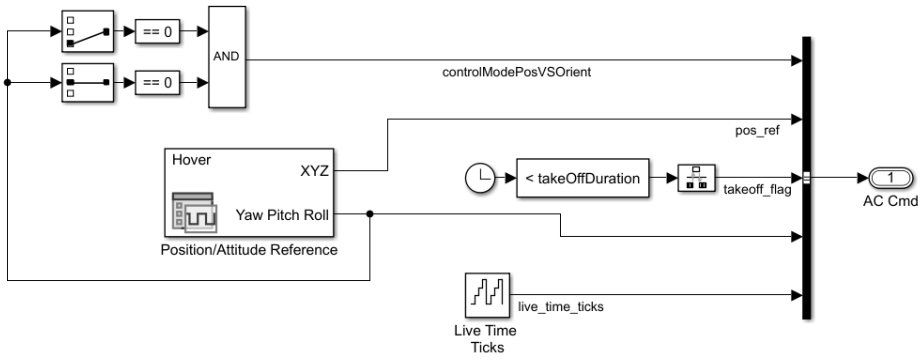


Figura 4.16: Signal editor

L'uscita AC Cmd rappresenta il vettore di variabili, creato con il blocco Bus Creator, che, tramite il bus, giunge al controllore del quadricottero; è strutturato come segue:

- *controlModelPosVSOrient*: variabile booleana che, se 'true' indica che i riferimenti di pitch e roll in quell'istante sono entrambi nulli (condizione di hover) e che il drone deve anche rispettare i riferimenti relativi alle variabili x e y; invece, se è valorizzata come 'false', viene comandato al drone di inseguire gli andamenti stabiliti dal generatore dei riferimenti, indipendentemenete dalle traiettorie x e y desiderate.

- *pos_ref*: riferimenti per le coordinate X, Y e Z.

- *takeoff_flag*: variabile booleana che rimane a 'true' durante il decollo del drone, ovvero per il primo secondo di volo (a livello di algoritmo, la variabile di *take_off* è true finché il valore del clock di sistema è inferiore alla variabile *takeOffDuration* del Workspace, pari ad 1 secondo), mentre quando diventa

'false' significa che viene impartito il riferimento della quota generato dal blocco Signal Builder.

- *orient_ref*: riferimenti per yaw, pitch e roll.
- *live_time_ticks*: segnale di clock che funge da contatore, generato dal blocco Live Time Ticks.

4.4 Sistema di controllo

Il blocco FCS è il blocco adibito alla stabilizzazione e al controllo del drone. Inoltre, questo sarà l'unico blocco che potrà essere trasferito effettivamente sul Parrot Mambo e codificato in linguaggio C, per poter pilotare il velivolo.

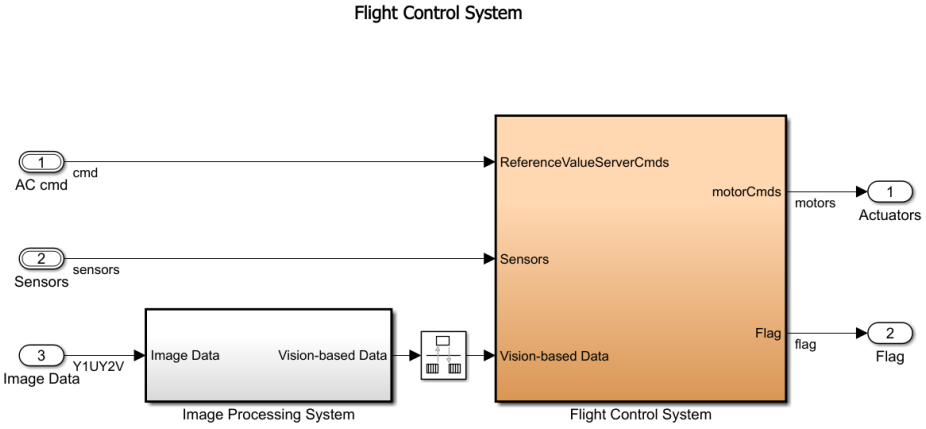


Figura 4.17: FCS

Come si osserva in Figura 4.17, esso è composto a sua volta da due grandi blocchi:

- **Image Processing System** è il blocco dedicato al processamento delle immagini ricevute a 60 FPS dalla telecamera verticale del drone. Queste immagini vengono in seguito utilizzate come dati per poter stimare la quota e guidare il drone nell'inseguimento della traiettoria Xy assegnatagli.
- **Flight Control System** è il blocco contenente tutta logica di controllo e stabilizzazione del quadricottero: esso infatti, ricevendo in ingresso i segnali di riferimento dal blocco Command, le informazioni raccolte dai sensori e le informazioni provenienti dall'Image Processing System, restituisce lo sforzo di controllo di ciascuno dei 4 attuatori del drone ed il valore della variabile di flag, utile per disattivare immediatamente i motori nel caso in cui il drone si imbatta in situazioni potenzialmente pericolose o semplicemente indesiderate, come collisioni con l'ambiente circostante o un'inclinazione particolarmente

problematica.
Questo blocco è stato cruciale nel lavoro svolto in quanto è dove sono state implementate le leggi di controllo mediante la tecnica sliding mode, andando a sostituire i controllori preesistenti nel progetto (argomento del capitolo 6).

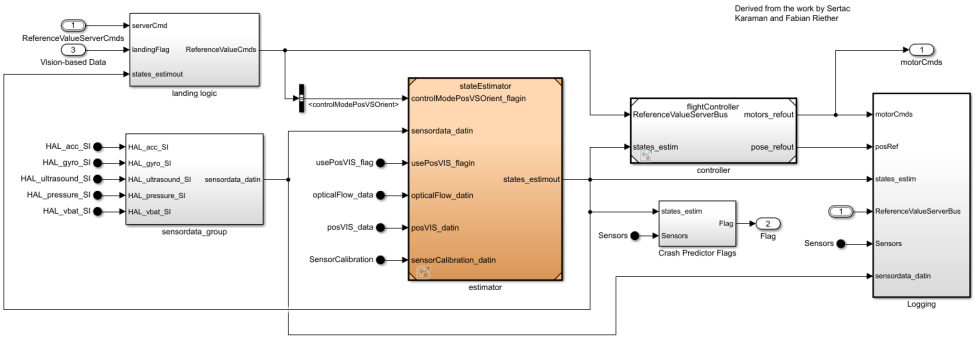


Figura 4.18: Flight Control System

Si può osservare, in figura 4.18, come il Flight Control System sia organizzato con la seguente struttura:

landing_logic è il blocco che, utilizzando le informazioni relative allo stato e le immagini processate dalla telecamera, gestisce la logica di atterraggio del drone, inviando in uscita i segnali di riferimento al blocco Controller. (Figura 4.19).

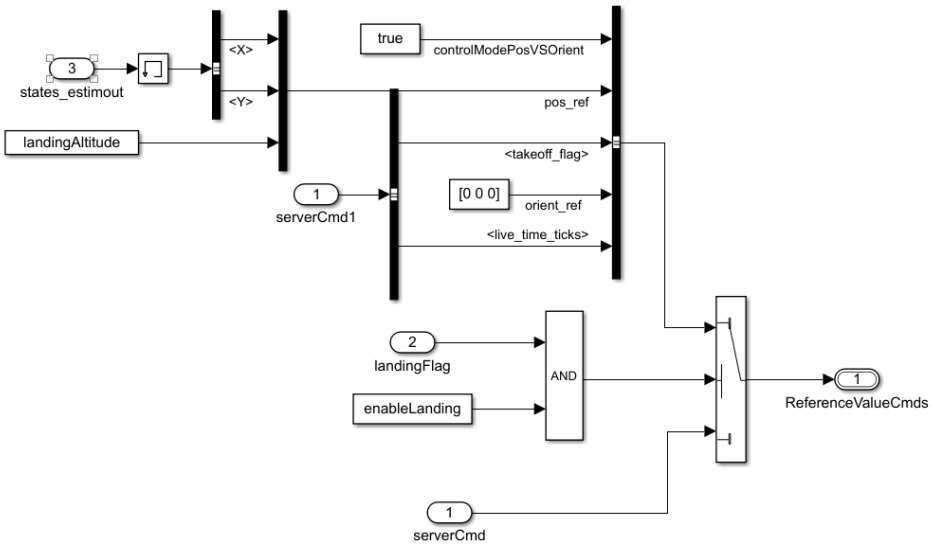


Figura 4.19: Landing logic

sensordata_group è il blocco che prende in ingresso i dati ricavati dai sensori montati sul drone per poi accorparli in una grande struttura dati che verrà poi inviata tramite il bus verso il blocco Estimator, che fornisce la stima dello stato del drone in quell'istante.

estimator è il blocco che, sulla base dei dati ricevuti dal sistema di sensoristica, fornisce le stime delle grandezze cinematiche del drone, ovvero del suo stato complessivo, costituito da posizioni, velocità, rotazioni rispetto al body frame (Figura 4.20). L'uscita viene poi utilizzata nei blocchi Controller e Crash Predictor Flags rispettivamente per il calcolo dell'errore relativo alle variabili controllate (dato dalla differenza tra la grandezza misurata e la grandezza desiderata) e per la prevenzione di situazioni insidiose durante il volo del Parrot Mambo.

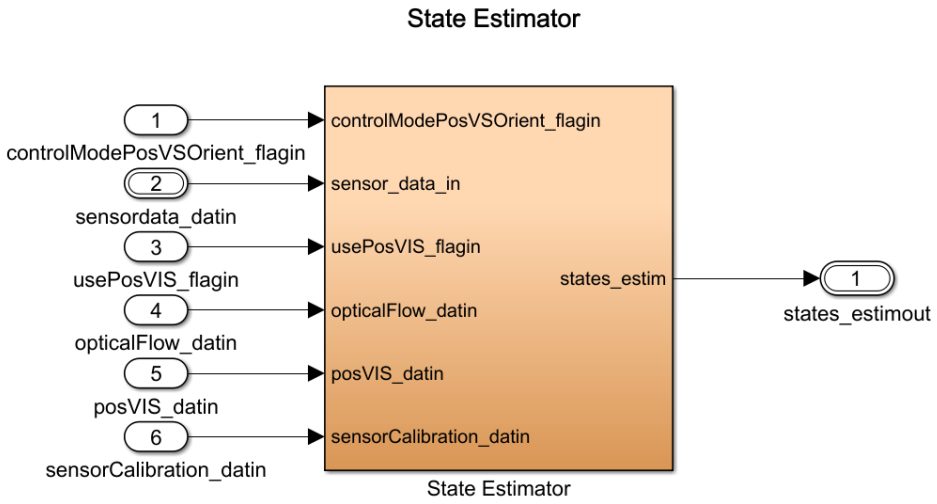


Figura 4.20: Estimator

Crash Predictor Flags è il blocco il cui algoritmo, analizzando in ingresso lo stato attuale stimato del drone, permette di predire eventuali situazioni potenzialmente dannose per il drone, producendo in uscita una flag che sta a significare che il volo del quadricottero è in una condizione sicura oppure che occorre disattivare urgentemente i motori, interrompendo il volo. (Figura 4.21).

Logging è il blocco che permette di andare a visualizzare tutte le informazioni relative al volo, non appena esso giunge a conclusione. Questo è possibile grazie ai blocchi To Workspace, che permettono la creazione direttamente nel file system del Parrot Mambo di MAT DATA file contenenti ad esempio le variabili di stato del drone, i segnali di riferimento, i comandi impartiti ai motori o le misure effettuate dai sensori. (Figura 4.22).

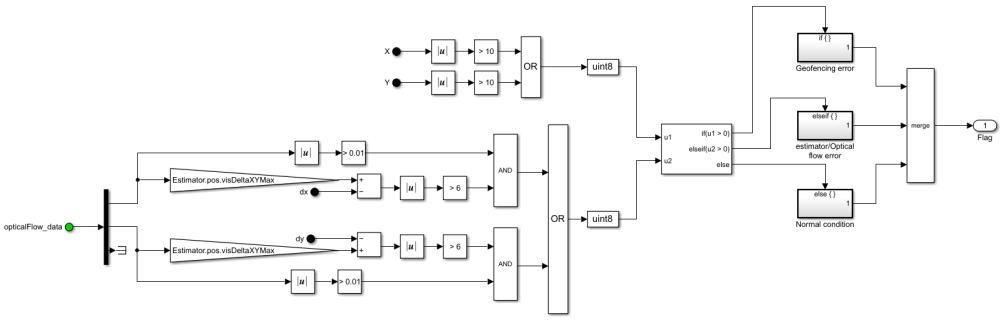


Figura 4.21: Crash predictor flags

controller è il macro-blocco (Figura 4.23) atto a fornire gli sforzi di controllo dei 4 motori, a partire dagli ingressi di riferimento provenienti dal blocco Command e dallo stato attuale del drone proveniente dal blocco Estimator: la differenza tra questi valori permette di calcolare l'errore di inseguimento, che verrà utilizzato dai controllori per generare gli sforzi di controllo. Entrando in questo blocco si ha accesso al blocco Flight Controller.

4.4.1 Flight controller

Scendiamo ora nel dettaglio, andando a studiare il blocco Flight Controller, rappresentato in figura 4.24, che contiene i blocchi dei controllori dei 6 gradi di libertà e i blocchi per la conversione dagli sforzi di controllo ai comandi da impartire ai motori. Andiamo a presentarli:

ControlMixer sfrutta un multiplexer per creare un vettore che raggruppa tutti gli sforzi di controllo generati dai controllori relativi a yaw, pitch, roll e quota. Questo vettore viene in seguito moltiplicato per la matrice *Controller.Q2Ts*, costituita da coefficienti per pesare il contributo dei controllori, tenendo conto del segno che indica il verso del momento torcente dovuto a ciascun rotore.
L'uscita sarà quindi il vettore *thrusts_refout*, che andrà poi in ingresso al secondo blocco ausiliario per l'effettiva conversione in segnali per i motori.(Figura 4.25).

thrustsToMotorCommands trasforma le spinte dei motori nelle velocità di rotazioni delle eliche dei 4 motori (espresse in giri al secondo): per fare ciò si usa il guadagno *Vehicle.Motor.thrustToMotorCommand* (presente nel Workspace) per amplificare lo sforzo di controllo, in concomitanza con un saturatore per evitare che gli attuatori si trovino a compiere sforzi che superano i loro limiti fisici, ed un altro guadagno MotorDirections che moltiplica le velocità rotazionali delle eliche per il segno associato al loro verso di rotazione. (Figura 4.26).

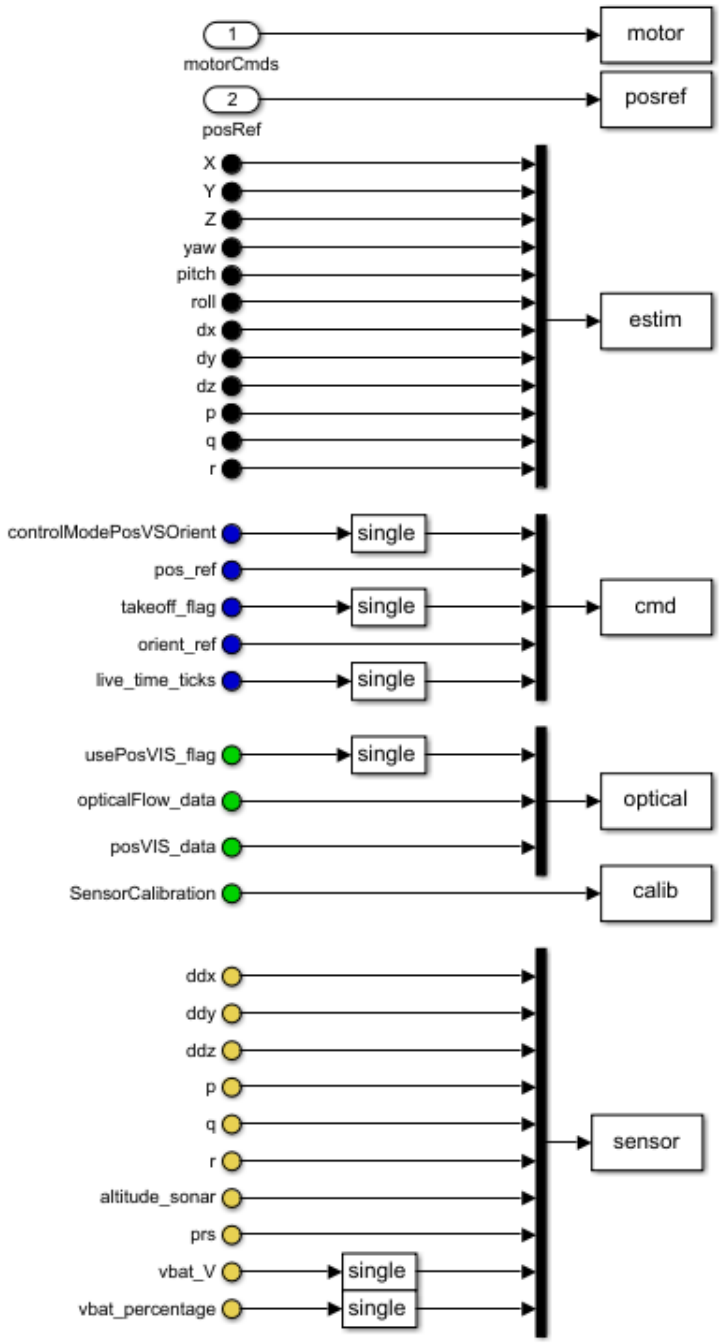


Figura 4.22: Logging

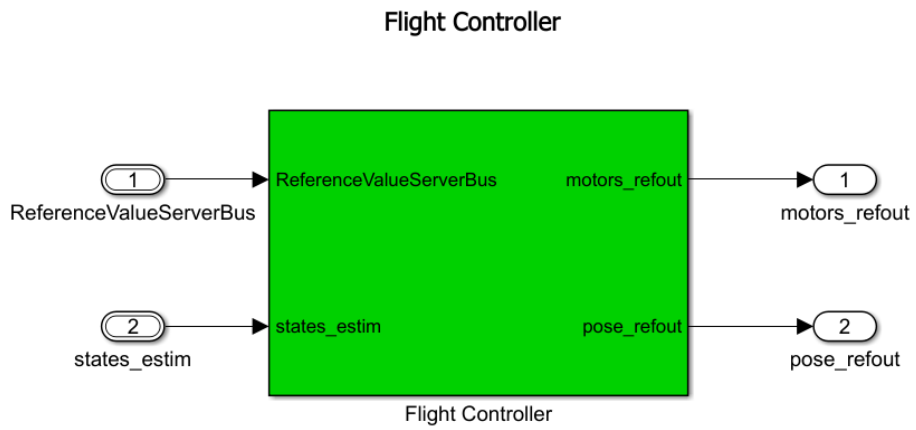


Figura 4.23: Controller

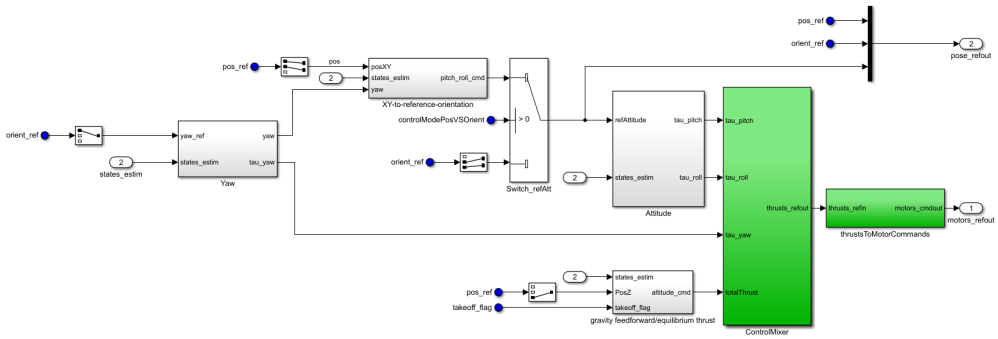


Figura 4.24: Flight Controller

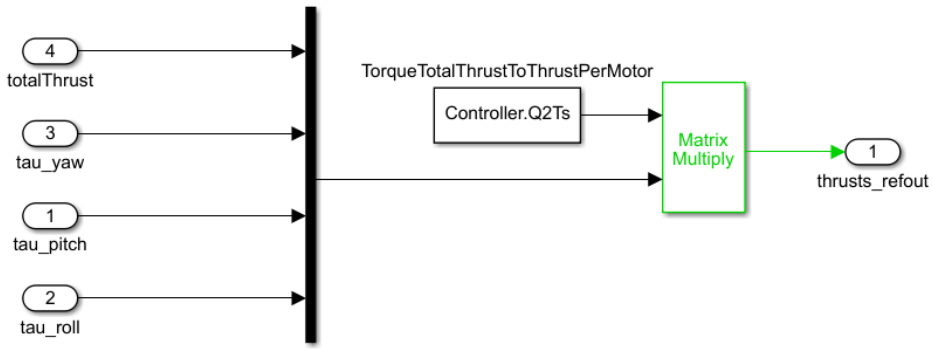


Figura 4.25: Control Mixer

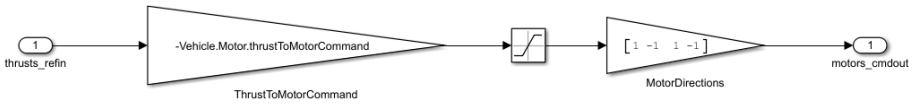


Figura 4.26: ThrustsToMotorCommands

Per quanto riguarda invece i controllori veri e propri, questi sono implementati con dei regolatori PID (Proportional Integral Derivative), particolari sistemi di controllo di natura non lineare, che si trovano nei seguenti blocchi:

- XY-to-reference-orientation
- Yaw
- Attitude
- gravity feedforward/equilibrium thrust

I controllori PID relativi alle variabili yaw, pitch, roll e z (quota) sono stati sostituiti dal controllore che sfrutta la tecnica sliding mode, ma a titolo informativo riportiamo qui di seguito la struttura che avevano in precedenza.

4.4.2 Controllo della traiettoria xy

Il blocco *XY-to-reference-orientation* è l'unico controllore che non è stato modificato, in quanto il nostro scopo era controllare i 3 angoli di Eulero e la quota. (Figura 4.27).

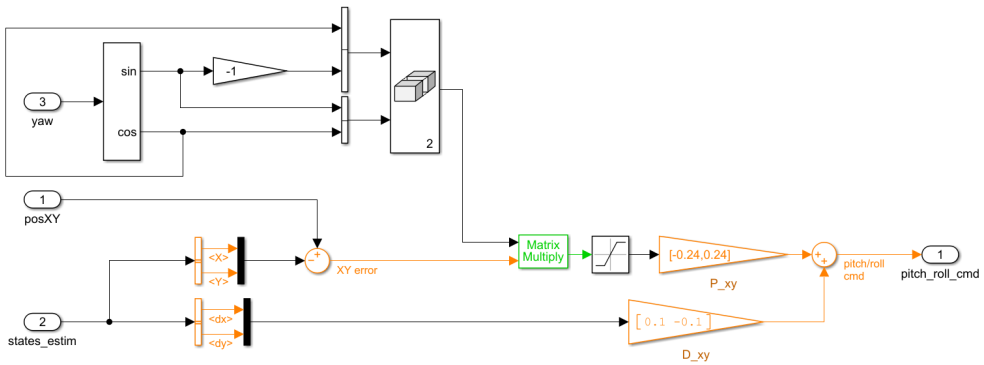


Figura 4.27: Controllore della traiettoria xy

4.4.3 Controllo dello yaw

Qui di seguito si può osservare il controllore dell'angolo di imbardata (*yaw*), contentuto nel blocco *Yaw*. (Figura 4.28).

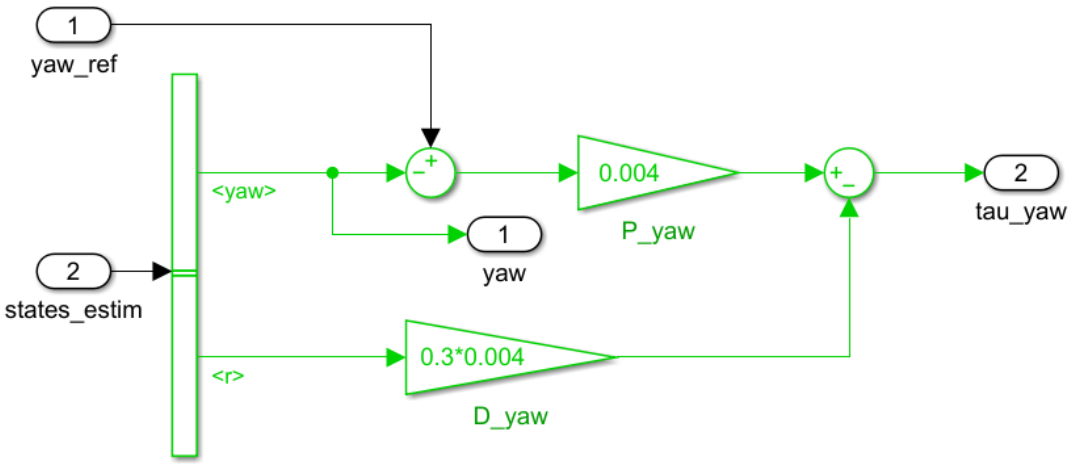


Figura 4.28: Controllore dello yaw

4.4.4 Controllo del pitch e del roll

Il blocco *Attitude* contiene il controllore per il mantenimento dell'assetto del velivolo (condizione di hovering), che avviene controllando gli angoli di rollio (*roll*) e beccheggio (*pitch*), i quali rappresentano l'inclinazione del drone. (Figura 4.29).

4.4.5 Controllo della quota

Nel blocco *gravity feedforward/equilibrium thrust* si trova il controllore relativo alla quota. (Figura 4.30).

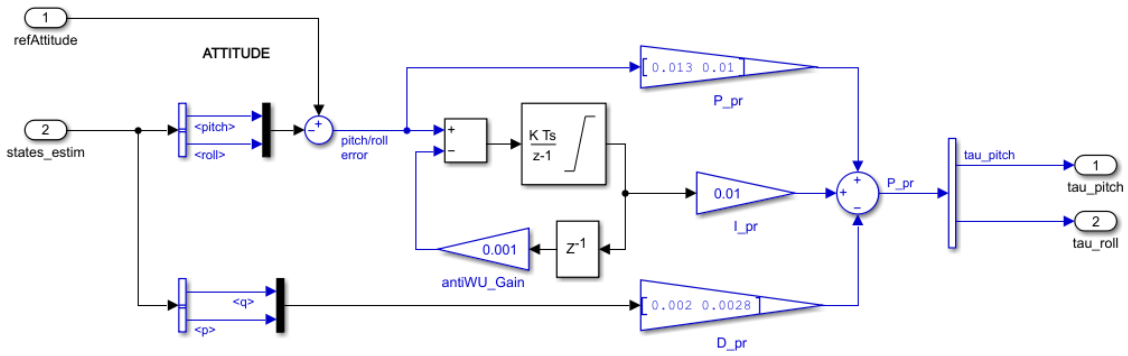


Figura 4.29: Controllore del roll e del pitch

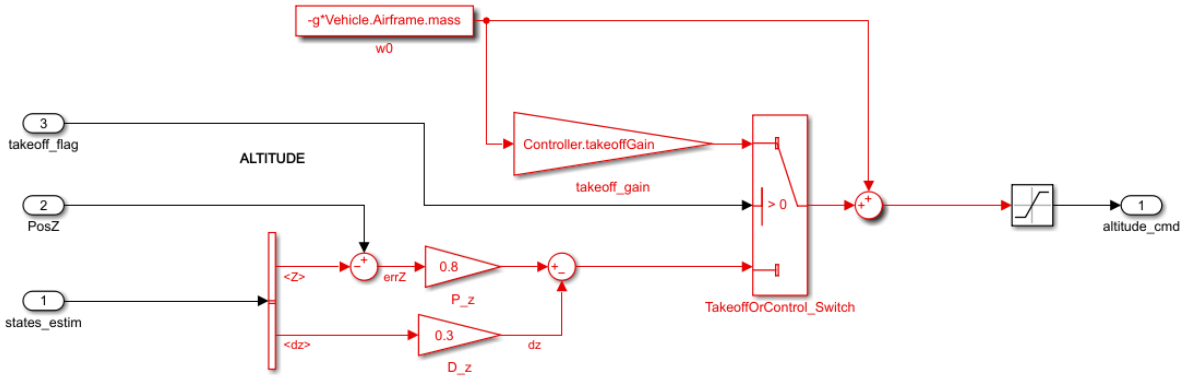


Figura 4.30: Controllore della quota

Capitolo 5

Controllo sliding mode

5.1 Generalità sulla teoria del controllo

L'accezione con cui ci apprestiamo ad utilizzare il termine "controllo" all'interno di questo elaborato non è quella, largamente diffusa nella lingua italiana, di "verificare" (in inglese *check*) ma piuttosto quella di "manipolare" (in inglese, per l'appunto, *control*).
I problemi di controllo consistono nel voler imporre un funzionamento desiderato ad un determinato processo, dove per processo si intende il sistema dinamico che si sta attualmente studiando: un impianto industriale, un braccio meccanico, un drone, un elettrodromestico, un fenomeno naturale, un motore, e molto altro.
Cosa significa imporre un funzionamento desiderato ad un sistema? Significa progettare un controllore che faccia sì che le grandezze di interesse del processo (dette variabili controllate) seguano lo stesso andamento nel tempo di un segnale scelto come riferimento. A tale scopo si selezionano delle grandezze del processo, nominate variabili di controllo, tramite le quali si può svolgere l'azione di controllo, andando dunque ad agire sull'andamento dello stesso.
Dato che però, nella realtà, non è possibile ottenere la perfetta coincidenza degli andamenti delle variabili controllate e dei segnali di riferimento, ci si accontenta di ottenere un errore di inseguimento (dato come differenza tra i due) accettabilmente piccolo, ovvero tendente allo zero.
I sistemi di controllo più utilizzati, in quanto permettono di compensare al meglio l'effetto di disturbi indesiderati, sono i sistemi di controllo in catena chiusa o in retroazione, come quello rappresentato in figura: il loro obiettivo è produrre una legge di controllo che vada a ridurre continuamente l'errore di inseguimento ottenuto dalla comparazione tra la variabile controllata e il riferimento imposto. Questo tipo sistema di controllo è quello che abbiamo già riscontrato nel progetto asbQuadcopter.

5.2 Tecnica di controllo sliding mode

Il controllo sliding mode è una tecnica di controllo a struttura variabile che si può applicare su sistemi non lineari, come il quadricottero in esame.

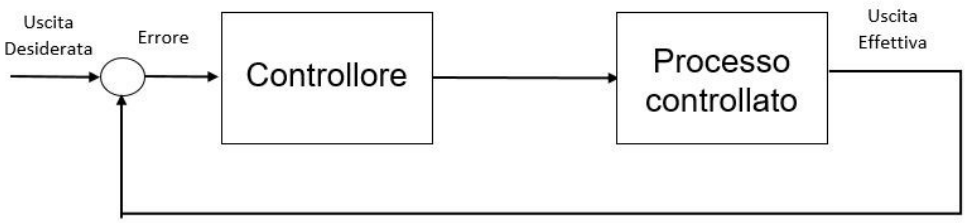


Figura 5.1: Sistema di controllo in catena chiusa

Il controllo a struttura variabile, in generale, è una tecnica che si basa sulla retroazione dallo stato ed è caratterizzato dalla continuità a tratti della variabile di controllo: infatti questa si ritrova, in base a certe condizioni imposte dal progettista, a effettuare uno switch, ovvero un passaggio da un certo valore ad un altro.
I punti di discontinuità sono situati su delle superfici, dette superfici di sliding che, non appena vengono attraversate dallo stato del sistema, determinano lo switching, cioè il cambiamento della struttura di controllo che, per l'appunto si dice variabile.
L'obiettivo di questa tecnica di controllo, dopo aver individuato e definito geometricamente le suddette superfici di sliding, è far sì che lo stato, a seguito di un transitorio iniziale, raggiunga in un tempo finito l'intersezione di queste superfici e che vi "scivoli" permanentemente: questa è la condizione che ci garantisce che l'evoluzione dello stato coincida con quella desiderata. Quindi, ricapitolando, i passi da eseguire per implementare un controllo in sliding mode sono:

1. definizione delle superfici di sliding, alle quali corrisponde il soddisfacimento di determinate specifiche.

2. progettazione della legge di controllo, che deve assicurare che lo stato scivoli sulle suddette superfici.

5.3 Condizione di sliding

Consideriamo un sistema dinamico descritto dalla seguente equazione differenziale vettoriale:

$$\dot{x} = f(x, u, t) \tag{5.1}$$

dove $x \in \mathbb{R}^n$, $u \in \mathbb{R}^m$, $t \in \mathbb{R}_+$.
Le superfici di sliding saranno invece descritte dall'equazione vettoriale qui riportata:

$$s(x) = 0 \tag{5.2}$$

dove $s(x) : \mathbb{R}^n \to \mathbb{R}^m$, cioè $s(x) = [s_1(x) \dots s_m(x)]^T = 0$.
La condizione che garantisce lo scivolamento sulla superficie di sliding è che lo stato,

indipendentemente da dove si trovi, si avvicini alla superficie. Questa condizione necessaria è data dalla seguente coppia di disequazioni:

$$\lim_{s\to 0^-} \dot{s} > 0 \quad \lim_{s\to 0^+} \dot{s} < 0 \tag{5.3}$$

che si possono riassumere nella seguente disequazione:

$$\lim_{s\to 0}(s\,\dot{s}) < 0 \tag{5.4}$$

Il significato geometrico di questa condizione si può spiegare ragionando, per semplicità, in $\mathbb{R}^2$: $\dot{s}$ si può infatti vedere come il prodotto scalare tra il gradiente di s (∇s) e il vettore velocità $\dot{x}$, cioè $\dot{s} = \nabla s \cdot \dot{x}$. Ricordando che il gradiente di una funzione costante rappresenta la direzione di massimo incremento della funzione associata alla curva, le 5.3 implicano dunque che:

- per valori di s che tende a 0^-, cioè se lo stato si trova in prossimità della superficie da sinistra, ∇s e $\dot{x}$ devono formare un angolo acuto, vale a dire che lo stato si deve dirigere verso la superficie.
- per valori di s che tende a 0^+, cioè se lo stato si trova in prossimità della superficie da destra, ∇s e $\dot{x}$ devono formare un angolo ottuso, vale a dire che lo stato si deve dirigere verso la superficie.

Appare dunque chiaro il motivo per cui queste condizioni sono definite condizioni di esistenza o condizioni di reaching: senza di esse non si ha nessuna garanzia che lo stato si muova verso la superficie.
A questo punto si pone il problema di estendere globalmente queste condizioni di esistenza dello sliding mode: a tale scopo adoperiamo la teoria della stabilità di Lyapunov.
Scegliamo come funzione di Lyapunov la funzione $V(s(x)) = \frac{1}{2}s^T(x)s(x)$, che risulta essere definita positiva. Possiamo dunque riscrivere la 5.4 come segue:

$$\dot{V}(s(x)) = s^T(x)\dot{s}(x) < 0 \tag{5.5}$$

imponendo cioè che la derivata della funzione di Lyapunov sia negativa.

5.4 Progettazione della legge di controllo

Lo sforzo di controllo generato con la tecnica sliding mode consta di due parti:

- controllo equivalente: è la componente che fa sì che lo stato si diriga verso la superficie e vi rimanga.
- controllo correttivo: è la componente che compensa le variazioni e disturbi che possono agire sullo stato mentre esso sta scivolando sulla superficie.

Lo sforzo di controllo avrà un aspetto simile:

$$U_i(x,t) = \begin{cases} U_i^+(x,t) & \text{se } s_i(x) \geq 0 \\ U_i^-(x,t) & \text{se } s_i(x) < 0 \end{cases} \tag{5.6}$$

con i=1...m, $U_i^+(x,t)$ e $U_i^-(x,t)$ funzioni continue tali che $U_i^+(x,t) \neq U_i^-(x,t)$ in corrispondenza di $s_i(x) = 0$.
Per determinare il controllo equivalente si utilizza il cosiddetto metodo equivalente, che consiste nel porre uguale a zero la derivata rispetto al tempo di s(x), cioè $\dot{s}(x) = 0$.
A livello geometrico, ritornando a quanto detto nel paragrafo precedente, questo significa che il prodotto scalare tra ∇s e $\dot{x}$ è nullo, cioè che la velocità è tangente alla superficie di scivolamento o, in altre parole, che lo stato scivola su di essa.
Tuttavia, raramente lo switching è ideale ed è molto probabile che la traiettoria possa essere soggetta ad un cosiddetto ciclo di isteresi di una determinata ampiezza: lo sforzo di controllo si trova cioè a commutare con una certa frequenza tra due valori. Questo fenomeno indesiderato prende il nome di chattering.
La soluzione a questa problematica sta nel rinunciare all'obiettivo che lo stato scivoli perfettamente sulla superficie di sliding e nell'accontentarsi che esso si trovi all'interno di una regione, detta boundary layer, in prossimità di essa: in formula, vogliamo ottenere $|s(x)| \leq \epsilon$, con $\epsilon > 0$.
Bisogna dunque andare a rivisitare la legge di controllo, che, al di fuori dell'intorno della superficie, avrà la seguente forma:

$$U_i(x,t) = \begin{cases} U_i^+(x,t) & \text{se } s_i(x) > \epsilon \\ U_i^-(x,t) & \text{se } s_i(x) < -\epsilon \end{cases} \tag{5.7}$$

All'interno della regione, invece, si vuole rendere la legge di controllo continua, eliminando la discontinuità. Per capire meglio questo aspetto, facciamo un esempio: consideriamo la legge di controllo $u(x,t) = K \cdot sign[s(x)]$, con $k > 0$.

$$u(x,t) = \begin{cases} k & \text{se } s(x) > \epsilon \\ -k & \text{se } s(x) < -\epsilon \\ k \cdot \frac{s(x)}{\epsilon} & \text{se } |s(x)| \leq \epsilon \end{cases} \tag{5.8}$$

che può essere sintetizzata nel seguente modo:

$$u(x,t) = \begin{cases} k \cdot sign[s(x)] & \text{se } |s(x)| \geq \epsilon \\ k \cdot \frac{s(x)}{\epsilon} & \text{se } |s(x)| \leq \epsilon \end{cases} \tag{5.9}$$

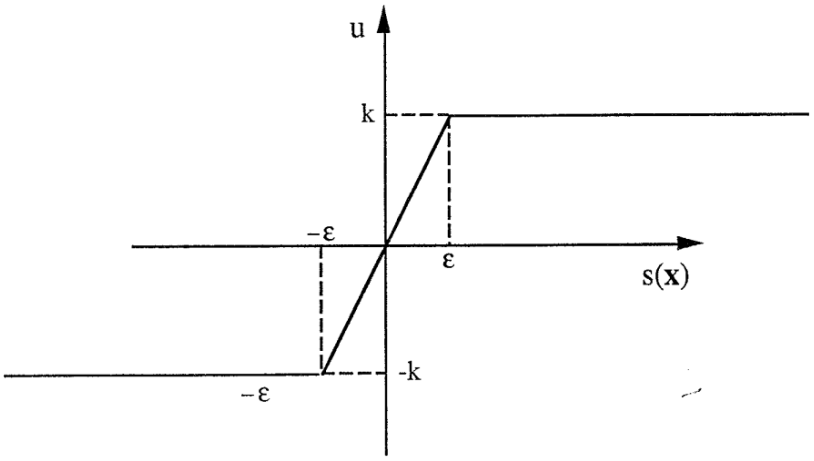


Figura 5.2: Legge di controllo boundary layer

5.5 Scelta della superficie di sliding

Come già detto in precedenza, la scelta della superficie di sliding, con le sue caratteristiche geometriche, si effettua sulla base delle specifiche che si vogliono impartire al sistema. Nel nostro caso, avendo a che fare con un problema di inseguimento di traiettorie, con l'obiettivo di annullare l'errore a regime permanente, dato dalla differenza tra il valore misurato dello stato e il valore desiderato, daremo alla superficie una struttura come la seguente:

$$s(x,t) = \dot{e} + \Lambda e = 0 \tag{5.10}$$

con Λ matrice diagonale $m \times m$, cioè $\Lambda = diag(\lambda_i)$, $\lambda_i > 0$, i=1...m.
Questa equazione differenziale significa che, mentre lo stato percorre la superficie, ogni componente i-esima dell'errore tende esponenzialmente a zero con costante di tempo pari a $\frac{1}{\lambda_i}$: ciò implica che, maggiore sarà il valore di λ_i, più velocemente lo stato raggiungerà il valore desiderato, scivolando sulla superficie.
Per completezza, andiamo a riassumere l'insieme delle superfici di sliding utilizzate per il controllo del drone:

$$\begin{cases} S_\phi = \dot{e_\phi} + \lambda_\phi e_\phi = 0 \\ S_\theta = \dot{e_\theta} + \lambda_\theta e_\theta = 0 \\ S_\psi = \dot{e_\psi} + \lambda_\psi e_\psi = 0 \\ S_z = \dot{e_z} + \lambda_z e_z = 0 \end{cases} \tag{5.11}$$

5.6 Sintesi dei controllori

5.6.1 Controllo equivalente

Imponendo la condizione di sliding ($\dot{s}(x) = 0$) per ogni superficie e ricordando le equazioni differenziali non lineari che costituiscono il nostro processo, e utilizzando il fatto che la matrice di trasformazione T_i^b è approssimabile con la matrice identità, otteniamo i controlli equivalenti:

- **roll**

$$\dot{S}_\phi = \ddot{e_\phi} + \lambda_\phi \dot{e}_\phi = (\ddot{\phi} - \ddot{\phi}_d) + \lambda_\phi \dot{e}_\phi = (\ddot{\phi} - \ddot{\phi}_d) + \lambda_\phi (p - p_d) = 0 \tag{5.12}$$

  Ponendo $\ddot{\phi}_d = 0$ e $p_d = 0$, e sostituendo la 2.19, ottengo il controllo equivalente del roll:

$$U_{\phi(eq)} = I_{xx}[-\lambda_\phi p - \frac{I_{yy} - I_{zz}}{I_{xx}} qr + \frac{J_r \Omega_r}{I_{xx}} q] \tag{5.13}$$

- **pitch**

$$\dot{S}_\theta = \ddot{e_\theta} + \lambda_\theta \dot{e}_\theta = (\ddot{\theta} - \ddot{\theta}_d) + \lambda_\theta \dot{e}_\theta = (\ddot{\theta} - \ddot{\theta}_d) + \lambda_\theta (q - q_d) = 0 \tag{5.14}$$

  Ponendo $\ddot{\theta}_d = 0$ e $q_d = 0$, e sostituendo la 2.19, ottengo il controllo equivalente del pitch:

$$U_{\theta(eq)} = I_{yy}[-\lambda_\theta q - \frac{I_{zz} - I_{xx}}{I_{yy}} pr - \frac{J_r \Omega_r}{I_{yy}} p] \tag{5.15}$$

- **yaw**

$$\dot{S}_\psi = \ddot{e_\psi} + \lambda_\psi \dot{e}_\psi = (\ddot{\psi} - \ddot{\psi}_d) + \lambda_\psi \dot{e}_\psi = (\ddot{\psi} - \ddot{\psi}_d) + \lambda_\psi (r - r_d) = 0 \tag{5.16}$$

  Ponendo $\ddot{\psi}_d = 0$ e $r_d = 0$, e sostituendo la 2.19, ottengo il controllo equivalente dello yaw:

$$U_{\psi(eq)} = I_{zz}[-\lambda_\psi r - \frac{I_{xx} - I_{yy}}{I_{zz}} pq] \tag{5.17}$$

- **quota**

$$\dot{S}_z = \ddot{e_z} + \lambda_z \dot{e}_z = (\ddot{z} - \ddot{z}_d) + \lambda_z (\dot{z} - \dot{z}_d) = 0 \tag{5.18}$$

  Ponendo $\ddot{z}_d = 0$ e $\dot{z}_d = 0$, e sostituendo la 2.18, ottengo il controllo equivalente della quota:

$$U_{z(eq)} = \frac{m}{cos\phi cos\theta}[-\lambda_z \dot{z}] \tag{5.19}$$

  Sono state tralasciate la forza peso e la forza di attrito viscoso in quanto il loro contributo viene già calcolato e considerato all'interno del progetto asbQuadcopter, nel blocco AC model.

5.6.2 Controllo correttivo

Passiamo ora a determinare la componente correttiva della legge di controllo, che si ottiene ricordando che in prossimità della superficie devono valere le condizioni di reaching per ogni superficie, ovvero $s_i(x) \cdot \dot{s}_i(x) < 0$.
Effettuando questo prodotto per ogni variabile controllata, ci troviamo ad elidere i termini di segno opposto per poi ottenere una forma semplificata, in cui la componente correttiva è l'incognita da individuare e che rende vera la disuguaglianza. Usando questa serie di passaggi, si ottengono i controlli correttivi, con l'aggiunta del boundary layer:

- **roll**

$$U_{\phi(c)} = \begin{cases} I_{xx}[-k_\phi sign(s_\phi)] & \text{se } |s_\phi| \geq \epsilon_\phi \\ I_{xx}[-k_\phi \frac{s_\phi}{\epsilon_\phi}] & \text{se } |s_\phi| < \epsilon_\phi \end{cases} \tag{5.20}$$

  Usando questo termine correttivo, con $k_\phi > 0$ e $\epsilon_\phi > 0$, si ottiene:

$$s_\phi \cdot \dot{s}_\phi = \begin{cases} -k_\phi |s_\phi| < 0 & \text{se } |s_\phi| \geq \epsilon_\phi \\ -\frac{k_\phi}{\epsilon_\phi} s_\phi^2 < 0 & \text{se } |s_\phi| < \epsilon_\phi. \end{cases} \tag{5.21}$$

- **pitch**

$$U_{\theta(c)} = \begin{cases} I_{yy}[-k_\theta sign(s_\theta)] & \text{se } |s_\theta| \geq \epsilon_\theta \\ I_{yy}[-k_\theta \frac{s_\theta}{\epsilon_\theta}] & \text{se } |s_\theta| < \epsilon_\theta \end{cases} \tag{5.22}$$

  Usando questo termine correttivo, con $k_\theta > 0$ e $\epsilon_\theta > 0$, si ottiene:

$$s_\theta \cdot \dot{s}_\theta = \begin{cases} -k_\theta |s_\theta| < 0 & \text{se } |s_\theta| \geq \epsilon_\theta \\ -\frac{k_\theta}{\epsilon_\theta} s_\theta^2 < 0 & \text{se } |s_\theta| < \epsilon_\theta. \end{cases} \tag{5.23}$$

- **yaw**

$$U_{\psi(c)} = \begin{cases} I_{zz}[-k_\psi sign(s_\psi)] & \text{se } |s_\psi| \geq \epsilon_\psi \\ I_{zz}[-k_\psi \frac{s_\psi}{\epsilon_\psi}] & \text{se } |s_\psi| < \epsilon_\psi \end{cases} \tag{5.24}$$

  Usando questo termine correttivo, con $k_\psi > 0$ e $\epsilon_\psi > 0$, si ottiene:

$$s_\psi \cdot \dot{s}_\psi = \begin{cases} -k_\psi |s_\psi| < 0 & \text{se } |s_\psi| \geq \epsilon_\psi \\ -\frac{k_\psi}{\epsilon_\psi} s_\psi^2 < 0 & \text{se } |s_\psi| < \epsilon_\psi. \end{cases} \tag{5.25}$$

- **quota**

$$U_{z(c)} = \begin{cases} \frac{m}{cos\phi cos\theta}[-k_z sign(s_z)] & \text{se } |s_z| \geq \epsilon_z \\ \frac{m}{cos\phi cos\theta}[-k_z \frac{s_z}{\epsilon_z}] & \text{se } |s_z| < \epsilon_z \end{cases} \tag{5.26}$$

Usando questo termine correttivo, con $k_z > 0$ e $\epsilon_z > 0$, si ottiene:

$$s_z \cdot \dot{s}_z = \begin{cases} -k_z|s_z| < 0 & \text{se } |s_z| \geq \epsilon_z \\ -\frac{k_z}{\epsilon_z} s_z^2 < 0 & \text{se } |s_z| < \epsilon_z. \end{cases} \tag{5.27}$$

5.6.3 Controllo complessivo

Andando a sommare rispettivamente la componente equivalente del controllo con la componente correttiva, si ottengono le leggi di controllo complessive che andiamo a riportare di seguito:

- **roll**

$$U_\phi = \begin{cases} I_{xx}[-\lambda_\phi p - \frac{I_{yy}-I_{zz}}{I_{xx}} qr + \frac{J_r\Omega_r}{I_{xx}} q - k_\phi sign(s_\phi)] & \text{se } |s_\phi| \geq \epsilon_\phi \\ I_{xx}[-\lambda_\phi p - \frac{I_{yy}-I_{zz}}{I_{xx}} qr + \frac{J_r\Omega_r}{I_{xx}} q - k_\phi \frac{s_\phi}{\epsilon_\phi}] & \text{se } |s_\phi| < \epsilon_\phi \end{cases} \tag{5.28}$$

- **pitch**

$$U_\theta = \begin{cases} I_{yy}[-\lambda_\theta p - \frac{I_{zz}-I_{xx}}{I_{yy}} pr - \frac{J_r\Omega_r}{I_{yy}} q - k_\theta sign(s_\theta)] & \text{se } |s_\theta| \geq \epsilon_\theta \\ I_{yy}[-\lambda_\theta p - \frac{I_{zz}-I_{xx}}{I_{yy}} pr - \frac{J_r\Omega_r}{I_{yy}} q - k_\theta \frac{s_\theta}{\epsilon_\theta}] & \text{se } |s_\theta| < \epsilon_\theta \end{cases} \tag{5.29}$$

- **yaw**

$$U_\psi = \begin{cases} I_{zz}[-\lambda_\psi r - \frac{I_{xx}-I_{yy}}{I_{zz}} pq - k_\psi sign(s_\psi)] & \text{se } |s_\psi| \geq \epsilon_\psi \\ I_{zz}[-\lambda_\psi r - \frac{I_{xx}-I_{yy}}{I_{zz}} pq - k_\psi \frac{s_\psi}{\epsilon_\psi}] & \text{se } |s_\psi| < \epsilon_\psi \end{cases} \tag{5.30}$$

- **quota**

$$U_z = \begin{cases} \frac{m}{cos\phi cos\theta}[-\lambda_z \dot{z} - k_z sign(s_z)] & \text{se } |s_z| \geq \epsilon_z \\ \frac{m}{cos\phi cos\theta}[-\lambda_z \dot{z} - k_z \frac{s_z}{\epsilon_z}] & \text{se } |s_z| < \epsilon_z \end{cases} \tag{5.31}$$

5.6.4 Parametri di progetto

Andiamo infine a riassumere nelle tabelle (5.1, 5.2, 5.3, 5.4) i parametri con cui abbiamo configurato il controllo sliding mode delle 4 variabili controllate.

Tabella 5.1: Parametri di progetto per l'angolo di roll

Parametro	**Valore**
λ_ϕ	10
k_ϕ	25
ϵ_ϕ	0.25

Tabella 5.2: Parametri di progetto per l'angolo di pitch

Parametro	**Valore**
λ_θ	10
k_θ	25
ϵ_θ	0.25

Tabella 5.3: Parametri di progetto per l'angolo di yaw

Parametro	**Valore**
λ_ψ	5
k_ψ	5
ϵ_ψ	0.25

Tabella 5.4: Parametri di progetto per la quota

Parametro	**Valore**
λz	20
k_z	5
ϵ_z	0.25

Capitolo 6

Implementazione dei controllori nel progetto asbQuadcopter

Rispetto al codice Simulink iniziale sono state apportate delle modifiche per poter implementare il controllo sliding mode.

6.1 Generazione dei segnali di riferimento

Per prima cosa, nel blocco Command, è stato modificato il blocco Signal editor (Figura 6.1), sostituendo il precedente Position/Attitude Reference con un Signal builder, che presenta un'interfaccia utente molto intuitiva con la quale si possono editare manualmente i segnali di riferimento da generare.
Questo ci ha permesso di generare i riferimenti in Figura 6.2, dove bisogna tenere conto che la quota desiderata (variabile z) ha un valore negativo in quanto l'asse z è rivolto verso il basso. I seguenti segnali di riferimento sono stati scelti con l'intenzione di fare al drone la seguente serie di azioni:

- entro il primo secondo di volo, si ha il decollo;
- dopo il primo secondo di volo, il drone deve mantenere una quota di 1.5 m;
- dopo il quinto secondo di volo, cioè dopo aver raggiunto la quota desiderata, il drone deve ruotare di un angolo di circa 11 gradi;
- al quindicesimo secondo di volo, cioè dopo aver effettuato la rotazione attorno all'asse di imbardata, deve spostarsi di 1 m lungo la direzione data dall'asse x e di 1 m lungo la direzione data dall'asse y.
- infine, il drone deve stabilizzarsi (angoli di roll e pitch nulli).

6.2 Blocco Flight Controller

All'interno del progetto asbQuadcopter, il blocco dove è stato implementato il controllo sliding mode è il Flight Controller, che è stato opportunamente modificato come si può osservare in Figura 6.3:
Le modifiche apportate sono le seguenti:

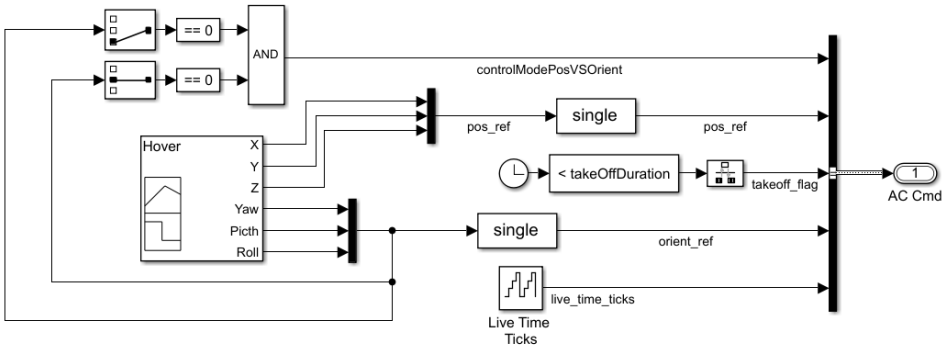


Figura 6.1: Signal editor

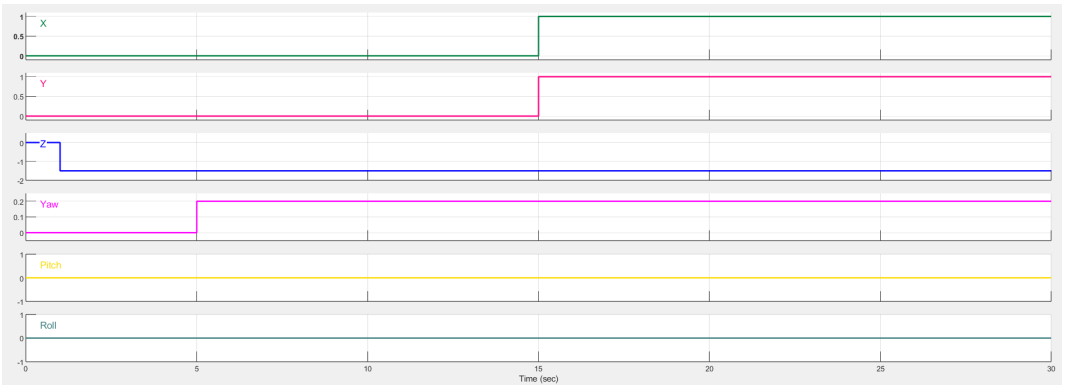


Figura 6.2: Segnali di riferimento a gradino

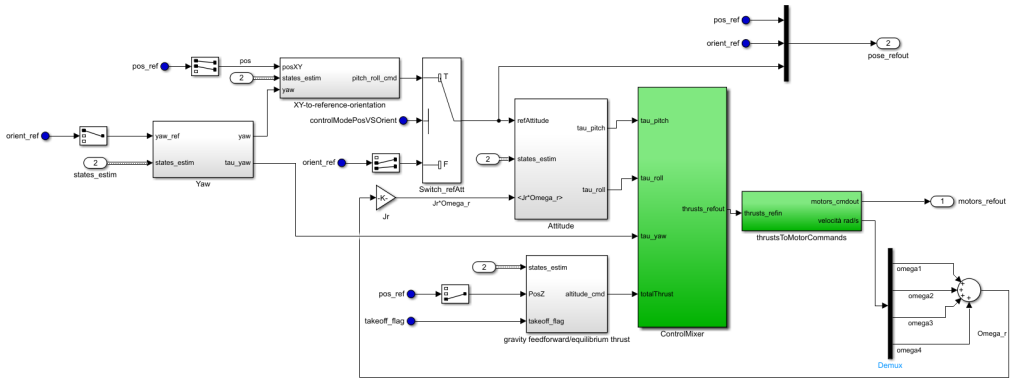


Figura 6.3: Blocco Flight Controller

- all'interno del blocco thrustsToMotorCommands si è moltiplicata la prima uscita (le velocità angolari dei 4 motori espresse in giri/sec) per 2π, in modo tale da ottenere le velocità angolari espresse in radianti/sec (seconda uscita del blocco).

- questa seconda uscita del blocco thrustsToMotorCommands passa poi per un demultiplexer in modo tale da poter ottenere il termine Ω_r, ovvero il 'residuo' di velocità angolare dei rotori, dato dalla somma delle velocità angolari dei motori, tenendo conto dei versi di rotazione. Questo valore viene moltiplicato poi per J_r, momento d'inerzia del rotore dei motori ed entra nel blocco Attitude per il controllo del roll e del pitch.

6.2.1 Controllori

In questo sottoparagrafo verranno mostrati i blocchi che implementano i controllori delle 4 variabili controllate.

- **Yaw** Il blocco Yaw (Figura 6.4) è stato modificato rispetto alla sua forma precedente, e al posto del regolatore PID è stato inserito il controllore sliding mode (Figura 6.5); inoltre sono stati aggiunti dei blocchi Scope per visualizzare lo sforzo di controllo e l'andamento dello yaw rispetto al suo riferimento.

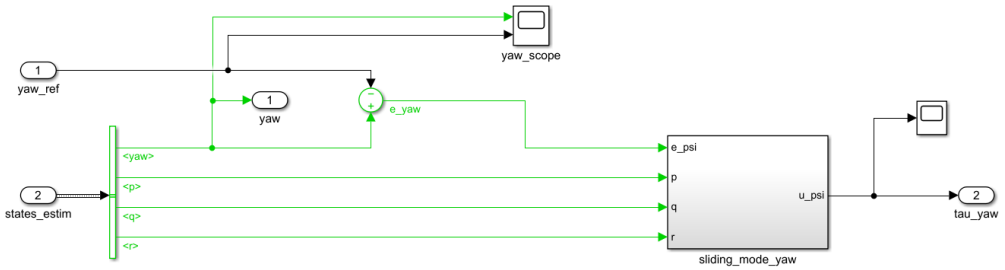


Figura 6.4: Blocco Yaw

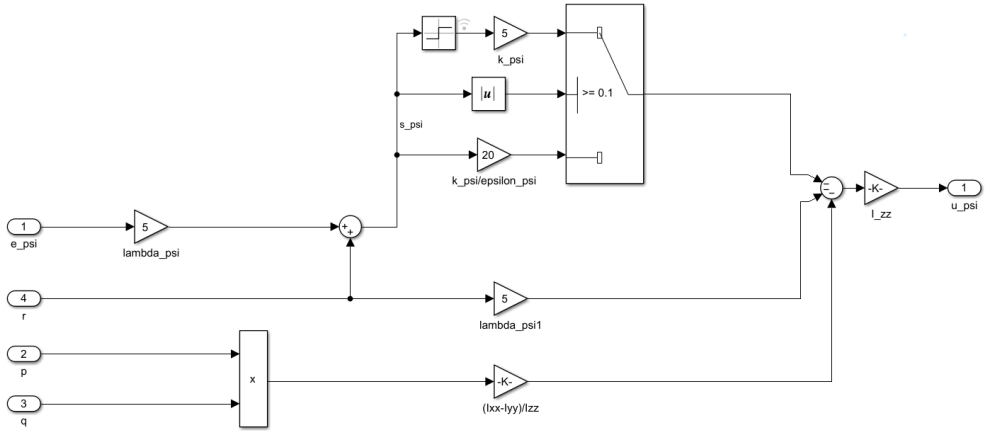


Figura 6.5: Controllore sliding mode dell'angolo di yaw

- **Attitude** Il blocco Attitude (Figura 6.6) è stato modificato rispetto alla sua forma precedente, e al posto dei due regolatori PID sono stati inseriti i controllori sliding mode, rispettivamente per l'angolo di roll (Figura 6.7) e per l'angolo di pitch (Figura 6.8); inoltre sono stati aggiunti dei blocchi Scope per visualizzare gli sforzi di controllo e l'andamento dei due angoli rispetto ai loro riferimenti.

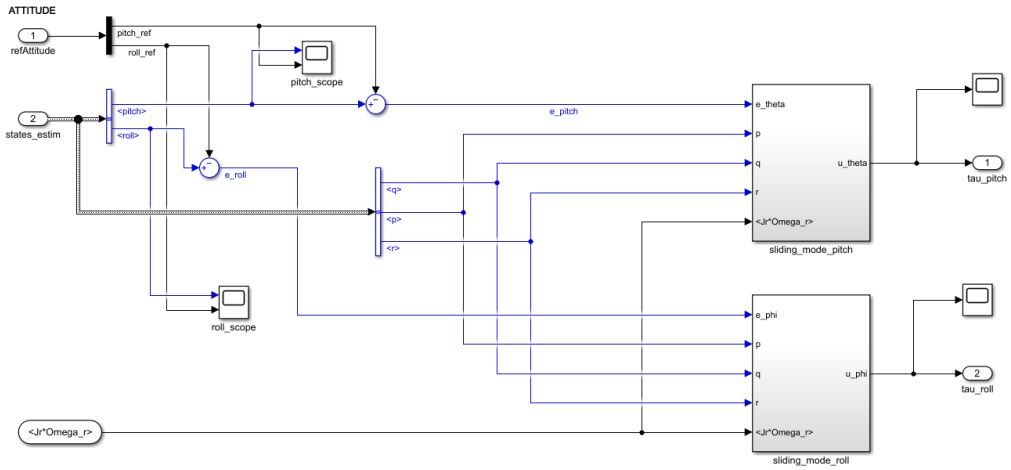


Figura 6.6: Blocco Attitude

- **gravity feedforward/equilibrium thrust** Il blocco gravity feedforward/equilibrium thrust (Figura 6.9) è stato modificato rispetto alla sua forma precedente, e al posto del regolatore PID è stato inserito il controllore sliding mode (Figura 6.10); inoltre sono stati aggiunti dei blocchi Scope per visualizzare lo sforzo di controllo e l'andamento della quota rispetto al suo riferimento.
  Da sottolineare il fatto che si è lasciato che nel primo secondo di volo, durante

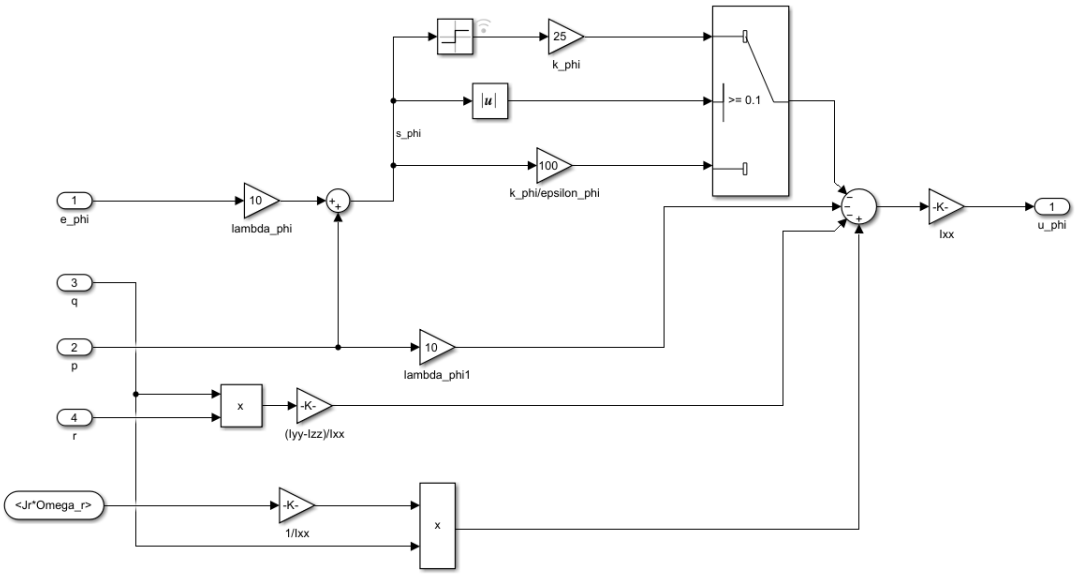


Figura 6.7: Controllore sliding mode dell'angolo di roll

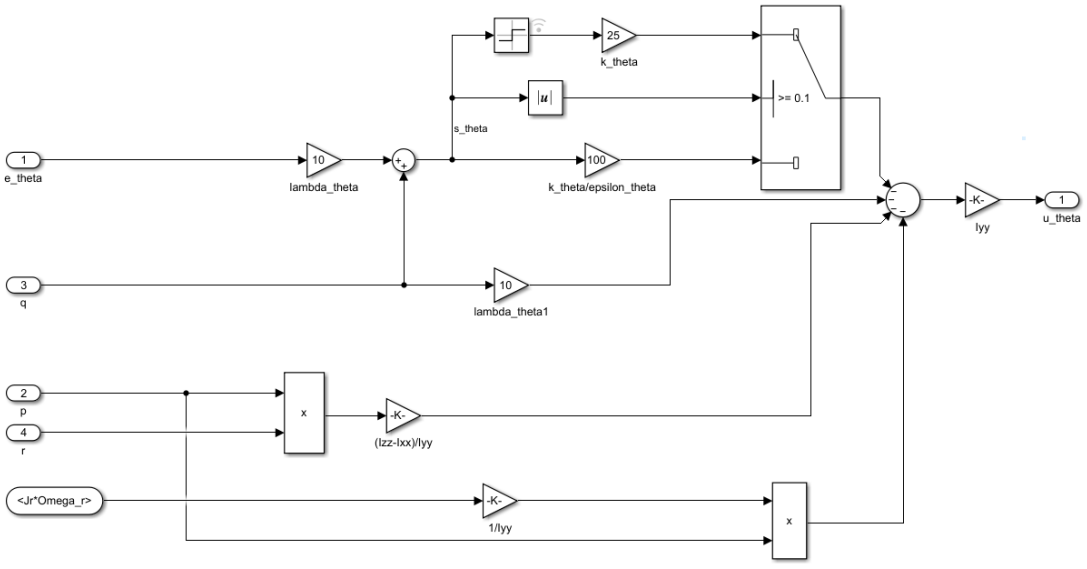


Figura 6.8: Controllore sliding mode dell'angolo di pitch

il quale la variabile booleana *takeoff_flag* è true, la spinta è data dalla forza di gravità moltiplicata per un guadagno. Inoltre la forza di gravità non è stata considerata all'interno del processo (in quanto già considerata nel blocco Environment), né dunque all'interno del controllore, ma è stata aggiunta come un disturbo costante che agisce sulla spinta.

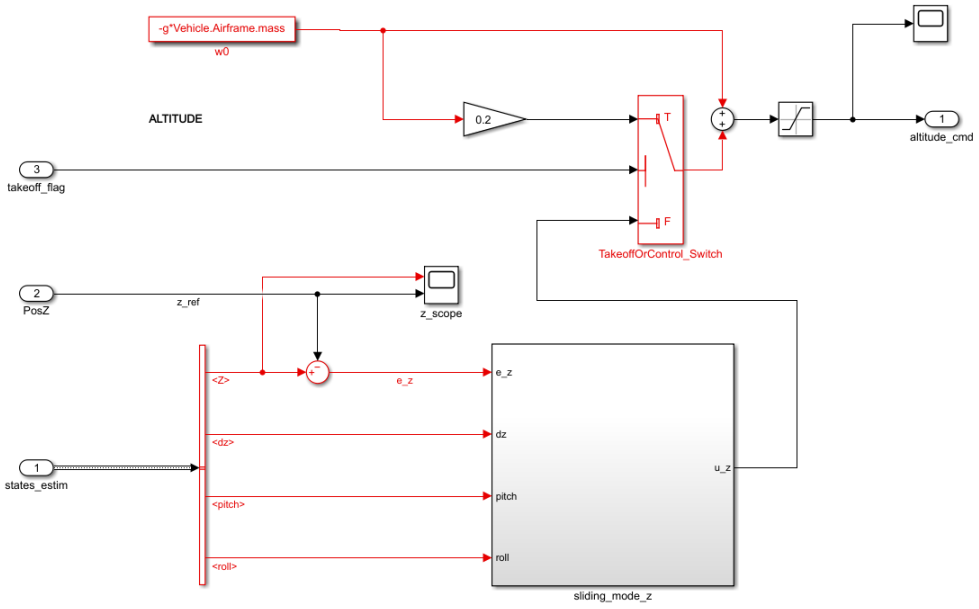


Figura 6.9: Blocco gravity feedforward/equilibrium thrust

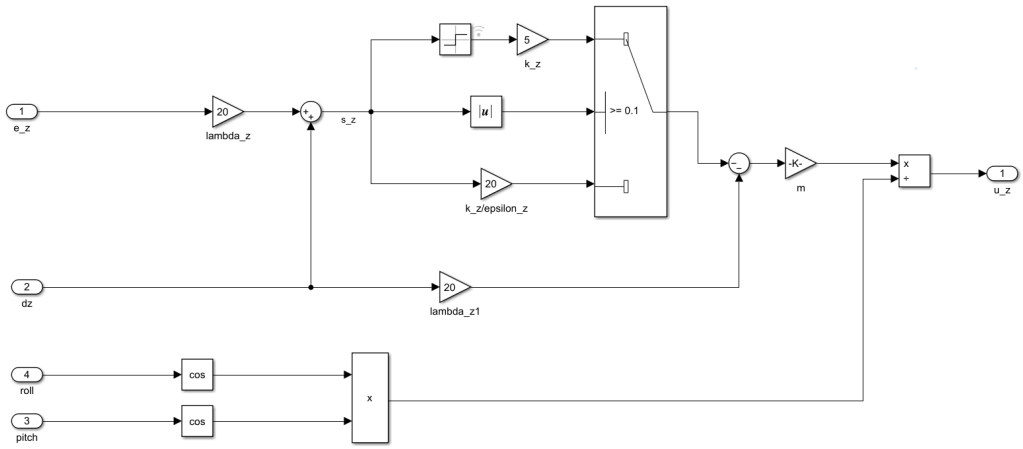


Figura 6.10: Controllore sliding mode della quota

Capitolo 7

Risultati sperimentali

Dopo aver illustrato le modifiche svolte sul progetto iniziale, lo step successivo è stato andare a visualizzare l'andamento dello stato del drone, in particolare dei gradi di libertà che ci interessano, una volta che vengono impartiti i segnali di riferimento definiti nel blocco Signal builder.

7.1 Simulazione al pc: riferimenti a gradino

7.1.1 Analisi qualitativa

Come prima cosa andiamo ad esaminare, in simulazione, grazie ai blocchi Scopes, i risultati relativi alla posizione (coordinate x e y), all'assetto e alla quota, facendo allo stesso tempo un confronto con l'andamento che i gradi di libertà avevano grazie all'azione dei PID originali.
La definiamo "analisi qualitativa" in quanto vedere i cambiamenti delle prestazioni del drone ad occhio non ci fornisce un criterio oggettivo con cui misurare il miglioramento del controllo del quadricottero. Nel prossimo paragrafo useremo infatti degli indici di prestazione che ci tireranno fuori dall'impasse.
Attualmente, osservando i grafici ottenuti nelle figure 7.1, 7.2, 7.3, 7.4, 7.5, 7.6, concludiamo che:

- il miglioramento nel controllo degli angoli di roll e pitch comporta anche un netto miglioramento dell'andamento della x e della y, come si può notare dalla scomparsa delle oscillazioni, una volta che il drone si trova nella posizione stabilita: ciò significa che il Parrot Mambo ritorna presto in condizione di hover;

- l'angolo di yaw, a seguito dello spostamento del drone che ne perturba leggermente il valore, ritorna presto al valore del riferimento;

- la diminuzione del guadagno della spinta necessaria al decollo fa sì che il drone non superi di troppo il valore del riferimento; dopodiché interviene lo sliding mode per cui il quadricottero si avvicina presto alla quota desiderata. In seguito, nonostante lo spostamento del drone che ne va a perturbare la quota, il velivolo ritorna presto al valore di riferimento.

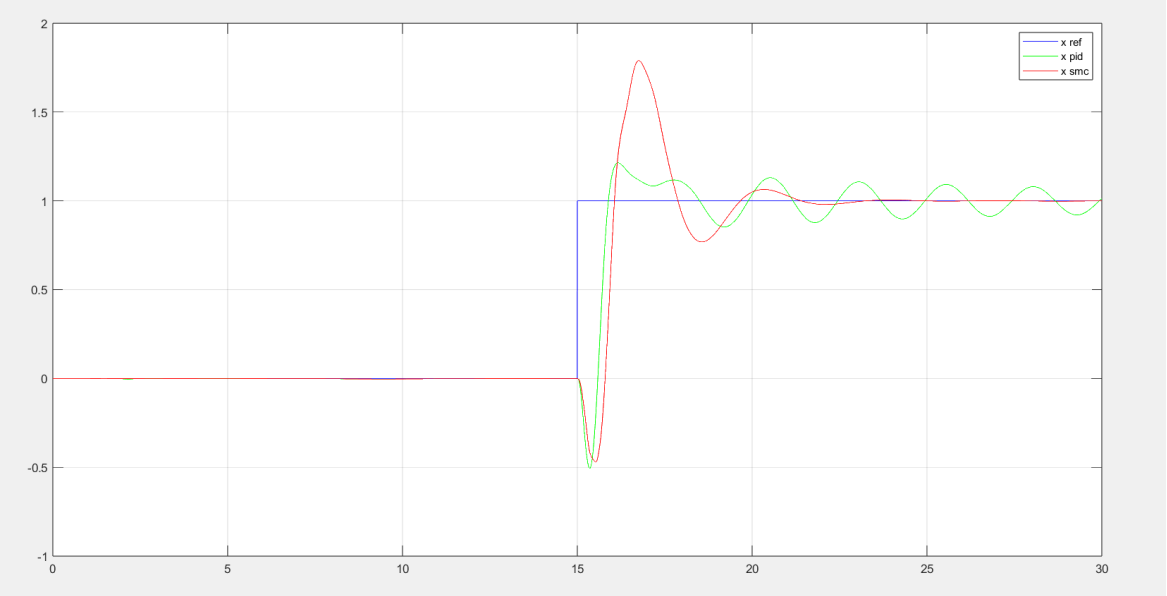


Figura 7.1: Andamento della coordinata x

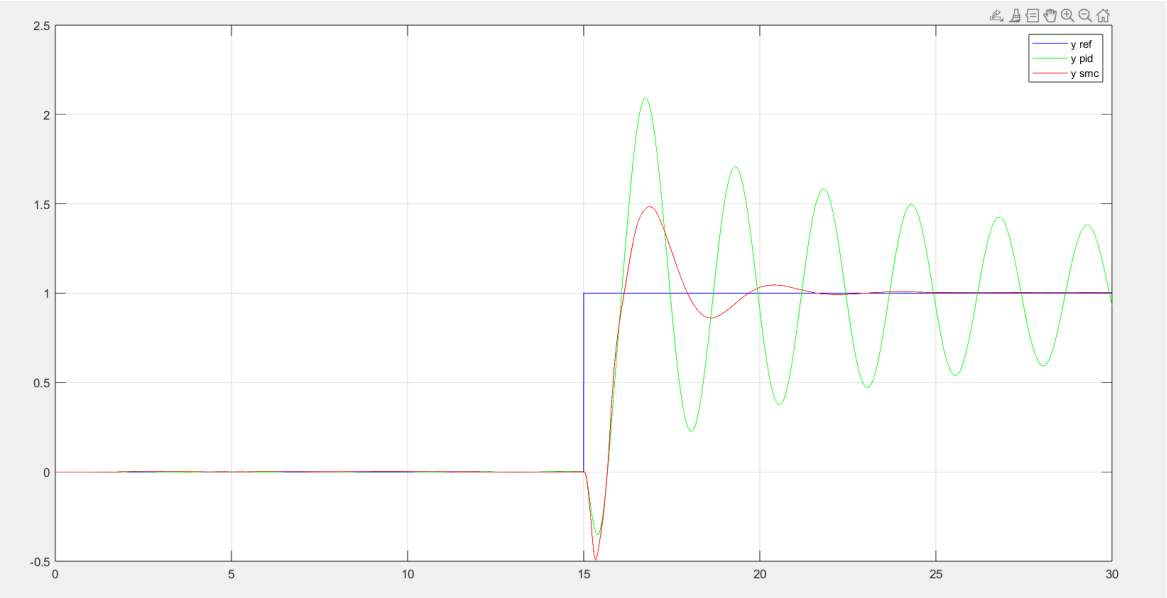


Figura 7.2: Andamento della coordinata y

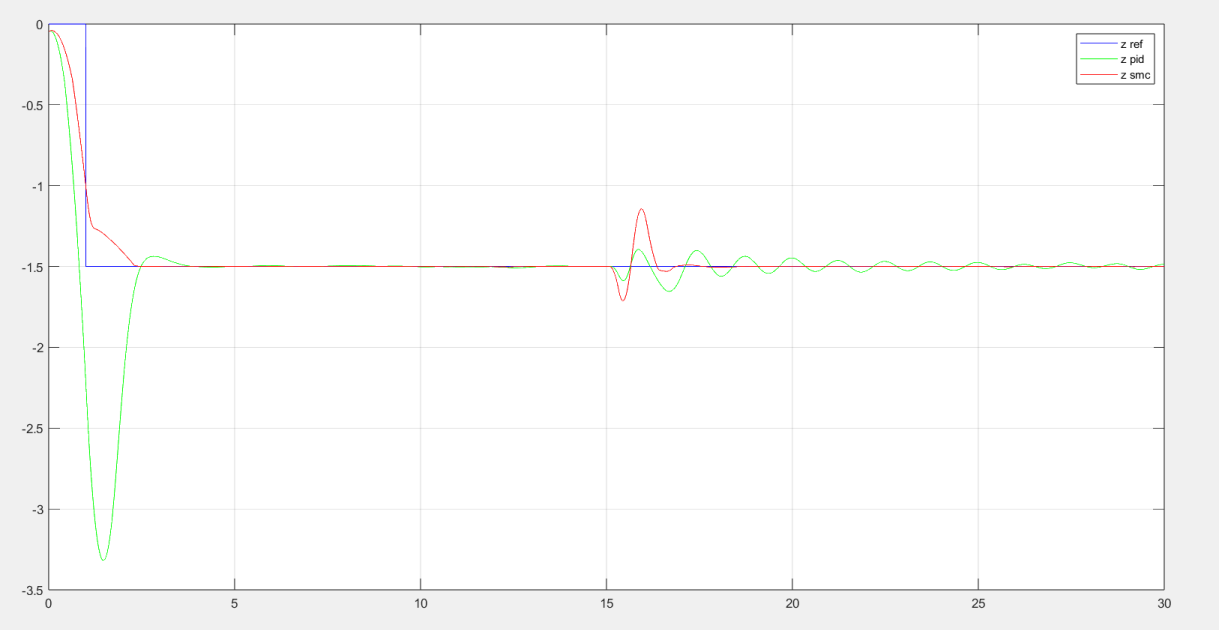


Figura 7.3: Andamento della quota z

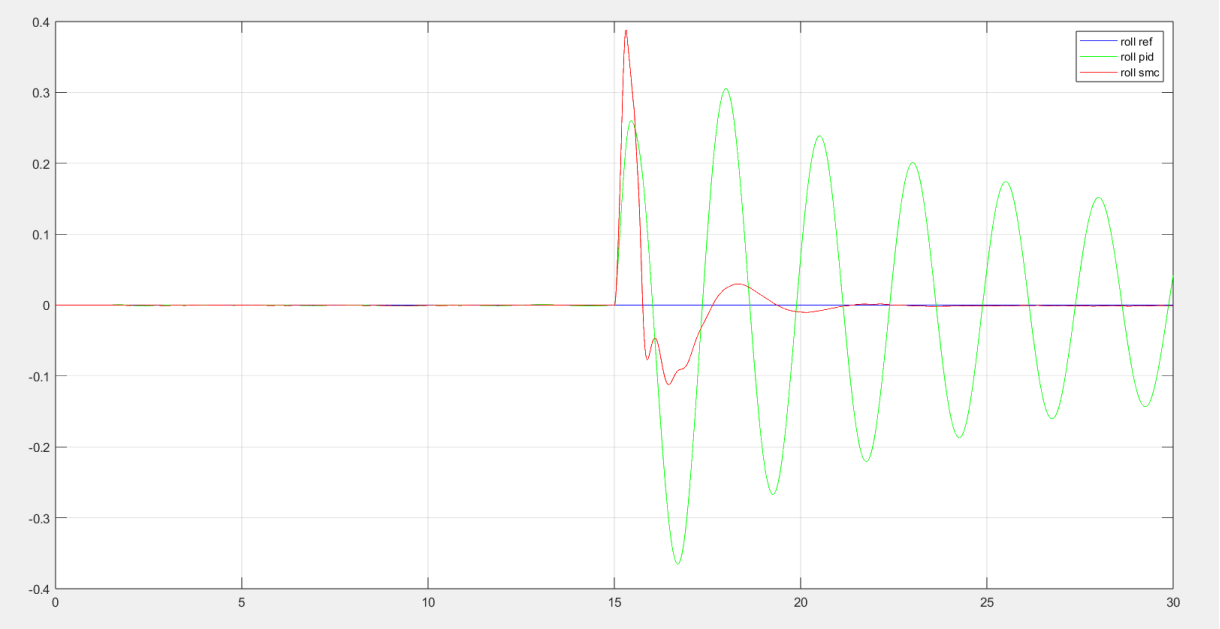


Figura 7.4: Andamento dell'angolo di roll

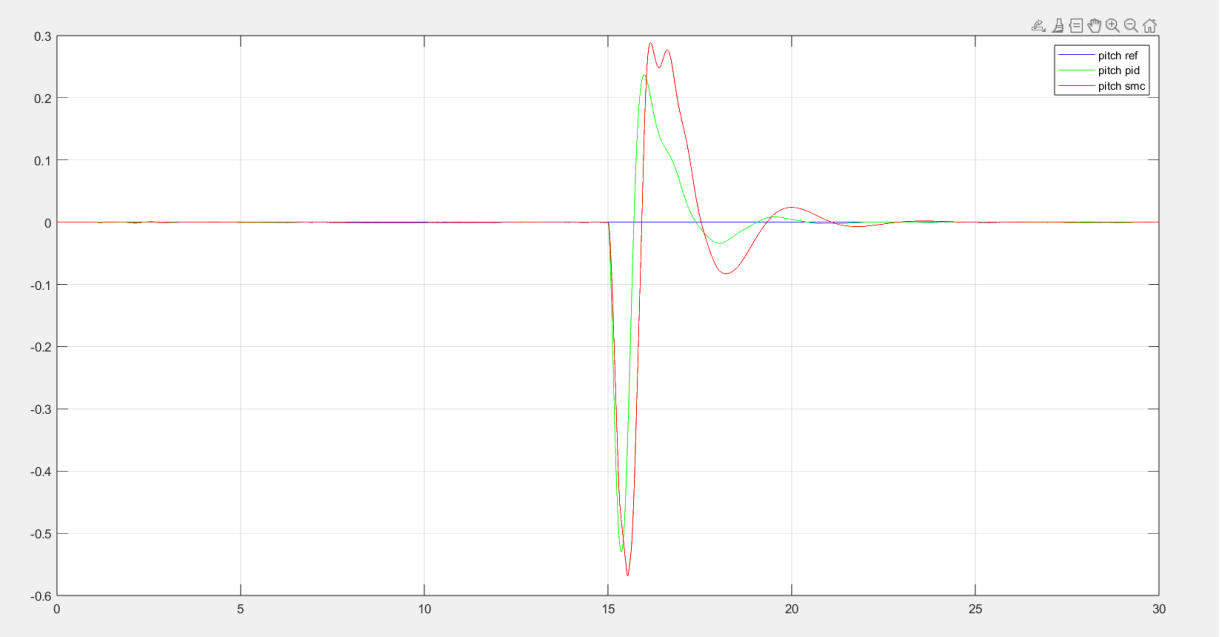


Figura 7.5: Andamento dell'angolo di pitch

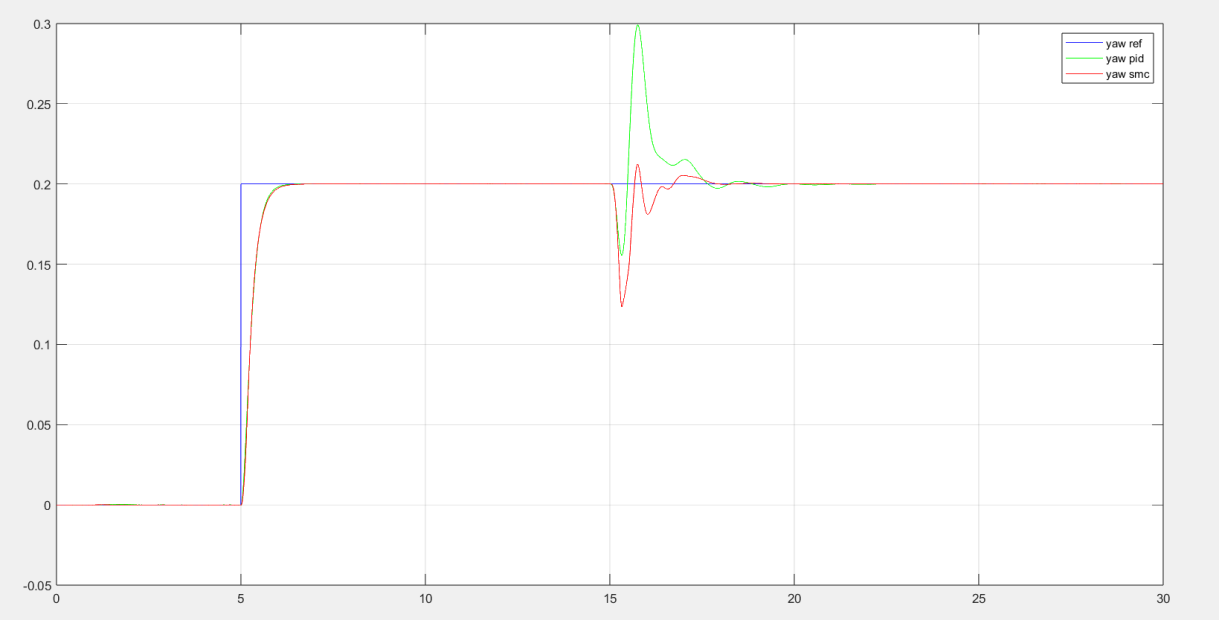


Figura 7.6: Andamento dell'angolo di yaw

7.1.2 Analisi quantitativa: indici di prestazione

Passiamo dunque a presentare gli indici di prestazione di un sistema di controllo, basati sull'errore commesso dall'azione di controllo.
Questo metodo si basa sul calcolo di tre indicatori: Integral of Absolute Error (IAE), Integral of Squared Error (ISE) e Integral of Time multiply Absolute Error (ITAE), che misurano l'errore del sistema rispetto ai riferimenti desiderati. I 3 indici che ci apprestiamo ad adoperare si trovano all'interno del toolbox di Matlab chiamato *Simulink Library: Performance Index*:

- **IAE - Integral of Absolute Error**: è dato dall'integrale del valore assoluto dell'errore nell'intervallo di tempo di interesse. Rappresenta l'errore complessivo che si commette e tratta con lo stesso peso tutti gli istanti di tempo.

$$IAE = \int_o^t |e(t)|\, dt \tag{7.1}$$

- **ISE - Integral of Squared Error**: è dato dall'integrale del quadrato dell'errore nell'intervallo di tempo di interesse. Anch'esso rappresenta l'errore complessivo ma dà maggior peso agli errori più grandi.

$$ISE = \int_o^t e(t)^2\, dt \tag{7.2}$$

- **ITAE - Integral of Time multiply Absolute Error**: è dato dall'integrale del valore assoluto dell'errore, pesato con l'istante nel quale viene calcolato. Come l'IAE e l'ISE, l'ITAE rappresenta l'errore complessivamente commesso, ma dà maggior peso agli errori commessi più avanti nel tempo, pesando meno quelli commessi nel transitorio.

$$ITAE = \int_o^t t|e(t)|\, dt \tag{7.3}$$

Per incorporare gli indicatori di prestazione nel progetto asbQuadcopter, è stato inserito un blocco chiamato Indici di prestazione, che aggiornerà i valori dei 3 parametri alla fine di ciascun intervallo di campionamento (scandito ogni 0.005 secondi); infine si potranno visualizzare all'interno dei blocchi Display. Come si può notare dalla Figura 7.8, il calcolo di ciascun indice di prestazione viene effettuato utilizzando i blocchi corrispondenti forniti dalla *Simulink Library: Performance Index*, che vanno a prendere l'errore calcolato come differenza tra la variabile misurata (dentro States) e la variabile di riferimento (dentro Command).

Nella tabella 7.1 riportiamo il confronto tra i valori degli indici di prestazione dei PID originali e dei controllori sliding mode. Come si può osservare dalla tabella, il controllo sliding mode ha migliorato notevolemente le prestazioni del Parrot Mambo: questo si può notare dal fatto che generalmente i valori relativi al controllo sliding

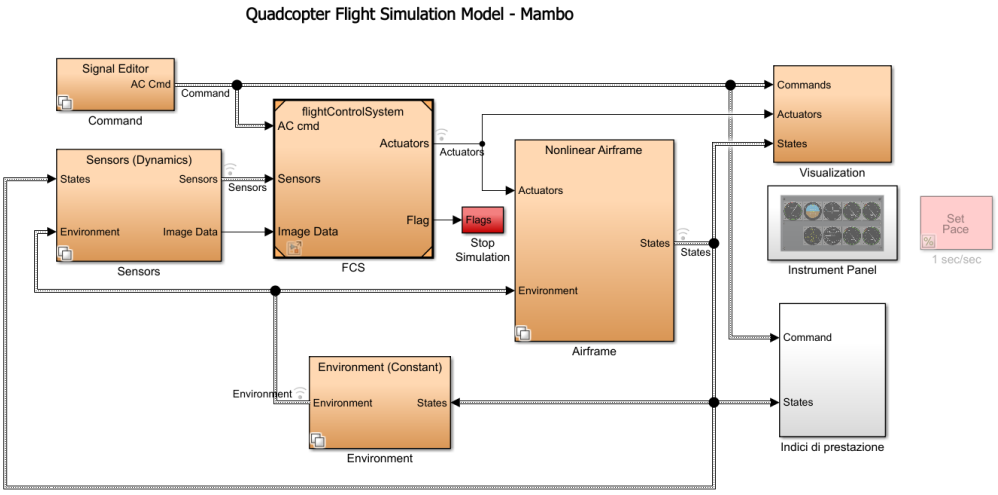


Figura 7.7: Aggiunta del blocco Indici di prestazione

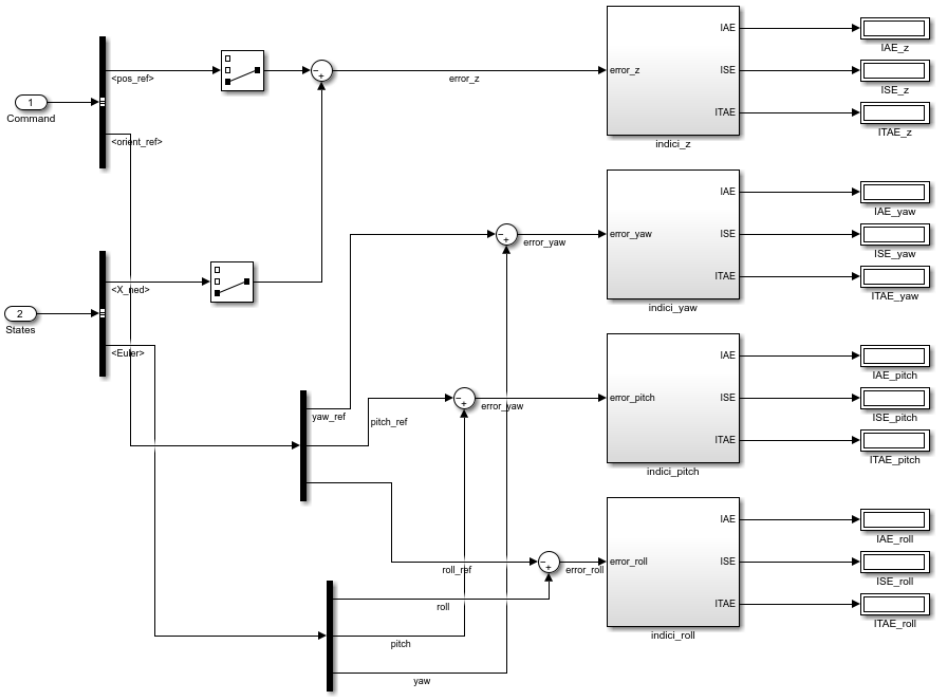


Figura 7.8: Blocco Indici di prestazione

mode sono minori dei valori relativi ai PID originali, eccezion fatta per l'angolo di pitch, il cui andamento risulta leggermente peggiorato, ma in un modo comunque accettabile.

Tabella 7.1: Indici IAE, ISE e ITAE a confronto (gradino)

	IAE PID	**IAE smc**	**ISE PID**	**ISE smc**	**ITAE PID**	**ITAE smc**
z	2.834	0.863	3.238	0.3237	11.65	4.821
yaw	0.1491	0.1247	0.01105	0.0093	1.846	1.501
pitch	0.4584	0.74124	0.1212	0.2132	7.35	12.25
roll	2.109	0.3376	0.3986	0.05863	44.89	5.509

7.2 Simulazione al pc: riferimenti a rampa

Non soddisfatti del risultato, e volendo osservare la risposta del drone ad un ingresso di riferimento a rampa per quanto riguarda le coordinate x e y, abbiamo modificato il blocco Signal builder per la generazione dei riferimenti (Figura 7.9).
La differenza concettuale rispetto all'uso di riferimenti a gradino giace nel fatto che se prima ci si aspettava che il drone istantaneamente passasse da un punto dello spazio ad un altro, ora ci si aspetta che esso si attenga alla traiettoria coincidente alla diagonale di un quadrato di lato 1 m e avente un vertice nell'origine.
A questo punto, per un'analisi qualitativa, verranno presentati i grafici degli sforzi di controllo relativi alle variabili di interesse e i grafici del confronto tra i 6 gradi di libertà del drone prima e dopo l'implementazione del controllo sliding mode e i riferimenti.
Infine, per quanto riguarda l'analisi quantitativa, verrà illustrata la tabella contenente il confronto tra gli indici di prestazione calcolati prima e dopo l'implementazione dello slididng mode.

Come si può osservare, gli sforzi di controllo relativi al roll e al pitch sono dell'ordine di 10^{-4} (Figura 7.10) e 10^{-5} (Figura 7.11), mentre lo sforzo di controllo dello yaw è dell'ordine di 10^{-6} per poi arrivare ad un picco di 10^{-4} quando al Parrot Mambo viene chiesto di ruotare attorno al suo angolo di imbardata (Figura 7.12). Infine, per quanto riguarda lo sforzo relativo alla quota, esso presenta un picco iniziale (dovuto al fatto che per il primo secondo di volo non agisce ancora il controllo sliding mode, ma semplicemente la spinta è data dalla forza peso moltiplicata per un guadagno), per poi stabilizzarsi (Figura 7.13).
Dai grafici relativi ai gradi di libertà del drone invece (Figure 7.14, 7.15, 7.16, 7.17, 7.18, 7.19), si può vedere in modo evidente come lo sliding mode riesca a seguire con più facilità il riferimento a rampa delle coordinate x e y, in quanto più "soft" di quello a gradino mentre, per quanto riguarda le altre variabili, si comporta in modo non troppo diverso rispetto al caso di riferimento a gradino: come si è già

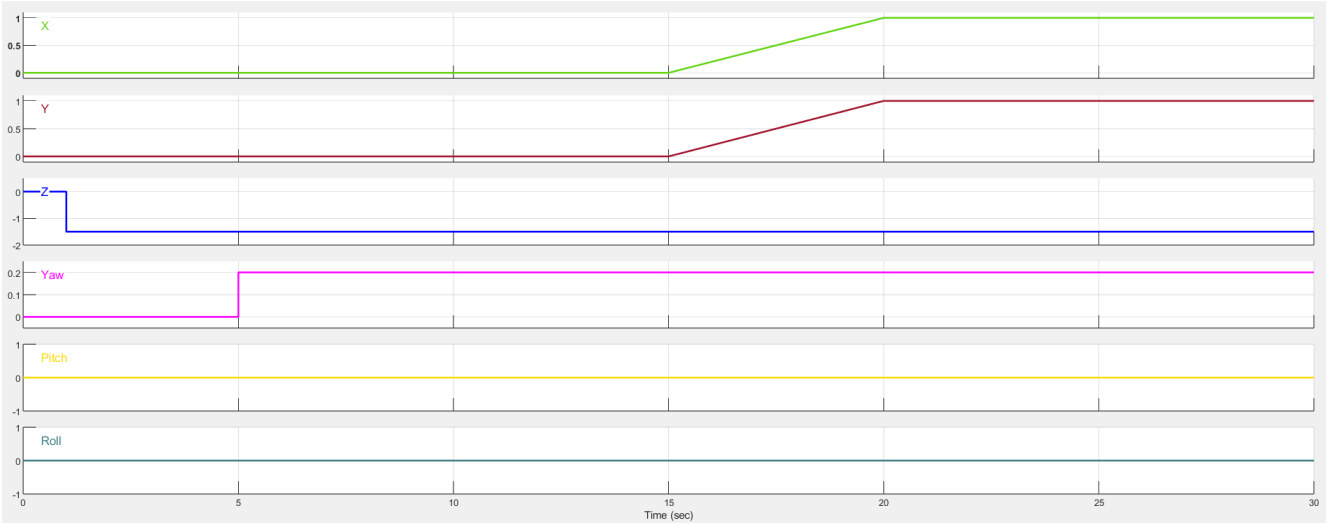


Figura 7.9: Segnali di riferimento a rampa

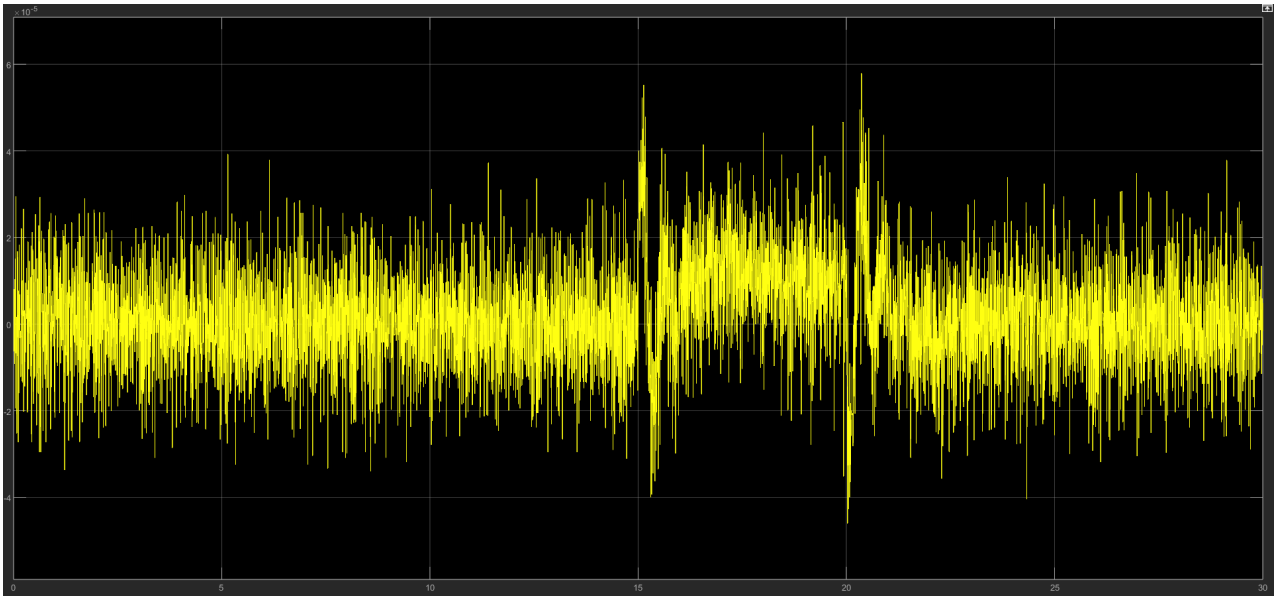


Figura 7.10: Sforzo di controllo per l'angolo di roll

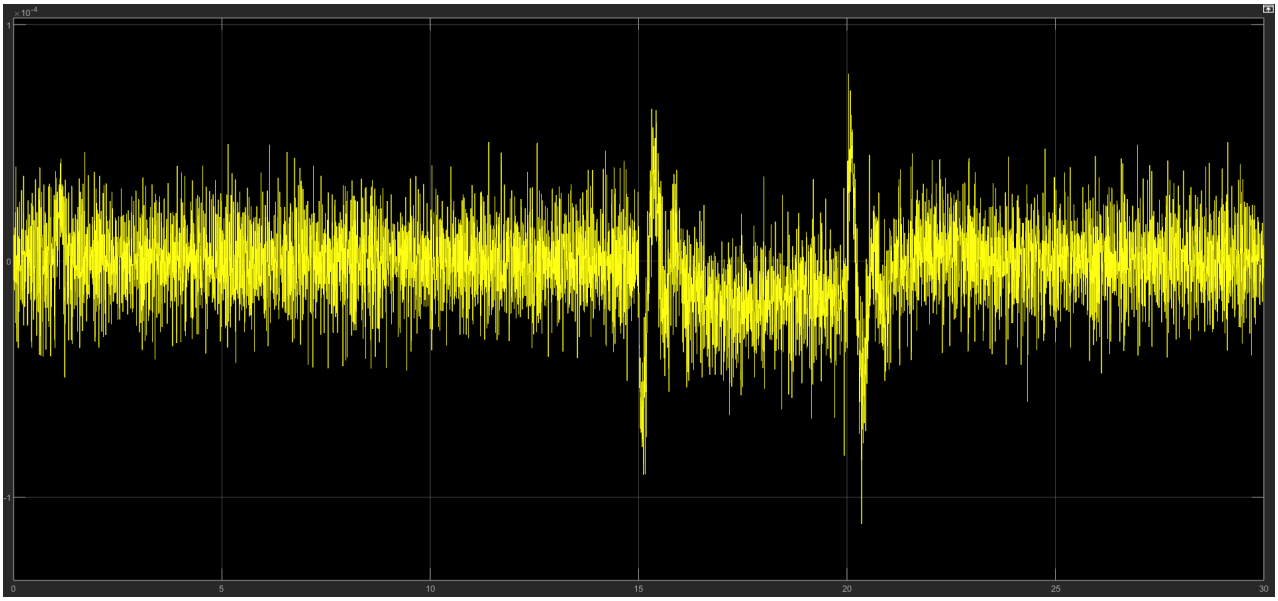


Figura 7.11: Sforzo di controllo per l'angolo di pitch

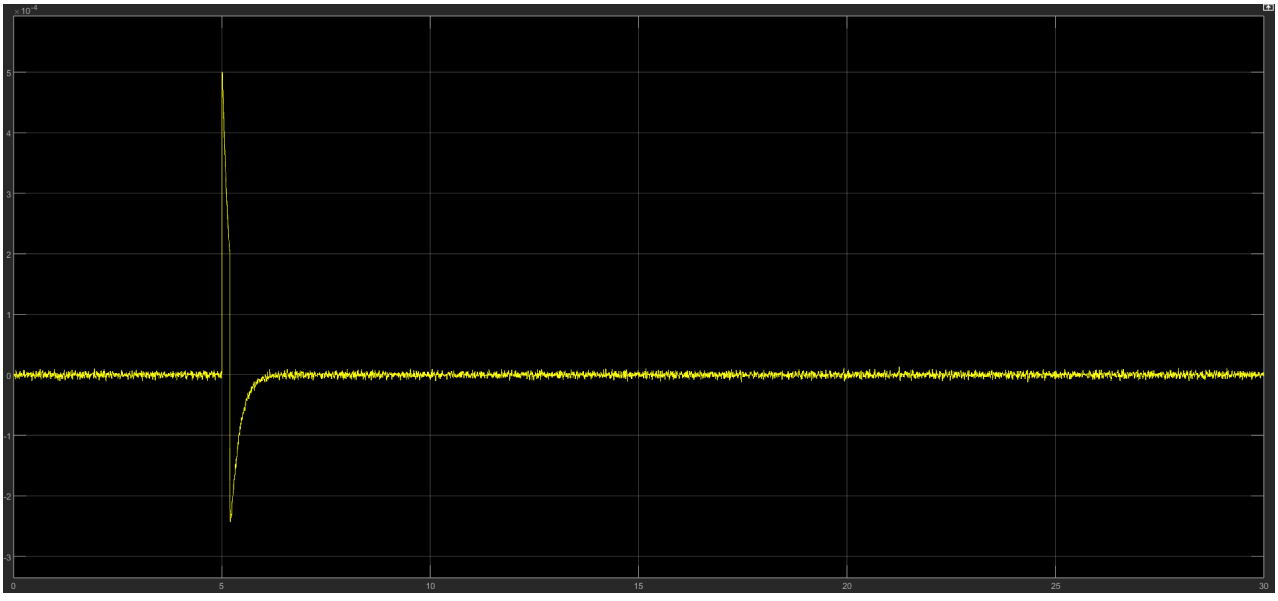


Figura 7.12: Sforzo di controllo per l'angolo di yaw

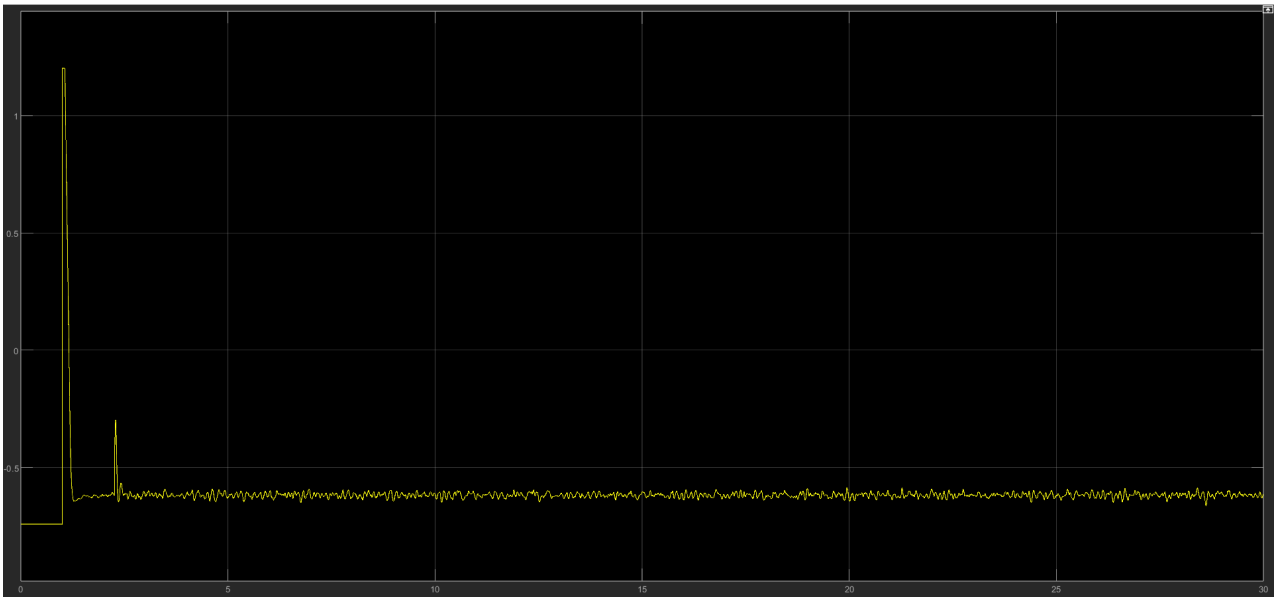


Figura 7.13: Sforzo di controllo per la quota z

visto, il miglior controllo degli angoli di roll e pitch (ovvero senza sovraelongazioni e oscillazioni indesiderate) implica un minore errore di inseguimento nelle traiettorie delle coordinate x e y.

Presentiamo ora la tabella 7.2, da cui si evince che il controllo sliding mode ha nettamente migliorato l'andamento delle 4 variabili di interesse.

Tabella 7.2: Indici IAE, ISE e ITAE a confronto (rampa)

	IAE PID	**IAE smc**	**ISE PID**	**ISE smc**	**ITAE PID**	**ITAE smc**
z	2.478	0.6544	3.212	0.2781	4.841	1.52
yaw	0.06482	0.06582	0.007367	0.007672	0.383	0.3871
pitch	0.08833	0.08964	0.002035	0.001619	1.627	1.686
roll	0.1241	0.06197	0.002065	0.0007241	2.344	1.161

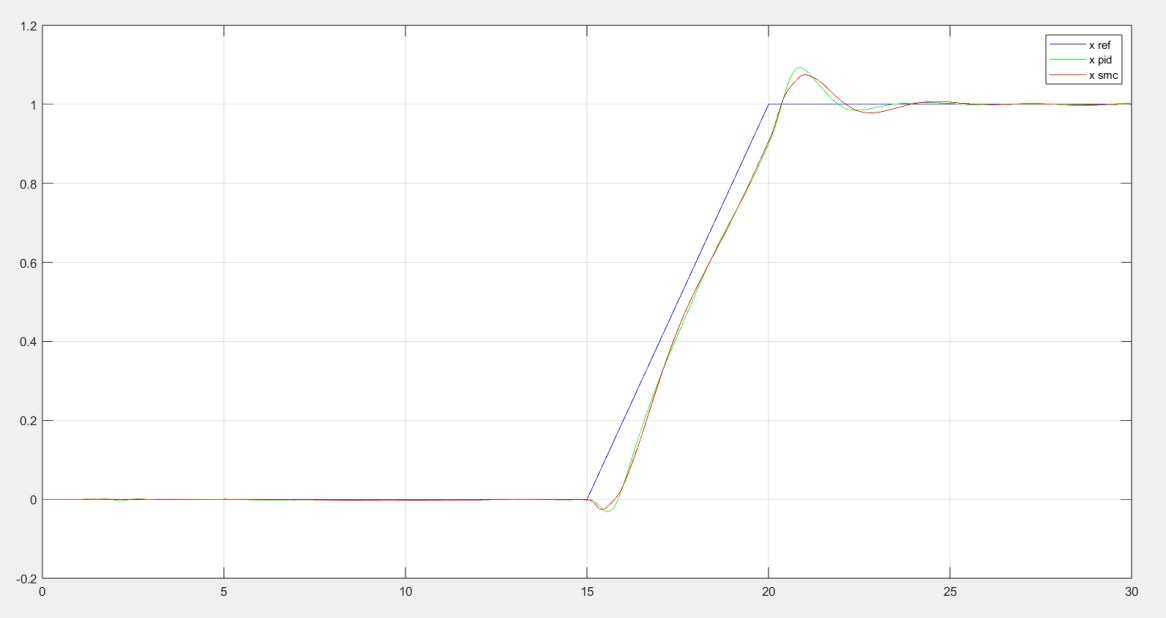


Figura 7.14: Andamento della coordinata x

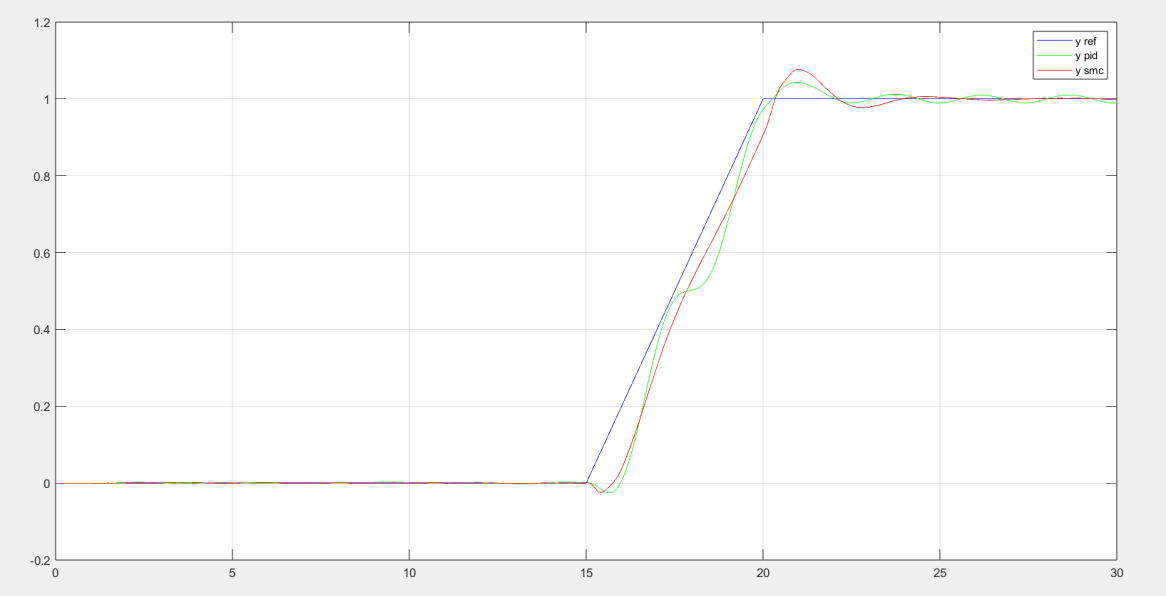


Figura 7.15: Andamento della coordinata y

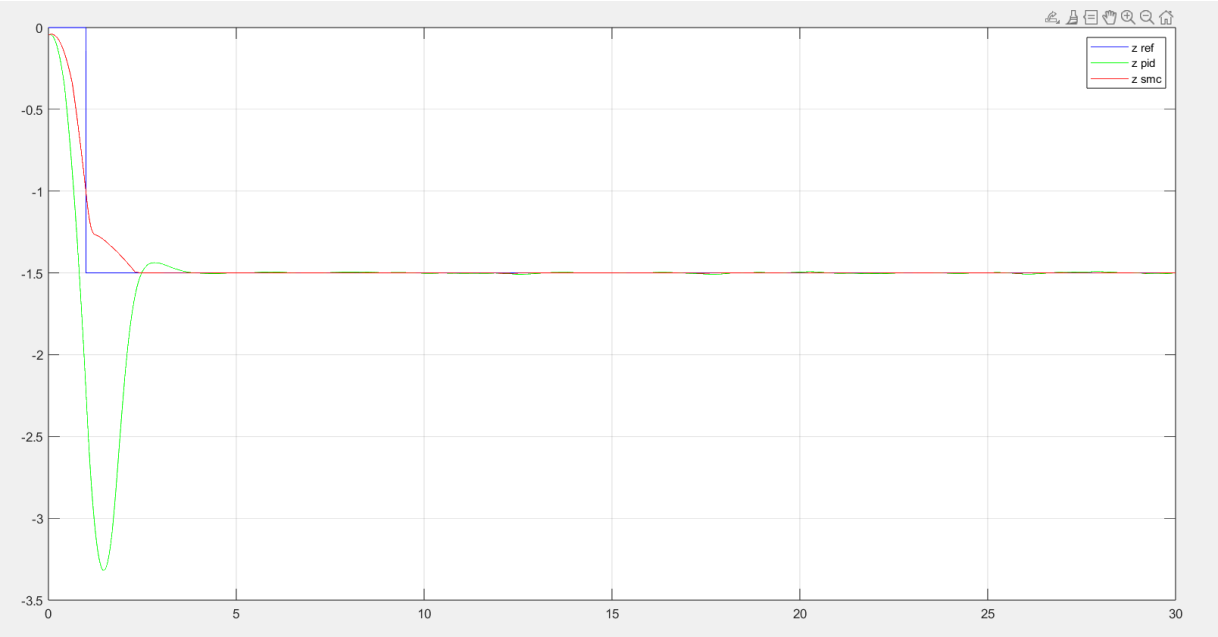


Figura 7.16: Andamento della quota z

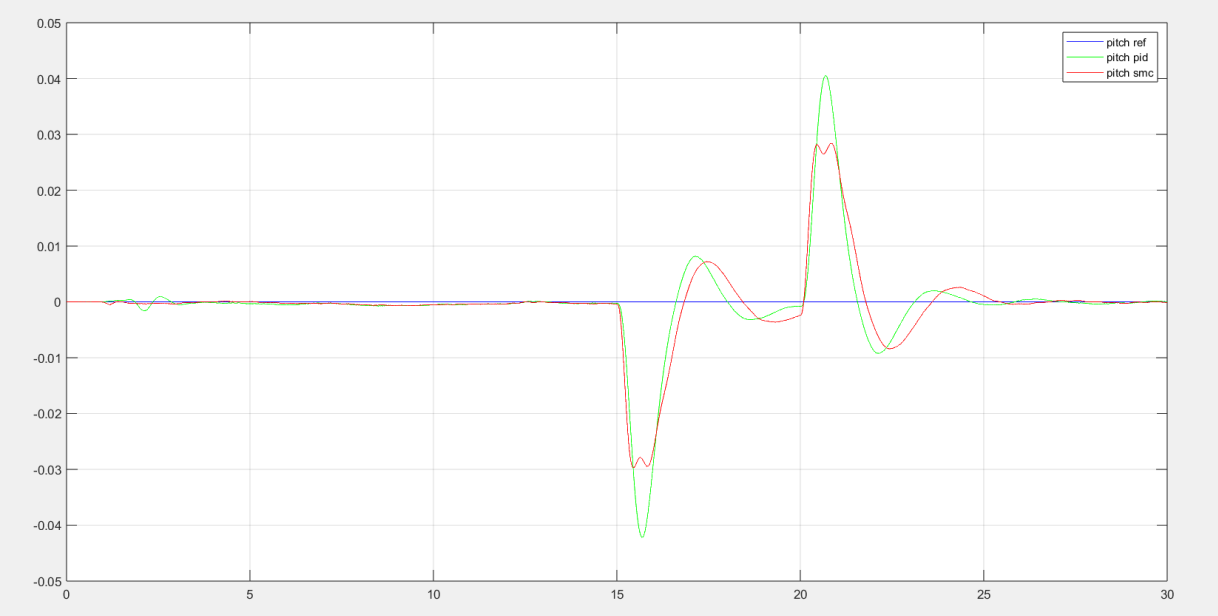


Figura 7.17: Andamento dell'angolo di roll

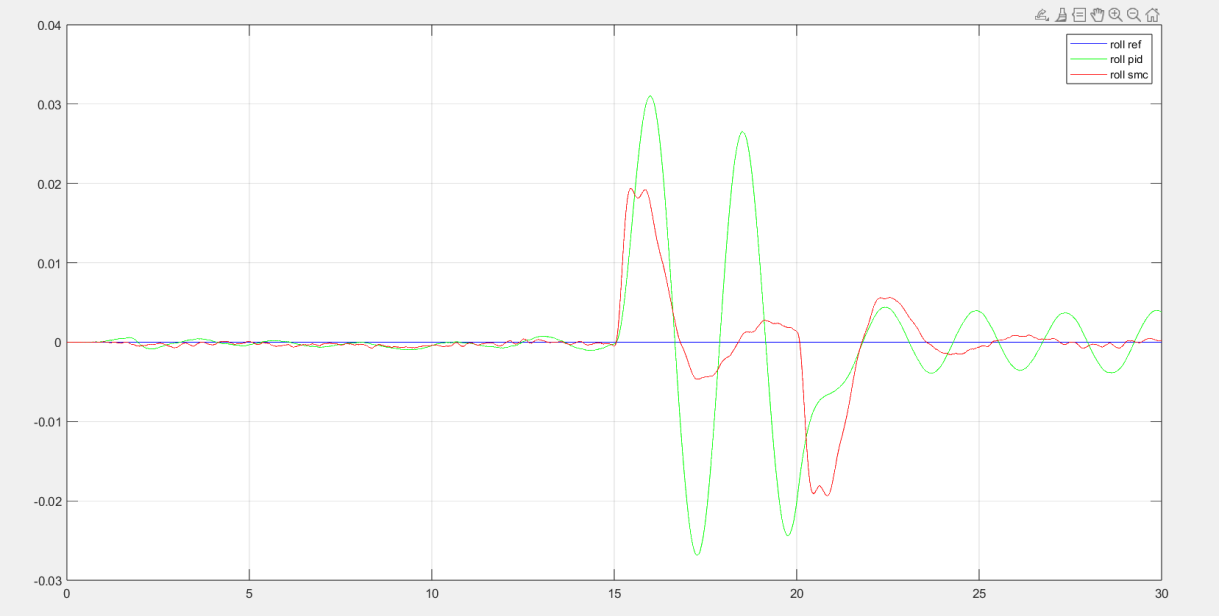


Figura 7.18: Andamento dell'angolo di pitch

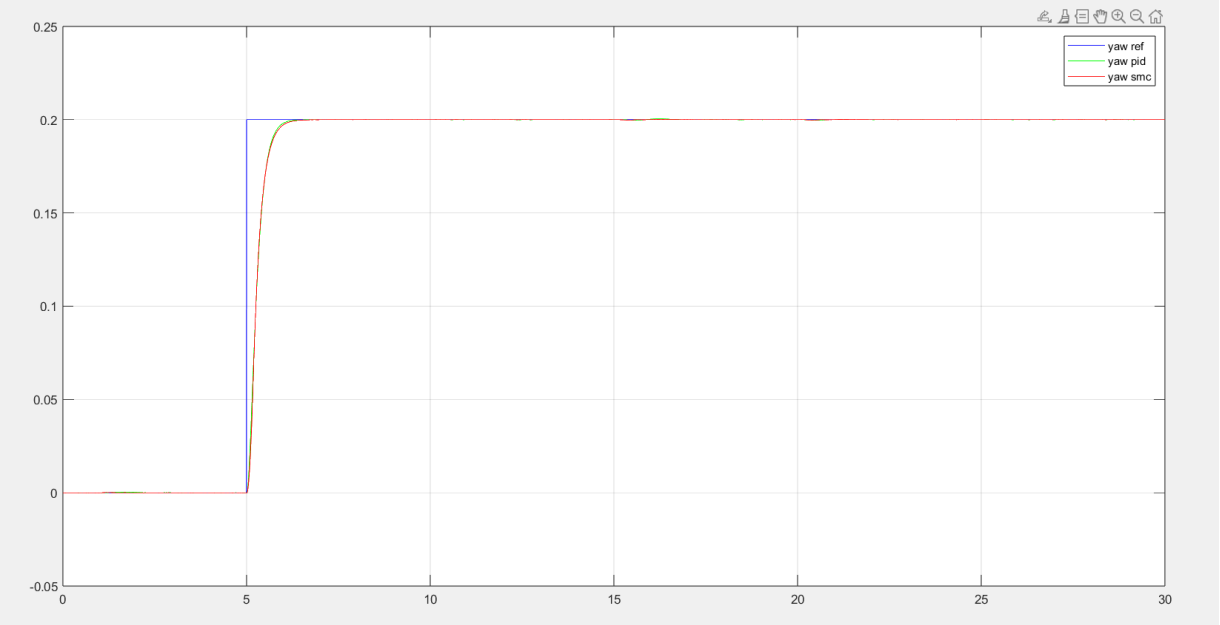


Figura 7.19: Andamento dell'angolo di yaw

Capitolo 8

Conclusioni

Nel presente elaborato di tesi, ci si è posti l'obiettivo di controllare con una tecnica di controllo non lineare la dinamica del Parrot Mambo, un quadricottero la cui modellazione e il cui progetto dei controllori ci sono stati facilitati dal progetto asbQuadcopter di Simulink. Nello specifico siamo andati a controllare con lo sliding mode i 3 angoli di Eulero del drone (angoli di rollio, beccheggio e imbardata) e la sua quota, mentre il controllo delle variabili x e y è stato affidato ai regolatori PID contenuti nel progetto originale.
In una prima fase sono stati scelti come ingressi di riferimento dei segnali a gradino: andando ad effettuare il confronto tra l'andamento delle variabili controllate prima e dopo l'implementazione dello sliding mode si è visto e valutato, grazie agli indici di prestazione, che questa tecnica ha migliorato le prestazioni di volo del quadricottero, mostrandoci una notevole velocità di risposta, eccezion fatta per l'angolo di beccheggio, il cui controllo risulta leggermente peggiorato.
Per questo motivo, in una seconda fase, sono stati scelti dei riferimenti a rampa, che risultano più morbidi: non ci ha sorpreso vedere come il controllore sliding mode abbia reagito ancora meglio, mostrandoci errori ancora più piccoli.

Rivolgendo uno sguardo al futuro, dato che ci si è fermati alla simulazione, si potrebbe pensare di implementare il controllore all'interno del drone reale, combinando i risultati ottenuti in questo elaborato con degli algoritmi di visione per l'inseguimento di traiettorie. In alternativa, ci si potrebbe divertire a progettare altre operazioni più complesse da far eseguire al drone, per valutare l'efficacia e i limiti operativi dello sliding mode.

Bibliografia

[1] Manuel Gizzarone. *Studio e sviluppo di sistemi di controllo lineari per l'assetto di droni.* Tesi triennale, UNIVPM - Università Politecnica delle Marche, 2022.

[2] Francesco Paoli Leonardi. *Studio e sviluppo di tecniche di controllo avanzate per droni.* Tesi triennale, UNIVPM - Università Politecnica delle Marche, 2020.

[3] Simone Terramani. *Studio e sviluppo di sistemi di controllo robusto per droni.* Tesi triennale, UNIVPM - Università Politecnica delle Marche, 2018.

[4] Alberto Isidori. *Sistemi di controllo.* Edizioni siderea edition, 1998.

[5] Riccardo Scattolini Nicola Schiavoni, Paolo Bolzern. *Fondamenti di controlli automatici.* McGraw-Hill, 2015.

[6] Weiping Li Jean-Jacques Slotine. *Applied nonlinear control.* Prentice Hall.

[7] V.I. Utkin. *Sliding modes and their application in variable structure systems.* Mosca, 1978.