UNIVERSITÀ POLITECNICA DELLE MARCHE

FACOLTÀ DI INGEGNERIA

Corso di Laurea Triennale in Ingegneria Informatica e dell'Automazione

Studio e sviluppo di regolatori di assetto per droni

Study and development of attitude regulators for drones

Relatore:
Prof. Gianluca Ippoliti

Tesi di Laurea di:
Silvestri Lorenzo

Correlatore:
Prof. Giuseppe Orlando

A.A. 2020/2021

Studio e sviluppo di regolatori di assetto per droni

Lorenzo Silvestri

Ottobre 2021

Abstract

La seguente Tesi ha come scopo quello di fornire un'analisi dettagliata del modello Simulink del minidrone "Mambo" allo scopo di procedere alla sintesi di un controllore mediante il **Luogo delle Radici** per il beccheggio (pitch) del dispositivo

Il tutto è stato possibile tramite l'ausilio del pacchetto Matlab *"Simulink Support Package for Parrot Minidrones"*; la prima cosa realizzata è stata un'analisi del modello per capirne il comportamento della dinamica e poi il successivo step è stato la realizzazione vera e propria del controllore.

Lo studio del comportamento del dispositivo ha permesso una comprensione ampia e dettagliata del comportamento del drone e ha permesso una conoscenza importante dei vari blocchi presenti e delle variabili di controllo.

Infine è stato realizzato il controllore, tramite la tecnica nominata sopra, ed è stato poi implementato in Simulink, sostituendo il vecchio controllore di default cioè il **Controllore PID**.

Indice

Introduzione

Il mondo dei droni è un universo in continua evoluzione che non smette di stupire.

Con il passare del tempo sono infatti sempre più ampie le applicazioni che questi piccoli velivoli trovano nei contesti più disparati. Il ventaglio di attività connesse agli apparecchi SAPR è davvero corposo. Molto diffuso è per esempio l'utilizzo dei droni in agricoltura.

Si ricorre altrettanto proficuamente all'ausilio dei droni per riprese aeree professionali che riguardano settori di ogni genere: dal turismo all'edilizia. Frequente è anche l'utilizzo dei droni in ambiti ambientali per monitorare il territorio dall'alto, stanando pericolose discariche abusive o impedendo il divampare di devastanti incendi boschivi. Altre aree di sviluppo dove i droni possono essere impegnati sono quella medico-sanitario e della logistica delle spedizioni a breve raggio, in cui i droni vengono utilizzati per il trasporto di merci pilotando da remoto il veicolo.

Un altro settore dove possiamo trovarli è quello militare nel quale vengono utilizzati per operazione di spionaggio o di ricerca dei dispersi grazie allo streaming delle immagini, che viene trasmesso di solito da una telecamera collegata al drone che permette a chi lo pilota di vedere e filmare tutto ciò che vede.

Una specifica classe di droni è costituita dai veicoli pale rotanti come "tricotteri", "quadricotteri", "eptacotteri", ecc. Questi si prestano grazie alla loro maneggevolezza (in particolare alla possibilità di mantenersi in volo stazionario), alle misure ridotte, al costante miglioramento dell'autonomia di volo, ad applicazioni sempre più estese.

Questa tesi si concentra sulla progettazione di un sistema di controllo per minidroni autonomi, pertanto prima è stato necessario ottenere un analisi del modello ed ottenere una rappresentazione fedele di esso, al fine di capire bene i meccanismi e la dinamica del veicolo, e infine, utlizzando le tecniche apprese, è stato realizzato il controllore vero e proprio. Questo descritto ora, è stato realizzato utilizzando un progetto all'interno del Simulink, in cui è stato implementato il nuovo controllore.

Tra le varie aziende costruttrici di droni troviamo l'azienda Parrot S.A. che in collaborazione con il gruppo di sviluppatori di Matlab ha messo a disposizione degli utenti un pacchetto il "Support Package for Parrot Minidrones" che permette di caricare un ambiente di modellazione e sviluppo, il simulatore di volo e tutte le variabili necessarie per lo studio del drone "Mambo".

All'interno di questo ambiente di lavoro e di sviluppo, è stato possibile modificare e settare determinati blocchi e varibili per poter ot-

tenere il funzionamento ricercato. Inoltre modificando una variabile di ambiente è possibile passare dal modello non-lineare al modello lineare, accedere a tutti i blocchi adibiti all'analisi dei dati provenienti dai sensori di bordo e comprendere il funzionamento degli algoritmi di controllo (implementati mediante **PID controller**). Sfruttando le caratteristiche di Simulink è stato possibile isolare, all'interno dei vari blocchi, le variabili coinvolte nei processi considerati. È stato possibile studiare un modello "ridotto" relativo esclusivamente ai legami ingresso/uscita di una delle sei variabili di controllo (nel nostro caso l'angolo di beccheggio o pitch).

Attraverso i dati ottenuti dalle varie analisi descritte in precedenza, è stato possibile analizzare il comportamento e l'andamento temporale del nostro nuovo controllore, tramite l'ausilio di un toolbox di matlab chiamato *sisotool*, tramite il quale è possibile ottenere i vari diagrammi di Bode, Nyquist, Luogo delle radici e risposta al gradino, della funzione implementata in matlab. Pertanto tramite questo toolbox è stato possibile realizzare un controllore tramite Luogo delle Radici, che è andato a sostituire il controllore di default già presente.

1 Modello Matematico del Quadcopter

La presente sezione fornisce una breve esposizione dei passaggi necessari per giungere ad un modello fisico di un quadricottero che descriva la dinamica del sistema. Il processo di modelling si basa su una serie di assunzioni che agevolano la trattazione e permettono di ottenere un modello semplificato implementabile a livello software:

1. Struttura rigida,possibilità di trascurare i contributi della sessione dell'intelaiatura;
2. Struttura simmetrica, possibilità di valutare una matrice di inerzia diagonale;
3. Coincidenza tra centro di gravità (CoG:Centre of Gravity) e origine del sistema di riferimento solidale al veicolo (Body Fixed Frame);
4. Rigidità delle eliche
5. Spinta (thrust) e resistenza aereodinamica (drag) proporzionali al quadrato delle velocità angolari delle eliche w^2.

1.1 Derivazione delle grandezze fisiche fondamentali

Un quadricottero, in termini generali, può essere dotato di 6 gradi di libertà (DoF : Degrees of Freedom):

- 3 direzioni di traslazione (X,Y,Z) : forniscono lo spostamento lineare del veicolo
- 3 direzioni di rotazione (angoli ϕ, θ, ψ) : forniscono lo spostamento angolare del veicolo

Per poter fornire una corretta rappresentazione nello spazio della dinamica del sistema, capace di comprendere sia movimenti lineari che angolari, è necessario stabilire tre diversi sistemi di riferimento:

1. Sistema di riferimento Inerziale;
2. Sistema di riferimento solidale al veicolo avente come origine il centro di gravità (Body Fixed Frame);
3. Sistema di riferimento allineato al quadro di riferimento inerziale con origine il centro di gravità (Vehicle Frame)

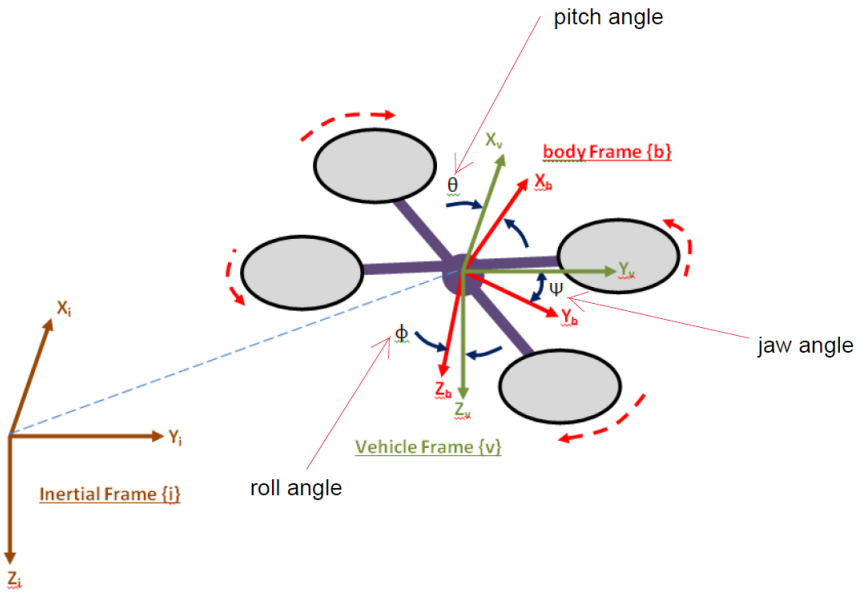


Figure 1: Earth,Body and Vehicle Frame

Le misurazioni lineari sono ottenute direttamente appena conosco la posizione del veicolo, mentre per conoscere lo spostamento angolare

ho bisogno di matrici quadrate particolari chiamate **Matrici di Rotazione** le quali definiscono gli spostamenti angolari ottenibili, fissato un asse, effettuando rotazioni rispetto agli altri due assi "liberi".

Gli angoli ϕ, θ, ψ sono gli angoli di Eulero e descrivono la rotazione del sistema di riferimento solidale al corpo rispetto a quello inerziale. Ecco le matrici definite sopra:

- Matrice di rotazione attorno all'asse X : R(X,ϕ):

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix}$$

- Matrice di rotazione attorno all'asse Y: R(Y,θ):

$$\begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix}$$

- Matrice di rotazione attorno all'asse Z: R(Z,ψ):

$$\begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Le matrici ottenute e riportate sopra sono matrici *ortonormali* cioè $R^T = R^{-1}$. La matrice di rotazione completa la si ottiene dal prodotto ordinato tra le diverse matrici di rotazione.

Denominando **b** il vettore delle coordinate rispetto al Body Fixed Frame e **r** il vettore delle coordinate rispetto al sistema inerziale otteniamo:

$$\begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{1}$$

cioè:

$$\begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} \cos\theta\cos\phi & \cos\theta\sin\phi & -\sin\theta \\ -\cos\phi\sin\psi + \sin\phi\sin\theta\cos\psi & \cos\phi\cos\psi + \sin\phi\sin\theta\sin\psi & \sin\phi\cos\theta \\ \sin\phi\sin\psi + \cos\phi\sin\theta\cos\psi & -\sin\phi\cos\psi + \cos\phi\sin\theta\sin\psi & \cos\phi\cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{2}$$

che in maniera compatta è scritta:

$$\begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} = R_i^b * \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{3}$$

con R_i^b matrice di trasferimento del sistema di riferimento inerziale al Body Fixed Frame. Allo stesso modo, sfruttando l'ortonormalità delle matrici, è possibile operare per ricavare la matrice di trasferimento opposta, cioè quella dal Body Fixed Frame al sistema di riferimento inerziale:

$$r = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = (R_i^b)^T * \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} = R_b^i * \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \tag{4}$$

per cui esplicitando i termini della matrice otteniamo:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} \cos\theta\cos\phi & -\cos\phi\sin\psi + \sin\phi\sin\theta\cos\psi & \sin\phi\sin\psi + \cos\phi\sin\theta\cos\psi \\ \cos\theta\sin\psi & \cos\phi\cos\psi + \sin\phi\sin\theta\sin\psi & -\sin\phi\cos\psi + \cos\phi\sin\theta\sin\psi \\ -\sin\theta & \sin\phi\cos\theta & \cos\phi\cos\theta \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \tag{5}$$

Ottenute le matrici che consentono il passaggio dal sistema di riferimento solidale al veicolo a quello inerziale, si possono ricavare le altre grandezze fisiche di interesse, ad iniziare dalla velocità.

Le componenti della velocità traslazionale relative al sistema di riferimento solidale al veicolo sono indicate all'interno del vettore v_b con le lettere u,v,w:

$$\mathbf{v}_b = \begin{bmatrix} u & v & w \end{bmatrix}^T$$

Per ottenere le velocità traslazionali rispetto alle coordinate di navigazione è sufficiente eseguire la seguente trasformazione:

$$\mathbf{v}_i = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = R_b^i * \begin{bmatrix} u \\ v \\ w \end{bmatrix} \tag{6}$$

I sensori che misurano le velocità angolari (sono nell'unità IMU), forniscono la variazione degli angoli di roll, pitch, e yaw, indicati all'interno del vettore w_b con le lettere p,q,r:

$$\mathbf{w}_b = \begin{bmatrix} p & q & r \end{bmatrix}^T$$

E' possibile passare da queste componenti a componenti relative alla variazione degli angoli euleriani, che descrivono la rotazione del sistema rispetto al sistema di riferimento inerziale, è garantito dalla seguente trasformazione:

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \sin\phi\frac{\sin\theta}{\cos\theta} & \cos\phi\frac{\sin\theta}{\cos\theta} \\ 0 & \cos\phi & -\sin\phi \\ 0 & \frac{\sin\phi}{\cos\theta} & \frac{\cos\phi}{\cos\theta} \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} = Q_b^i \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{7}$$

Possiamo notare come la variazione di ψ dipende dagli altri due angoli ovvero ϕ e θ, cioè le manovre di pitch e roll possono causare una rotazione lungo l'asse verticale del veicolo (yaw).

La trasformazione inversa, che fornisce le variazioni delle velocità angolari rispetto al sistema di riferimento solidale al veicolo, si ottiene applicando la seguente matrice:

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} 1 & 0 & -\sin\theta \\ 0 & \cos\phi & \sin\phi\cos\theta \\ 0 & -\sin\phi & \cos\phi\cos\theta \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = Q_i^b \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \tag{8}$$

1.2 Rappresentazione della dinamica rotazionale del sistema in termini langrangiani

Una volta ricavate le grandezze fisiche fondamentali è possibile avvalersi delle equazioni di **Eulero-Lagrange** per descrivere la dinamica rotazionale del sistema. Poichè abbiamo supposto la rigidità della struttura, ovvero un vincolo di tipo olonomo, possiamo utilizzare un sistema di coordinate generalizzate (in particolare gli "angoli di Eulero" introdotti precedentemente e le componenti del vettore posizione valutate rispetto al *Body Fixed Frame*) per descrivere la rotazione del sistema.

Un sistema di coordinate generalizzate, in presenza di forze di tipo conservativo, permette di esprimere le equazioni del moto in termini di funzioni scalari, considerando esclusivamente l'*Energia Cinetica (T)* e l'*Energia Potenziale (V)*. La **Langrangiana** (L) di un sistema fisico è definita come la differenza tra l'energia cinetica e l'energia potenziale L=T-V, se questa quantità è nota allora le equazioni della dinamica del sistema si possono scrivere:

$$\Gamma_i = \frac{\mathrm{d}}{\mathrm{dt}}\left(\frac{\partial}{\partial \dot{q}_i} L\right) - \frac{\partial}{\partial q_i} L \tag{9}$$

dove Γ_i sono le forze generalizzate agenti sul sistema e q_i sono le coordinate generalizzate.

E' necessario calcolare T e V cioè energia cinetica e energia potenziale. Per calcolare in termini langrangiani l'energia cinetica, è necessario trasformare le componenti della velocità, espresse in coordinate cartesiane, in coordinate generalizzate utilizzando R_b^i come matrice per il cambiamento di base. Delle componenti del vettore velocità cosi ottenute è necessario quindi calcolare il quadrato del modulo. Sviluppiamo i vari termini e raccolti i prodotti analoghi si ottiene:

$$\begin{aligned}
\|\mathbf{v}_i(b_1,b_2,b_3)\|^2 &= (b_2^2+b_3^2) * (\dot{\psi}^2\sin^2\theta - 2\sin\theta\dot{\phi}\dot{\psi} + \dot{\phi}^2) \\
&+ (b_1^2+b_3^2) * (\dot{\psi}^2\sin^2\phi\cos^2\theta + 2\sin\phi\cos\phi\cos\theta\dot{\theta}\dot{\psi} + \cos^2\phi\dot{\theta^2}) \\
&+ (b_1^2+b_2^2) * (\dot{\psi}^2\cos^2\phi\cos^2\theta - 2\sin\phi\cos\phi\cos\theta\dot{\theta}\dot{\psi} + \sin^2\phi\theta^2) \\
&+ 2b_1b_2 * (\dot{\psi}^2\sin\phi\sin\theta\cos\theta + \dot{\psi}(\cos\phi\sin\theta\dot{\theta} - \sin\phi\cos\theta\dot{\phi}) - \cos\phi\dot{\phi}\dot{\theta}) \\
&+ 2b_1b_3 * (\dot{\psi}^2\cos\phi\sin\theta\cos\theta + \dot{\psi}(-\cos\phi\cos\theta\dot{\phi} - \sin\phi\sin\theta\dot{\theta}) + \sin\phi\dot{\phi}\dot{\theta} \\
&+ 2b_2b_3 * (-\dot{\psi}^2\sin\phi\cos\theta\cos^2\theta + \dot{\phi}(\sin^2\phi\cos\theta\dot{\theta} - \cos^2\phi\cos\theta\dot{\theta}) + \sin\phi\cos\phi\dot{\theta^2})
\end{aligned}$$

Per calcolare l'energia cinetica, è necessario conoscere il contributo di ognuno delle masse infinitesime che compongono il sistema rispetto alla propria posizione espressa attraverso le coordinate generalizzate. L'energia cinetica può essere espressa come:

$$T = \frac{1}{2} \int \mathrm{dm}(\mathbf{r}_i(b_1, b_2, b_3)) * \|\mathbf{v}_i(b_1, b_2, b_3\|^2 \tag{10}$$

Possiamo semplificare l'espressione ottenendo l'espressione seguente:

$$I = \int \mathrm{dm} \begin{bmatrix} (b_2^2 + b_3^2) & -b_1 b_2 & -b_1 b_3 \\ -b_1 b_2 & (b_1^2 + b_3^2) & -b_2 b_3 \\ -b_1 b_3 & -b_2 b_3 & (b_1^2 + b_2^2) \end{bmatrix} \tag{11}$$

dove i termini sulla diagonale sono i momenti di inerzia rispetto agli assi principali del *Body Fixed Frame*, mentre gli altri sono detti **prodotti di inerzia** e si possono vedere come l'accoppiamento incrociato dell'inerzia degli assi. Poichè abbiamo supposto la simmetria del veicolo si possono omettere i prodotti di inerzia e quindi la matrice di inerzia diventa:

$$I = \int \mathrm{dm} \begin{bmatrix} (b_2^2 + b_3^2) & 0 & 0 \\ 0 & (b_1^2 + b_3^2) & 0 \\ 0 & 0 & (b_1^2 + b_2^2) \end{bmatrix} = \begin{bmatrix} I_{b1,b1} & 0 & 0 \\ 0 & I_{b2,b2} & 0 \\ 0 & 0 & I_{b3,b3} \end{bmatrix} \tag{12}$$

Questa proprietà consente di eliminare tutti i termini legati ai prodotti di inerzia dalla scrittura dell'energia cinetica ottendendo:

$$T = \frac{1}{2} I_{b1,b1} (\dot{\phi} - \dot{\psi} \sin\theta)^2 + \frac{1}{2} I_{b2,b2} (\dot{\theta} \cos\phi + \dot{\psi} \sin\phi \cos\theta)^2 + \frac{1}{2} I_{b3,b3} (\dot{\theta} \sin\phi - \dot{\psi} \cos\phi \cos\theta)^2 \tag{13}$$

Ora possiamo passare al calcolo dell'energia potenziale.

L'energia potenziale rispetto al sistema di coordinate generalizzate si ottiene a partire dalla quota verticale nel sistema di coordinate inerziali (z) calcolando il contributo di ogni massa infinitesima:

$$V = g \int z * \mathrm{dm}(\mathbf{r}_i(b_1, b_2, b_3)) = g \int (-\sin\theta b_1 + \sin\phi \cos\theta b_2 + \cos\phi \cos\theta b_3 * \mathrm{dm}(\mathbf{r}_i(b_1, b_2, b_3)) \tag{14}$$

che può essere riscritta, separando le componenti, come:

$$V = \int b_1(-g * \sin\theta)\mathrm{dm}(b_1) + \int b_2(g * \sin\phi \cos\theta)\mathrm{dm}(b_2) + \int b_3(g * \cos\phi \cos\theta)\mathrm{dm}(b_3) \tag{15}$$

Infine considerando L=T-V, posso scrivere le equazioni relative alla dinamica ro-

tazionale specificando le forze e le coordinate generalizzate:

$$\tau_\phi = \frac{\mathrm{d}}{\mathrm{dt}}(\frac{\partial}{\partial\dot{\phi}}L) - \frac{\partial}{\partial\phi}L$$
$$\tau_\theta = \frac{\mathrm{d}}{\mathrm{dt}}(\frac{\partial}{\partial\dot{\theta}}L) - \frac{\partial}{\partial\theta}L$$
$$\tau_\psi = \frac{\mathrm{d}}{\mathrm{dt}}(\frac{\partial}{\partial\dot{\psi}}L) - \frac{\partial}{\partial\psi}L$$

In particolare le forze generalizzate si presentano come momenti meccanici(*torques*) che descrivono l'effetto giroscopico generato dalla rotazione del corpo rigido. Dopo aver sviluppato le derivate e applicato alcune manipolazioni algebriche per semplificare le equazioni, trascurando i termini riguardanti l'energia potenziale (cioè considerando nulla la quota iniziale) e considerando esclusivamente variazioni per piccoli angoli ($\cos x = 1$, $\sin x = 0$) si ottiene:

$$\tau_\phi = I_{b1,b1}\ddot{\phi} = (I_{b2,b2} - I_{b3,b3})\dot{\theta}\dot{\psi}$$
$$\tau_\theta = I_{b2,b2}\ddot{\theta} = (I_{b3,b3} - I_{b1,b1})\dot{\phi}\dot{\psi}$$
$$\tau_\psi = I_{b3,b3}\ddot{\psi} = (I_{b1,b1} - I_{b2,b2})\dot{\phi}\dot{\theta}$$

Queste equazioni descrivono il moto rotazionale del sistema in assenza di momenti meccanici esterni.

1.3 Equazioni della Dinamica

Nel calcolo delle equazioni generali del sistema dovremmo considerare anche i contributi derivati dai momenti meccanici esterni, dovuti per esempio all'azione dei motori. Essendo un sistema fisico con 6 gradi di libertà (3 gradi di libertà traslazionale e 3 gradi di libertà rotazionale) ma essendo dotato di soli quattro motori, un quadricottero è un tipico sistema sotto-attuato (*underactuated system*), ciò comporta che solo alcuni movimenti potranno essere controllati direttamente tramite l'azione dei motori mentre gli altri movimenti saranno controllati agendo su combinazioni di questi ultimi:

1. Spinta verticale (Thrust);
2. Beccheggio (Pitch);
3. Rollio (Roll);
4. Imbardata (Yaw);

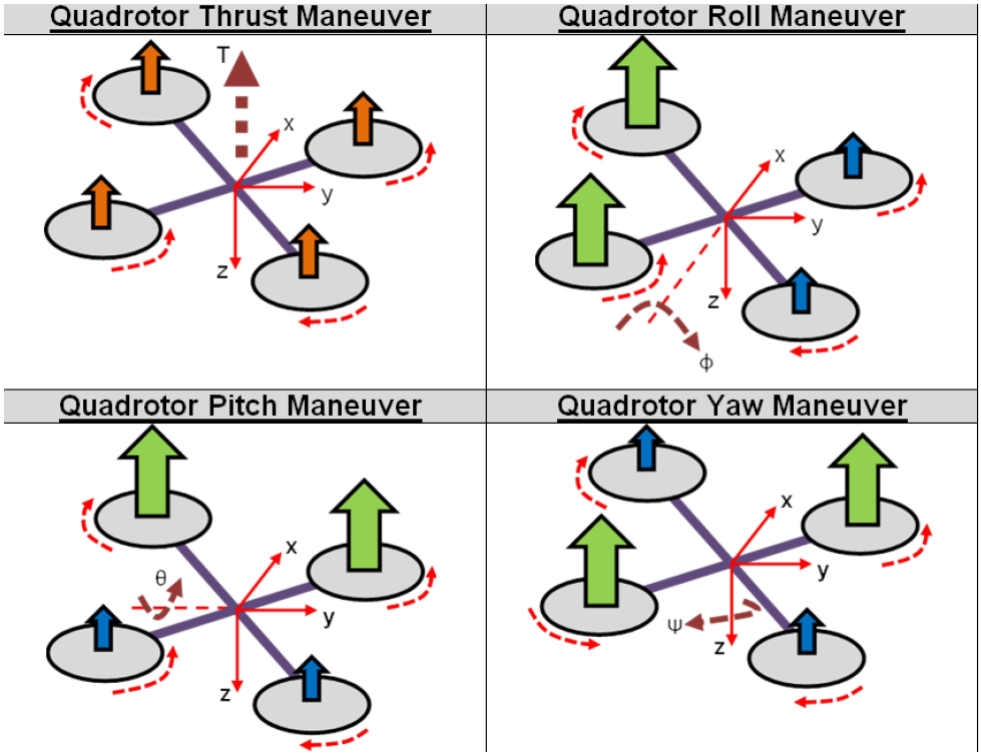
La spinta verticale è dovuta all'azione sincronizzata dei quattro motori che, attraverso la rotazione delle eliche, convogliano verso il basso una massa di aria tale da generare una differenza di pressione sufficiente a sollevare il quadricottero.

Il beccheggio è dovuto al momento generato dalle spinte non uniformi dovute alle diverse velocità di rotazione dei motori anteriori e posteriori. Analogamente il rollio è dovuto al momento generato dalle spinte non uniformi dovute alle diverse velocità di rotazione dei motori posizionati sul lato destro rispetto a quelli posti sul lato sinistro.

L'imbardata si ottiene invece in condizioni di equilibrio per quanto riguarda le spinte dovute all'azione dei motori disposti simmetricamente che annullano reciprocamente i momenti meccanici generati sul veicolo il quale, infatti, non si inclina.

Lo Yaw è dovuto alla massa delle eliche che, una volta messe in rotazione, generano un momento angolare che in presenza di un vincolo, rappresentato dalla struttura dell'aeromobile, tende a far ruotare il sistema. In questo caso l'azione dei motori *front left* e *back right* (cosi come quella di front right e back left) che ruotano nella stesso verso, risulterà accoppiata.

Poichè nel progetto si è utilizzato un modello di quadricottero con configurazione "a croce", ovvero con i motori disposti simmetricamente rispetto all'asse principale del veicolo, l'azione dei motori può essere schematizzata in questo modo:



Alcuni movimenti non sono controllabili direttamente come visto in precedenza, ma lo saranno attraverso azioni accoppiate. Ad esempio, quelli in direzione longitudinale e quelli in direzione laterale saranno ottenuti inclinando preventivamente il drone verso la direzione richiesta.

Dei quattro propulsori due ruotano in verso orario due in verso antiorario: in particolare su ciascun braccio della croce i due motori opposti ruotano equiversi (1-3 e 2-4). Su ciascuna diagonale quindi si viene ad avere una coppia agente sull'asse su cui sono posizionati i motori. Se tutti i motori girano ad un valore di giri identico, non c'è necessità di rotori aggiuntivi per equilibrare le coppie generate, in quanto uguali ed opposte, ottenendo accelerazione angolare nulla intorno agli assi di rotazione.

Ne consegue che modificando opportunamente i giri, quindi le forze, si ottiene uno scompenso di forze e/o momenti in grado di far effettuare al veicolo le manovre desiderate.

Nel dettaglio, la manovra di salita, la più semplice viene eseguita aumentando la spinta di tutti i rotori allo stesso modo. La condizione di volo a punto fisso, detta anche hovering, è possibile considerarla un caso particolare della manovra di salita in cui ciascun motore ha tiro pari ad un quarto del peso del drone raggiungendo il bilancio delle forze tra spinta e peso. Incrementando la potenza di un rotore e decrementando quella dell'opposto si mantengono le coppie complessive globalmente nulle ma si provoca una rotazione attorno dell'asse su cui i motori non cambiano regime: il laterale per il beccheggio (y) e il longitudinale per il rollio (x),ovvero le manovre desiderate. Riguardo al movimento di imbardata, esso è ricavato fornendo potenza ugualmente su una coppia di motori e togliendone allo stesso modo sull'altra.

A questo punto è possibile inserire nelle equazioni che descrivono la dinamica del sistema anche l'azione dei momenti meccanici esterni dovuti all'azione dei motori. Indicando con b il coefficiente di spinta dei motori; con lx, ly le proiezioni dei bracci del minidrone rispettivamente sugli assi x e y; con d il coefficiente di resistenza dell'aria, otteniamo:

$$\tau_\phi = I_{b1,b1}\ddot{\phi} = (I_{b2,b2} - I_{b3,b3})\dot{\theta}\dot{\psi} + bl_y(-\Omega_1^2 - \Omega_2^2 + \Omega_3^2 + \Omega_4^2)$$
$$\tau_\theta = I_{b2,b2}\ddot{\theta} = (I_{b3,b3} - I_{b1,b1})\dot{\phi}\dot{\psi} + bl_x(\Omega_1^2 - \Omega_2^2 - \Omega_3^2 + \Omega_4^2)$$
$$\tau_\psi = I_{b3,b3}\ddot{\psi} = (I_{b1,b1} - I_{b2,b2})\dot{\phi}\dot{\theta} + \mathrm{d}(\Omega_1^2 - \Omega_2^2 + \Omega_3^2 - \Omega_4^2)$$

Il modello fornito sopra dovrebbe essere ulteriormente espanso considerando l'effetto giroscopico risultante dalla rotazione delle eliche in caso di inclinazione del veicolo (variazione angoli ϕ e θ) indicando con J_r il momento di inerzia del rotore, si ha:

$$\tau'_\phi = J_r\dot{\theta}(\Omega_1 - \Omega_2 - \Omega_3 + \Omega_4)$$
$$\tau'_\theta = J_r\dot{\phi}(\Omega_1 - \Omega_2 + \Omega_3 - \Omega_4)$$

Per giungere al modello generale del sistema bisogna aggiungere ai momenti meccanici l'azione delle forze agenti sul sistema. In questo caso l'analisi è molto più semplice poiché l'unica forza agente sul sistema è la forza peso che è facilmente espressa in relazione al sistema di riferimento inerziale: Fp= $[0\ 0\ \mathrm{mg}]^T$.

L'azione dovuta ai motori e la spinta verticale atta a contrastare proprio la forza peso. Esprimendo i contributi di tale spinta in relazione al sistema di riferimento inerziale, cioè utilizzando la matrice R_b^i come matrice del cambiamento di base otteniamo:

$$\begin{aligned}
m\ddot{x} &= (\sin\phi\sin\psi + \cos\phi\sin\theta\cos\psi)b(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \\
m\ddot{y} &= (-\sin\phi\cos\psi + \cos\phi\sin\theta\cos\psi)b(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \\
m\ddot{z} &= mg - (\cos\phi\cos\psi)b(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2)
\end{aligned}$$

Le equazioni generali riguardanti la dinamica del sistema possono infine essere espresse in forma unitaria come:

$$\begin{cases}
F_x = (\sin\phi\sin\psi + \cos\phi\sin\theta\cos\psi)b(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \\
F_y = (-\sin\phi\cos\psi + \cos\phi\sin\theta\cos\psi)b(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \\
F_z = mg - (\cos\phi\cos\psi)b(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \\
\tau_\phi = I_{b1,b1}\ddot{\phi} = (I_{b2,b2} - I_{b3,b3})\dot{\theta}\dot{\psi} + bl_y(-\Omega_1^2 - \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \\
\tau_\theta = I_{b2,b2}\ddot{\theta} = (I_{b3,b3} - I_{b1,b1})\dot{\phi}\dot{\psi} + bl_x(\Omega_1^2 - \Omega_2^2 - \Omega_3^2 + \Omega_4^2) \\
\tau_\psi = I_{b3,b3}\ddot{\psi} = (I_{b1,b1} - I_{b2,b2})\dot{\phi}\dot{\theta} + \mathrm{d}(\Omega_1^2 - \Omega_2^2 + \Omega_3^2 - \Omega_4^2)
\end{cases}$$

Nella modellazione del quadricottero si sono effettuate alcune assunzioni che hanno permesso di semplificare le equazioni ottenute mantenendo contemporaneamente il modello fedele al reale comportamento del drone:

1. Il quadricottero ha una struttura perfettamente simmetrica e la matrice di inerzia è diagonale
2. Il quadricottero, ad eccezione delle eliche, è un corpo rigido
3. La spinta è proporzionale al quadrato della velocità di rotazione
4. Il centro di gravità è fisso e coincide sempre con l'origine del sistema di riferimento solidale con il drone
5. Il sistema è tempo invariante, massa e struttura non variano nel tempo
6. I quattro motori sono identici, quindi analizzarne uno singolarmente non fa perdere di generalità

Una volta raggiunto un modello matematico della dinamica di un quadricottero adeguato ai nostri scopi è necessario valutare tale modello generale in relazione alle caratteristiche specifiche del minidrone Mambo.

In particolare nei prossimi capitoli verranno analizzate le caratteristiche generali di un drone Mambo e soprattutto il modello in Simulink, necessario per una comprensione sufficiente a raggiungere l'obiettivo iniziale

2 Caratteristiche tecniche del minidrone Mambo

Il drone preso in considerazione per questo elaborato è il Parrot Mambo, dell'azienda francese Parrot. La scelta è ricaduta su questo modello in quanto Matlab fornisce un modello Simulink del drone già implementato e scaricabile gratuitamente dal link **https://it.mathworks.com/hardware-support/parrot-minidrones.html**. Su tale modello è possibile compiere ogni azione di modifica, sia a livello di processo sia a livello di controllore.

Il minidrone Mambo è un quadricottero, ovvero un aereomobile dotato di quattro eliche rotanti azionate da motori **coreless DC brusched** (motori che non utilizzano un rotore di materiale ferromagnetico su cui sono collocati gli avvolgimenti rotorici ma dispongono gli avvolgimenti rotorici a formare una struttura cilindrica auto-supportante. Ciò permette di ridurre ampiamente la massa e l'inerzia dei rotori tradizionali garantendo maggiore velocita di risposta ed efficienza)

Il Mambo è dotato di un sistema di sensori integrati comprendente:

1. **Sensore ad ultrasuoni**, montato sul lato inferiore, utile a misurare la quota verticale mediante la determinazione del tempo di propagazione del segnale emesso dal drone e riflesso dal suolo;
2. **Fotocamera** montata anch'essa sul lato inferiore, da 60 FPS (frame per secondo) che fornisce i dati necessari al software integrato di elaborazione delle immagini 'Optical Flow'. Questa tecnologia, sfruttando la posizione relativa degli oggetti tra un fotogramma e l'altro è in grado di stimare il movimento orizzontale del drone e quindi la sua velocità;
3. **Sensore di pressione** utile per la misurazione indiretta dell'altitudine
4. **Unità di misura inerziale (IMU)** utile per valutare la velocità, l'inclinazione e il contatto con gli ostacoli (accelerometro a 3 assi e giroscopio a 3 assi)

(a) Mambo - lato inferiore (b) Mambo - lato superiore

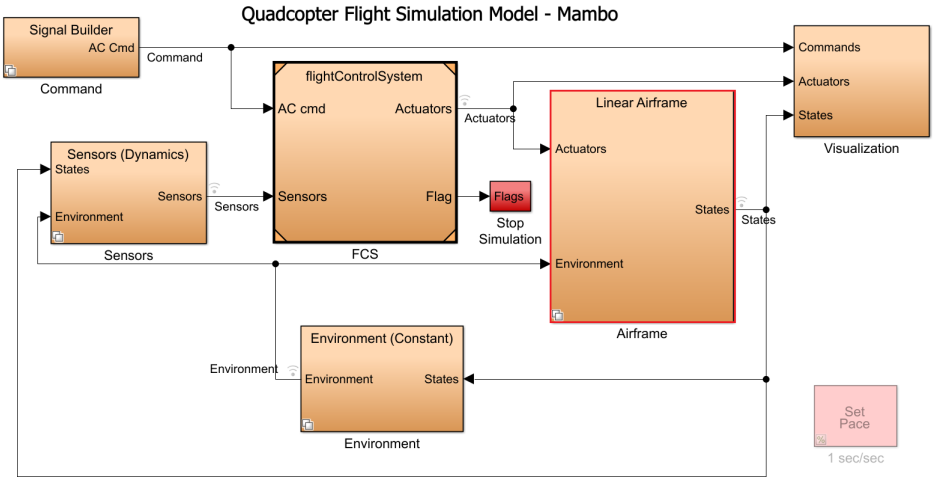
Infine possiamo dire che il minidrone Mambo è dotato di connettività wireless Bluetooth Low Energy attraverso la quale è possibile interfacciarsi al sistema di controllo del drone.

3 Analisi del Modello in Simulink

Simulink è un software per la modellazione, simulazione e analisi di sistemi dinamici, sviluppato dalla compagnia statunitense Mathworks. Questo software è strettamente integrato con Matlab.

Il modello Simulink del minidrone Mambo, fornito all'interno del Simulink *Support Package for Parrot Minidrones*, si compone di diversi blocchi, analizzabili separatamente, cioè:

- **Signal Builder** : generatore di segnali di riferimento;
- **Flight Control System** : controllore del processo;
- **Aiframe** : Modello del processo;
- **Environment** : blocco che contiene le variabili di ambiente;
- **Sensors** : blocco contenente le costanti di trasduzione e le relazioni tra i diversi sensori presenti a bordo;
- **Visualization** : blocco ausiliario che consente la visualizzazione dei valori delle variabili di stato e dei tracciati dei segnali di controllo.



Oltre al modello Simulink il pacchetto di supporto contiene anche un'interfaccia di tipo testuale all'interno della quale è possibile settare le diverse variabili di ambiente oppure passare dal modello non-lineare del sistema a quello lineare.

La possibilità di disporre di un modello lineare del sistema è molto importante in fase di progettazione poichè consente di determinare piu facilmente la struttura e il settaggio del controllore. D'altra parte il modello non-lineare garantisce una maggiore fedeltà rispetto al caso reale in quanto si basa sul modello matematico sviluppato nella prima sezione, il quale contiene diversi contributi non lineari.

3.1 Processo

All'interno del blocco Aiframe è presente il modello del processo, il quale, a seconda del nostro utilizzo o della nostra analisi, può essere settato a modello non lineare o modello lineare. La scelta del modello viene eseguita tramite la modifica di una variabile, già presente all'interno del progetto, chiamata **VSS VEHICLE**:

- VSS VEHICLE=0 il modello è settato a Linear Aiframe
- VSS VEHICLE=1 il modello è settato a NonLinear Aiframe

La nostra variabile può essere modificata tramite la window di Matlab oppure direttamente all'interno del file .m presente nel progetto.

Il nostro primo obiettivo è analizzare il comportamento del modello non lineare, pertanto settiamo VSS VEHICLE=1.

Nel momento in cui facciamo questo, e aperto il blocco corrispondente al *Non-Linear Aiframe* otteniamo questi tre blocchi seguenti:

- **AC Model** : il blocco che contiene la parte della dinamica, i suoi ingressi sono le velocità angolari w_b e le velocità lineari v_b espresse rispetto al sistema solidale con il body del drone ed uscenti dal blocco 6DOF. In uscita si hanno forze e momenti su tutti e tre gli assi;

- **6DOF Quaternion** : blocco fornito da Matlab, prende in ingresso i momenti e le forze. Determina in uscita tutte le grandezze di velocità, acccelerazioni e angoli;
- **Bus Setup** : compone un vettore di bus. Contiene tutte le velocità in entrambi i sistemi di riferimento oltre a posizione espressa in ned,North.East-Down, accellerazioni e angoli di rotazione.

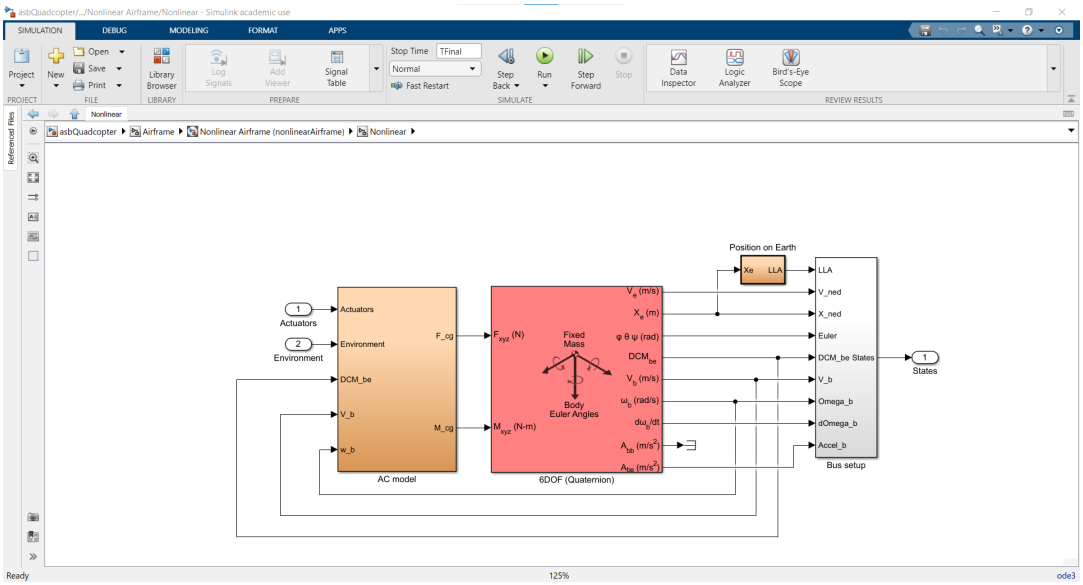


Figure 3: Modello Non Lineare

In particolare, possiamo notare come le forze e i momenti meccanici, calcolati rispetto al centro di gravità del veicolo, siano generati dal blocco AC model. Questo blocco ha come variabili di ingresso:

- L'azione di spinta degli attuatori *Actuators*;
- L'azione delle variabili di ambiente *Environment*;
- I valori della matrice DCM_{be} (Direct Cosine Matrix) necessari per il passaggio dal sistema di riferimento del corpo rigido a quello inerziale;
- La velocità lineare, misurata nel sistema di riferimento del corpo rigido o Body Fixed Frame (v_b);
- La velocità angolare, misurata nel sistema di riferimento del corpo rigido o Body Fixed Frame (w_b).

Le formule che descrivono l'azione del blocco riprendono e specificano quelle del modello matematico generale. Per le forze avremo che:

$$m\dot{v_b} = -mw_b \times v_b + mgR_i^b e_3 + \sum_{N,E,S,W} t_i \tag{16}$$

Mentre per quanto riguarda i momenti meccanici:

$$I\dot{w_b} = -w_b \times Iw_b + \sum_{N,E,S,W} (m_i + q_i) \tag{17}$$

dove le lettere N,E,S,W indicano il posizionamento dei quattro motori.

All'interno del blocco AC Model sono presenti determinati blocchi adibiti proprio al calcolo delle forze e dei momenti esterni agenti sul drone, e tali blocchi sono ***Motor Forces and Torques*** e ***Applied Force Calculation*** i quali sono visibili anche nella seguente immagine:

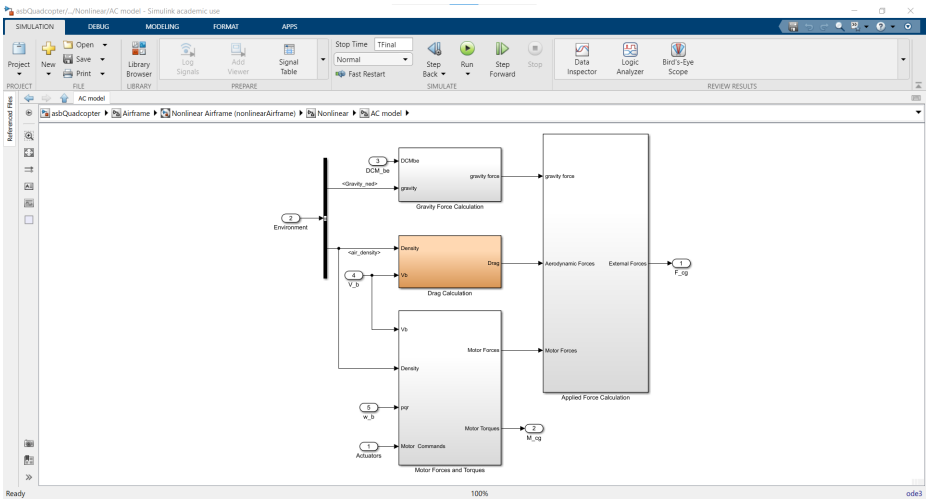


Figure 4: AC Model

Oltre ai due blocchi nominati in precedenza, sono presenti altri due blocchi importanti ovvero:

- **Gravity Force Calculation** : calcola la forza di gravità rispetto al sistema solidale con il drone. Ha come ingresso la matrice di rotazione DCM_{be} e la costante di gravità.

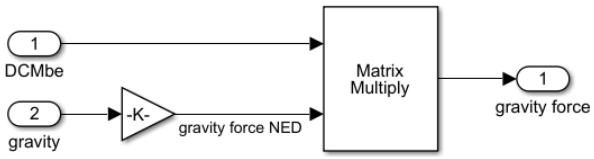


Figure 5: Gravity Force Calculation

- **Drag Calculation** : Calcola la forza di resistenza che l'aria imprime al drone in movimento. Tale forza è dovuta allo spostamento del drone ed è esprimibile come:

$$D = \frac{1}{2}\rho V^2 S C_D \tag{18}$$

dove:

  - ρ è la densità dell'aria
  - V è la velocità di traslazione del drone
  - S è la superficie che impatta l'aria durante la traslazione
  - C_D è il coefficiente di drag

Il vettore S che entra nel blocco moltiplicatore, presente nella figura 4, contiene la quantità di superficie che va ad impattare con il fluido (aria).

Le prime due componenti di tale vettore (x,y) sono $\frac{1}{4}$ della terza componente del medesimo vettore S. Ciò accade in quanto si considera il drone come un parallelepipedo tale per cui vi è la seguente relazione tra le superfici:

$$SUP_{superiore} = SUP_{inferiore} = diamiter = \sum_{i=1}^{4} SUP_{lateralei} \tag{19}$$

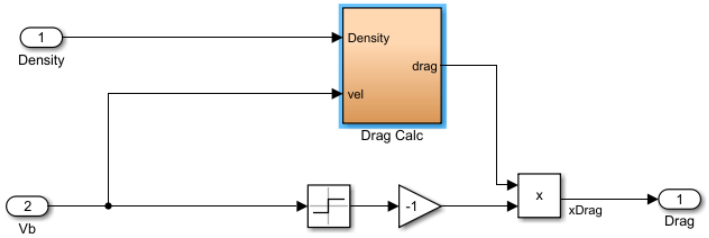


Figure 6: Drag Calculation 1

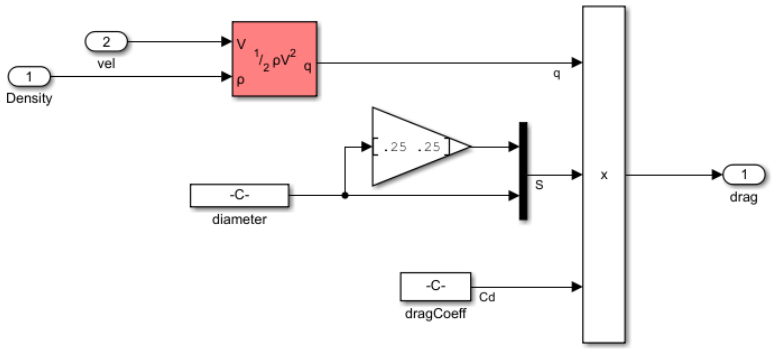


Figure 7: Drag Calculation 2

Nelle immagini precedenti si può osservare bene l'applicazione al nostro problema della formula (18) per il calcolo della forza di drag.

Una volta analizzati i blocchi meno significativi dell'AC model, è possibile dare una breve analisi dei due blocchi principali, già nominati in precedenza, cioè ***Motor Forces and Torques*** e ***Applied Force Calculation***:

- **Applied Force Calculation** : Questo blocco contiene un blocco Simulink che compone un vettore avente come componenti: forze di drag, forza di gravità e le forze generate dai motori.

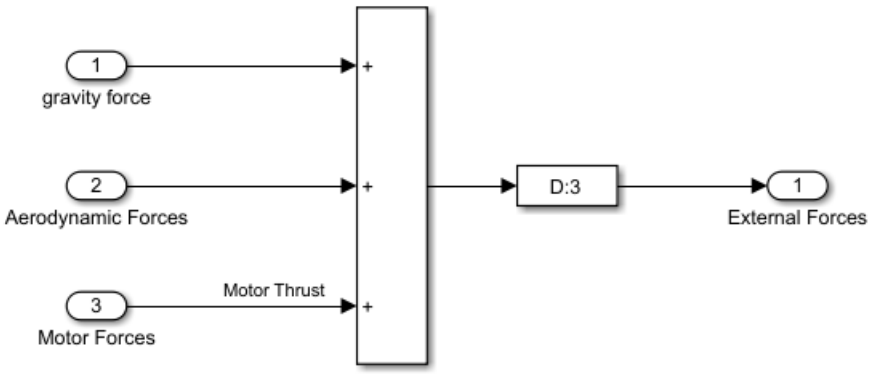


Figure 8: Applied Force Calculation

- **Motor Forces and Torques** : Esso contiene al proprio interno 3 blocchi:
  1. Blocco centrale il quale ha come ingressi le velocità lineari e angolari rispetto al body frame, il vettore Motor Commands il quale contiene lo sforzo di controllo che arriva dal controllore ed infine il vettore D che descrive il posizionamento rispetto al centro di massa dei rotori.
  2. Blocco MotorToW il quale trasforma lo sforzo di controllo in uno sforzo interpretabile dai motori. (Tale conversione avviene attraverso blocchi di Gain e saturazioni).
  3. Blocco ForEach che ci permette di analizzare ogni attuatore in particolari ingressi e uscite per il calcolo di momenti e forze.

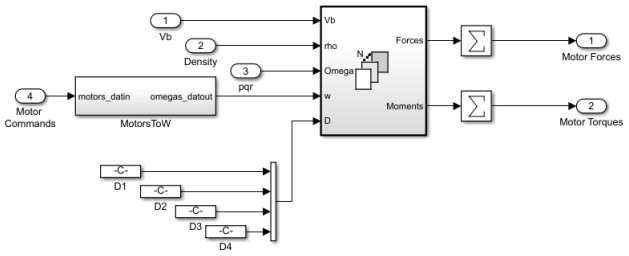


Figure 9: Motor Forces and Torques

Figure 10: MotorToW

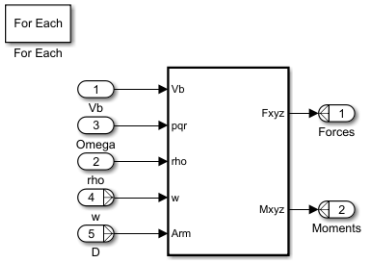


Figure 11: Dinamica per ogni motore

Abbiamo già detto che il blocco ***Motor Forces and Torques*** è adibito al calcolo delle forze e dei momenti esterni, e ora andiamo a vedere i blocchi presenti che vengono utilizzati per i vari calcoli matematici. Per osservare tali blocchi è necessario, all'interno del progetto in Simulink, seguire il seguente percorso: AC Model/Motor Forces and Torques/For Each Subsystems/Rotor Dinamics. Quello che otteniamo è riportato di seguito:

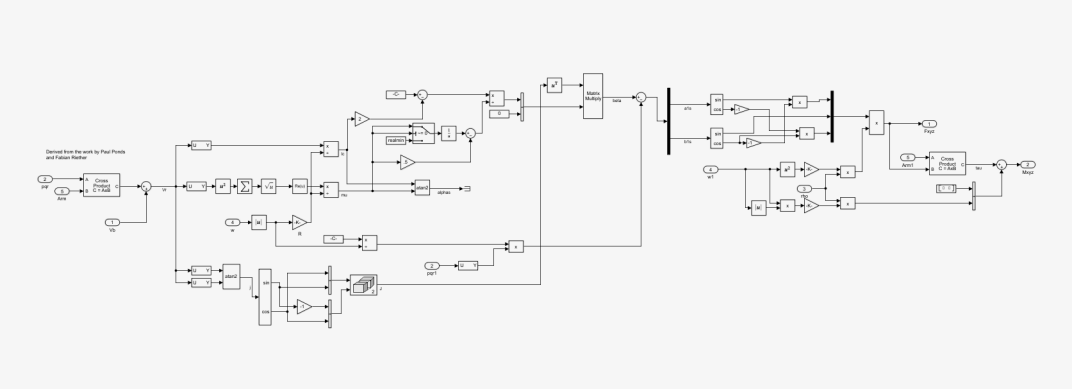


Figure 12: Modello di calcolo forze e momenti

In uscita a tale modello abbiamo forze e momenti meccanici che andranno direttamente in ingresso al blocco ***6DOF Euler Angles*** che si occupa di modellare la dinamica del sistema, fornendo le variabili di stato ai blocchi che ne hanno bisogno.

In particolare avremo le seguenti variabili (visibili anche nella figura 3):

- V_e[m/s] : Velocità traslazionale nel sistema di riferimento inerziale (*Earth Fixed Frame*);
- X_e [m] : Posizione rispetto all'*Earth Fixed Frame*;
- (ϕ, θ, ψ) : Angoli di Eulero rispetto al *Body Frame*;
- DCM_{be} : Coefficienti della matrice Direct Cosine Matrix;
- v_b [m/s] : Velocità traslazionale rispetto al *Body Frame*;
- w_b [rad/s] : Velocità angolare rispetto al *Body Frame*;
- $\frac{dwb}{dt}$ [rad/s^2] : Derivata velocità angolare rispetto al *Body Frame*;
- A_{be} [m/s^2] : Accelerazione lineare corpo rigido rispetto all'*Earth Fixed Frame*;
- LLA : latitudine, longitudine e altitudine calcolate in coordinate geodetiche;

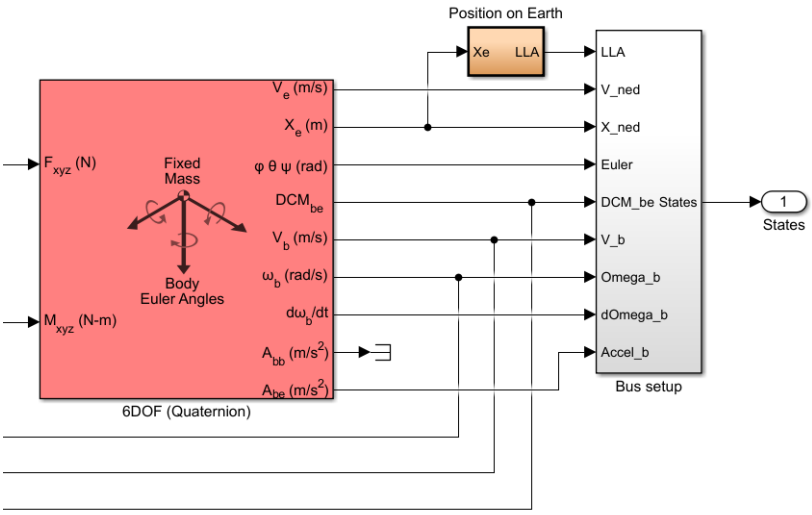


Figure 13: Variabili di stato

3.2 Generatori di segnali

Il blocco che ci permette di scegliere quale segnale di riferimento utilizzare in ingresso al modello, è il blocco ***Command***. In particolare, è possibile scegliere il segnale di riferimento tramite una variabile chiamata *VSScommand* e a seconda del valore che assume otteniamo:

- Il blocco **Signal Builder**, cioè il costruttore di segnali di default (VSScommand=0);
- Il blocco **Joystick** che permette di gestire manualmente il comportamento del veicolo (VSScommand = 1);
- Il blocco **Data** che consente di selezionare i riferimenti tra quelli presenti in un file .mat (VSScommand = 2);
- Il blocco **Spreadsheet Data** che consente di selezionare i valori dei riferimenti da un foglio di calcolo (VSScommand = 3);

Selezionando il Signal Builder è possibile fornire al sistema segnali di riferimento di posizione (X,Y,Z) che riferimenti di orientamento (Pitch,Roll,Yaw).

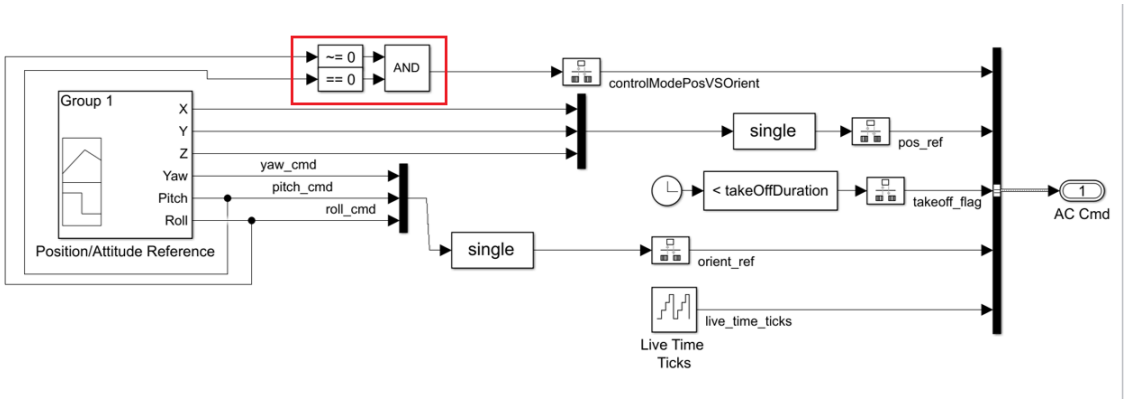


Figure 14: Blocco Signal Builder

Particolare attenzione, come vedremo anche in seguito, va verso la porta logica AND che mi permette di passare dalla modalità di controllo di posizione (se Pitch e Roll sono uguali a 0) altrimenti rimane in modalità di controllo di orientamento.

All'interno del blocco ***Position/Attitude reference*** è possibile regolare l'andamento delle variabili di riferimento relative alla posizione: X, Y, Z (in quest'ultimo caso il riferimento è negativo poiché l'asse Z nel modello è orientato verso il basso) e all'orientamento in aria del veicolo: angoli di Roll, Pitch, Yaw.

L'interfaccia grafica consente di modificare anche l'ampiezza dell'intervallo temporale da considerare e, ovviamente, l'andamento delle diverse variabili nel corso del tempo.

Figure 15: Segnali di Riferimento

3.3 Blocco di Controllo

Il blocco che si occupa del controllo vero e proprio del sistema e sul quale si è lavorato per il nuovo controllore è il ***Flight Control System***. Nel seguente blocco troviamo tutta la parte che si occupa del controllo del drone cioè la parte che in fase finale del progetto, verrà compilata e trasformata in codice binario dal calcolatore e implementata direttamente nel drone fisico.

Esso in ingresso riceve i segnali di riferimento provenienti dal *Signal Builder* e i valori delle variabili di stato provenienti dai sensori in retroazione. D'altro canto esso da in uscita i segnali di riferimento degli attuatori che effettueranno il controllo sui motori del modello non lineare e un'uscita di sicurezza (data da *Flags*), che fornisce un segnale di controllo che interrompe la simulazione.

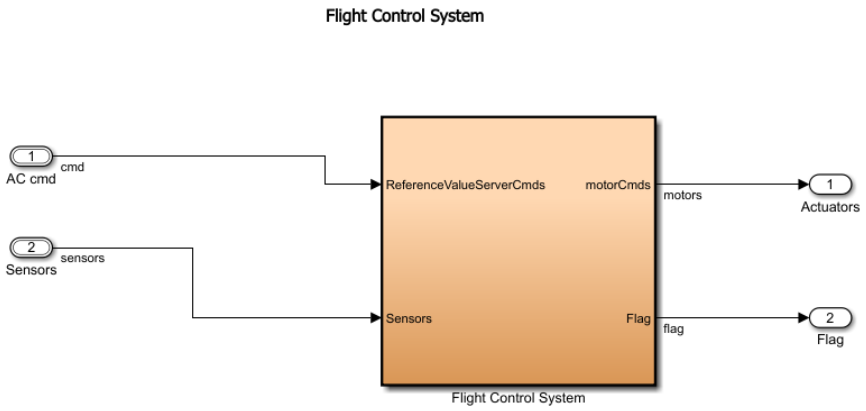
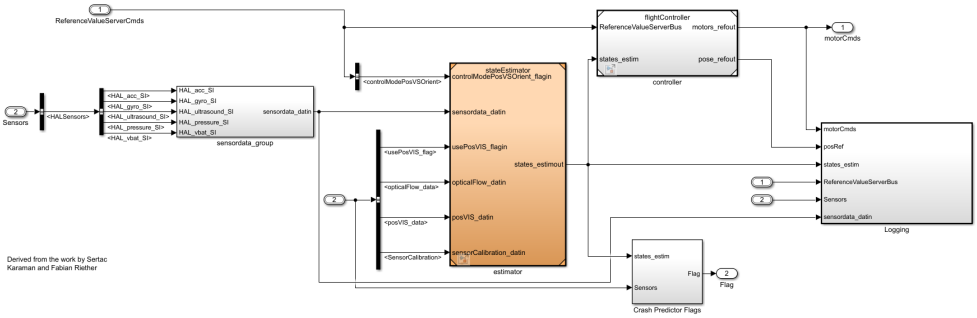


Figure 16: Flight Control System

Il blocco *FCS* si suddivide in 3 blocchi ulteriori:

1. Il blocco **State Estimator** riceve ed elabora i segnali provenienti dai diversi sensori presenti sul veicolo e fornisce tali informazioni al blocco Flight Controller;
2. Il blocco **Flight Controller** effettua l'azione di controllo e fornisce in uscita i segnali di controllo diretti ai motori;
3. Il blocco **Crush Predictor Flag** elabora le informazioni fornite direttamente dai sensori e dal blocco State Estimator , e determina confrontandole coi parametri base se ci sono eventuali problemi nella sessione di volo.



Inoltre possiamo dire che all'interno del blocco *FCS* sono presenti due sotto-sistemi ausiliari molto importanti ovvero:

1. ***Logging*** : fornisce il valore delle variabili in ingresso e uscita dal blocco FCS. Crea dei file di log durante le simulazioni virtuali o reali sul drone. Questi file sono utili per verificare il corretto andamento desiderato post volo e per determinare eventuali cause di arresto.
2. ***Sensordata group*** : svolge una pre-elaborazione (ridenominazione delle variabili) sui dati forniti dai vari sensori montati sul veicolo (ad esclusione di quelli ottici);

Una volta fatta una panoramica di ciò che è presente all'interno del blocco di controllo, è il momento di analizzare nello specifico i due blocchi fondamentali ovvero *State Estimator* e *Flight Controller*. Iniziamo con il primo.

Il blocco ***State Estimator*** è composto da un blocco principale (*Sensor Preprocessing*) che effettua il pre-processamento dei dati provenienti dall'IMU e dall'unità ottica. Insieme a questo blocco, sono presenti ulteriori 3 blocchi secondari:

- **Complementary Filter** elabora i dati provenienti dal sistema di pre-processamento e fornisce una stima dell'orientamento del sistema: in particolare calcola gli angoli di roll,pitch e yaw e le variazioni angolari (p; q; r) rispetto al *Earth Fixed Frame*;
- **Estimator Altitude** elabora i dati provenienti dal sensore di pressione e dal sonar e, incrociandolo con i dati riguardanti l'orientamento del sistema, determina la quota e la velocità del veicolo;
- **Estimator X-Y position** elabora i dati provenienti dal sensore ottico per determinare posizione e velocità longitudinale e latitudinale del drone;

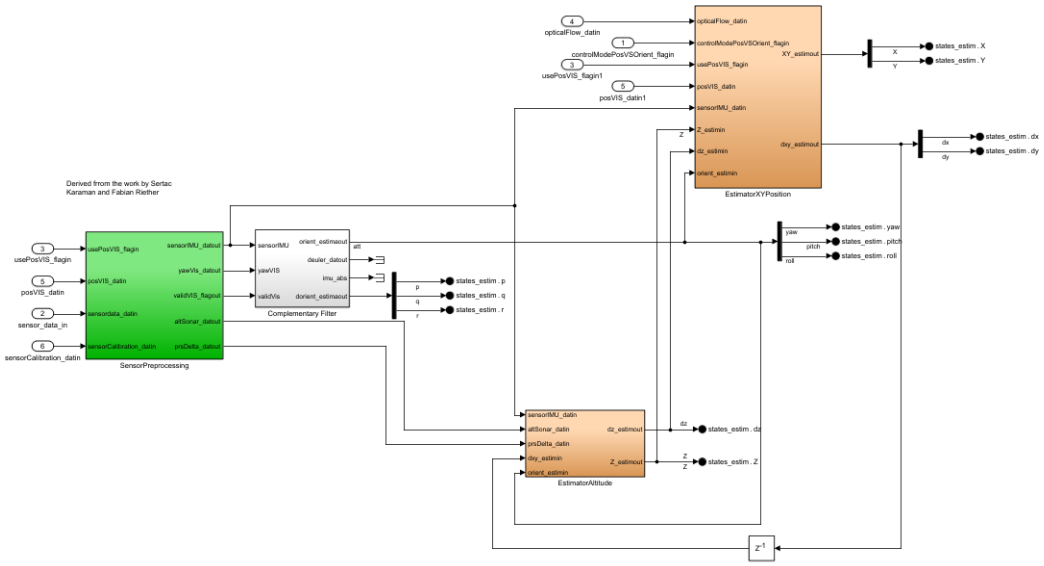


Figure 17: Blocco State Estimator

Il blocco ***Flight Controller*** è costituito da quattro sotto-sistemi principali e tre blocchi ausiliari; i primi si occupano di generare i segnali che andranno a definire il comportamento e quindi il controllo del veicolo. I Sotto-blocchi ausiliari invece sono lo *Switch Ref Att* , che permette di passare dal segnale di riferimento rispetto alla posizione al segnale di riferimento rispetto all'orientamento e altri 2 blocchi che vedremo successivamente il *Control Mixer* e *Thrust to Motor Command* che forniscono alcune costanti relative agli attuatori che ci serviranno in fase di progettazione.

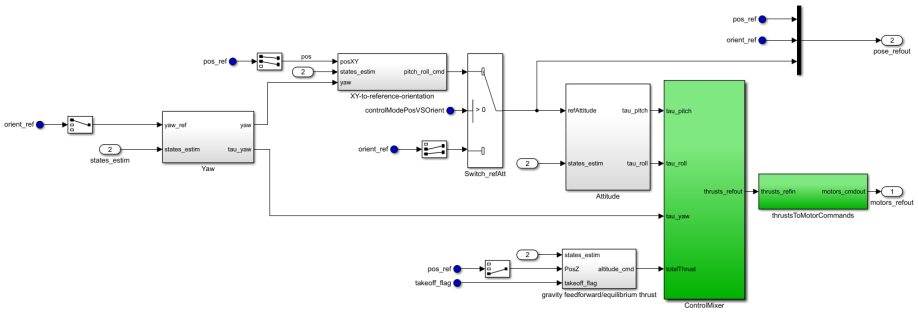


Figure 18: Blocco Flight Controller

I quattro sotto-sistemi principali si occupano direttamente della generazione

dei segnali di controllo che definiranno il comportamento del veicolo:

1. ***XY to reference orientation*** riceve in ingresso il riferimento alla posizione del veicolo (longitudinale e latitudinale), l'angolo di yaw e il valore corrente della posizione e della velocità (longitudinali e latitudinali). In uscita elabora un segnale di controllo indicante il valore che devono assumere gli angoli di pitch e roll per mantenere la posizione di riferimento;
2. ***Attitude*** riceve in ingresso il valore corrente degli angoli di pitch e roll, e delle relative velocità angolari, e, a seconda del valore dello switch, il segnale di riferimento relativo alla posizione oppure quello relativo all'orientamento. In uscita fornisce i segnali di controllo per gli angoli di pitch e roll;
3. ***Yaw*** riceve in ingresso il riferimento relativo all'angolo di Yaw, il valore corrente di tale angolo e la relativa velocità angolare. In uscita fornisce il segnale di controllo per l'angolo di Yaw;
4. ***Gravity Feedrorward/Equilibrium Thrust*** riceve in ingresso il segnale di riferimento relativo alla quota (Z), il valore corrente di tale variabile, la velocità verticale del veicolo e il segnale Take-Off Flag che regola l'azione di decollo. In uscita elabora il segnale di controllo relativo all'altitudine del veicolo.

I sistemi sopra elencati sono quelli che si occupano del controllo dell'angolo di beccheggio e rollio (pitch and roll) quindi sono i sistemi che andremo a modificare per effettuare il controllo del nostro drone. Di seguito analizzeremo ognuno di questi e vedremo qual è il loro funzionamento e cosa controllano.

3.3.1 Sistemi di controllo per gli angoli di pitch e roll

Il blocco *XY to reference Orientation* si compone di una parte superiore in cui, utilizzando l'angolo di yaw, si calcolano le componenti X, Y rispetto al sistema di riferimento inerziale dovute alla rotazione del veicolo sull'asse Z (nei fatti si applicano le prime due righe della matrice R (Z; ϕ)).

Nella parte inferiore i valori così ottenuti vengono sommati agli errori sulla posizione longitudinale e latitudinale (ottenuti confrontandoli con quelli stimati di X e Y con i loro relativi riferimenti) e poi vengono moltiplicati con le opportune costanti. Nel complesso sommando i due rami si va a formare un controllore PD (Proporzionale Derivativo).

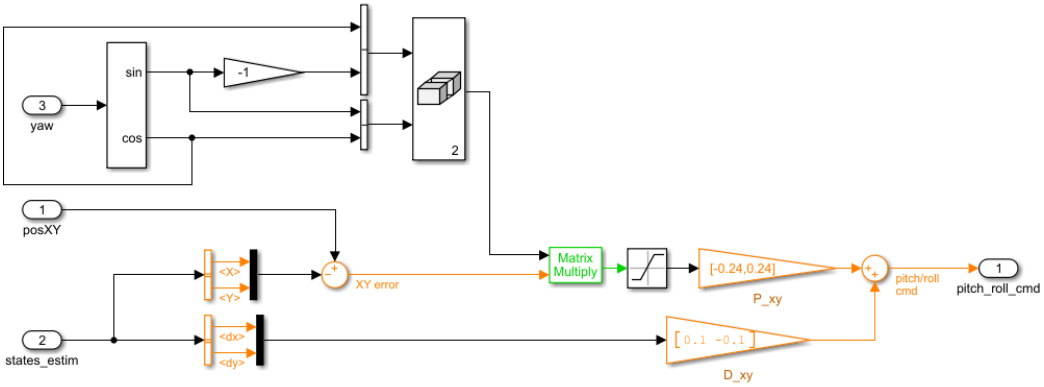


Figure 19: Blocco XY to reference Orientation

Il blocco *Attitude* riceve in ingresso, a seconda di come è impostato lo switch precedente a esso, o il segnale dal *Signal Builder*, allora in quel caso abbiamo un controllo per orientamento, o il segnale dall' *XY orientation* allora abbiamo un controllo per posizione. Per questo motivo all'interno del blocco attitude il segnale è chiamato **Ref Attitude**.

All'interno di questo blocco entra un doppio segnale il quale corrisponde agli angoli di pitch e roll, che viene confrontato con valori derivanti dallo *State Estimator*, in modo da ricavare l'errore rispetto al riferimento. Questi valori alimenteranno un blocco proporzionale e un blocco integratore; a questi valori vengono sommati le velocità p e q che prima vengono moltiplicate per opportune costanti poi aggiunte tramite un nodo sommatore. Si ottiene un controllore PID (Proporzionale Integrativo Derivativo).

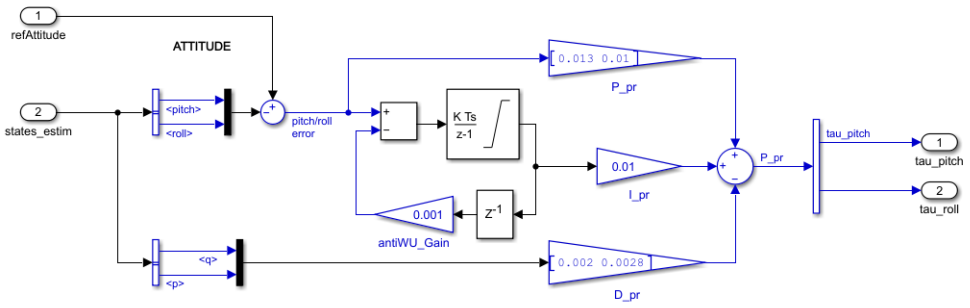


Figure 20: Blocco Attitude

3.3.2 Sistema di controllo per l'angolo di Yaw

Il blocco *Yaw* è un controllore *Proporzionale Integrativo* con una struttura del tipo *Derivative Output of Controller* dove la parte derivativa non viene eseguita sull'errore ma sull'uscita, in questo caso r. Inoltre, la parte derivativa entra in contro reazione.

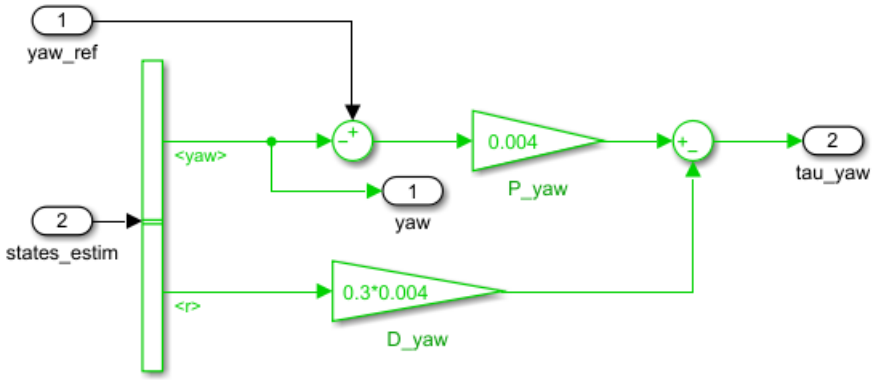


Figure 21: Blocco Yaw

3.3.3 Sistema di controllo Altitudine

Il blocco *Gravity Feedwar/Equilibrium Thrust* contiene il controllore che agisce sulla z. Qui troviamo un controllore di tipo **PD** (*Proporzionale – Derivativo*). Si nota la presenza di uno switch il quale ha il compito, durante il decollo, di fornire una potenza maggiore agli attuatori e di isolare il controllore. Finito il decollo, il flag *takeoffflag* torna ad essere NULL e dunque lo switch seleziona il controllore. Utilizza il parametro **TakeOffGain**.

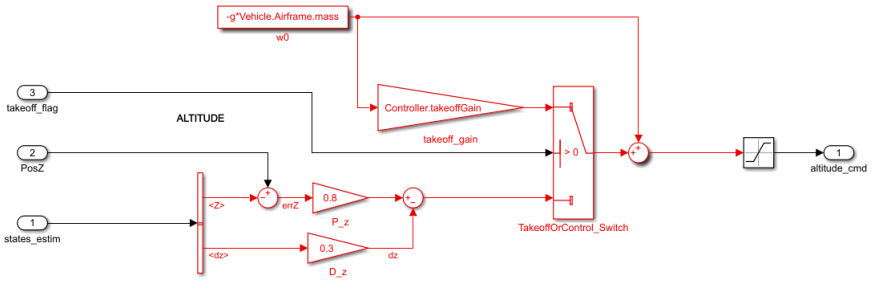


Figure 22: Blocco Gravity Feedwar/Equilibrium Thrust

3.4 Sensori

Il Blocco ***Sensors*** si occupa della gestione e dell'implementazione dei vari sensori di cui è provvisto il drone, in particolare i sensori del drone sono due ovvero il sensore posizionato sulla telecamera posta nella zona inferiore al drone e l'IMU (Pressure).

- ***Camera*** : Sottosistema che si occupa di analizzare le immagini provenienti dalla fotocamera e in base, in realtà non è compito di Simulink gestirlo poiché viene gestito direttamente dal firmware fornito da Parrot.
- ***IMU Pressure*** : Sottosistema che si occupa della gestione di tutti i dati provenienti dagli altri sensori (Accelerometri, Giroscopi, Sensore di Pressione e Sonar).

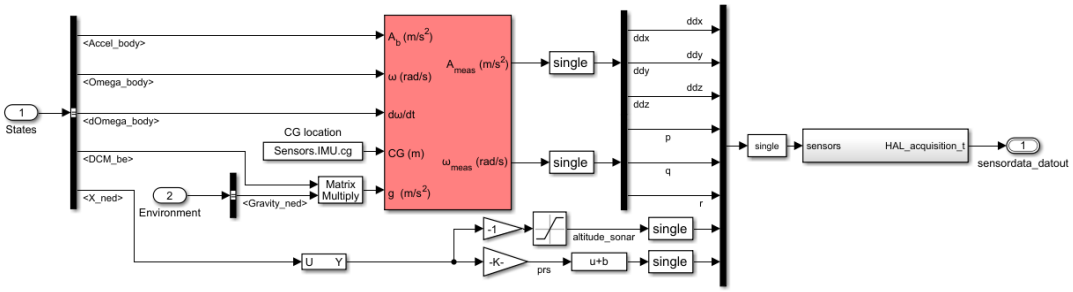


Figure 23: Blocco IMU Pressure

L'*IMU Pressure*, riportato nell'immagine soprastante, si occupa di trasformare i dati dei sensori forniti dal *Body Frame* in dati utili per l' *Earth Fixed Frame* .

4 Controllore e Tecniche di controllo

In questa sezione andrò ad analizzare due tecniche di controllo fondamentali per l'obiettivo finale ovvero il *PID Controller* e la tecnica con il *Luogo delle Radici*. Queste due tecniche sono fondamentali per il nostro caso, perchè il controllore di default presente è realizzato tramite PID e il nuovo controllore da sostituirgli deve essere realizzato tramite il Luogo delle Radici.

4.1 PID Controller

Il ***controllo PID*** (Proporzionale Integrativo Derivativo) è uno dei più utilizzati nel settore del controllo industriale poiché facilmente realizzabile e molto affidabile.

È un sistema in retroazione negativa che molto spesso viene utilizzato nella versione PI (cioè una versione in cui il termine derivativo non viene CIRO utilizzato). Grazie a un input che determina il valore iniziale, è in grado di reagire a un eventuale errore, positivo o negativo, tendendo verso il valore 0. La reazione all'errore può essere regolata e questo rende il sistema molto versatile.

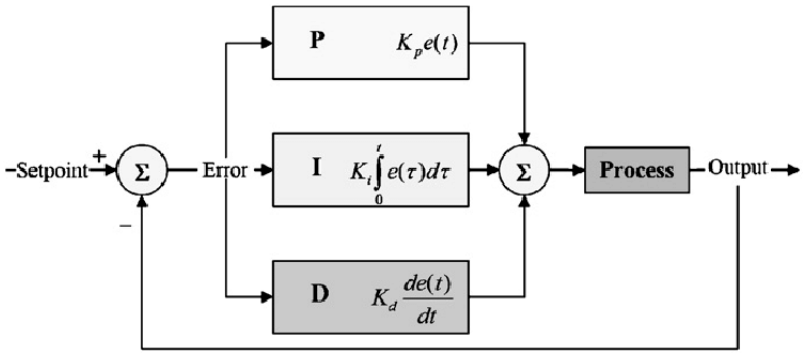


Figure 24: PID Controller

Il controllore acquisisce in ingresso un valore dal processo e lo confronta con un valore di riferimento. La differenza, il cosiddetto segnale d'errore, viene quindi usato per determinare la variabile d'uscita del controllore, che è la variabile manipolabile dal processo.

Il PID regola l'uscita in base a 3 componenti:

1. Il valore del segnale d'errore (Azione Proporzionale);
2. Il valore passato del segnale d'errore (Azione Integrale);
3. Quanto velocemente il segnale d'errore varia (Azione Derivativa);

I Controllori di questo tipo sono molto semplici da gestire e da settare, utilizzando le tecniche di Ziegler-Nichols, questi risultano buoni controllori per la stabilizzazione della maggior parte dei processi.

Hanno qualche limitazione sul fatto che non posso essere utilizzati se abbiamo parametri variabili, non sono molto stabili in alcuni casi a causa dell'azione integrativa che genera l'Integral Windup, inoltre le tecniche di Ziegler-Nichols per la progettazione non possono essere utilizzate in ogni condizione ma è necessario avere condizioni ben precise per l'uso.

4.1.1 Azione di controllo di un PID

Le tre azioni di un PID vengono calcolate separatamente e semplicemente sommate algebricamente.

$$\mu = \mu_p + \mu_i + \mu_d \tag{20}$$

L'**azione proporzionale** è ottenuta moltiplicando il segnale d'errore e con un'opportuna costante.

$$\mu_p = K_p * e \tag{21}$$

È possibile regolare un processo con simile controllore, anche processi instabili, tuttavia non è possibile richiedere che il segnale di errore "e" converga a 0 : questo perché è un'azione di controllo che è possibile solo se "e" è diverso da 0.

L'**azione integrale** è proporzionale all'integrale nel tempo del segnale d' errore e, moltiplicato per una costante.

$$\mu_i = K_i * \int e(t)\mathrm{dt} \tag{22}$$

Questa definizione dell'azione integrale fa sì che il controllore abbia memoria dei valori passati del segnale d'errore; in particolare, il valore dell'azione integrale non è necessariamente nullo se è nullo il segnale d'errore. Questa proprietà dà al PID la capacità di portare il processo esattamente al punto di riferimento richiesto, dove la sola azione proporzionale risulterebbe nulla.

L'**azione derivativa** serve a migliorare le prestazioni di controllo:

$$\mu_d = K_d * \frac{de}{dt} \tag{23}$$

L'idea è compensare rapidamente le variazioni del segnale di errore: se vediamo che "e" sta aumentando, l'azione derivativa cerca di compensare questa deviazione in ragione della sua velocità di cambiamento, senza aspettare che l'errore diventi significativo (azione proporzionale) o che persista per un certo tempo (azione integrale). L'azione derivativa è spesso tralasciata nelle implementazioni dei PID perché li rende troppo sensibili.

4.2 Tecnica del Luogo delle Radici

In analisi complessa il luogo delle radici è il luogo geometrico delle radici di una funzione complessa descritto al variare di un suo parametro reale (K'), rappresentato sul piano di Gauss.

Si parte dalla funzione di trasferimento del nostro sistema in catena aperta ovvero dalla F(s):

$$F(s) = K\frac{\prod_{i=1}^{m}(s - z_i)}{\prod_{i=1}^{n}(s - p_i)} \tag{24}$$

dove K è il guadagno,m zeri, ed n poli e se n>m la funzione F(s) è strettamente propria.

Dall'equazione di F(s) è possibile trovare le radici dell'equazione algebrica ponendola uguale a zero:

$$\prod_{i=1}^{n}(s - p_i) + K\prod_{i=1}^{m}(s - z_i) \tag{25}$$

Le radici di questa equazione coincidono con i poli della funzione di trasferimento a ciclo chiuso del sistema retroazionato:

$$W(s) = \frac{F(s)}{1 + F(s)} = K\frac{\prod_{i=1}^{m}(s - z_i)}{\prod_{i=1}^{n}(s - p_i) + K\prod_{i=1}^{m}(s - z_i)} \tag{26}$$

dove il denominatore è chiamato f(s,k) e sarà poi importante quando si parlerà delle regole per il tracciamento.

L'utilizzo del luogo delle radici è presente, nello studio di un sistema di controllo a controreazione, per correlare alcune importanti caratteristiche della funzione di trasferimento a ciclo chiuso ai valori dei parametri che caratterizzano la funzione di trasferimento in catena diretta. Il procedimento si basa nell'individuazione, esatta o qualitativa a seconda della complessità del caso, del luogo geometrico dei punti del piano complesso al variare del parametro moltiplicativo K.

4.2.1 Regole per il tracciamento

Il metodo del luogo delle radici è un metodo grafico che ci permette quindi di individuare quali sono i poli o gli zeri che non soddisfano le specifiche richieste, e attraverso opportune modifiche permettere a questi zeri o poli non accettabili di entrare in condizioni di essere accettati.

Riprendiamo la f(s,k), già nominata in precedenza, e scriviamola nel seguente modo:

$$\prod_{i=1}^{n}(s - p_i) = -K\prod_{i=1}^{m}(s - z_i) \tag{27}$$

Per poi sostituire a quest'ultima la coppia di uguaglianza tra grandezze reali:

$$\prod_{i=1}^{n}|s - p_i| = K\prod_{i=1}^{m}|s - z_i| \tag{28}$$

$$\sum_{i=1}^{n} < (s - p_i) = \pi + < K + \sum_{i=1}^{m} < (s - z_i) + 2h\pi \tag{29}$$

La prima di queste viene detta condizione di modulo, la seconda condizione di fase.

Alla prima di tali condizioni corrisponde la proprietà che, in ogni punto del luogo, il valore del modulo di K' è pari al prodotto tra le lunghezze dei vettori che congiungono il punto stesso con i poli della F(s) diviso il prodotto delle grandezze dei vettori che congiungono il punto con gli zeri della F(s).

La seconda condizione ha un utilizzo analogo ma stavolta invece di prodotti compaiono somme e differenze, sta ad indicare che la somma degli argomenti dei vettori orientati dai poli della F(s) verso un punto del luogo meno la somma degli argomenti dei vettori orientati dagli zeri della F(s) verso lo stesso punto del luogo è pari a π o 0. Il termine 2hπ serve a distinguere se ci troviamo nel piano positivo o quello negativo.

È necessario definire alcune regole per il tracciamento:

- Per K=0. il luogo delle radici parte dai poli in catena aperta;
- Il luogo delle radici è simmetrico rispetto all'asse reale;
- Tutto l'asse reale appartiene al luogo delle radici;
- Appartengono al luogo positivo quelle porzioni di asse che lasciano a destra un numero dispari di poli e/o zeri di F(s), contati con la loro molteplicità. Gli altri punti appartengono al luogo negativo;
- I *punti singolari* sono radici multiple della f(s,k), in particolare sono quei punti che annullano la sua derivata e la funzione stessa;
- Per K che tende ad infinito la f(s,k) ha n soluzioni, essendo n>m i rami del luogo partono da n poli. Inoltre, m rami tendono agli zeri mentre gli altri n-m rami tendono ad n-m semirette centrate nel **centro degli asintoti**:

$$s_0 = \frac{\prod_{i=1}^{n} p_i - \prod_{i=1}^{m} z_i}{n - m} \tag{30}$$

5 Analisi del comportamento del modello con Controllore PID

Come è stato già illustrato in precedenza, il Mambo "Parrot" implementa al suo interno un controllore realizzato mediante PID; il nostro scopo è quello di analizzarne il comportamento.

Per poter realizzare un'analisi completa del comportamento, abbiamo bisogno di inserire, all'interno del sotto-sistema ausiliario *Logging*, dei vari blocchi di tipo

Scope all'interno dei quali è possibile consultare l'andamento delle varie variabili (X,Y,Z,pitch,roll,yaw) coinvolte nel modello.

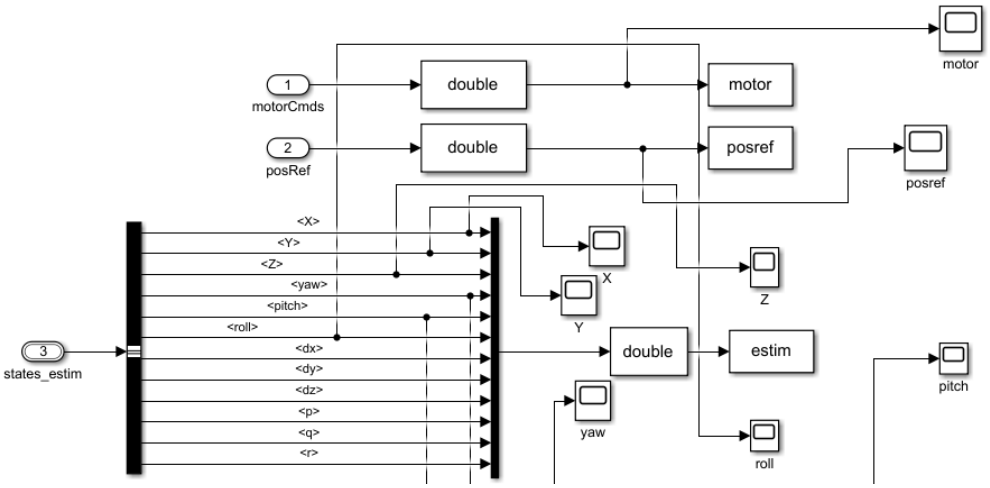


Figure 25: Blocco Logging con Scopes

L'analisi del comportamento del controllore PID può essere fatta sia per il modello non lineare e sia per il modello lineare, basta cambiare la variabile VSS VEHICLE per passare da un modello all'altro. Ora di seguito è presente l'andamento delle varie variabili per il **modello non lineare**:

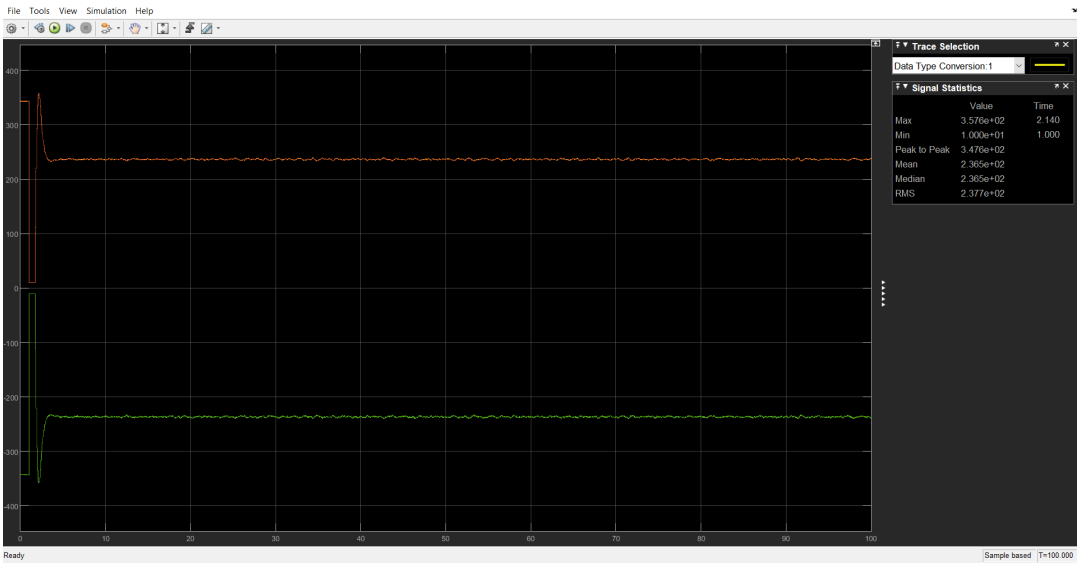


Figure 26: Tracciato dei motori con Controller PID

Notiamo come i motori ricevano un segnale che li porta prima a una certa soglia per poi stabilizzarsi, questo è necessario affinché il drone raggiunga una certa quota.

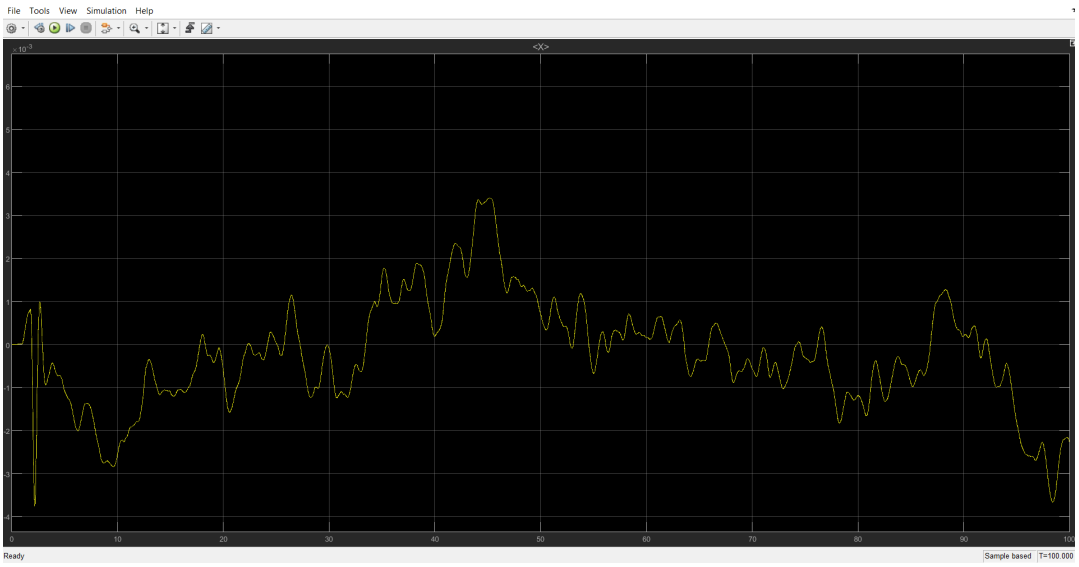


Figure 27: Tracciato X con Controller PID

Notiamo che il controllo sulla posizione X garantisce delle oscillazioni limitate rispetto al riferimento, nell'ordine di 10^{-3} [m], questo testimonia la buona qualità dell'azione di controllo del PID del modello originale. Analoghe considerazioni valgono per il controllo sulla posizione Y.

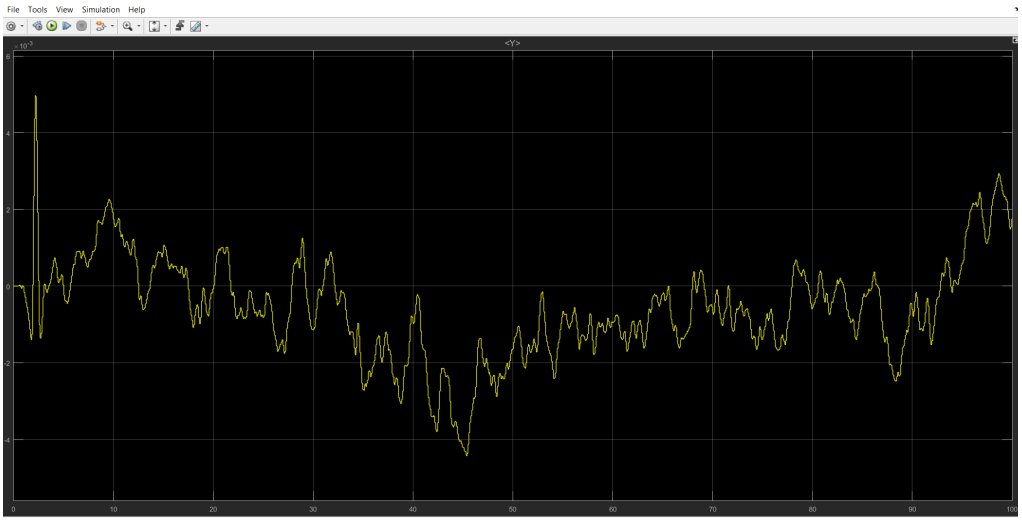


Figure 28: Tracciato Y con Controller PID

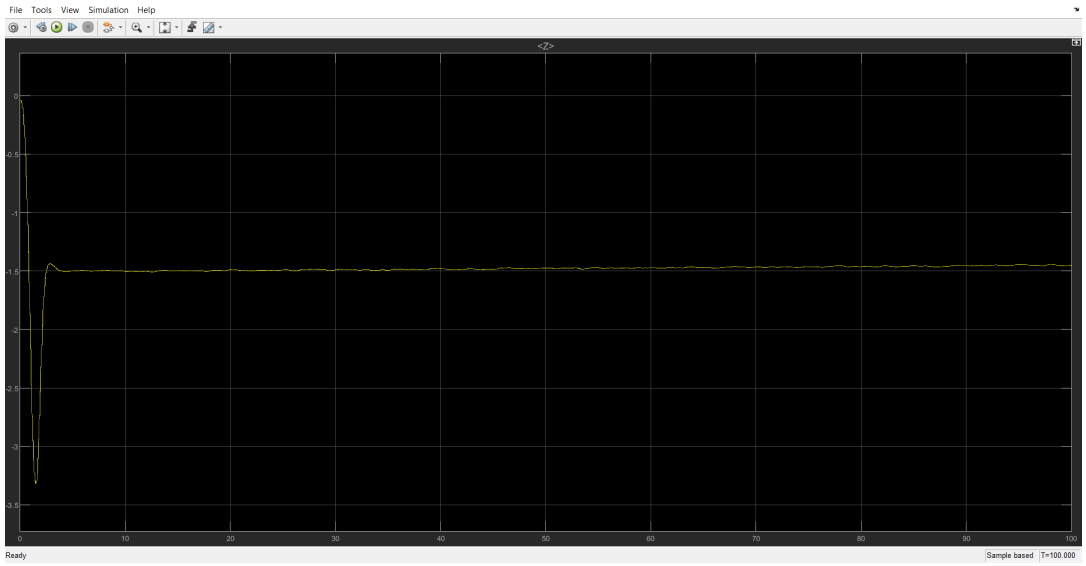


Figure 29: Tracciato Z con Controller PID

La posizione Z si assesta, dopo una sovraelongazione iniziale, al valore di 1.5 m sul suolo. L'ordinata è negativa in quanto il sistema di riferimento è di tipo NED (North East Down).

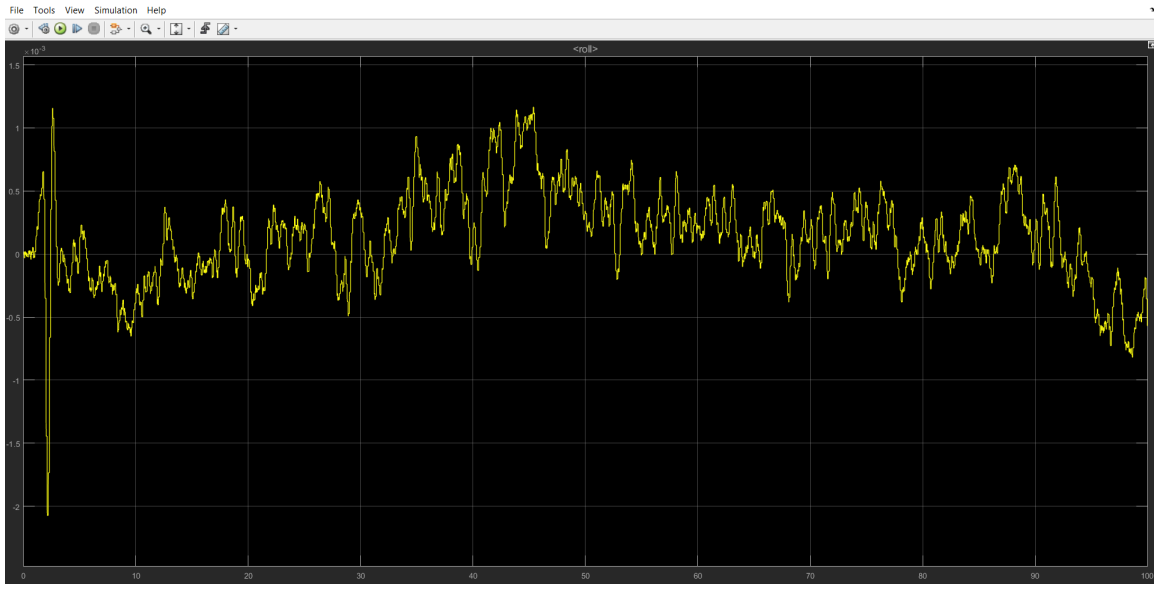


Figure 30: Tracciato roll con Controller PID

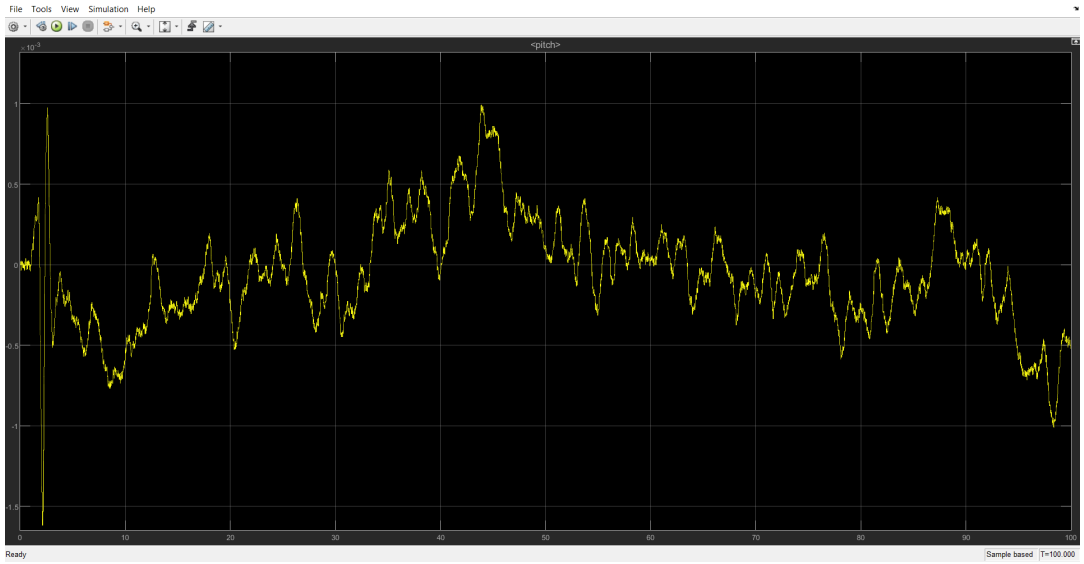


Figure 31: Tracciato pitch con Controller PID

Per gli angoli di Pitch e Roll facciamo le stesse considerazione fatte per X e Y: le variazioni sono dell'ordine di 10^{-3} [m] e questo testimonia il buon funzionamento del controller originale.

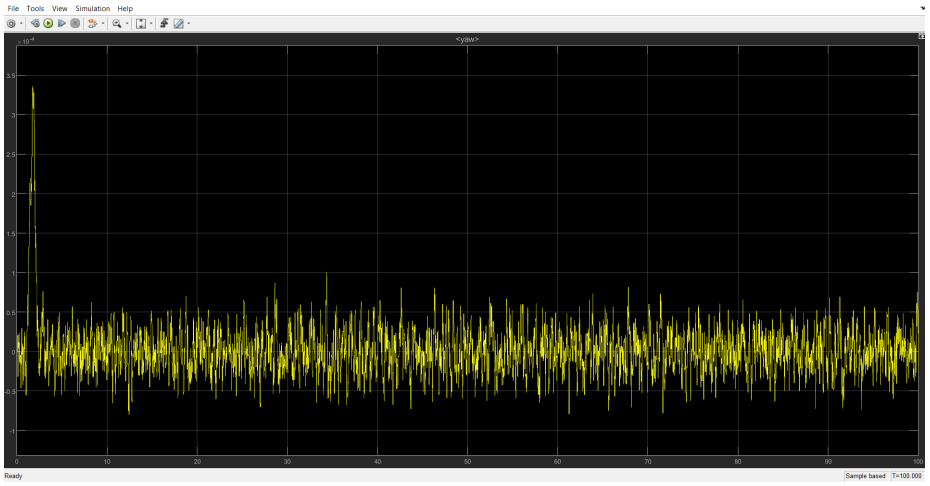


Figure 32: Tracciato yaw con Controller PID

Il controllo per quanto riguarda lo Yaw, come abbiamo detto anche precedentemente, è direttamente collegato al controllo degli angoli di Pitch e Roll, è ancora più preciso; dopo una fase di assestamento iniziale si stabilizza intorno a valori che variano nell'ordine del 10^{-4} [m].

I grafici ottenuti dall'analisi del comportamento del modello non lineare, valgono anche per il modello lineare in quanto gli andamenti delle varie variabili sono uguali.

Una volta capito l'andamento del modello utilizzando il controller PID, bisogna andare a progettare il nuovo controllore tramite il luogo delle radici, ed è quello di cui parlerà la prossima sezione.

6 Sintesi e implementazione del controllore con Luogo delle radici

In questa sezione vediamo come è stato realizzato il nuovo controllore tramite il Luogo delle radici. Per poter realizzare il nuovo controllore è stato necessario utlizzare il modello lineare, implementato in Matlab, perchè ci ha fornito una panoramica semplificata sulla dinamica del modello e delle variabili di stato.

Da questo è stato possibile estrapolare il sottosistema relativo all'angolo di pitch, sul quale è stato progettato un controllore che poi è stato inserito all'interno del blocco attitude, il modello lineare permette di trattare separatamente i legami tra le varie variabili.

Sono stati isolati gli ingressi e le uscite relative all'angolo di Pitch e in questo modo si è riuscito a trovare un legame ingresso/uscita dal quale poi abbiamo ricavato la funzione di trasferimento che ci è servita per costruire il controllore con il metodo del "Luogo delle radici", il controllore ha dovuto soddisfare determinate specifiche dette univoche e lasche.

Una volta realizzato il controllore, esso è stato implementato in Simulink e dopo aver effettuato alcuni test , è stato verificato il suo funzionamento sia per modello lineare che per modelli non lineare.

6.1 Progetto del Controllore

Il progetto del controllore si divide in piu fasi:

1. Disaccoppiamento Ingresso/Uscite del modello lineare per ricavare il sottosistema relativo all'angolo di Pitch e successivamente calcolo della funzione di trasferimento;
2. Luogo delle Radici e calcolo della G(s) del controllore;
3. Implementazione del controllore in Simulink

6.1.1 Disaccoppiamento Ingressi/Uscite

Per effettuare il Disaccoppiamento Ingressi/Uscite è possibile utilizzare il modello lineare del nostro sistema (Linear Airframe), all'interno del quale sono contenute le matrici A,B,C,D che descrivono l'intero modello.

1. ***Matrice A*** : Matrice della dinamica degli stati 12x12, racchiude tutte le variabili del sistema;
2. ***Matrice B*** : Matrice degli ingressi del sistema 12x14;
3. ***Matrice C*** : Matrice delle uscite del sistema 33x12, descrive le uscite del sistema rispetto agli stati;
4. ***Matrice D*** : Matrice 33x14 che descrive l'azione delle variabili d'ingresso sull'uscita

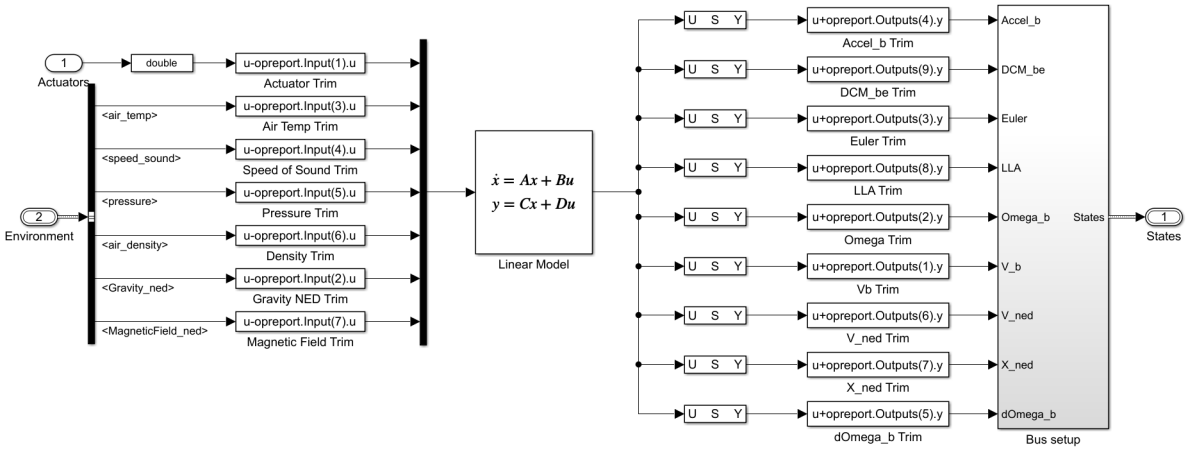


Figure 33: Modello Lineare del Processo

E' stato possibile, da queste matrici, considerare solo la parte di sistema che descrive l'angolo di pitch, cioè è stato sufficiente considerare:

- Nella **matrice A** i coefficienti relativi alle variabili di stato θ e q;
- Nella **matrice B** i coefficienti relativi alle variabili di stato θ e q relative agli ***Actuators(1,2,3,4)***;
- Nella **matrice C** i coefficienti relativi all'uscita *Euler2* relativi all'angolo di pitch;
- Nella **matrice D** i coefficienti relativi all'uscita *Euler2* legati agli ***Actuators(1,2,3,4)***;

I valori dei coefficienti delle matrici A,B,C,D sono consultabili attraverso il Workspace di Matlab, all'interno del gruppo di variabili denominato ***linsys***.

Quindi per ottenere i valori delle quattro matrici, è stato necessario, prima caricare il modello e poi, tramite linsys, ottenere i valori, e il tutto è stato realizzato con queste righe di codice:

Carico il modello lineare e visualizzo a schermo le matrici A,B,C,D

```
load('linearizedAirframe.mat', 'linsys')
linsys.a
linsys.b
linsys.c
lynsis.d
```

Figure 34: Codice per ottenere le Matrici

Il sottosistema ottenuto è il seguente:

$$A = \begin{bmatrix} 0 & 1 \\ 0 & -1.692 \end{bmatrix} \qquad B = \begin{bmatrix} 0 \\ 0.3133 \end{bmatrix} \qquad C = \begin{bmatrix} 1 & 0 \end{bmatrix} \qquad D = \begin{bmatrix} 0 \end{bmatrix}$$

Una volta trovate le matrici A,B,C,D e caricate all'interno del workspace, bisogna trovare due costanti moltiplicative essenziali per la costruzione del legame ingresso-uscita relativo alla variabile da controllare (pitch).

Queste due costanti sono **k1** e **k2**, in cui k1 ci fornisce la partizione del segnale di controllo tra i diversi motori, mentre k2 rappresenta il coefficiente di spinta dei motori..

L'azione della prima costante è descritta all'interno del blocco *FCS/FlightController/Control Mixer*, ed è ottenibile lanciando il comando ***Controller.Q2Ts***, che ci fa ottenere una matrice che riguarda le costanti di controllo; la prima colonna fornisce i coefficienti riferiti alla spinta verticale (*totalThurst*), la seconda quelli riferiti al momento di Yaw (*tau yaw*), i valori della terza colonna rappresentano l'angolo di pitch (*tau pitch*) che è *5.6679* e l'ultima quelli riferiti al momento di roll (*tau roll*).

```
>> Controller.Q2Ts

ans =

    0.2500  103.5736   -5.6659   -5.6659
    0.2500 -103.5736   -5.6659    5.6659
    0.2500  103.5736    5.6659    5.6659
    0.2500 -103.5736    5.6659   -5.6659
```

L'azione della seconda costante è contenuta all'interno del blocco *ThrustToMotorCommands* sempre all'internio del *FCS* e con il comando ***Vehicle.Motor.thrustToMotor Command*** otteniamo il coefficiente di spinta dei motori, il quale è $1.5307{*}10^3$.

```
>> Vehicle.Motor.thrustToMotorCommand

ans =

   1.5307e+03
```

Una volta trovate anche le costanti k1 e k2 abbiamo tutto il necessario per trovare la funzione di trasferimento, che in Matlab è possibile calcolare tramite il comando *tf*. In particolare la formula per la fdt è:

$$FDT = C(SI - A)^{-1}B \tag{31}$$

e inserendo le opportune matrici in Matlab e applicando il comando tf con k1 e k2, otteniamo;

```
>> A=[0 1; 0 -1.692];
>> B=[0; 0.3133];
>> C=[1 0];
>> D=[0];
>> syms s;
>> sys=ss(A,B,C,D);
>> plant=tf(sys);
>> k1=5.6679;
>> k2=1530.7;
>> Ps=plant*k1*k2

Ps =

      2718
  -------------
  s^2 + 1.692 s

Continuous-time transfer function.
```

6.1.2 Luogo delle Radici e calcolo G(s) del controllore

Il sistema deve soddisfare due specifiche univoche ed un'unica specifica lasca. La prima specifica univoca è gia rispettata dal sistema in sè perchè riguarda il tipo del sistema, cioè la prima specifica è che il nostro sistema sia di tipo 1 il che è già soddisfatta dal momento in cui P(s) ha un polo nell'origine. La seconda specifica univoca riguarda l'errore a regime permanente $\tilde{e}$, la quale è $\tilde{e} \leq 0.01$.

L'ultima specifica che il sistema deve soddisfare è una specifica lasca e riguarda i poli in catena chiusa, essi devono avere parte reale < -1.

Come abbiamo già detto la specifica riguardante il tipo del sistema è già soddisfatta, per la seconda specifica, considerando la formula generale dell'errore a

regime permanente, affinchè la specifica sia soddisfatta dobbiamo considerare un K abbastanza piccolo ovvero $K \leq 0.037$.

Abbiamo trovato un controllore che deve soddisfare le specifiche univoche, e tramite il luogo delle radici vediamo se soddisfa anche quella lasca. Per lavorare piu rapidamente ho utilizzato un toolbox messo a disposizione da Matlab ovvero *Sisotool* che mi permette di visualizzare a schermo i diagrammi di bode, luogo delle radici e risposta al gradino di una determinata funzione passatagli in ingresso. Quello che otteniamo con il primo controllore sono i seguenti grafici:

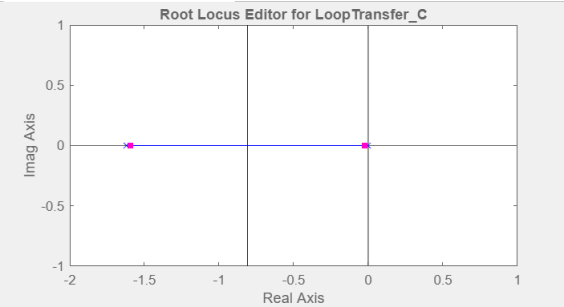


Figure 35: Luodo delle Radici dopo controllore di primo tentativo

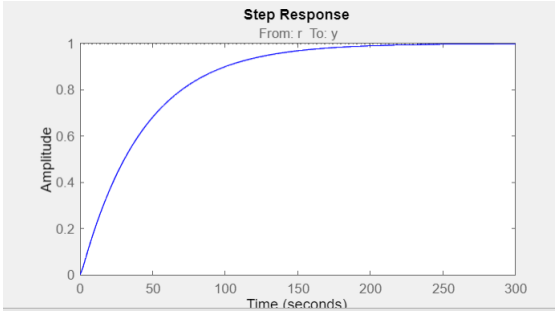


Figure 36: Risposta al gradino dopo controllore di primo tentativo

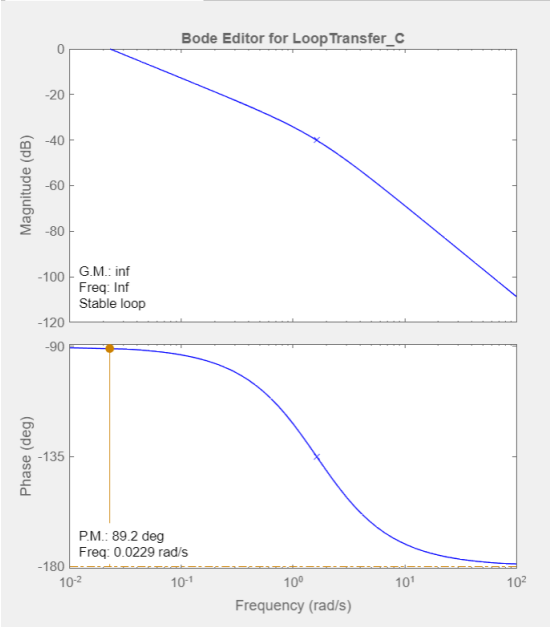


Figure 37: Diagrammi di Bode dopo controllore di primo tentativo

Il luogo tracciato dal toolbox fa subito notare il fatto che le specifiche univoche sono soddisfatte e che la specifica riguardante i poli in catena chiusa non è soddisfatta, ragion per cui è necessario progettare una funzione compensatrice capace di spostare il centro degli asintoti e di conseguenza anche i vari poli e zeri.

Affinchè i poli siano nella regione di stabilità è necessario progettare un controllore di questo tipo:

$$G(s) = K\frac{(s - z_i)}{(s - p_i)} \tag{32}$$

cioè introduco la coppia polo-zero la quale permette lo spostamento dei poli e degli zeri.

Alla fine della progettazione, il nuovo controllore ha questa forma qui:

$$G(s) = 0.037\frac{(s + 2)}{(s + 100)} \tag{33}$$

e per vedere se le specifiche univoche continuano ad essere soddisfatte e se è soddisfatta anche quella lasca, ci torna in aiuto il toolbox Sisotool per tracciare i vari grafici:

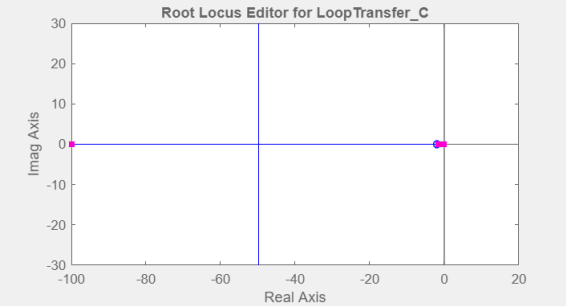


Figure 38: Luogo delle radici con nuovo controllore

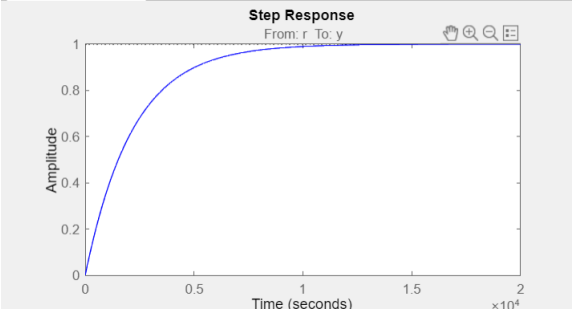


Figure 39: Risposta al gradino con nuovo controllore

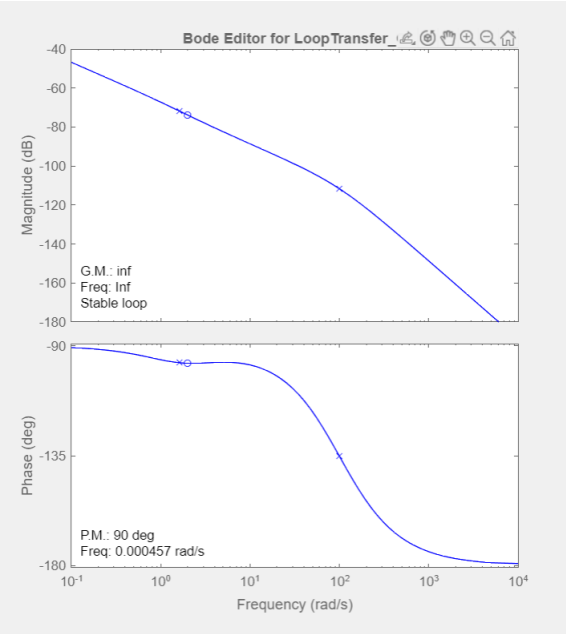


Figure 40: Diagrammi di Bode con nuovo controllore

6.1.3 Implementazione del controllore in Simulink

Una volta calcolato il nuovo controllore G(s), è stato implementato in Simulink. Inizialmente pitch e roll erano controllati entrambi da un controllore PID, e per questo è stato necessario disaccoppiare il controllo di questi attraverso un *demultiplexer* e di conseguenza l'angolo di roll è sempre il PID a controllarlo, mentre l'angolo di pitch viene controllato dal nuovo controllore appena progettato:

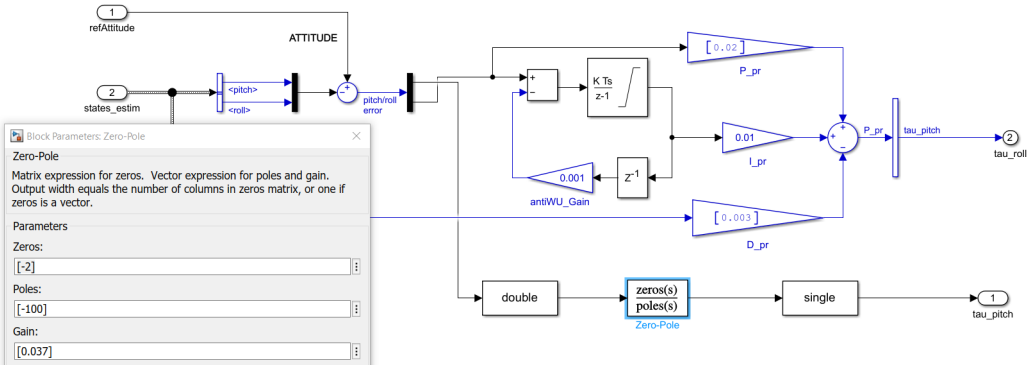


Figure 41: Implementazione controllore in Simulink

Da notare come l'ingresso *Refattitude* provenga dal blocco *Switch refAtt*, che in base a come è settato, decide se prendere in ingresso il segnale dell'XY orientation o il segnale dal Signal Builder. Questa scelta non è ininfluente poichè, se il segnale proviene dal blocco XY orientation, che è anche lui un controllore, si crea un controllo a cascata che genera un segnale variabile nel tempo, mentre se il segnale proviene dal Signal Builder il segnale è un segnale costante nel tempo.

E' possibile forzare il segnale dal Signal Builder attraverso le porte logiche AND,modificate in precedenza, e far scegliere al nostro sistema sempre un riferimento costante per l'angolo di pitch.

6.2 Simulazione del comportamento del controllore

L'ultima cosa da fare, una volta implementato il nuovo controllore all'interno del Simulink, è vedere il comportamento delle varie variabili rispetto al nuovo controllore.

Come già detto in precedenza, questo sistema possiede un modello lineare e un modello non lineare, a seconda del valore di *VSS VEHICLE*, perciò verrà analizzato il comportamento sia del modello lineare che del modello non lineare.

Inoltre verrà analizzato il comportamento dei due modelli in relazione a due segnali fondamentali ovvero:

- Segnale di riferimento variabile;
- Segnale di riferimento costante.

La scelta tra i due viene effettuata nella porta logica AND, ovvero:

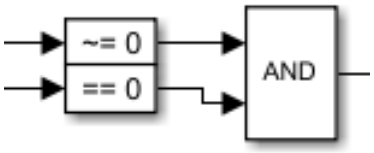


Figure 42: Segnale di riferimento costante

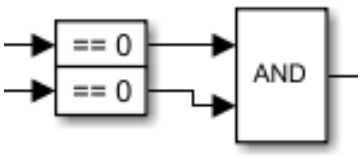


Figure 43: Segnale di riferimento variabile

Questi due sono i settaggi della porta AND per ottenere un riferimento variabile e costante.

Il primo comportamento che verrà analizzato è quello del **modello lineare con**, in ingresso, **un riferimento variabile**:

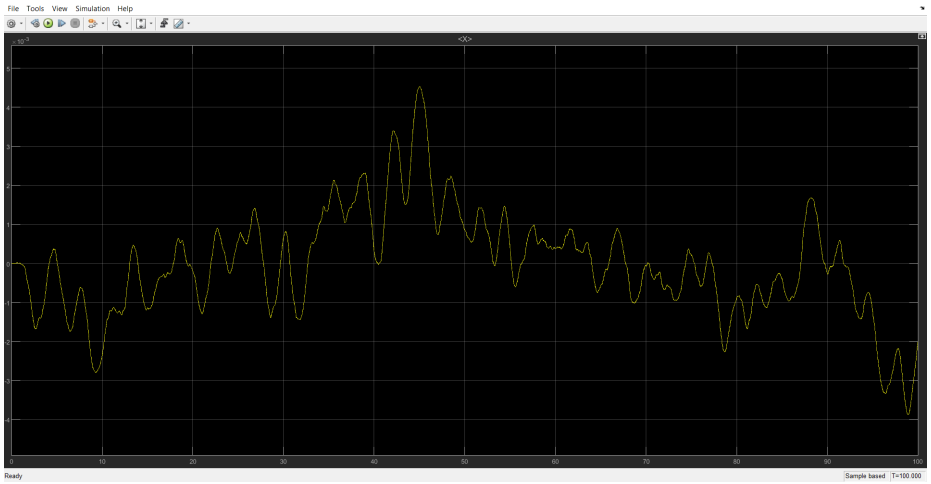


Figure 44: Tracciato X

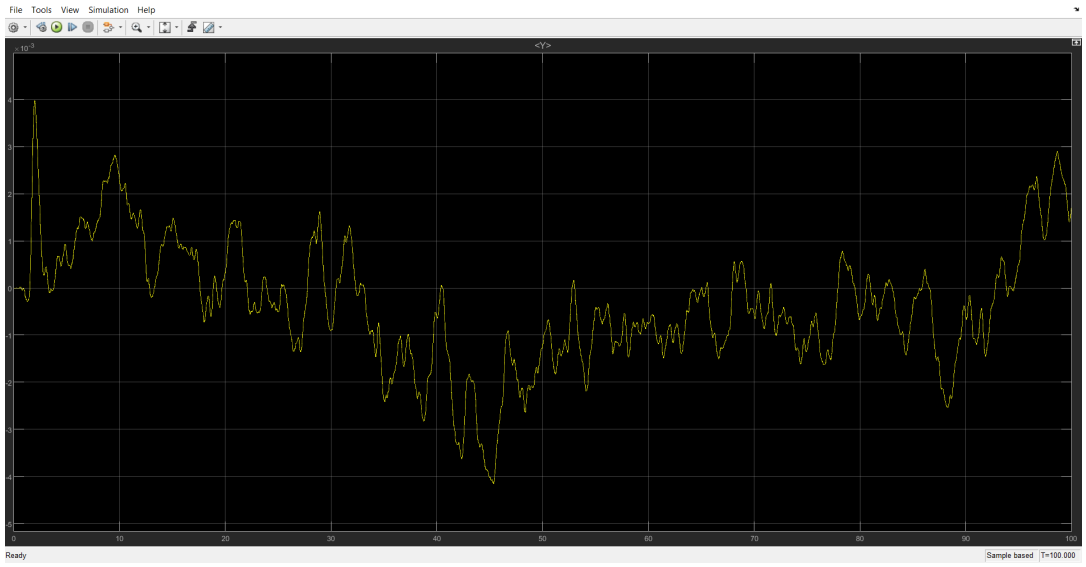


Figure 45: Tracciato Y

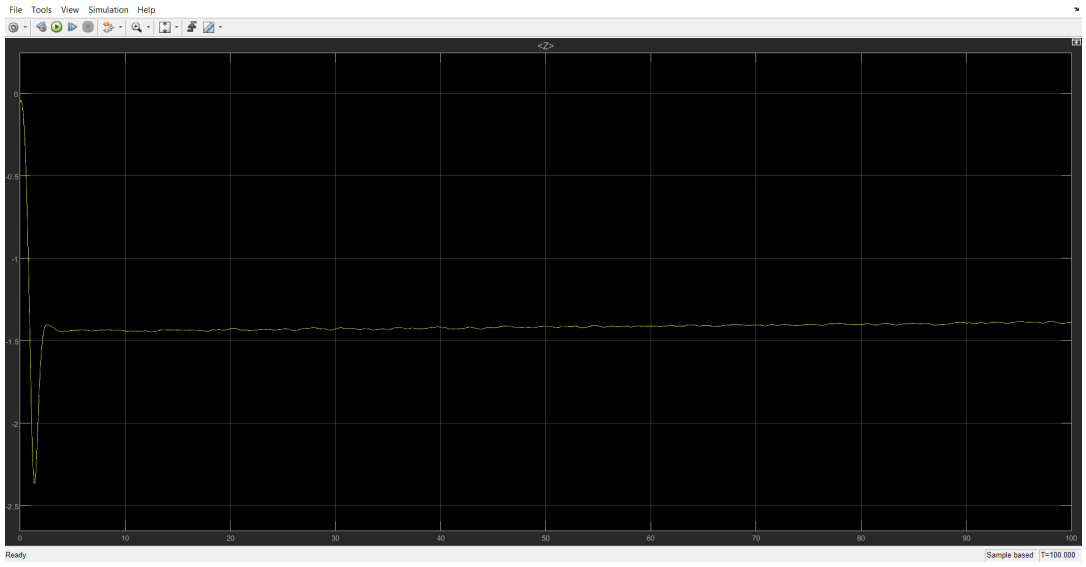


Figure 46: Tracciato Z

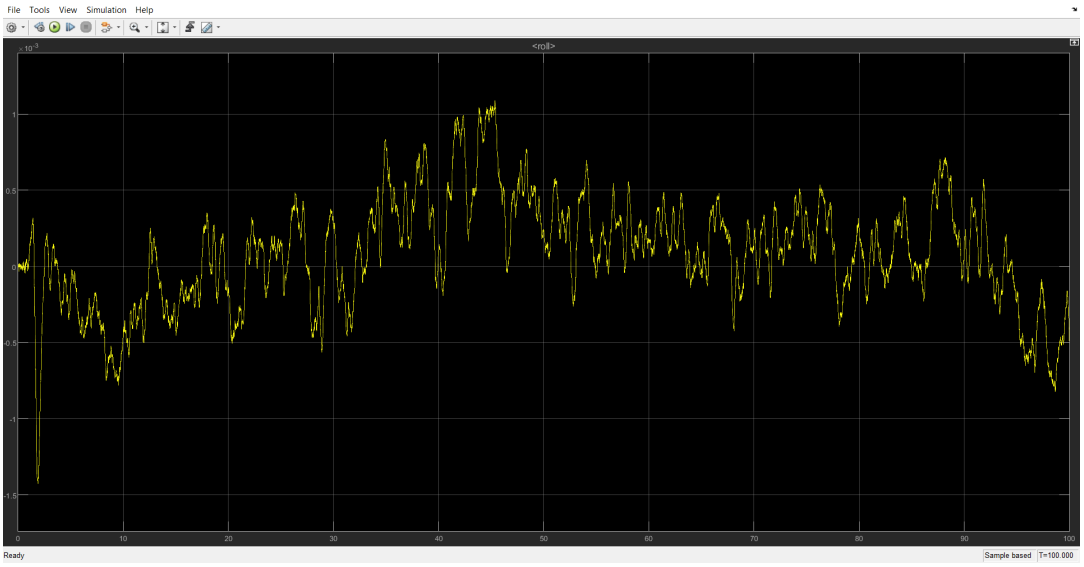


Figure 47: Tracciato Roll

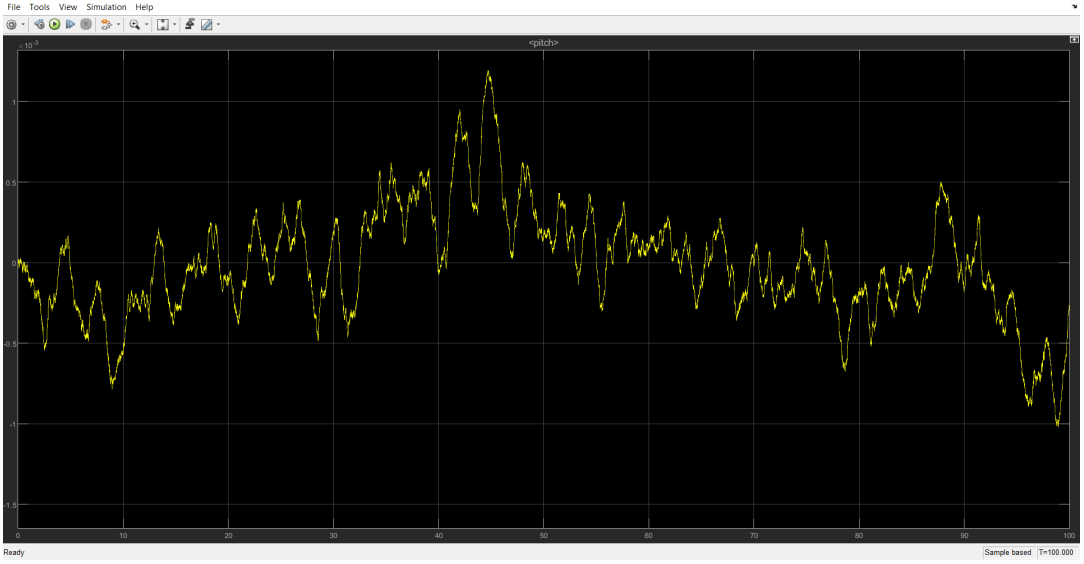


Figure 48: Tracciato Pitch

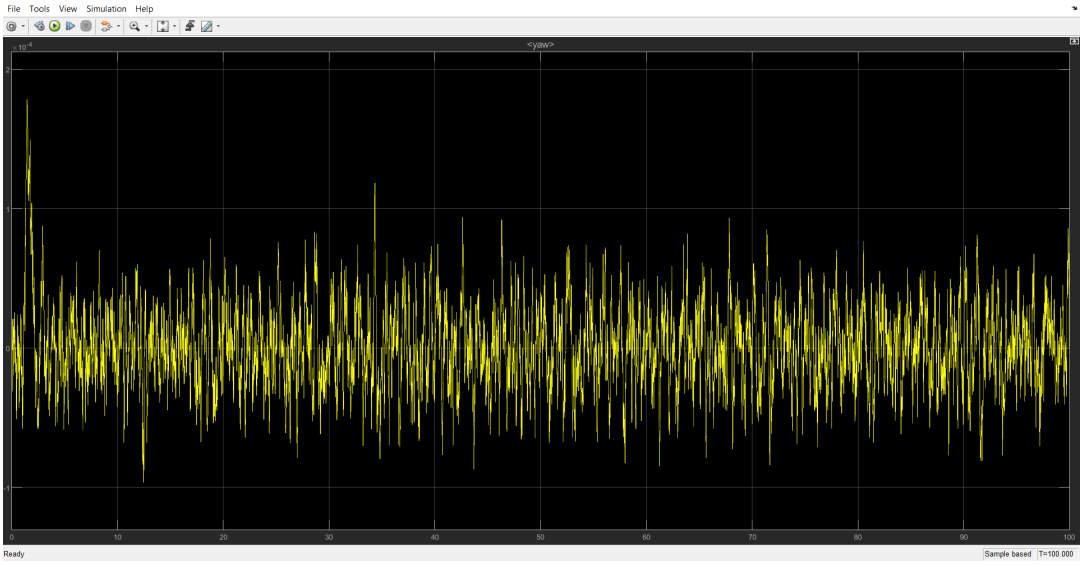


Figure 49: Tracciato Yaw

Una volta analizzato il modello lineare con riferimento variabile, ora verrà analizzato il **modello lineare con riferimento costante**:

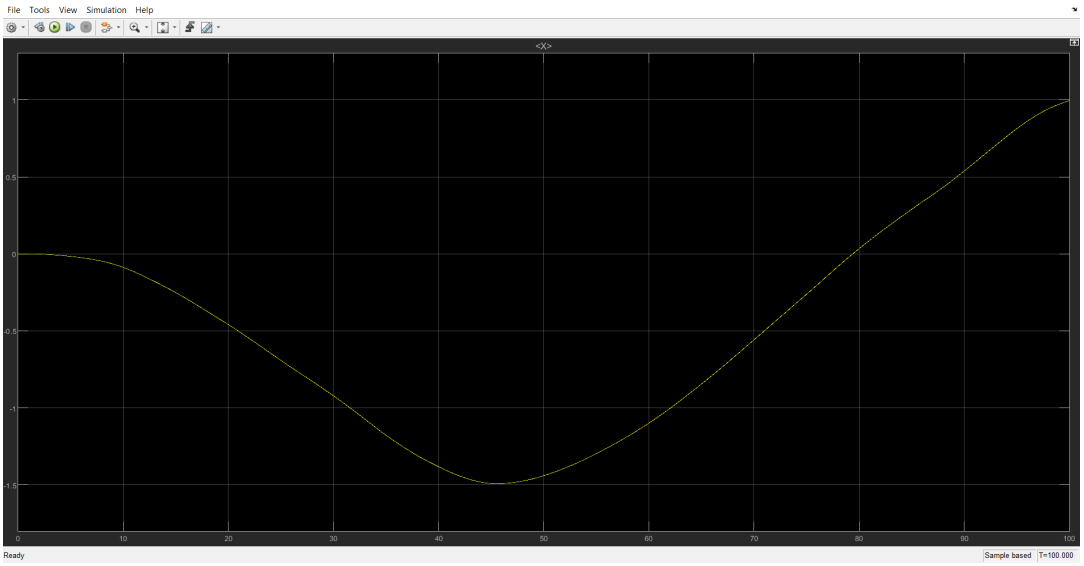


Figure 50: Tracciato X

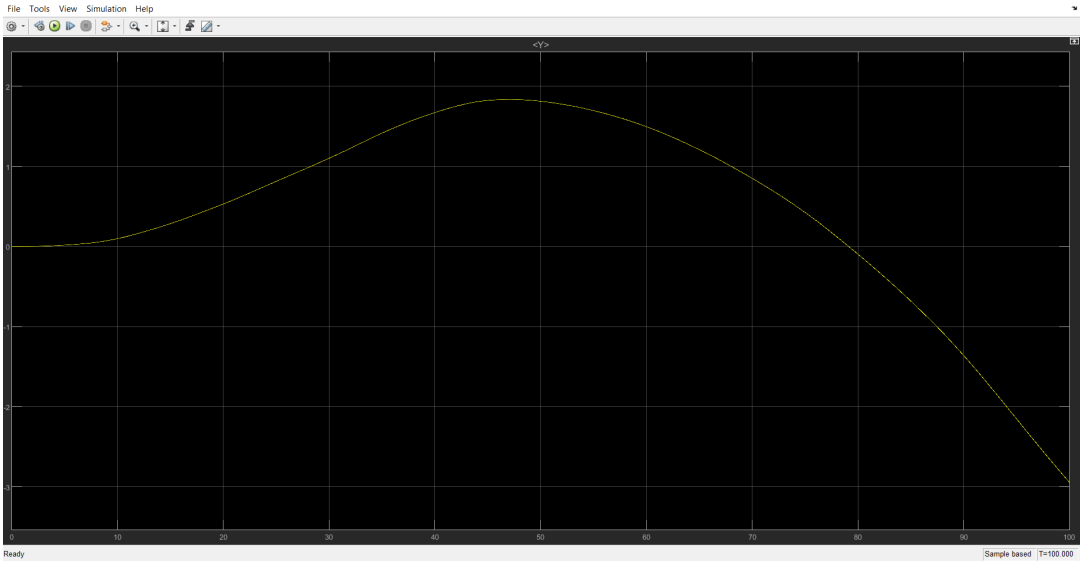


Figure 51: Tracciato Y

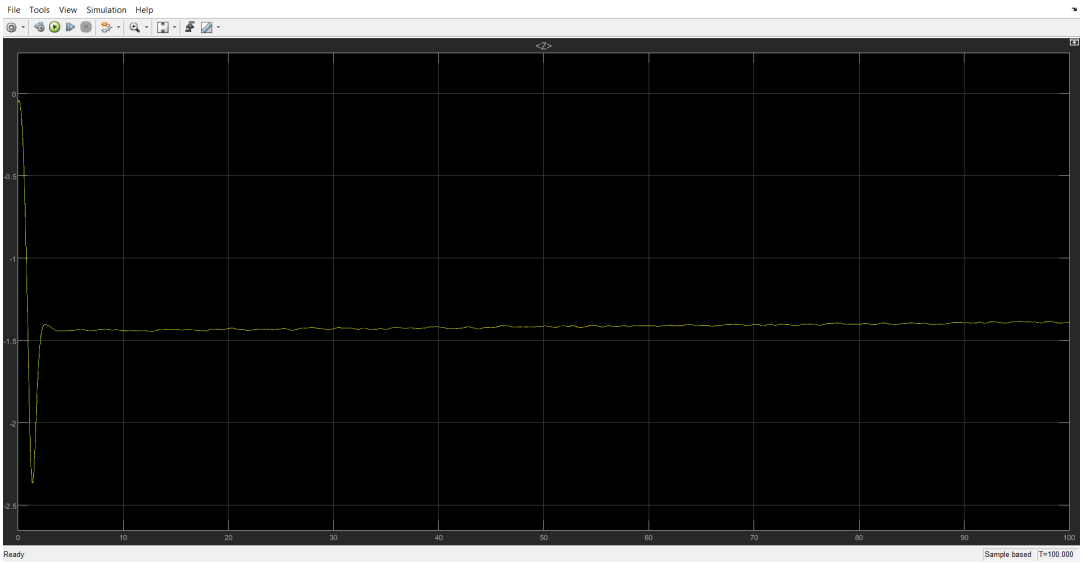


Figure 52: Tracciato Z

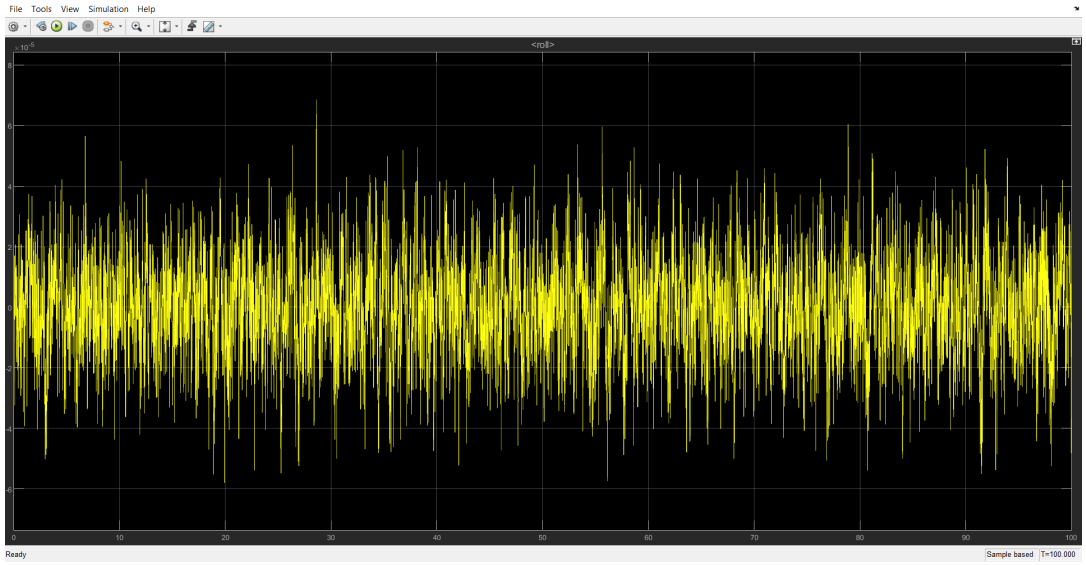


Figure 53: Tracciato Roll

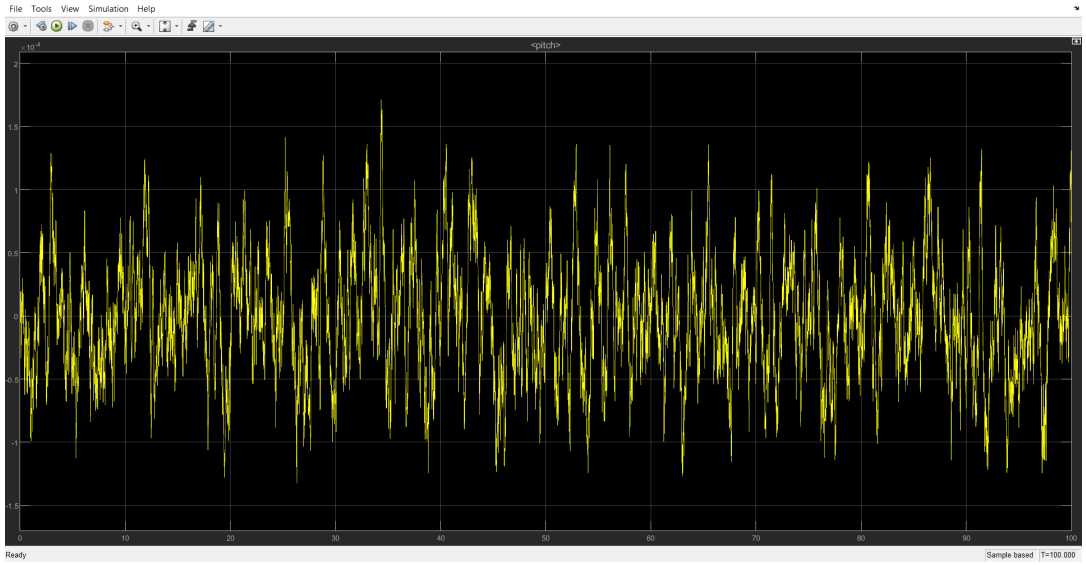


Figure 54: Tracciato Pitch

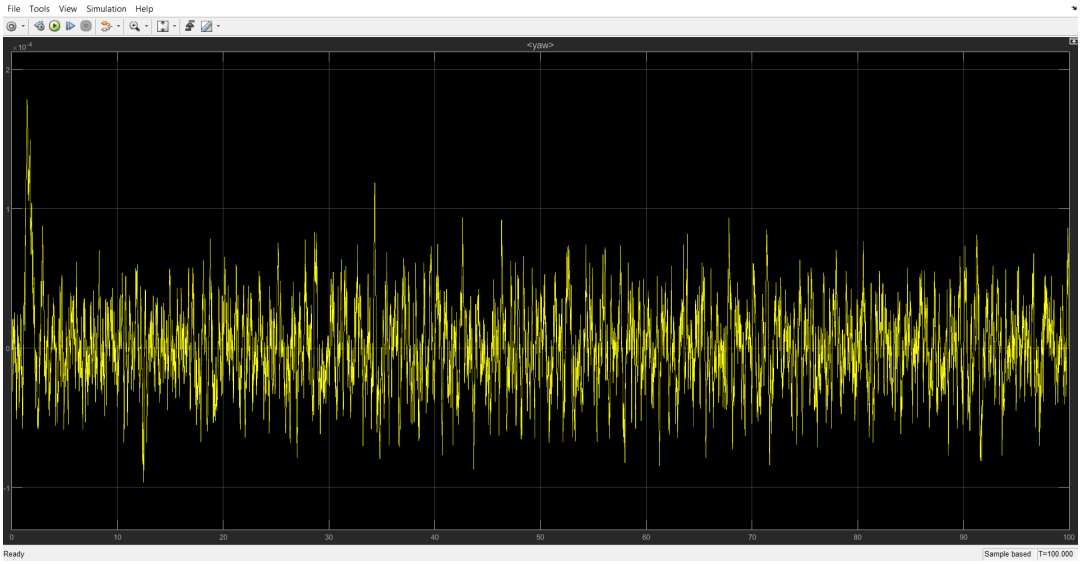


Figure 55: Tracciato Yaw

Ora passiamo all'analisi del modello non lineare e iniziamo con il **modello non lineare con riferimento variabile**:

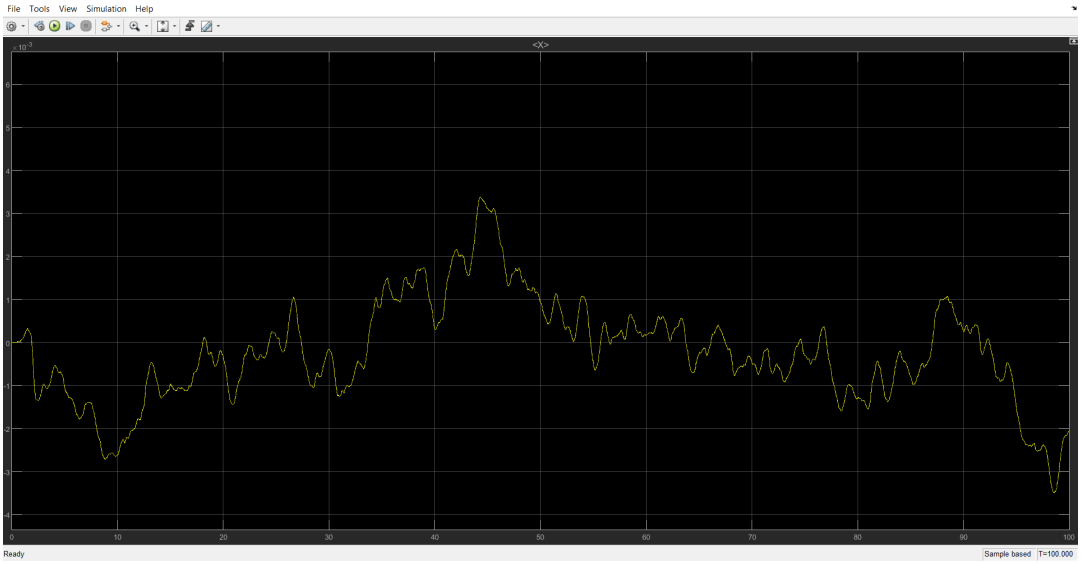


Figure 56: Tracciato X

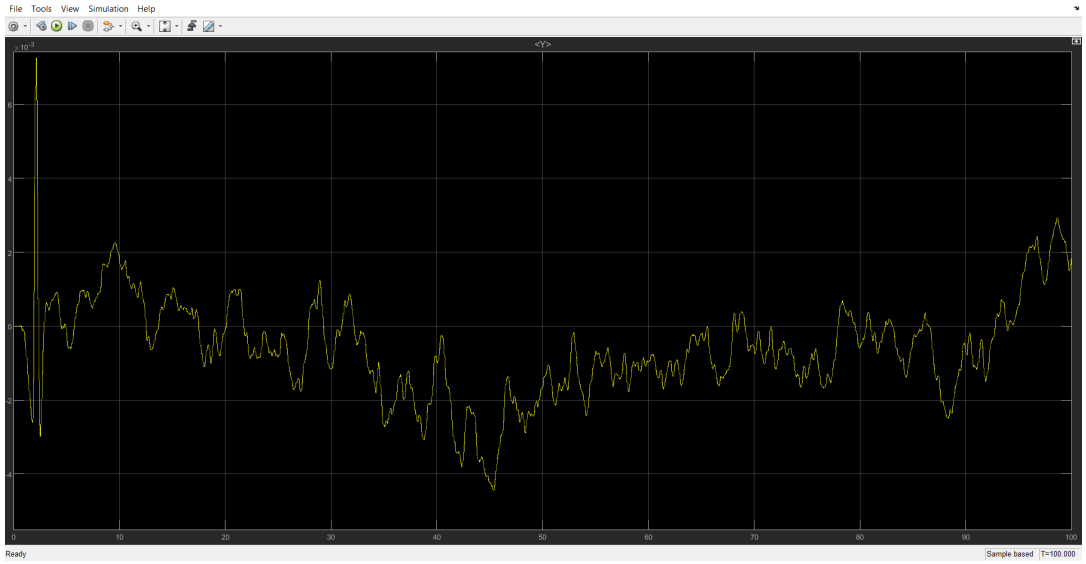


Figure 57: Tracciato Y

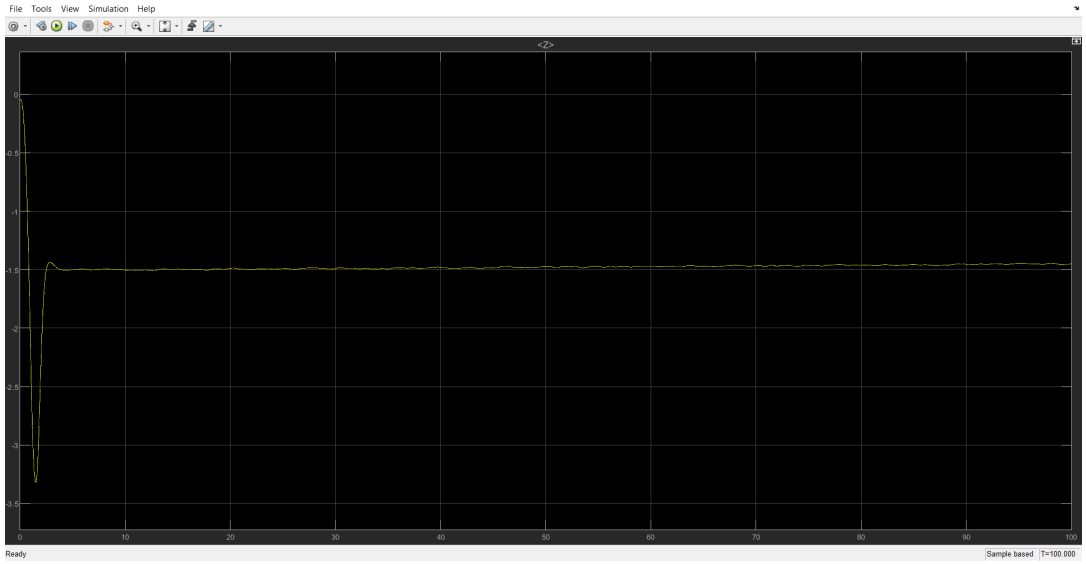


Figure 58: Tracciato Z

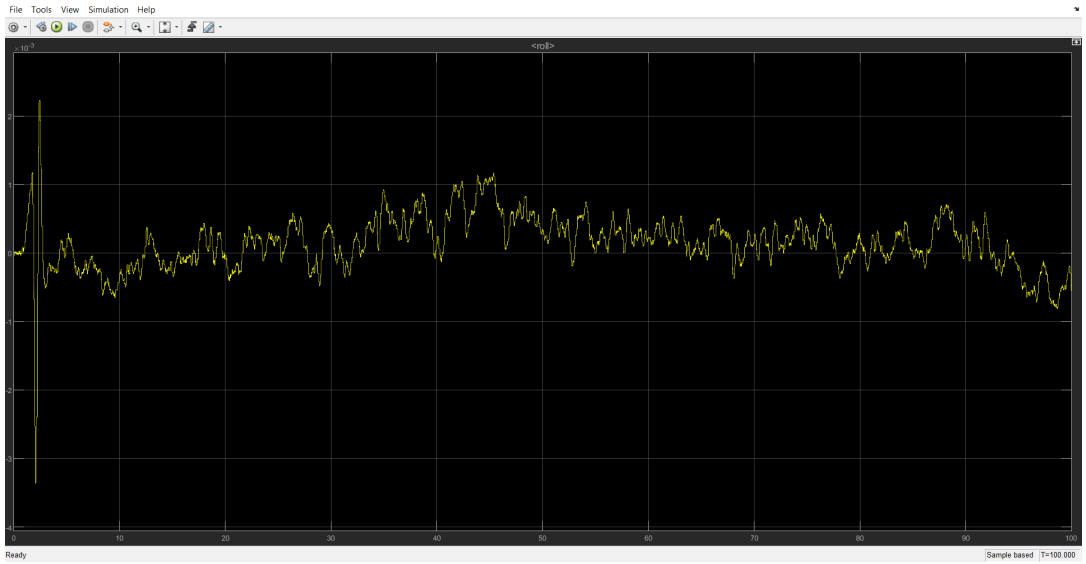


Figure 59: Tracciato Roll

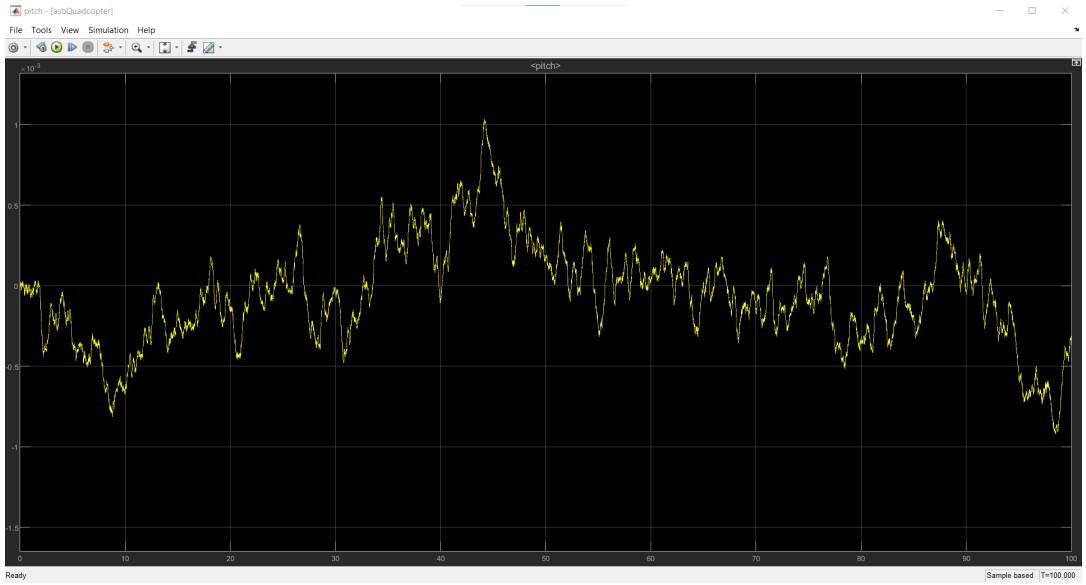


Figure 60: Tracciato Pitch

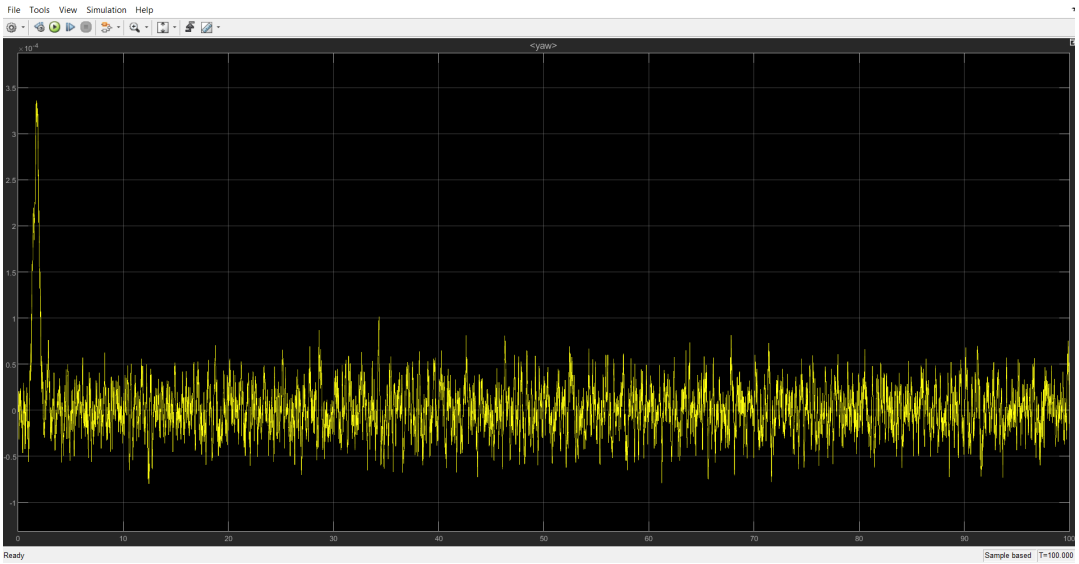


Figure 61: Tracciato Yaw

L'ultima analisi è del **modello non lineare con riferimento costante**:

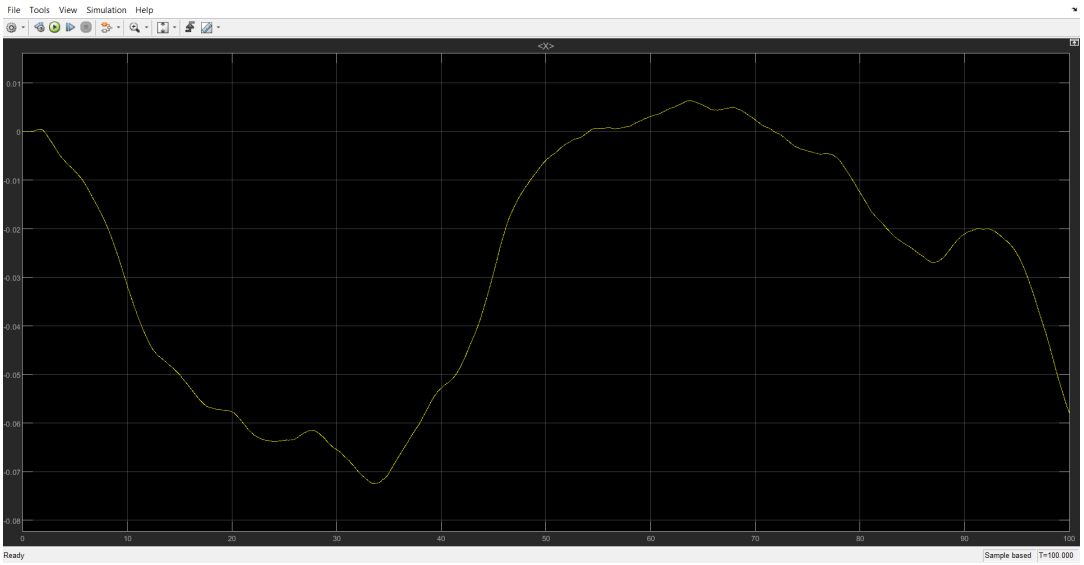


Figure 62: Tracciato X

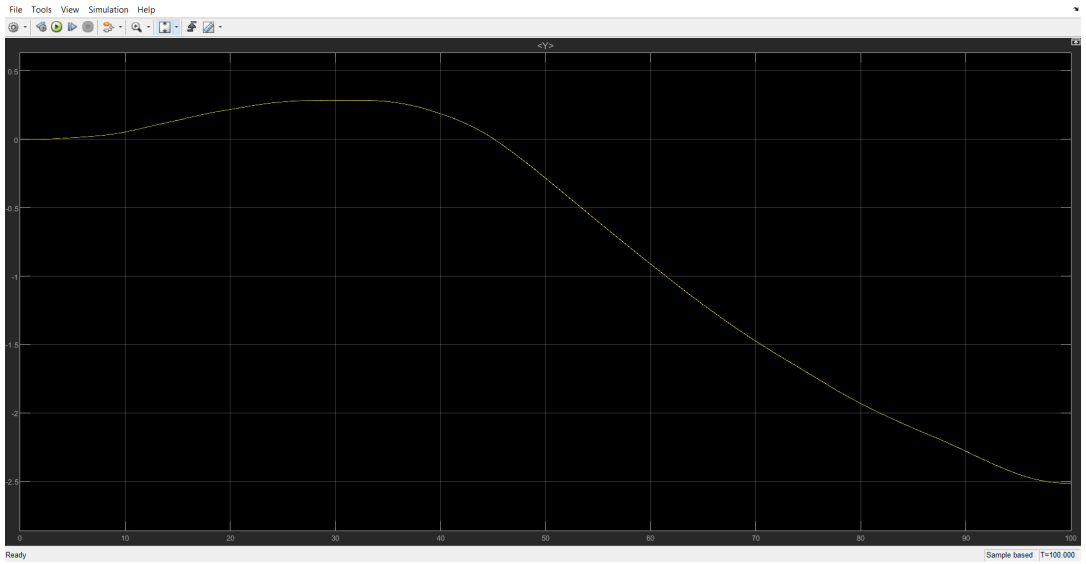


Figure 63: Tracciato Y

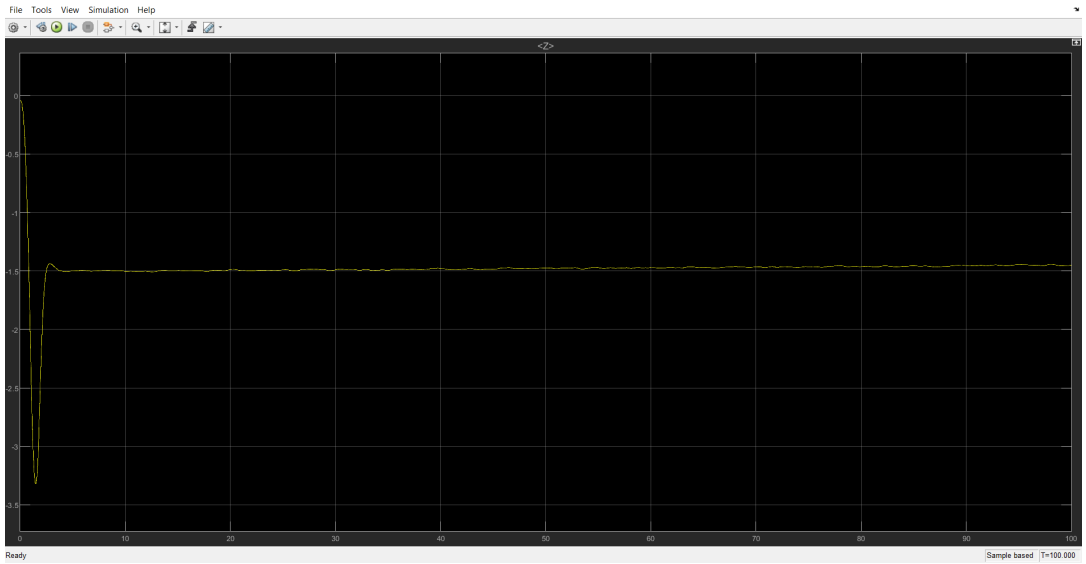


Figure 64: Tracciato Z

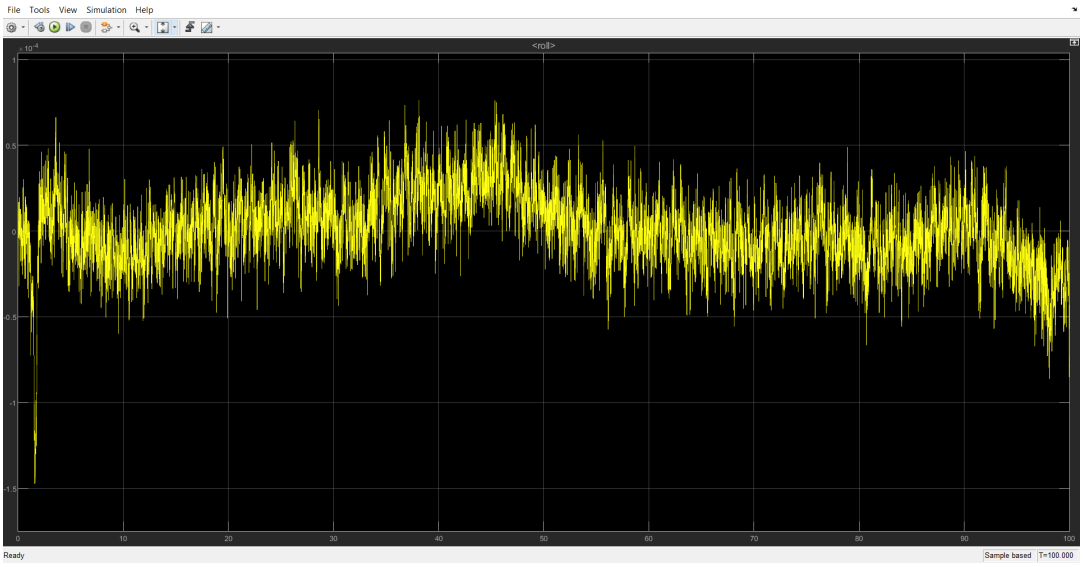


Figure 65: Tracciato Roll

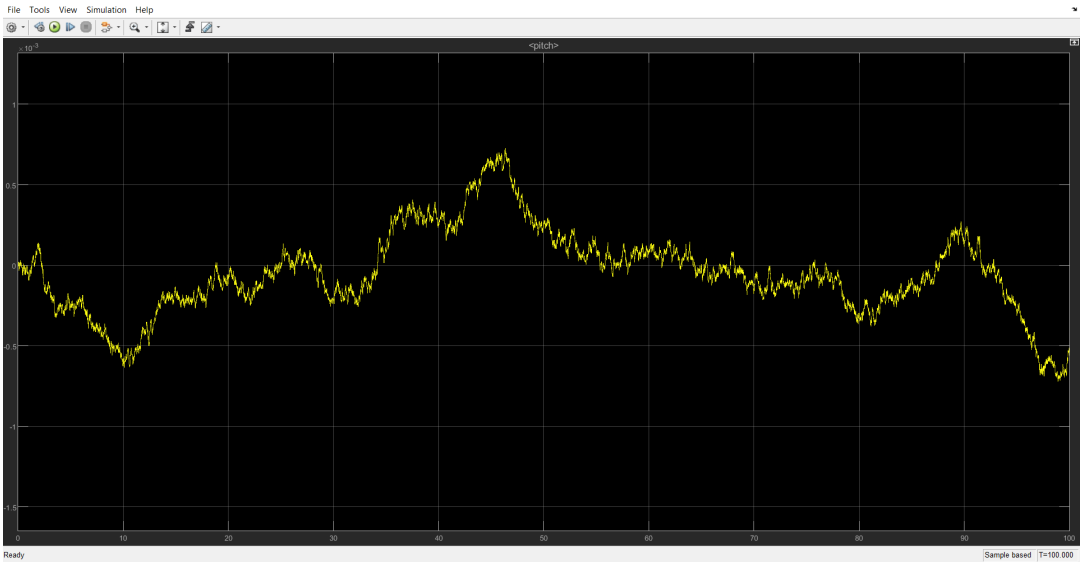


Figure 66: Tracciato Pitch

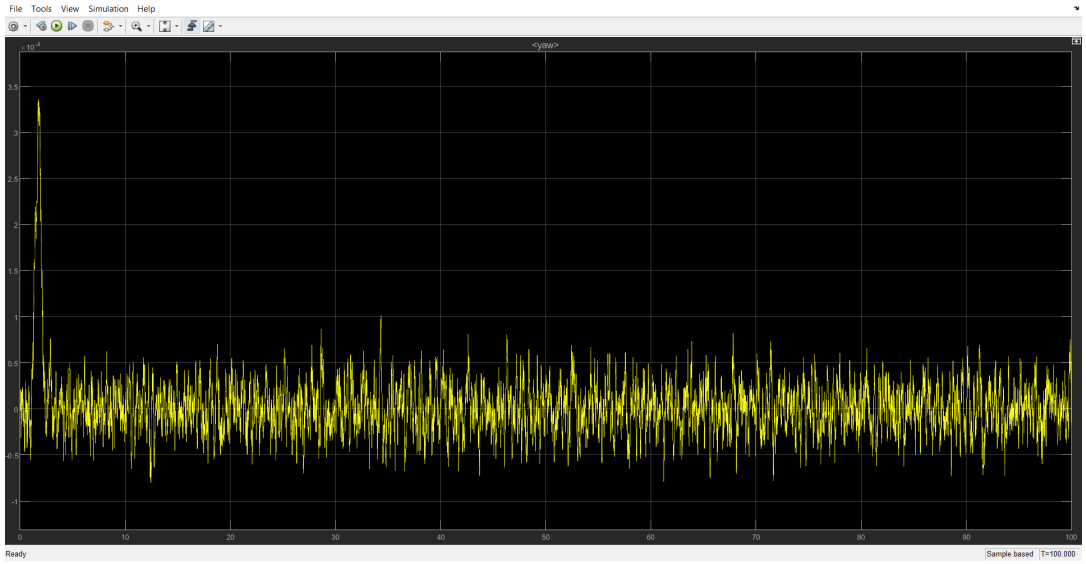


Figure 67: Tracciato Yaw

7 Conclusioni

Il controllore progettato, mediante l'ausilio della sintesi con il luogo delle radici, ha un comportamento quasi analogo al controllore di default, posto all'interno del nostro dispositivo. Il controllo mantiene delle oscillazioni sull'ordine di grandezza di 10^{-3} [m], che sono abbastanza contenute. Osservando i grafici ottenuti dall'analisi del modello lineare e del modello non lineare, possiamo notare che i modelli hanno un comportamento analogo.

Un leggero problema visibile dai grafici ottenuti, riguarda l'angolo di roll, che insieme all'angolo di pitch controlla la posizione. Possiamo notare e dire che, inizialmente esso assume valori variabili e discostanti tra loro per poi ottenere un assestamento dei propri valori.

Infine possiamo dire che il tracciato Z è uguale in tutte le analisi eseguite, ed è dovuto alla spinta dei motori, che, come si osserva, forniscono al drone una spinta maggiore all'inizio che gli consente di prendere quota e stabilizzarsi, sia in altitudine che per i motori ad un valore costante.

Nel complesso possiamo affermare che il sistema di controllo realizzato si è mostrato in grado di soddisfare le specifiche richieste e di ottenere un comportamento paragonabile al controllore PID implementato nel modello originale.

8 Bibliografia

1. https://it.mathworks.com/help/aeroblks/quadcopter-project.html
2. Isidori A. (1993), Sistemi di controllo, Siderea
3. http://wpage.unina.it/framato/materiale
4. S. Bouabdallah Design and Control of Quadrotors with Application to Autonomous Flying, Lausanne, 2007