Università Politecnica delle Marche

Dipartimento di Ingegneria dell'Informazione

Corso di Laurea Magistrale in Ingegneria Informatica e dell'Automazione

---

Stima del guasto di attuazione su un drone quadrirotore

Corso di Manutenzione Preventiva per la Robotica e l'Automazione Intelligente

Professore

Prof. Alessandro Freddi
Prof. Alessandro Baldini

Gruppo H1

Michele Pasqualini
Denil Nicolosi
Eris Prifti

Anno accademico 2022-2023

Indice

1 Modello

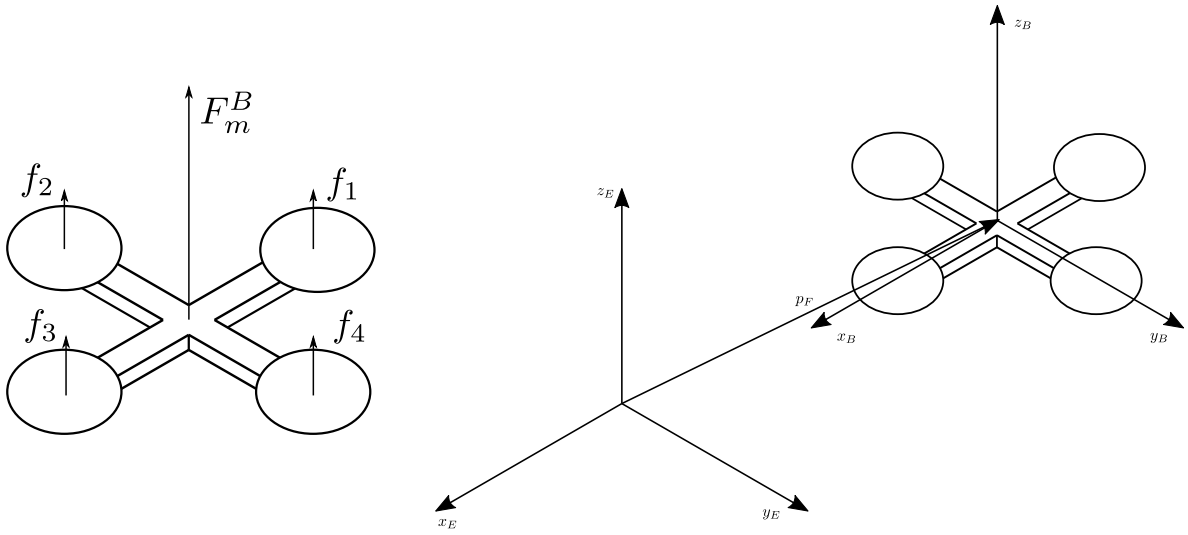

Figura 1: Modello fisico

L'intero sistema è approssimato come un unico corpo rigido. Si considerano quindi due terne di riferimento, denominate R_E e R_B. La terna R_E è considerata earth-fixed, e si assume sia inerziale, mentre la terna R_B è solidale al corpo (body-fixed), centrata al centro di massa del frame (i.e. del corpo centrale).

1.1 Parametri

Si denota con m la massa totale del corpo, con k_t il coefficiente di attrito lineare, con k_r il coefficiente di attrito angolare, con g l'accelerazione gravitazionale, con $J = \text{diag}(J_x, J_y, J_z)$ il tensore di inerzia. Le grandezze sono riportate nella tabella sottostante, dove tutte le unità di misura sono espresse nel sistema internazionale.

Parametro	**Valore**	**Unità di misura**
Massa totale (m)	3.95	kg
Coefficiente di attrito lineare (k_t)	0	$N * s/m$
Coefficiente di attrito angolare (k_r)	0	$N * s * m$
Inerzia lungo x_B (J_x)	0.363	$kg * m^2$
Inerzia lungo y_B (J_y)	0.363	$kg * m^2$
Inerzia lungo z_B (J_z)	0.651	$kg * m^2$
Accelerazione gravitazionale (g)	9.81	m/s^2
Lunghezza braccio (l)	0.45	m
Coefficiente di lift (c_L)	$3.13 * 10^{-5}$	$N * s^2$
Coefficiente di drag (c_D)	$7.5 * 10^{-7}$	$N * m * s^2$

1.2 Variabili

Si denomina con $p_F = \text{col}(x_F\ ,\ y_F\ ,\ z_F\ )$ la posizione del frame rispetto alla terna R_E, con $\eta = \text{col}(\varphi, \theta, \psi)$ il vettore contenente gli angoli di roll, pitch e yaw che esprimono la rotazione che R_B ha rispetto a R_E, con $\omega = \text{col}(p, q, r)$ la velocità angolare del corpo. Eventuali forze esterne (espresse in R_E) sono denominate con F_e^E.

1.3 Ingressi di controllo

Si assume di poter controllare direttamente le forze sprigionate dai motori. Esse sono dunque ingressi di controllo, raggruppati nel vettore $u = \text{col}(u_1, u_2, u_3, u_4)$. Denominando con F_m^B e M_m^B la forza totale ed il momento totale agenti sul frame e dovuti ai motori (espressa in R_B), si considerano le equazioni

$$\begin{aligned} F_m^B &= F_1 u \\ M_m^B &= F_2 u \end{aligned} \tag{1}$$

dove

$$F_1 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix} \; F_2 = \begin{bmatrix} 0 & l & 0 & -l \\ -l & 0 & l & 0 \\ -\frac{c_D}{c_L} & \frac{c_D}{c_L} & -\frac{c_D}{c_L} & \frac{c_D}{c_L} \end{bmatrix} \tag{2}$$

I parametri c_L e c_D sono detti coefficiente di lift (o thrust) e coefficiente di drag, mentre l è la distanza tra il centro di massa del sistema ed i motori, detto braccio.

1.4 Vettori canonici e matrici trasposte

I vettori della base canonica di $\mathbb{R}^3$ sono indicati con e_1, e_2, e_3, ovvero

$$e_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \; e_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \; e_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{3}$$

Se A è una matrice/vettore, allora A^T denota la sua trasposta.

1.5 Guasti

I guasti sono modellati come disturbi additivi, ovvero la forza $\Delta F_m^B(t) \in \mathbb{R}^3$ ed il momento $\Delta M_m^B(t) \in \mathbb{R}^3$.

1.6 Modello quadrotor

Sotto la nomenclatura descritta, il modello del quadrotor che si adotta è

$$m\ddot{p}_F = -k_t \dot{p}_F - mge_3 + R_B^E(R_m^B + \Delta F_m^B) + F_e \tag{4}$$

$$J\dot{\omega} = -k_r \omega - \omega \wedge J\omega + M_m^B + \Delta M_m^B \tag{5}$$

$$\dot{\eta} = T(\eta)\omega \tag{6}$$

dove

$$\begin{aligned} R_B^E &= \begin{bmatrix} cos(\psi)cos(\theta) & cos(\psi)sen(\varphi)sen(\theta) - cos(\varphi)sen(\psi) & sen(\varphi)sen(\psi) + cos(\varphi)cos(\psi)sen(\theta) \\ cos(\theta)sen(\psi) & cos(\varphi)cos(\psi) + sen(\varphi)sen(\psi)sen(\theta) & cos(\varphi)sen(\psi)sen(\theta) - cos(\psi)sen(\varphi) \\ -sen(\theta) & cos(\theta)sen(\varphi) & cos(\varphi)cos(\theta) \end{bmatrix} \\ T(\eta) &= \begin{bmatrix} 1 & sen(\varphi)tan(\theta) & cos(\varphi)tan(\theta) \\ 0 & cos(\varphi) & -sen(\varphi) \\ 0 & sen(\varphi)/cos(\theta) & cos(\varphi)/cos(\theta) \end{bmatrix} \end{aligned} \tag{7}$$

1.6.1 Dinamica Traslazionale

L'equazione della dinamica traslazionale è stata manipolata come viene riportato di seguito.

$$m\ddot{p}_F = -k_t\dot{p}_F - mge_3 + R_B^E(R_m^B + \Delta F_m^B) + F_e \tag{8}$$

Considerando il parametro k_t pari a zero, il primo termine dell'equazione viene annullato. Isolando $\ddot{p}_F$ si ottiene l'equazione seguente.

$$\ddot{p}_F = \frac{1}{m}\begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} + \frac{1}{m}R_B^E F_1 u + \frac{1}{m}R_B^E \Delta F_m^B + \frac{1}{m}F_e \tag{9}$$

Dopo aver calcolato $R_B^E F_1 u$, considerando i disturbi e le forze esterne nulle, viene ottenuto il medesimo risultato.

$$\begin{bmatrix} \ddot{x_F} \\ \ddot{y_F} \\ \ddot{z_F} \end{bmatrix} = \begin{bmatrix} \frac{1}{m}(sin\varphi sin\psi + cos\varphi cos\psi sin\theta)(u_1+u_2+u_3+u_4) \\ \frac{1}{m}(cos\varphi sin\psi sin\theta - cos\psi sin\varphi)(u_1+u_2+u_3+u_4) \\ \frac{1}{m}(cos\varphi cos\theta)(u_1+u_2+u_3+u_4) - g \end{bmatrix} \tag{10}$$

1.6.2 Dinamica Rotazionale

L'equazione sulla dinamica rotazionale è stata manipolata come viene riportato di seguito

$$J\dot{\omega} = -k_r\omega - \omega \wedge J\omega + M_m^B + \Delta M_m^B \tag{11}$$

Considerando il parametro k_r pari a zero, il primo termine dell'equazione viene annullato.

$$J\dot{\omega} = -\omega \wedge J\omega + F_2 u + \Delta M_m^B \tag{12}$$

Andando ad eseguire il prodotto vettoriale $-\omega \wedge J\omega$ e considerando inizialmente nulli i disturbi, il risultato ottenuto è il seguente.

$$\begin{bmatrix} \ddot{\varphi} \\ \ddot{\theta} \\ \ddot{\psi} \end{bmatrix} = \begin{bmatrix} -\frac{1}{J_x}[\dot{\theta}\dot{\psi}(J_y - J_z] + l(u_2+u_4) \\ \frac{1}{J_y}[\dot{\varphi}\dot{\psi}(J_x - J_z] - l(u_1-u_3) \\ -\frac{1}{J_z}[\dot{\varphi}\dot{\theta}(J_x - J_y] - 0.024(u_1+u_3-u_2-u_4) \end{bmatrix} \tag{13}$$

1.6.3 Cinematica Rotazionale

L'equazione sulla cinematica rotazionale viene descritta di seguito ed è stato manipolata direttamente all'interno del modello in Simulink.

$$\dot{\eta} = T(\eta)\omega \tag{14}$$

2 Controllore

Per incominciare, viene preso in considerazione un controllore inner/outer loop come progettato nell'articolo di riferimento [1]. Prestare attenzione alla nomenclatura, che differisce da quella del presente file (l'ultima è da preferire).

2.1 Inner loop

L'inner loop prende in ingresso i riferimenti in altezza $z_{F_r}(t)$ e di assetto $\eta(t)$=col$(\varphi_r(t), \theta_r(t), \psi_r(t))$, e genera di conseguenza l'ingresso di controllo $u(t)$ affinché le variabili di stato $z_F, \varphi, \theta, \psi$ seguano i riferimenti (con

ovvia nomenclatura). Data la struttura F_1, si ha $F_1 u = f_z e_3$ dove $f_z = \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix} u$.
Considerando che $z_F = e_3^T p_F$, il modello del quadrotor ristretto all'inner loop è

$$m\ddot{z}_F = -k_t \dot{z}_F - mg + r_{33} f_z + E_3^T \Delta F_m^B + E_3^T \Delta F_e^E \tag{15}$$

$$J\dot{\omega} = -k_r \omega - \omega \wedge J\omega + M_m^B + \Delta M_m^B \tag{16}$$

$$\dot{\eta} = T(\eta)\omega \tag{17}$$

dove $r_{33} = E_3^T R_B^E e_3$. La legge di controllo che si prende in considerazione è quindi

$$u = F^{-1} A^{-1} \left(\begin{bmatrix} -b_z \\ -b_\eta \end{bmatrix} + \begin{bmatrix} v_z \\ v_\eta \end{bmatrix} \right) \tag{18}$$

dove:

$$\begin{aligned} b_z &= -\tfrac{k_t}{m} \dot{z}_F - g + \tfrac{1}{m} e_3^T F_e^E \\ b_\eta &= \dot{T}(\eta)\omega - k_r T(\eta) J^{-1} \omega - T(\eta) J^{-1} (\omega \wedge J\omega) + T(\eta) J^{-1} \Delta M_m^B \\ A^{-1} &= \begin{bmatrix} \tfrac{1}{m} r_{33} & 0 \\ 0 & T(\eta) J^{-1} \end{bmatrix}^{-1} \\ F^{-1} &= \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & l & 0 & -l \\ -l & 0 & l & 0 \\ -\frac{c_D}{c_L} & \frac{c_D}{c_L} & -\frac{c_D}{c_L} & \frac{c_D}{c_L} \end{bmatrix}^{-1} \end{aligned} \tag{19}$$

La derivata della matrice $T(\eta)$ viene calcolata andando a derivare ciascuna delle componenti della matrice rispetto agli angoli.

$$\dot{T}(\eta) = \begin{bmatrix} 0 & \begin{array}{l} cos(\varphi)tan(\theta)(\rho + r * cos(\varphi)tan(\theta) \\ +q * sin(\varphi)tan(\theta)) \\ +sin(\varphi)tan^2(\theta+1)(q * cos(\varphi) - r * sin(\varphi)) \end{array} & \begin{array}{l} -sin(\varphi)tan(theta) \\ (\rho + r * cos(\varphi)tan(\theta) + q * sin(\varphi)tan(\theta)) \\ +cos(\varphi)tan^2(\theta+1) \\ (q * cos(\varphi) - r * sin(\varphi)) \end{array} \\ 0 & \begin{array}{l} -sin(\varphi)(\rho + r * cos(\varphi)tan(\theta) \\ +q * sin(\varphi)tan(\theta)) \end{array} & \begin{array}{l} -cos(\varphi)(\rho + r * cos(\varphi)tan(\theta) \\ +q * sin(\varphi)tan(\theta)) \end{array} \\ 0 & \begin{array}{l} \frac{cos(\varphi)}{cos(\theta)}(\rho + r * cos(\varphi)tan(\theta) \\ +q * sin(\varphi)tan(\theta)) \\ +\frac{sin(\varphi)sin(\theta)}{cos^2(\theta)}(q * cos(\varphi) - r * sin(\varphi)) \end{array} & \begin{array}{l} \frac{-sin(\varphi)}{cos(\theta)}(\rho + r * cos(\varphi)tan(\theta) \\ +q * sin(\varphi)tan(\theta)) \\ +\frac{cos(\varphi)}{cos^2(\theta)} sin(\theta)(q * cos(\varphi) \\ -r * sin(\varphi)) \end{array} \end{bmatrix} \tag{20}$$

Le quantità $v_z(t) \in \mathbb{R}$ e $v_\eta(t) \in \mathbb{R}^3$ sono ingressi ausiliari, definiti come:

$$v_z = \ddot{z}_{Fr} - \alpha_{z1}(\dot{z}_F - \dot{z}_{Fr}) - \alpha_{z0}(z_F - z_{Fr}) \tag{21}$$

$$v_\eta = \ddot{\eta}_r - \alpha_{n1}(T(\eta)\omega - \dot{\eta}_r) - \alpha_{\eta 0}(\eta - \eta_r) \tag{22}$$

dove i coefficienti $\alpha_{z1}, \alpha_{z0} \in \mathbb{R}$ e $\alpha_{\eta 1}, \alpha_{\eta 0} \in \mathbb{R}^{3\times 3}$ piazzano gli autovalori del ciclo chiuso. La dinamica errore è infatti

$$\begin{aligned} \ddot{e}_z + \alpha_{z1}\dot{e}_z + \alpha_{z0} e_z = 0 \\ \ddot{e}_\eta + \alpha_{\eta 1}\dot{e}_\eta + \alpha_{\eta 0} e_\eta = 0 \end{aligned} \tag{23}$$

I coefficienti $\alpha_{\eta 1}, \alpha_{\eta 0}$ sono stati scelti in forma diagonale:

$$\alpha_{\eta 0} = \begin{bmatrix} 150 & 0 & 0 \\ 0 & 150 & 0 \\ 0 & 0 & 12 \end{bmatrix} \alpha_{\eta 1} = \begin{bmatrix} 35 & 0 & 0 \\ 0 & 35 & 0 \\ 0 & 0 & 7 \end{bmatrix} \tag{24}$$

Gli autovalori che ne derivano sono rispettivamente: [-30 -30 -3] e [-5 -5 -4]. Invece, gli autovalori piazzati dai coefficienti α_{z0}, α_{z1}, corrispondenti rispettivamente ai valori di 9 e 6, sono: -3 e -3. Tutti gli autovalori vengono scelti negativi in modo tale da garantire la stabilità del sistema.

2.2 Outer loop

L'outer loop (anello esterno) viene progettato sotto l'approssimazione per piccoli angoli, ovvero si considerano

$$\begin{aligned} \varphi(t) &\approx 0 \\ \theta(t) &\approx 0 \end{aligned} \tag{25}$$

Geometricamente parlando, queste due condizioni sono verificate quando il drone è inclinato di pochi gradi rispetto al suo piano (x_B, y_B) (appunto, piccoli angoli). Questa condizione è inoltre verificata in hovering (drone che è capace di restare fermo, orizzontalmente, in una posizione dello spazio), per cui è a volte detto che il drone è considerato vicino alla condizione di hovering (near hovering condition). Per un piccolo x $\approx$ 0 si hanno le approssimazioni (suggerite dallo sviluppo in serie di taylor) sin(x) $\approx$ x e cos(x) $\approx$ 1. Segue quindi che:

$$\begin{aligned} cos(\varphi(t)) \approx 1, sin(\varphi(t)) \approx \varphi(t), \\ cos(\theta(t)) \approx 1, sin(\theta(t)) \approx \theta(t), \end{aligned} \tag{26}$$

Conseguenza diretta dell'approssimazione è:

$$\begin{aligned} R_B^E F_m^B &= \begin{bmatrix} \star & \star & sen(\varphi)sen(\psi) + cos(\varphi)cos(\psi)sen(\theta) \\ \star & \star & cos(\varphi)sen(\psi)sen(\theta) - cos(\psi)sen(\varphi) \\ \star & \star & \star \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ \mathrm{f}_z \end{bmatrix} \\ &= f_z \begin{bmatrix} sen(\varphi)sen(\psi) + cos(\varphi)cos(\psi)sen(\theta) \\ cos(\varphi)sen(\psi)sen(\theta) - cos(\psi)sen(\varphi) \\ \star \end{bmatrix} \\ &\approx f_z \begin{bmatrix} \varphi sen(\psi) + cos(\psi)\theta \\ sen(\psi)\theta - cos(\psi)\varphi \\ \star \end{bmatrix} \\ &\approx f_z \begin{bmatrix} sen(\psi) & cos(\psi) \\ -cos(\psi) & sen(\psi) \\ \star & \star \end{bmatrix} \begin{bmatrix} \varphi \\ \theta \end{bmatrix} \end{aligned} \tag{27}$$

L'outer loop si progetta sulle equazioni dinamiche di $x_F(t)$ e $y_F(t)$, che si ricorda essere le prime due componenti di $p_F(t)$. Tali equazioni sono quindi:

$$m \begin{bmatrix} \ddot{x}_F \\ \ddot{y}_F \end{bmatrix} = -k_t \begin{bmatrix} \dot{x}_F \\ \dot{y}_F \end{bmatrix} + (f_z + \Delta f_z) \begin{bmatrix} sen(\psi) & cos(\psi) \\ -cos(\psi) & sen(\psi) \end{bmatrix} \begin{bmatrix} \varphi \\ \theta \end{bmatrix} + \begin{bmatrix} F_{e,x} \\ F_{e,y} \end{bmatrix} \tag{28}$$

dove si indicano con $F_e^E = col(F_{e,x}, F_{e,y}, F_{e,z})$ e $\Delta F_m^B = col(0, 0, \Delta f_z)$. Supponendo di poter controllare direttamente $\varphi(t)$ e $\theta(t)$ attraverso l'inner loop, si progetta una legge prendendo questi come ingressi. La

legge presa in considerazione è quindi:

$$\begin{bmatrix}\varphi\\ \theta\end{bmatrix} = \frac{m}{f_z+\Delta f_z}\begin{bmatrix}sen(\psi) & cos(\psi)\\ -cos(\psi) & sen(\psi)\end{bmatrix}^{-1}\left(\frac{k_t}{m}\begin{bmatrix}\dot{x}_F\\ \dot{y}_F\end{bmatrix} - \frac{1}{m}\begin{bmatrix}F_{e,x}\\ F_{e,y}\end{bmatrix} + \begin{bmatrix}v_x\\ v_y\end{bmatrix}\right) \tag{29}$$

dove $v_x(t), v_y(t) \in \mathbb{R}$ sono ingressi ausiliari. Nella realtà non siamo in grado di comandare $\varphi(t), \theta(t)$ direttamente, quindi i valori ottenuti sono in realtà dei riferimenti per inner loop. Inoltre non conosciamo i valori del guasto $\Delta f_z(t)$ e delle componenti del disturbo esterno $F_{e,x}(t), F_{e,y}(t)$, quindi dovremo accontentarci di stime, ovvero $\hat{\Delta} f_z(t), \hat{F}_{e,x}(t), \hat{F}_{e,y}(t)$. La legge implementabile è quindi

$$\begin{bmatrix}\varphi_r\\ \theta_r\end{bmatrix} = \frac{m}{f_z+\hat{\Delta} f_z}\begin{bmatrix}sen(\psi) & cos(\psi)\\ -cos(\psi) & sen(\psi)\end{bmatrix}^{-1}\left(\frac{k_t}{m}\begin{bmatrix}\dot{x}_F\\ \dot{y}_F\end{bmatrix} - \frac{1}{m}\begin{bmatrix}\hat{F}_{e,x}\\ \hat{F}_{e,y}\end{bmatrix} + \begin{bmatrix}v_x\\ v_y\end{bmatrix}\right) \tag{30}$$

Sotto l'approssimazione per piccoli angoli, una perfetta conoscenza dei disturbi e guasti (i.e. $\hat{\Delta} f_z = \Delta f_z, \hat{F}_{e,x} = F_{e,x}, \hat{F}_{e,y} = F_{e,y}$) ed una perfetta convergenza dell'inner loop ($\varphi(t) = \varphi_r(t)$ e, $\theta(t) = \theta_r(t)$), viene ricavata la dinamica di $x_F(t)$ e $y_F(t)$ in funzione di $v_x(t)$ e $v_y(t)$:

$$m\begin{bmatrix}\ddot{x}_F\\ \ddot{y}_F\end{bmatrix} = -k_t\begin{bmatrix}\dot{x}_F\\ \dot{y}_F\end{bmatrix} + (f_z+\Delta f_z)\begin{bmatrix}sen(\psi) & cos(\psi)\\ -cos(\psi) & sen(\psi)\end{bmatrix}\left(\frac{m}{f_z+\Delta f_z}\begin{bmatrix}sen(\psi) & cos(\psi)\\ -cos(\psi) & sen(\psi)\end{bmatrix}^{-1}\left(\frac{k_t}{m}\begin{bmatrix}\dot{x}_F\\ \dot{y}_F\end{bmatrix} - \frac{1}{m}\begin{bmatrix}\hat{F}_{e,x}\\ \hat{F}_{e,y}\end{bmatrix} + \begin{bmatrix}v_x\\ v_y\end{bmatrix}\right)\right) + \begin{bmatrix}F_{e,x}\\ F_{e,y}\end{bmatrix}$$

$$m\begin{bmatrix}\ddot{x}_F\\ \ddot{y}_F\end{bmatrix} = -k_t\begin{bmatrix}\dot{x}_F\\ \dot{y}_F\end{bmatrix} + k_t\begin{bmatrix}\dot{x}_F\\ \dot{y}_F\end{bmatrix} - \begin{bmatrix}F_{e,x}\\ F_{e,y}\end{bmatrix} + m\begin{bmatrix}v_x\\ v_y\end{bmatrix} + \begin{bmatrix}F_{e,x}\\ F_{e,y}\end{bmatrix} \tag{31}$$

$$\begin{bmatrix}\ddot{x}_F\\ \ddot{y}_F\end{bmatrix} = \begin{bmatrix}v_x\\ v_y\end{bmatrix} \tag{32}$$

Il termine v_x controlla completamente la dinamica di x_F, mentre il termine v_y controlla completamente la dinamica di y_F. Ricordiamo che v_x e v_y sono ingressi ausiliari, ovvero ingressi di appoggio per sintetizzare il controllore. Quindi, si hanno due sistemi indipendenti/disaccoppiati. Come fatto già per l'Inner Loop, scriveremo v_x e v_y che piazzano gli autovalori. Sono stati progettati i termini $v_x(t)$ e $v_y(t)$ in maniera tale che le variabili di errore $e_x(t) = x_F(t) - x_{Fr}(t)$ e $e_y(t) = y_F(t) - y_{Fr}(t)$ convergano a zero. Indicando con $x_{Fr}(t)$ il riferimento per $x_F(t)$ e $y_{Fr}(t)$ il riferimento per $y_F(t)$, porremo:

$$v_x = \ddot{x}_{Fr} - \alpha_{x1}(\dot{x}_F - \dot{x}_{Fr}) - \alpha_{x0}(x_F - x_{Fr}) \tag{33}$$

$$v_y = \ddot{y}_{Fr} - \alpha_{y1}(\dot{y}_F - \dot{y}_{Fr}) - \alpha_{y0}(y_F - y_{Fr}) \tag{34}$$

3 Fault Detection

Si vuole progettare un generatore di residuo (basato su osservatore) per risolvere il problema della fault detection. In parole povere, si vuol costruire un sistema

$$\dot{z} = F(z, x, u) \tag{35}$$

$$r = H(z, x, y) \tag{36}$$

tale per cui $r(t) \in R$ mostra un andamento convergente a zero in assenza di faults, mentre diverge da zero in presenza di faults. La soluzione al problema è immediata. Supponiamo il modello del sistema in esame sia della forma

$$\dot{x} = f(x) + \sum_{i=1}^{m} g_i(x)u_i + \sum_{i=1}^{q} w_i p_i(x) \tag{37}$$

dove x rappresenta il vettore di stato, le u_i sono gli ingressi di controllo e le w_i i faults. Quando non ci sono faults, si ha $w_i = 0$ per ogni i, mentre la presenza di un fault genera $w_j \neq 0$ per qualche j. Il generatore di residuo è quindi una replica del sistema, dove (ovviamente) i fault non sono considerati poichè incogniti, a cui viene aggiunto un termine stabilizzante. In altre parole, la soluzione è banalmente

$$\dot{z} = f(x) + \sum_{i=1}^{n} g_i(x)u_i + L(x - z) \tag{38}$$

$$r = ||x - z|| \tag{39}$$

4 Simulazione

4.1 Implementazione generale

La simulazione verrà effettuata tramite Matlab e Simulink, grazie ai quali la programmazione del controllore e del processo risulta essere abbastanza semplice. Inoltre, sono presenti blocchi come lo scope, il delay e alcuni blocchi integrativi che consentono di regolare e analizzare diversi parametri. La durata della simulazione è stata impostata a 20.0 secondi simulati. Inizialmente, sono stati definiti tutti i parametri descritti nel paragrafo 1.1 all'interno de file `Parametri_Drone.m`. Il primo blocco implementato in Simulink è stato il plant del quadrotor. Successivamente si è passati all'implementazione dell'inner loop e, in un secondo momento, dell'outer loop. Infine, lo step finale ha riguardato la progettazione di un generatore di residuo, basato su osservatore, per risolvere il problema della fault detection. Tutti i plot che verranno mostrati nella parte 5 sono stati simulati con il file `Plot.m`. Il modello completo è rappresentato dallo schema a blocchi nella figura sottostante.

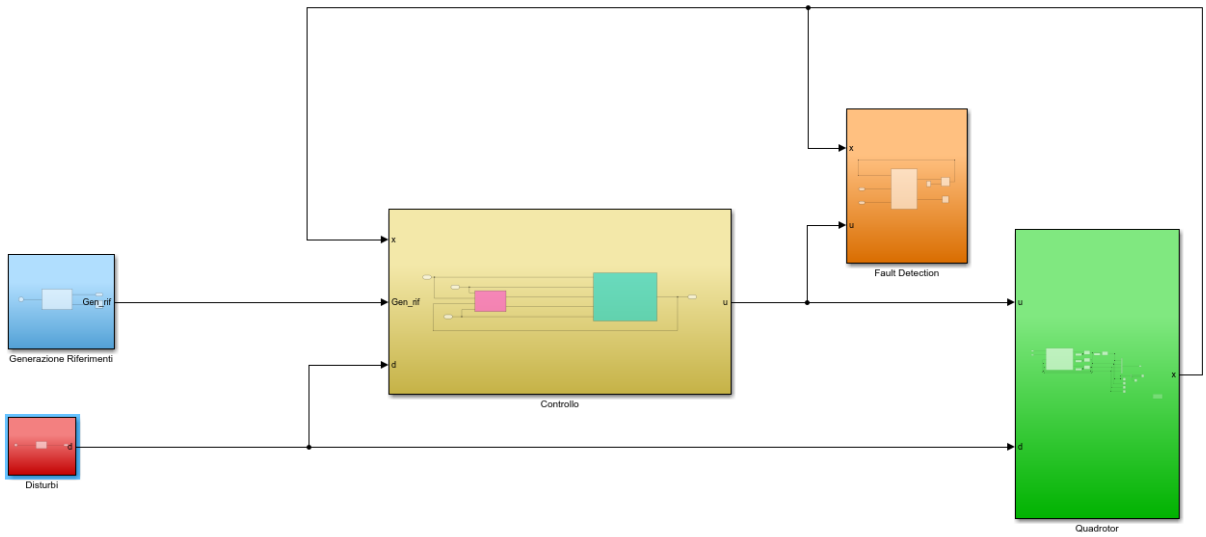

Figura 2: Schema a blocchi generale

4.2 Implementazione del quadrotor

L'implementazione del modello viene effettuata tramite Matlab e Simulink. In particolare, nello schema a blocchi viene utilizzata una Matlab Function che permette di calcolare le equazioni del drone. Tali equazioni corrispondono a quelle scritte precedentemente nella parte introduttiva e fanno riferimento alle equazioni

4, 5 e 6. La Matlab Function, denominata *Drone*, prende in input una serie di ingressi dallo schema e dei parametri che vengono passati da un file Matlab. Tra gli ingressi troviamo gli ingressi di controllo, i guasti modellati come disturbi additivi ed eventuali forze esterne. La funzione restituirà in output $\ddot{p}_F$, $\dot{\omega}$ e $\dot{\eta}$. Inserendo degli opportuni blocchi integrativi con delle condizioni iniziali, vengono ricavati i valori di p_F, ω e η. Gli output sono stati dati in pasto ad un blocco "Demux" che genera un unico vettore di stato il quale contiene la posizione, la velocità, la velocità angolare e gli angoli del drone ($x(p_F,v,\omega,\eta)$).

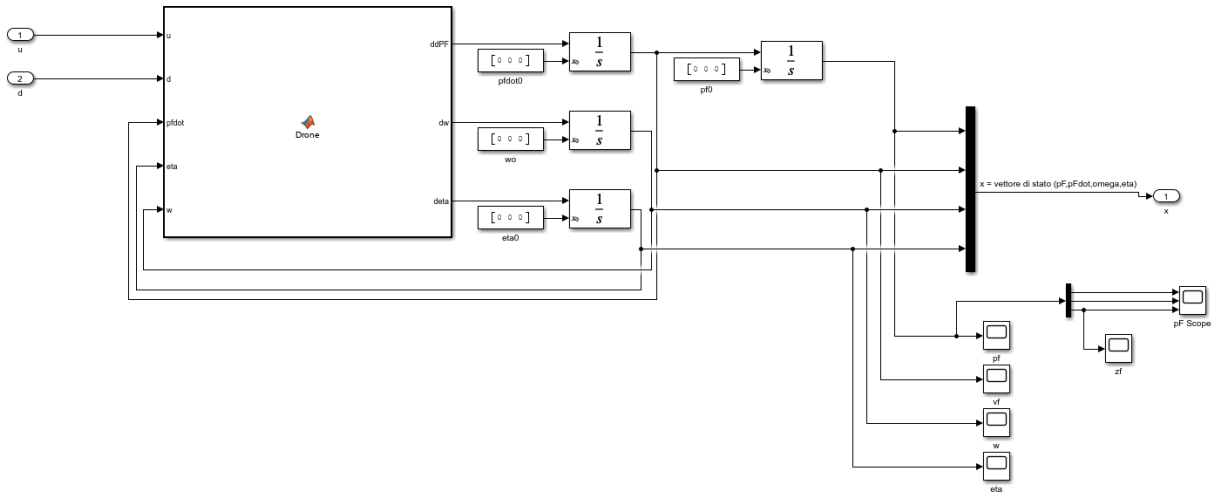

Figura 3: Plant del quadrotor

4.3 Implementazione dell'Inner Loop

L'inner loop è stato realizzato in una matlab function, che riceve in input i seguenti parametri: il vettore di stato (x), i riferimenti generati (Gen_rif), i riferimenti per gli angoli φ_r e θ_r e il vettore dei disturbi (d). In output si ottiene il vettore u di dimensioni 4x1 che rappresenta le forze dei motori. All'interno del blocco inner loop vengono posizionati anche alcuni scope che permettono di visualizzare gli errori tra il valore attuale e il valore di riferimento per le variabili z, φ, θ e ψ.

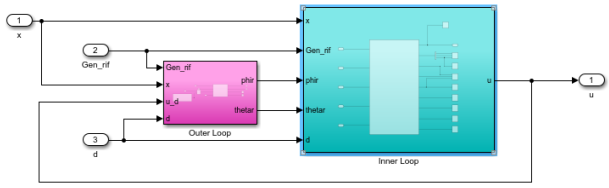

Figura 4: Schema generale del controllore

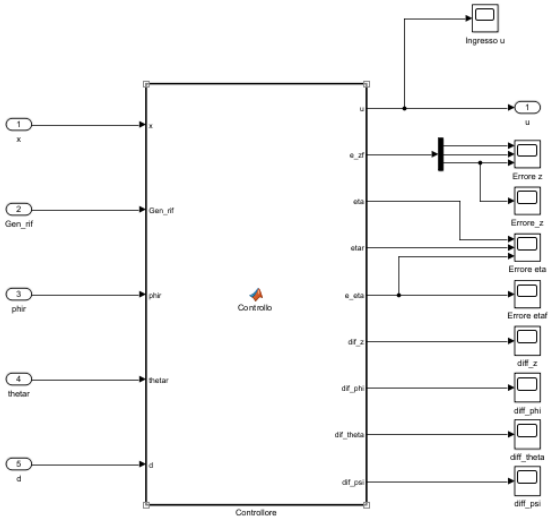

Figura 5: Inner Loop

4.4 Implementazione dell'Outer Loop

L'implementazione dell'outer viene eseguita per mezzo di una matlab function che prende in ingresso i riferimenti tempo varianti (Gen_rif), il vettore di stato (x), l'ingresso di controllo (u) in output dall'inner loop e il vettore dei disturbi (d). L'outer loop genera riferimenti tempo varianti per gli angoli φ_r, θ_r. Inoltre, la Matlab function dell'outer loop, mostra in output, attraverso due scope, l'errore in x e y.

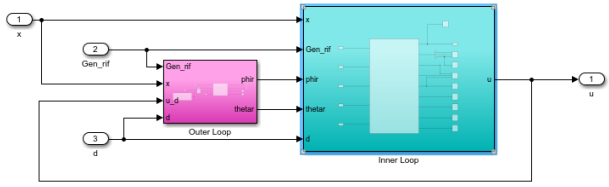

Figura 6: Schema generale del controllore

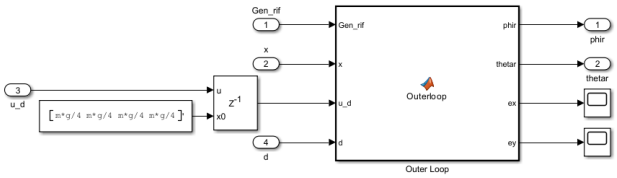

Figura 7: Outer Loop

4.5 Implementazione dei disturbi

L'implementazione dei disturbi viene fatta per mezzo di una Matlab function dove all'interno vengono definiti i valori di ogni componente vettoriale dei disturbi ΔM_m^B, ΔF_m^B e F_e. Viene impostata una condizione per cui nei primi 10 secondi della simulazione, i disturbi sono nulli, mentre nei successivi istanti di tempo, essi assumono un valore costante generalmente piccolo.

Figura 8: Schema a blocchi dei disturbi

4.6 Implementazione della fault detection

L'implementazione dell'ultima funzione, ovvero del rilevamento di un fault, viene fatta attraverso una Matlab function che prende in ingresso il vettore di stato (x), l'ingresso di controllo (u) e la componente $\dot{z}$ riferita all'equazione 38, il quale viene ricavata attraverso un blocco integrativo. All'interno della Matlab function vengono replicate esattamente le equazioni del nostro sistema 4 e 5, prive delle componenti relative ai disturbi e con la matrice $L(x - z)$. La matrice L viene scelta in forma diagonale. Quindi, in output, avremo il generatore di residuo (r) che segnalerà la presenza di un fault quando il valore di r supera una certa soglia.

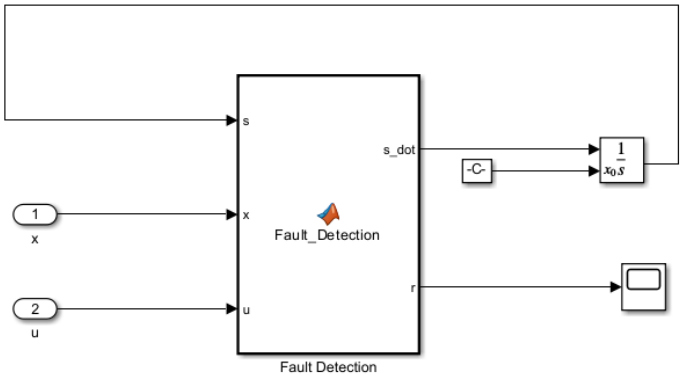

Figura 9: Schema a blocchi dell'osservatore

5 Risultati della simulazione

In questo paragrafo vengono mostrati i risultati ottenuti andando a implementare differenti tipi di traiettorie:

- nella prima traiettoria si hanno i segnali di riferimento costanti e nulli;
- nella seconda traiettoria si hanno i segnali di riferimento costanti e non nulli;
- nella terza traiettoria si hanno i segnali di riferimento che descrivono una circonferenza sul piano x e y mentre z è nullo insieme a ψ;
- nella quarta traiettoria si hanno i segnali di riferimento che descrivono una circonferenza sul piano x e y, con una rampa su z e ψ è costante

Vengono considerate sempre tre classi di disturbi: disturbi nulli nei primi istanti di tempo della simulazione fino a $t = 10$, disturbi lentamente variabili (con funzione $\frac{t+sin(t)}{5}$) nella parte centrale della simulazione con $10 \leq t \leq 15$ e disturbi costanti nella parte finale della simulazione con $t \geq 15$.

5.1 Riferimento costante nullo (hoovering ideale)

Nel primo caso viene testata una traiettoria del drone in cui i riferimenti generati hanno un valore costante nullo (condizione di hovering ideale). Nei grafici sottostanti si possono osservare i risultati che vengono ottenuti per la posizione finale e l'errore negli angoli. Dal grafico tridimensionale della posizione si può vedere la traiettoria seguita dal drone. Dato che in questo grafico non viene mostrato il tempo t, bisogna considerare che fino a $t \leq 10$ il drone si trovava costantemente alla posizione $(0, 0, 0)$, e che lo spostamento visualizzato si riferisce quindi al movimento compiuto durante l'introduzione del disturbo ($t \geq 10$).

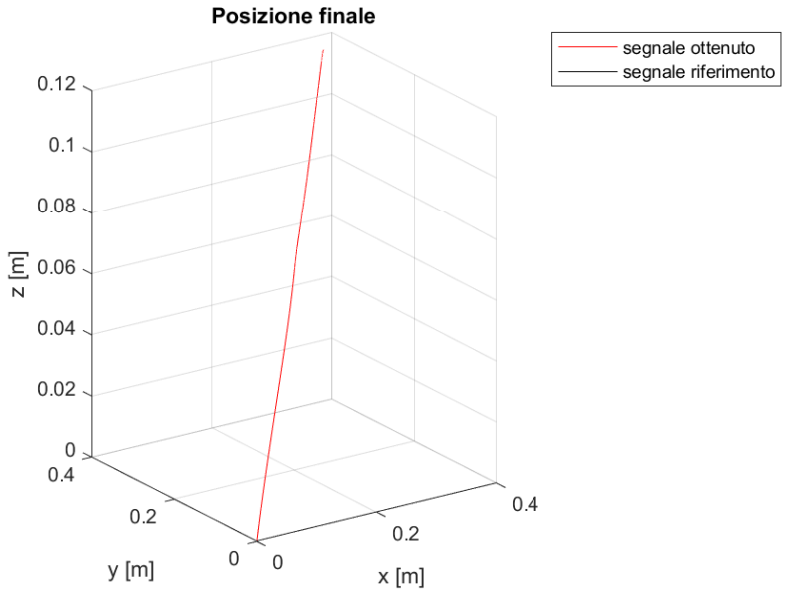

Figura 10: Posizione

Vediamo come l'errore sugli angoli viene influenzato dai disturbi, in particolare ψ rimane costante sullo zero in quanto stiamo testando in una condizione di hovering ideale. Mentre θ e ψ subiscono delle oscillazioni in prossimità di quando si presentano i disturbi.

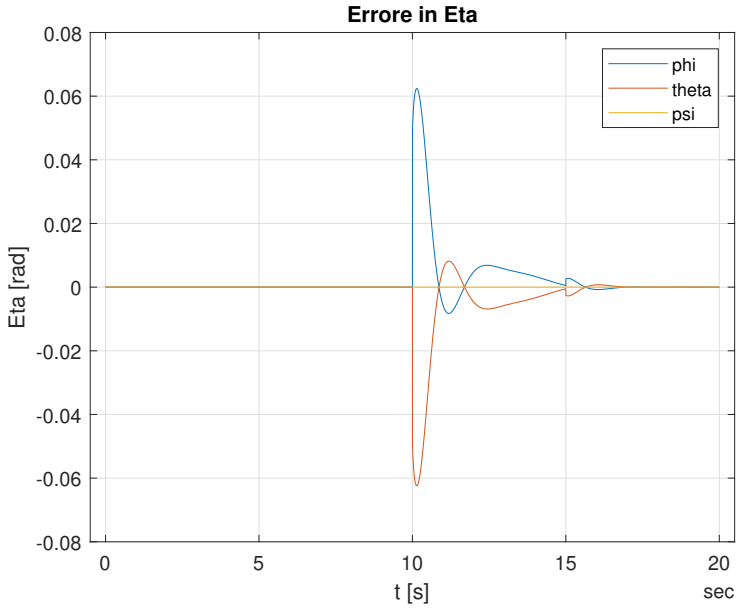

Figura 11: Errore in eta

5.2 Riferimento costante non nullo (hoovering)

Nel secondo caso viene testata una traiettoria del drone in cui i riferimenti generati hanno un valore costante non nullo (hovering). Nei grafici sottostanti si possono osservare i risultati che vengono ottenuti per la posizione finale e l'errore negli angoli. Dal grafico tridimensionale della posizione si può vedere la traiettoria seguita dal drone. Dato che in questo grafico non viene mostrato il tempo t, bisogna considerare che fino a $t \leq 10$ il drone si trovava costantemente alla posizione $(0, 0, 0)$, e che lo spostamento visualizzato si riferisce quindi al movimento compiuto durante l'introduzione del disturbo ($t \geq 10$). Nella parte superiore della traiettoria si nota un ulteriore cambio di posizione, dovuto all'assestamento del disturbo a un valore costante per $t \geq 15$.

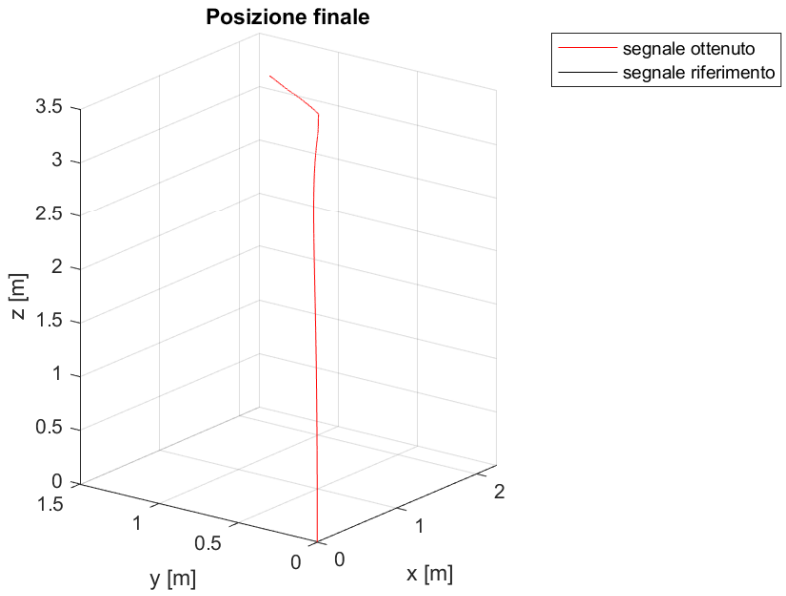

Figura 12: Posizione

Analizzando il grafico dell'errore in η, inizialmente, vediamo come gli angoli partono dai riferimenti dati per poi assestarsi su un valore costante (e quindi con errore prossimo allo zero). L'errore sugli angoli subisce una variazione oscillatoria a causa dei disturbi all'istante $t = 10$, per poi assestarsi nuovamente su un valore costante tendente allo zero.

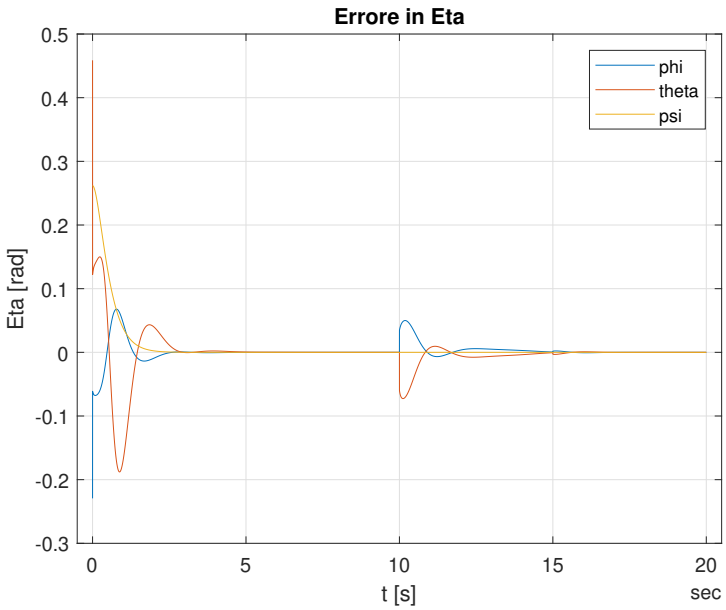

Figura 13: Errore in eta

5.3 Riferimento con circonferenza x-y e ψ nullo

Nel terzo caso si ha una generazione del segnale di riferimento leggermente più particolare rispetto ai primi due perché si va a rappresentare la traiettoria descritta da una circonferenza. Nei grafici sottostanti si possono osservare i risultati che vengono ottenuti per la posizione finale e l'errore negli angoli. Dal grafico tridimensionale della posizione si può vedere la traiettoria seguita dal drone. Nella prima metà della simulazione (fino a $t \leq 10$) il drone ha seguito il riferimento della circonferenza, invece dall'introduzione del disturbo per $t \geq 10$ la posizione ottenuta subisce delle variazioni, soprattutto nell'asse z, cercando di seguire comunque la forma di una circonferenza.

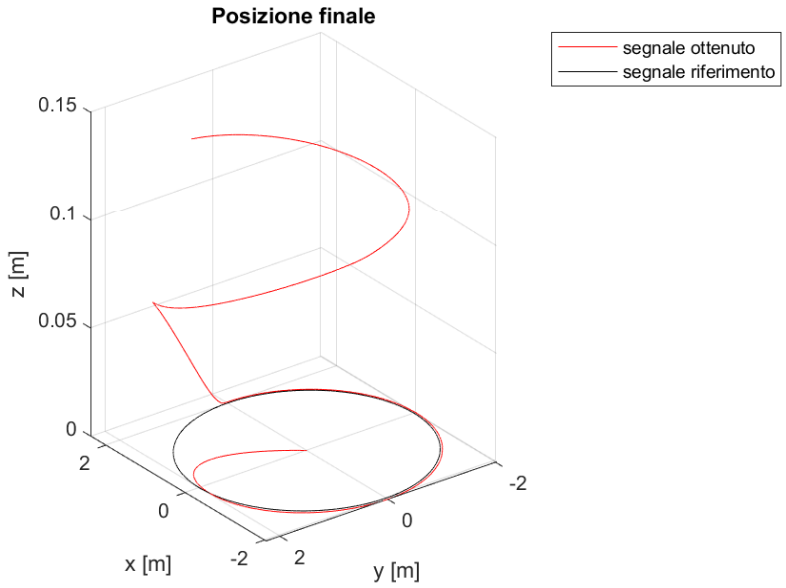

Figura 14: Posizione

Analizzando il grafico dell'errore in η, inizialmente, vediamo come gli angoli partono dai riferimenti dati per poi assestarsi su un valore costante (e quindi con errore prossimo allo zero). L'errore sugli angoli subisce una variazione oscillatoria a causa dei disturbi all'istante $t = 10$, per poi assestarsi nuovamente su un valore costante tendente allo zero.

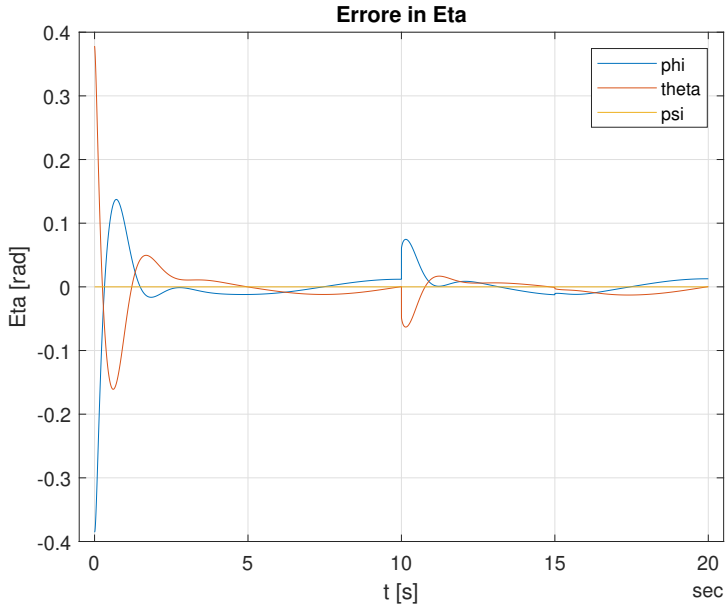

Figura 15: Errore in eta

5.4 Riferimento con circonferenza x-y, rampa su z e ψ costante

Nel quarto caso, la traiettoria del drone è descritta da una circonferenza nel piano x-y, con una rampa su z e ψ costante. Essendo un caso di studio particolare vengono riportati un numero di grafici maggiori rispetto agli altri casi, con un focus particolare sul residuo per la stima dei guasti. In rosso avremo il segnale generato dalla simulazione e in nero il segnale di riferimento. Inoltre, vengono riportati tre grafici che mostrano la posizione del drone sugli assi x, y e z evidenziando il valore ottenuto e il valore di riferimento. Vediamo come nel grafico tridimensionale, il segnale ottenuto segue molto bene la traiettoria di riferimento del drone durante la simulazione, fino ad un punto in cui si discosta per compensare l'effetto del disturbo. Nei tre grafici (x, y, z) si può notare che lo scostamento della posizione inizia per $t \geq 10$.

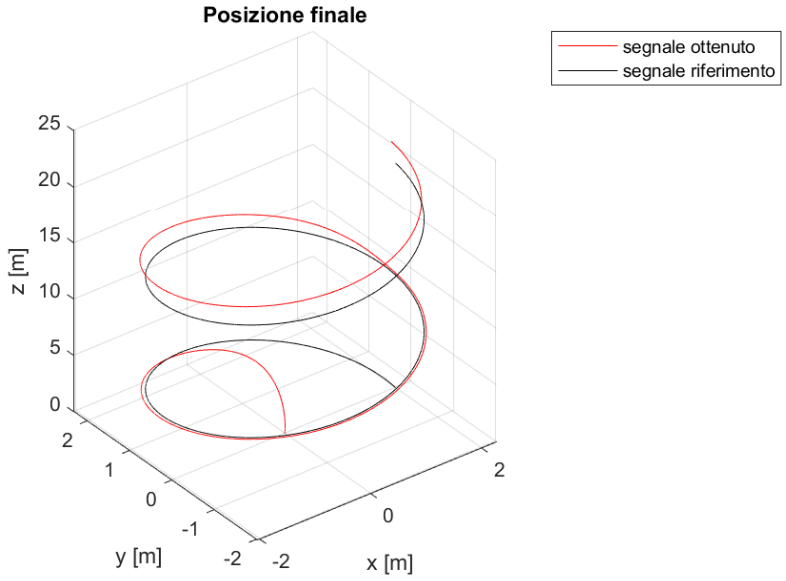

Figura 16: Traiettoria ottenuta del drone e di riferimento

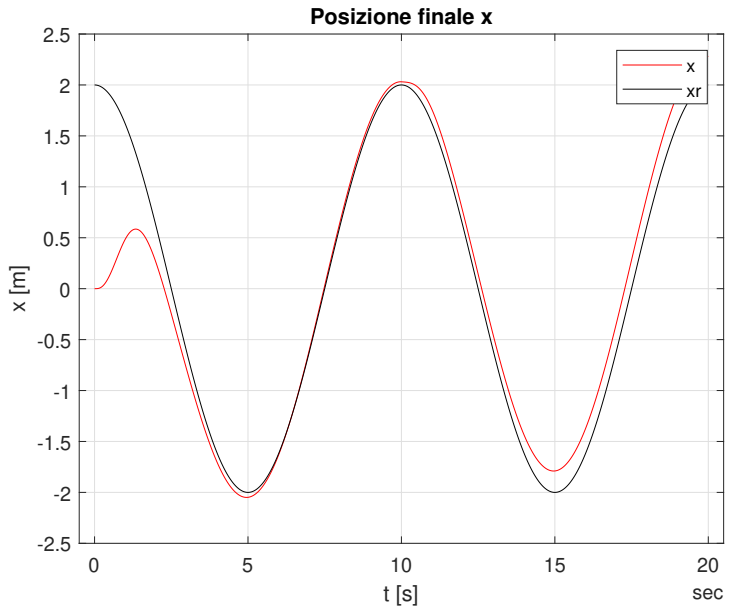

Figura 17: Posizione nell'asse X del drone e il riferimento

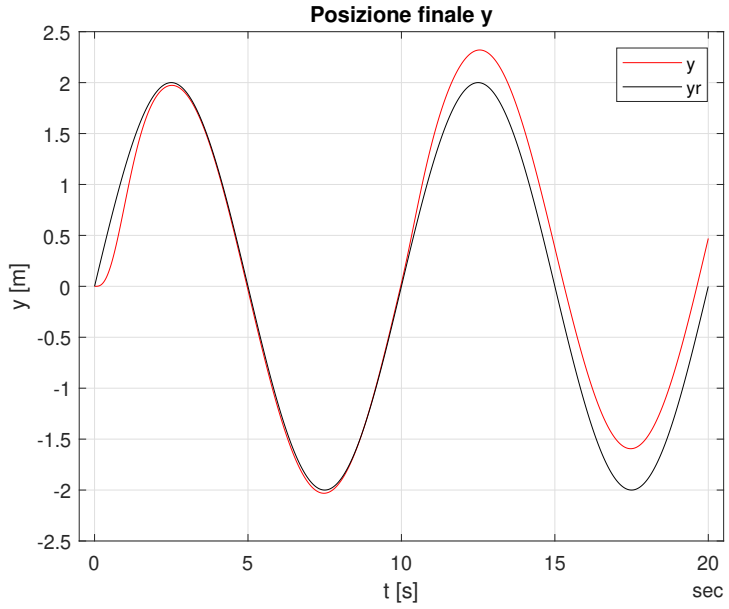

Figura 18: Posizione nell'asse Y del drone e il riferimento

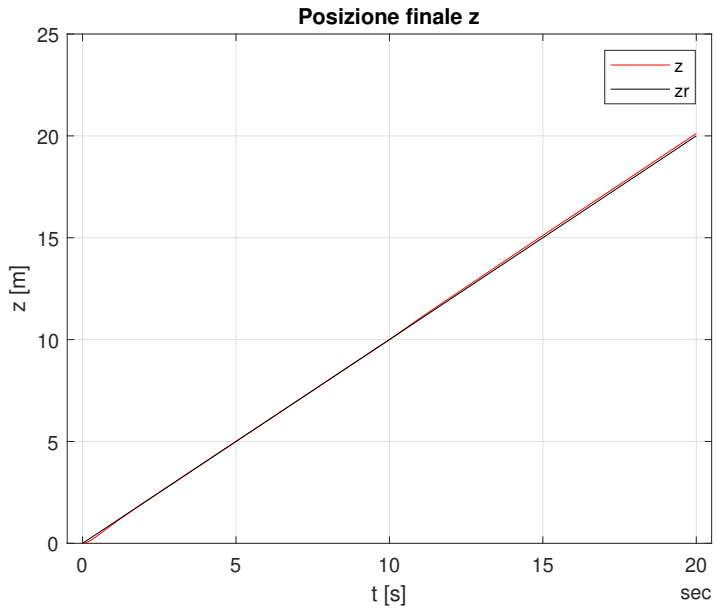

Figura 19: Posizione nell'asse Z del drone e il riferimento

Nei grafici successivi vengono riportate gli scostamenti tra i segnali ottenuti e di riferimento per gli angoli θ, ψ e ϕ. Anche qui si può notare che all'istante $t \geq 10$ aumenta la differenza con il riferimento, data dall'introduzione dei disturbi.

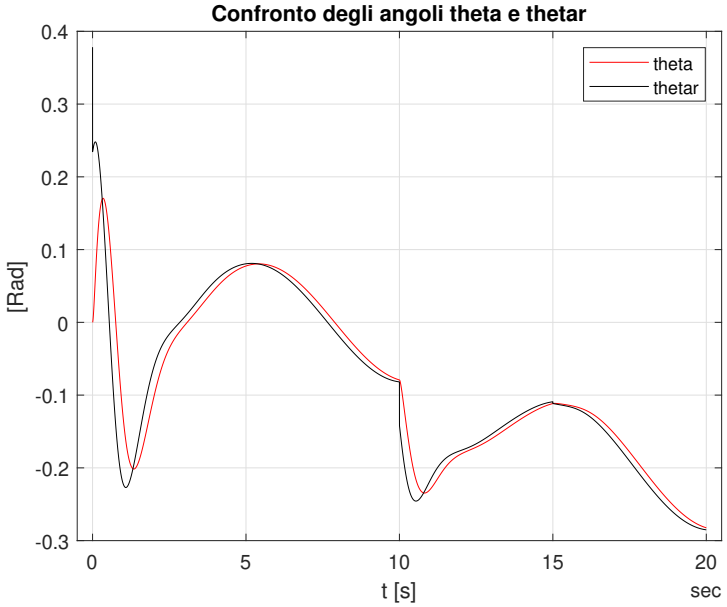

Figura 20: Angolo theta del drone e il riferimento

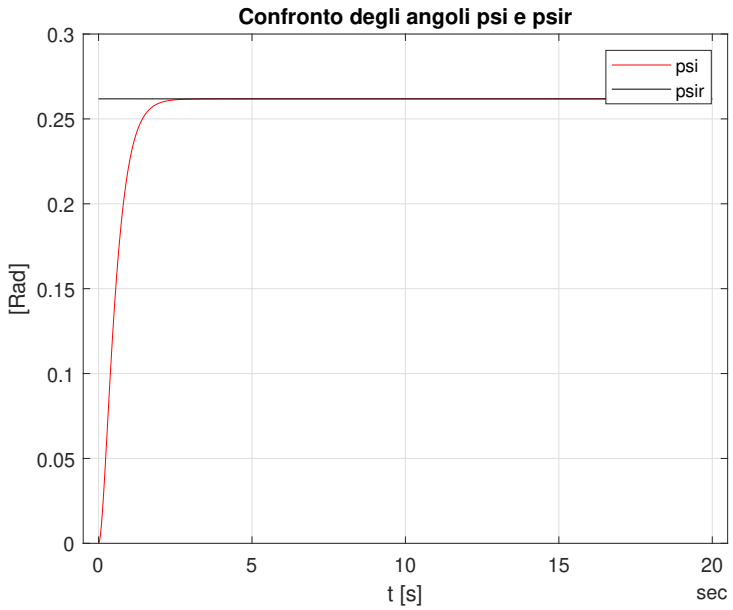

Figura 21: Angolo psi del drone e il riferimento

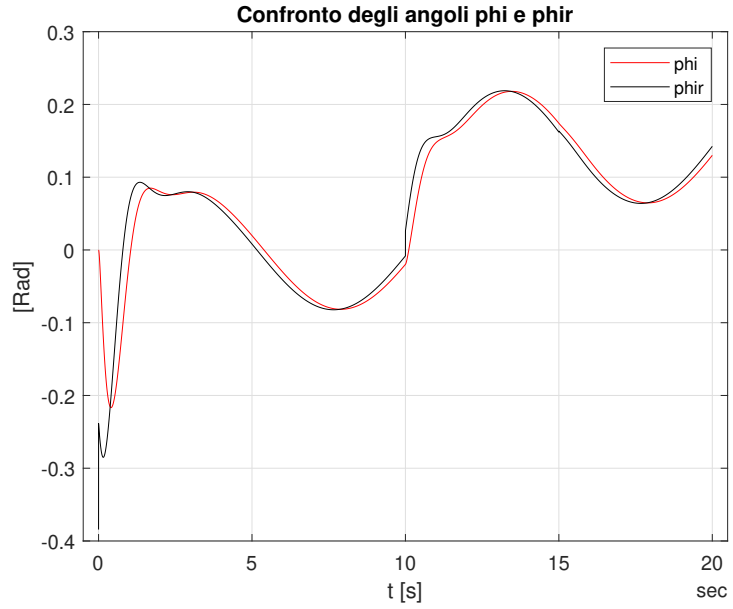

Figura 22: Angolo phi del drone e il riferimento

Analizzando il grafico dell'errore in η, inizialmente, vediamo come gli angoli partono dai riferimenti dati per poi assestarsi su un valore costante (e quindi con errore prossimo allo zero). L'errore sugli angoli subisce una variazione oscillatoria a causa dei disturbi all'istante $t = 10$, per poi assestarsi nuovamente su un valore costante tendente allo zero.

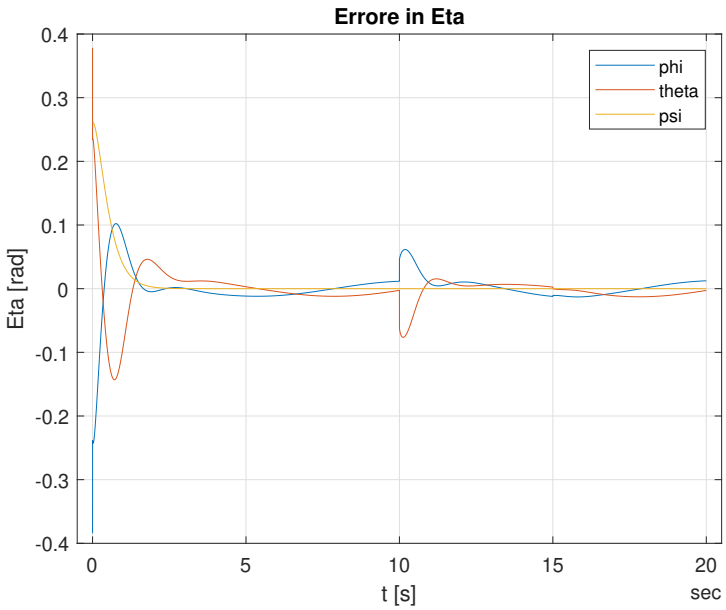

Figura 23: Errore in eta

Nel grafico sottostante vengono riportati gli ingressi di controllo che rappresentano le forze dei motori del quadrotore. Vediamo che nei primi istanti della simulazione, l'ingresso $u = (u_1, u_2, u_3, u_4)$ si assesta su valore costante, ma all'istante in cui si presentano i disturbi ($t = 10$) subisce una variazione dividendo le forze a due a due.

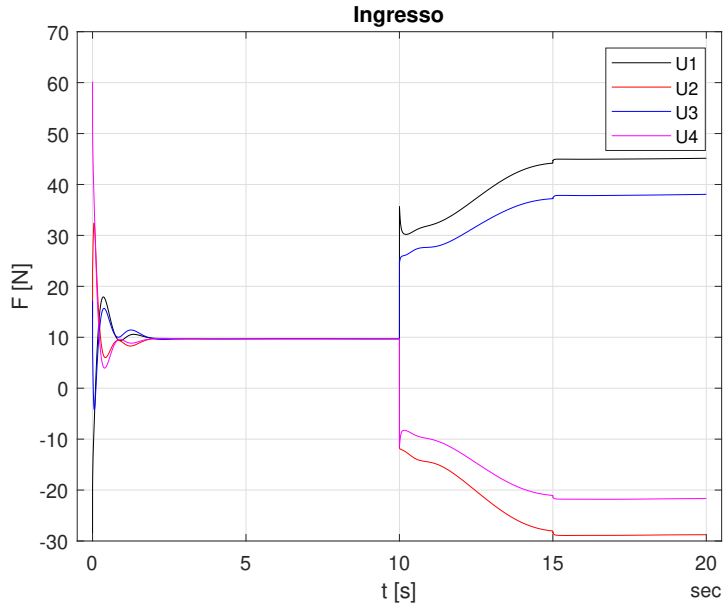

Figura 24: Ingressi di controllo dei motori del drone

Il generatore di residuo viene testato per la classe di guasti lentamente variabili, ovvero partono da zero e si assestano su un valore non nullo. Per vedere se la tecnica riesce a generare i residui, nei primi 10 secondi

della simulazione vengono mantenuti i disturbi nulli, quindi il residuo tende a zero. Vengono poi considerati disturbi lentamente variabili per $10 \leq t \leq 15$ e disturbi costanti nella parte finale della simulazione con $t \geq 15$.

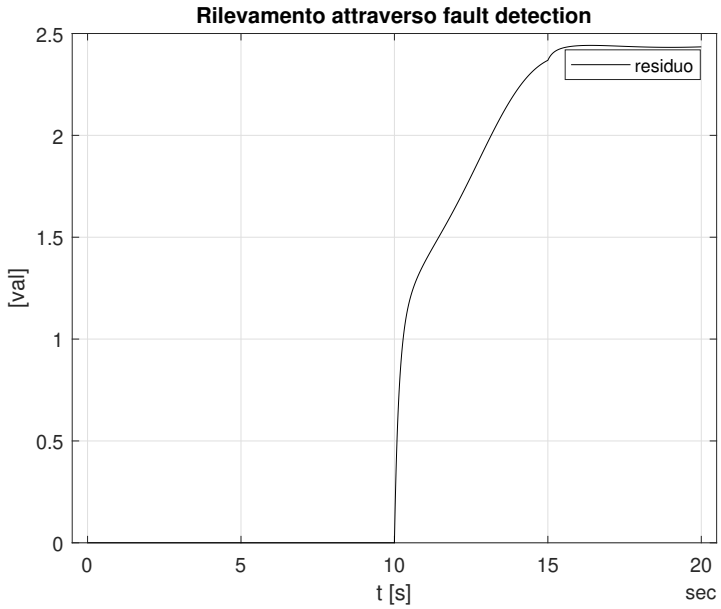

Figura 25: Grafico del residuo

6 Conclusioni

In conclusione si può dire che lo scopo iniziale di andare a stimare un guasto di attuazione su un drone quadrirotore è stato raggiunto. Lo studio, partendo dalla modellazione matematica di tutti i componenti, ci ha permesso di capire come realizzare ogni parte del sistema e di generare un residuo per fare rilevamento guasti. Come si può vedere dall'ultima figura 25 il generatore di residuo basato su osservatore funziona e svolge un buon lavoro, è in grado di segnalare la presenza di fault quando i disturbi non sono nulli. Per quanto riguarda i futuri sviluppi, si potrebbero testare diverse tecniche per il task di fault detection, testare una tipologia di controllore diverso oppure programmare una sequenza di punti da far seguire al drone, tracciando in questo modo una traiettoria più complessa volta ad evitare eventuali ostacoli.

Riferimenti bibliografici

[1] [Baldini, Alessandro, et al.] *"Fault-tolerant disturbance observer based control for altitude and attitude tracking of a quadrotor." 2018 26th Mediterranean Conference on Control and Automation (MED). IEEE, 2018.*