UNIVERSITÀ DEGLI STUDI DEL SANNIO

Facoltà di Ingegneria

Corso di Laurea Specialistica in Ingegneria dell'Automazione

Tesi di Laurea

PROGETTAZIONE E SVILUPPO DI UN FILTRO DI KALMAN ESTESO PER LA STIMA D'ASSETTO DI UN MICRO UAV

Relatore:
Prof. LUIGI IANNELLI

Laureando:
GIUSEPPE GIORDANO 391/139

Correlatore:
Dott.ssa DANIELA MEOLA

ANNO ACCADEMICO 2011-2012

Se ho visto più lontano
è perché stavo sulle spalle di giganti.
Isaac Newton

Indice

Acronimi

UAV	Unmanned Aerial Vehicle
IMU	Inertial Measurement Unit
GPS	Global Positioning System
GCS	Ground Control Station
ECI	Earth-Centered Inertial Coordinate System
ECEF	Earth-Centered Earth-Fixed Coordinate System
NED	Nord-East-Down Coordinate System
WGS-84	World Geodetic System 1984
FCS	Flight Control System
SAS	Stability Augmentation System
CAS	Control Augmentation System
ACS	Automatic Control System
RGA	Relative Gain Array
MIMO	Multi-Input Multi-Output
LQR	Linear Quadratic Regulator
LOS	Line Of Sight
AHRS	Attitude Heading Reference System
INS	Inertial Navigation System
WMM	World Magnetic Model
MEMS	Micro Electro Mechanical System
EKF	Extended Kalman Filter
SIL	Software in the loop
PIL	Processor in the loop
UKF	Unscented Kalman Filter
TRIAD	Three Axis Attitude Determination

Sommario

La presente tesi si colloca nell'ambito di un progetto nato dalla collaborazione tra l'Università degli Studi del Sannio, l'Università del Minnesota e l'Università di Budapest, nel campo dei velivoli privi di pilota (Unmanned Aerial Vehicle).

In particolare, il contributo di questa tesi è stato focalizzato sull'implementazione e validazione di un filtro di Kalman per la stima dell'assetto di un velivolo.

Nell'ambito di un sistema di controllo del volo, per eseguire correttamente gli algoritmi di navigazione autonoma, le strutture di controllo hanno bisogno di diverse informazioni che, in genere, sono fornite dall'unità di misurazione inerziale posta a bordo del velivolo. Non sempre, però, tutte le informazioni sono direttamente accessibili dalla strumentazione di bordo poiché l'utilizzo di tutte le tipologie di sensori potrebbe risultare proibitivo sia in termini di peso che di costo. Per tale motivo occorre fare ricorso alla tecnica della *sensor fusion* per ricavare le informazioni necessarie per la navigazione che non sono fornite da nessun sensore specifico, quale ad esempio l'assetto del velivolo.

L'*Attitude and Heading Reference System* è tradizionalmente considerato come soluzione al problema della stima dell'assetto. Esso combina ed elabora, tramite un Filtro di Kalman, le misure provenienti dai sensori disponibili al fine di ricavare le informazioni desiderate, nell'ipotesi di condizioni di volo stazionarie per il velivolo in esame. Dopo aver considerato e discusso tale soluzione, il lavoro di tesi è stato incentrato sull'analisi ed il miglioramento di tale stimatore in modo da fornire risultati soddisfacenti anche in condizioni dinamiche di volo.

Uno studio dello stato dell'arte ha condotto alla realizzazione di una soluzione alternativa al problema della stima dell'assetto: essa utilizza il filtro di Kalman esteso unitamente alle informazioni provenienti da magnetometri e GPS, limitando al minimo l'utilizzo degli accelerometri, che costituiscono la causa principale delle problematiche riscontrate nella soluzione standard. Questi ultimi, infatti, sono stati utilizzati solo in caso di perdita del segnale GPS. La soluzione

così realizzata è di tipo multi-mode e permette anche la stima di polarizzazioni lentamente variabili sui giroscopi.

L'algoritmo di stima così sintetizzato è stato implementato prima in ambiente Matlab/Simulink e poi in linguaggio C per una validazione *Software in the loop*. Diversi scenari di testing hanno permesso di confrontare le prestazioni della nuova soluzione rispetto alla soluzione standard, evidenziando le migliorie apportate.

Introduzione

Negli ultimi decenni è molto cresciuto l'interesse nei confronti della progettazione e dello sviluppo di piccoli velivoli sprovvisti di pilota (*Unmanned Aerial Vehicle* - UAV) che in maniera autonoma possano effettuare missioni sia militari che civili, quali inseguimento di obiettivi mobili e non, monitoraggio ambientale, del traffico e della sicurezza nelle città. Gli UAVs sono sistemi integrati relativamente complessi, in grado di spostarsi in modo indipendente in un ambiente specifico, equipaggiati con strumentazione di bordo per la navigazione e la trasmissione dei dati e che, per questo, hanno suscitato interesse anche nell'ambito della ricerca scientifica.

Il seguente lavoro di tesi è parte di un progetto di ricerca nato alcuni anni fa dalla collaborazione tra l'Università degli Studi del Sannio, l'Università del Minnesota e l'Università di Budapest, e focalizzato sullo sviluppo di un sistema di navigazione, di strategie di guida e di controllo per il volo autonomo in ambienti indoor e outdoor. L'obiettivo della ricerca, però, non è solo indirizzato agli aspetti tecnologici ma, soprattutto, alla riduzione dei costi e alla condivisione delle conoscenze di design e sviluppo di hardware e software all'insegna della filosofia open source [1].

Questo lavoro poggia le sue basi su precedenti lavori di tesi [2], [3], [4] e [5]. In particolare in [2] e [3] sono stati affrontati i tasks relativi alla realizzazione di un prototipo, lo *Yardstik*, e alla scelta e integrazione dell'avionica di bordo, per la navigazione e la trasmissione dei dati. In [4] è stato implementato un sistema di controllo per il volo autonomo (*Flight Control System*) che è stato poi validato con simulazioni *Software-In-the-Loop* (SIL) al fine di verificare che il velivolo sia in grado di portare a termine una missione assegnatagli, nel rispetto delle specifiche definite. Infine in [5] le medesime simulazioni sono state condotte in un ambiente di simulazione *Hardware-In-the-Loop* (HIL), dove gli algoritmi di controllo sono

stati codificati su un microprocessore ed eseguiti in real-time, chiudendo il ciclo con il modello di simulazione.

Stato dell'arte del progetto

Il progetto è configurato in modo tale da soddisfare alcuni requisiti fondamentali, tra cui quello più importante è rappresentato dalla flessibilità del sistema, che deve essere facilmente riconfigurabile per essere in grado di assicurare, senza grosse difficoltà, gli stessi risultati su differenti tipologie di velivoli. Un secondo requisito è quello economico, relativo al contenimento dei costi, che permette di orientarsi verso un sistema realizzato con componenti low cost. Inoltre tutte le conoscenze ed i risultati ottenuti vengono condivisi all'insegna della filosofia Open Source.

La Fig. 1 mostra l'architettura complessiva dell'UAV finora sviluppato.

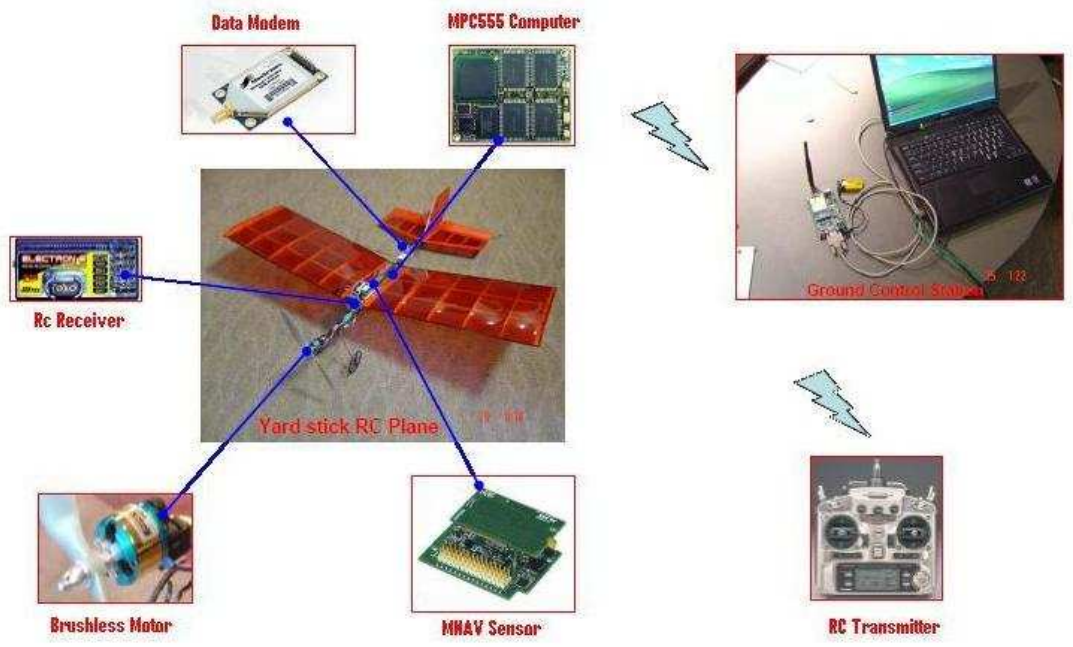


Figura 1: Architettura complessiva dello Yardstik

L'airframe utilizzato è lo Yardstick [6], un aeromodello a motore elettrico adatto al volo nei parchi. Esso rientra nella categoria dei micro UAV, ovvero velivoli privi di pilota caratterizzati dai seguenti parametri [7]:

- raggio operativo minore di 10 km;
- quota di volo minore di 250 m;
- durata di volo minore di 1 ora;
- peso massimo al decollo minore di 5 kg.

Durante il volo dello Yardstick, i segnali inviati da terra dalla trasmittente vengono decodificati da una ricevente a bordo dell'aeromodello e vengono inviati ai servo meccanismi [8], tramite gli opportuni canali, per attuare il movimento desiderato.

L'avionica di bordo comprende tutti i sistemi elettronici realizzati per essere usati su un aeromodello, quali:

- una unità di misura inerziale (*Inertial Measurement Unit* - IMU) che include anche misure di posizione e velocità provenienti dal GPS [9];
- un processore Motorola MPC555 PowerPC, della Freescale semiconductor [10], che riceve informazioni dai sensori, tramite comunicazione seriale, ed esegue algoritmi di navigazione e di controllo per la generazione dei comandi da inviare agli attuatori, per il controllo dell'elevatore, del timone e della spinta del motore;
- un modem per la comunicazione con una stazione di terra (*Ground Control Station* - GCS) per l'acquisizione in real-time dei dati provenienti dal velivolo [11].

In Fig. 2 è mostrato un diagramma funzionale dell'avionica usata per il velivolo.

Il progetto, che prevede la realizzazione di un velivolo autonomo, per via della sua complessità è stato suddiviso in unità di lavoro più piccole, per una più semplice gestione dello stesso e per una migliore comprensione e gestione dei problemi che via via sono sorti o che sorgeranno.

Tasks finora realizzati comprendono:

- assemblaggio dell'aeromodello;
- integrazione dei componenti avionici;
- realizzazione del sistema di acquisizione dei dati;
- implementazione prototipale di una GCS;

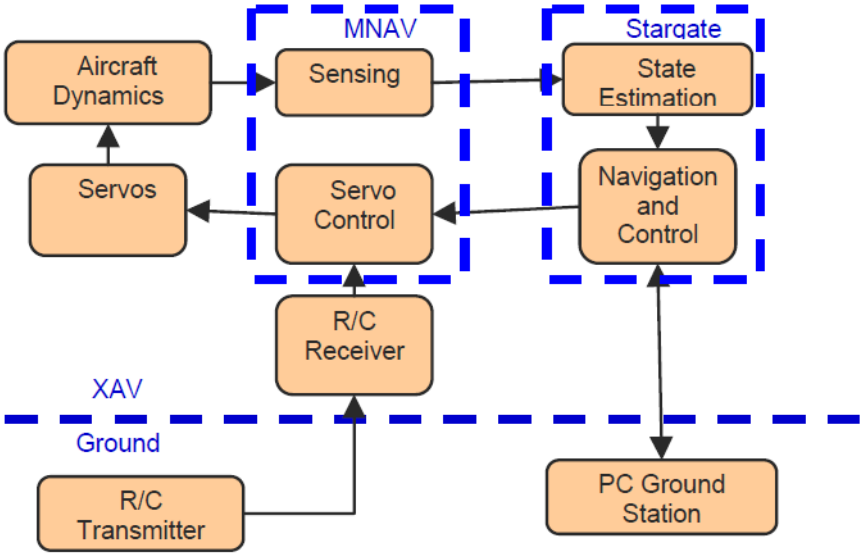


Figura 2: Schematizzazione dell'avionica a bordo

- studio del processore e del Sistema Operativo;

- progettazione e validazione del sistema di controllo di volo (FCS);

di cui si può leggere in dettaglio in [2], [3], [4] e [5].

Contributi della tesi

L'ultimo task realizzato, ovvero la progettazione del sistema di controllo di volo, richiede che siano note le grandezze che descrivono l'assetto del velivolo durante il volo. Queste grandezze però, a causa del requisito relativo al contenimento dei costi ma anche per ragioni strutturali di realizzazione, non sono direttamente accessibili dai sensori a disposizione, ma possono essere solo stimate a partire dalle altre grandezze misurate. Il software a cui compete il compito di stimare l'assetto è chiamato *Attitude and Heading Reference System* (AHRS).

In [4], dovendo realizzare un sistema di controllo di volo completo, è stata implementata una prima versione dell'AHRS basandosi sulla struttura proposta in [12]. Alla base dell'AHRS vi è un osservatore di stato ovvero un oggetto matematico che, a partire da un set di misure e di equazioni matematiche, è in grado di fornire una stima delle grandezze interessate. L'osservatore utilizzato è il Filtro di Kalman Esteso (EKF). Tale struttura è stata ripresa e discussa anche

nel presente lavoro. I test realizzati hanno dimostrato però che, a causa di alcune problematiche che verranno presentate nel corso della trattazione, la performance di tale stimatore non è soddisfacente, soprattutto in condizioni dinamiche di volo.

Nel presente lavoro di tesi l'attenzione è stata focalizzata, dunque, sulla ricerca di nuove soluzioni al problema della stima d'assetto e sulla realizzazione di un nuovo stimatore che migliori le prestazioni.

La struttura del lavoro è la seguente:

- il Capitolo 1 presenta il modello matematico non lineare del velivolo. Le equazioni che governano il moto sono derivate applicando le leggi di Newton per le forze ed i momenti, in sistemi di riferimento noti, e sono equazioni fortenemente non lineari per la presenza dei coefficienti aerodinamici. Viene inoltre presentato il sistema di controllo del volo realizzato in [4]. Un algoritmo di navigazione con linea di vista migliorata permette di tracciare in tempo reale la direzione di volo che il velivolo deve inseguire al fine di realizzare la missione assegnatagli. Algoritmi di controllo nel piano longitudinale e latero-direzionale cooperano per il controllo dell'assetto al fine di mantenere il percorso di volo specificato;

- il Capitolo 2 presenta la soluzione per l'AHRS standard utilizzata in [4]. I segnali provenienti dai sensori inerziali presenti nell'avionica di bordo (in particolare giroscopi, magnetometri e accelerometri) vengono utilizzati per descrivere l'assetto del velivolo e stimare polarizzazioni costanti sui giroscopi;

- il Capitolo 3 analizza in generale la problematica della stima dell'assetto ed espone i problemi relativi all'utilizzo dello stimatore standard presentato al Capitolo 2. A partire dall'analisi di tali problematiche viene presentata una soluzione multi-mode per la stima d'assetto, che utilizza un diverso set di misure (in particolare il GPS) e modifica la struttura generale del filtro.

- il Capitolo 4 è dedicato al testing e alla validazione dell'algoritmo di stima multi-mode nonché al confronto del nuovo stimatore con la soluzione AHRS standard finora utilizzata. Tutte le soluzioni sono, inoltre, state implementate in codice C per permettere una validazione *Software-In-The-Loop* (SIL).

Capitolo 1

Modello e controllo dell'UAV

Il presente capitolo è dedicato alla descrizione del sistema UAV. Vengono prima descritti i sistemi di riferimento, le grandezze utilizzate per rappresentare l'assetto del velivolo e vengono introdotte le equazioni del moto che compongono il modello matematico di un velivolo a sei gradi di libertà, che si muove su una piccola area di superficie terrestre. Infine, viene presentato il sistema di controllo del volo precedentemente sviluppato in [4].

1.1 Sistemi di riferimento

Al fine di esprimere lo stato di un velivolo e le relative equazioni dinamiche è necessario definire un sistema di riferimento. Esistono, in realtà, diversi sistemi di riferimento definiti nell'ambito della meccanica del volo e quelli di interesse per la derivazione delle equazioni sono i seguenti:

- *Inerziale* o *ECI* (Earth-centered inertial) con origine geocentrica e orientato rispetto alle stelle fisse. L'asse Z_i è orientato secondo l'asse di rotazione terrestre e passa per il polo nord geografico, l'asse X_i è orientato secondo la direzione dell'equinozio di primavera e l'asse Y_i, invece, va orientato in modo da rendere la terna levogira.

- *Terrestre* o *ECEF* (Earth-centered Earth-fixed) avente origine ed asse Z_e coincidenti con il precedente, mentre l'asse X_e passa per il meridiano di Greenwich e l'asse Y_e è tale da rendere la terna levogira. Assumendo tra-

scurabile il moto di rivoluzione terrestre, questo sistema di assi non risulta inerziale solo a causa del moto di rotazione[1].

- *Verticale-locale* o *NED* (Nord-East-Down) è assunto con l'origine fissa al baricentro del velivolo e l'asse Z_v orientato verso il basso secondo la direzione della verticale locale. Gli assi X_v e Y_v sono tali da individuare un piano normale all'asse Z_v, con X_v orientato verso il Nord geografico e Y_v orientato verso Est in modo da rendere la terna levogira. Questa terna segue il velivolo nel moto intorno alla Terra mantenendo invariato il suo orientamento.

- *Assi corpo* con origine nel baricentro del velivolo, l'asse X_b coincidente con l'asse longitudinale del velivolo e diretto secondo la direzione di volo, l'asse Z_b è contenuto nel piano di simmetria longitudinale del velivolo e diretto verso il basso nelle condizioni di volo orizzontale, l'asse Y_b è tale da rendere la terna levogira.

- *Assi vento* con origine nel baricentro, è definito con l'asse X_w coincidente con direzione e verso della velocità relativa, che comprende l'eventuale contributo del vento, e l'asse Z_w ortogonale ad esso e giacente nel piano di simmetria (X_b, Y_b) del velivolo.

1.2 Navigazione terrestre

Alcune idee alla base della geodesia sono spesso indispensabili per simulare il moto di un velivolo o di un veicolo aerospaziale attorno alla Terra e per capire come è strutturato in linea generale l'ambiente in cui il velivolo si muove.

La Terra è descritta da un *ellissoide* (o *sferoide*) che ne risalta la forma schiacciata ai poli. Diverso è l'andamento delineato dal *geoide*, che è la superficie su cui giacciono i punti allo stesso potenziale gravitazionale, ed è molto più irregolare rispetto allo sferoide a causa della non uniforme massa terrestre; esso coincide con il livello del mare.

Di conseguenza come evidenziato in Fig. 1.1 la normale al geoide, lungo la quale è diretta la forza di gravità, non coincide con la normale allo sferoide; l'angolo di sfasamento tra le due è detto *deflessione della verticale*.

[1]Sebbene la terna ECEF non sia inerziale, per i velivoli privi di pilota esistono le condizioni affinché sia considerata tale. Infatti, nell'ipotesi di tempi di funzionamento molto brevi, il moto di rotazione può considerarsi trascurabile.

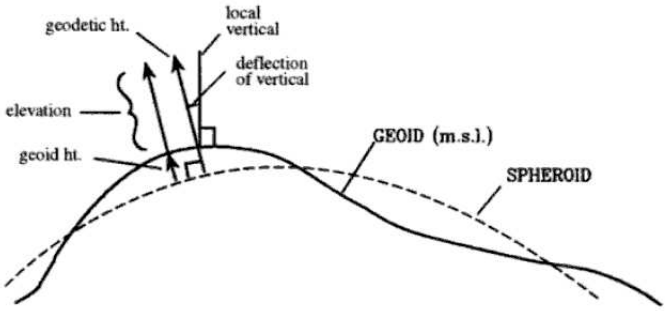


Figura 1.1: Ellissoide e geoide

1.2.1 Coordinate geocentriche

Facendo riferimento alla Fig. 1.2, dato un punto P nell'atmosfera, che rappresenta il centro di massa di un velivolo, ai sistemi di riferimento ECI ed ECEF appartengono le cosiddette *coordinate geocentriche* di P definite da:

- latitudine geocentrica Ψ: angolo che il segmento congiungente il punto considerato ed il centro della Terra forma con il piano equatoriale;
- longitudine terrestre l (se misurata in ECEF): angolo tra il piano contenente il meridiano di Greenwich ed il piano contenente il meridiano passante per il punto considerato;
- longitudine celeste λ (se misurata in ECI): angolo tra il piano contenente l'asse X_i, orientato verso l'equinozio di primavera, ed il piano contenente il meridiano passante per il punto considerato.

Il legame tra longitudine terrestre e celeste è espresso dalla formula:

$$\lambda - \lambda_0 = l - l_0 + \omega_E t \tag{1.1}$$

dove t è il tempo, ω_E è la velocità di rotazione della Terra e λ_0 e l_0 sono i valori iniziali di λ e l al tempo $t = 0$.

1.2.2 Coordinate geodetiche

Sempre nell'ambito del sistema ECEF, esiste un riferimento, denominato WGS-84 (*World Geodetic System 1984*), costruito attraverso osservazioni spaziali e a cui è associato l'ellissoide WGS-84 anche esso definito attraverso questo tipo

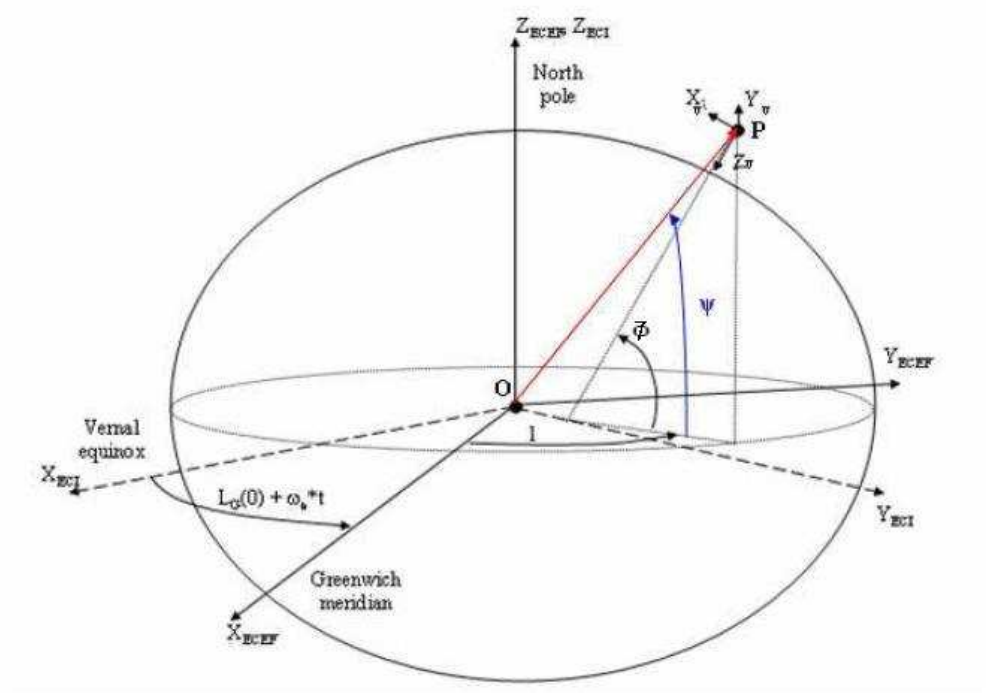


Figura 1.2: Ellissoide terrestre e sistemi di riferimento

di osservazioni. In questo riferimento anziché parlare di coordinate geocentriche si parla di *coordinate geodetiche*.

Le coordinate geodetiche sono usate nelle mappe e nella navigazione aerea mediante GPS (*Global Positioning System*) e sono calcolate usando la normale del punto P allo sferoide: mentre la definizione della longitudine resta inalterata possiamo definire:

- la latitudine geodetica $\bar{\phi}$ come l'angolo che la normale allo sferoide forma con il piano equatoriale; a causa dello schiacciamento dello sferoide la normale non passa per il centro della Terra;

- l'altezza geodetica h è l'altezza sopra lo sferoide lungo la normale.

Esistono delle relazioni [13] che legano le coordinate geodetiche alle componenti di velocità nel sistema geografico NED:

$$\dot{\bar{\phi}} = \frac{V_N}{R_m + h} \tag{1.2a}$$

$$\dot{l} = \frac{V_E}{(R_n + h)\cos\bar{\phi}} \tag{1.2b}$$

dove R_m è il raggio di curvatura meridiano ed R_n è il raggio normale di curvatura. L'altezza geodetica è pari alla quota di volo.

E' integrando tali equazioni che si ottengono informazioni circa la latitudine, la longitudine e l'altitudine di un velivolo relativamente al suo punto di massa.

1.2.3 Trasformazione di coordinate relative alla Terra

Le equazioni del moto di un velivolo richiederanno trasformazioni di coordinate da un sistema di riferimento all'altro, realizzabili attraverso matrici di rotazione [13].

La rotazione tra ECI ed ECEF è un piano di rotazione attorno all'asse Z_i. L'Eq. (1.1) fornisce l'angolo di rotazione:

$$\mu = \lambda_0 - l_0 + \omega_E t$$

e la rotazione da ECI ad ECEF può essere ottenuta tramite la seguente matrice:

$$C_{e/i} = \begin{bmatrix} \cos\mu & \sin\mu & 0 \\ -\sin\mu & \cos\mu & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{1.3}$$

Spesso è utile passare da coordinate espresse nel riferimento ECEF alle coordinate geografiche o NED. In tal caso la convenzione è eseguire per prima una rotazione elementare attorno all'asse Z_e di un angolo pari alla longitudine l, in modo tale da muoversi alla corretta longitudine. Una rotazione di 90° attorno al nuovo asse Y', effettuata secondo la regola della mano sinistra, è necessaria per fare in modo che l'asse X'' punti a nord e l'asse Z'' punti verso il basso. A questo punto non resta che portarsi al valore corretto di latitudine effettuando un'ultima rotazione attorno all'asse Y'' secondo la regola della mano sinistra e di un angolo pari alla latitudine $\bar{\phi}$. In questo modo si ottiene l'allineamento con il sistema di coordinate NED.

Abbreviando la scrittura del seno e del coseno con s e c, la matrice di rotazione complessiva, ottenuta moltiplicando da destra verso sinistra le matrici delle rotazioni elementari sopra discusse, è data da:

$$C_{n/e} = \begin{bmatrix} c_{\bar{\phi}} & 0 & s_{\bar{\phi}} \\ 0 & 1 & 0 \\ -s_{\bar{\phi}} & 0 & c_{\bar{\phi}} \end{bmatrix} \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{bmatrix} \begin{bmatrix} c_l & s_l & 0 \\ -s_l & c_l & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Dunque:

$$C_{n/e} = \begin{bmatrix} -s_{\bar{\phi}}c_l & -s_{\bar{\phi}}s_l & c_{\bar{\phi}} \\ -s_l & c_l & 0 \\ -c_{\bar{\phi}}c_l & -c_{\bar{\phi}}s_l & -s_{\bar{\phi}} \end{bmatrix} \tag{1.4}$$

1.3 Angoli di Eulero

Date due generiche terne di riferimento ortogonali, orientate nello spazio in modo qualsiasi, è sempre possibile allineare una terna con l'altra attraverso rotazioni successive.

Esistono sei diverse sequenze di rotazioni con cui è possibile ottenere l'allineamento delle terne e specificando una di esse, si descrive l'orientamento di una terna rispetto all'altra. In particolare, quando si considerano gli assi verticali-locali e gli assi corpo, si usa descrivere l'orientamento della seconda rispetto alla prima mediante la sequenza di *angoli di Eulero*: ϕ, θ e ψ. In questo modo è possibile descrivere l'assetto di un velivolo, indicando il suo orientamento rispetto agli assi verticali-locali.

Gli angoli di Eulero rappresentano tre rotazioni successive da eseguire intorno agli assi di una terna inizialmente allineata con gli assi verticali-locali per allinearla agli assi corpo. La prima rotazione avviene intorno all'asse Z_v di un angolo pari a ψ, detto *angolo di imbardata* (*yaw angle*), ottenendo una terna intermedia $T^{'}$, il cui asse $X^{'}$ è allineato con la proiezione dell'asse X_b sul piano orizzontale individuato dagli assi X_v Y_v . La successiva rotazione avviene intorno all'asse $Y^{'}$ di un angolo pari a θ , detto *angolo di beccheggio* (*pitch angle*), ottenendo una terna $T^{''}$, il cui asse $X^{''}$ risulta definitivamente allineato con l'asse X_b. L'ultima rotazione avviene intorno all'asse $X^{''}$ di un angolo pari a ϕ , detto *angolo di rollio* (*roll angle*), che allinea completamente le due terne.

Gli angoli di Eulero, rappresentando le rotazioni da compiere per allineare gli assi della terna verticale-locale con gli assi corpo, sono utilizzati per descrivere l'assetto del velivolo.

1.3.1 Matrice dei coseni direttori

La rotazione globale, rispetto ad una terna fissa di riferimento, è caratterizzata da una matrice ottenuta moltiplicando da destra verso sinistra le matrici delle

rotazioni elementari sopra discusse:

$$\mathbf{u}^b = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{u}^r \tag{1.5}$$

La matrice $C_{b/r}$ denota la trasformazione completa dal sistema di riferimento, nel nostro caso assi verticali-locali, al sistema assi corpo. Abbreviando la scrittura del seno e del coseno con s e c[2], si ottiene:

$$C_{b/r} = \begin{bmatrix} c_\theta c_\psi & c_\theta s_\psi & -s_\theta \\ (-c_\phi s_\psi + s_\phi s_\theta c_\psi) & (c_\phi c_\psi + s_\phi s_\theta s_\psi) & s_\phi c_\theta \\ (s_\phi s_\psi + c_\phi s_\theta c_\psi) & (-s_\phi c_\psi + c_\phi s_\theta s_\psi) & c_\phi c_\theta \end{bmatrix} \tag{1.6}$$

La matrice $C_{b/r}$ è ortogonale e la trasformazione inversa è data da $C_{r/b} = C_{b/r}^T$.

1.4 Quaternioni

Nell'ambito della navigazione inerziale i quaternioni sono impiegati per descrivere la rotazione dei sistemi di riferimento. Essi rappresentano una conveniente alternativa agli angoli di Eulero nella descrizione dell'assetto dei velivoli nei sistemi di calcolo per l'assenza di punti di singolarità e per vantaggi dal punto di vista computazionale.

I quaternioni sono quadruple di numeri reali considerati in un preciso ordine. Il quaternione può essere rappresentato come un numero complesso generalizzato a quattro componenti:

$$\mathbf{q} = q_0 + q_1\mathbf{i} + q_2\mathbf{j} + q_3\mathbf{k}$$

con $\mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = \mathbf{ijk} = -1$, $\mathbf{ij} = \mathbf{k}$, $\mathbf{jk} = \mathbf{i}$, $\mathbf{ki} = \mathbf{j} = \mathbf{-ik}$ e costituisce un tentativo di generalizzare i numeri complessi nel piano a tre dimensioni. Maggiori dettagli e proprietà possono essere studiate in [13].

1.4.1 Trasformazione di vettori

Secondo il Teorema di Eulero, qualsiasi sequenza di rotazioni nello spazio può essere sostituita da una singola rotazione intorno ad un solo opportuno asse. Un

[2]Da questo momento, le due notazioni saranno utilizzate indistintamente a seconda dei casi.

quaternione che rappresenti una rotazione ha modulo unitario e le sue componenti hanno il seguente significato:

$$\mathbf{q} = \cos\frac{\vartheta}{2} + \mathbf{r}\sin\frac{\vartheta}{2}$$

dove $\mathbf{r}$ è la parte vettoriale del quaternione ed è usato per definire l'asse di rotazione e ϑ è la parte scalare e definisce l'angolo di rotazione. Vale, inoltre, il vincolo $q_0^2 + q_1^2 + q_2^2 + q_3^2 = 1$ da cui il nome di quaternione *unitario* [14].

La trasformazione:

$$\mathbf{q}^{-1} * u * \mathbf{q} \tag{1.7}$$

fornisce la rotazione secondo la regola della mano sinistra di un vettore u di un angolo ϑ attorno all'asse $\mathbf{r}$ [13]. Sia u^a un vettore del sistema di riferimento A, al fine di ottenere lo stesso vettore u^b espresso in un sistema di riferimento B, sviluppando la formula (1.7), si ottiene:

$$C_{b/a} = \begin{bmatrix} (q_0^2 + q_1^2 - q_2^2 - q_3^2) & 2(q_1q_2 + q_0q_3) & 2(q_1q_3 - q_0q_2) \\ 2(q_1q_2 - q_0q_3) & (q_0^2 - q_1^2 + q_2^2 - q_3^2) & 2(q_2q_3 + q_0q_1) \\ 2(q_1q_3 + q_0q_2) & 2(q_2q_3 - q_0q_1) & (q_0^2 - q_1^2 - q_2^2 + q_3^2) \end{bmatrix} \tag{1.8}$$

che rappresenta la matrice di rotazione espressa attraverso i quaternioni.

E' chiaro che, quando il quaternione $\mathbf{q}$ descrive l'assetto del velivolo, la matrice (1.8) costituisce la matrice di trasformazione $C_{b/r}$ dal sistema assi verticali-locali al sistema assi corpo.

1.4.2 Relazioni con gli angoli di Eulero

Anche se l'assetto viene calcolato mediante i quaternioni, è indispensabile disporre di relazioni che leghino i quaternioni e gli angoli di Eulero perché questi ultimi ne forniscono una descrizione più pratica.

Le relazioni cercate si ottengono dal confronto dei termini delle matrici di trasformazione $C_{b/r}$ fra sistema assi verticali-locali e sistema assi corpo, espresse mediante quaternione e mediante gli angoli di Eulero. Dal confronto della (1.6) e della (1.8) si ha che:

$$\begin{aligned} \phi &= \arctan\frac{2(q_0q_1 + q_2q_3)}{1 - 2(q_1^2 + q_2^2)} \\ \theta &= \arcsin[2(q_0q_2 - q_1q_3)] \\ \psi &= \arctan\frac{2(q_0q_3 + q_1q_2)}{1 - 2(q_2^2 + q_3^2)} \end{aligned} \tag{1.9}$$

Sono possibili anche relazioni opposte che forniscono le componenti del quaternione in funzione degli angoli di Eulero [13], anche se costituiscono un tipo di trasformazione quasi mai utilizzata.

1.5 Equazioni del modello

In questo paragrafo vengono presentate le equazioni del modello di un UAV. Tale equazioni sono indispensabili per la sintesi del controllore, dello stimatore e per realizzare un simulatore su cui testare il controllo.

In Fig. 1.3 è rappresentato un velivolo e le relative coordinate utilizzate. Il velivolo viene controllato usando solo due superfici di controllo, ossia l'elevatore ed il timone presenti sul piano di coda orizzontale e verticale [4].

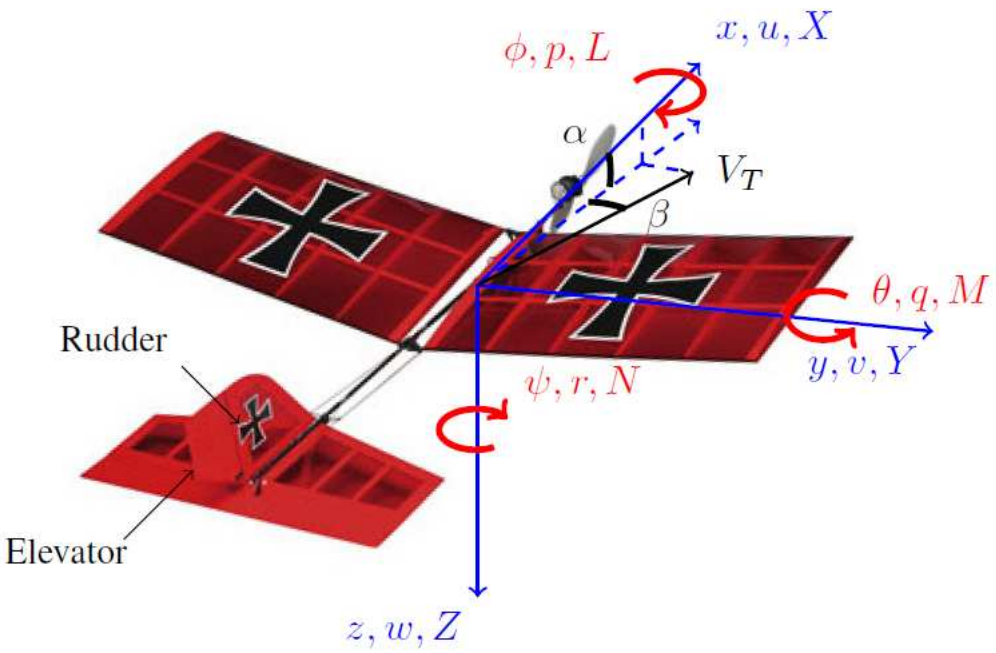


Figura 1.3: UAV con le definizioni delle coordinate.

1.5.1 Modello dinamico

Per modellare e simulare il sistema UAV vengono usate le equazioni standard del moto per un aeromodello convenzionale. Considerando il velivolo come un corpo rigido simmetrico (rispetto al piano $x - z$) e la Terra piatta (velivolo in

volo su una piccola area terrestre), le dinamiche del velivolo sono descritte da un set di dodici equazioni differenziali non lineari, che coinvolgono equazioni di forze e momenti, equazioni cinematiche ed equazioni di navigazione [15].

- **Equazioni delle forze**:

$$\dot{u} = rv - qw - g\sin\theta + \frac{X + F_{\text{prop}}}{m} \tag{1.10a}$$

$$\dot{v} = -ru + pw + g\sin\phi\cos\theta + \frac{Y}{m} \tag{1.10b}$$

$$\dot{w} = qu - pv + g\cos\phi\cos\theta + \frac{Z}{m} \tag{1.10c}$$

dove u, v e w (m/s) sono le velocità lineari del velivolo in assi corpo, p, q ed r (rad/s) sono le velocità angolari in assi corpo, ϕ, θ e ψ (rad) sono gli angoli di Eulero e F_{prop} (N) è la forza generata dal sistema di propulsione lungo l'asse longitudinale. X, Y e Z sono le forze aerodinamiche nel riferimento assi corpo e sono date da:

$$X = \bar{q}\,S\,C_X(\alpha, \dot{\alpha}, \beta, \delta_e, \delta_r, p, q, r, V_T) \tag{1.11a}$$

$$Y = \bar{q}\,S\,C_Y(\beta, \delta_e, \delta_r, p, r, V_T) \tag{1.11b}$$

$$Z = \bar{q}\,S\,C_Z(\alpha, \dot{\alpha}, \beta, \delta_e, \delta_r, p, q, r, V_T), \tag{1.11c}$$

con $\bar{q}$ (Pa) che rappresenta la pressione dinamica e α (rad), $\dot{\alpha}$ (rad/s) e β (rad) rispettivamente, l'angolo di attacco, la sua variazione nel tempo e l'angolo di derapata del velivolo in assi vento che possono essere definite in termini di componenti delle velocità

$$\alpha = \tan^{-1}\left(\frac{w}{u}\right) \tag{1.12a}$$

$$\beta = \sin^{-1}\left(\frac{v}{V_T}\right) \tag{1.12b}$$

dove $V_T = (u^2 + v^2 + w^2)^{\frac{1}{2}}$ (m/s) è la velocità dell'aria. C_X, C_Y e C_Z sono i coefficienti delle forze aerodinamiche espresse in assi corpo e sono funzioni non lineari di diverse variabili, come la velocità angolare del velivolo e gli ingressi di controllo, ma anche di parametri fisici.

- **Equazioni dei momenti**:

$$\begin{aligned}\Gamma\dot{p} &= J_{xz}[J_x - J_y + J_z]pq + \\ &\quad - [J_z(J_z - J_y) + J_{xz}^2]qr + J_z L + J_{xz} N \qquad (1.13a)\\ J_y\dot{q} &= (J_z - J_x)pr - J_{xz}(p^2 - r^2) + M \qquad (1.13b)\\ \Gamma\dot{r} &= [(J_x - J_y)J_x + J_{xz}^2]pq + \\ &\quad - J_{xz}[J_x - J_y + J_z]qr + J_{xz} L + J_x N \qquad (1.13c)\end{aligned}$$

con $\Gamma = J_x J_z - J_{xz}^2$ (coefficienti del momento d'inerzia), e L, M ed N che sono i momenti rispetto agli assi x, y e z del riferimento assi corpo:

$$L = \bar{q}\, S\, b\, c_l(\beta, \delta_r, p, r, V_T) \qquad (1.14a)$$

$$M = \bar{q}\, S\, \bar{c}\, c_m(\alpha, \dot{\alpha}, \delta_e, q, V_T) \qquad (1.14b)$$

$$N = \bar{q}\, S\, b\, c_n(\beta, \delta_r, p, r, V_T) \qquad (1.14c)$$

dove i coefficienti dei momenti aerodinamici c_l, c_m and c_n sono funzioni di diverse variabili.

- **Equazioni cinematiche**: per poter descrivere pienamente il moto del velivolo occorre aggiungere l'equazione cinematica della velocità angolare che permette di calcolare la velocità angolare di un corpo rigido quando siano noti il suo orientamento corrente, in termini di angoli di Eulero, e le derivate degli angoli stessi.

  La trasformazione è indicata con:

$$\begin{bmatrix} P \\ Q \\ R \end{bmatrix} = \begin{bmatrix} 1 & 0 & -\sin\theta \\ 0 & \cos\phi & \sin\phi\cos\theta \\ 0 & -\sin\phi & \cos\phi\cos\theta \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \qquad (1.15)$$

  da cui le equazioni cinematiche che descrivono la variazione dell'assetto del velivolo nel tempo sono date da:

$$\dot{\phi} = p + \tan\theta(q\sin\phi + r\cos\phi) \qquad (1.16a)$$

$$\dot{\theta} = q\cos\phi - r\sin\phi \qquad (1.16b)$$

$$\dot{\psi} = \frac{q\sin\phi + r\cos\phi}{\cos\theta} \qquad (1.16c)$$

- **Equazioni di navigazione**:

$$\begin{aligned}\dot{p}_N &= u\cos\theta\cos\psi + \\ &+ v(-\cos\phi\sin\psi + \sin\phi\sin\theta\cos\psi) + \\ &+ w(\sin\phi\sin\psi + \cos\phi\sin\theta\cos\psi) \end{aligned} \tag{1.17a}$$

$$\begin{aligned}\dot{p}_E &= u\cos\theta\sin\psi + \\ &+ v(\cos\phi\cos\psi + \sin\phi\sin\theta\sin\psi) + \\ &+ w(-\sin\phi\cos\psi + \cos\phi\sin\theta\sin\psi) \end{aligned} \tag{1.17b}$$

$$\dot{h} = u\sin\theta - v\sin\phi\cos\theta - w\cos\phi\cos\theta \tag{1.17c}$$

  dove p_N, p_E e h (m) sono, rispettivamente, la distanza percorsa verso Nord, verso Est e la quota di volo che individuano la posizione del velivolo rispetto alla Terra.

Le dinamiche di volo dei micro UAV sono molto sensibili alle dinamiche del sistema di propulsione che, nel caso in esame, sono date da:

$$(J_{\text{mot}} + J_{\text{prop}})\dot{\omega}_p = T_{\text{mot}} - T_{\text{prop}} \tag{1.18}$$

dove ω_p è la velocità angolare dell'elica (rad/s) e viene usata per descrivere le dinamiche del sistema di propulsione, J_{mot} e J_{prop} ($\text{kg}\,\text{m}^2$) sono i momenti di inerzia, rispettivamente, del corpo rotante del motore e dell'elica, e T_{mot} e T_{prop} (N m) sono la coppia in uscita all'albero motore e la coppia generata dall'elica.

La Fig. 1.4 mostra il grafico della relazione tra la posizione dello stick relativo al motore δ_T e la potenza in uscita all'albero motore P_0.

In condizioni ideali la coppia generata all'albero motore a partire dalla potenza in uscita è

$$T_{\text{mot}} = \frac{P_0}{\omega_p}. \tag{1.19}$$

La spinta che fa avanzare il velivolo è generata dalla rotazione dell'elica usando la coppia generata in uscita all'albero motore. La velocità di rotazione dell'elica dipende sia dalla coppia disponibile e sia dalla velocità dell'aria che fluisce tra le pale dell'elica stessa. Tale relazione è data dal rapporto di avanzamento J che caratterizza il coefficiente di spinta C_T e di potenza C_P:

$$J = \frac{\pi V_T}{\omega_p R} \tag{1.20}$$

$$C_T = \frac{F_{\text{prop}}\pi^2}{4\rho R^4\omega_p^2} \tag{1.21}$$

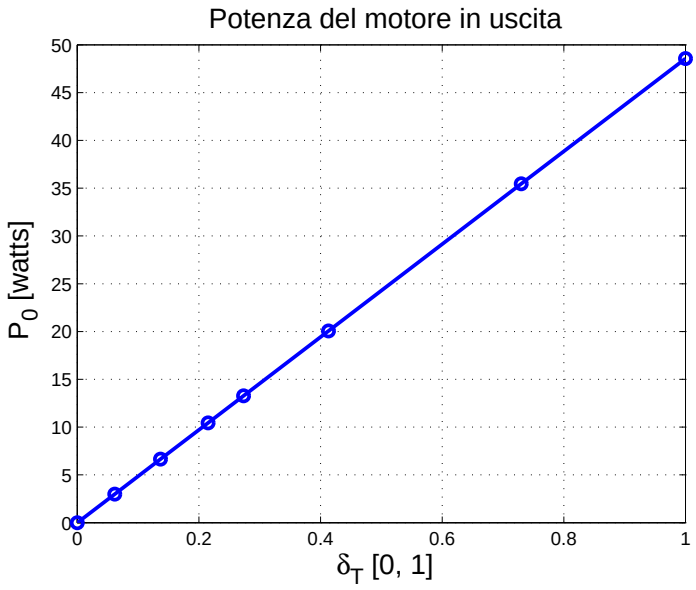


Figura 1.4: Potenza in uscita all'albero motore.

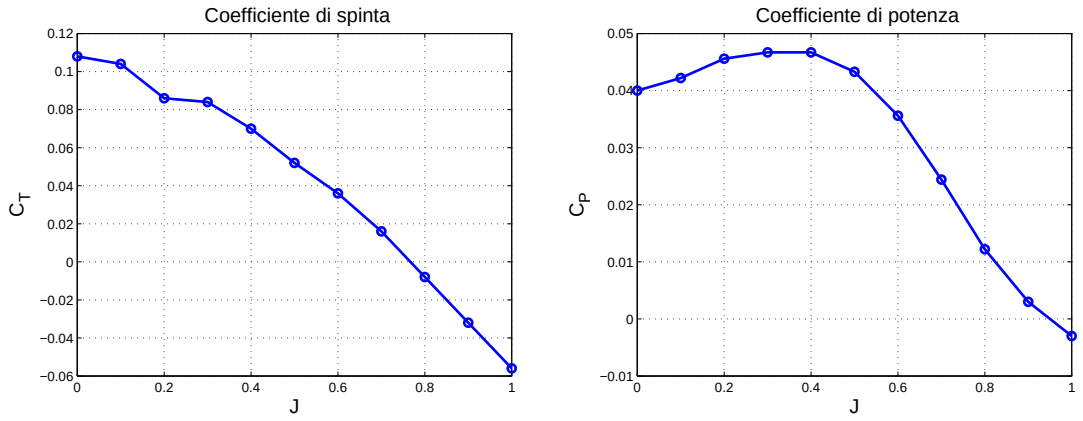
$$C_P = \frac{T_{\text{prop}}\pi^3}{4\rho R^5 \omega_p^2} \tag{1.22}$$

dove ρ (kg/m^3) è la densità dell'aria e R (m) è il raggio dell'elica. In altri termini, data l'airspeed V_T e la velocità angolare dell'elica ω_p, è possibile calcolare il rapporto di avanzamento J che fornisce, a sua volta, C_T e C_P attraverso delle look-up table (Fig. 1.5) identificate attraverso dei test sperimentali nella galleria del vento, facendo variare i valori di airspeed e di posizione dello stick. Dalle (1.20)-(1.21)-(1.22) è possibile calcolare la coppia di spinta T_{prop} e la forza di spinta F_{prop}, grandezze richieste dalla dinamica di propulsione (1.18) e dalle equazioni delle forze (1.10).

1.5.2 Modello lineare

Lo studio del modello linearizzato di un velivolo è un passo utile per l'implementazione di un sistema di controllo.

Le equazioni del modello presentato al paragrafo precedente sono non lineari poiché contengono termini che coinvolgono prodotti e quadrati di variabili dipendenti e termini trascendenti che le rendono difficili da gestire. Dunque, per poter sintetizzare leggi di controllo e discutere la stabilità dinamica del velivolo, è stato necessario ricorrere alla linearizzazione attorno ad una condizione nota



(a) Coefficiente di spinta dell'elica (b) Coefficiente di potenza dell'elica

Figura 1.5: Dati di prestazione dell'elica

di equilibrio, ricorrendo alla teoria dei piccoli disturbi. Si considera, allora, che il moto del velivolo sia dato dalla somma di due componenti: un moto medio rappresentato dalla condizione di equilibrio e un moto dinamico che considera perturbazioni attorno all'equilibrio, cosicché lo scopo del controllo sia rendere le deviazioni dallo stato di equilibrio il più piccole possibili.

Nell'ipotesi che nella maggior parte del tempo il velivolo si trovi in volo di crociera, la condizione che è stata scelta è quella di moto rettilineo uniforme.

Il modello lineare nello spazio di stato è dato da:

$$\dot{x} = Ax + B\delta \tag{1.23}$$

con lo stato $x = [u, v, w, \phi, \theta, \psi, p, q, r, h, \omega_p]^T$ e gli ingressi di controllo $\delta = [\delta_e, \delta_r, \delta_T]^T$ costituiti dall'elevatore, dal timone e dal segnale di controllo relativo al motore. A questo punto è pratica comune disaccoppiare il modello linearizzato nei modi longitudinali e latero-direzionali per semplificare l'analisi di controllo, sotto l'assunzione che gli effetti del mutuo accoppiamento tra i due modi siano trascurabili. Per il velivolo considerato in questo progetto, questa assunzione è valida se le superfici di controllo sono tali che l'angolo di rollio ϕ sia 0 e le velocità angolari p e r siano trascurabili. In generale non è facile capire se il disaccoppiamento è in grado di catturare tutto il comportamento del modello, anche perché non è immediato capire quando le velocità angolari p e r sono trascurabili rispetto alle altre variabili di stato.

Per tale motivo, in [4] è stata condotta un'analisi modale utilizzando i fattori di partecipazione, che ha permesso di giustificare il disaccoppiamento del sistema

complessivo nei seguenti sottosistemi:

- modello lineare longitudinale:

$$\dot{\mathbf{x}}_{lon} = A_{lon}\mathbf{x}_{lon} + B_{lon}\delta_{lon}$$
$$\mathbf{y}_{lon} = C_{lon}\mathbf{x}_{lon}$$

  dove:

$$\mathbf{x}_{lon} = [u, w, q, \theta, h, \omega_p]^T$$
$$\delta_{lon} = [\delta_e, \delta_T]^T$$
$$\mathbf{y}_{lon} = [V_T, \alpha, q, \theta, h]^T$$

- modello lineare latero-direzionale:

$$\dot{\mathbf{x}}_{lat} = A_{lat}\mathbf{x}_{lat} + B_{lat}\delta_{lat}$$
$$\mathbf{y}_{lat} = C_{lat}\mathbf{x}_{lat} + D_{lat}\delta_{lat}$$

  dove:

$$\mathbf{x}_{lat} = [v, p, r, \phi, \psi]^T$$
$$\delta_{lat} = [\delta_r]^T$$
$$\mathbf{y}_{lat} = [\beta, p, r, \phi, \psi]^T$$

Maggiori dettagli sui modelli linearizzati e sull'analisi di stabilità del sistema, possono essere trovati in [4].

1.6 Sistema di controllo di volo

Un volo autonomo per una missione di un UAV consiste nel volare ad una condizione di volo desiderata, eseguendo una navigazione tra waypoint e mantenendo i valori di airspeed e di altitudine costanti e quanto più prossimi ai relativi valori di trim. Per raggiungere questo obiettivo, è necessario implementare un

sistema di controllo di volo (FCS, *flight control system*) che agisca sulle superfici di controllo che si hanno a disposizione, in questo caso timone e elevatore, e sulla spinta del motore.

Il sistema di controllo, presentato in questo paragrafo e discusso in dettaglio in [4], è composto da diverse parti, come mostra la Fig. 1.6. L'autopilota principale ha il compito di guidare il velivolo attraverso un percorso definito dall'utente e di stabilizzarlo, controllandone l'assetto e la velocità. Questa funzione può essere suddivisa in due sistemi specifici: il *waypoint guidance system* che, con l'aiuto del GPS, determina l'orientamento di riferimento per inseguire il percorso desiderato sul piano laterale ed invia tale riferimento al *control system* che, invece, agisce sulle superfici di controllo per posizionare il velivolo all'altezza, alla velocità e alla posizione desiderata. Si noti che tale approccio permette di definire un sistema di controllo modulare composto da diversi strati organizzati in modo gerarchico.

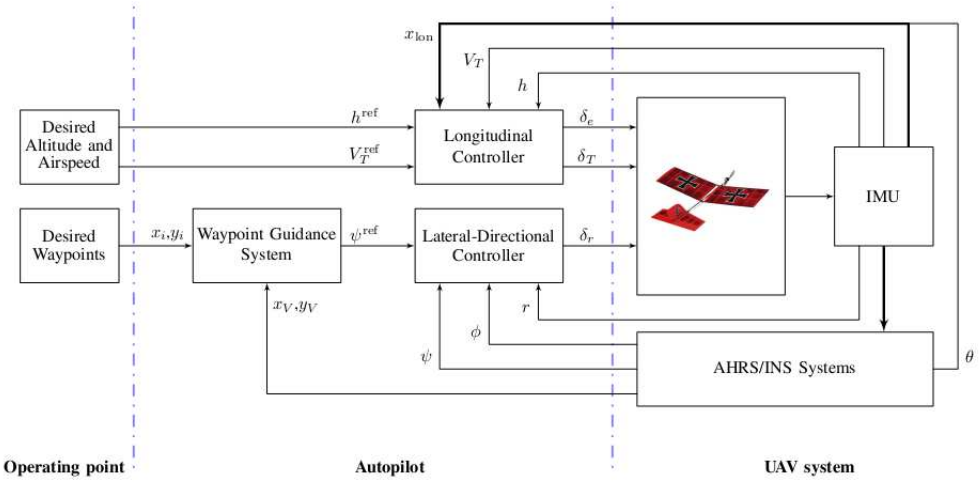


Figura 1.6: Architettura sistema controllo di volo per Yardstik.

Il velivolo in esame, inoltre, è dotato di avionica di bordo: un'unità di misurazione inerziale (IMU), con GPS integrato, fornisce informazioni sullo stato del velivolo; a causa, però, dell'assenza di sensori per l'assetto, occorre utilizzare un *attitude and heading reference system* (AHRS), ovvero uno stimatore che fornisca informazioni sull'assetto in modo da controllore il moto del velivolo.

Nei capitoli successivi verrà approfondita questa tematica in quanto lo scopo del presente lavoro è proprio quello di migliorare le prestazioni di tale stimatore, a fronte delle problematiche che verranno presentate successivamente.

1.6.1 Progettazione del sistema di controllo

Il sistema di controllo di volo lavora separatamente sul piano orizzontale e verticale, sfruttando la decomposizione nei modelli longitudinale e latero-direzionale [4].

Il sistema di controllo complessivo è stato, dunque, implementato in maniera decentralizzata, realizzando due azioni in retroazione indipendenti l'una dall'altra: ognuna di esse è interconnessa ad un sottoinsieme delle variabili di ingresso che permettono di controllare un sottoinsieme delle variabili di uscita [16].

Controllore nel piano longitudinale

Il controllore nel piano longitudinale ha l'obiettivo di stabilizzare quota e velocità del velivolo ad un valore di riferimento, agendo sull'elevatore e sulla spinta del motore. In generale questo tipo di controllo è implementato in maniera decentralizzata, sebbene la scelta dell'accoppiamento tra le variabili di controllo e le variabili controllate non sia ben definita a causa della mutua influenza tra velocità e altitudine [17].

Per questo motivo, guardando al modello MIMO, è stata implementata una strategia di controllo *lineare quadratico* (LQR) che, una volta definiti i criteri di prestazione, permette il calcolo simultaneo dei guadagni di controllo.

Al fine di ottenere, oltre alla stabilità del sistema ad anello chiuso anche la precisione a regime nell'inseguimento di un segnale a gradino, è stato considerato il problema di controllo ottimo:

$$\min_{\delta_{lon}} \frac{1}{2} \int_0^{+\infty} \left[\bar{x}_{lon}^T(t) Q \bar{x}_{lon}(t) + \delta_{lon}^T(t) R \delta_{lon}(t) \right] dt \tag{1.24}$$

s.t.

$$\dot{\bar{\mathbf{x}}}_{lon} = \bar{A}_{lon} \bar{\mathbf{x}}_{lon} + \bar{B}_{lon} \delta_{lon}$$

dove

$$\bar{\mathbf{x}}_{lon} \triangleq \begin{bmatrix} \mathbf{x}_{lon} \\ \mathbf{z} \end{bmatrix} \qquad \bar{A}_{lon} \triangleq \begin{bmatrix} A_{lon_{6\times 6}} & 0_{6\times 1} \\ -T_{2\times 6} & 0_{2\times 1} \end{bmatrix}$$

$$\bar{B}_{lon} \triangleq \begin{bmatrix} B_{lon_{6\times 2}} \\ 0_{2\times 2} \end{bmatrix}$$

dove T è una sottomatrice delle matrici delle uscite C_{lon} e

$$\mathbf{z} = \begin{bmatrix} z_{V_T} \\ z_h \end{bmatrix} = \begin{bmatrix} \int_{t_0}^{\infty} (V_T^{ref} - V_T)\, dt \\ \int_{t_0}^{\infty} (h^{ref} - h)\, dt \end{bmatrix} \tag{1.25}$$

Poiché le dinamiche longitudinali sono sia controllabili che osservabili [4], il controllo LQR esegue un controllo ottimo su orizzonte infinito:

$$\delta_{lon}^{*} = -R^{-1} \bar{B}_{lon}^{T} P(t) \bar{x}_{lon}(t) = -K \bar{x}_{lon}(t) \tag{1.26}$$

dove P è la soluzione dell'equazione algebrica di Riccati:

$$P\bar{A}_{lon} + \bar{A}_{lon}^{T} P - P\bar{B}_{lon} R^{-1} \bar{B}_{lon}^{T} P + Q = 0$$

Decomponendo K come:

$$K = \begin{bmatrix} K_{fb_1} & K_{fb_2} \end{bmatrix}$$

l'azione di controllo complessiva può essere scritta come:

$$\delta_{lon}^{*} = -K_{fb_1} \bar{x}_{lon} - K_{fb_2} z \tag{1.27}$$

Le matrici di peso scelte sono:

$$Q = 0.1 \cdot \bar{C}_{lon}^{T} \bar{C}_{lon} = \begin{bmatrix} 0.1 & 0.0017 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.0017 & 0.0013 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.1 \end{bmatrix}$$

$$R = \begin{bmatrix} 3 & 0 \\ 0 & 1 \end{bmatrix}$$

da cui i valori del controllo proporzionale ottenuti sono:

$$K_{fb_1} = \begin{bmatrix} -0.0160 & 0.0775 & -0.1127 & -4.6363 & -0.4479 & -0.0000 \\ 0.3801 & -0.0167 & 0.0193 & 0.9406 & 0.3155 & 0.0015 \end{bmatrix}$$

$$K_{fb_2} = \begin{bmatrix} 0.0698 & -0.1687 \\ 0.2922 & 0.1209 \end{bmatrix}$$

.

Controllore nel piano latero-direzionale

Il sistema di controllo del modello laterale consiste in un sistema di aumento della stabilità (SAS), come può essere uno smorzatore della velocità di imbardata (*yaw damper*), e in un controllo dell'heading, che agisce sulla superficie di controllo del timone. Tale controllo è costituito da una classica azione PID multipla, che consiste in tre loop di feedback in cascata (velocità di imbardata, angolo di rollio e angolo di imbardata), in modo da attuare l'unica superficie di controllo a disposizione, il timone.

L'architettura del controllore latero-direzionale è mostrata in Fig. 1.7. E' stato progettato uno *yaw damper* per migliorare la stabilità laterale. Utilizzando la tecnica del luogo delle radici sulla funzione di trasferimento a ciclo aperto tra la velocità di imbardata e l'input fornito dal timone, è stato scelto un guadagno che smorzasse i poli complessi. Inoltre un filtro passa-alto del primo ordine, chiamato *washout filter*, è stato aggiunto nella retroazione per filtrare velocità di imbardata alle basse frequenze. Nel loop centrale è stata inserita un'azione PI per controllare il rollio del velivolo. L'angolo di rollio di riferimento (ϕ_{ref}) viene calcolato da un controllore PD esterno, a partire dall'orientamento di riferimento (ψ_{ref}).

In definitiva, il loop più interno si occupa della stabilità del velivolo, quelli esterni sono stati progettati per ottenere buone prestazione nell'inseguimento dell'heading di riferimento. Ulteriori dettagli possono essere trovati in [4].

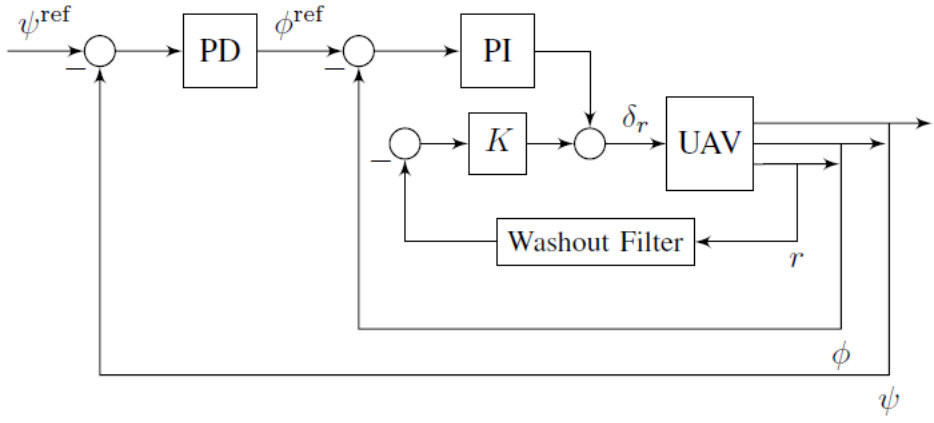


Figura 1.7: Schema di controllo complessivo per l'autopilota nel piano latero-direzionale.

1.6.2 Navigazione tra waypoints

Una volta progettati gli autopiloti per le dinamiche longitudinali e laterodirezionali, non resta che completare l'architettura di controllo con lo stadio di livello superiore: l'algoritmo per la *navigazione*.

Il *waypoint guidance system* fornisce il riferimento per l'orientamento in modo da inseguire un percorso desiderato sul piano laterale, definito da una serie di *waypoints* che il velivolo deve attraversare nell'ordine stabilito. Per quanto riguarda il piano longitudinale, la coordinata relativa all'altitudine del waypoint viene utilizzata come riferimento per il controllo di altitudine.

I waypoints possono essere memorizzati, tramite le rispettive coordinate geodetiche, nel computer di bordo e, quindi, definiti a priori, oppure possono essere definiti in tempo reale e inviati attraverso una comunicazione wireless dalla GCS al velivolo. I waypoints sono, inoltre, elaborati dall'algoritmo di navigazione che, considerando anche la posizione corrente del velivolo, determina i segnali di controllo da inviare agli autopiloti posti al livello inferiore, al fine di portare a termine la missione assegnata al velivolo.

Algoritmo di navigazione

Il metodo più semplice per definire una traiettoria tra waypoints successivi è l'algoritmo con linea di vista (LOS, *line-of-sight*) [18]. L'algoritmo fornisce l'angolo di riferimento ψ_0^{ref} che guiderà il velivolo dalla sua posizione corrente (x_V, y_V) verso il waypoint $P_i = (x_i, y_i)$:

$$\tan(\psi_0^{\text{ref}}) = \frac{y_i - y_V}{x_i - x_V} \tag{1.28}$$

Al fine di migliorare le prestazioni ed evitare i transitori causati dal passaggio tra due waypoint consecutivi, sono state considerate delle regole per indentificare i waypoint mancati ed è stato migliorato l'algoritmo della linea di vista. In particolare, se un algoritmo di navigazione tra waypoint non tiene conto della posizione del prossimo waypoint prima di raggiungere il waypoint corrente, è possibile che il velivolo inizi a girare verso il prossimo waypoint troppo tardi, allontanandosi dal percorso desiderato. Per questo motivo viene applicata una correzione all'algoritmo con linea di vista, come proposto in [19]:

$$\psi^{\text{ref}} = \psi_0^{\text{ref}} + \psi_c \tag{1.29}$$

dove ψ_c è il termine correttivo determinato dalla relazione geometrica tra la posizione del waypoint corrente $P_i = (x_i, y_i)$, quella del successivo $P_{i+1} = (x_{i+1}, y_{i+1})$

e le coordinate del velivolo (x_V, y_V). Maggiori dettagli sugli algoritmi utilizzati possono essere trovati in [19].

Capitolo 2

Stima dell'assetto

Per l'implementazione dello schema di controllo introdotto nel capitolo precedente è necessario ottenere le informazioni sull'assetto del velivolo, non direttamente accessibili dall'avionica di bordo. Avendo, infatti, i micro UAV capacità di carico limitata, l'utilizzo di tutte le tipologie di sensori potrebbe risultare proibitiva sia in termini di peso che di costi. Ciò rende necessario il ricorso ad algoritmi di stima. Nel presente capitolo, dunque, è affrontato il problema della stima dell'assetto ed è presentato l'*Attitude and Heading Reference System*, generalmente utilizzato come soluzione al problema.

2.1 Unità di misurazione inerziale

Quando si progetta un controllore si ipotizza che l'intero vettore di stato del sistema sia disponibile attraverso misurazioni e quindi utilizzabile nella sintesi delle leggi di controllo. Per la maggior parte dei sistemi ciò non è vero e bisogna ricorrere ad algoritmi di stima per la definizione delle variabili di interesse.

Il velivolo in esame è dotato di un'unità di misurazione inerziale, ovvero di un insieme di sensori necessario per la navigazione e il controllo di veicoli robotici di terra e di aria in miniatura, e integrati in un unico modulo compatto.

In particolare, l'unità utilizzata per lo Yardstik è il MicroNav MNAV100CA prodotto dalla CrossBow [9] e mostrato in Fig. 2.1. La struttura sfrutta la moderna tecnologia MEMS (*Micro Electro Mechanical System*) nella configurazione strapdown, in cui i sensori sono rigidamente fissati al baricentro del velivolo e orientati in modo coerente con il sistema di riferimento ad esso solidale. Il MicroNav consiste dei seguenti sensori:

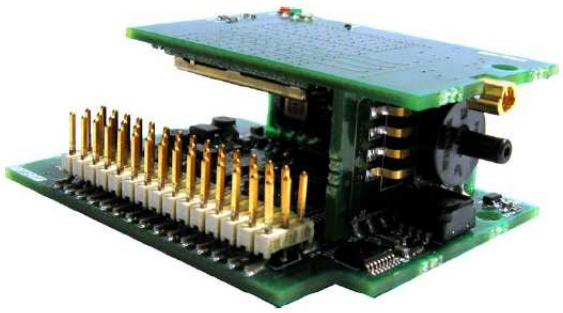


Figura 2.1: Unità di misurazione inerziale.

- **sensori inerziali** che comprendono tre accelerometri, tre giroscopi, tre sensori di temperatura, uno per ogni asse ortogonale;
- **magnetometri** uno per ogni asse, che possono essere usati per il calcolo della direzione di volo;
- **ricevitore GPS** per le misure di posizione e velocità del baricentro del velivolo;
- **sensori di pressione** che possono essere usati per il calcolo dell'altitudine (pressione statica) e per il calcolo dell'airspeed (pressione dinamica).

Confrontando questo set di sensori con le grandezze necessarie per il controllo, presentate nel capitolo precedente, si evince che le informazioni fornite dai sensori sono insufficienti per il funzionamento delle leggi di controllo sintetizzate.

In particolare le principali grandezze mancanti sono quelle che descrivono l'assetto del velivolo, ovvero gli angoli ϕ, θ e ψ (*angoli di Eulero*, Par. 1.3). Tali grandezze sono fondamentali per la corretta implementazione degli algoritmi di controllo e, inoltre, da esse dipendono anche altre grandezze non disponibili direttamente ma necessarie anch'esse per il controllo.

Si potrebbe, allora, pensare di introdurre nuovi sensori, inclinomentri in questo caso, che forniscano i valori delle grandezze mancanti. Il poter misurare tutte la variabili necessarie, però, si tradurrebbe in maggiori costi e anche maggiori ingombri. Velivoli privi di pilota, al contrario, possono sostenere un carico limitato e ciò proibisce l'utilizzo di sensori pesanti ed ingombranti; inoltre il sistema che si intende realizzare deve essere a basso costo, demandando all'implementazione

software tutto ciò che non può essere realizzato per via hardware. Le soluzioni potrebbero essere, allora, riformulare le leggi di controllo senza fare uso delle componenti dello stato mancanti, oppure adottare degli osservatori per la stima delle componenti di interesse a partire dalle misure disponibili. Nel presente e nel successivo capitolo si affronterà il problema nell'ottica della soluzione basata su osservatori in grado di stimare l'assetto del velivolo.

2.2 Filtro di Kalman

In letteratura esistono molteplici osservatori che risolvono il problema della stima delle componenti non osservabili di un sistema dinamico. Tra questi il più utilizzato è sicuramente il Filtro di Kalman, a causa della sua semplicità, ottimalità e robustezza, soprattutto quando entrano in gioco segnali affetti da errori o rumori di misurazione, come nel caso in esame.

Il Filtro di Kalman [20] è un filtro ricorsivo per l'elaborazione dei dati basato sul metodo dei minimi quadrati e sviluppato dal matematico ungherese Rudolph Emily Kalman nel 1960. Esso consente di unire informazioni provenienti da fonti di misura differenti, al fine di ottenere una stima delle variabili di interesse.

In generale gli elementi necessari per l'applicazione di tale filtro sono:

- la conoscenza di un modello matematico del fenomeno fisico in esame;
- la descrizione statistica dei rumori;
- la stima delle condizioni iniziali delle variabili di interesse.

Sia il processo di misura di una grandezza fisica reale tramite opportuna strumentazione, sia la modellizzazione matematica di un processo fisico, introducono fonti di errore. Le componenti deterministiche dell'errore possono essere ridotte introducendo delle equazioni che ne esprimano il comportamento e le relative correzioni, mentre le componenti random dell'errore vengono ridotte e gestite dal filtro. E' necessario, però, fornire una descrizione statistica delle componenti random che, tipicamente, vengono modellate come rumore bianco, gaussiano a media nulla.

Come già accennato, tale filtro è ricorsivo, ovvero non necessita dell'immagazzinamento di enormi serie di dati relative alla storia temporale del segnale stimato e delle misure acquisite; al contrario, esso memorizza solo l'ultimo valore

del segnale stimato e questo consente un notevole risparmio in termini di memoria e di velocità computazionale, con notevole vantaggio nelle applicazioni real-time.

L'idea fondamentale alla base è quello di migliorare le stime fornite dal modello matematico con le misure (dette anche *osservazioni*) che vengono fornite ad ogni passo (istante temporale in cui sono disponibili le osservazioni). L'effetto del filtro, come sarà possibile notare nella successiva formulazione, sul vettore di stato è costituito sostanzialmente da due contributi: uno predittivo ed uno correttivo. Il contributo predittivo è basato sulla stima dell'errore all'epoca precedente e sull'osservazione che l'errore evolve con una dinamica analoga a quella del processo in esame. Il contributo correttivo, invece, si basa sul calcolo dei *residui*, dove ogni residuo è definito come la differenza tra grandezza fornita dalla generica misura e la stessa grandezza stimata dal sistema tramite il modello. A tale scopo non è necessario che le grandezze da confrontare siano misurate direttamente ma è indispensabile che esse siano coerenti.

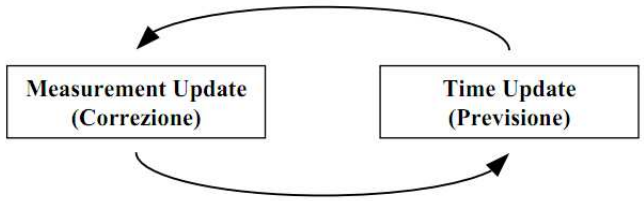


Figura 2.2: Il Filtro di Kalman è una procedura ricorsiva in cui le equazioni di *time update* forniscono una previsione e quelle di *measurement update* determinano un miglioramento della stima introducendo l'informazione contenuta nella misurazione.

Con il passare degli anni, l'impiego del filtro ha trovato sempre maggiore diffusione ed oggi costituisce un elemento fondamentale in varie applicazioni quali, ad esempio, i sistemi di guida e controllo per la robotica, i missili, i velivoli e soprattutto la navigazione in genere.

2.2.1 Descrizione dell'algoritmo

Per la descrizione del filtro si considerino i seguenti simboli, tenendo conto che i pedici k sono utilizzati per tenere conto dell'istante in cui si effettua l'operazione di aggiornamento della stima:

- x_k vettore di stato;

- u_k vettore degli ingressi;
- z_k vettore delle misure: lo stato di un sistema è a volte direttamente misurabile, a volte deve essere misurato attraverso grandezze loro equivalenti o il cui valore rappresenta la combinazione di una o più variabili di stato, possibilmente sporcato da errori di misura o da imprecisione nella modellizzazione matematica del sistema lineare;
- ω_k disturbo sullo stato: rappresenta l'incertezza che si commette nella descrizione matematica del processo;
- v_k rumore sulle misure: rappresenta il rumore di lettura presente nel vettore delle misure;
- Q_k matrice di covarianza del disturbo sullo stato: rappresenta la variabilità statistica del vettore dei disturbi sullo stato;
- R_k matrice di covarianza del rumore sulle misure: rappresenta la variabilità statistica del vettore dei disturbi di misura;
- P_k matrice di varianza dell'errore sullo stato: rappresenta la variabilità dell'errore sulla stima dello stato conseguente ai due fattori di disturbo (errore di misura e disturbo dello stato);
- K_k matrice di correzione della stima: indica il livello di fiducia assegnato alla misura rispetto alla fiducia assegnata alla stima dello stato in base al valore precedente e al modello matematico che ne rappresenta l'evoluzione; tanto maggiore è il valore di K_k tanto minore fiducia merita la stima basata sul modello rispetto alla misura riportata;
- F_k matrice di stato: matrice descrittiva dell'evoluzione libera della variabile di stato rispetto al suo valore attuale;
- G_k matrice degli ingressi: matrice descrittiva dell'evoluzione forzata della variabile di stato rispetto al valore attuale dell'ingresso;
- H_k matrice delle uscite: matrice descrittiva del valore assunto dalle variabili misurate in funzione del valore attuale della variabile di stato.

Le grandezze elencate hanno le seguenti proprietà e definizioni:

- $Q_k = E[\omega_k {\omega_k}^T]$; essendo $E[\omega_j {\omega_k}^T] = 0$ per $j \neq k$;

- $R_k = E[v_k {v_k}^T]$; essendo $E[v_j {v_k}^T] = 0$ per $j \neq k$;
- $E[\omega_j {v_k}^T] = 0 \qquad \forall j, k$;
- $P_0 = E[(x_0 - x)(x_0 - x)]$ avendo definito $x = E[x_0]$.

Il modello matematico del sistema è definito dalle seguenti equazioni:

$$\textit{Evoluzione dello stato} \qquad x_k = F_{k-1}x_{k-1} + G_{k-1}u_{k-1} \tag{2.1}$$

$$\textit{Evoluzione della misura} \qquad z_k = H_k x_k \tag{2.2}$$

Per implementare l'algoritmo è necessario:

- possedere le matrici che rappresentano l'evoluzione dello stato (F, G, H);
- aggiornare le matrici che rappresentano la varianza dei disturbi sullo stato (Q_k) e del rumore sulla misura (R_k).

Le matrici Q_k e R_k sono solitamente utilizzate e definite come matrici costanti. Spesso, infatti, non ha senso modificarne il valore durante le stime successive dello stato. Talvolta, può essere consigliabile modificare Q_k e R_k per tener conto di particolari fattori come, ad esempio, una condizione di maggiore disturbo sullo stato del sistema, oppure presenza di maggiori errori sulla misura, motivati ad esempio dal malfunzionamento di un sensore.

Predizione del modello

- **Previsione dello stato**: si esegue la predizione delle variabili di stato in base alle informazioni certe che si hanno del modello:

$$\hat{x}_{k/k-1} = F_{k-1}\hat{x}_{k-1} + G_{k-1}u_{k-1} \tag{2.3}$$

  Qualora gli ingressi non siano noti, sarà necessario considerare il loro contributo come un disturbo non misurabile e quindi da inserire nella matrice Q_k.

- **Aggiornamento della varianza dell'errore sullo stato**: si aggiorna il valore della varianza dell'errore in base all'ultima stima del disturbo sullo stato:

$$P_{k/k-1} = F_k P_{k-1} F_k^T + Q_{k-1} \tag{2.4}$$

La legge di aggiornamento della matrice di varianza dell'errore sullo stato è di semplice interpretazione:

- se Q_k è nullo la varianza dell'errore sullo stato andrà a diminuire quanto più stabilizzante è la matrice di stato F_k;
- quanto maggiore è Q_k, tanto maggiore inciderà il disturbo sullo stato nell'errore complessivo di stima.

Calcolo del guadagno di Kalman

Il guadagno K_k risulterà maggiore o minore a seconda se prevale l'incertezza dello stato P_k o l'incertezza sulla misura R_k:

$$K_k = P_{k/k-1} H_k^T [H_k P_{k/k-1} H_k^T + R_k]^{-1} \tag{2.5}$$

Con $K_k = I$, supposto che $H_k = I$, l'algoritmo da piena fiducia alla misura effettuata. Per K_k tendente a zero, l'algoritmo tenderà a confermare il valore predetto dalla stima, come illustrato nel passo successivo.

Aggiornamento della stima della stato

Si pesa con la matrice K_k il valore della variabile predetta per propagazione dello stato $x_{k/k-1}$ con l'errore attuale della misura:

$$\hat{x}_k = \hat{x}_{k/k-1} + K_k[z_k - H_k \hat{x}_{k/k-1}] \tag{2.6}$$

Come detto in precedenza, la matrice K_k indica la fiducia posta dall'algoritmo nella misura. K_k tendente a zero indica una scarsa affidabilità della misura rispetto alla fiducia posta nella propagazione del modello.

Aggiornamento della varianza dei disturbi sullo stato

Al contrario dell'aggiornamento della varianza dell'errore sullo stato, l'aggiornamento della matrice di varianza dei disturbi sullo stato viene fatta a posteriori, utilizzando le informazioni contenute nella matrice K_k sulla affidabilità della misura rispetto alla qualità della previsione sullo stato.

$$P_k = [I - K_k H_k] P_{k/k-1} \tag{2.7}$$

Come detto al passo precedente, la matrice K_k indica la fiducia posta dall'algoritmo nella misura; con K_k tendente a zero, la matrice P_k rimane sostanzialmente invariata nell'aggiornamento dell'algoritmo da un passo al successivo. A questo punto si ritorna alla previsione dello stato, con l'aggiornamento del ciclo.

2.3 Filtro di Kalman esteso

Il filtro di Kalman risolve il problema generale della stima dello stato $x \in \mathbb{R}^n$ di un sistema governato da equazioni differenziali lineari stocastiche. Se, invece, il processo da stimare è non lineare, in maniera simile a quanto si fa con la serie di Taylor, si può linearizzare il sistema attorno alla stima corrente usando le derivate parziali delle funzioni di stato e di misura.

Un filtro di Kalman che linearizza intorno alla stima ed alla covarianza correnti è detto filtro di Kalman esteso (*Extended Kalman Filter*).

L'analisi descritta è limitata al tempo discreto. Si assume che il processo abbia un vettore di stato $x \in \mathbb{R}^n$ e sia governato da una equazione differenziale stocastica non lineare:

$$x_k = f(x_{k-1}, u_{k-1}, \omega_{k-1}) \tag{2.8}$$

con misure $z \in \mathbb{R}^m$ date da:

$$z_k = h(x_k, v_k) \tag{2.9}$$

dove ω_k e v_k rappresentano rispettivamente i rumori di processo e di misura: ω_k e v_k sono processi aleatori a media nulla:

$$\omega_k \sim N(0, Q_k)$$

$$v_k \sim N(0, R_k)$$

dove Q_k e R_k sono le matrici di covarianza dei processi aleatori.

Poiché non si conosce istante per istante il valore di ω_k e v_k si possono approssimare le Eqs. (2.8) e (2.9) come:

$$\tilde{x}_k = f(\hat{x}_{k-1}, u_{k-1}, 0) \tag{2.10}$$

$$\tilde{z}_k = h(\tilde{x}_k, 0) \tag{2.11}$$

dove $\hat{x}_{k-1}$ è la stima a posteriori dello stato (relativo al precedente istante di campionamento).

2.3.1 Descrizione del filtro

Si linearizzano intorno alla stima le Eqs. (2.8) e (2.9):

$$x_k \approx \tilde{x}_k + F(x_{k-1} - \hat{x}_{k-1}) + W\omega_{k-1} \tag{2.12}$$

$$z_k \approx \tilde{z}_k + H(x_k - \tilde{x}_k) + Vv_k \tag{2.13}$$

dove:

- x_k e z_k sono i vettori di stato e di misura attuali;
- $\tilde{x}_k$ e $\tilde{z}_k$ sono i vettori di stato e di misura approssimati dati dalle equazioni (2.10) e (2.11);
- F_k è lo Jacobiano di f rispetto ad x, calcolato in $(\hat{x}_{k-1}, u_{k-1}, 0)$, cioè attorno alla stima corrente;
- W_k è lo Jacobiano di f rispetto ad ω, calcolato in $(\hat{x}_{k-1}, u_{k-1}, 0)$;
- H_k è lo Jacobiano di h rispetto a x, calcolato in $(\tilde{x}_k, 0)$;
- V_k è lo Jacobiano di h rispetto a v, calcolato in $(\tilde{x}_k, 0)$.

  Per semplicità di notazione non si utilizza il pedice k per gli Jacobiani, sebbene essi siano, in generale, differenti ad ogni passo.

Si definisce l'errore di predizione come:

$$\tilde{e}_{x_k} \equiv x_k - \tilde{x}_k \tag{2.14}$$

e l'errore di misura come:

$$\tilde{e}_{z_k} \equiv z_k - \tilde{z}_k \tag{2.15}$$

Si ricorda che nella pratica non si ha accesso direttamente ad x_k, cioè al vettore di stato attuale, che è la quantità da stimare. E' possibile invece usare le z_k per tentare di stimare x_k. Dalle Eqs. (2.12) e (2.13) possiamo scrivere le (2.14) e (2.15) come:

$$\begin{aligned} \tilde{e}_{x_k} &= F(x_{k-1} - \hat{x}_{k-1}) + \epsilon_k \\ \tilde{e}_{z_k} &= H\tilde{e}_{x_k} + \eta_k \end{aligned} \tag{2.16}$$

dove ϵ_k ed η_k rappresentano nuove variabili aleatorie aventi media nulla e matrice di covarianza WQW^T e VRV^T rispettivamente, dove Q ed R sono le matrici di covarianza dei processi aleatori ω e v.

Si nota che le Eqs in (2.16) sono lineari e molto simili alle equazioni del filtro di Kalman lineare, per cui si può usare l'errore di misura $\tilde{e}_{z_k}$ dato dalla (2.15) ed un secondo ed ipotetico filtro di Kalman per stimare l'errore di predizione $\tilde{e}_{x_k}$.

Questa stima che si indicherà con $\hat{e}_k$, potrà essere utilizzata per ottenere una stima a posteriori per il processo non lineare di partenza come:

$$\hat{x}_k = \tilde{x}_k + \hat{e}_k \tag{2.17}$$

Ora, assumendo che il valore predetto di $\hat{e}_k$ sia zero, l'equazione del filtro di Kalman usata per stimare $\hat{e}_k$ è:

$$\hat{e}_k = K_k \tilde{e}_{z_k} \tag{2.18}$$

Sostituendo la Eq. (2.18) nella (2.17) ed usando la (2.15) si ha:

$$\hat{x}_k = \tilde{x}_k + K_k(z_k - \tilde{z}_k) \tag{2.19}$$

L'Eq. (2.19) può essere utilizzata per la correzione del filtro di Kalman esteso, con $\tilde{x}_k$ e $\tilde{z}_k$ date dalle (2.10) e (2.11) ed il guadagno di Kalman K_k dato dalla (2.5). Il set completo delle equazioni è descritto nel Par. 2.3.2 e sintetizzato in Fig. 2.3.

Una importante caratteristica dell'EKF è che lo Jacobiano H_k nell'equazione del guadagno di Kalman K_k serve a propagare correttamente o ad amplificare solo le componenti rilevanti dell'informazione di misura. Ad esempio, se non c'è una corrispondenza uno a uno tra le misure z_k e lo stato attraverso la funzione di uscita h, lo Jacobiano H_k influisce sul guadagno di Kalman in modo da amplificare solo la porzione del residuo $z_k - h(\hat{x}_{k/k-1}, 0)$ che influisce sullo stato.

2.3.2 Implementazione dell'EKF

Dato un sistema non lineare nella forma:

$$\begin{cases} x_k &= f(x_{k-1}, u_{k-1}, \omega_{k-1}) \\ z_k &= h(x_k, v_k) \end{cases} \tag{2.20}$$

il processo di osservazione avviene in due fasi:

1. **Time update**

   - vengono generate le proiezioni dello stato del sistema, a partire dalla conoscenza del modello e della stima dello stato precedente:

$$\hat{x}_{k/k-1} = f(\hat{x}_{k-1}, u_{k-1}, 0) \tag{2.21}$$

- vengono aggiornati i parametri probabilistici di errore sulla misura dello stato:

$$P_{k/k-1} = F_k P_{k-1} F_k^T + Q_k \tag{2.22}$$

2. **Measurement update**

- vengono corrette le stime prodotte al passo precedente, sulla base delle osservazioni delle uscite:

$$K_k = P_{k/k-1} H_k^T (H_k P_{k/k-1} H_k^T + R_k) \tag{2.23}$$

$$\hat{x}_k = \hat{x}_{k/k-1} + K_k(z_k - h(\hat{x}_{k/k-1}, 0)) \tag{2.24}$$

- vengono aggiornati i parametri probabilistici di errore sulla misura delle uscite:

$$P_k = (I - K_k H_k) P_{k/k-1} \tag{2.25}$$

dove:

- $\hat{x}_{k-1}$ è lo stato stimato al passo precedente (stima corrente);
- $\hat{x}_{k/k-1}$ è lo stato predetto, nota la stima corrente;
- z_k sono le misure;
- $h(\hat{x}_{k/k-1}, 0)$ è l'uscita predetta dal filtro;
- $\hat{x}_k$ è lo stato stimato dal filtro;
- F_k è lo Jacobiano di f rispetto ad x, calcolato in $(\hat{x}_k, u_k, 0)$, cioè attorno alla stima corrente;
- H_k è lo Jacobiano di h rispetto ad x, calcolato in $(\hat{x}_{k/k-1}, 0)$, cioè attorno alla predizione.

2.4 Attitude and Heading Reference System

L'*Attitude and Heading Reference System*, meglio noto come AHRS, è un software che combina informazioni provenienti da diversi sensori, quali giroscopi, accelerometri e magnetometri triassiali, al fine di ricavare informazioni per l'assetto e la direzione di volo di un velivolo che si muova nello spazio libero.

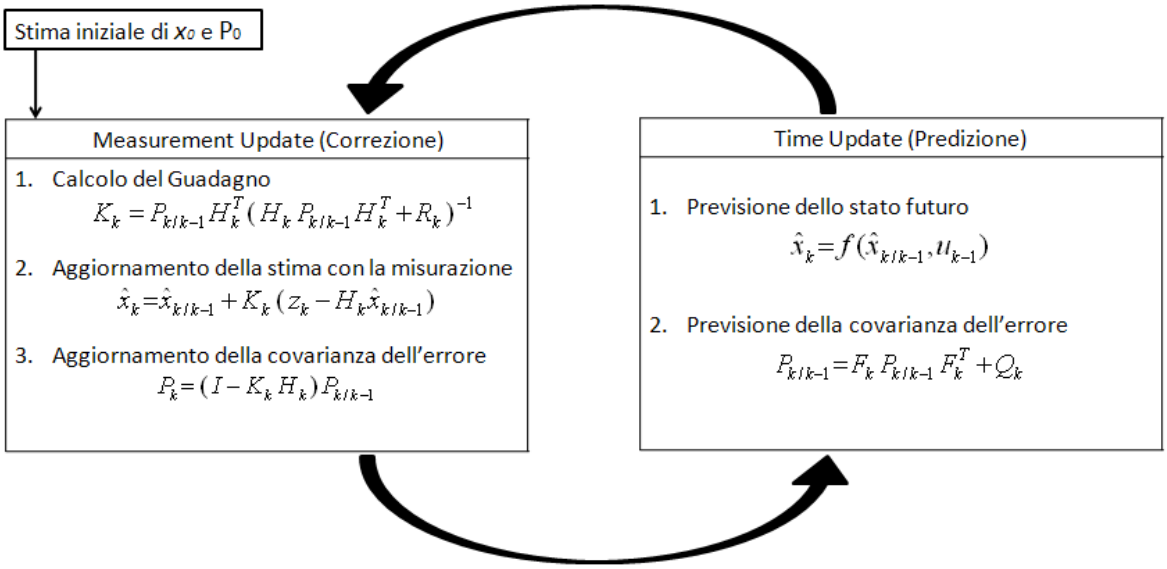


Figura 2.3: Algoritmo che compone il Filtro di Kalman esteso.

Un filtro di Kalman è solitamente utilizzato per elaborare i dati provenienti da diverse fonti ed ottenere la soluzione richiesta. Dall'analisi presentata nei paragrafi precedenti si evince che, per implementare praticamente uno stimatore basato su Filtro di Kalman, occorre esprimere un modello matematico che descriva la dinamica delle grandezze da stimare (utilizzato nel passo di predizione del filtro) e individuare il set di grandezze misurate da utilizzare per correggere la stima.

2.4.1 Stimatore standard

Lo stimatore proposto in [12] e sviluppato in [4] rappresenta la soluzione maggiormente utilizzata in letteratura al problema della stima d'assetto. La sua struttura di base è riportata in Fig. 2.4.

Il funzionamento di tale AHRS è basato sull'integrazione delle equazioni differenziali che descrivono l'evoluzione nel tempo dell'assetto in funzione delle velocità angolari misurate da giroscopi triassiali installati solidalmente al velivolo. In questo modo gli angoli di assetto ϕ, θ e ψ vengono stimati ed aggiornati ad alta frequenza (50 Hz). Principalmente a causa degli errori di misura dei giroscopi, si sviluppano degli errori di assetto che crescono rapidamente nel tempo.

Così, ad opportuni intervalli di tempo (25 Hz), dalle misure degli accelerometri si effettua una rilevazione del vettore gravità, mediante il quale si ricava una correzione per gli angoli di assetto, in modo da correggere eventuali errori che si sviluppano dall'integrazione delle velocità angolari. Il filtro di Kalman elabora i dati provenienti dalla previsione del modello dinamico e dalle misure dell'assetto, per ottenere degli angoli stimati ed aggiornati ad alta frequenza ed affetti da errori limitati. Dagli angoli di rollio ϕ e di beccheggio θ forniti dal filtro, il vettore campo magnetico terrestre, misurato da magnetometri triassiali, viene ruotato dalla terna di riferimento solidale al velivolo alla terna geografica. Dalla proiezione del vettore campo magnetico sul piano parallelo a quello tangente alla superficie terrestre, si determina la prua magnetica del velivolo, la cui misura corregge la stima dell'angolo di imbardata o heading, aggiornando i dati in bassa frequenza (10 Hz). Naturalmente, dovendosi riferire all'heading geografico e non magnetico[1], e per coerenza con i riferimenti di posizione forniti dal GPS appunto nel sistema geografico, si considera la correzione mediante l'angolo di declinazione magnetica.

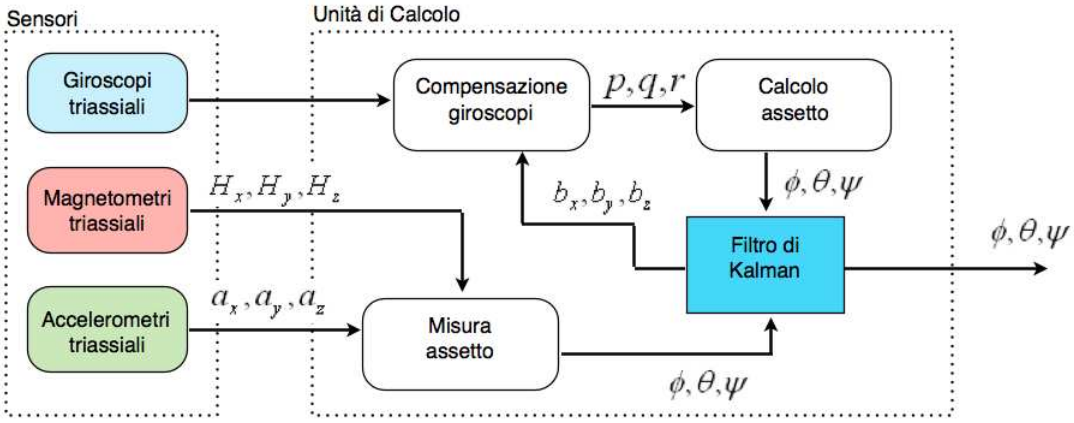


Figura 2.4: Soluzione dell'Attitude and Heading Reference System standard.

Le misure di accelerazione (accelerometri), campo magnetico (magnetometri) e velocità angolare (giroscopi) sono tipicamente corrotte da errori di bias e rumori di misura. Mentre questi ultimi possono essere filtrati da uno stimatore ben

[1]La direzione di volo del velivolo potrebbe anche essere riferita al nord magnetico invece che al nord geografico, poiché data la nostra posizione geografica e considerando che il velivolo privo di pilota può essere in volo per intervalli di tempo ridotti, non si commetterebbero grossi errori.

calibrato, i bias possono essere stimati come se fossero stati del sistema. Nell'implementazione del filtro di Kalman viene, dunque, effettuata la stima degli errori di *bias* dei giroscopi, i quali rientrano nelle principali cause di errore nel calcolo dell'assetto poiché ne comportano la deriva. I giroscopi, infatti, hanno delle polarizzazioni che possono essere supposte costanti[2] e, se le misure vengono integrate, producono un errore crescente nel tempo. E' necessario allora stimare tali polarizzazioni, o bias, in modo da compensarle.

Modello del sistema

La rappresentazione più comune dell'assetto di un velivolo è quella attraverso la matrice dei coseni direttori, ossia una matrice reale ortogonale che mappa vettori da un sistema di riferimento inerziale al sistema di riferimento assi corpo. Tale matrice richiede nove parametri, dei quali solo tre sono indipendenti, per cui gli angoli di Eulero forniscono la rappresentazione con il numero di parametri minimale. La matrice dei coseni direttori ricavata a partire da Eulero è stata discussa nel Par. 1.3.1. Nonostante il significato chiaro ed intuitivo, tale rappresentazione esibisce delle singolarità per certi angoli, per cui la rappresentazione dell'assetto più conveniente anche se non minima è quella mediante quaternioni (Par. 1.4) [21].

La matrice dei coseni direttori ad essi associata è nel seguito riportata:

$$C_{b/n} = \begin{bmatrix} (q_0^2 + q_1^2 - q_2^2 - q_3^2) & 2(q_1 q_2 + q_0 q_3) & 2(q_1 q_3 - q_0 q_2) \\ 2(q_1 q_2 - q_0 q_3) & (q_0^2 - q_1^2 + q_2^2 - q_3^2) & 2(q_2 q_3 + q_0 q_1) \\ 2(q_1 q_3 + q_0 q_2) & 2(q_2 q_3 - q_0 q_1) & (q_0^2 - q_1^2 - q_2^2 + q_3^2) \end{bmatrix} \quad (2.26)$$

Per quanto riguarda, invece, la dinamica dell'assetto la scelta ricade sulle equazioni cinematiche (1.16) descritte nel Par. 1.5, in quanto esse esprimono, in modo semplice, la dinamica degli angoli d'assetto in funzione delle velocità angolari del sistema. Anche in questo caso, in realtà, viene utilizzata una rappresentazione con i quaternioni, in quanto tale formulazione conduce ad una descrizione più accurata e stabile (in termini di calcolo numerico) della dinamica del filtro. Le relazioni che legano gli angoli di Eulero con i quaternioni sono riportate nel Par. 1.4.2. In ogni caso, le uscite finali dello stimatore sono rappresentate dai tre angoli di Eulero.

[2] In realtà le polarizzazioni dipendono dalla temperatura per cui non sono costanti nel tempo. Si può ipotizzare però che la temperatura non vari nell'arco della durata del volo e che dunque le polarizzazioni siano costanti.

La dinamica dell'assetto di un corpo rigido, utilizzando i quaternioni, è quindi data da [13]:

$$\dot{q} = \frac{1}{2}\Omega(\omega)q \tag{2.27}$$

dove:

$$\Omega(\omega) = \begin{bmatrix} 0 & -P & -Q & -R \\ P & 0 & R & -Q \\ Q & -R & 0 & P \\ R & Q & -P & 0 \end{bmatrix} \tag{2.28}$$

Sui quaternioni deve essere soddisfatto il vincolo di norma unitaria, ($\|q\| = 1$), per cui ogni volta che i quaternioni verranno aggiornati nell'algoritmo, sarà necessario effettuare la normalizzazione.

Poiché la stima di q viene da una integrazione delle velocità misurate con i giroscopi è necessario compensare la polarizzazione di questi sensori per ottenere stime più realistiche. Per questa ragione, il vettore di stato è:

$$x = \begin{bmatrix} q \\ b \end{bmatrix} \tag{2.29}$$

dove q sono i quaternioni stimati e b sono le stime delle polarizzazioni (o bias) dei giroscopi. La polarizzazione sui giroscopi è modellata come evento randomatico in modo da avere $\dot{b} = 0$.

Filtro di Kalman esteso per AHRS

Il sistema al quale applicare il filtro di Kalman esteso è descritto da:

$$\begin{cases} \dot{x} = f(x, \omega) + w \\ z = h(x) + v \end{cases} \tag{2.30}$$

dove $x \in \mathbf{R}^{7\times1}$ è lo stato del sistema, composto da $q \in \mathbf{R}^{4\times1}$ e da $b \in \mathbf{R}^{3\times1}$, ed $\omega \in \mathbf{R}^{3\times1}$ sono le velocità angolari [12]. Inoltre, abbiamo che:

$$z = \begin{bmatrix} a \\ \psi_{mag} \end{bmatrix} \tag{2.31}$$

dove $a \in \mathbf{R}^{3\times1}$ sono le misure di accelerazione prodotte dagli accelerometri nel sistema assi corpo, e ψ_{mag} è l'assetto magnetico del velivolo. Il rumore di processo

è dato da w, mentre il rumore sulla misura è dato da v_1. Le funzioni non lineari $f(x,\omega)$ e $h(x)$ sono date da:

$$f(x,\omega) = \begin{bmatrix} \frac{1}{2}\begin{bmatrix} -q_1 & -q_2 & -q_3 \\ q_0 & -q_3 & q_2 \\ q_3 & q_0 & -q_1 \\ -q_2 & q_1 & q_0 \end{bmatrix} \begin{bmatrix} P - b_p \\ Q - b_q \\ R - b_r \end{bmatrix} \\ \mathbf{0}^{3\times 1} \end{bmatrix} \tag{2.32}$$

$$h(x) = \begin{bmatrix} 2g(q_1q_3 - q_0q_2) \\ 2g(q_2q_3 + q_0q_1) \\ g({q_0}^2 - {q_1}^2 - {q_2}^2 + {q_3}^2) \\ \tan^{-1}\left(\dfrac{2(q_1q_2 + q_0q_3)}{{q_0}^2 + {q_1}^2 - {q_2}^2 - {q_3}^2}\right) \end{bmatrix} \tag{2.33}$$

Nel presente filtro la stima dell'assetto è corretta e migliorata attraverso le misure fornite da accelerometri e magnetometri. La correzione si basa sul calcolo del residuo ossia sulla differenza tra le generica misura e la stessa grandezza stimata tramite il modello. In particolare avendo a disposizione le misure prodotte dagli accelerometri, le prime tre componenti della funzione $h(x)$ rappresentano le corrispondenti grandezze stimate dal modello lungo i tre assi. Tali grandezze altro non sono che il campo gravitazione riportato nel piano assi corpo attraverso la matrice dei coseni direttori.

La scelta di modellare le misure accelerometriche mediante il solo campo gravitazione nasce dalla seguente osservazione. Gli accelerometri sono sensibili alla forza di gravità ed alle accelerazioni a cui il velivolo è soggetto rispetto ad un riferimento inerziale. Quando il velivolo è in volo di crociera, ossia si muove di moto rettilineo uniforme, le misure degli accelerometri indicheranno in prevalenza il vettore gravità[3]. Naturalmente, il modello soffre di una limitazione: nei casi in cui l'accelerazione del velivolo si sovrappone all'accelerazione di gravità, il residuo cresce e la stima dell'assetto potrebbe subire una degradazione in particolare nelle forti virate. Tale problematica sarà discussa ampiamente nel capitolo successivo.

[3]Un accelerometro, in realtà, non rileva l'accelerazione gravitazionale, bensì rileva una deviazione dalla caduta libera. Per cui se l'accelerometro è fermo o si muove di moto rettilineo uniforme misurerà una accelerazione di $1\mathbf{g}$ verso l'alto; al contrario se è in caduta libera misurerà zero.

L'ultima componente della funziona $h(x)$ è la modellazione dell'heading magnetico ψ_{mag} attraverso i quaternioni (Par. 1.4.2), che sarà confrontato con l'heading magnetico misurato attraverso i magnetometri.

Formulazione del modello

Il filtro di Kalman è stato realizzato in forma estesa e discreta. Il modello matematico che descrive il sistema presenta delle non linearità dovute alle equazioni differenziali delle componenti del quaternione. Il filtro va implementato in forma estesa, ossia basato sulla linearizzazione del modello attorno alla stima corrente, non potendo individuare a priori una traiettoria nominale per effettuare una buona approssimazione lineare del modello. Inoltre, per rendere il sistema direttamente realizzabile a livello pratico, il filtro di Kalman è realizzato in forma discreta.

Per la linearizzazione occorre definire lo Jacobiano della matrice delle dinamiche e delle misure. Queste matrici sono valutate nella stima corrente, rispettivamente, dello stato e delle misure. Esse sono definite come:

$$F_k = \left.\frac{\partial f}{\partial x}\right|_{x=\hat{x}_k,\omega=\omega_k} \qquad H_k = \left.\frac{\partial h}{\partial x}\right|_{x=\hat{x}_k} \tag{2.34}$$

dove $\hat{x}_k$ è la stima dello stato e ω_k è il vettore delle velocità angolari misurate, entrambe valutate al tempo k.

Lo Jacobiano F_k è dato da:

$$F_k = \frac{\partial f}{\partial x} = \begin{bmatrix} F_{11} & F_{12} \\ \mathbf{0}^{3\times4} & \mathbf{0}^{3\times3} \end{bmatrix} \tag{2.35}$$

dove

$$F_{11} = \frac{\partial}{\partial q}\left\{\Omega(\omega - b)q\right\} = \Omega(\omega - b)$$

e

$$F_{12} = \frac{\partial}{\partial b}\left\{\Omega(\omega - b)q\right\} = \frac{1}{2}\begin{bmatrix} q_1 & q_2 & q_3 \\ -q_0 & q_3 & -q_2 \\ -q_3 & -q_0 & q_1 \\ q_2 & -q_1 & -q0 \end{bmatrix}.$$

La matrice H_k introdotta nella (2.34) è, invece, data da:

$$H_k = \frac{\partial h}{\partial x} = \begin{bmatrix} H_{11} & \mathbf{0}^{3\times1} \\ H_{21} & \mathbf{0}^{3\times1} \end{bmatrix} \tag{2.36}$$

dove

$$H_{11} = \frac{\partial a}{\partial q} = \begin{bmatrix} 2gq_2 & -2gq_3 & 2gq_0 & -2gq_1 \\ -2gq_1 & -2gq_0 & -2gq_3 & -2gq_2 \\ -2gq_0 & -2gq_1 & 2gq_2 & -2gq_3 \end{bmatrix} \tag{2.37}$$

e

$$H_{21} = \frac{\partial \psi_{mag}}{\partial q} = \frac{\partial}{\partial q} \left\{ tan^{-1} \left(\frac{2(q_1 q_2 + q_0 q_3)}{{q_0}^2 + {q_1}^2 - {q_2}^2 - {q_3}^2} \right) \right\} = \frac{2\beta}{\alpha^2 + \beta^2} \begin{bmatrix} q_3 \\ q_2 \\ q_1 + 2q_2\alpha \\ q_0 + 2q_3\alpha \end{bmatrix}^T \tag{2.38}$$

con

$$\alpha = 2(q_1 q_2 + q_0 q_3) \qquad \beta = 1 - 2({q_2}^2 + {q_3}^2).$$

Discretizzazione

Il modello utilizzato dal filtro di Kalman discreto necessita di un modello del tipo:

$$\begin{aligned} x_{k+1} &= \Phi_k x_k \\ z_k &= H_k x_k \end{aligned} \tag{2.39}$$

Φ è la matrice di transizione data da:

$$\Phi_k = e^{F_k T} \tag{2.40}$$

dove $T = t_k - t_{k-1}$ è il periodo di campionamento. Sviluppando l'espressione in serie di Taylor arrestata al primo ordine:

$$\Phi_k \approx I + F_k T. \tag{2.41}$$

L'algoritmo stima così sintetizzato è stato implementato in linguaggio C (Appendice A).

Correzione magnetica

Il sistema di navigazione usa anche il campo magnetico terrestre per determinare la direzione di volo del velivolo.

Il campo magnetico terrestre ha sempre una componente parallela alla superficie della terra e diretta verso il nord magnetico: questa caratteristica è alla base del funzionamento della bussola magnetica. Il campo, infatti, può essere approssimato con il modello di un dipolo, come mostrato in Fig. 2.5: le linee di campo si originano in un punto vicino al polo sud e terminano in uno vicino al polo nord (punti detti poli magnetici) e possono variare in modulo e direzione attorno alla superficie della terra. La direzione ed il modulo del campo magnetico H_e può essere rappresentato dalla componenti H_x, H_y e H_z sui tre assi, ma solo le prime due contribuiscono al calcolo dell'heading rispetto al nord magnetico, mentre la componente verticale è ignorata [22].

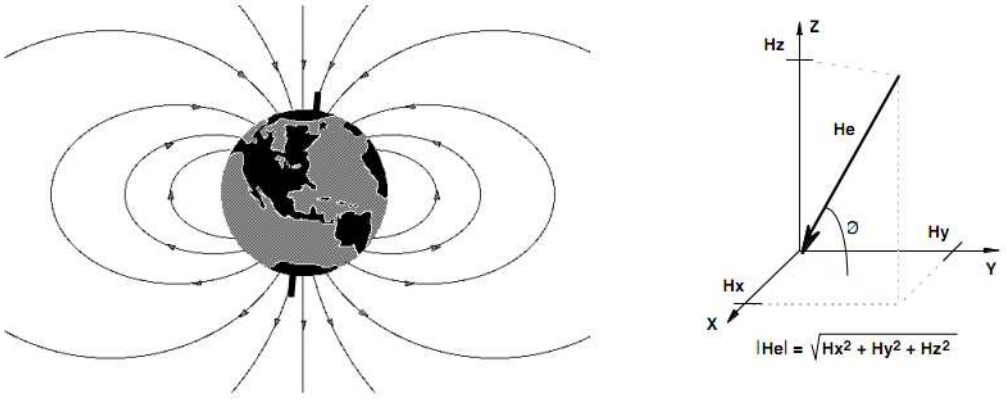


Figura 2.5: Campo magnetico terrestre

Se il velivolo viaggiasse sempre su un piano parallelo a quello tangente alla superficie della terra, le misure X_h ed Y_h nel piano orizzontale coinciderebbero con le componenti H_x ed H_y misurate dal magnetometro nel sistema assi corpo solidale. Sapendo che il campo magnetico punta sempre verso nord, se si suppone di partire con il velivolo orientato verso tale direzione allora X_h sarebbe al suo valore massimo mentre Y_h sarebbe nulla[4]. Se il velivolo iniziasse poi a girare in senso orario verso est, allora X_h inizierebbe a diminuire fino a zero mentre Y_h a crescere verso il suo massimo positivo. Continuando, se il velivolo fosse rivolto a sud, allora X_h sarebbe al suo valore massimo negativo, mentre Y_h ritornerebbe a zero. Dopo un giro completo, l'andamento delle componenti è mostrato in Fig. 2.6 [22].

[4]Si ricorda che il sistema di riferimento assi corpo è tale per cui l'asse X_b è diretto lungo la fusoliera del velivolo e l'asse Y_b è alla sua destra, lungo la direzione dell'ala.

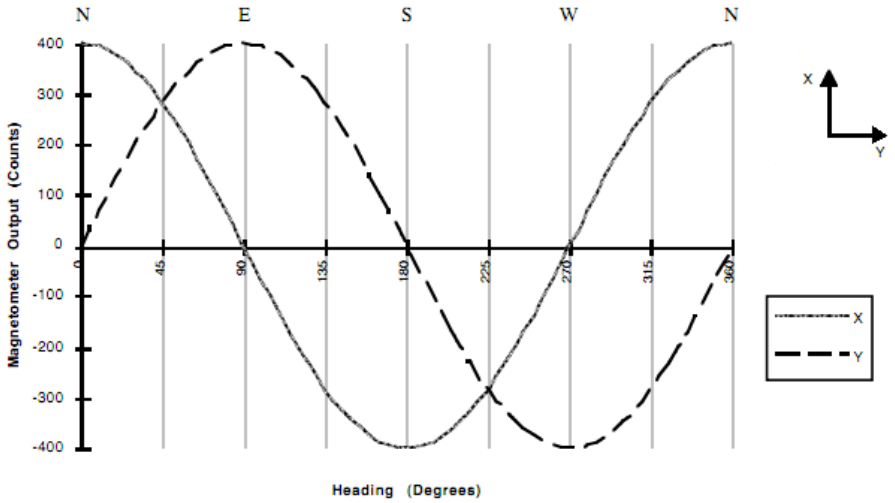


Figura 2.6: Letture H_x ed H_y del magnetometro per differenti angoli di heading.

Si ricava, dunque; che:

$$\begin{aligned} X_h &= \cos\psi_{mag} \\ Y_h &= -\sin\psi_{mag} \end{aligned} \tag{2.42}$$

ossia

$$\psi_{mag} = \tan^{-1}\left(\frac{-Y_h}{X_h}\right) \tag{2.43}$$

La (2.43) fornisce il calcolo dell'angolo di imbardata del velivolo rispetto al nord magnetico.

Nella realtà, però, il velivolo è soggetto a manovre ed il sensore che misura il campo magnetico non è posto su una superficie piana, parallela alla superficie della terra. In tal caso risulta più difficile calcolare la direzione di volo, poiché si introducono errori dovuti ad angoli di inclinazione che possono essere corretti conoscendo gli angoli di rollio ϕ e di beccheggio θ (Fig. 2.7) [23].

Note le misure H_x, H_y e H_z del magnetometro, il campo magnetico può essere ruotato di un angolo pari a $-\phi$ attorno all'asse X_b del sistema assi corpo, e poi di un angolo $-\theta$ attorno all'asse Y' della nuova terna. In questo moto il velivolo sarà allineato al piano orizzontale, determinando le proiezioni:

$$\begin{aligned} X_h &= H_x\cos\theta + H_y\sin\phi\sin\theta + H_z\cos\phi\sin\theta \\ Y_h &= H_y\cos\phi - H_z\sin\phi \end{aligned} \tag{2.44}$$

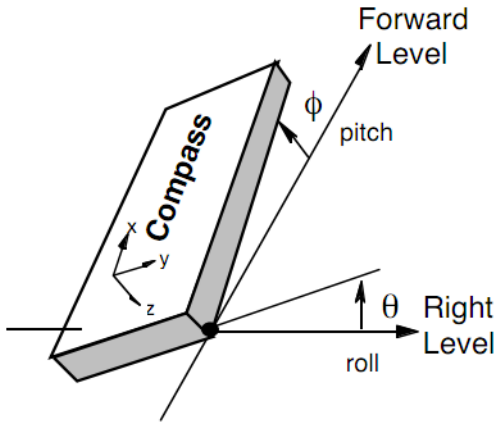


Figura 2.7: Inclinazione del sistema rispetto al piano orizzontale terrestre

L'Eq. (2.43) ancora determina la direzione di volo del velivolo rispetto al nord magnetico, a partire dalle misure del magnetometro. A questo punto, per riferirsi al nord geografico, basta effettuare la correzione con l'angolo di declinazione ψ_d, ossia l'angolo tra il nord magnetico e quello geografico.

$$\psi = \psi_{mag} + \psi_d \tag{2.45}$$

ricordando che tale shift è positivo se il velivolo si muove nell'emisfero est, negativo se si muove nell'emisfero ovest.

Capitolo 3

Algoritmo di stima multi-mode

Il seguente capitolo affronta le problematiche connesse all'utilizzo di uno stimatore standard e si propone di realizzare un AHRS basato su più modalità di funzionamento. L'obiettivo è cercare di superare i limiti imposti dalla soluzione presentata al capitolo precedente, ottenendo un algoritmo di stima che sia in grado di migliorare le prestazioni, in particolar modo, in condizioni dinamiche di volo.

3.1 Introduzione

Il filtro di Kalman esteso proposto nel capitolo precedente ed utilizzato per la stima dell'assetto del velivolo, adottando l'ipotesi di stazionarietà del volo, può portare a prestazioni non soddisfacenti [4]. Questo ha motivato molti autori, in letteratura, a cercare soluzioni migliori e più performanti.

Al fine di utilizzare in modo ottimale il set di sensori a disposizione, in [24] e in [25], ad esempio, è effettuata la stima dell'assetto utilizzando l'*Unscented Kalman Filter* (UKF) combinato con l'algoritmo *Three Axis Attitude Determination* (TRIAD). Il TRIAD è un metodo deterministico per calcolare direttamente la matrice dei coseni direttori a partire da un set di vettori di misure [26], mentre l'UKF utilizza una tecnica di campionamento deterministico (*unscented trasformation*) in grado individuare un set minimo di punti (*sigma points*), relativi alla media della variabile casuale da stimare, attraverso i quali è possibile calcolare in modo più accurato la matrice di covarianza P [27]. Altri lavori ([28] e [29]) utilizzano un approccio *adattativo* per modificare dinamicamente la matrice di covarianza delle misure R. In questo modo, infatti, è possibile basare la stima

principalmente sulla parte predittiva del modello, associando un peso minore alle misurazioni che, in caso di condizioni non stazionarie di volo, possono introdurre delle imprecisioni. Puntando, invece, su un filtro in grado di predire in modo più accurato la dinamica del velivolo, in [30] è implementato un *filtro di Kalman moltiplicativo*. Tale approccio, correggendo il calcolo dei quaternioni in modo da rispettare il vincolo di norma unitaria, senza esplicitare l'operazione di normalizzazione, è in grado di fornire una stima corretta anche su un set di traiettorie che si discostano da quella stazionaria. Infine altre metodologie, non basate su filtro di Kalman, adoperano il *nonlinear model predictive control* [31] o le *reti neurali* [32]. Tutti questi approcci, però, complicano significativamente la struttura dello stimatore.

L'obiettivo del presente lavoro è quello di trovare una soluzione più performante, partendo dalla soluzione standard attualmente in uso, senza tuttavia modificarne la struttura complessiva. Per questo motivo l'attenzione è stata focalizzata sulle problematiche connesse allo stimatore standard, in modo da individuare i punti sui quali è possibile agire per migliorare le prestazioni dell'algoritmo di stima.

L'AHRS presentato al capitolo precedente, così come altri esempi presenti in letteratura (cfr. [12], [33], [34], [29]) utilizzano le misure provenienti da accelerometri e magnetometri per correggere la previsione della stima. In queste circostanze i problemi principali sorgono dall'utilizzo delle misure provenienti dagli accelerometri. Il vettore accelerazione, infatti, può essere utilizzato per stimare l'assetto solo se esso coincide con il vettore di gravità terrestre. Questo è vero solo se il velivolo è a terra o si muove di moto rettilineo uniforme, altrimenti il vettore di accelerazione misurato conterrà al suo interno anche altri contributi di accelerazione dinamica (come nel caso di forti virate) e non potrà essere confrontato correttamente con la relativa equazione d'uscita (2.24) che, invece, modella semplicemente l'accelerazione gravitazionale. Tra i lavori che utilizzano tale struttura di filtro, solo [33] e [34] asseriscono che tale condizione di stazionarietà o quasi-stazionarietà è vera solo per elicotteri e quadricotteri.

In [29] è proposta un correzione delle misure di accelerazione complicando l'equazione d'uscita del filtro, in modo da rendere il vettore misurato più simile a quello di gravità. Tale correzione, però, non può essere effettuata correttamente in quanto la nuova equazione d'uscita, ricavata a partire dalle equazioni (1.10) del Par. 1.5, richiederebbe delle informazioni non disponibili direttamente, e potrebbe solo essere approssimata utilizzando la misura di airspeed proveniente dal sensore

di pressione per modellare la velocità lineare longitudinale e trascurando tutte le altre grandezze. In [35] tale modifica è stata testata ma non ha portato a miglioramenti significativi.

Un altro punto su cui occorre focalizzare l'attenzione è relativo alla stima dei bias sui giroscopi. Nella precedente trattazione questi sono stati supposti costanti, nell'ipotesi che la durata del volo sia tanto breve da non coinvolgere cambi di temperatura. Un passo avanti sarebbe quello di considerare tali bias lentamente variabili, così da descrivere in modo più preciso la natura del fenomeno, e vedere come sia possibile migliorare l'algoritmo di stima in tali circostanze [33].

3.2 Definizione della struttura

Dopo aver individuato le principali fonti di imprecisione della soluzione standard, occorre effettuare altre analisi per definire un nuovo modello dello stimatore.

Il filtro di Kalman esteso costituisce ancora il metodo di filtraggio utilizzato cosicché sarà possibile riprendere gran parte del lavoro precedentemente sviluppato. La rappresentazione dell'assetto continuerà, dunque, ad essere espressa in termini di quaternioni, in modo da evitare singolarità ed utilizzare una equazione dinamica semplice e che porti ad una formulazione accurata del filtro. Lo stato del sistema sarà, quindi, ancora costituito dai quattro quaternioni e dai bias sui giroscopi che, come già detto, è necessario stimare per evitare effetti di deriva nel corso della stima.

3.2.1 Analisi di osservabilità

Per giungere a migliorie in termini di prestazione per la stima, bisogna effettuare uno studio completo sul numero minimo di vettori di misura (e quindi il numero di variabili di uscita del modello del filtro) che rendano possibile la stima dello stato del filtro. E' stata effettuata quindi un'analisi di osservabilità del sistema linearizzato al fine di capire quante misure, tra quelle disponibili, è necessario considerare. Lo stato del sistema è composto dai quattro quaternioni e dai tre bias sui giroscopi, per un totale di 7 variabili di stato. Utilizzando una solo vettore di tre componenti (es. solo accelerazioni, campo magnetico o GPS) per modellare l'uscita del sistema, il rango della matrice di osservabilità è 5. Se, invece, si utilizza una coppia di vettori (es. accelerazioni e GPS, accelerazioni e

campo magnetico o campo magnetico e GPS) il rango sale a 7 e il sistema diventa osservabile.

Quest'analisi dimostra che la stima dell'assetto e dei bias dei giroscopi richiede almeno due vettori di misura indipendenti. Con le considerazioni precedentemente effettuate, riguardante la problematica relativa agli accelerometri, si può giungere alla conclusione che conviene utilizzare congiuntamente le misure di campo magnetico e GPS, ricorrendo agli accelerometri solo quando il GPS non è disponibile. In questo modo si riduce al minimo l'utilizzo del vettore accelerazione.

Queste considerazioni conducono all'idea di realizzare un filtro multi-mode che, nella parte relativa alla correzione, può commutare tra due coppie di segnali di misura a seconda della necessità [35].

3.2.2 Modi del filtro

A questo punto è possibile definire i differenti modi di funzionamento del filtro e la strategia per passare da un modo ad un altro, in base all'analisi svolta in precedenza. I modi di funzionamento individuati sono due:

- MODO 1: *Volo con GPS*, dove la stima è basata su magnetometri e misure GPS.
- MODO 2: *Volo senza GPS*, dove la stima basata su magnetometri e accelerometri.

La condizione per passare da un modo all'altro è rappresentata, semplicemente, dalla perdita del segnale GPS (da 1 a 2) o dal ritorno di un segnale GPS valido (da 2 a 1).

3.2.3 Struttura complessiva

Lo schema della struttura complessiva dello stimatore è rappresentata in Fig. 3.1: l'integrazione delle equazioni differenziali che descrivono l'evoluzione nel tempo dell'assetto avviene ad alta frequenza (50 Hz). Tale integrazione fornisce la previsione degli angoli di assetto ϕ, θ e ψ. La fase di correzione, necessaria a causa degli errori di misura dei giroscopi e delle approssimazioni introdotte dal modello, viene effettuata ad opportuni intervalli di tempo (25 Hz): se è disponibile un segnale GPS valido allora i magnetometri effettuano una rilevazione del campo

magnetico terrestre mentre il GPS è utilizzato per misurare l'orientamento del velivolo; altrimenti il GPS è rimpiazzato dalla rilevazione del vettore di gravità effettuata dagli accelerometri. Da queste misurazioni si ricava una correzione alla previsione precedentemente calcolata e la stima degli angoli di assetto viene fornita in uscita al filtro.

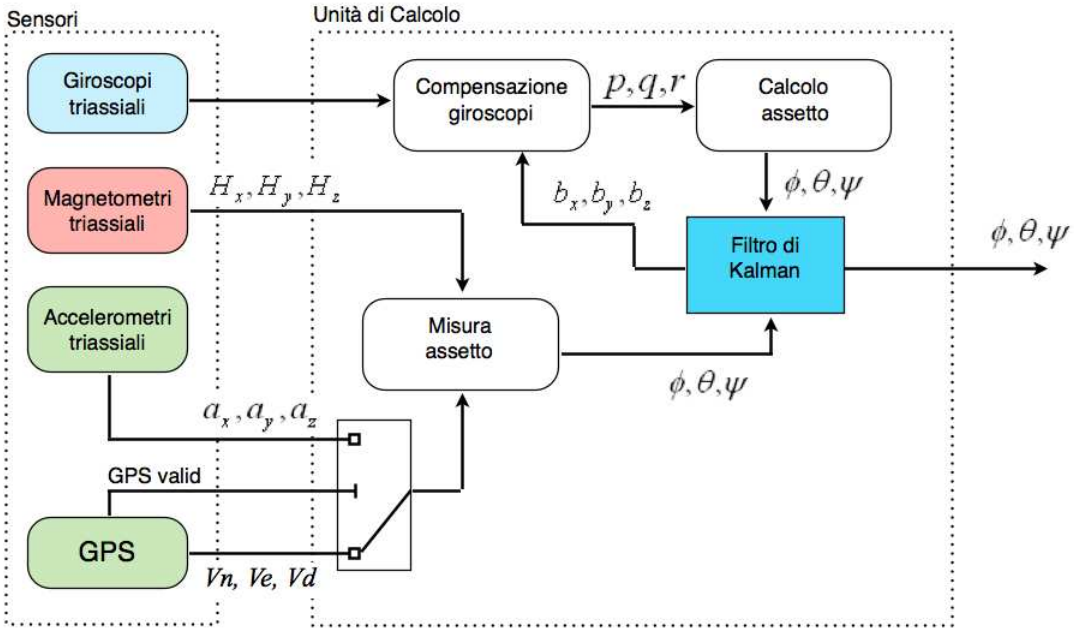


Figura 3.1: Struttura del nuovo Attitude and Heading Reference System.

3.3 Modellazione del sistema

Si vogliono, ora, individuare le equazioni che andranno a descrivere la parte dinamica dello stimatore e le relative variabili di stato.

3.3.1 Dinamica dei bias

Il bias, o errore di bias, di un giroscopio è il segnale che il giroscopio fornisce in uscita quando quest'ultimo non sta eseguendo nessuna rotazione. In generale i bias vengono considerati costanti ma nella realtà essi tendono a variare a causa di cambiamenti di temperatura oppure quando i giroscopi vengono utilizzati per un periodo di tempo molto lungo [36]. Benché i micro UAV siano caratterizzati da un

tempo di volo relativamente breve (in genere minore di un'ora) [7], è comunque possibile che essi vadano incontro a cambi di temperatura in grado di causare variazioni nelle polarizzazioni dei giroscopi [33]. Per questo motivo bisogna far in modo che il nuovo filtro sia in grado di stimare tali variazioni. L'obiettivo è quello di stimare anche bias lentamente variabili. La dinamica dei bias sarà, dunque, modellata come un sistema guidato da un rumore Gaussiano bianco a media nulla [35]:

$$\dot{b} = v^b \tag{3.1}$$

dove $b = \begin{bmatrix} b_P & b_Q & b_R \end{bmatrix}^T$ è il vettore dei bias e $v^b = \begin{bmatrix} v_{bP} & v_{bQ} & v_{bR} \end{bmatrix}^T$ rappresenta il rumore sui bias dei giroscopi. In questo modo, tenendo conto anche di questo rumore nell'implementazione del modello stocastico del filtro, è possibile giungere ad una descrizione dinamica della matrice di covarianza di processo Q, in grado di adattarsi dinamicamente alle variazioni dei bias.

3.3.2 Equazioni cinematiche

La rappresentazione dell'assetto è ricavata a partire dalla matrice dei coseni direttori scritta in termini di quaternioni:

$$C_{b/n} = \begin{bmatrix} (q_0^2 + q_1^2 - q_2^2 - q_3^2) & 2(q_1q_2 + q_0q_3) & 2(q_1q_3 - q_0q_2) \\ 2(q_1q_2 - q_0q_3) & (q_0^2 - q_1^2 + q_2^2 - q_3^2) & 2(q_2q_3 + q_0q_1) \\ 2(q_1q_3 + q_0q_2) & 2(q_2q_3 - q_0q_1) & (q_0^2 - q_1^2 - q_2^2 + q_3^2) \end{bmatrix} \tag{3.2}$$

Per quanto riguarda la dinamica dell'assetto di un corpo rigido si ha che:

$$\dot{q} = \frac{1}{2}\Omega(\omega)q \tag{3.3}$$

dove:

$$q = \begin{bmatrix} q_0 & q_1 & q_2 & q_3 \end{bmatrix}^T \tag{3.4}$$

$$\Omega(\omega) = \begin{bmatrix} 0 & -\bar{P} & -\bar{Q} & -\bar{R} \\ \bar{P} & 0 & \bar{R} & -\bar{Q} \\ \bar{Q} & -\bar{R} & 0 & \bar{P} \\ \bar{R} & \bar{Q} & -\bar{P} & 0 \end{bmatrix} \tag{3.5}$$

In (3.5), $\bar{P}$ rappresenta la reale velocità angolare di rollio, cioè la velocità misurata (P) corretta con bias e rumore Gaussiano bianco a media nulla: $\bar{P} = P - b_P - v_P$. La stessa convenzione è utilizzata per le variabili di beccheggio e

imbardata. Diversamente dal caso precedente, si tiene conto di tale definizione per le velocità angolari proprio per risolvere la problematica relativa all'ipotesi di bias costanti.

Sostituendo l'espressione di $\bar{P}$, $\bar{Q}$ e $\bar{R}$, la dinamica (3.3), per i singoli quaternioni, diventa:

$$\begin{aligned}
\dot{q}_0 &= -Pq_1 + b_P q_1 + v_P q_1 - Qq_2 + b_Q q_2 + v_Q q_2 - Rq_3 + b_R q_3 + v_R q_3 \\
\dot{q}_1 &= Pq_0 - b_P q_0 - v_P q_0 + Rq_2 - b_R q_2 - v_R q_2 - Qq_3 + b_Q q_3 + v_Q q_3 \\
\dot{q}_2 &= Qq_0 - b_Q q_0 - v_Q q_0 - Rq_1 + b_R q_1 + v_R q_1 + Pq_3 - b_P q_3 - v_P q_3 \\
\dot{q}_3 &= Rq_0 - b_R q_0 - v_R q_0 + Qq_1 - b_Q q_1 - v_Q q_1 - Pq_2 + b_P q_2 + v_P q_2
\end{aligned} \tag{3.6}$$

Riorganizzando la (3.6) in forma matriciale si ottiene:

$$\dot{q} = A_1(P, Q, R)q + A_2(q)b + V_1(q)v^q \tag{3.7}$$

dove v^q rappresenta il vettore dei rumori sui giroscopi, mentre le tre matrici hanno le seguenti espressioni:

$$A_1(P, Q, R) = \frac{1}{2}\begin{bmatrix} 0 & -P & -Q & -R \\ P & 0 & R & -Q \\ Q & -R & 0 & P \\ R & Q & -P & 0 \end{bmatrix} \tag{3.8}$$

$$A_2(q) = \frac{1}{2}\begin{bmatrix} q_1 & q_2 & q_3 \\ -q_0 & q_3 & -q_2 \\ -q_3 & -q_0 & q_1 \\ q_2 & -q_1 & -q_0 \end{bmatrix} \tag{3.9}$$

$$V_1(q) = A_2(q) \tag{3.10}$$

Dunque, le equazioni (3.1) e (3.7) descrivono la parte dinamica del modello del filtro e, come precedentemente accennato, lo stato del filtro è rappresentato dal vettore:

$$x = \begin{bmatrix} q \\ b \end{bmatrix} \tag{3.11}$$

Questo sistema di equazioni è non lineare, di conseguenza si rende necessaria la formulazione del filtro di Kalman esteso presentata al Par. 2.3.

3.4 Filtro di Kalman esteso per AHRS

La formulazione del processo utilizzato dal filtro di Kalman esteso è la seguente:

$$\begin{cases} \dot{x} = f(x, \omega) + w \\ z = h(x) + v \end{cases} \tag{3.12}$$

Nel caso in esame $x \in \mathbf{R}^{7\times1}$ è lo stato del sistema, composto da $q \in \mathbf{R}^{4\times1}$ e da $b \in \mathbf{R}^{3\times1}$, ed $\omega \in \mathbf{R}^{3\times1}$ sono le velocità angolari. w e v sono rispettivamente i rumori di processo e di misura che dovranno essere filtrati dallo stimatore.

3.4.1 Equazione dinamica

La funzione non lineare $f(x, \omega)$ è data da:

$$f(x, \omega) = \begin{bmatrix} A_1(\omega)q + A_2(q)b \\ \mathbf{0}^{3\times1} \end{bmatrix} \tag{3.13}$$

dove la prima riga è relativa alla dinamica dei quaternioni, la seconda alla dinamica dei bias.

Dalla formulazione del modello effettuata nel Par. 3.3 si evidenzia che il rumore di misura sui giroscopi v^q diventa rumore di processo sui quaternioni attraverso l'espressione $V_1(q)v^q$, mentre quello sui bias è stato indicato con il termine v^b. Quindi il rumore w può essere scritto come:

$$w = \begin{bmatrix} V_1(q)v^q \\ v^b \end{bmatrix} \tag{3.14}$$

3.4.2 Equazioni di uscita

L'equazione d'uscita $h(x)$ può essere ricavata facendo riferimento alle diverse modalità di funzionamento in cui si può trovare lo stimatore, secondo quanto descritto al Par. 3.2.

Nel **Modo 1** la stima è corretta utilizzando magnetometri e GPS. I magnetometri, contrariamente al caso precedente dove venivano utilizzati per individuare la prua magnetica del velivolo, sono utilizzati per descrivere esclusivamente l'andamento del campo magnetico terrestre mentre il GPS fornisce una misura dell'angolo di imbardata ψ.

Nel **Modo 2** viene utilizzata la misura del campo magnetico come nel modo 1 e le misure del GPS vengono sostituite con quelle degli accelerometri per descrivere l'andamento del vettore accelerazione.

Per questi tre vettori di misura occorre sviluppare delle equazioni attraverso le quali modellare l'andamento di tali grandezze (campo magnetico, heading geografico e vettore accelerazione) in funzione delle variabili di stato del filtro.

Vettore campo magnetico

Il vettore campo magnetico V^B può essere descritto dalla seguente relazione, nel riferimento assi-corpo, e rappresenta la prima equazione d'uscita h_1:

$$V^B = C_{b/n}(q)V^E = h_1(q, V^E) \tag{3.15}$$

dove V^E (vettore magnetico terrestre nel sistema inerziale) è noto dall'inizializzazione e può essere considerato costante durante tutta la durata del volo, mentre $C_{b/n}$ denota la matrice di rotazione dei quaternioni dal sistema di riferimento al sistema assi-corpo.

Vettore accelerazione

La seconda equazione d'uscita h_2 è relativa al vettore accelerazione:

$$a^B = C_{b/n}(q) \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} = h_2(q) \tag{3.16}$$

Così come nel caso precedente, il vettore accelerazione viene modellato semplicemente come il vettore di gravità terrestre ruotato nel sistema assi-corpo. Da qui nascono tutte le problematiche relative all'utilizzo degli accelerometri per cui, nel presente stimatore, tale formulazione sarà utilizzata solo se strettamente necessaria.

Angolo di imbardata

L'ultima variabile del vettore di uscita del filtro è l'angolo ψ. Questo può essere calcolato sia dalle misure di posizione che di velocità fornite dal GPS. La misura della velocità è generalmente più accurata e l'angolo può essere calcolato a partire dal suo valore istantaneo. Quindi la misura è ricavata dalle velocità est, $Evel$, e nord, $Nvel$, fornite dal GPS:

$$\psi = atan2\left(\frac{Evel}{Nvel}\right) \tag{3.17}$$

Per esplicitare tale equazione in funzione delle variabili del filtro si può far riferimento alla matrice di rotazione nei quaternioni (3.2), confrontandola con la stessa matrice descritta, però, in termini di angoli di Eulero:

$$C_{b/n} = \begin{bmatrix} c_\theta c_\psi & c_\theta s_\psi & -s_\theta \\ (-c_\phi s_\psi + s_\phi s_\theta c_\psi) & (c_\phi c_\psi + s_\phi s_\theta s_\psi) & s_\phi c_\theta \\ (s_\phi s_\psi + c_\phi s_\theta c_\psi) & (-s_\phi c_\psi + c_\phi s_\theta s_\psi) & c_\phi c_\theta \end{bmatrix} \tag{3.18}$$

Si può notare, infatti, che facendo il rapporto tra il secondo e primo elemento della prima riga di tale matrice si ottiene la tangente dell'angolo di imbardata ψ. Dunque l'equazione d'uscita può essere scritta come:

$$\psi = atan2\left(\frac{C_{b/n}(q)(1,2)}{C_{b/n}(q)(1,1)}\right) = h_3(q) \tag{3.19}$$

Formulazione finale

Riassumendo, per quanto riguarda l'uscita z, si ha che:

- Modo 1:

$$z = z_1 = \begin{bmatrix} V^B_{mag} \\ \psi_{gps} \end{bmatrix} \tag{3.20}$$

- Modo 2:

$$z = z_2 = \begin{bmatrix} V^B_{mag} \\ a^B_{acc} \end{bmatrix} \tag{3.21}$$

dove $V^B_{mag} \in \mathbf{R}^{3\times 1}$ sono le misure prodotte dai magnetometri, $a^B_{acc} \in \mathbf{R}^{3\times 1}$ quelle degli accelerometri e ψ_{gps} è l'orientamento del velivolo ricavata dalle velocità GPS.

Mentre per le funzioni $h(x)$ si ha che:

- Modo 1:

$$h(x, V^E) = \begin{bmatrix} h_1 \\ h_3 \end{bmatrix} = \begin{bmatrix} C_{b/n}(q)V^E \\ \tan^{-1}\left(\dfrac{2(q_1q_2 + q_0q_3)}{{q_0}^2 + {q_1}^2 - {q_2}^2 - {q_3}^2}\right) \end{bmatrix} \tag{3.22}$$

- Modo 2:

$$h(x, V^E) = \begin{bmatrix} h_1 \\ h_2 \end{bmatrix} = \begin{bmatrix} C_{b/n}(q)V^E \\ \begin{bmatrix} 2g(q_1q_3 - q_0q_2) \\ 2g(q_2q_3 + q_0q_1) \\ g({q_0}^2 - {q_1}^2 - {q_2}^2 + {q_3}^2) \end{bmatrix} \end{bmatrix} \tag{3.23}$$

3.4.3 Formulazione del modello

Anche in questo caso il filtro di Kalman viene realizzato in forma estesa e discreta: estesa a causa della natura non lineare delle equazioni utilizzate al suo interno; discreta per rendere lo stimatore direttamente realizzabile a livello pratico.

Così come nel caso precedente è necessario definire lo Jacobiano della funzione dinamica e delle misure. La linearizzazione è effettuata attorno al valore corrente di stima e misura:

$$F_k = \left.\frac{\partial f}{\partial x}\right|_{x=\hat{x}_k, \omega=\omega_k} \qquad H_k = \left.\frac{\partial h}{\partial x}\right|_{x=\hat{x}_k} \tag{3.24}$$

dove $\hat{x}_k$ è la stima dello stato e ω_k è il vettore delle velocità angolari misurate, entrambe valutate al tempo k.

Lo Jacobiano F_k è dato da:

$$F_k = \frac{\partial f}{\partial x} = \begin{bmatrix} F_{11} & F_{12} \\ \mathbf{0}^{3\times4} & \mathbf{0}^{3\times3} \end{bmatrix} \tag{3.25}$$

dove

$$F_{11} = \frac{\partial}{\partial q}\left\{\Omega(\omega - b)q\right\} = \Omega(\omega - b)$$

e

$$F_{12} = \frac{\partial}{\partial b}\left\{\Omega(\omega - b)q\right\} = \frac{1}{2}\begin{bmatrix} q_1 & q_2 & q_3 \\ -q_0 & q_3 & -q_2 \\ -q_3 & -q_0 & q_1 \\ q_2 & -q_1 & -q0 \end{bmatrix}.$$

La matrice H_k, invece, dipende dal modo di funzionamento dello stimatore ed è data da:

- MODO 1: $H_k = H_{k1} = \begin{bmatrix} C_1 & \mathbf{0}^{3\times3} \\ C_3 & \mathbf{0}^{1\times3} \end{bmatrix}$

- MODO 2: $H_k = H_{k2} = \begin{bmatrix} C_1 & \mathbf{0}^{3\times3} \\ C_2 & \mathbf{0}^{3\times3} \end{bmatrix}$

dove:

$$C_1 = \frac{\partial h_1(q, V^E)}{\partial q} \tag{3.26}$$

$$C_2 = \frac{\partial h_2(q)}{\partial q} \tag{3.27}$$

$$C_3 = \frac{\partial h_3(q)}{\partial q} \tag{3.28}$$

La matrice H_k nel primo caso ha dimensione 4x7, nel secondo caso 6x7. L'aggiunta degli zeri è dovuta al fatto che le equazioni d'uscita non dipendono dalle ultime tre variabili di stato b_P, b_Q, b_R.

3.4.4 Discretizzazione

A questo punto è stata effettuata la discretizzazione del sistema complessivo (Eq. (3.7) e (3.1)) utilizzando il metodo di Eulero in avanti:

$$\begin{gathered} q_{k+1} = q_k + dt[A_1(P,Q,R)q_k + A_2(q_k)b_k + V_1(q_k)v_k^q] \\ b_{k+1} = b_k + dtv_k^b \end{gathered} \tag{3.29}$$

dove si è indicato con dt il passo di integrazione. In forma matriciale, tale espressione diventa:

$$\begin{bmatrix} q_{k+1} \\ b_{k+1} \end{bmatrix} = \begin{bmatrix} I_4 + dtA_1(P,Q,R) & dtA_2(q_k) \\ 0 & I_3 \end{bmatrix} \begin{bmatrix} q_k \\ b_k \end{bmatrix} + \begin{bmatrix} dtV_1(q_k) & 0 \\ 0 & I_3 dt \end{bmatrix} \begin{bmatrix} v_k^q \\ v_k^b \end{bmatrix} \tag{3.30}$$

La matrice che moltiplica il vettore di stato sarà chiamata A_k, mentre quella che moltiplica i rumori V_k. In questo modo la (3.30) può essere riscritta come:

$$\begin{bmatrix} q_{k+1} \\ b_{k+1} \end{bmatrix} = A_k \begin{bmatrix} q_k \\ b_k \end{bmatrix} + V_k \begin{bmatrix} v_k^q \\ v_k^b \end{bmatrix} \tag{3.31}$$

Si noti che il sistema, anche in questa formulazione, mantiene la sua non linearità. L'eq. (3.31) verrà utilizzata nel passo di predizione del filtro, mentre lo Jacobiano F_k verrà utilizzato nel calcolo della matrice P_k.

3.4.5 Inizializzazione

L'inizializzazione dell'algoritmo dell'EKF coinvolge la produzione della stima dello stato $\hat{x}_0$ e della matrice di covarianza dell'errore associata a tale stima P_0, che per semplicità si assume essere diagonale. Si considera, inoltre, che per ottenere una buona stima del bias iniziale $\hat{b}_0$ potrebbe essere fatta una semplice stima sui giroscopi mentre il velivolo è in volo, mentre per inizializzare $\hat{q}_0$ potrebbe essere fatta una stima del'assetto a partire dalle condizioni iniziali del velivolo, ricordando che la scelta dei valori iniziali dei quaternioni deve essere fatta nel rispetto della proprietà di norma unitaria.

3.5 Implementazione dell'algoritmo

L'ultimo passo dell'implementazione dell'AHRS consiste nella sintesi dell'algoritmo di stima. Nel caso in esame, l'algoritmo presenta le seguenti caratteristiche che lo differenziano dalla formulazione precedente:

- la correzione della stima non viene effettuata ad ogni passo. La predizione, infatti, avviene con una frequenza di 50 Hz, mentre la correzione avviene a 25 Hz, in modo da rispettare anche i tempi reali di lavoro dei sensori;
- la correzione dipende dal modo di funzionamento in cui si trova il filtro. Tale stato è determinato dal segnale GPS: se il segnale è valido allora si utilizzerà un set di misure (MODO 1), altrimenti il set di misure sarà diverso (MODO 2) poiché, per qualche motivo (perdita di segnale, errori), il GPS non è disponibile;
- la matrice che rappresenta la varianza dei disturbi sullo stato (Q_k) non è costante ma dipende dai rumori sulle misure (giroscopi e bias dei giroscopi) e dai quaternioni attraverso la matrice V_k. Tale formulazione permette un adattamento continuo di tale matrice ai valori correnti delle variabili d'assetto e incrementa le prestazioni del filtro, permettendo di stimare anche valori non costanti dei bias sui giroscopi.

I passi implementati del filtro sono, dunque, i seguenti [35]:

1. PASSO k: solo predizione
   - $\bar{x}_k = A_{k-1}\hat{x}_{k-1} \rightarrow V_1(\bar{q}_k) \Rightarrow V_{k-1}$

- $\bar{P}_k = F_{k-1} P_{k-1} F_{k-1}^T + V_{k-1} V V_{k-1}^T$
- $\hat{x}_k = \bar{x}_k, P_k = \bar{P}_k$

2. PASSO $k+1$: predizione e correzione

   - $\bar{x}_{k+1} = A_k \hat{x}_k \rightarrow V_1(\bar{q}_{k+1}) \Rightarrow V_k$
   - $\bar{P}_{k+1} = F_k P_k F_k^T + V_k V V_k^T$
   - **Se GPS è valido**, allora $H_{k+1}(\bar{q}_{k+1}) = H_{1_{k+1}}$
     altrimenti $H_{k+1}(\bar{q}_{k+1}) = H_{2_{k+1}}$
   - $K_{k+1} = \bar{P}_{k+1} H_{k+1}^T [H_{k+1} \bar{P}_{k+1} H_{k+1}^T + W]^{-1}$
   - $\hat{x}_{k+1} = \bar{x}_{k+1} + K_{k+1}[z_{k+1} - h(\bar{x}_{k+1})]$
   - $P_{k+1} = [I - K_{k+1} H_{k+1}] \bar{P}_{k+1}$

dove x_k è il vettore di stato $\begin{bmatrix} q & b \end{bmatrix}^T$, V è la matrice di covarianza del rumore sulle velocità angolari e sui bias dei giroscopi, W è la matrice di covarianza dell'errore sulle misure, z è il vettore delle misure. Si noti che le dimensioni di W, H e z sono diverse nei diversi modi di funzionamento. P è la matrice di covarianza dell'errore sullo stato stimato, K è la matrice del guadagno di Kalman, per la correzione della stima. La notazione $\bar{(.)}$ indica una grandezza predetta, mentre $\hat{(.)}$ indica una grandezza corretta.

L'implementazione in linguaggio C dell'algoritmo appena presentato può essere trovata nell'Appendice A.

Capitolo 4

Risultati

Il seguente capitolo è dedicato alla validazione dell'Attitude and Heading Reference System multi-mode implementato nel Capitolo 3. Questo è stato confrontato con lo stimatore standard, presentato al Capitolo 2, il cui uso è ormai consolidato in letteratura, al fine di evidenziare le differenze e le eventuali migliorie apportate.

4.1 Contesto di riferimento

L'algoritmo AHRS per la derivazione dell'assetto è stato sviluppato e studiato in ambiente Matlab/Simulink, per poterne testare la validità e il corretto funzionamento.

La Fig. 4.1 mostra un diagramma a blocchi del modello complessivo utilizzato per le simulazioni. I sensori IMU disponibili misurano le relative grandezze di stato del velivolo, demandando all'algoritmo AHRS la stima dell'assetto. Tale stima, assieme alle misure rilevate da accelerometri e GPS, viene utilizzata da un algoritmo di navigazione inerziale [4] per il calcolo di posizione e velocità. Nota la posizione del velivolo e quella del prossimo waypoint da raggiungere, viene pianificata la direzione di riferimento a quell'istante di tempo e gli autopiloti lavorano per il controllo dell'assetto e della direzione di volo del velivolo, garantendo inoltre la stabilizzazione della quota e della velocità di volo.

Lo scenario di riferimento è quello in cui un pilota porta il velivolo ad una altitudine di sicurezza e raggiunge la condizione di volo di crociera con l'ausilio del radiocomando, per poi commutare, attraverso un designato canale, dalla configurazione manuale a quella automatica.

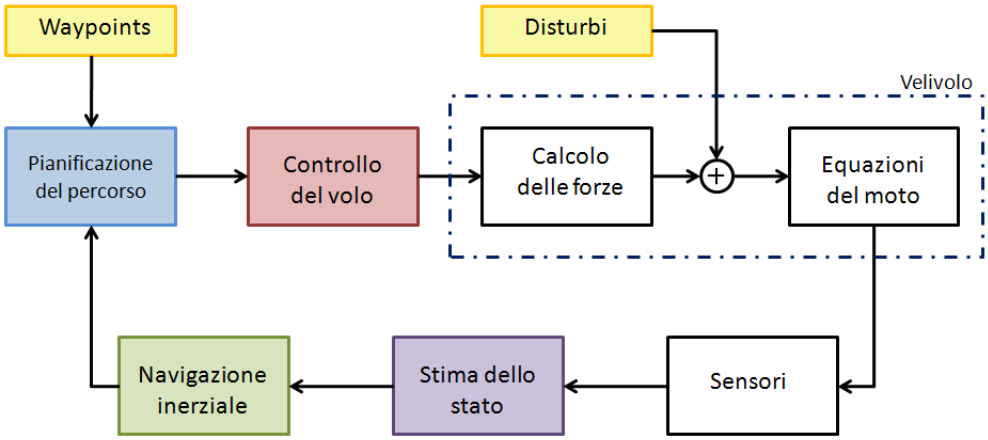


Figura 4.1: Diagramma a blocchi per le simulazioni.

Nei test di volo simulati si è supposto che il velivolo si trovi ad un'altezza nominale di 50 m con una velocità nominale di 9 m/s rispetto al suolo. La missione che deve essere in grado di completare autonomamente è quella di sorvolare una limitata zona, segnata da waypoints, e localizzata sulla città di Benevento alle coordinate $(41.1326N, 14.7823E)$.

Nelle Figs. 4.2 e 4.3 è mostrato il percorso di volo del velivolo segnato da waypoints, sia nel piano bidimensionale che nello spazio libero. Tali simulazioni sono state ottenute considerando lo schema di controllo implementato in [4] e chiuso in retroazione, dove, però, gli angoli di Eulero necessari per il funzionamento degli autopiloti sono quelli ottenuti dal modello di simulazione del velivolo.

Partendo da questo contesto, occorre inserire nel loop il nuovo AHRS, verificando che la stima prodotta sia quanto più prossima al valore reale delle grandezze stimate.

4.1.1 Approccio adottato

Il nuovo AHRS si propone di stimare gli angoli di assetto del velivolo e le polarizzazioni dei giroscopi nell'ipotesi che questi siano lentamente variabili, utilizzando un filtro multi-mode che adopera principalmente GPS e magnetometri, demandando l'uso degli accelerometri solo nei casi in cui c'è assenza di un segnale GPS valido.

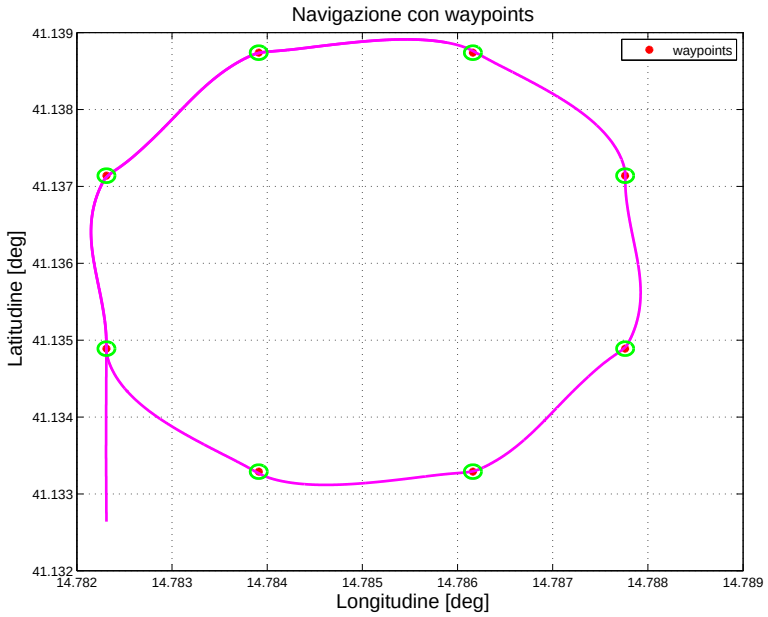


Figura 4.2: Waypoints e traiettoria 2D.

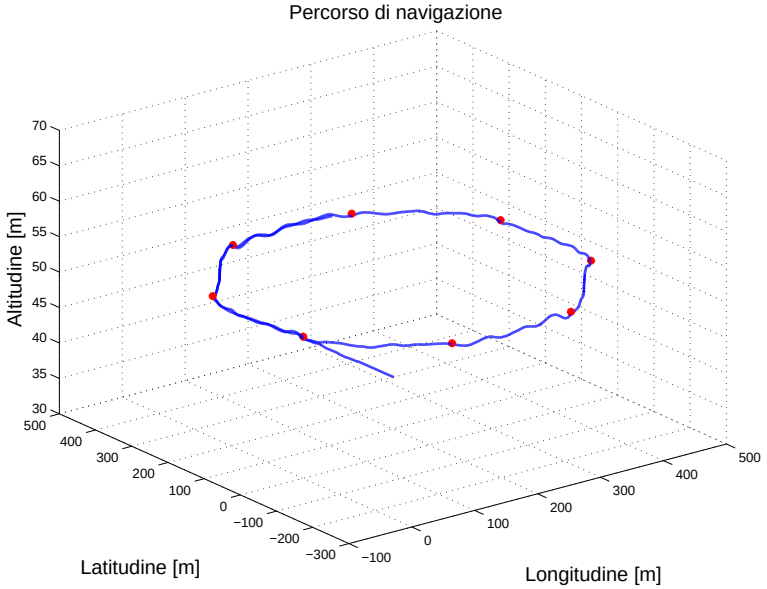


Figura 4.3: Waypoints e traiettoria 3D.

L'approccio pensato ed utilizzato per la validazione dell'algoritmo di stima è il seguente:

- una prima versione del nuovo AHRS è stata implementata in codice Matlab, al fine di una validazione *Model-In-The-Loop*. In questa fase è stato effettuato il tuning dei parametri del simulatore;
- il codice Matlab è stato ottimizzato considerando la struttura sparsa della maggior parte delle matrici del sistema;
- l'algoritmo AHRS è stato inglobato all'interno di una S-function. La S-function è la descrizione in un linguaggio di programmazione di un blocco Simulink e si compone di una particolare sintassi che la abilita ad interagire con gli "equation solvers" dell'ambiente di simulazione. Il linguaggio scelto per la descrizione del modello è il C per ovvi motivi: una volta validato il codice in fase di simulazione, infatti, esso sarà depurato di eventuali macro legate all'ambiente Matlab/Simulink per essere, poi, usato come software di navigazione in esecuzione sul micro processore reale. Questo tipo di validazione è chiamata *Software in the Loop* (SIL): essa permette di verificare che il codice sintetizzato nell'ambiente di simulazione continui ad essere valido anche convertito nel linguaggio adatto al computer di bordo del velivolo. Nell'appendice A sono riportati i codici delle S-function implementate;
- è stato effettuato un confronto tra i risultati del codice Matlab e del codice C, onde verificare che i due algoritmi fornissero gli stessi risultati.

Una volta superate con successo queste fasi di testing, il passo successivo sarebbe quello di scaricare il codice sul target embedded ed eseguirlo in real-time chiudendo il ciclo con il modello di simulazione. Tale fase prende il nome di *Processor in the loop* (PIL) e costituisce un passo intermedio per validare il sistema complessivo, eseguendolo in real time sul micro processore prima ancora che sul velivolo reale. Gradualmente l'hardware presente nel ciclo potrebbe essere aumentato includendo anche i sensori e il software di interfacciamento I/O tra i vari componenti. La fase finale concerne il test reale di volo. Quest'ultime trattazioni esulano, però, dall'obiettivo principale di questo lavoro di tesi.

4.2 Risultati per l'AHRS

In questa sezione sono presentati i risultati ottenuti seguendo l'approccio discusso in precedenza. La validazione dell'algoritmo di stima è stata eseguita in maniera graduale in modo da aggiungere complessità ad ogni step, fino ad ottenere e testare il filtro nel caso più generale possibile.

4.2.1 Stimatore semplificato

Come primo passo, lo stimatore è stato simulato considerando la seguente ipotesi semplificativa: si ha sempre a disposizione un segnale GPS valido, in questo modo il filtro si trova sempre ad operare nel *Modo 1* di funzionamento, dove utilizza esclusivamente le misure provenienti da magnetometri e GPS. I bias sui giroscopi, inoltre, sono supposti costanti e pari ai valori: $(b_P, b_Q, b_R) = (0.1, 0.25, 0.5)$. Infine tutti i sensori sono affetti da rumore.

Simulazioni Model-In-The-Loop

Seguendo l'approccio indicato precedentemente, l'algoritmo di stima è stato dapprima implementato in ambiente Matlab/Simulink. In tale circostanza, l'algoritmo non è stato inserito nel loop di controllo e le grandezze stimate sono state confrontate con le rispettive grandezze fornite dal modello di simulazione del velivolo.

Le Figs. 4.4 e 4.5 mostrano le stime degli angoli di Eulero e dei bias sui giroscopi. Nei grafici è mostrata la comparazione delle grandezze stimate (in verde) con le grandezze calcolate dal modello di simulazione numerico del velivolo (in blu). Dalle figure si evince che la stima è effettuata correttamente.

Simulazioni Software-In-The-Loop

Una volta verificato che il comportamento in ambiente Matlab/Simulink corrisponde a quello desiderato, la fase di validazione successiva considerata è quella *Software-In-The-Loop*, in cui lo stimatore, scritto in codice C, è stato inglobato in una *S-function*.

La Fig. 4.6 mostra la stima degli angoli di Eulero ottenuti in tale fase. In particolare, si sono voluti confrontare i risultati della simulazione *Model-In-The-Loop*, eseguita con lo stimatore implementato in Matlab (curva in blu) con i risultati della simulazione *Software-In-The-Loop* (curva in verde), eseguita con

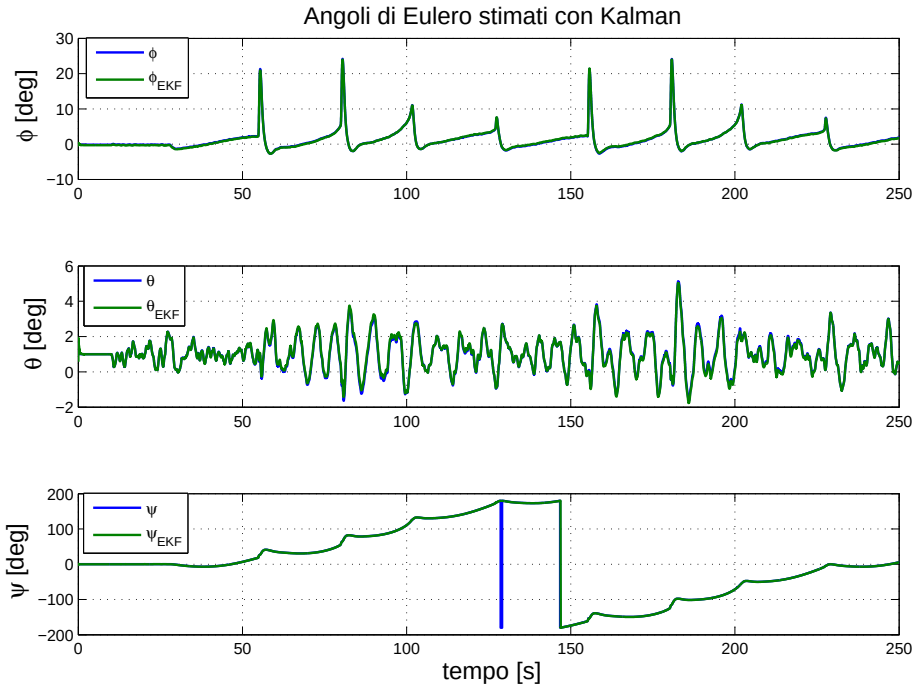


Figura 4.4: Angoli di assetto del velivolo stimati con l'algoritmo AHRS. L'uscita del filtro (in verde) insegue correttamente il valore di riferimento fornito dal modello di simulazione (in blu).

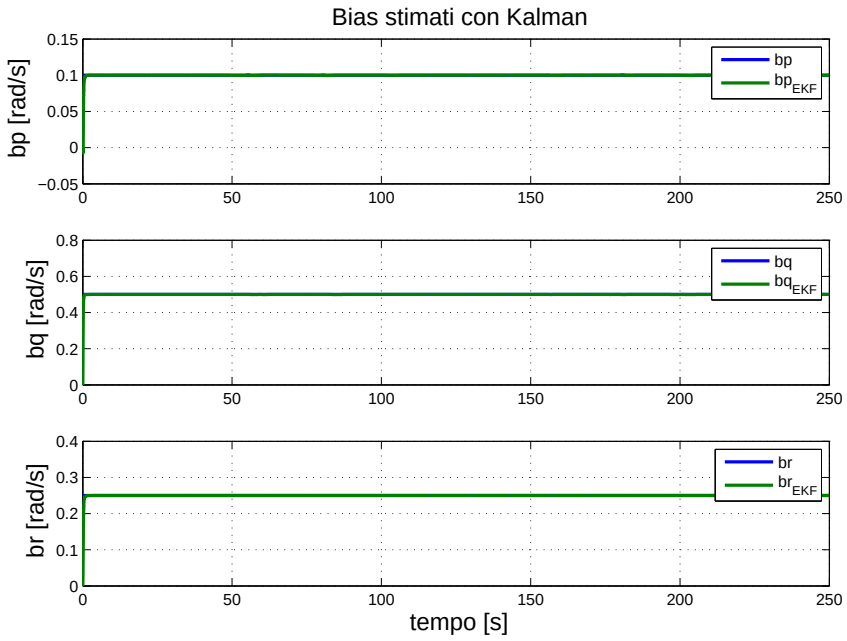


Figura 4.5: Polarizzazione dei giroscopi stimati con l'algoritmo AHRS (in verde) confrontati con quelle fornite dal modello di simulazione (in blu).

lo stimatore tradotto in linguaggio C. E' possibile notare che i risultati, nei due casi, sono identici. Per tale motivo il filtro implementato in C è stato utilizzato per le prove successive.

Confronto con l'AHRS standard

Fino a questo momento è stato possibile validare il funzionamento di un filtro di Kalman per la stima d'assetto del velivolo, scritto in C, che utilizzi esclusivamente magnetometri e GPS nella fase di correzione. A questo punto, il passo successivo è stato quello di mettere a confronto questo nuovo stimatore con lo stimatore standard, anch'esso implementato in codice C. Per maggiore chiarezza i grafici dei tre angoli di Eulero sono stati riportati separatamente nelle Figs. 4.7, 4.8, 4.9. In Fig. 4.10 è rappresentato l'andamento dell'errore di stima dei due filtri. In particolare si nota come la stima dell'angolo di rollio sia migliorata significativamente, in quanto tale angolo è quello che varia maggiormente in presenza di virate e che, quindi, subisce l'effetto degli errori di stima derivanti dall'utilizzo degli accelerometri. In media, su tutti e tre gli angoli, si è constatato un miglioramento di circa il 50% in termini di stima.

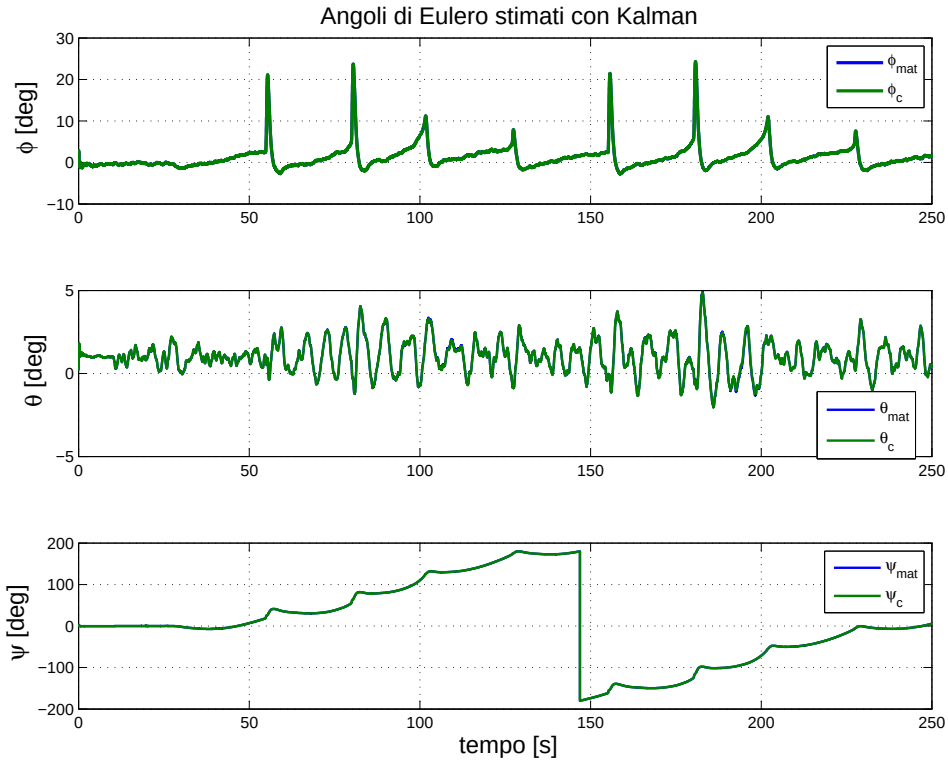


Figura 4.6: Confronto della stima degli angoli di Eulero. Lo stesso filtro è implementato in codice Matlab (in blu) e in codice C (in verde). Le due simulazioni forniscono gli stessi risultati.

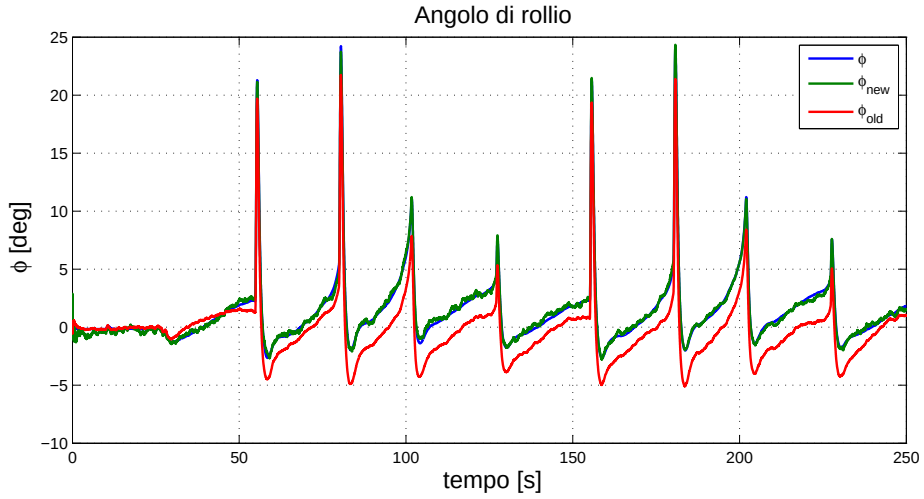


Figura 4.7: Stimatori a confronto: angolo di rollio. Lo stimatore multi-mode (in verde) stima correttamente l'angolo di rollio di riferimento (in blu). Al contrario il filtro standard (in rosso) presenta un offset soprattutto in corrispondenza delle virate.

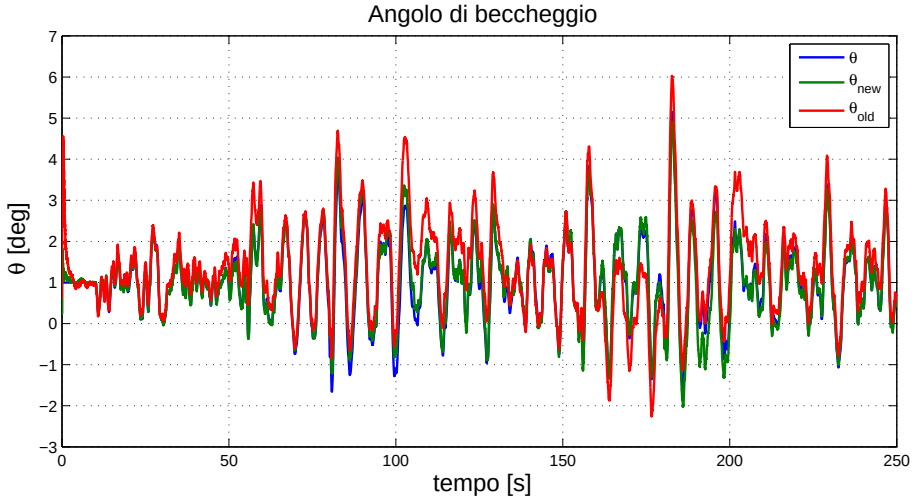


Figura 4.8: Stimatori a confronto: angolo di beccheggio. Lo stimatore multi-mode (in verde) stima correttamente l'angolo di beccheggio di riferimento (in blu). La curva rossa indica la stima fornita dal filtro standard che si discosta leggermente dalle altre due.

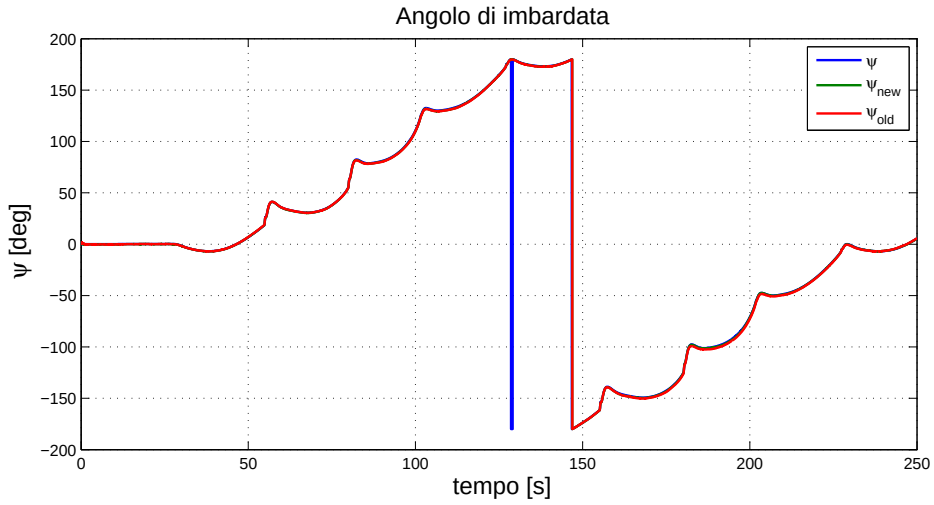


Figura 4.9: Stimatori a confronto: angolo di imbardata. I risultati dei due stimatori (in rosso lo standard, in verde il multi-mode) appaiono sovrapposti. Un'analisi dell'errore evidenzia un leggero scostamento tra le due curve.

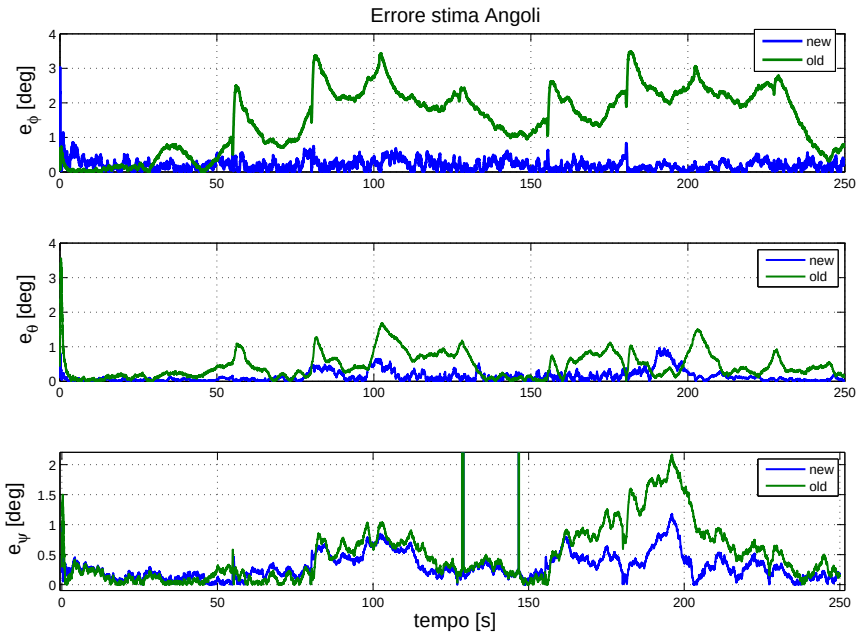


Figura 4.10: Stimatori a confronto: errori di stima. In blu è rappresentato l'errore dei stima dell'AHRS multi-mode, mentre in verde quello dello stimatore standard. In media, su tutti e tre gli angoli, si è constatato un miglioramento di circa il 50% in termini di stima.

Simulazioni a ciclo chiuso

A questo punto lo stimatore è stato inserito nel loop di controllo, in modo tale che gli autopiloti possano utilizzare, per il loro funzionamento, non le grandezze fornite dal modello di simulazione bensì quelle stimate dal filtro.

In tale condizione, la stima degli angoli di Eulero è riportata in Fig. 4.11 dove è stato messo a confronto l'uscita del filtro (linea in verde) con il riferimento fornito dal modello (linea in blu). In Fig. 4.12 è invece riportata la stima dei bias sui giroscopi (linea in verde) che, come si può notare, sono correttamente stimati ai valori costanti di riferimento (linea in blu). Infine, in Fig. 4.13 è rappresentata la traiettoria effettuata dal velivolo nel piano bidimensionale. Tale traiettoria (linea in viola) è stata confrontata con la traiettoria ottenuta quando gli angoli di assetto sono quelli provenienti dal modello di simulazione del velivolo.

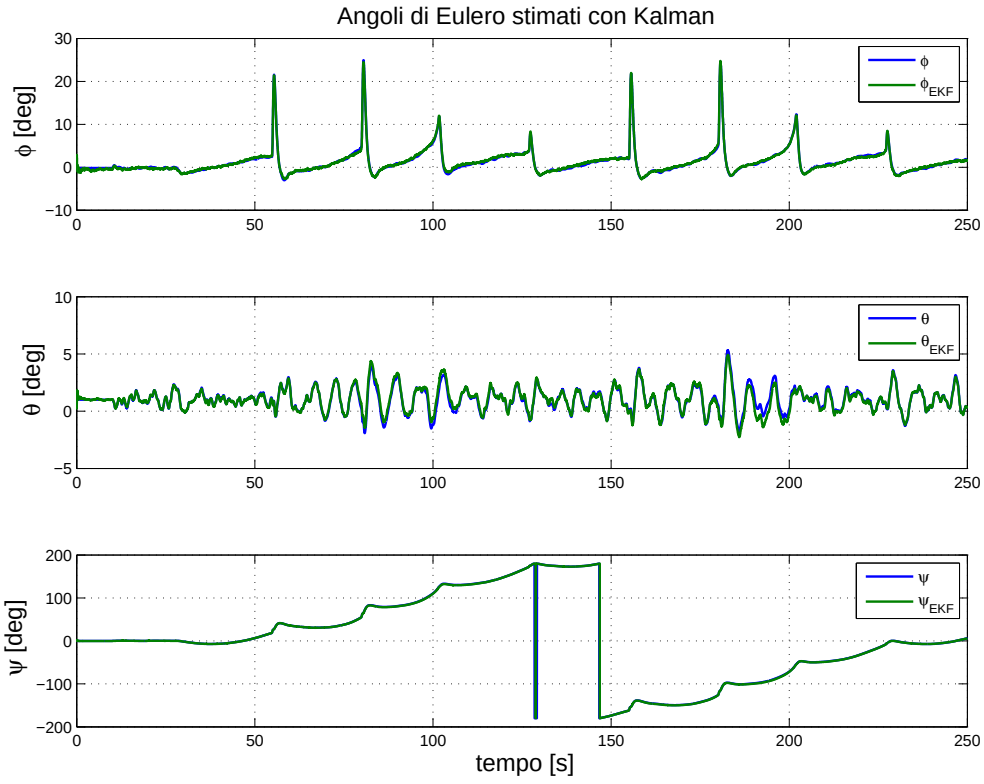


Figura 4.11: Stima degli angoli di Eulero a ciclo chiuso. In verde è rappresentata la stima degli angoli di Eulero quando il loop di controllo è chiuso sulle grandezze stimate; in blu sono rappresentati gli angoli forniti dal modello di simulazione.

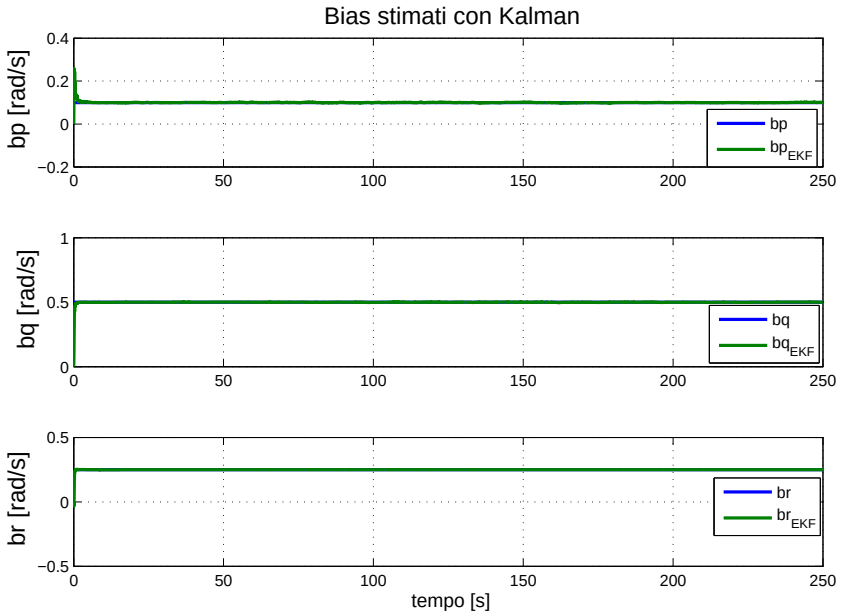


Figura 4.12: Stima dei bias sui giroscopi a ciclo chiuso. In verde è rappresentata la stima dei bias quando il loop di controllo è chiuso sulle grandezze stimate; in blu sono rappresentati i bias forniti dal modello di simulazione.

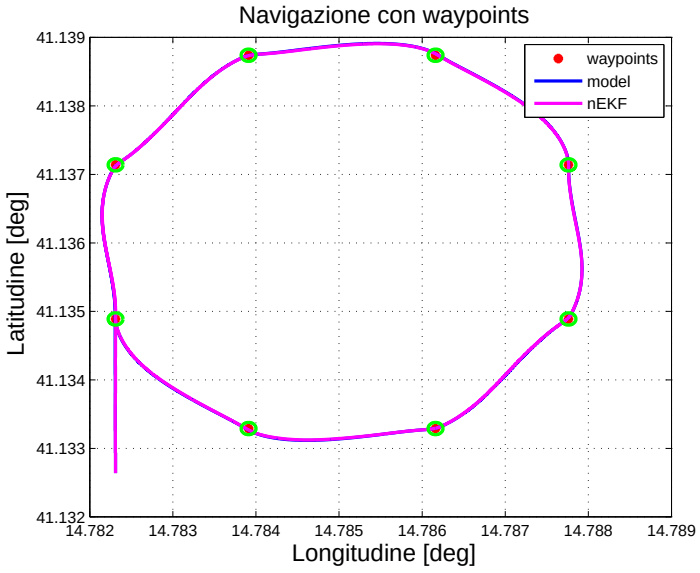


Figura 4.13: Traiettoria a ciclo chiuso. La traiettoria ottenuta chiudendo il loop di controllo sulle grandezze stimate dal filtro multi-mode è confrontata con quella di riferimento ottenuta quando gli angoli di assetto sono quelli dal modello.

4.2.2 Stimatore multi-mode

Una volta validato lo stimatore nelle sue condizioni ideali, ossia quando il GPS è sempre disponibile, è possibile testarlo nel caso più realistico: il segnale GPS non è sempre disponibile e lo stimatore dovrà commutare tra i due modi di funzionamento[1]. In particolare, nelle prove effettuate, si è ipotizzato che il segnale GPS venga perso per intervalli di 25 secondi e poi riacquisito secondo l'andamento di Fig. 4.14.

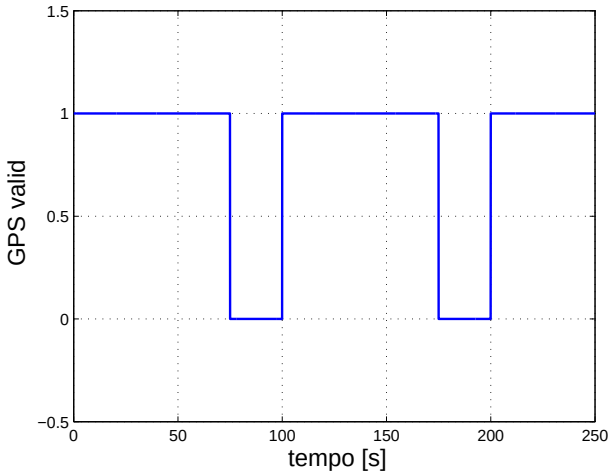


Figura 4.14: Validità del segnale GPS. Si ipotizza che il segnale GPS venga perso ogni 75 secondi per un intervallo di tempo di 25 secondi.

Si può notare che per gli intervalli di tempo in cui il segnale GPS è perso, la stima degli angoli di Eulero non presenta significativi peggioramenti rispetto al caso precedente. In questi intervalli, non essendo possibile utilizzare le misure del GPS per la correzione delle grandezze stimate dal filtro, si fa affidamento alle misure degli accelerometri unitamente a quelle dei magnetometri.

La Figs 4.15 mostra le simulazioni ottenute con il filtro multi-mode, dove le barre rosse ravvicinate delimitano l'intervallo di tempo nel quale viene perso il segnale GPS. Gli angoli di assetto stimati (curve in verde) sono stati confrontati con gli stessi angoli provenienti, però, dal modello di simulazione del velivolo.

In Fig. 4.16 è riportato, invece, l'andamento degli errori commessi dai due stimatori in questa circostanza. Anche in questo caso si nota un miglioramento

[1]Per un approfondimento sull'implementazione dell'algoritmo di stima si faccia riferimento all'Appendice A.

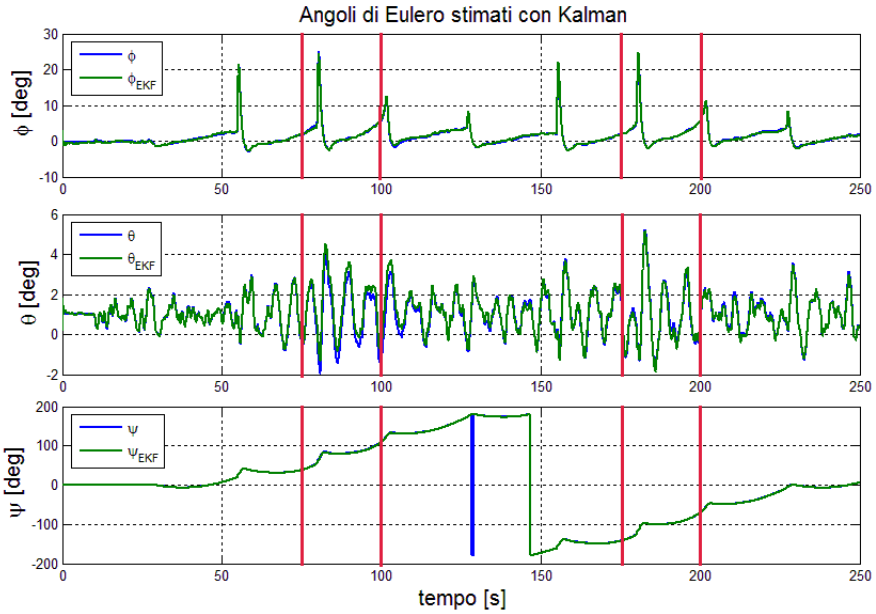


Figura 4.15: Stimatore complessivo: angoli di Eulero. Le curve in verde rappresentano la stima effettuata dallo stimatore multi-mode confrontata con il riferimento (in blu); le barre rosse ravvicinate delimitano l'intervallo di tempo nel quale viene perso il segnale GPS.

della stima: il filtro multi-mode (linee in blu) commette un errore che è sempre minore di 1°, mentre il filtro standard (linee in verde) presenta dei picchi elevati soprattutto in corrispondenza delle virate.

Stima con bias lentamente variabili

Fino ad ora si è sempre supposto che le polarizzazioni dei giroscopi fossero costanti, ipotesi che a questo punto è possibile rilassare.

Prove sperimentali dimostrano che può essere considerato costante solo il bias relativo alla velocità di imbardata, mentre per gli altri due (velocità di rollio e di beccheggio) è opportuno considerare un andamento lentamente variabile. In particolare è stato scelto un andamento sinusoidale attorno al valore costante precedentemente considerato [33], ovvero:

$$\begin{aligned} b_P &= 0.1 + 0.05 sin\left(\tfrac{2\pi}{150} t\right) rad/s \\ b_Q &= 0.25 + 0.05 sin\left(\tfrac{2\pi}{150} t\right) rad/s \\ b_R &= 0.5 rad/s \end{aligned} \tag{4.1}$$

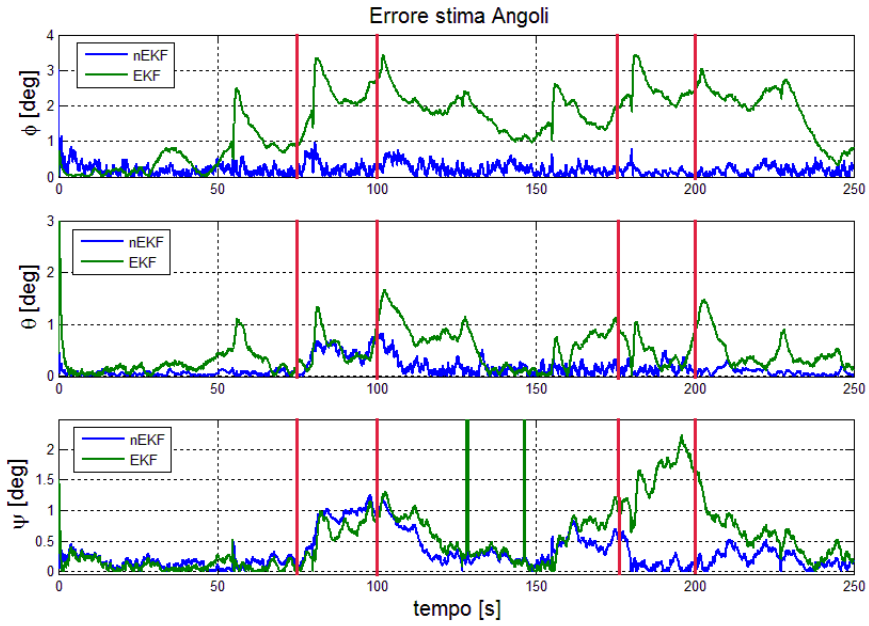


Figura 4.16: Stimatore complessivo: errori di stima. Le curve in blu rappresentano l'errore di stima del filtro multi-mode confrontato con quello dell'AHRS standard (in verde); le barre rosse ravvicinate delimitano l'intervallo di tempo nel quale viene perso il segnale GPS.

Le Figs. 4.17 e 4.18 mostrano che sia gli angoli d'assetto che le polarizzazioni dei giroscopi stimate (linee in verde) inseguono correttamente il riferimento fornito dal modello di simulazione (linee in blu). In Fig. 4.19 è riportata la traiettoria del velivolo nel piano bidimensionale, ottenuta considerando l'AHRS multi-mode all'interno del ciclo di controllo ed in presenza di bias lentamente variabili. E' possibile notare che tale traiettoria (linea in viola) si discosta di poco da quella che il velivolo effettuerebbe quando gli angoli di assetto sono quelli forniti dal modello di simulazione.

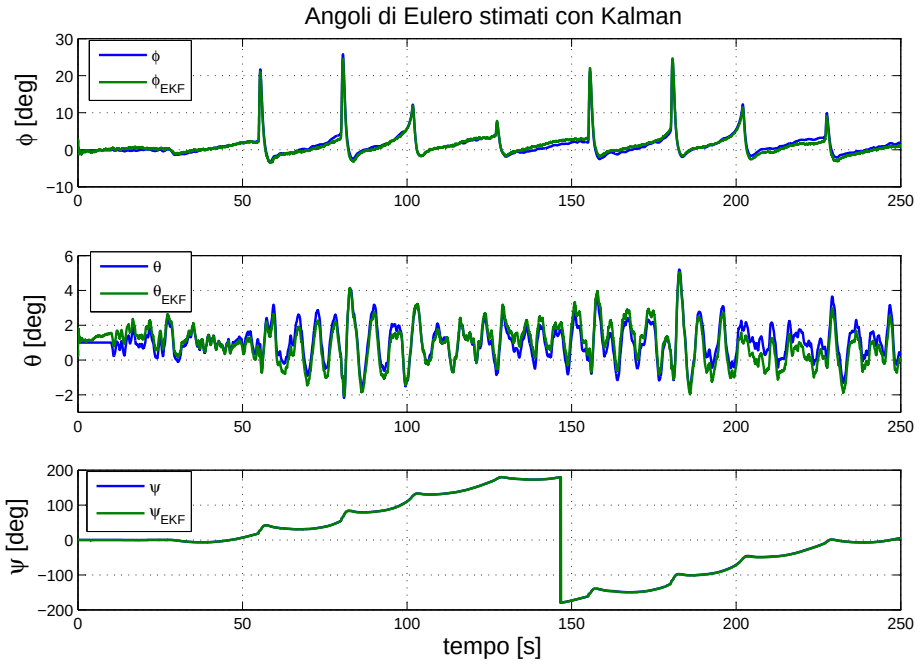


Figura 4.17: Stima angoli di Eulero con bias sui giroscopi lentamente variabili. Le curve in verde rappresentano la stima degli angoli del filtro multi-mode; in blu sono rappresentati i rispettivi valori di riferimento forniti dal modello.

In Fig. 4.20, infine, la stima ottenuta con l'AHRS multi-mode (linea verede) è messa a confronto con quella ottenuta dall'AHRS standard (linea in rosso). Quest'ultima, in presenza di polarizzazioni lentamente variabili, subisce forti degradazioni, compromettendo completamente il sistema di controllo di volo. La causa è da ricercare nella non corretta stima delle polarizzazioni, come mostra la Fig. 4.21, in quanto l'AHRS è stato implementato considerando l'ipotesi di polarizzazioni costanti.

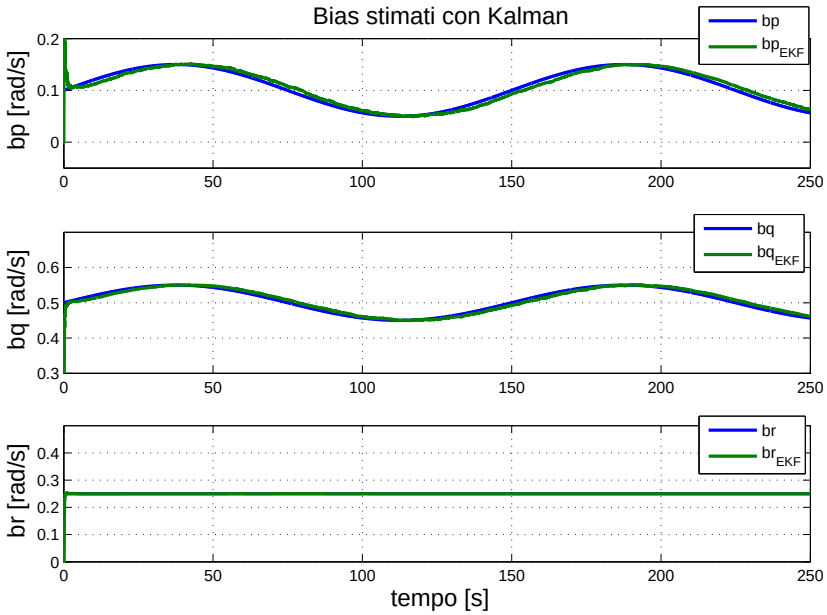


Figura 4.18: Stima bias sui giroscopi lentamente variabili. Le curve in verde rappresentano la stima dei bias del filtro multi-mode; in blu sono rappresentati i rispettivi valori di riferimento forniti dal modello.

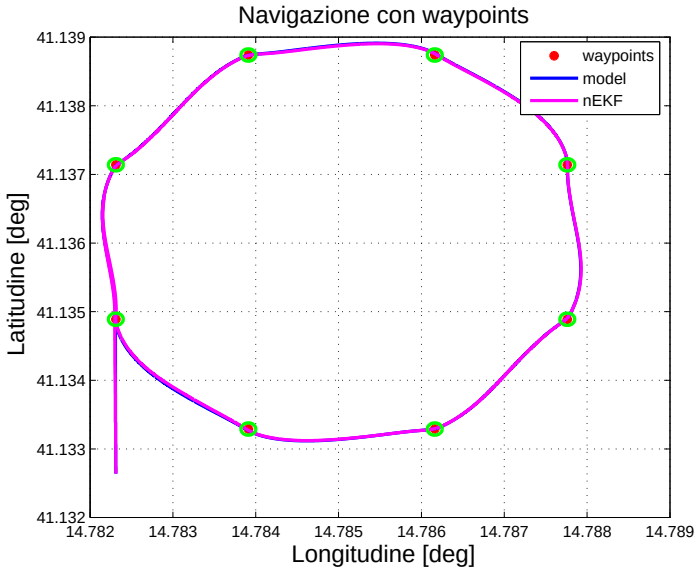


Figura 4.19: Traiettoria a ciclo chiuso con bias sui giroscopi lentamente variabili. La traiettoria ottenuta chiudendo il loop di controllo sulle grandezze stimate dal filtro multi-mode è confrontata con quella di riferimento fornita dal modello.

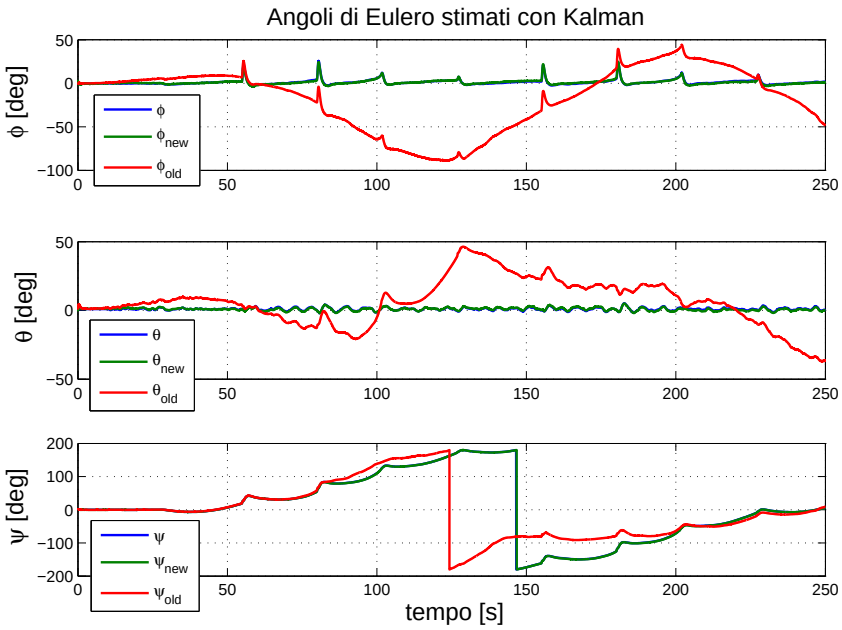


Figura 4.20: Confronto stima angoli di Eulero con bias sui giroscopi lentamente variabili. Le curve in rosso rappresentano la stima fornita dall'AHRS standard. E' evidente che esse si discostano notevolmente dalla stima dell'AHRS multi-mode (curve in verde) e dal modello (curve in blu).

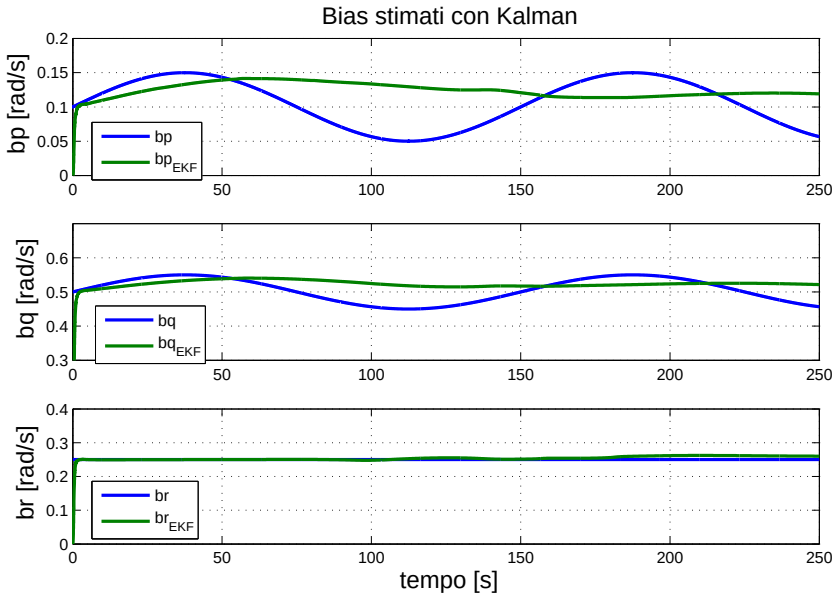


Figura 4.21: Stimatore standard: stima bias sui giroscopi lentamente variabili. L'AHRS standard (curve in verde) non riesce a stimare correttamente i bias lentamente variabili (curve in blu).

4.2.3 Altri scenari di simulazione

I risultati presentati finora hanno mostrato che il nuovo AHRS, riducendo al minimo l'utilizzo degli accelerometri, offre prestazioni migliori rispetto allo stimatore standard, soprattutto quando il velivolo compie manovre con componenti di accelerazione molto elevate. A questo punto è interessante approfondire questo aspetto e testare il comportamento del filtro quando il velivolo compie una traiettoria con virate più spinte rispetto a quelle presentate finora.

La traiettoria di riferimento utilizzata per queste prove è riportata in Fig. 4.22 ed è stata ottenuta avvicinando tra loro alcuni waypoints. Naturalmente l'algoritmo di navigazione con linea di vista (cfr. Par. 1.6.2) fa in modo che il velivolo si prepari al waypoint $n+1$, modificando il suo riferimento, prima di raggiungere il waypoint n, evitando di compiere virate troppo brusche. Ciò nonostante la nuova traiettoria presenta andamenti più dinamici della precedente.

Per le simulazioni di seguito presentate è stato utilizzato il filtro multi-mode implementato in codice C, chiuso in retroazione con lo schema di controllo. Inoltre, è presente rumore di misura su tutti i sensori interessati.

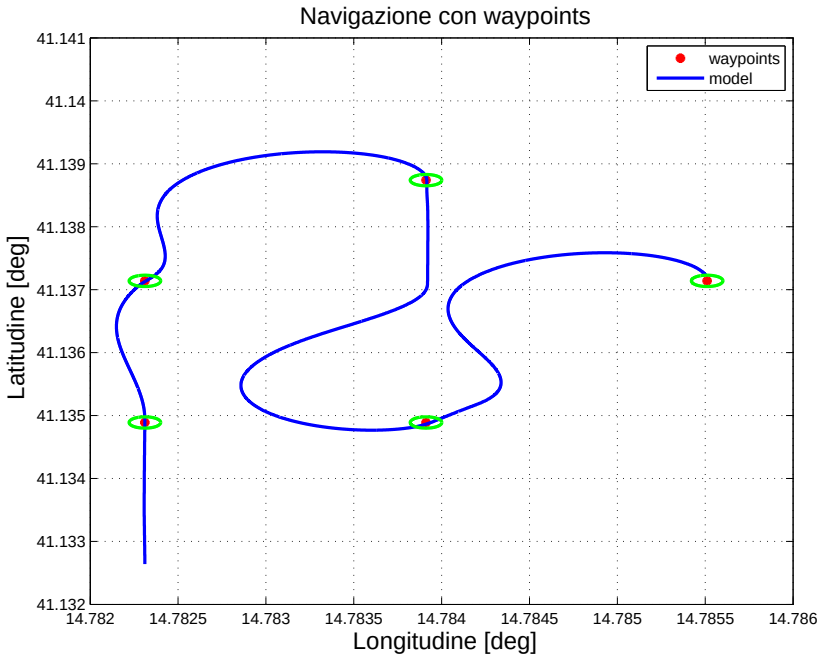


Figura 4.22: Waypoints con nuova traiettoria da inseguire.

Stima con bias costanti

Le prime prove sono state effettuate imponendo valori costanti per le polarizzazioni sui giroscopi al fine di confrontare i risultati con quelli dello stimatore standard. Nelle Figs. 4.23, 4.24 e 4.25 si evince che lo stimatore standard (curve in rosso) presenta dei peggioramenti in corrispondenza, soprattutto, degli angoli di rollio e di beccheggio, mentre l'AHRS multi-mode insegue correttamente l'andamento di tali grandezze (curva in verde). Lo stesso risultato si riscontra nell'analisi degli errori di Fig. 4.26. Al contrario, i bias sono correttamente stimati in entrambi i casi.

Stima con bias lentamente variabili

A questo punto la stessa traiettoria è stata utilizzata, per l'AHRS multi-mode, imponendo dei valori lentamente variabili per le polarizzazioni sui giroscopi. I risultati presentati nelle Figs. 4.27 e 4.28 dimostrano che, anche in questo caso, il nuovo AHRS stima correttamente sia bias che gli angoli di Eulero, fornendo al controllore i valori opportuni per permettere di portare a termine la missione assegnata (Fig. 4.29).

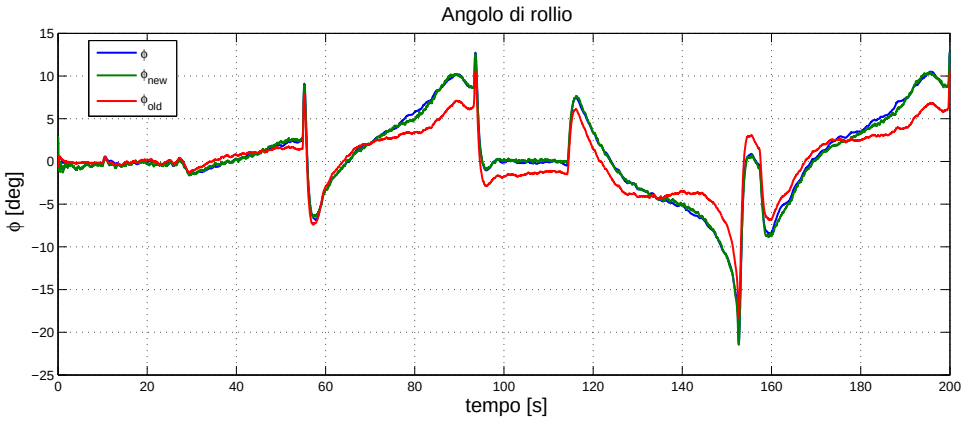


Figura 4.23: Nuova traiettoria: confronto stima angolo di rollio. L'AHRS multi-mode (curva in verde) riesce a stimare correttamente il riferimento fornito dal modello (curva in blu), al contrario dello stimatore standard (curva in rosso).

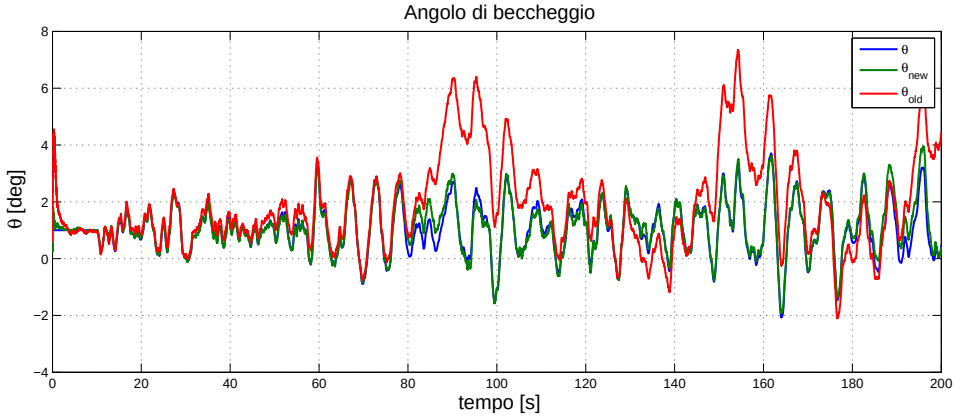


Figura 4.24: Nuova traiettoria: confronto stima angolo di beccheggio. L'AHRS multi-mode (curva in verde) riesce a stimare correttamente il riferimento fornito dal modello (curva in blu), al contrario dello stimatore standard (curva in rosso).

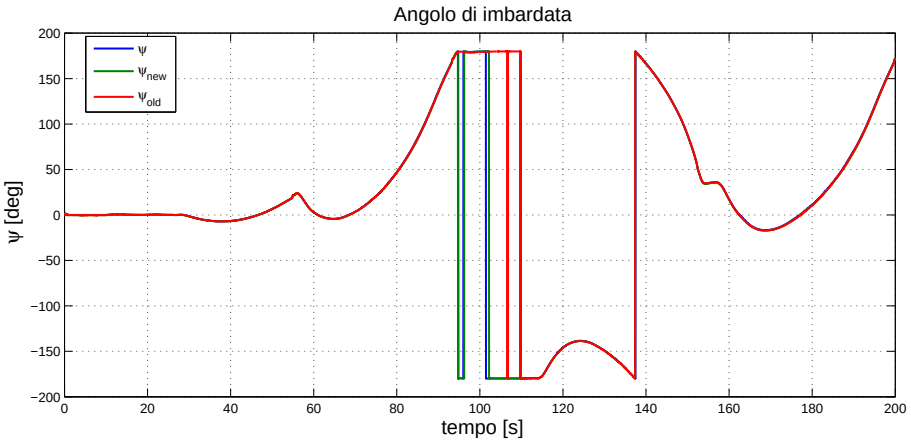


Figura 4.25: Nuova traiettoria: confronto stima angolo di imbardata. L'AHRS multi-mode (curva in verde) riesce a stimare correttamente il riferimento fornito dal modello (curva in blu), al contrario dello stimatore standard (curva in rosso).

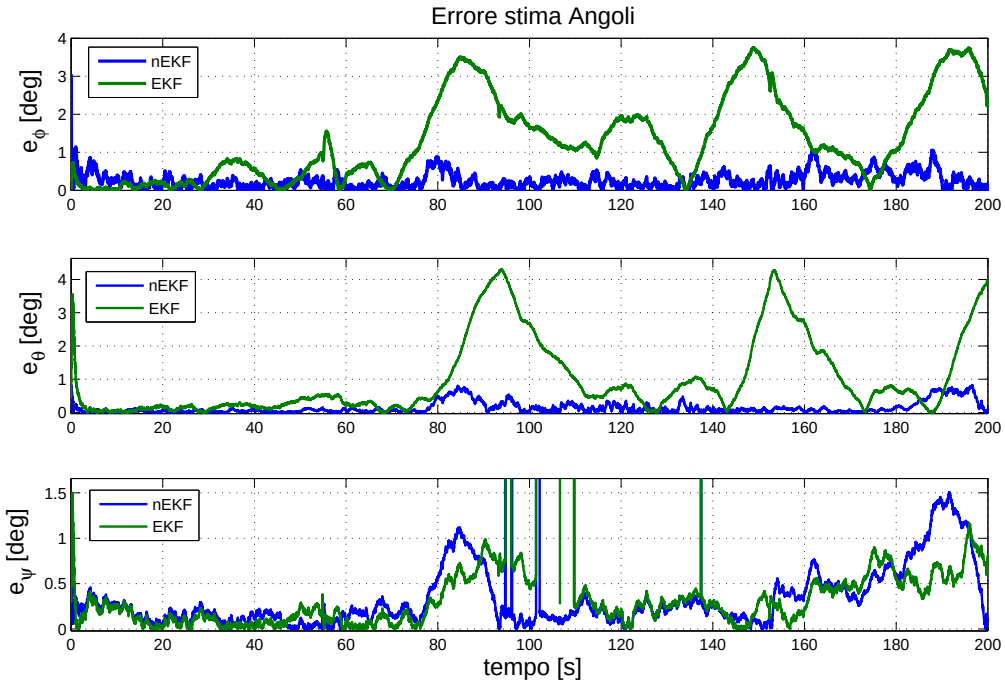


Figura 4.26: Nuova traiettoria: errori di stima. L'errore assoluto di stima del filtro standard (curve in verde) è maggiore rispetto a quello dello stimatore multi-mode (curve in blu).

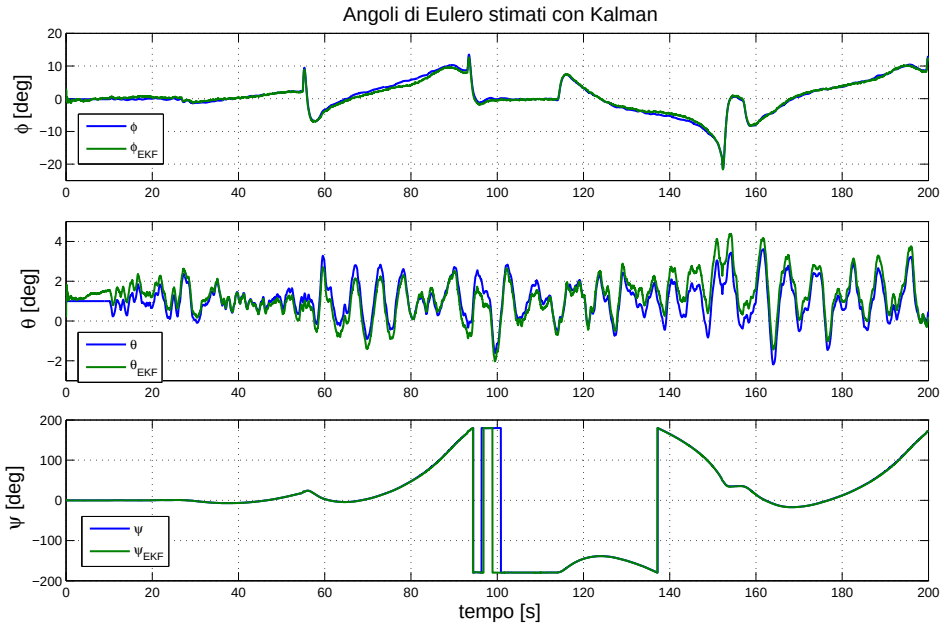


Figura 4.27: Nuova traiettoria: stima angoli di Eulero con bias sui giroscopi lentamente variabili. La stima degli angoli di Eulero (in verde) eseguita dall'AHRS multi-mode insegue correttamente i riferimenti forniti dal modello (in blu).

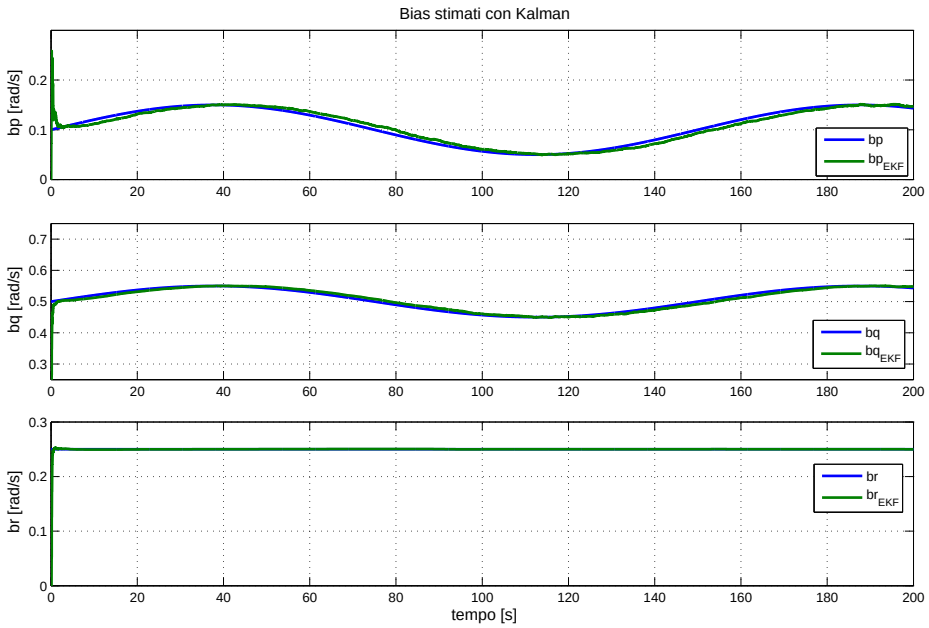


Figura 4.28: Nuova traiettoria: stima bias sui giroscopi lentamente variabili. La stima dei bias (in verde) eseguita dall'AHRS multi-mode insegue correttamente i riferimenti forniti dal modello (in blu).

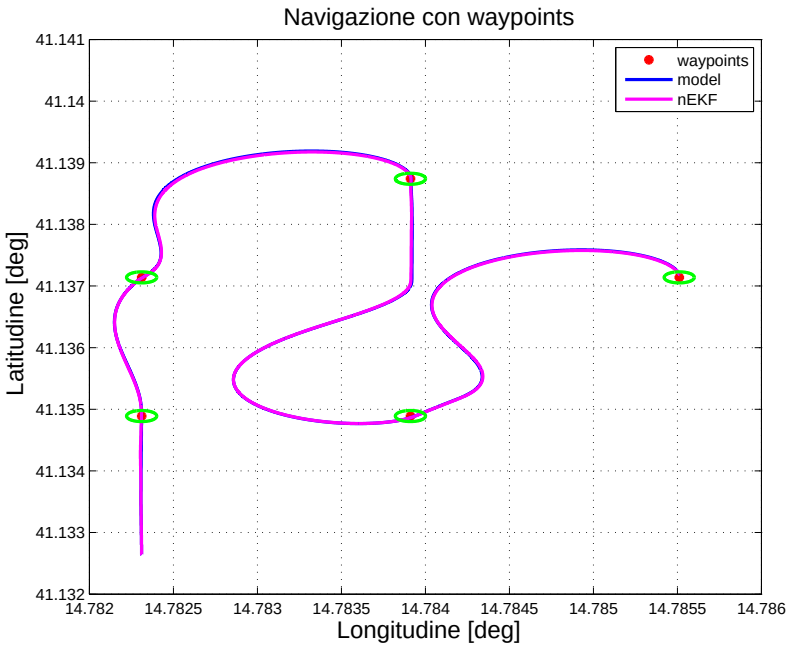


Figura 4.29: Nuova traiettoria a ciclo chiuso con bias sui giroscopi lentamente variabili. In viola è rappresentata la traiettoria ottenuta chiudendo il loop di controllo sulle grandezze stimate dal filtro multi-mode; in blu è rappresentata la traiettoria di riferimento del modello.

Stima in condizioni perturbate

Un'ultima prova è stata effettuata considerando la presenza di vento. La scelta dell'entita del disturbo da considerare è stata guidata dall'osservazione che il prototipo Yardstik è un aeromodello elettrico adatto ai voli indoor e che male si presta a voli in presenza di forti raffiche di vento. Per questo, è stata supposta la presenza di una *brezza leggera* di vento caratterizzata da velocità comprese tra 1.6 e 3.3 m/s, secondo la *scala di Beaufolt*[2].

In Fig. 4.30 sono presentati i risultati relativi alla stima degli angoli di Eulero. E' facile notare che la stima subisce alcuni peggioramenti soprattutto in corrispondenza dell'angolo di rollio e di beccheggio. I bias lentamente variabili, al contrario, continuano ad essere stimati correttamente (Fig. 4.31). Benché in presenza di vento si verifichi una degradazione della stima dell'assetto, i waypoints di riferimento continuano ad essere raggiunti correttamente, come dimostra la Fig. 4.32. Lo scostamento tra la traiettoria a ciclo chiuso (curva in viola) e quella di riferimento fornita dal modello (curva in blu) è accettabile se viene considerato anche il fatto che la traiettoria di riferimento è stata ottenuta chiudendo il loop di controllo sulle variabili fornite dalla simulazione del modello del velivolo in condizioni non perturbate.

[2]La Scala di Beaufort è una misura empirica dell'intensità del vento basata originariamente sullo stato del mare o le condizioni delle onde. Il merito di avere perfezionato, nel 1805, una scala contenente dei criteri relativamente precisi per quantificare il vento in mare si deve all'ammiraglio britannico Francis Beaufort. In seguito altri criteri furono aggiunti per estendere la sua applicazione anche a terra. Questo sistema di valutazione ha validità internazionale dal 1° Gennaio 1949 [37].

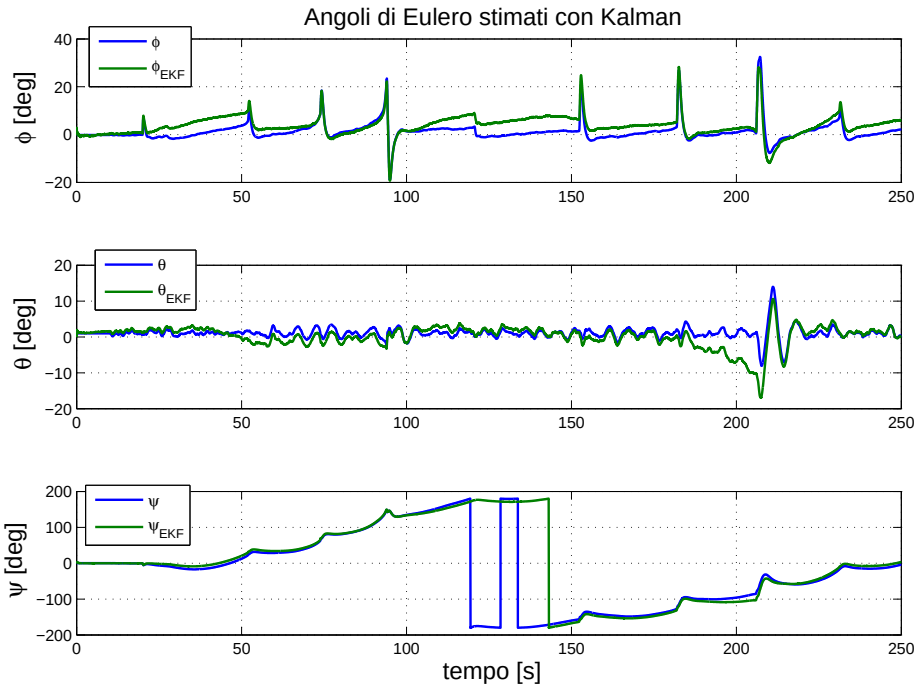


Figura 4.30: Condizioni perturbate: stima d'assetto. In presenza di vento la stima degli angoli di Eulero dell'AHRS multi-mode (curve in verde) subisce qualche degradazione.

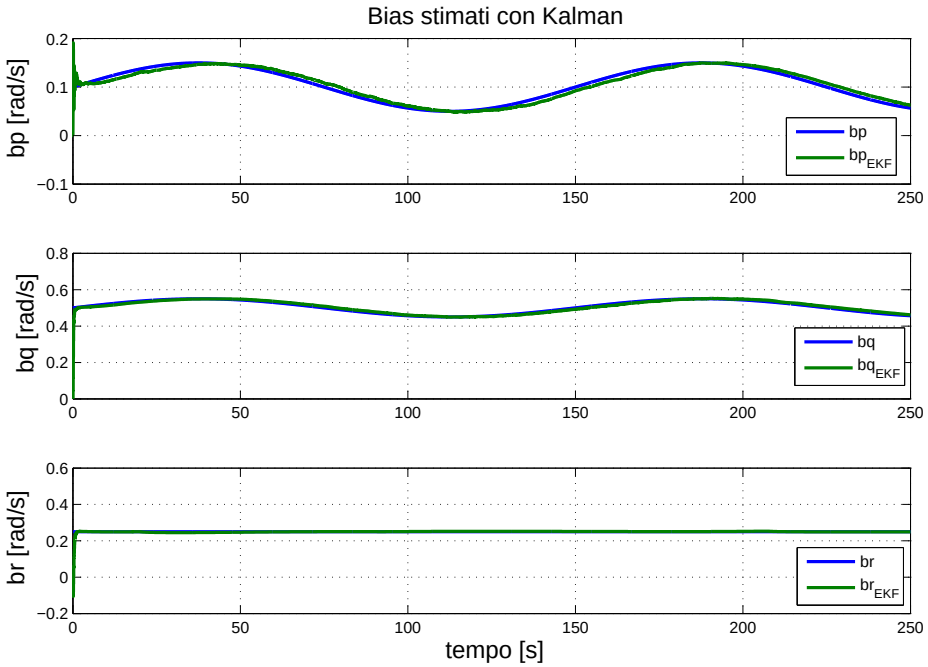


Figura 4.31: Condizioni perturbate: stima dei bias sui giroscopi. In presenza di vento le polarizzazioni sui giroscopi (curve in verde) continuano ad essere stimate correttamente.

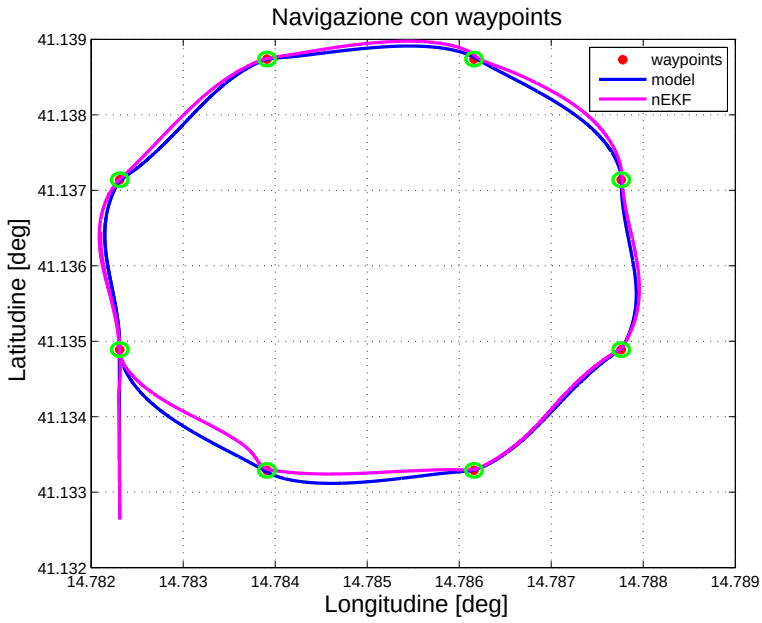


Figura 4.32: Condizioni perturbate: traiettoria 2D. In presenza di vento la traiettoria ottenuta chiudendo il loop di controllo sulle grandezze stimate (curva in viola) si discosta leggermente da quello di riferimento fornita dal modello (curva in blu).

Conclusioni

Il lavoro di questa tesi ha riguardato lo sviluppo e la validazione di un filtro di Kalman esteso per la stima d'assetto di un velivolo.

Nel Capitolo 1 sono state presentate le equazioni che descrivono il moto di un velivolo ed è stato introdotto il sistema di controllo del volo (FCS) realizzato in precedenti lavori. La trattazione ha permesso sia di introdurre il sistema UAV che di inquadrare lo scenario di riferimento nel quale si colloca il presente lavoro.

Nel Capitolo 2 è stata affrontata la tematica riguardante la stima dell'assetto di un velivolo ed è stato presentato l'AHRS nella formulazione attualmente utilizzata nel settore della navigazione e controllo di veicoli robotici di terra e di aria in miniatura. Tale soluzione utilizza, nel set di misure, gli accelerometri che, in condizioni dinamiche di volo, sono fonte di imprecisione per la stima dell'assetto.

Nel Capitolo 3, dopo un'attenta analisi della letteratura che ha permesso di individuare le principali problematiche connesse ad uno stimatore standard, valido in condizioni stazionarie, è stata descritta l'implementazione di un algoritmo di stima multi-mode, più performante rispetto a quello presentato nel capitolo precedente, che utilizza in modo ottimale il set di misure a disposizione. Questo ha portato ad impiegare principalmente le misure provenienti dai magnetometri e GPS, limitando al minimo quelle provenienti dagli acceleromentri e sviluppando una modalità di emergenza in caso di perdita del segnale GPS. Come osservatore è stato utilizzato il filtro di Kalman esteso, applicato ad una rappresentazione tramite quaternioni della dinamica dell'assetto

Nel Capitolo 4, infine, l'algoritmo di stima multi-mode è stato validato mediante la tecnica Software-In-The-Loop. I risultati sono stati confrontati con quelli dello stimatore standard per verificare le migliorie apportate, in diversi scenari di funzionamento.

Principali contributi

Rispetto ai lavori precedenti e allo standard presente in letteratura, il filtro di Kalman esteso sviluppato nel presente lavoro ha apportato dei miglioramenti alla stima degli angoli di Eulero.

Nella versione standard la grandezza che risente maggiormente delle problematiche relative all'utilizzo degli accelerometri era l'angolo di rollio. Quest'ultimo, infatti, presenta generalmente dinamiche molto accentuate in corrispondenza di variazioni di moto nel piano latero-direzionale; in queste condizioni, lo stimatore standard non fornisce valori adeguati, introducendo un offset dovuto alla non corretta descrizione del vettore accelerazione. Lo stimatore multi-mode, al contrario, rimpiazzando le misure di accelerazione, ha permesso di stimare correttamente tale grandezza.

Lo stesso risultato è stato verificato imponendo al velivolo una traiettoria nel piano lateralo-direzionale che richiedesse al velivolo di compiere virate più spinte. Anche in questo caso il nuovo stimatore ha mostrato un buon comportamento, mentre le prestazioni del filtro standard sono peggiorate anche in relazione all'angolo di beccheggio. Tali peggioramenti non compromettono completamente il controllo del volo ma impongono al velivolo di compiere oscillazioni anche nel piano longitudinale del moto.

Un'altra problematica è relativa alle ipotesi adottate per le polarizzazioni dei giroscopi. In un primo approccio esse sono state considerate costanti e l'AHRS standard è stato in grado di stimarle. In realtà le polarizzazioni dipendono dalla temperatura e un modello più accurato della dinamica d'assetto impone che tali bias siano considerati lentamente variabili. In questa ipotesi lo stimatore standard fornisce delle stime non esatte, compromettendo il comportamento complessivo del sistema di controllo di volo. Al contrario l'AHRS multi-mode, attraverso una descrizione dinamica della matrice di errore di processo, è riuscito a stimare correttamente anche queste variazioni nelle polarizzazioni.

Sviluppi futuri

A seguito dei risultati ottenuti dalle simulazioni *Software in the loop*, i prossimi passi da realizzare sono essenzialmente orientati verso le successive fasi di validazione:

- *processor in the loop*, in cui si dovrà testare il codice del filtro in esecuzione "real time" sul microprocessore interfacciato all'ambiente di simulazione;

- *test di volo* con il velivolo per valutare le prestazioni degli algoritmi nell'ambiente reale. In tale fase sarà necessario anche effettuare la calibrazione dei sensori, l'interfacciamento degli stessi con il microprocessore e sarà necessario gestire la fase di acquisizione delle grandezze misurate.

Ad ogni modo, benché i risultati ottenuti in questo lavoro siano soddisfacenti, non è preclusa la possibilità di modificare o migliorare lo stimatore presentato, in base ad eventuali informazioni che potrebbero essere acquisite durante le successive fasi di testing.

Altri scenari che si aprono per futuri lavori sono nel seguito riportati.

- Miglioramento della stima in condizione perturbate - In presenza di vento la stima degli angoli di Eulero subisce dei peggioramenti. Si potrebbe pensare di effettuare una stima del campo di velocità del vento per compensare tale disturbo.

- Completamento della struttura del filtro - In questo lavoro è stata considerata esclusivamente la fase di volo. Si potrebbe pensare di completare la trattazione considerando anche la fase di decollo e di atterraggio. Questo potrebbe portare alla modifica o, eventualmente, all'aggiunta di modi di funzionamento, anche alla luce delle diverse dinamiche che entrerebbero in gioco.

- Modifiche del metodo di discretizzazione del modello del filtro - Il passo di predizione dell'algoritmo di stima è stato ottenuto discretizzando la dinamica dell'assetto con un metodo del primo ordine (Eulero in avanti). Si potrebbe pensare di utilizzare schemi di discretizzazione di ordine superiore che possano fornire predizioni più accurate, facendo attenzione a non appesantire eccessivamente l'onere computazionale.

Bibliografia

[1] *www.aem.umn.edu/uav*, Sito UAV.

[2] D. Meola, "Sviluppo di un aereo in miniatura sprovvisto di pilota." Thesis, Universita' degli Studi del Sannio, 2007.

[3] M. Kazazi, "Sviluppo di un aereo in miniatura sprovvisto di pilota (2)." Thesis, Universita' degli Studi del Sannio, 2007.

[4] D. Meola, "Sistema di controllo di volo per velivoli privi di pilota: progettazione e validazione software-in-the-loop," Master's thesis, Universita' degli Studi del Sannio, 2011.

[5] M. Kazazi, "Simulazione hardware-in-the-loop di un controllore di volo per velivoli privi di pilota," Master's thesis, Universita' degli Studi del Sannio, 2012.

[6] *www.electriy.com/parkyers/gpma1100.html*, Modello yardstick.

[7] *2011 - 2012 UAS Yearbook - UAS: The Global Perspective*, 9th ed., Blyenburgh & Co., June 2011.

[8] *www.hitecrcd.com/Servo/hs55.htm*, Servo Hitec.

[9] *www.xbow.com*, Crossbow.

[10] *www.freescale.com*, Freescale semiconductor.

[11] *www.maxstream.net/products/xstream/oemrf-module.php*, MaxStream Data Modem.

[12] J. S. Jang and D. Liccardo, "Automation of small uavs using a low cost mems sensor and embedded computing platform," in *25th Digital Avionics Systems Conference, 2006 IEEE/AIAA*, oct. 2006, pp. 1 –9.

[13] B. Stevens and F. Lewis, *Aircraft control and simulation*, 2nd ed. Wiley.

[14] Siciliano, Sciavicco, Villani, and Oriolo, *Robotica, modellistica pianificazione e controllo*, 3rd ed. McGraw-Hill.

[15] R. C. Nelson, *Flight stability and automatic control*. McGraw-Hill.

[16] Franklin, Powell, and Emami-Naeini, *Feedback control of dynamic systems*, 3rd ed. Addison-Wesley.

[17] P. Y. Chai, "Synthesis and validation of flight control for uav," Ph.D. dissertation, University of Minnesota.

[18] A. Healey and D. Lienard, "Multivariable sliding mode control for autonomous diving and steering of unmanned underwater vehicles," *Oceanic Engineering, IEEE Journal of*, vol. 18, no. 3, pp. 327 –339, jul 1993.

[19] V. Bakaric, Z. Vukic, and R. Antonic, "Improved basic planar algorithm of vehicle guidance through waypoints by the line of sight," in *Control, Communications and Signal Processing, 2004. First International Symposium on*, 2004, pp. 541 – 544.

[20] L. Daga, *Filtro di Kalman*, http://leonardodaga.insyde.it/IngSis/FiltroKalman.htm.

[21] D. B. Kingston and R. W. Beard, *Real-time attitude and position estimation for small UAVs using low-cost sensors*, Department of Electrical and Computer Engineering, Brigham Young University.

[22] *Compass heading using magnetometers*, Honeywell.

[23] M. J. Caruso, *Applications of magnetoresistive sensors in navigation systems*, Honeywell Inc.

[24] H. de Marina, F. Pereda, J. Giron-Sierra, and F. Espinosa, "Uav attitude estimation using unscented kalman filter and triad," *Industrial Electronics, IEEE Transactions on*, vol. 59, no. 11, pp. 4465 –4474, nov. 2012.

[25] Y. Hao, Z. Xiong, F. Sun, and X. Wang, "Comparison of unscented kalman filters," in *Mechatronics and Automation, 2007. ICMA 2007. International Conference on*, aug. 2007, pp. 895 –899.

[26] M. D. Shuster and S. D. Oh, "Three-axis attitude determination from vector observations," *Journal of Guidance and Control*, vol. 4, no. 1, pp. 70 – 77, jan 1981.

[27] S. Julier and J. Uhlmann, "Unscented filtering and nonlinear estimation," *Proceedings of the IEEE*, vol. 92, no. 3, pp. 401 – 422, mar 2004.

[28] X. Hu, Q. Li, C. He, and Y. Liu, "An adaptive kalman filter for three dimensional attitude tracking," in *VR Innovation (ISVRI), 2011 IEEE International Symposium on*, march 2011, pp. 151 –154.

[29] C. Liu, Z. Zhou, and X. Fu, "Attitude determination for mavs using a kalman filter," *Tsinghua Science and Technology*, vol. 13, no. 5, pp. 593 –597, oct. 2008.

[30] P. Martin and E. Salaun, "Generalized multiplicative extended kalman filter for aided attitude and heading reference system," *AIAA Guidance, Navigation and Control Conference*, aug. 2010.

[31] Y. Kang and J. Hedrick, "Linear tracking for a fixed-wing uav using nonlinear model predictive control," *Control Systems Technology, IEEE Transactions on*, vol. 17, no. 5, pp. 1202 –1210, sept. 2009.

[32] M. Cordoba, "Attitude and heading refernce system i-ahrs for the efigenia autonomous unmanned aerial vehicles uav based on mems sensor and a neural network strategy for attitude estimation," in *Control Automation, 2007. MED '07. Mediterranean Conference on*, june 2007, pp. 1 –8.

[33] N. Metni, J.-M. Pflimlin, T. Hamel, and P. Soueres, "Attitude and gyro bias estimation for a flying uav," in *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on*, aug. 2005, pp. 1114 – 1120.

[34] P. Martin and E. Salan, "Design and implementation of a low-cost observer-based attitude and heading reference system," in *Control Engineering Practice*, vol. 18, jul. 2010, pp. 712 – 722.

[35] P. Bauer and J. Bokor, "Development and hardware-in-the-loop testing of an extended kalman filter for attitude estimation," in *Computational Intelligence and Informatics (CINTI), 2010 11th International Symposium on*, nov. 2010, pp. 57 –62.

[36] D. Gebre-Egziabher, R. Hayward, and J. Powell, "A low-cost gps/inertial attitude heading reference system (ahrs) for general aviation applications," in *Position Location and Navigation Symposium, IEEE 1998*, 1998, pp. 518 – 525.

[37] *http://www.eurometeo.com*, Sito meteorologico EuroMeteo - La Scala di Beaufort.

Appendice A

Codici degli algoritmi di stima

In questa appendice vengono riportati i codici in linguaggio C degli algoritmi di stima sintetizzati nel presente lavoro di tesi. Gli algoritmi sono stati implementati tramite S-Function. La S-function è la descrizione in un linguaggio di programmazione di un blocco Simulink e si compone di una particolare sintassi che la abilita ad interagire con gli *equation solvers* dell'ambiente di simulazione. Le funzioni principali di tale sintassi sono:

- *static void mdlInitializeConditions*: permette l'inizializzazione delle variabili di stato del modello implementato nel filtro;
- *static void mdlStart*: è la funzione che viene chiamata un'unica volta all'inizio dell'esecuzione. Permette di inizializzare altre grandezze dell'algoritmo e definisce le variabili che devono essere scambiate in passi di integrazione successivi;
- *static void mdlOutputs*: contiene il corpo dell'algoritmo. E' la funzione che viene eseguita ad ogni passo di integrazione e permette di definire le uscite dell'algoritmo.

A.1 AHRS standard

```
#define S_FUNCTION_NAME  KalmanFilter
#define S_FUNCTION_LEVEL 2

#include "simstruc.h"
#include "matrix.c"
#include <time.h>
```

```
#include <stdio.h>

#define NIN 3
#define NOUT 5
#define NPAR 0

/*magnetometer hard-iron calibration   */
#define         bBy     0.0
#define         bBx     0.0
#define         sfx      1
#define         sfy      1
/*err covariance of accelerometers:    */
#define         var_az  1.962361        //(0.1*g)^2
#define         var_ax  1.962361
#define         var_ay  1.962361

/*=========================================================*
 *     sensor noise characteristics                        *
 *=========================================================*/

//predefined variables
#define g     9.81 //m/sec^2
#define g2      19.62    //2*g
#define r2d     57.2958     //radiant to degree
#define d2r     0.01745 //degree to radiant
#define pi      3.141592
#define pi2     6.283184 //pi*2

#define    ANGLE_Update  25      // Angle update in Hz
#define    MAG_Update    10      // Mag. update in Hz
#define    AHRS_Run      50      // Ahrs runs at 50 Hz

//err covariance of magnetometer heading
#define     var_psi   0.014924  //(7*d2r)^2

//sign function
#define         sign(arg) (arg>=0 ? 1:-1)

/*=========================================================*
 *     Initialization                                      *
 *=========================================================*/

static void mdlInitializeSizes(SimStruct *S)
{
```

```
 ssSetNumSFcnParams(S, NPAR);
 if(ssGetNumSFcnParams(S) != ssGetSFcnParamsCount(S))   return;

 if(!ssSetNumInputPorts(S, NIN)) return;

    {
        int_T   i;
        for(i = 0; i < NIN; i++){
            ssSetInputPortWidth(S, i, DYNAMICALLY_SIZED);
            ssSetInputPortDirectFeedThrough(S, i, 1);
        }
    }

    if(!ssSetNumOutputPorts(S,NOUT))    return;

    {
        int_T i;
        for(i = 0; i < NOUT; i++)
            ssSetOutputPortWidth(S, i, DYNAMICALLY_SIZED);
     }

    ssSetNumContStates(S, 0);
    ssSetNumDiscStates(S, 7);
    ssSetNumSampleTimes(S, 1);
    ssSetNumRWork(S, 3);
    ssSetNumIWork(S, 2);
    ssSetNumPWork(S, 16);
    ssSetNumModes(S, 0);

}


/*=====================================================*
 *              Initialize Sample Time                 *
 *=====================================================*/

static void mdlInitializeSampleTimes(SimStruct *S)
{
       ssSetSampleTime(S, 0, 0.02);
       ssSetOffsetTime(S, 0, 0.0);

}
```

```
#define MDL_INITIALIZE_CONDITIONS
#if defined(MDL_INITIALIZE_CONDITIONS)

  static void mdlInitializeConditions(SimStruct *S)
  {
      real_T    *xs = ssGetRealDiscStates(S);
      int_T     numStates  = ssGetNumDiscStates(S);
      int_T     i;

      xs[0] = 1.0;
      for(i = 1; i < 7; i++)
          xs[i] = 0.0;
  }
#endif /* MDL_INITIALIZE_CONDITIONS */


/*====================================================*
 *                     Simulation Start Time          *
 *====================================================*/

#define MDL_START
#if defined(MDL_START)

  static void mdlStart(SimStruct *S)
{
      //real_T    *xs= ssGetRealDiscStates(S);
      // global variable

      MATRIX aP,aQ,aR,aK,Fsys,Hj,Iden;
      MATRIX tmp73,tmp33,tmp77,tmpr,Rinv,mat77;
      MATRIX Hpsi,Kpsi,tmp71;

      double tempPhi, tempThe, tempPsi;

      short vgCheck,magCheck,count;

      if(ssIsFirstInitCond(S)){

//initialization of err, meas, and process cov. matrices
          aP = mat_creat(7,7,ZERO_MATRIX);
          aQ = mat_creat(7,7,ZERO_MATRIX);
          aR = mat_creat(3,3,ZERO_MATRIX);
```

```
aP[0][0]= 1.0e-1;
aP[1][1]= 1.0e-1;
aP[2][2]= 1.0e-1;
aP[3][3]= 1.0e-1;
aP[4][4]= aP[5][5]=aP[6][6]=1.0e-1;

aQ[0][0]= 1e-8;
aQ[1][1]= 1e-8;
aQ[2][2]= 1e-8;
aQ[3][3]= 1e-8;
aQ[4][4]= aQ[5][5]=aQ[6][6]=1.0e-12;

aR[0][0]= var_ax;
aR[1][1]= var_ax;
aR[2][2]= var_ax;

//initialization of gain matrix
aK = mat_creat(7,3,ZERO_MATRIX);

//initialization of state transition matrix
Fsys = mat_creat(7,7,UNIT_MATRIX);

//initialization of Identity matrix
Iden = mat_creat(7,7,UNIT_MATRIX);

//initialization of Jacobian matrix
Hj   = mat_creat(3,7,ZERO_MATRIX);

//initialization related to heading
Hpsi  = mat_creat(1,7,ZERO_MATRIX);
Kpsi  = mat_creat(7,1,ZERO_MATRIX);
tmp71 = mat_creat(7,1,ZERO_MATRIX);

//initialization of other matrice used in AHRS
Rinv  = mat_creat(3,3,ZERO_MATRIX);
tmp33 = mat_creat(3,3,ZERO_MATRIX);
tmp73 = mat_creat(7,3,ZERO_MATRIX);
tmp77 = mat_creat(7,7,ZERO_MATRIX);
tmpr  = mat_creat(7,7,ZERO_MATRIX);
mat77 = mat_creat(7,7,ZERO_MATRIX);

vgCheck=0,magCheck=0; count=0;

tempThe = 0.0;
```

```
          tempPhi = 0.0;
          tempPsi = 0.0;

          ssSetPWorkValue(S, 0, aP);
          ssSetPWorkValue(S, 1, aQ);
          ssSetPWorkValue(S, 2, aR);
          ssSetPWorkValue(S, 3, aK);
          ssSetPWorkValue(S, 4, Fsys);
          ssSetPWorkValue(S, 5, Iden);
          ssSetPWorkValue(S, 6, Hj);
          ssSetPWorkValue(S, 7, Hpsi);
          ssSetPWorkValue(S, 8, Kpsi);
          ssSetPWorkValue(S, 9, tmp71);
          ssSetPWorkValue(S, 10, Rinv);
          ssSetPWorkValue(S, 11, tmp33);
          ssSetPWorkValue(S, 12, tmp73);
          ssSetPWorkValue(S, 13, tmp77);
          ssSetPWorkValue(S, 14, tmpr);
          ssSetPWorkValue(S, 15, mat77);

          ssSetIWorkValue(S, 0, vgCheck);
          ssSetIWorkValue(S, 1, count);

          ssSetRWorkValue(S, 0, tempPhi);
          ssSetRWorkValue(S, 1, tempThe);
          ssSetRWorkValue(S, 2, tempPsi);
    }

  }
#endif /*  MDL_START */

//+++++++++++++++++++++++++++++++++++++++++++
// wrap around for -180 and + 180
//+++++++++++++++++++++++++++++++++++++++++++

double wraparound(double dta)
{

   //bound heading angle between -180 and 180
    if(dta >  pi) dta -= pi2;
    if(dta < -pi) dta += pi2;

   return dta;
}
```

```
/*=========================================*
 *          Output                         *
 *=========================================*/

static void mdlOutputs(SimStruct *S, int_T tid)
{
    // local variables

    static double tnow,tprev=0;
    double pc,qc,rc;
    double h[3]={0.,},cPHI,sPHI;
    double xsn[4]={0.,};
    double norm,Bxc,Byc,invR;
    double dt,Hdt;
    double coeff1[3]={0,},temp[2]={0,};
    short  i=0,j=0;


    double tempHx,tempHy,tempHz;
    double tempPsi,tempThe,tempPhi,prova;
    double tempAx,tempAy,tempAz;
    double tempP,tempQ,tempR;

    real_T    *y, *y1, *y2, *y3, *y4;

    real_T    *xs= ssGetRealDiscStates(S);
    MATRIX aP,aQ,aR,aK,Fsys,Hj,Iden;
    MATRIX tmp73,tmp33,tmp77,tmpr,Rinv,mat77;
    MATRIX Hpsi,Kpsi,tmp71;

    short  vgCheck,magCheck,count;


    aP = ssGetPWorkValue(S, 0);
    aQ = ssGetPWorkValue(S, 1);
    aR = ssGetPWorkValue(S, 2);
    aK = ssGetPWorkValue(S, 3);
    Fsys = ssGetPWorkValue(S, 4);
    Iden = ssGetPWorkValue(S, 5);
    Hj = ssGetPWorkValue(S, 6);
    Hpsi = ssGetPWorkValue(S, 7);
    Kpsi = ssGetPWorkValue(S, 8);
    tmp71 = ssGetPWorkValue(S, 9);
    Rinv = ssGetPWorkValue(S, 10);
```

```
tmp33 = ssGetPWorkValue(S, 11);
tmp73 = ssGetPWorkValue(S, 12);
tmp77 = ssGetPWorkValue(S, 13);
tmpr = ssGetPWorkValue(S, 14);
mat77 = ssGetPWorkValue(S, 15);

vgCheck = ssGetIWorkValue(S, 0);
count = ssGetIWorkValue(S, 1);

tempPhi = ssGetRWorkValue(S, 0);
tempThe = ssGetRWorkValue(S, 1);
tempPsi = ssGetRWorkValue(S, 2);


tempHx = *ssGetInputPortRealSignalPtrs(S, 0)[0];
tempHy = *ssGetInputPortRealSignalPtrs(S, 0)[1];
tempHz = *ssGetInputPortRealSignalPtrs(S, 0)[2];
tempAx = *ssGetInputPortRealSignalPtrs(S, 1)[0];
tempAy = *ssGetInputPortRealSignalPtrs(S, 1)[1];
tempAz = *ssGetInputPortRealSignalPtrs(S, 1)[2];
tempP = *ssGetInputPortRealSignalPtrs(S, 2)[0];
tempQ = *ssGetInputPortRealSignalPtrs(S, 2)[1];
tempR = *ssGetInputPortRealSignalPtrs(S, 2)[2];

y =  ssGetOutputPortRealSignal(S,0); //Euler Angles
y1 = ssGetOutputPortRealSignal(S,1); //bias
y2 = ssGetOutputPortRealSignal(S,2); //h(x)
y3 = ssGetOutputPortRealSignal(S,3); //quaternions
y4 = ssGetOutputPortRealSignal(S,4); //quaternions

//snap the time interval, dt, of this routine
dt = 0.02;

Hdt = 0.5*dt;

/*assign new variables */
pc = (tempP - xs[4])*Hdt;
qc = (tempQ - xs[5])*Hdt;
rc = (tempR - xs[6])*Hdt;

/*state transition matrix */

Fsys[0][1] = -pc; Fsys[0][2] = -qc; Fsys[0][3] = -rc;
Fsys[1][0] =  pc; Fsys[1][2] =  rc; Fsys[1][3] = -qc;
```

```
Fsys[2][0] =  qc; Fsys[2][1] = -rc; Fsys[2][3] =  pc;
Fsys[3][0] =  rc; Fsys[3][1] =  qc; Fsys[3][2] = -pc;

Fsys[0][4] = xs[1]*Hdt;  Fsys[0][5] = xs[2]*Hdt;
Fsys[0][6] = xs[3]*Hdt;
Fsys[1][4] =-xs[0]*Hdt;  Fsys[1][5] = xs[3]*Hdt;
Fsys[1][6] =-Fsys[0][5];
Fsys[2][4] =-Fsys[1][5]; Fsys[2][5] = Fsys[1][4];
Fsys[2][6] = Fsys[0][4];
Fsys[3][4] = Fsys[0][5]; Fsys[3][5] =-Fsys[0][4];
Fsys[3][6] = Fsys[1][4];


//+++++++++++++++++++++++++++++++++++++++++
//Extended Kalman filter: prediction step
//+++++++++++++++++++++++++++++++++++++++++
/*propagation of quaternion using gyro measurement
  at a given sampling interval dt */

xsn[0] = xs[0] - pc*xs[1] - qc*xs[2] - rc*xs[3];
xsn[1] = xs[1] + pc*xs[0] - qc*xs[3] + rc*xs[2];
xsn[2] = xs[2] + pc*xs[3] + qc*xs[0] - rc*xs[1];
xsn[3] = xs[3] -pc*xs[2] + qc*xs[1] + rc*xs[0];

for(i=0; i<4; i++) xs[i] = xsn[i];

//error covriance propagation: P = Fsys*P*Fsys' + Q
mat_mymul2(Fsys,aP,tmp77,3);
mat_mymul3(tmp77,Fsys,aP,3);

for(i=0;i<7;i++)
    aP[i][i] += aQ[i][i];

count++;
// Pitch and Roll Update at 25 Hz
 if(count%(AHRS_Run/ANGLE_Update)==0)
{
 // ++++++++++++++++++++++++++++++++++++++++++
  //EKF: correction step for pitch and roll
 // ++++++++++++++++++++++++++++++++++++++++++
 //nonlinear measurement equation of h(x)
 h[0]=-g2*(xs[1]*xs[3]-xs[0]*xs[2]);
 h[1]=-g2*(xs[0]*xs[1]+xs[2]*xs[3]);
 h[2]=-g*(xs[0]*xs[0]-xs[1]*xs[1]-xs[2]*xs[2]+xs[3]*xs[3]);
```

```
      y2[0]=h[0];
      y2[1]=h[1];
      y2[2]=h[2];

     // compute Jacobian matrix of h(x)
     Hj[0][0] = g2*xs[2]; Hj[0][1] =-g2*xs[3];
     Hj[0][2] = g2*xs[0]; Hj[0][3] = -g2*xs[1];
     Hj[1][0] = Hj[0][3]; Hj[1][1] =-Hj[0][2];
     Hj[1][2] = Hj[0][1]; Hj[1][3] = -Hj[0][0];
     Hj[2][0] =-Hj[0][2]; Hj[2][1] =-Hj[0][3];
     Hj[2][2] = Hj[0][0]; Hj[2][3] =  Hj[0][1];

     // gain matrix aK = aP*Hj'*(Hj*aP*Hj' + aR)^-1
      mat_mymul4(aP,Hj,tmp73,3);
      mat_mymul(Hj,tmp73,tmp33,3);

      for(i=0;i<3;i++) tmp33[i][i] += aR[i][i];
      mat_inv(tmp33,Rinv);
      mat_mul(tmp73,Rinv,aK);

      //state update
      for(i=0;i<7;i++)
      {
   xs[i] += aK[i][0]*(tempAx - h[0])
         +  aK[i][1]*(tempAy - h[1])
         +  aK[i][2]*(tempAz - h[2]);
      }


      //error covariance matrix update aP = (I - aK*Hj)*aP
      mat_mymul1(aK,Hj,mat77,3);
      mat_sub(Iden,mat77,tmpr);
      mat_mymul5(tmpr,aP,tmp77,3);
      mat_copy(tmp77,aP);
    }


   // Heading update at 10 Hz
   if(count%(AHRS_Run/MAG_Update)==1)
   {

// ++++++++++++++++++++++++++++++++++++++++++++++++++++
// second stage EKF update to estimate the heading angle
```

```
// +++++++++++++++++++++++++++++++++++++++++++++++++++++++
//  hard-iron calibration
tempHx = sfx*(tempHx) - bBx;
tempHy = sfy*(tempHy) - bBy;
//
// magnetic heading correction due to roll and pitch angle
cPHI= cos(tempPhi);
sPHI= sin(tempPhi);
Bxc=(tempHx)*cos(tempThe)+((tempHy)*sPHI+(tempHz)*cPHI)*sin(tempThe);
Byc=(tempHy)*cPHI-(tempHz)*sPHI;

//Jacobian
norm = 1.0/sqrt(xs[0]*xs[0]+xs[1]*xs[1]+xs[2]*xs[2]+xs[3]*xs[3]);
 for(i=0;i<4;i++) xs[i] = xs[i]*norm;

 coeff1[0]= 2*(xs[1]*xs[2]+ xs[0]*xs[3]);
 coeff1[1]= 1 - 2*(xs[2]*xs[2]+xs[3]*xs[3]);
 coeff1[2]= 2/(coeff1[0]*coeff1[0]+coeff1[1]*coeff1[1]);

 temp[0] = coeff1[1]*coeff1[2];
 temp[1] = coeff1[0]*coeff1[2]*coeff1[1];

 Hpsi[0][0] = xs[3]*temp[0];
 Hpsi[0][1] = xs[2]*temp[0];
 Hpsi[0][2] = xs[1]*temp[0]+2*xs[2]*temp[1];
 Hpsi[0][3] = xs[0]*temp[0]+2*xs[3]*temp[1];

// gain matrix Kpsi = aP*Hpsi'*(Hpsi*aP*Hpsi' + Rpsi)^-1
mat_mymul4(aP,Hpsi,tmp71,3);
invR = 1/(Hpsi[0][0]*tmp71[0][0]+Hpsi[0][1]*tmp71[1][0]+...
Hpsi[0][2]*tmp71[2][0]+Hpsi[0][3]*tmp71[3][0]+var_psi);

mat_scalMul(tmp71,invR,Kpsi);

tempPsi = atan2(coeff1[0],coeff1[1]);
for(i=0;i<7;i++) {
  Kpsi[i][0] = invR*tmp71[i][0];
  xs[i] += Kpsi[i][0]*wraparound(atan2(-Byc,Bxc) - tempPsi);
}

y4[1] = atan2(-Byc,Bxc);
y4[2] = atan2(coeff1[0],coeff1[1]);

// error covariance matrix update aP = (I - Kpsi*Hpsi)*aP
```

```
          mat_mymul1(Kpsi,Hpsi,mat77,3);
          mat_sub(Iden,mat77,tmpr);
          mat_mymul5(tmpr,aP, tmp77,3);
          mat_copy(tmp77,aP);

    }

//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
//scaling of quertonian,||q||^2 = 1
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
norm = 1.0/sqrt(xs[0]*xs[0]+xs[1]*xs[1]+xs[2]*xs[2]+xs[3]*xs[3]);
for(i=0;i<4;i++) xs[i] = xs[i]*norm;

//obtain euler angles from quaternion
tempThe=asin(-2*(xs[1]*xs[3]-xs[0]*xs[2]));
tempPhi=atan2(2*(xs[0]*xs[1]+xs[2]*xs[3]),1-2*(xs[1]*xs[1]+xs[2]*xs[2]));
tempPsi=atan2(2*(xs[1]*xs[2]+xs[0]*xs[3]),1-2*(xs[2]*xs[2]+xs[3]*xs[3]));

vgCheck = !vgCheck;
ssSetIWorkValue(S, 0, vgCheck);
ssSetIWorkValue(S, 1, count);

ssSetRWorkValue(S, 0, tempPhi);
ssSetRWorkValue(S, 1, tempThe);
ssSetRWorkValue(S, 2, tempPsi);
ssSetPWorkValue(S, 0, aP);


y1[0] = xs[4];
y1[1] = xs[5];
y1[2] = xs[6];


y[0] = tempPhi;
y[1] = tempThe;
y[2] = tempPsi;

// Output Quaternions
y3[0] = xs[0];
y3[1] = xs[1];
y3[2] = xs[2];

y4[0] = xs[3];
```

```
}

/*==============================*
 *            Termination       *
 *==============================*/

static void mdlTerminate(SimStruct *S)
{
    int i;
    MATRIX v;


    for(i = 0; i < ssGetNumPWork(S); i++){
        if(ssGetPWorkValue(S, i) != NULL){
            v = ssGetPWorkValue(S, i);
            mat_free(v);
        }
    }

    for(i = 0; i < ssGetNumIWork(S); i++){
        if(ssGetIWorkValue(S, i) != NULL){
            free((void*)ssGetIWorkValue(S, i));
        }
    }
}

#ifdef  MATLAB_MEX_FILE
#include "simulink.c"
#else
#include "cg_sfun.h"
#endif
```

A.2 AHRS multi-mode

```
#define S_FUNCTION_NAME  NewKalmanFilter
#define S_FUNCTION_LEVEL 2

#include "simstruc.h"
#include "matrix.c"
#include <time.h>
#include <stdio.h>
```

```
#define NIN 4
#define NOUT 7
#define NPAR 0

/*magnetometer hard-iron calibration: */
#define         bBy     0.0
#define         bBx     0.0
#define         sfx      1
#define         sfy      1
/*err covariance of accelerometers: */
#define         var_az  1.962361        //(0.1*g)^2
#define         var_ax  1.962361
#define         var_ay  1.962361

/*=======================================*
 *sensor noise characteristics            *
 *=======================================*/

//predefined variables
#define g     9.81 //m/sec^2
#define g2      19.62    //2*g
#define r2d     57.2958     //radiant to degree
#define d2r     0.01745 //degree to radiant
#define pi      3.141592
#define pi2     6.283184 //pi*2

#define  ANGLE_Update  25  // Angle update in Hz
#define  MAG_Update    10  // Mag. update in Hz
#define  AHRS_Run      50  // Ahrs runs at 50 Hz

//err covariance of gps heading
#define     var_psi   0.001

//sign function
#define         sign(arg) (arg>=0 ? 1:-1)

/*===================================*
 *       Initialization              *
 *===================================*/

static void mdlInitializeSizes(SimStruct *S)
{
```

```
ssSetNumSFcnParams(S, NPAR);
if(ssGetNumSFcnParams(S)!=ssGetSFcnParamsCount(S)) return;

    if(!ssSetNumInputPorts(S, NIN)) return;
    {
        int_T   i;
        for(i = 0; i < NIN; i++){
            ssSetInputPortWidth(S, i, DYNAMICALLY_SIZED);
            ssSetInputPortDirectFeedThrough(S, i, 1);
        }
    }

    if(!ssSetNumOutputPorts(S,NOUT))    return;

    {
        int_T i;
        for(i = 0; i < NOUT; i++)
            ssSetOutputPortWidth(S, i, DYNAMICALLY_SIZED);
     }

    ssSetNumContStates(S, 0);
    ssSetNumDiscStates(S, 7);

    ssSetNumSampleTimes(S, 1);
    ssSetNumRWork(S, 6);
    ssSetNumIWork(S, 2);
    ssSetNumPWork(S, 19);
    ssSetNumModes(S, 0);

}


/*====================================*
 * Initialize Sample Time            *
 *====================================*/

static void mdlInitializeSampleTimes(SimStruct *S)
{
       ssSetSampleTime(S, 0, 0.02);
       ssSetOffsetTime(S, 0, 0.0);

}

#define MDL_INITIALIZE_CONDITIONS
```

```
#if defined(MDL_INITIALIZE_CONDITIONS)

static void mdlInitializeConditions(SimStruct *S)
{
 real_T    *xs = ssGetRealDiscStates(S);
 int_T     numStates  = ssGetNumDiscStates(S);
 int_T     i;

      xs[0] = 1.0;
      for(i = 1; i < 7; i++)
          xs[i] = 0.0;
  }

#endif /* MDL_INITIALIZE_CONDITIONS */


/*================================*
 *   Simulation Start Time        *
 *================================*/

#define MDL_START
#if defined(MDL_START)

static void mdlStart(SimStruct *S)
{
 MATRIX aP,aQ,aR1,aR2,aK1,aK2,Fsys,Hj1,Hj2,Iden;
 MATRIX tmp74,tmp76,tmp44,tmp66,tmp77,tmpr,Rinv1,Rinv2,mat77;

 double tempPhi, tempThe, tempPsi, Hcost0,Hcost1,Hcost2;
 short vgCheck,magCheck,count;

 if(ssIsFirstInitCond(S)){

 //initialization of err, meas, and process cov. matrices
 aP = mat_creat(7,7,ZERO_MATRIX);
 aQ = mat_creat(6,6,ZERO_MATRIX);
 aR1 = mat_creat(4,4,ZERO_MATRIX);
 aR2 = mat_creat(6,6,ZERO_MATRIX);

 aP[0][0]= 1.0e-1;
 aP[1][1]= 1.0e-1;
 aP[2][2]= 1.0e-1;
 aP[3][3]=1.0e-1;
 aP[4][4]= aP[5][5]= aP[6][6]=1.0e-1;
```

```
aQ[0][0] = 1.0e-4;
aQ[1][1] = 1.0e-4;
aQ[2][2] = 1.0e-4;
aQ[3][3] = 0.25e-4;
aQ[4][4] = 0.25e-4;
aQ[5][5] = 1.0e-8;

//matrice R per il modo1
aR1[0][0] = 7.0e-5;
aR1[1][1] = 7.0e-5;
aR1[2][2] = 7.0e-5;
aR1[3][3] = var_psi;

//matrice R per il modo2
aR2[0][0] = 7.0e-5;
aR2[1][1] = 7.0e-5;
aR2[2][2] = 7.0e-5;
aR2[3][3] = var_ax;
aR2[4][4] = var_ax;
aR2[5][5] = var_ax;

//initialization of gain matrix
aK1 = mat_creat(7,4,ZERO_MATRIX);
aK2 = mat_creat(7,6,ZERO_MATRIX);

//initialization of state transition matrix
Fsys = mat_creat(7,7,UNIT_MATRIX);

//initialization of Identity matrix
Iden = mat_creat(7,7,UNIT_MATRIX);

//initialization of Jacobian matrix
Hj1   = mat_creat(4,7,ZERO_MATRIX);
Hj2   = mat_creat(6,7,ZERO_MATRIX);

//initialization of other matrice used in ahrs
Rinv1  = mat_creat(4,4,ZERO_MATRIX);
Rinv2  = mat_creat(6,6,ZERO_MATRIX);
tmp66 = mat_creat(6,6,ZERO_MATRIX);
tmp44 = mat_creat(4,4,ZERO_MATRIX);
tmp74 = mat_creat(7,4,ZERO_MATRIX);
tmp76 = mat_creat(7,6,ZERO_MATRIX);
tmp77 = mat_creat(7,7,ZERO_MATRIX);
```

```
tmpr  = mat_creat(7,7,ZERO_MATRIX);
mat77 = mat_creat(7,7,ZERO_MATRIX);

//campo magnetico costante come modello
Hcost0 = 2.4762e-1; //G
Hcost1 = 824.65e-5; //G
Hcost2 = 3.865e-1; //G

vgCheck=0,magCheck=0; count=0;

tempThe = 0.0;
tempPhi = 0.0;
tempPsi = 0.0;

ssSetPWorkValue(S, 0, aP);
ssSetPWorkValue(S, 1, aQ);
ssSetPWorkValue(S, 2, aR1);
ssSetPWorkValue(S, 3, aK1);
ssSetPWorkValue(S, 4, Fsys);
ssSetPWorkValue(S, 5, Iden);
ssSetPWorkValue(S, 6, Hj1);

ssSetPWorkValue(S, 7, aR2);
ssSetPWorkValue(S, 8, aK2);
ssSetPWorkValue(S, 9, Hj2);

ssSetPWorkValue(S, 10, Rinv1);
ssSetPWorkValue(S, 11, tmp44);
ssSetPWorkValue(S, 12, tmp74);

ssSetPWorkValue(S, 13, tmp77);
ssSetPWorkValue(S, 14, tmpr);
ssSetPWorkValue(S, 15, mat77);

ssSetPWorkValue(S, 16, Rinv2);
ssSetPWorkValue(S, 17, tmp66);
ssSetPWorkValue(S, 18, tmp76);

ssSetIWorkValue(S, 0, vgCheck);
ssSetIWorkValue(S, 1, count);

ssSetRWorkValue(S, 0, tempPhi);
ssSetRWorkValue(S, 1, tempThe);
ssSetRWorkValue(S, 2, tempPsi);
```

```
 ssSetRWorkValue(S, 3, Hcost0);
 ssSetRWorkValue(S, 4, Hcost1);
 ssSetRWorkValue(S, 5, Hcost2);

    }
  }
#endif /*  MDL_START */

//++++++++++++++++++++++++++++++++
// wrap around for -180 and + 180
//++++++++++++++++++++++++++++++++

double wraparound(double dta)
{

   //bound heading angle between -180 and 180
    if(dta >  pi) dta -= pi2;
    else if(dta < -pi) dta += pi2;

   return dta;
}
/*======================*
 *   Output             *
 *======================*/

static void mdlOutputs(SimStruct *S, int_T tid)
{
    // local variables

    static double tnow,tprev=0;
    double pc,qc,rc;
    double h1[4]={0.,},h2[6]={0.,},cPHI,sPHI;
    double xsn[4]={0.,};
    double norm,Bxc,Byc,invR;
    double dt,Hdt;
    double coeff1[3]={0,},temp[2]={0,};
    short  i=0,j=0;

    double Hcost0, Hcost1, Hcost2;

    double tempHx,tempHy,tempHz;
    double tempPsi,tempThe,tempPhi,prova,psi_m;
    double tempAx,tempAy,tempAz;
    double tempP,tempQ,tempR;
```

```
double tempVn,tempVe;
short gps_valid;

real_T    *y, *y1, *y2, *y3, *y4, *y5, *y6;

real_T    *xs= ssGetRealDiscStates(S);
MATRIX aP,aQ,aR1,aR2,aK1,aK2,Fsys,Hj1,Hj2,Iden,Vk,VkaQ,aQnew;
MATRIX tmp76,tmp66,tmp74,tmp44,tmp77,tmpr,Rinv1,Rinv2,mat77;
MATRIX DCM;

short  vgCheck,magCheck,count;

Vk = mat_creat(7,6,ZERO_MATRIX);
VkaQ = mat_creat(7,6,ZERO_MATRIX);
aQnew = mat_creat(7,7,ZERO_MATRIX);

//matrice di rotazione
DCM = mat_creat(3,3,ZERO_MATRIX);

aP = ssGetPWorkValue(S, 0);
aQ = ssGetPWorkValue(S, 1);
aR1 = ssGetPWorkValue(S, 2);
aK1 = ssGetPWorkValue(S, 3);
Fsys = ssGetPWorkValue(S, 4);
Iden = ssGetPWorkValue(S, 5);
Hj1 = ssGetPWorkValue(S, 6);

aR2 = ssGetPWorkValue(S, 7);
aK2 = ssGetPWorkValue(S, 8);
Hj2 = ssGetPWorkValue(S, 9);

Rinv1 = ssGetPWorkValue(S, 10);
tmp44 = ssGetPWorkValue(S, 11);
tmp74 = ssGetPWorkValue(S, 12);

tmp77 = ssGetPWorkValue(S, 13);
tmpr = ssGetPWorkValue(S, 14);
mat77 = ssGetPWorkValue(S, 15);

Rinv2 = ssGetPWorkValue(S, 16);
tmp66 = ssGetPWorkValue(S, 17);
tmp76 = ssGetPWorkValue(S, 18);

vgCheck = ssGetIWorkValue(S, 0);
```

```
    count = ssGetIWorkValue(S, 1);

    tempPhi = ssGetRWorkValue(S, 0);
    tempThe = ssGetRWorkValue(S, 1);
    tempPsi = ssGetRWorkValue(S, 2);
    Hcost0 = ssGetRWorkValue(S, 3);
    Hcost1 = ssGetRWorkValue(S, 4);
    Hcost2 = ssGetRWorkValue(S, 5);

    //leggo gli ingressi
    tempHx = *ssGetInputPortRealSignalPtrs(S, 0)[0];
    tempHy = *ssGetInputPortRealSignalPtrs(S, 0)[1];
    tempHz = *ssGetInputPortRealSignalPtrs(S, 0)[2];
    tempAx = *ssGetInputPortRealSignalPtrs(S, 1)[0];
    tempAy = *ssGetInputPortRealSignalPtrs(S, 1)[1];
    tempAz = *ssGetInputPortRealSignalPtrs(S, 1)[2];
    tempP = *ssGetInputPortRealSignalPtrs(S, 2)[0];
    tempQ = *ssGetInputPortRealSignalPtrs(S, 2)[1];
    tempR = *ssGetInputPortRealSignalPtrs(S, 2)[2];
    tempVn = *ssGetInputPortRealSignalPtrs(S, 3)[0];
    tempVe = *ssGetInputPortRealSignalPtrs(S, 3)[1];
    gps_valid = *ssGetInputPortRealSignalPtrs(S, 3)[2];

    //definisco i vettori per le uscite
    y =  ssGetOutputPortRealSignal(S,0); //angoli eulero
    y1 = ssGetOutputPortRealSignal(S,1); //bias
    y2 = ssGetOutputPortRealSignal(S,2);
    y3 = ssGetOutputPortRealSignal(S,3);
    y4 = ssGetOutputPortRealSignal(S,4);
    y5 = ssGetOutputPortRealSignal(S,5);
    y6 = ssGetOutputPortRealSignal(S,6);

//snap the time interval, dt, of this routine
    dt = 0.02;
    Hdt = 0.5*dt;

    /*assign new variables */
    pc = (tempP - xs[4])*Hdt;
    qc = (tempQ - xs[5])*Hdt;
    rc = (tempR - xs[6])*Hdt;

    /*state transition matrix*/

    Fsys[0][1] = -pc; Fsys[0][2] = -qc; Fsys[0][3] = -rc;
```

```
    Fsys[1][0] =  pc; Fsys[1][2] =  rc; Fsys[1][3] = -qc;
    Fsys[2][0] =  qc; Fsys[2][1] = -rc; Fsys[2][3] =  pc;
    Fsys[3][0] =  rc; Fsys[3][1] =  qc; Fsys[3][2] = -pc;

    Fsys[0][4] = xs[1]*Hdt;  Fsys[0][5] = xs[2]*Hdt;
    Fsys[0][6] = xs[3]*Hdt;
    Fsys[1][4] =-xs[0]*Hdt;  Fsys[1][5] = xs[3]*Hdt;
    Fsys[1][6] =-Fsys[0][5];
    Fsys[2][4] =-Fsys[1][5]; Fsys[2][5] = Fsys[1][4];
    Fsys[2][6] = Fsys[0][4];
    Fsys[3][4] = Fsys[0][5]; Fsys[3][5] =-Fsys[0][4];
    Fsys[3][6] = Fsys[1][4];

    //+++++++++++++++++++++++++++++++++++++++++++++++
    //Extended Kalman filter: prediction step
    //+++++++++++++++++++++++++++++++++++++++++++++++
    /*propagation of quaternion using gyro measurement
      at a given sampling interval dt    */

    xsn[0] = xs[0] - pc*xs[1] - qc*xs[2] - rc*xs[3];
    xsn[1] = xs[1] + pc*xs[0] - qc*xs[3] + rc*xs[2];
    xsn[2] = xs[2] + pc*xs[3] + qc*xs[0] - rc*xs[1];
    xsn[3] = xs[3] - pc*xs[2] + qc*xs[1] + rc*xs[0];

    for(i=0; i<4; i++) xs[i] = xsn[i];

//Normalizzazione
norm = 1.0/sqrt(xs[0]*xs[0]+xs[1]*xs[1]+xs[2]*xs[2]+xs[3]*xs[3]);
for(i=0;i<4;i++) xs[i] = xs[i]*norm;

/* aggiorno matrice Q */
 //determino matrice Vk
 Vk[0][0]=Hdt*xs[1]; Vk[0][1]=Hdt*xs[2]; Vk[0][2]=Hdt*xs[3];
 Vk[1][0]=-Hdt*xs[0]; Vk[1][1]=Hdt*xs[3]; Vk[1][2]=-Hdt*xs[2];
 Vk[2][0]=-Hdt*xs[3]; Vk[2][1]=-Hdt*xs[0]; Vk[2][2]=Hdt*xs[1];
 Vk[3][0]=Hdt*xs[2]; Vk[3][1]=-Hdt*xs[1]; Vk[3][2]=-Hdt*xs[0];
 Vk[4][3]=dt; Vk[5][4] = dt; Vk[6][5] = dt;
    //calcolo la nuova Q => aQnew= Vk*aQ*Vk'
 mat_mul(Vk,aQ,VkaQ);
 mat_mulT(VkaQ,Vk,aQnew);

    //error covriance propagation: P = Fsys*P*Fsys' + Q
    mat_mymul2(Fsys,aP,tmp77,3);
    mat_mymul3(tmp77,Fsys,aP,3);
```

```
for(i=0;i<7;i++)
    aP[i][i] += aQnew[i][i];

count++;

// Pitch and Roll Update at 25 Hz
if(count%(AHRS_Run/ANGLE_Update)==0){

//definisco la matrice di rotazione DCM nei quaternioni
 DCM[0][0]=xs[0]*xs[0]+xs[1]*xs[1]-xs[2]*xs[2]-xs[3]*xs[3];
 DCM[0][1]=2*(xs[1]*xs[2]+xs[0]*xs[3]);
 DCM[0][2]=2*(xs[1]*xs[3]-xs[0]*xs[2]);
 DCM[1][0]=2*(xs[1]*xs[2]-xs[0]*xs[3]);
 DCM[1][1]=xs[0]*xs[0]-xs[1]*xs[1]+xs[2]*xs[2]-xs[3]*xs[3];
 DCM[1][2]=2*(xs[2]*xs[3]+xs[0]*xs[1]);
 DCM[2][0]=2*(xs[1]*xs[3]+xs[0]*xs[2]);
 DCM[2][1]=2*(xs[2]*xs[3]-xs[0]*xs[1]);
 DCM[2][2]=xs[0]*xs[0]-xs[1]*xs[1]-xs[2]*xs[2]+xs[3]*xs[3];
```

A questo punto occorre verificare la presenza di un segnale GPS valido. Il sensore GPS è implementato in modo da inviare un pacchetto di dati composto da diversi campi. In questi campi, oltre all'informazione relativa alla misurazione, è presente anche un flag che indica, eventualmente, la presenza di errori invalidanti la misura effettuata [9]. Nella porzione di codice successiva tale flag è stato chiamato gps_valid.

```
if(gps_valid==1){
/////////////// MODE 1 ////////////////////////////

//coeff per psi (h1[4])
coeff1[0]= 2*(xs[1]*xs[2]+ xs[0]*xs[3]); //alfa
coeff1[1]= 1 - 2*(xs[2]*xs[2]+xs[3]*xs[3]); //beta
coeff1[2]= 2/(coeff1[0]*coeff1[0]+coeff1[1]*coeff1[1]);

temp[0] = coeff1[1]*coeff1[2];
temp[1] = coeff1[0]*coeff1[2]*coeff1[1];

//campo magnetico in bodyframe (DCM*Hcost)
h1[0] = DCM[0][0]*Hcost0+DCM[0][1]*Hcost1+DCM[0][2]*Hcost2;
h1[1] = DCM[1][0]*Hcost0+DCM[1][1]*Hcost1+DCM[1][2]*Hcost2;
h1[2] = DCM[2][0]*Hcost0+DCM[2][1]*Hcost1+DCM[2][2]*Hcost2;
//modello per psi
```

```
    h1[3] = atan2(coeff1[0],coeff1[1]);

    y2[0]=h1[0];
    y2[1]=h1[1];
    y2[2]=h1[2];
    y4[1]=h1[3];

//Jacobian matrix of h(x) - campo magnetico
Hj1[0][0] = 2*(xs[0]*Hcost0 + xs[3]*Hcost1 - xs[2]*Hcost2);
Hj1[0][1] = 2*(xs[1]*Hcost0 + xs[2]*Hcost1 + xs[3]*Hcost2);
Hj1[0][2] = 2*(-xs[2]*Hcost0 + xs[1]*Hcost1 - xs[0]*Hcost2);
Hj1[0][3] = 2*(-xs[3]*Hcost0 + xs[0]*Hcost1 + xs[1]*Hcost2);
Hj1[1][0] = 2*(-xs[3]*Hcost0 + xs[0]*Hcost1 + xs[1]*Hcost2);
Hj1[1][1] = 2*(xs[2]*Hcost0 - xs[1]*Hcost1 + xs[0]*Hcost2);
Hj1[1][2] = 2*(xs[1]*Hcost0 + xs[2]*Hcost1 + xs[3]*Hcost2);
Hj1[1][3] = 2*(-xs[0]*Hcost0 - xs[3]*Hcost1 + xs[2]*Hcost2);
Hj1[2][0] = 2*(xs[2]*Hcost0 - xs[1]*Hcost1 + xs[0]*Hcost2);
Hj1[2][1] = 2*(xs[3]*Hcost0 - xs[0]*Hcost1 - xs[1]*Hcost2);
Hj1[2][2] = 2*(xs[0]*Hcost0 + xs[3]*Hcost1 - xs[2]*Hcost2);
Hj1[2][3] = 2*(xs[1]*Hcost0 + xs[2]*Hcost1 + xs[3]*Hcost2);
// Jacobiano psi GPS
Hj1[3][0] = xs[3]*temp[0];
Hj1[3][1] = xs[2]*temp[0];
Hj1[3][2] = xs[1]*temp[0]+2*xs[2]*temp[1];
Hj1[3][3] = xs[0]*temp[0]+2*xs[3]*temp[1];

     // gain matrix aK = aP*Hj'*(Hj*aP*Hj' + aR)^-1
      mat_mymul4(aP,Hj1,tmp74,3);
      mat_mymul(Hj1,tmp74,tmp44,3);

      for(i=0;i<4;i++) tmp44[i][i] += aR1[i][i];
      mat_inv(tmp44,Rinv1);
      mat_mul(tmp74,Rinv1,aK1);

      // misura per psi
      psi_m = atan2(tempVe,tempVn);

      //state update
      for(i=0;i<7;i++)
      {
   xs[i] += aK1[i][0]*(tempHx - h1[0])
         +  aK1[i][1]*(tempHy - h1[1])
         +  aK1[i][2]*(tempHz - h1[2])
             +  aK1[i][3]*wraparound(psi_m - h1[3]);
```

```
      }

//Normalizzazione
norm = 1.0/sqrt(xs[0]*xs[0]+xs[1]*xs[1]+xs[2]*xs[2]+xs[3]*xs[3]);
for(i=0;i<4;i++) xs[i] = xs[i]*norm;

//error covariance matrix update aP = (I - aK*Hj)*aP
mat_mymul1(aK1,Hj1,mat77,3);
mat_sub(Iden,mat77,tmpr);
mat_mymul5(tmpr,aP,tmp77,3);
mat_copy(tmp77,aP);
    }
    else{ // no gps
        /////////////// MODE 2 ////////////

    //campo magnetico in bodyframe (DCM*Hcost) + ACC
    h2[0] = DCM[0][0]*Hcost0+DCM[0][1]*Hcost1+DCM[0][2]*Hcost2;
    h2[1] = DCM[1][0]*Hcost0+DCM[1][1]*Hcost1+DCM[1][2]*Hcost2;
    h2[2] = DCM[2][0]*Hcost0+DCM[2][1]*Hcost1+DCM[2][2]*Hcost2;

     //modello con le accelerazioni
     h2[3] = -g2*(xs[1]*xs[3]-xs[0]*xs[2]);
     h2[4] = -g2*(xs[0]*xs[1]+xs[2]*xs[3]);
     h2[5] = -g*(xs[0]*xs[0]-xs[1]*xs[1]-xs[2]*xs[2]+xs[3]*xs[3]);

     //uscite
     y2[0] = h2[0];
     y2[1] = h2[1];
     y2[2] = h2[2];
     y5[0] = h2[3];
     y5[1] = h2[4];
     y5[2] = h2[5];

// Jacobian matrix of h(x) - campo magnetico
Hj2[0][0] = 2*(xs[0]*Hcost0 + xs[3]*Hcost1 - xs[2]*Hcost2);
Hj2[0][1] = 2*(xs[1]*Hcost0 + xs[2]*Hcost1 + xs[3]*Hcost2);
Hj2[0][2] = 2*(-xs[2]*Hcost0 + xs[1]*Hcost1 - xs[0]*Hcost2);
Hj2[0][3] = 2*(-xs[3]*Hcost0 + xs[0]*Hcost1 + xs[1]*Hcost2);
Hj2[1][0] = 2*(-xs[3]*Hcost0 + xs[0]*Hcost1 + xs[1]*Hcost2);
Hj2[1][1] = 2*(xs[2]*Hcost0 - xs[1]*Hcost1 + xs[0]*Hcost2);
Hj2[1][2] = 2*(xs[1]*Hcost0 + xs[2]*Hcost1 + xs[3]*Hcost2);
Hj2[1][3] = 2*(-xs[0]*Hcost0 - xs[3]*Hcost1 + xs[2]*Hcost2);
Hj2[2][0] = 2*(xs[2]*Hcost0 - xs[1]*Hcost1 + xs[0]*Hcost2);
Hj2[2][1] = 2*(xs[3]*Hcost0 - xs[0]*Hcost1 - xs[1]*Hcost2);
```

```
Hj2[2][2] = 2*(xs[0]*Hcost0 + xs[3]*Hcost1 - xs[2]*Hcost2);
Hj2[2][3] = 2*(xs[1]*Hcost0 + xs[2]*Hcost1 + xs[3]*Hcost2);
//jacobiano accelerazioni
Hj2[3][0] = g2*xs[2]; Hj2[3][1] =-g2*xs[3];
Hj2[3][2] = g2*xs[0]; Hj2[3][3] = -g2*xs[1];
Hj2[4][0] = Hj2[3][3]; Hj2[4][1] =-Hj2[3][2];
Hj2[4][2] = Hj2[3][1]; Hj2[4][3] = -Hj2[3][0];
Hj2[5][0] =-Hj2[3][2]; Hj2[5][1] =-Hj2[3][3];
Hj2[5][2] = Hj2[3][0]; Hj2[5][3] =  Hj2[3][1];

      // gain matrix aK = aP*Hj'*(Hj*aP*Hj' + aR)^-1
      mat_mymul4(aP,Hj2,tmp76,3);
      mat_mymul(Hj2,tmp76,tmp66,3);

      for(i=0;i<6;i++) tmp66[i][i] += aR2[i][i];
      mat_inv(tmp66,Rinv2);
      mat_mul(tmp76,Rinv2,aK2);

      //state update
      for(i=0;i<7;i++)
      {
   xs[i] += aK2[i][0]*(tempHx - h2[0])
         +  aK2[i][1]*(tempHy - h2[1])
         +  aK2[i][2]*(tempHz - h2[2])
             +  aK2[i][3]*(tempAx - h2[3])
             +  aK2[i][4]*(tempAy - h2[4])
             +  aK2[i][5]*(tempAz - h2[5]);
      }

//Normalizzazione
norm = 1.0/sqrt(xs[0]*xs[0]+xs[1]*xs[1]+xs[2]*xs[2]+xs[3]*xs[3]);
for(i=0;i<4;i++) xs[i] = xs[i]*norm;

//error covariance matrix update aP = (I - aK*Hj)*aP
mat_mymul1(aK2,Hj2,mat77,3);
mat_sub(Iden,mat77,tmpr);
mat_mymul5(tmpr,aP,tmp77,3);
mat_copy(tmp77,aP);

    }
}

//obtain euler angles from quaternion
tempThe = asin(-2*(xs[1]*xs[3]-xs[0]*xs[2]));
```

```
tempPhi = atan2(2*(xs[0]*xs[1]+xs[2]*xs[3]),1-2*(xs[1]*xs[1]+xs[2]*xs[2]));
tempPsi = atan2(2*(xs[1]*xs[2]+xs[0]*xs[3]),1-2*(xs[2]*xs[2]+xs[3]*xs[3]));

   vgCheck = !vgCheck;
   ssSetIWorkValue(S, 0, vgCheck);
   ssSetIWorkValue(S, 1, count);

   ssSetRWorkValue(S, 0, tempPhi);
   ssSetRWorkValue(S, 1, tempThe);
   ssSetRWorkValue(S, 2, tempPsi);
   ssSetPWorkValue(S, 0, aP);


   y1[0] = xs[4];
   y1[1] = xs[5];
   y1[2] = xs[6];


   y[0] = tempPhi;
   y[1] = tempThe;
   y[2] = tempPsi;

   // Output Quaternions
    y3[0] = xs[0];
    y3[1] = xs[1];
    y3[2] = xs[2];

    y4[0] = xs[3];
    y4[2] = gps_valid;

    y6[0] = aQnew[3][3];
    y6[1] = aQnew[4][4];
    y6[2] = aQnew[5][5];
}

/*====================================*
 *          Termination               *
 *====================================*/

static void mdlTerminate(SimStruct *S)
{
    int i;
    MATRIX v;
```

```
    for(i = 0; i < ssGetNumPWork(S); i++){
        if(ssGetPWorkValue(S, i) != NULL){
            v = ssGetPWorkValue(S, i);
            mat_free(v);
        }
    }

    for(i = 0; i < ssGetNumIWork(S); i++){
        if(ssGetIWorkValue(S, i) != NULL){
            free((void*)ssGetIWorkValue(S, i));
        }
    }
}

#ifdef  MATLAB_MEX_FILE
#include "simulink.c"
#else
#include "cg_sfun.h"
#endif
```

Appendice B

Sensori inerziali

Una *Unità di Misura Inerziale* è composta da una terna ortogonale di giroscopi e accelerometri, in modo da consentire il monitoraggio degli stati della navigazione. Altri sensori presenti sono: il barometro, per le misure di altitudine rispetto al livello medio del mare; tubo di Pitot, per le misure relative all'airspeed; sensori triassiali di temperatura e magnetometri.

B.1 Errori di misura

Trattandosi di sensori di natura elettro-meccanica le misure effettuate risultano affette da due tipi di errore:

- *errori sistematici*: sono legati ad errori di modello ed hanno la caratteristica di conservare valore e segno. Rientrano in questa categoria:
  - *bias*: sono errori indipendenti dalle misure effettuate, non realmente costanti ma che variano ad ogni accensione del dispositivo e dipendono dalla temperatura. E' però ragionevole supporre che durante una osservazione continua nel tempo, le loro variazioni siano talmente lente da potersi ritenere costanti, e che possano essere determinati in fase di analisi dei dati.
  - *fattori di scala*: è il rapporto tra la variabile misurata ed il valore effettivo di tale variabile, e come tale si vuole che sia il più possibile prossimo all'unità.

  In genere è necessaria una procedura di calibrazione per modellare e compensare gli errori sistematici.

- *errori accidentali*: sono errori non prevedibili a priori e dunque aleatori, legati alla presenza di *rumori di misura*, di natura interna, o *disturbi*, di natura esterna. Le incertezze di misura che ne scaturiscono possono essere ridotte, ma non eliminate completamente attraverso algoritmi di filtraggio del segnale o filtri passa-basso.

B.2 Giroscopi

Le velocità angolari attorno agli assi X_b, Y_b e Z_b del sistema assi corpo, sono misurate attraverso dei giroscopi. In genere questi strumenti di misura sono molto accurati alle basse frequenze, ma non sono molto buoni a frequenze più elevate cosicchè un giroscopio può essere modellato come un filtro passa-basso:

$$H_{gyro}(s) = \frac{1}{s + \omega_{br}}$$

dove ω_{br} rappresenta la frequenza di taglio del giroscopio, sotto la quale le sue prestazioni cominciano a decadere, ed è in genere un valore abbastanza alto.

Il giroscopio può anche essere semplicemente modellato come $H_{gyro}(s) = 1$, poichè la sua frequenza di taglio è tipicamente più grande rispetto alla più alta frequenza di taglio delle funzioni di trasferimento che modellano il legame tra gli angoli di deflessione alle superfici di controllo e le velocità angolari che descrivono il velivolo.

Le velocità angolari misurate sono modellate come:

$$\omega_{x,y,z}^{mean} = \omega_{SF}\omega_{x,y,z} + \omega_{bias} + noise$$

dove per semplicità si suppone che i fattori di scala siano unitari, per cui ω_{SF} è una matrice identità, mentre i bias sono stimati dall'algoritmo di *Attitude Heading Reference System*.

Le misure dei sensori sono, inoltre, limitate nel range ± 150 deg/s mediante una saturazione.

B.3 Accelerometri

Le accelerazioni lineari lungo agli assi X_b, Y_b e Z_b del sistema assi corpo, sono misurate attraverso tre accelerometri. Analogamente ai giroscopi, anche questi sono modellati come sensori ideali, tale per cui $H_{accels}(s) = 1$.

Le velocità angolari misurate sono modellate come:

$$A_{x,y,z}^{mean} = A_{SF}A_{x,y,z} + A_{bias} + noise$$

dove per semplicità si suppone che i fattori di scala siano unitari, per cui A_{SF} è una matrice identità, come nel caso dei giroscopi, mentre i bias sono stimati dall'algoritmo di *Inertial Navigation System*.

Le misure dei sensori sono, inoltre, limitate nel range ± 2 **g** mediante una saturazione (1 **g** = 9.80 m).