

POLITECNICO DI TORINO



Laurea Magistrale
Ingegneria Aerospaziale

Simulazione del Volo

Tutorial del simulatore di volo di un UAV con autopilota

Prof.ssa Elisa Capello

RELAZIONE DI PIERLUIGI VERGARI

TEAM 3



Indice

1	Introduzione	2
2	Struttura del simulatore	2
2.1	File di testo	3
2.2	Fasi di simulazione	4
3	Struttura del codice	4
3.1	Il main	4
3.2	Le librerie	5
3.3	Le funzioni	6
4	Modelli utilizzati per la simulazione	9
4.1	Autopilota	9
4.2	Il modello matematico	11
5	Simulazione	13
5.1	Pre-processing	13
5.2	Processing	14
5.3	Messaggi di errore e warning	17
6	Mantenimento dello stato di trim	20
7	Esecuzione di un percorso	22

1 Introduzione

Il drone MH850 è un innovativo sistema aereo senza pilota progettato per una vasta gamma di applicazioni. Caratterizzato da una fusoliera compatta e leggera, questo drone offre una combinazione di eccellente manovrabilità, stabilità e capacità di carico utile. Esso è dotato di un sistema di propulsione elettrica avanzato che contribuisce alla sua elevata autonomia, consentendo missioni di lunga durata.

Il documento proposto fornisce un tutorial intuitivo per il simulatore di volo dell'MH850. È suddiviso in diverse sezioni che descrivono il funzionamento del drone a vari livelli di dettaglio. Si inizia con una panoramica della sua struttura, includendo le diverse fasi di funzionamento e fornendo dettagli sui file di testo utilizzati durante la simulazione. Successivamente, si offre una descrizione approfondita del codice e delle varie funzioni coinvolte in ogni step della simulazione. Vengono fornite informazioni dettagliate anche sui modelli matematici utilizzati, tra cui la logica di autopilota.

Nella sezione dedicata alla simulazione, vengono spiegati tutti i vari step accompagnati dalle relative formulazioni matematiche. Inoltre, viene posta particolare attenzione ai possibili messaggi di errore e warning che potrebbero essere generati durante l'esecuzione. Infine, vengono presentati due esempi pratici di utilizzo del simulatore, compresa la simulazione del **mantenimento del trim** con condizioni iniziali specificate dall'utente e la simulazione di un **percorso** selezionato tra le opzioni disponibili nel progetto.

2 Struttura del simulatore

Il progetto è stato sviluppato interamente utilizzando l'IDE (Integrated Development Environment) di CLion ed è stato organizzato in maniera strutturata in un insieme di file e cartelle. La cartella principale è denominata *Simulatore* e al suo interno sono contenuti gli elementi principali del progetto tra cui:

- *cmake-build-debug*: al suo interno è contenuto un file eseguibile (*FlightSimulator.exe*) che è proprio il simulatore di volo, assieme a diversi file di testo e sottocartelle da cui vengono prelevati dati o sui quali vengono scritti. È inoltre presente il file MATLAB *Plot.m* necessario per effettuare il post-processing e l'analisi dei risultati. Tra le sopracitate sottocartelle ci sono:
 - *Drone*: contenente i file di testo che vengono letti e dai quali vengono estrapolate delle grandezze di interesse utili alla simulazione, e una cartella *PERCORSI* nella quale ci sono diversi file di testo, ognuno dei quali contenente la time history dei riferimenti dell'angolo ψ di heading del drone necessari a seguire determinati percorsi.
 - *Validation*: contenente i file di testo usati per la validazione, descritti più nel dettaglio nella sezione successiva.
- *Functions*: contiene delle sottocartelle con tutte le funzioni e routine che vengono richiamate nel *Main.c* o nelle funzioni stesse. In ogni sottocartella sono presenti sia file.c che file.h. Quest'ultima tipologia è richiamata ogni volta che vengono usate le funzioni contenute al suo interno, attraverso il comando `#include "file.h"`.

- *Main.c*: è la funzione che rappresenta l'ingresso principale del programma, da cui parte l'esecuzione in modo sequenziale seguendo l'ordine delle funzioni e delle routine richiamate. In esso è contenuto il flusso del programma che verrà approfondito successivamente.
- *CMakeLists.txt*: è un file di testo che definisce come il progetto deve essere compilato e costruito in modo da essere eseguibile.

2.1 File di testo

I file di testo sono suddivisi in tre categorie:

- File in **lettura** nella cartella cmake-build-debug/Drone
 - *dba.txt*: diversi database che variano con la quota nei quali sono contenute le principali caratteristiche fisiche e geometriche del drone, oltre che le derivate aerodinamiche per angoli di incidenza $\alpha \in [-5^\circ, 16.8^\circ]$;
 - *engine.ini*: contiene le caratteristiche del sistema propulsivo composto da un motore elettrico, come massimo e minimo numero di giri e tensione di alimentazione;
 - *propeller.ini*: contiene le caratteristiche geometriche dell'elica e quelle aerodinamiche per le 100 stazioni in cui è stato suddiviso il raggio;
 - *battery.ini*: contiene le caratteristiche delle batterie, come capacità nominale e rendimento di scarica;
 - *percorsi.txt*: come descritto precedentemente, sono dei file di testo che contengono i vari ψ di riferimento utilizzati per seguire determinati percorsi, le quali forme rispecchiano il nome del file stesso.
- File in **scrittura per post-processing** nella cartella cmake-build-debug
 - *Integrazione.txt*: contiene i valori di tutte le variabili di stato ad ogni step di integrazione;
 - *Control_vector.txt*: contiene i valori che assumono le grandezze del vettore di controllo ad ogni step di integrazione, tra cui equilibratore, alettone e manetta.
- File in **scrittura per verifica** nella cartella cmake-build-debug/Validation
 - *dba.ini*: verifica la corretta esecuzione delle funzioni di lettura per il file *dba_quota[i].txt* se la quota scelta dall'utente è esattamente quella per il quale è stato scritto il file *dba*, oppure verifica la corretta esecuzione della funzione di interpolazione dei due file *dba_quota[i].txt* e *dba_quota[i+1].txt*, se la quota scelta è compresa tra *quota[i]* e *quota[i+1]*;
 - *engine.ini*: verifica la corretta esecuzione delle funzioni di lettura per il file *engine.ini*;
 - *propeller.ini*: verifica la corretta esecuzione delle funzioni di lettura per il file *propeller.ini*;

- *battery.ini*: verifica la corretta esecuzione delle funzioni di lettura per il file *battery.ini*;
- *Interpolated.coefficients.txt*: verifica la corretta esecuzione della funzione di interpolazione dei coefficienti aerodinamici dato un certo angolo di incidenza α , definita in *Interpolazione.h*.

2.2 Fasi di simulazione

La simulazione si articola nelle seguenti 3 fasi principali:

1. **Pre-processing**: a seconda che si voglia simulare il mantenimento del trim o un percorso predefinito, vengono inseriti dall'utente o settati automaticamente i tre parametri iniziali che sono la velocità di volo, la quota e l'angolo di rampa. Partendo da questi dati, tramite la funzione **atmosphere** vengono calcolati i valori di densità, pressione e temperatura di esercizio ai quali verrà fatta la simulazione. Dai file di testo si estrapolano le caratteristiche legate alla geometria del velivolo e al propulsore elettrico, oltre che ai vari coefficienti aerodinamici, i quali dipendono dalla quota di volo.
2. **Processing**: comprende la simulazione vera e propria partendo dal calcolo delle **condizioni di trim** e della **stabilità** statica e dinamica del velivolo. Dopodiché vengono ricavate le grandezze relative ai modi longitudinali, in particolare fugoide e corto periodo. Infine viene eseguita l'integrazione delle equazioni del moto partendo dalle condizioni iniziali di trim, in seguito alla scelta del percorso o alla scelta del tempo di integrazione in caso di simulazione del mantenimento del trim.
3. **Post-processing**: durante la simulazione i risultati vengono salvati in degli appositi file di testo. Questi vengono poi importati in Matlab dove vengono creati i vari grafici che mostrano gli andamenti nel tempo delle varie grandezze fisiche di interesse.

Di seguito, nella Figura 1, viene illustrato uno schema rappresentativo della struttura funzionale del simulatore.

3 Struttura del codice

3.1 Il main

Il file "*Main.c*" è il punto di ingresso del programma C e contiene la funzione principale "*main()*", che rappresenta il punto di partenza dell'esecuzione. È la prima funzione alla quale viene passato il controllo e da cui vengono richiamate altre funzioni per scandire il flusso logico del programma. Il contenuto del *main*, nel caso del simulatore in questione, può essere schematizzato in quattro sezioni:

1. Inclusione delle librerie necessarie e dei file *header*;
2. Dichiarazione delle variabili e selezione automatizzata della working directory;
3. Flusso strutturato in cui vengono richiamati i processi implementati nei file *source*.
4. Indicazione di corretto svolgimento della simulazione.

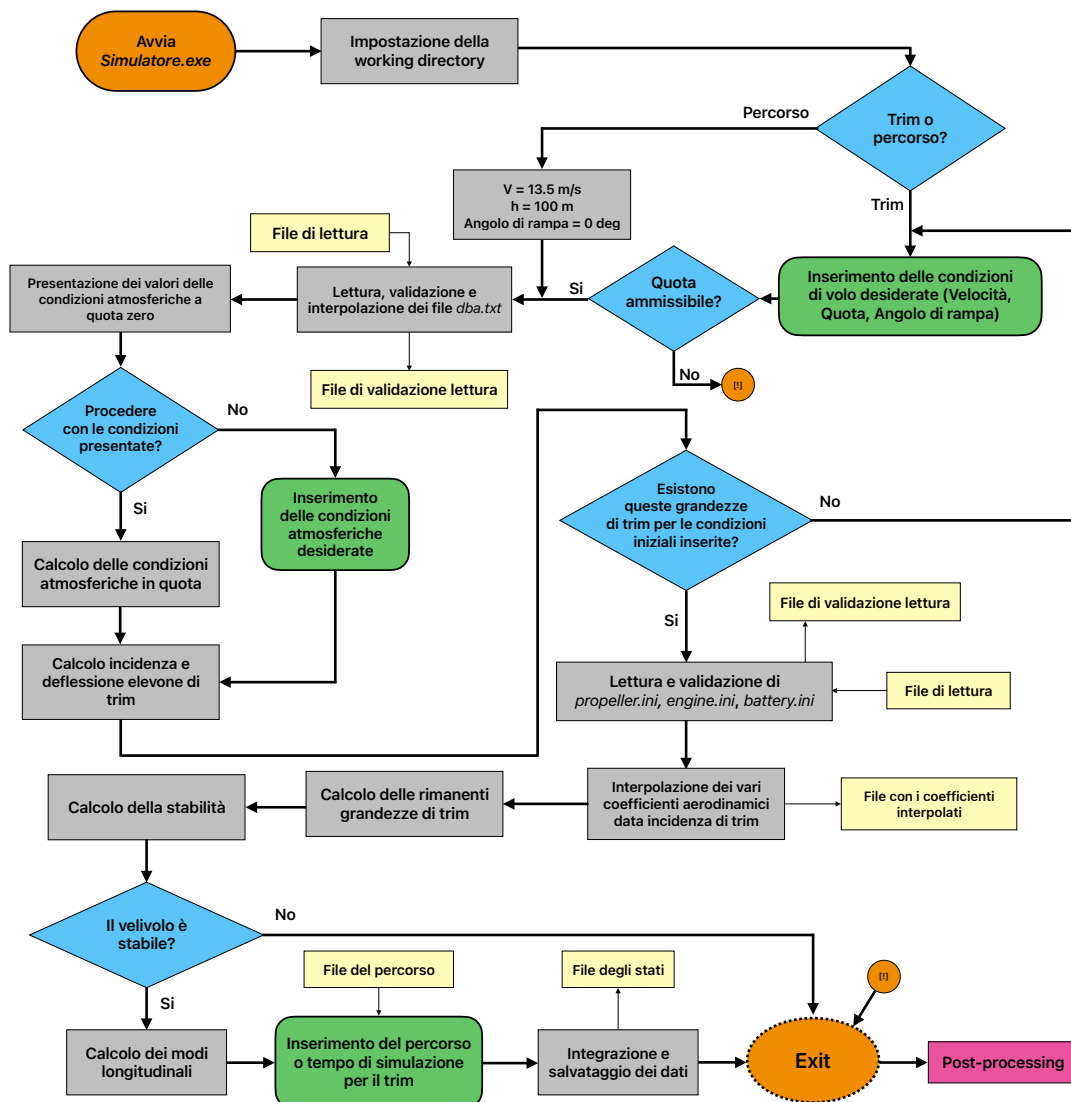


Figura 1: Struttura funzionale del simulatore

3.2 Le librerie

Le librerie standard di C utilizzate sono:

- *stdio.h*: per le funzioni base di C, ad esempio prendere in input o stampare variabili e per interagire con i file;
- *stdlib.h*: per operazioni che coinvolgono la gestione della memoria, la gestione degli errori e la manipolazione dei tipi di dati;
- *unistd.h*: per l'interazione con l'ambiente di esecuzione, come l'accesso ai file, l'esecuzione di processi e altre operazioni di sistema;
- *string.h*: per manipolare stringhe di caratteri;
- *math.h*: per svolgere operazioni numeriche complesse tramite funzioni matematiche comuni;
- *ctype.h*: per il controllo e la manipolazione dei caratteri.

3.3 Le funzioni

Per lo sviluppo del simulatore in maniera modulare ed articolata è stato necessario sviluppare delle funzioni che sono contenute nella cartella " *Functions* ", presente a sua volta nel folder principale del Simulatore. A tal proposito, di seguito sono elencati i nomi dei file *source* (nominati allo stesso modo dei file *header*) e le descrizioni delle varie funzioni in essi contenute.

- **ManageFile.c**

- *WorkingDirectory*: esecuzione automatica del set dei percorsi.
- *ReadAndSaveStatic*: lettura dei file relativi a motore e batteria, validazione e salvataggio dei valori in delle variabili inizializzate nel main e trattate usando i loro puntatori.
- *C_h_interp*: funzione richiamata da *DbRead* contenente la formula di interpolazione.
- *DbRead*: lettura ed interpolazione (se necessaria) dei database aerodinamici variabili in funzione della quota, validazione dei coefficienti letti ed interpolati. Salvataggio dei coefficienti interpolati in delle matrici definite nel main e aggiornante nella funzione.
- *ReadAndSavePropeller*: lettura del file relativo al propeller, validazione e salvataggio dei valori letti in vettori definiti nel main e maneggiati nella funzione tramite i loro puntatori.

- **Atmosfera.c**

- *loadCI*: richiesta, acquisizione da tastiera e controllo delle condizioni iniziali (quota, velocità, angolo di rampa).
- *AtmosphereChoice*: visualizzazione a schermo dei dati atmosferici iniziali con possibilità di mantenere i valori di riferimento fissati al sea level, di modificarli o di immettere manualmente i dati atmosferici alla specifica quota di volo scelta dall'utente.
- *AtmosphereCalc*: calcolo dei dati atmosferici alla quota di interesse, in base alla scelta precedente dell'utente.

- **Alpha_trim.c**

- *calcolo_alpha_trim*: calcolo dell'angolo di incidenza e dell'angolo di deflessione dell'equilibratore di trim in volo longitudinale. Verifica dell'esistenza di questi due angoli per le condizioni iniziali scelte. Verifica che la velocità inserita dall'utente nelle condizioni iniziali sia maggiore di quella minima ammissibile, a sua volta calcolata in questa funzione.

- **Interpolazione.c**

- *interp_alpha*: interpolazione lineare di due coefficienti (aerodinamici) consecutivi in funzione dell'incidenza desiderata.
- *interpola_coeffs*: interpolazione di tutti i coefficienti aerodinamici e salvataggio dei valori interpolati in un file di testo.

- *MarixCoeff*: allocazione in memoria di una matrice contenente tutti i coefficienti interpolati, tramite la lettura del suddetto file di testo.
- **Condizioni_trim.c**
 - *Condizioni_trim*: calcolo e visualizzazione a schermo del vettore di stato di trim, del vettore di controllo di trim, della spinta e RPM di trim.
- **Calcolo_spinta_RPM.c**
 - *propel_trim*: calcolo degli RPM di trim data la spinta di trim, velocità di volo e densità in quota. Questa funzione viene utilizzata solo una volta nel calcolo del trim.
 - *propel*: calcolo della spinta dati in ingresso gli RPM, la velocità e la densità in quota. Questa funzione viene utilizzata nell'integrazione, quando si necessita del valore della spinta ad ogni istante di tempo.
- **Routh.c**
 - *Routh*: verifica della stabilità statica e dinamica del velivolo attraverso il criterio di Routh.
- **Modi.c**
 - *long_modi*: calcolo delle caratteristiche fondamentali dei modi longitudinali di fugoide e corto periodo, tra cui velocità angolare, smorzamento, periodo e tempo di dimezzamento.
- **Integrazione.c**
 - *integration*: acquisizione da tastiera del tempo di simulazione solo in caso di mantenimento del trim, calcolo delle forze e dei momenti agenti sull'UAV, integrazione delle equazioni del moto, scrittura su file dei risultati in termini di variabili di stato e variabili di controllo nel tempo.
 - *lat_dir_controller*: implementazione del controllore per la dinamica latero-direzionale composto da due PID in cascata che agiscono in modo da comandare l'angolo di deflessione del contributo alettone $\Delta\delta_a$. In ingresso a questa funzione viene dato il vettore di ψ_{ref} (angolo di heading comandato) fornito dall'algoritmo di guida, in modo che vengano eseguiti i percorsi.
 - *long_controller*: implementazione del controllore per la dinamica longitudinale composto da tre PID che agiscono in modo da comandare l'angolo di deflessione del contributo equilibratore $\Delta\delta_e$ e la manetta $\Delta\delta_{th}$.
 - *PID*: generazione del comando del controllore in funzione dell'errore allo step attuale e allo step precedente, dato dalla somma dei termini proporzionale, integrativo e derivativo. Per evitare di avere dei problemi dati dalle saturazioni è implementata anche una logica anti-wind up all'interno della funzione. Si considera anche la presenza di un filtro passa-basso per il derivativo.

- *saturation*: saturazione di una generica grandezza, come ad esempio l'angolo dell'equilibratore, l'angolo dell'alettone o la manetta. Questa funzione aggiorna anche dei flag utili nella simulazione.
- *ReadPsi*: scelta del percorso, lettura del file di testo corrispondente, e assegnazione di alcune grandezze come tempo di simulazione (dipendente dal file di testo) e coordinate del waypoint iniziale.

Alcune delle funzioni sopra elencate vengono richiamate e condivise tra più funzioni. Nello specifico si sta facendo riferimento a quelle contenute nei seguenti header:

- *Calcolo_spinta_RPM.h*
- *Interpolazione.h*

Inoltre, alcune delle funzioni includono controlli interni sui dati, necessari a determinare il corretto funzionamento della simulazione. Altri controlli vengono fatti per verificare la corretta lettura e gestione dei file di testo. Infine, il post-processing, elaborato dal file MATLAB *Plot.m*, genera i plot rilevanti dei vari dati di output ottenuti dalla simulazione e che sono estratti dai file di testo denominati *Integrazione.txt* e *Control_vector.txt*.

Nella Figura 2 viene rappresentato uno schema dettagliato su come tutte le varie funzioni ed i file di testo vengono richiamati nei vari step di simulazione.

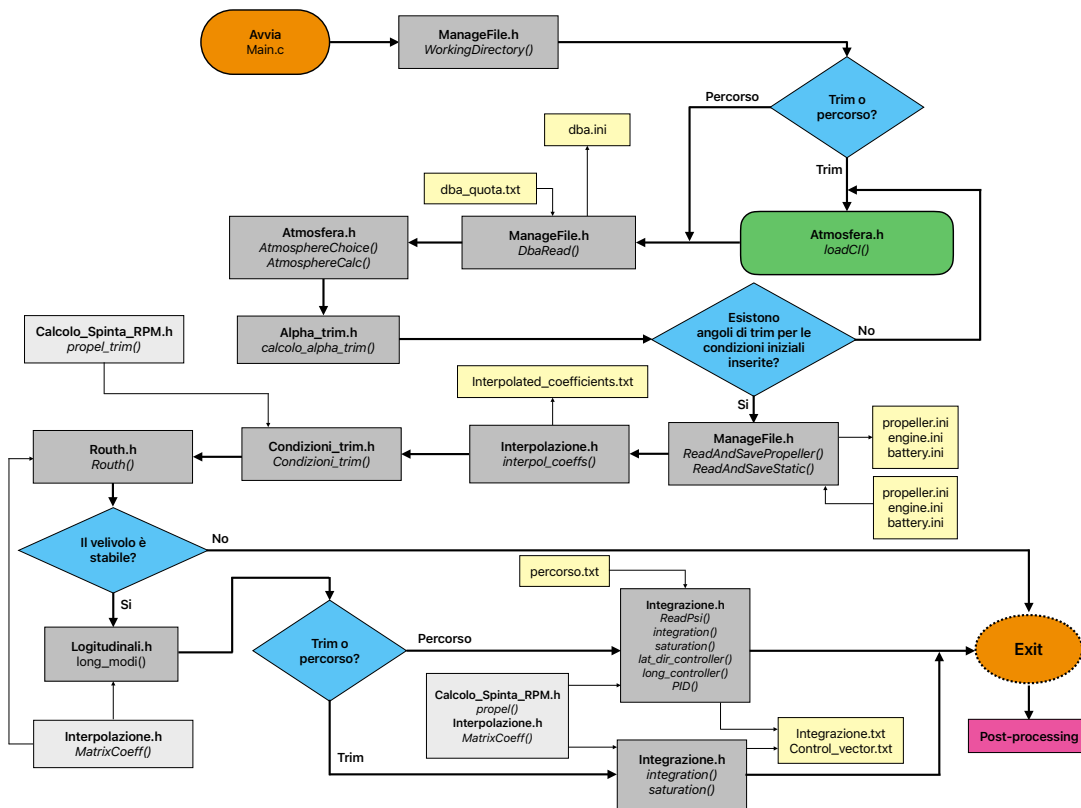


Figura 2: Schema del simulatore

4 Modelli utilizzati per la simulazione

4.1 Autopilota

L'autopilota è un sistema progettato per consentire il volo autonomo del velivolo senza pilota e viene gestito, nel caso del drone simulato, da un controllore PID. Tra i vari sistemi di controllo, il controllore PID si distingue come il più comune e semplice. L'autopilota quindi è designato appositamente per consentire al drone di inseguire una traiettoria specifica, prendendo in input dei riferimenti forniti dall'esterno, tra cui l'angolo di heading ψ_{ref} , calcolato mediante un apposito algoritmo di guida offline. Si potrebbe eventualmente pensare di integrare l'algoritmo di guida nel simulatore completo, in modo tale da calcolare i riferimenti da dare in input all'autopilota in tempo reale con la simulazione.

Per permettere al drone di seguire la traiettoria desiderata, l'errore tra il valore di riferimento (comandato) e il valore effettivo (misurato tramite un sensore nel contesto pratico) per ogni variabile controllata viene elaborato dal controllore PID utilizzando una combinazione di tre azioni:

- **Proporzionale:** agisce istantaneamente moltiplicando l'errore per un guadagno K_p , quindi maggiore è l'errore maggiore è il contributo fornito da questa azione. Bisogna scegliere appositamente questo guadagno perché, se troppo elevato, potrebbe portare ad instabilità del sistema.
- **Integrativa:** tiene conto dell'integrale dell'errore nel tempo e quindi della sua evoluzione. Questo contributo risulta essere necessario per azzerare l'errore stazionario. Anche esso, come il proporzionale, se non opportunamente trattato, potrebbe portare alla comparsa di instabilità e potrebbe agire male in caso di saturazioni. Ecco perché una logica anti-wind up potrebbe essere necessaria.
- **Derivativa:** tiene conto della derivata dell'errore nel tempo, consentendo quindi di avere una previsione del suo andamento. Idealmente questa azione dovrebbe fornire un contributo che smorza le oscillazioni causate dall'integratore. Nel caso di un'implementazione pratica con sensori, è necessario integrare un filtro passa-basso altrimenti si rischia di amplificare dei rumori ad alta frequenza prodotti dai sensori, che potrebbero portare poi ad una conseguente instabilità piuttosto che ad uno smorzamento.

Il segnale di controllo impartito è quindi rappresentabile, nel dominio di Laplace usando dei guadagni K , nel seguente modo:

$$U(s) = P(s) + I(s) + D(s) = \left(K_p + \frac{K_i}{s} + K_d \cdot \frac{N}{1 + \frac{N}{s}}\right) \cdot E(s) \quad (1)$$

dove $E(s)$ rappresenta l'errore, $N = \frac{1}{\tau}$ è il coefficiente di filtraggio per il contributo derivativo, e τ è una costante di tempo. L'autopilota integrato nel simulatore agisce per mezzo di cinque diversi controllori PID, come mostrato in Figura 3. I controllori agiscono sia sulla dinamica longitudinale che latero-direzionale del drone. Infine, con lo scopo di rendere la simulazione più realistica, sono stati presi in considerazione i limiti fisici che vincolano i movimenti delle superfici mobili e della manetta. Per farlo, sono stati introdotti dei saturatori nelle equazioni per limitare sia gli angoli e

4.2 Il modello matematico

Sebbene l'autopilota esposto prima sia stato progettato ragionando su dinamica longitudinale e dinamica latero-direzionale, la simulazione del moto del drone si realizza integrando le equazioni complete.

È necessario scegliere dei sistemi di riferimento rispetto ai quali scrivere le suddette equazioni, in particolare:

- *Sistema di riferimento NED*: l'origine è coincidente con il baricentro del drone e gli assi sono orientati sempre secondo le direzioni Nord, Est (del riferimento terrestre) e verso il centro della terra. Questo sistema di riferimento è utilizzato per disegnare la traiettoria del drone nello spazio.
- *Sistema di riferimento body*: l'origine è nel centro di massa del drone. L'asse x_B è contenuto nel piano di simmetria del velivolo, diretto verso prua; l'asse z_B è normale a x_B , contenuto nel piano di simmetria e orientato verso il ventre del velivolo; l'asse y_B completa la terna del riferimento destrorsa. Questo sistema di riferimento è utilizzato per descrivere la dinamica del drone.

Successivamente si sceglie il modello dell'atmosfera, tramite il quale si possono calcolare la temperatura, la pressione, la densità dell'aria e la velocità del suono. Il modello scelto è quello dell'Atmosfera Standard Internazionale (ISA). Si considera il drone come un corpo rigido di massa costante e simmetrico rispetto al piano longitudinale $x - z$. Infine si definiscono le equazioni differenziali della dinamica e di navigazione che fanno parte del sistema che verrà integrato numericamente.

Equazioni di equilibrio alla traslazione:

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

Equazioni di equilibrio alla rotazione:

$$\begin{cases} \dot{p} = -\frac{J_z - J_y}{J_x} qr + (pq + \dot{r}) \frac{J_{xz}}{J_x} + \frac{L}{J_x} \\ \dot{q} = -\frac{J_x - J_z}{J_y} pr - (p^2 - r^2) \frac{J_{yz}}{J_y} + \frac{M}{J_y} \\ \dot{r} = -\frac{J_y - J_x}{J_z} pq - (qr - \dot{p}) \frac{J_{xz}}{J_z} + \frac{N}{J_z} \end{cases}$$

Equazioni della cinematica:

$$\begin{cases} \dot{\phi} = p + q \sin \phi \tan \theta + r \cos \phi \tan \theta \\ \dot{\theta} = q \cos \phi - r \sin \phi \\ \dot{\psi} = q \frac{\sin \phi}{\cos \theta} + r \frac{\cos \phi}{\cos \theta} \end{cases}$$

Equazioni di navigazione:

$$\begin{cases} \dot{p}_N = u \theta c \psi + v(-c \phi s \psi + s \phi s \theta c \psi) + w(s \phi s \psi + c \phi s \theta c \psi) \\ \dot{p}_E = u \theta s \psi + v(c \phi c \psi + s \phi s \theta s \psi) + w(-s \phi c \psi + c \phi s \theta s \psi) \\ \dot{h} = -u s \theta + v s \phi c \theta + w c \phi c \theta \end{cases}$$

dove $c \equiv \cos()$ e $s \equiv \sin()$.

Le forze e i momenti indicati nelle equazioni esposte sopra sono ricavati tramite l'utilizzo dei coefficienti aerodinamici. Questi vengono estrapolati dai database costruiti su determinate quote (definiti precedentemente come *dba*), per cui è necessario fare l'interpolazione in caso di quote intermedie agli intervalli per i quali sono definiti i database. Infine, un'ulteriore interpolazione in base all'angolo di incidenza α è poi necessaria per calcolare i coefficienti aerodinamici che vengono poi inseriti nelle seguenti equazioni di forze e momenti.

Forze aerodinamiche:

$$\begin{cases} X = \frac{1}{2}\rho V^2 S(C_{X_{ss}} + C_{X_\alpha}\alpha + C_{X_\beta}\beta + C_{X_p}\hat{p} + C_{X_q}\hat{q} + C_{X_r}\hat{r} + C_{X_{\delta_a}}\delta_a + C_{X_{\delta_e}}\delta_e) \\ Y = \frac{1}{2}\rho V^2 S(C_{Y_{ss}} + C_{Y_\alpha}\alpha + C_{Y_\beta}\beta + C_{Y_p}\hat{p} + C_{Y_q}\hat{q} + C_{Y_r}\hat{r} + C_{Y_{\delta_a}}\delta_a + C_{Y_{\delta_e}}\delta_e) \\ Z = \frac{1}{2}\rho V^2 S(C_{Z_{ss}} + C_{Z_\alpha}\alpha + C_{Z_\beta}\beta + C_{Z_p}\hat{p} + C_{Z_q}\hat{q} + C_{Z_r}\hat{r} + C_{Z_{\delta_a}}\delta_a + C_{Z_{\delta_e}}\delta_e) \end{cases}$$

Momenti aerodinamici:

$$\begin{cases} L = \frac{1}{2}\rho V^2 S b(C_{l_{ss}} + C_{l_\alpha}\alpha + C_{l_\beta}\beta + C_{l_p}\hat{p} + C_{l_q}\hat{q} + C_{l_r}\hat{r} + C_{l_{\delta_a}}\delta_a + C_{l_{\delta_e}}\delta_e) \\ M = \frac{1}{2}\rho V^2 S c(C_{m_{ss}} + C_{m_\alpha}\alpha + C_{m_\beta}\beta + C_{m_p}\hat{p} + C_{m_q}\hat{q} + C_{m_r}\hat{r} + C_{m_{\delta_a}}\delta_a + C_{m_{\delta_e}}\delta_e) \\ N = \frac{1}{2}\rho V^2 S b(C_{n_{ss}} + C_{n_\alpha}\alpha + C_{n_\beta}\beta + C_{n_p}\hat{p} + C_{n_q}\hat{q} + C_{n_r}\hat{r} + C_{n_{\delta_a}}\delta_a + C_{n_{\delta_e}}\delta_e) \end{cases}$$

Per quanto concerne il modello propulsivo viene sfruttata la teoria dell'elemento di pala di Glauert. Questa è una teoria semplificata utilizzata per calcolare la spinta e la coppia generata da un'elica in base ai giri al minuto (RPM) dell'elica stessa. In particolare si suddivide l'elica in tante stazioni e per ognuna di queste vengono calcolate la coppia e la spinta generata, per poi essere sommate tra loro. La potenza all'albero richiesta per generare la spinta non deve mai superare il valore massimo $P_{max}=160$ W. Per l'elica in questione tutte le stazioni sono contribuenti in termini di spinta e coppia dato che non si ha l'ogiva.

In questo simulatore, come metodo di propagazione numerica delle equazioni, è stato sfruttato il metodo di Eulero esplicito che integra ogni stato del sistema nel seguente modo:

$$x_{n+1} = x_n + \Delta t \cdot f(t_n, x_n) \quad (2)$$

dove la funzione $f(t_n, x_n)$ è il residuo calcolato con le equazioni scritte sopra valutate per t_n ed x_n , e Δt è il passo di integrazione. Questo, al fine di evitare instabilità numeriche, deve essere scelto in modo che sia più piccolo della costante di tempo minore del sistema (in questo caso quella del corto periodo), così da catturare adeguatamente la dinamica del sistema. Non deve essere scelto però troppo piccolo altrimenti si rischia di aumentare notevolmente il tempo di simulazione. In questo caso studio è stato scelto un $\Delta t = 0.01$ s che si è dimostrato essere un buon compromesso. Inoltre, la scelta di questo passo di integrazione è conveniente anche perché la time history di ψ_{ref} è fornita dall'algoritmo di guida ad intervalli di 0.01 s. Per ogni passo di integrazione sono calcolati le forze e i momenti aerodinamici, la manetta, le varie deflessioni delle superfici mobili, l'angolo di incidenza, l'angolo di sideslip e le varie velocità.

Non è necessario effettuare un controllo sulla massa del velivolo perché questa si mantiene costante dato che si usano delle batterie ed un motore elettrico.

Infine, per la valutazione della stabilità statica e dinamica del velivolo, si utilizza il criterio di Routh. Dopo aver calcolato i coefficienti aerodinamici di interesse, si possono ricavare con un metodo approssimato le grandezze relative ai modi longitudinali del velivolo, in particolare di *corto periodo* e *fugoide*.

5 Simulazione

5.1 Pre-processing

La prima fase consiste nella scelta del tipo di simulazione che si vuole eseguire, che può essere la verifica del mantenimento del trim o la simulazione di un percorso prestabilito (deciso successivamente).

Se si sceglie di simulare il mantenimento del trim, viene richiesto l'inserimento delle condizioni iniziali da tastiera. Viene poi verificato che i valori inseriti siano plausibili, nei limiti imposti dalla simulazione. Di seguito vengono riportati le variabili di input e i rispettivi limiti:

- **Quota (h):** $0 < h \leq 2000m$
- **Velocità (V):** $V_{min} \leq V \leq 25 \text{ m/s}$, dove $V_{min} = \sqrt{\frac{2 \cdot W/S}{\rho \cdot C_{L_{max}}}}$
- **Angolo di rampa (γ):** $-20^\circ < \gamma < 20^\circ$

Se la velocità inserita eccede il valore massimo, sarà impostata automaticamente a 25 m/s. Se invece è minore del valore minimo, verrà impostata alla velocità minima per la quota stabilita che verrà calcolata dopo scelta del modello atmosferico e della lettura dei database aerodinamici (varianti con la quota). A tal proposito, subito dopo l'inserimento della velocità, comparirà a video una nota che spiega quanto appena detto. Se la quota è minore o uguale a zero, oppure superiore alla quota di tangenza (2000 m), la simulazione si arresta. Infine, se viene impostato un angolo di rampa γ non compreso nell'intervallo $(-20^\circ, 20^\circ)$, viene richiesto di reinserire correttamente il suo valore.

Se invece si sceglie di seguire un percorso prestabilito, le condizioni iniziali vengono automaticamente impostate ai seguenti valori di default:

- $V = 13.5 \text{ m/s}$
- $h = 100 \text{ m}$
- $\gamma = 0^\circ$

Questo viene fatto perché i file di testo che contengono i riferimenti vengono generati dall'algoritmo di guida usando le condizioni appena definite.

Successivamente viene chiesto di definire il modello di atmosfera che si vuole utilizzare. All'utente verranno proposte le seguenti opzioni:

1. Utilizzare il modello standard dell'atmosfera (ISA), impostando i valori standard per i dati atmosferici di riferimento al *sea level*;
2. Modificare i dati atmosferici di riferimento in modo da ricavare i dati alla quota di volo utilizzando le equazioni del modello standard;
3. Scegliere direttamente i dati atmosferici alla quota di volo prestabilita.

```

SIMULATORE DI VOLO:

Selezionare una tra le seguenti scelte:
[1] Simulazione del mantenimento del trim.
[2] Simulazione di un percorso predefinito.
1
-----
CONDIZIONI INIZIALI:

Inserisci la quota alla quale stai volando in [m]: 1100
Inserisci il valore della velocita' in [m/s]: 14.6

NOTA
La verifica sul valore minimo della velocita' verra' effettuata a seguito della lettura
dei database aerodinamici.
Nel caso in cui la velocita' sia inferiore alla velocita' di stallo, il suo valore sara'
impostato alla velocita' minima per la quota scelta

Inserire l'angolo di rampa in [deg]: 0

```

(a) Simulazione mantenimento trim

```

SIMULATORE DI VOLO:

Selezionare una tra le seguenti scelte:
[1] Simulazione del mantenimento del trim.
[2] Simulazione di un percorso predefinito.
2
-----
CONDIZIONI INIZIALI:

Le condizioni iniziali sono state automaticamente settate a:
V = 13.5 [m/s]
h = 100 [m]
gamma = 0 [deg]

```

(b) Simulazione percorso

Figura 5: Schermata iniziale

Per maggior chiarezza è stato preferito mostrare nelle immagini successive i dati ottenuti supponendo di portare avanti la simulazione di mantenimento del trim. Le schermate sarebbero esattamente le stesse nel caso di simulazione di un percorso, ma con dati diversi.

```

ATMOSFERA ISA:

La simulazione fa riferimento al modello atmosferico ISA, avente i seguenti valori per quota
h=0m (sea level):

Pressione:          P = 101325 Pa
Temperatura:        T = 15 C
Densita':           rho = 1.225 kg/m^3
Velocita' del Suono: a = 340 m/s

Se non si desidera procedere con i suddetti parametri e' possibile modificarli, reinserendoli
manualmente o scegliendo una quota differente.

Premere:
[1] se si desidera procedere con i valori iniziali precedentemente indicati
[2] se si desidera modificare i valori iniziali a h=0
[3] se si desidera immettere i valori manualmente a una quota specifica (richiesta in
seguito)

```

Figura 6: Schermata scelta modello atmosfera

5.2 Processing

Una volta fatta l'interpolazione (se necessaria) dei file dba, si inizia la fase di processing con il calcolo dell'angolo di **incidenza di trim** (α_{trim}), considerando il velivolo in volo longitudinale alle condizioni definite nella fase di pre-processing. Per effettuare il calcolo viene utilizzata una procedura approssimata e viene valutato il residuo dell'equazione di equilibrio lungo l'asse z. Si procede iterativamente, partendo dall'angolo di primo tentativo $\alpha_{trim} = \alpha_{min} = -5^\circ$ e incrementandolo di 0.01° ad ogni ciclo. I coefficienti aerodinamici vengono opportunamente interpolati in

funzione dell'angolo corrente ed il coefficiente complessivo $C_{Z_{tot}}$ viene calcolato, tenendo anche in considerazione il contributo dato dall'angolo dell'equilibratore. La condizione di arresto e di uscita dal ciclo è la seguente:

$$\left| mg \cos(\alpha + \gamma) + \frac{1}{2} C_{Z_{tot}} \rho S V^2 \right| \leq res \quad (3)$$

dove res rappresenta un residuo che è imposto pari a 0.001 N. Se questa condizione non viene verificata, si incrementa il valore di primo tentativo e si ricomincia il ciclo. La deflessione dell'equilibratore viene calcolata a posteriori dato l'angolo di incidenza ad ogni step, tramite l'equazione di equilibrio al momento attorno all'asse y del velivolo.

$$\delta_{e_{trim}} = - \frac{(C_{M_{ss}} + C_{M_{\alpha}}(\alpha_{trim}))}{(C_{M_{\delta_e}})}; \quad (4)$$

Potrebbe succedere che non esista una coppia di angoli α e δ_e nel range di valori ammissibili ($-5^\circ \leq \alpha \leq 16.8^\circ$, $-20^\circ \leq \delta_e \leq 20^\circ$) e quindi per le condizioni iniziali impostate il trim non è valutabile. Se si ricade in questa casistica, la simulazione riparte dalla fase di pre-processing in cui viene richiesto di inserire un nuovo set di dati iniziali. Inoltre, nel caso in cui dovesse valere la condizione $|\delta_e^{TRIM}| > |\delta_{e_{max}}|$, essendo che il drone sfrutta le stesse superfici mobili anche con funzione da alettoni, non si imposta $|\delta_e^{TRIM}| = |\delta_{e_{max}}|$, ma si richiede di ripartire con l'assegnazione delle condizioni iniziali perché altrimenti la superficie mobile non avrebbe spazio per manovrare.

A seguito del calcolo di α_{trim} e in funzione di questo angolo, tutti i coefficienti aerodinamici vengono interpolati linearmente e stampati su un file di testo. In questo modo, i coefficienti interpolati possono essere usati negli step successivi di simulazione, sfruttando una funzione che crea una matrice di coefficienti all'occorrenza.

Nello step successivo vengono calcolati e visualizzati a schermo il **vettore di stato** e il **vettore di controllo** di trim. Lo stato è composto da 12 variabili: 3 per le velocità lineari (u, v, w), 3 per l'assetto (ϕ, θ, ψ), 3 per le velocità angolari (p, q, r) e 3 per la posizione (x, y, h). Dato che la condizione di volo iniziale è quella di volo longitudinale rettilineo, alcune di queste variabili sono automaticamente impostate a un valore nullo:

$$\begin{aligned} \phi, \psi, v, p, q, r &= 0 \\ h = h_0, \theta &= \alpha_{trim} + \gamma, u = V \cdot \cos(\alpha_{trim}), w = V \cdot \sin(\alpha_{trim}); \end{aligned} \quad (5)$$

Per quanto riguarda invece x e y si possono avere diverse inizializzazioni a seconda che si stia simulando il mantenimento del trim o uno specifico percorso.

Oltre al vettore di stato ed al vettore di controllo vengono calcolati e visualizzati a video **RPM** e **spinta** di trim. Per fare ciò si calcola la spinta che il motore dovrebbe esercitare per vincere un contributo del peso e la resistenza aerodinamica.

$$T_{trim} = mg \sin(\theta_{trim}) - \frac{1}{2} \rho V^2 C_{X_{tot}} S; \quad (6)$$

dove $C_{X_{tot}}$ è funzione di α_{trim} e $\delta_{e_{trim}}$ calcolati precedentemente. Per trovare gli RPM necessari a fornire la spinta di trim, si utilizza un processo iterativo. Partendo

quindi dal valore minimo di RPM come primo tentativo, si procede iterativamente incrementando di 100 RPM ad ogni ciclo, fino ad ottenere una spinta T (funzione degli RPM) uguale a quella di trim, a meno di un residuo fissato $res = 0.01N$.

$$|T - T_{trim}| \leq res \quad (7)$$

Questa è la condiziona di uscita dal ciclo iterativo. Se l'equazione non è soddisfatta per alcun valore di RPM nel range dei valori ammissibili ($3600 \leq RPM \leq 30000$), allora l'algoritmo imposta gli RPM di trim ad un valore medio pari a $\frac{1}{2} \cdot (RPM_{max} + RPM_{min})$. Una volta calcolati spinta ed RPM di trim, essi vengono stampati a video insieme a tutte le altre grandezze di trim.

Si calcola inoltre la potenza trasmessa all'albero durante il trim, mediante il prodotto della coppia dell'elica per la velocità angolare ω del motore, che deve rimanere inferiore a 160 W. Nel caso in cui la potenza calcolata ecceda tale valore massimo, e quindi $P = C \cdot \omega > 160$ W, si impostano gli RPM ad un valore medio.

Infine, il vettore di controllo di trim comprende la deflessione dell'elevone con funzione di equilibratore (δ_e), la deflessione dell'elevone con funzione di alettone (δ_a) e la manetta (δ_{th}), che in questa trattazione è stata considerata linearmente dipendente dagli RPM del motore, calcolata tramite un'interpolazione considerando $\delta_{th_{max}} = 1$ associato a $RPM_{max} = 30000$ e $\delta_{th_{min}} = 0.1$ associato a $RPM_{min} = 3600$.

$$\delta_e = \delta_{e_{trim}}, \delta_a = 0, \delta_{th_{trim}} = \delta_{th_{min}} + \frac{(\delta_{th_{max}} - \delta_{th_{min}})}{RPM_{max} - RPM_{min}}(RPM_{trim} - RPM_{min}) \quad (8)$$

```

I dati atmosferici e di potenza per la quota di 1100.000000 m sono:
Temperatura:      281.000000 [K]
Pressione:        88789.263036 [Pa]
Densita':         1.100770 [kg/m^3]
Potenza:          0.160000 [kW]
-----
CONDIZIONI DI TRIM:
alpha_trim:      3.014 [deg]      (0.053 [rad])
delta_e_trim:    -2.644 [deg]     (-0.046 [rad])

Il vettore di stato di trim e':
u_trim: 14.579804 [m/s]
v_trim: 0.000000 [m/s]
w_trim: 0.767668 [m/s]
p_trim: 0.000000 [rad/s]
q_trim: 0.000000 [rad/s]
r_trim: 0.000000 [rad/s]
phi_trim: 0.000000 [deg]
theta_trim: 3.014000 [deg]
psi_trim: 0.000000 [deg]
h_trim: 1100.000000 [m]
x_trim: 0.000000 [m]
y_trim: 0.000000 [m]

Spinta di trim: 0.419060 [N]
RPM di trim: 8000.000000

Il vettore di controllo di trim e':
De_trim: -2.644212 [deg]
Da_trim: 0.000000 [deg]
Dth_trim: 0.250000 [percentuale]

```

Figura 7: Schermata grandezze di trim

A questo punto, supponendo che la condizione trim sia stata trovata, si prosegue nella simulazione con uno studio della stabilità statica e dinamica del velivolo. Per tale scopo viene utilizzato il **criterio di Routh** una volta calcolati tutti i coefficienti

aerodinamici necessari. Le condizioni di stabilità vengono mostrate a video in modo da informare l'utente e, in caso di velivolo instabile, si esce dalla simulazione. Se il criterio è soddisfatto, si procede con il calcolo delle caratteristiche principali dei **modi longitudinali** (corto periodo e fugaide), che comprendono in particolare: pulsazione, smorzamento, periodo e tempo di dimezzamento.

A questo punto della simulazione, a seconda che sia stato scelto di simulare il mantenimento del trim o un percorso, si presenteranno due scenari differenti. Nel primo caso verrà chiesto di inserire il tempo di simulazione, mentre nel secondo caso verrà chiesto di scegliere uno tra i percorsi proposti. Potrebbero verificarsi alcune condizioni che portano la simulazione a non andare a buon fine:

- La quota scende a valori inferiori o uguali a zero metri, il che vuol dire che il drone si è schiantato al suolo;
- La quota supera quella di tangenza che è la massima ammissibile;
- La potenza supera quella massima, pari a 160 W, durante l'integrazione;
- L'autonomia del drone è terminata e la batteria è scarica.

```
STABILITA':
Il velivolo e' staticamente stabile.
Il velivolo e' dinamicamente stabile.
-----
MODI LONGITUDINALI:
MODO DI FUGAIDE:
Pulsazione: 0.950235 [rad/s]
Smorzamento: 0.040895
Periodo: 6.617778 [s]
Tempo di dimezzamento: 17.835996 [s]
MODO DI CORTO PERIODO:
Pulsazione: 15.813587 [rad/s]
Smorzamento: 0.444460
Periodo: 0.443546 [s]
Tempo di dimezzamento: 0.098613 [s]
```

Figura 8: Schermata stabilità e modi longitudinali

Possono presentarsi a video alcuni warning che avvisano l'utente in caso di saturazioni oppure in caso il drone raggiungesse i limiti di velocità. In quest'ultimo caso la velocità verrà impostata a quella limite e la simulazione continua. Sarebbe più corretto uscire dalla simulazione, ma è stato scelto di continuare per consentire all'utente di poter testare anche percorsi che richiedono manovre più aggressive. La fase di processing quindi si concluderà una volta finito di integrare le equazioni con un messaggio a video indicante lo stato di batteria del drone e la corretta esecuzione della simulazione. Nei prossimi capitoli (6-7) verranno mostrati i risultati che si ottengono nella fase di post-processing in Matlab.

5.3 Messaggi di errore e warning

Gli errori e i warning sono definiti con lo scopo di informare l'utente di situazioni particolari che si potrebbero verificare durante la simulazione. In base alla loro gravità si potrebbe richiedere una terminazione della simulazione oppure una ridefinizione dei parametri. In alcuni casi, l'utente può intervenire manualmente, mentre in altri casi l'intervento può avvenire in modo automatico. Di seguito è riportata una lista dei messaggi organizzati in base alla loro posizione nelle diverse funzioni.

ManageFile.c

- [!]ERRORE nella lettura del file engine.txt - Il file engine.txt non è stato aperto correttamente o non è stato trovato nella directory impostata.
- [!]ERRORE nella lettura del file battery.txt - Il file battery.txt non è stato aperto correttamente o non è stato trovato nella directory impostata.
- [!]ERRORE nella lettura del file propeller.txt - Il file propeller.txt non è stato aperto correttamente o non è stato trovato nella directory impostata.
- [!]ERRORE nella lettura del file validazione engine.txt - Il file di validazione associato a engine.txt non è stato creato correttamente.
- [!]ERRORE nella lettura del file validazione battery.txt - Il file di validazione associato a battery.txt non è stato creato correttamente.
- [!]ERRORE nella lettura del file validazione propeller.txt - Il file di validazione associato a propeller.txt non è stato creato correttamente.
- [!]ERRORE nell'apertura del file dba relativo alla quota[i] - Il file dba_quota[i].txt non è stato aperto correttamente o non è stato trovato nella directory impostata.
- [!]ERRORE nella lettura del file validazione dba.txt - Il file di validazione associato al dba_quota[i].txt o all'interpolazione di dba_quota[i].txt e dba_quota[i+1].txt non è stato creato correttamente.

Atmosfera.c

- [!]ERRORE valore di quota minore o uguale a zero - E' stato inserito un valore di quota negativo o pari a zero.
- [!]ERRORE La quota inserita e' superiore a quella di tangenza - E' stata inserita un valore di quota superiore a quella di tangenza (2000 m).
- [!]ERRORE Il Mach di volo e' superiore a quello di drag rise (0.6) - E' stato calcolato un numero di Mach superiore al valore di Drag Rise.
- [!]WARNING immettere un valore positivo - E' stato inserito un valore di pressione, densità e velocità del suono negativi.
- [!]WARNING La quota inserita e' quella di tangenza - E' stato inserito un valore di quota pari a quello di tangenza (2000m).
- [!]WARNING Velocità minore o uguale a zero - E' stato inserito un valore di velocità minore o uguale a zero.
- [!]WARNING La velocità inserita e' al di sopra del limite consentito - E' stato inserito un valore di velocità superiore a quello massimo (25 m/s).
- [!]WARNING Angolo non consentito - E' stato inserito un angolo non compreso tra -20 e 20 gradi.
- [!]WARNING immettere un numero da 1 a 3 - E' stato inserito un valore diverso da 1, 2 o 3. Reimmettere un numero corretto.

Alpha_trim.c

- ERRORE[!] Condizioni di trim non trovate: nessun angolo di incidenza ammissibile soddisfa l'equilibrio longitudinale - L'algoritmo non ha

individuato alcun angolo di incidenza che garantisce l'equilibrio longitudinale tra i valori ammissibili.

- **ERRORE[!]** Deflessione dell'elevone non raggiungibile - L'angolo dell'elevone necessario a garantire l'equilibrio al momento attorno all'asse y_{body} del velivolo è al di fuori del range di valori ammissibili.
- **WARNING[!]** La velocità inserita è inferiore alla velocità di stallo - È stata inserita in input una velocità inferiore a quella di stallo.

Calcolo_spinta_RPM

- **[!]ERRORE** Il valore di RPM necessario per mantenere il trim è inferiore al valore minimo consentito - Il valore di RPM è inferiore a 3600.
- **[!]WARNING** Potenza all'albero maggiore della potenza massima - È stata raggiunta la potenza massima e gli RPM vengono impostati ad un valore medio.
- **[!]WARNING** Il valore di RPM necessario per mantenere il trim è superiore al valore massimo consentito - Il valore di RPM è superiore a 30000 e quindi gli RPM vengono impostati ad un valore medio.

Interpolazione.c

- **[!]ERRORE:** apertura del file dei coefficienti aerodinamici non riuscita - Il codice non è stato in grado di leggere il file di validazione *Validation/Interpolated_coefficients.txt* dei coefficienti aerodinamici.

Integrazione.c

- **[!]ERRORE:** apertura del file di salvataggio delle variabili di stato non riuscita - Il codice non è stato in grado di leggere il file *Integrazione.txt* di salvataggio delle variabili di stato.
- **[!]ERRORE:** apertura del file di salvataggio delle variabili di controllo non riuscita - Il codice non è stato in grado di leggere il file *Control_vector.txt* di salvataggio delle variabili di controllo.
- **[!]ERRORE:** è stata superata la quota di tangenza - È stato calcolato durante l'integrazione un valore di quota superiore a quello di tangenza (2000 m).
- **[!]ERRORE:** È stato raggiunto il suolo - È stato calcolato durante l'integrazione un valore di quota pari a 0 metri o negativo.
- **[!]ERRORE:** apertura del file_PsiReference non riuscita - Il file relativo al percorso non è stato aperto correttamente o non è stato trovato nella directory.
- **[!]ERRORE:** la batteria è scarica - L'energia consumata ha raggiunto il limite massimo.
- **[!]WARNING:** È stata raggiunta la quota di tangenza - È stato calcolato durante l'integrazione un valore di quota pari a quello di tangenza. La simulazione continua.
- **[!]WARNING:** la deflessione di elevone dedicata all'alettone satura durante la manovra - Saturazione della deflessione relativa all'alettone.

- [!]WARNING: la deflessione di elevone dedicata all'equilibratore satura durante la manovra - Saturazione della deflessione dovuta all'equilibratore
- [!]WARNING: la manetta satura durante la manovra - Saturazione della manetta.
- [!]WARNING: la velocità è minore di quello di stallo e quindi è stata settata pari alla V_{min} - E' stata calcolata durante l'integrazione una velocità inferiore a quella di stallo e di conseguenza viene posta pari a quella di stallo.
- [!]WARNING: la velocità supera quella massima e quindi è stata settata pari alla V_{max} - E' stata calcolata durante l'integrazione una velocità maggiore della massima ammissibile e di conseguenza è stata settata pari al valore massimo.
- [!]WARNING: inserire tempo di integrazione positivo - E' stato inserito un tempo di integrazione negativo, viene richiesto l'inserimento di un tempo di integrazione positivo.

Routh.c

- [!]ERRORE: Il drone e' staticamente instabile - Il drone non rispetta le condizioni di stabilità statica.
- [!]ERRORE: Il drone e' dinamicamente instabile - Il drone non rispetta le condizioni di stabilità dinamica.

6 Mantenimento dello stato di trim

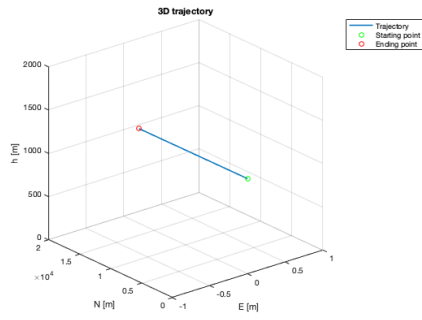
Nella seguente sezione viene mostrato un esempio sui risultati che riguardano il mantenimento dello stato di trim per il set di condizioni iniziali che sono state impostate nel capitolo precedente:

- Quota: $h = 1100$ m
- Velocità: $V = 14.6$ m/s
- Angolo di rampa: $\gamma = 0$ deg

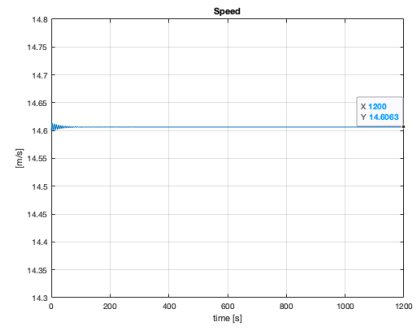
La simulazione viene eseguita per un tempo $T = 1200$ secondi.

Commento:

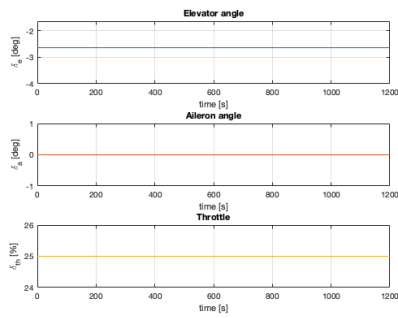
Per simulare il mantenimento del trim il controllore viene escluso e si può osservare dai grafici riportati in basso che, dopo un transitorio di alcuni secondi in cui si manifestano i moti oscillatori, tutte le grandezze si assestano ai loro valori di equilibrio e si stabilizzano. Le grandezze maggiormente oscillanti sono proprio quelle coinvolte nella dinamica longitudinale del velivolo e mostrano degli andamenti assimilabili alla composizione dei modi longitudinali in termini di ampiezze e frequenze. In particolare queste grandezze sono: u, w, θ, q, α . Inoltre, dal grafico della quota in funzione del tempo si può notare che si ha un calo di circa 7.5 metri in 1200 secondi, imputabile al fatto che si sta usando un modello di integrazione numerica senza la presenza di un controllore in feedback e al fatto che sono state utilizzate delle grandezze di trim ricavate con metodi approssimati. L'andamento dell'angolo di incidenza α non ha un reale senso fisico dato che assume la forma di un'onda quadra. Ciò è dovuto al fatto che i dati vengono salvati in output con dei float fino a 5 cifre decimali e quindi piccole variazioni vengono graficate come dei gradini.



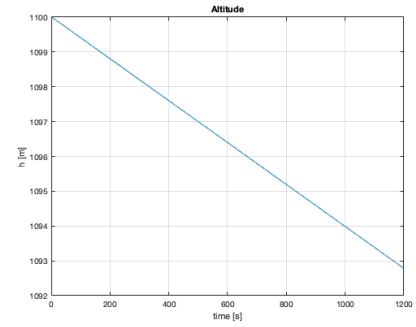
(a) Traiettorie



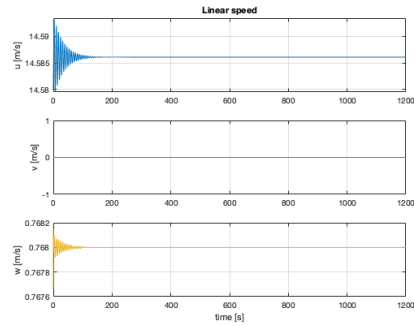
(b) Velocità



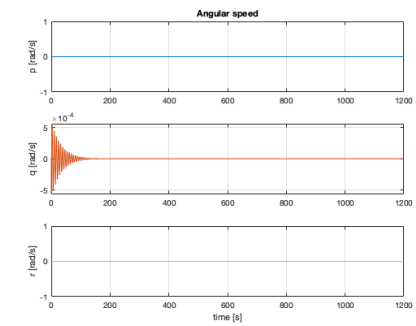
(c) Comandi



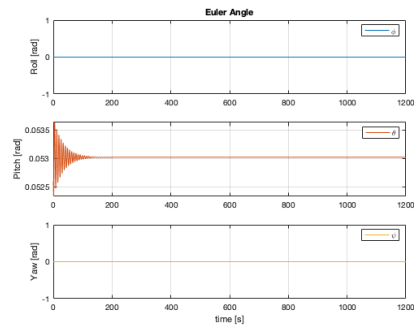
(d) Quota



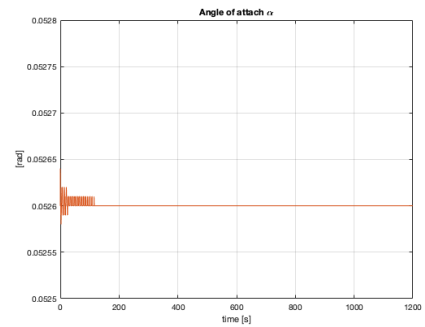
(e) Velocità lineari



(f) Velocità angolari



(g) Angoli di Eulero



(h) Angolo di incidenza

7 Esecuzione di un percorso

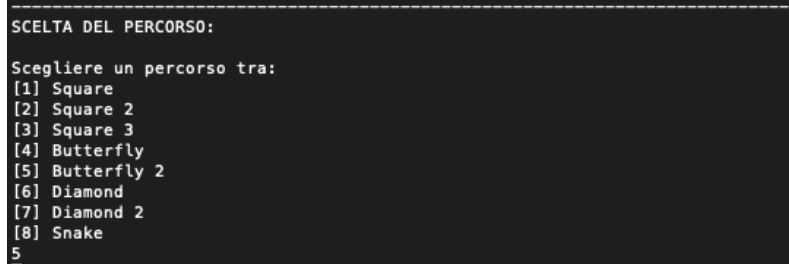
In questa sezione si analizzerà un percorso di esempio e nello specifico quello denominato "Butterfly_2" che è già stato usato come esempio precedentemente. Esso corrisponde al comando [5] da inserire da terminale una volta arrivati alla scelta del percorso come si può vedere dalla Figura 10. L'andamento dell'angolo di heading di riferimento è rappresentato nella Figura 4, insieme alla posizione dei vari waypoints.

Condizioni iniziali:

Le condizioni iniziali utilizzate sono quelle per cui sono stati generati i riferimenti dall'algoritmo di guida, come accennato nei precedenti paragrafi. Per comodità di lettura queste condizioni vengono qui ripetute:

- $h = 100$ m
- $V = 13.5$ m/s
- $\gamma = 0$ deg

Il tempo totale di simulazione sarà 250 secondi.



```

SCELTA DEL PERCORSO:
Scegliere un percorso tra:
[1] Square
[2] Square 2
[3] Square 3
[4] Butterfly
[5] Butterfly 2
[6] Diamond
[7] Diamond 2
[8] Snake
5

```

Figura 10: Schermata di scelta del percorso

Logica sull'errore dell'angolo ψ :

L'angolo di heading sembrerebbe non seguire bene il riferimento al primo e all'ultimo step. Questo accade perché è stata implementata una logica sull'errore tra heading comandato (ψ_{ref}) e heading effettivo (ψ) del seguente tipo:

- Se l'errore $\psi_{err} = \psi_{ref} - \psi$ è minore di -180° allora si aggiungono 360° a ψ_{err} ;
- Se l'errore $\psi_{err} = \psi_{ref} - \psi$ è maggiore di 180° allora si sottraggono 360° a ψ_{err} ;
- Se nessuna delle due condizioni si verifica allora si procede senza fare modifiche a ψ_{err} ;

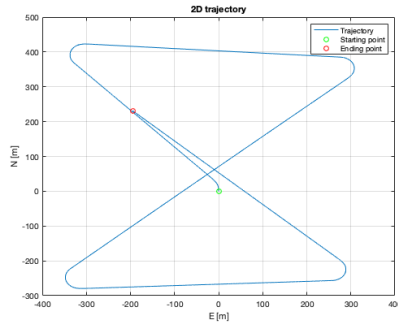
Questa logica considera sempre l'errore più piccolo rispetto al riferimento che si vuole raggiungere e risulta essere necessaria in caso di angoli comandati molto grandi. Essa viene applicata in tutti i percorsi disponibili nel simulatore. Per fornire un maggiore chiarimento si può osservare che al primo step imposto di ψ_{ref} , si comandano 5.53 rad che equivalgono a 317° ; se quindi si parte da zero gradi come riferimento, allora l'errore risulta essere proprio 317° . L'angolo che invece viene seguito con la logica sopra esposta è -0.75 rad che equivale a -43° . Essendo che una circonferenza equivale a 360° , allora comandare -43° o comandare 317° è analogo dal punto di vista fisico del drone. Usando l'errore più piccolo migliora il funzionamento dell'autopilota perché ci si discosta meno dal punto per cui i controllori sono stati progettati.

Commento:

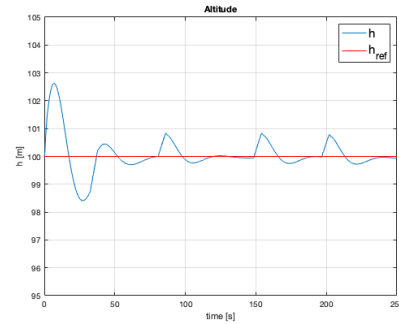
Come si può notare dalla figura della traiettoria, non viene esattamente seguito il percorso e si manifestano delle oscillazioni più accentuate all'inizio della simulazione sia per la velocità che per la quota. Questo potrebbe essere dovuto al fatto che il simulatore e l'algoritmo di guida non vengono eseguiti contemporaneamente. In aggiunta, il riferimento tra waypoint 0 e 1 è stato generato supponendo di avere un decollo che in questo caso non avviene perché si parte già in quota alle condizioni di trim. Pertanto si può concludere che la distanza percorsa nel primo segmento in assi North, East risulta essere maggiore di quella che ci si aspetterebbe se ci fosse un decollo. Il percorso effettuato risulta quindi essere traslato.

Si può però concludere che l'autopilota funziona correttamente dato che i riferimenti imposti vengono seguiti bene dal velivolo. Osservando gli angoli di Eulero si nota che nel caso del *roll angle* si hanno dei picchi positivi e negativi rappresentanti le manovre di virata che il drone esegue ogni volta che raggiunge un waypoint per poi puntare verso quello successivo. Quanto appena detto può essere anche verificato osservando l'andamento nel tempo del contributo alettone (*aileron angle*) che manifesta dei picchi per ogni virata concordi a quelli del *roll angle*.

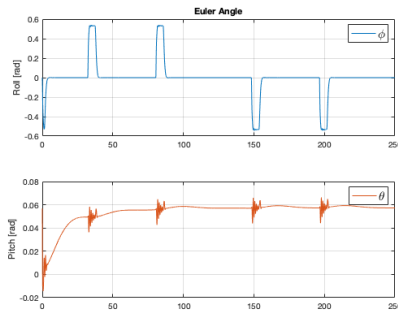
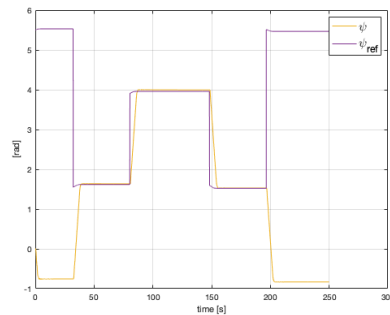
Infine, confrontando le figure (12c) e (12d) si può notare una somiglianza negli andamenti delle seguenti grandezze: $V - u$, $\alpha - w$ e $\beta - v$.

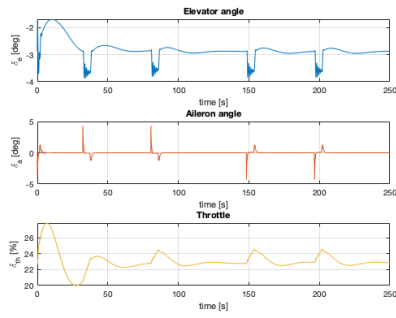


(a) Traiettoria

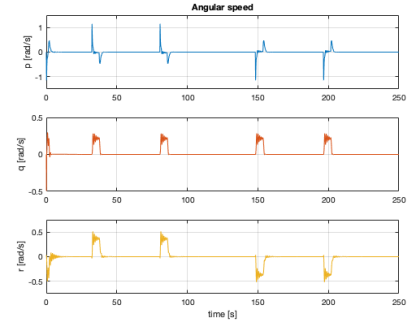


(b) Quota

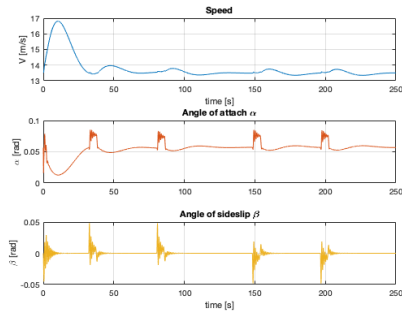

(c) Angoli di Eulero: ϕ, θ

(d) Angolo di heading: ψ



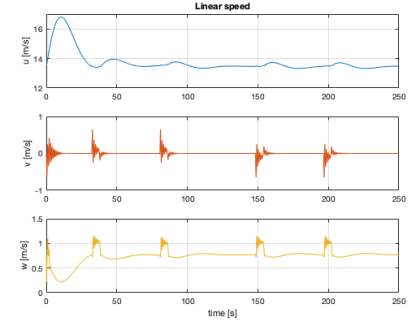
(a) Output comandi autopilota



(b) Velocità angolari



(c) Velocità, incidenza e sideslip



(d) Velocità lineari

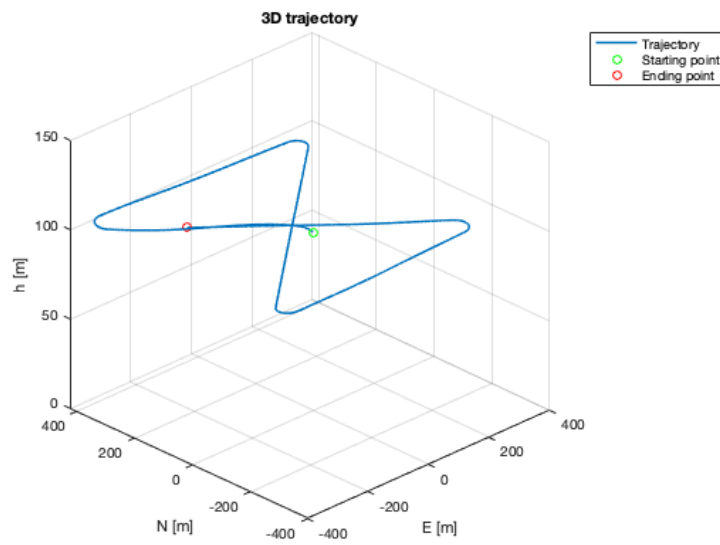


Figura 13: Traiettorie 3D